THESIS ON INFORMATICS AND SYSTEM ENGINEERING C100

Non-Parametric Bayesian Models for Computational Morphology

KAIRIT SIRTS

TUT PRESS

Faculty of Information Technology
Department of Informatics
TALLINN UNIVERSITY OF TECHNOLOGY

This dissertation was accepted for the defence of the degree of Doctor of Philosophy in Informatics on May 4, 2015.

Supervisors: Sharon Goldwater, Ph.D., University of Edinburgh
Prof. Emer. Leo Võhandu, Tallinn University of Technology

Opponents: Mikko Kurimo, Ph.D., Aalto University
Kristina Toutanova, Ph.D., Microsoft Research

Defence: June 18, 2015

Declaration:
Hereby I declare that this doctoral thesis, my original investigation and achievement, submitted for the doctoral degree at Tallinn University of Technology has not been submitted for doctoral or equivalent academic degree.

/ Kairit Sirts /

ISSN 1406-4731
ISBN 978-9949-23-776-0 (publication)
ISBN 978-9949-23-777-7 (PDF)

INFORMAATIKA JA SÜSTEEMITEHNIKA C100

Mitteparameetrilised Bayesi mudelid arvutusliku morfoloogia jaoks

KAIRIT SIRTS

TTÜ
KIRJASTUS

Contents

List of publications

This dissertation is based on the following original publications.

I. Sirts, K. and Alumäe, T. (2012). A hierarchical Dirichlet process model for joint part-of-speech and morphology induction. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 407–416.

II. Sirts, K. and Goldwater, S. (2013). Minimally-supervised morphological segmentation using adaptor grammars. *Transactions of the Association for Computational Linguistics*, 1(May):231–242.

III. Sirts, K., Eisenstein, J., Elsner, M., Goldwater, S. (2014). POS induction with distributional and morphological information using a distance-dependent Chinese restaurant process. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 265–271.

Author's contribution to the publications

I. The author's contribution was everything except the original idea—model formulation, implementation, experimentation and paper writing.

II. The author's contribution was the idea of using model selection for weakly-supervised learning, implementation of all described methods, performing all experiments, and writing the paper together with the co-author.

III. The author's contribution was implementing the model and conducting all experiments, model formulation and paper writing was done in collaboration with the co-authors.

Abstract

Morphology is a complex phenomenon that encompasses several different computational tasks, such as morphological segmentation or analysis and (morpho)syntactic clustering. We hypothesize that in an unsupervised or weakly-supervised learning setting, using joint learning models that address several aspects of the complex morphological processes simultaneously will be beneficial because during joint learning, different aspects of the same process will help to disambiguate each other.

In this dissertation, we develop three unsupervised or weakly-supervised models of computational morphology that employ joint learning in different ways. We adopt non-parametric Bayesian modeling, which provides a flexible framework for learning with both observed and latent variables and additionally, provides suitable prior distributions for linguistic data. In addition to empirically demonstrating the performance of our models, we seek to show that 1) joint modeling provides benefits over non-joint modeling and 2) modeling some latent aspects of the process (in addition to those that we are directly interested in) provides further advantages.

The first model deals with the joint unsupervised learning of part-of-speech (POS) tags and morphological segmentations. The goal was to define a model where both tasks influence and constrain the search space of each other. The empirical results are mixed: the POS induction part performs well, producing state-of-the-art results in comparison with the other unsupervised POS induction models. However, the segmentation results are rather mediocre, suggesting that the tagging assignments do not influence the segmentation decisions as well as expected, referring that the segmentation component has room for improvement.

The second set of models focuses solely on the task of morphological segmentation, in both unsupervised and weakly-supervised setting. We define the models using the Adaptor Grammar framework (Johnson et al., 2007), which combines non-parametric Bayesian modeling with probabilistic context-free grammars and can be used to define various morphology generation grammars. The joint learning in those models is performed over morphemes and their substructures, both of which are latent. Although

linguistically, those substructures have no meaning, we show empirically that the grammars employing such substructures perform better than the flat grammar generating the morpheme sequences only. In addition, our best hierarchical grammars perform in the same range with the state-of-the-art morphological segmentation systems across several languages.

The last model is about morphosyntactic clustering, which can be considered as the unsupervised counterpart to morphological analysis. We combine both distributional information (in the form of continuous word embeddings) and orthographic cues (via suffix features) in a non-parametric mixture model using a distance-dependent Chinese restaurant process prior (Blei and Frazier, 2011). The model performs joint learning of morphosyntactic clusters and a log-linear suffix similarity function employed in the prior. We demonstrate state-of-the-art results in English. We also show that in other languages, although the absolute scores are not too good, the model using both distributional and morphological cues is better than the mixture models utilizing distributional information only, especially for low-frequency words.

In general, most of our experimental results support the hypothesis about the benefits of the joint modeling. However, some experimental results (notably obtained with the first and the last model) are not as good as expected. Analyzing these results leads to an understanding that our assumptions about the relationships between morphemic suffixes and syntactic tags may be too simplistic for the morphologically rich languages, motivating future work to capture these relationships more effectively.

Kokkuvõte

Loomuliku keele morfoloogia on kompleksne nähtus, millega on seotud mitmeid erinevaid arvutuslingvistilisi ülesandeid, nagu näiteks morfoloogiline segmenteerimine, morfoloogiline analüüs või morfosüntaktiline klasterdamine. Meie hüpoteesiks on, et morfoloogia modelleerimisel juhendamata või nõrgalt juhendatud automaatsete meetoditega on eelis nendel mudelitel, mis kasutavad ühisõppimise printsiipe, mille käigus süsteem õpib samaaegselt mitut omavahel seotud ülesannet, sest see võimaldab mudeli erinevatel osadel teineteise mitmesusi lahendada ehk ühestada.

Käesolevas doktoritöös esitame kolm juhendamata või nõrgalt juhendatud arvutusliku morfoloogia mudelit, mis on defineeritud mitteparameetrilise Bayesi raamistikus ja kasutavad ühisõppimist erinevatel viisidel. Bayesi modelleerimine võimaldab defineerida statistilisi mudeleid, mis kasutavad nii vaadeldavaid kui ka varjatud juhuslikke suuruseid. Mitteparameetriliste eeltõenäosusjaotuste kasutamine annab lisaks võimaluse modelleerida astmejaotuseid, mis on omased loomuliku keele struktuuridele. Lisaks mudelite soorituse empiirilisele hindamisele on meie eesmärgiks näidata, et 1) ühisõppimist kasutavad mudelid töötavad paremini kui seda mitte kasutavad mudelid ning 2) mõningate latentsete aspektide modelleerimine lisaks sihtmärgiks olevatele struktuuridele parendab mudeli sooritust.

Esimese mudeli sisuks on sõnaliikide (nagu näiteks nimisõna, verb või omadussõna) ning morfoloogiliste segmentatsioonide juhendamata modelleerimine, kus ühisõppimise eesmärgiks on mõlema ülesande otsinguruumi vastastikune piiramine. Sõnaliikide märgendamise empiirilised tulemused on võrreldavad seni avaldatud parimate juhendamata süntaktilise märgendamise mudelite tulemustega. Samas, morfoloogilise segmenteerimise tulemused on keskpärased, mis viitab sellele, et märgendatud sõnaliigid ei mõjutanud segmenteerimise otsuseid nii hästi kui loodetud ning et ühismudelis kasutatud morfoloogia komponendi struktuur on liiga lihtne.

Järgmine mudel keskendub täielikult morfoloogilise segmenteerimise ülesandele nii juhendamata kui ka nõrgalt juhendatud seadistuses. Me kasutame adaptiivsete grammatikate formalismi (Johnson et al., 2007), mis

ühendab mitteparameetrilise Bayesi modelleerimise statistiliste kontekstivabade grammatikatega ja mida saab kasutada erinevate generatiivsete morfoloogiliste grammatikate defineerimiseks. Me eksperimenteerime erinevate grammatikatega, mis lisaks morfeemidele genereerivad ka morfeemide alamstruktuure, millel otsene lingvistiline tähendus puudub. Ühisõppimine toimub nendes mudelites üle kõikide struktuuride, kusjuures nii morfeemid kui ka nende alamstruktuurid on varjatud antud kontekstis. Me näitame empiiriliselt, et latentseid alamstruktuure modelleerivad grammatikad töötavad paremini kui ainult morfeemijadasid genereerivad grammatikad. Meie parimate grammatikate abil saadud tulemused on sarnased teiste parimate morfoloogilise segmenteerimise süsteemide tulemustega üle erinevate keelte.

Viimane mudel tegeleb morfosüntaktilise klasterdamisega, mida võib vaadelda kui juhendamata vastet morfoloogilisele analüüsile. Me kombineerime sõnade jaotusinformatsiooni (kasutades pidevaid sõnavektoreid) ning ortograafilisi suffikstunnuseid mitteparameetrilises Gaussi segumudelis kasutades eeljaotusena kaugustest sõltuvat Hiina restorani protsessi (Blei and Frazier, 2011). Mudel teostab ühisõppimist üle morfosüntaktiliste klastrite ning eeljaotuses kasutatava log-lineaarse suffiksite sarnasusfunktsiooni. Me demonstreerime häid tulemusi inglise keeles. Teiste keelte puhul, mille tulemused absoluutnumbrites pole eriti head, me näitame, et nii jaotus- kui ka morfoloogilist informatsiooni kasutav ühismodel on parem segumudelitest, mis kasutavad klasterdamiseks ainult jaotusinformatsiooni, seda eriti madala sagedusega sõnade puhul.

Kokkuvõtteks, enamus esitatud tulemusi toetavad hüpoteesi ühisõppimise eeliste kohta. Samas, osad tulemused (saadud esimese ja kolmanda mudeliga) ei ole nii head kui eeldatud. Nende tulemuste analüüs viitab, et tavapäraselt tehtavad eeldused morfoloogiliste suffiksite ning sõnaliikide vaheliste seoste osas on morfoloogiliselt keerukate keelte puhul liiga lihtsakoelised, mis motiveerib edasist tööd nende seoste paremaks ja efektiivsemaks modelleerimiseks.

Acknowledgements

My greatest and deepest expression of gratitude goes to my supervisor Sharon Goldwater. I have learned immensely during those three years we worked together. I appreciate the thoroughness with which she thought along with my research problems, helped me write my papers and commented the draft of this dissertation. I feel very lucky to have had an opportunity to work with her.

I thank Tanel Alumäe from the Speech and Phonetics Laboratory in the Institute of Cybernetics at TUT for employing me and giving me the opportunity to focus on research. I am also grateful to my formal supervisor in TUT, professor emeritus Leo Võhandu, for recommending my admission as a doctorate student, although at that time, I believe, neither of us had no clear clue what my dissertation will be about.

I appreciate the time I spent in University of Edinburgh in winter 2011/12 and the people I met there. I'm especially happy about meeting Micha Elsner with whom I had several useful discussions during my visit and many more after that—I consider you my friend.

I am grateful to all the people I met while being a visiting student in Aalto University in March 2010. It was there where I was first learned about the non-parametric Bayesian modeling techniques. I can still remember my excitement of learning that such methods are available.

I must also thank Innar Liiv from TUT for perhaps half unwittingly giving me several constructive impulses over the years. Also, I thank Ottokar Tilk for reading some earlier parts of my dissertation.

Finally, I am deeply grateful to my husband Risto for his love and support throughout the years. Many times he has carried more than his fair share of caring for our household and children, Markus and Joonas, especially during those five months when I was away in Edinburgh. I am also grateful to my children, first of all for simply sharing their life with me. It was due to their birth that I was able to take a break from my daily work in a bank and reconsider what is it that I would like to do in my life.

Lastly, I thank A.P. first for all those hours that have made completing this dissertation much more difficult, but mostly for patience and love.

Abbreviations

1-1	one-to-one mapping
AG	Adaptor Grammar
ASR	automatic speech recognition
CFG	context-free grammar
CRF	conditional random fields
CRP	Chinese restaurant process
ddCRP	distance-dependent Chinese restaurant process
DP	Dirichlet process
FN	false negatives
FP	false positives
GMM	Gaussian mixture model
HMM	hidden Markov model
IGMM	infinite Gaussian mixture model
IR	information retrieval
LM	language model
LDA	latent Dirichlet allocation
LSA	latent semantic analysis
M-1	many-to-one mapping
MAP	maximum a posteriori
MCMC	Markov chain Monte Carlo
MDL	minimum description length
NLG	natural language generation
NLP	natural language processing
NPB	non-parametric Bayesian
NVI	normalized variation of information
PCFG	probabilistic context-free grammar
POS	part-of-speech
PYP	Pitman-Yor process
SBF1	segment border F1-score
SMT	statistical machine translation
TP	true positives
V-m	V-measure
WSJ	Wall Street Journal corpus (Marcus et al., 1993)

Chapter 1

Introduction

The linguistic processes responsible for creating new words can be gathered under the term *morphology*. Morphology is the sub-field of linguistics that studies words' internal structure. For instance, the word *disconnections* is made up of four parts: *dis*, *connect*, *ion* and *s*, each adding another piece of meaning to the whole word. Computational morphology, then, is an interdisciplinary field involving both computer science and linguistics, nowadays extensively using machine learning techniques, with the aim of modeling morphological processes for several reasons. The most common goal, which we discuss further below, is to use models of morphology as a component in a natural language processing (NLP) system, such as automatic speech recognition or statistical machine translation, i.e. for engineering purposes. Another, less common reason is to use computational morphology models in cognitive science, in which case the typical scenario involves simulating the learning of some morphological process with the aim of capturing the human learning process as closely as possible. Finally, computational models of morphology could also be used in linguistic exploratory studies, with the hope of discovering something new about the modeled process itself.

1.1 Motivation

This dissertation presents computational models of morphology that are developed with the most common, the engineering goal, in mind. It is widely accepted that natural languages have an unbounded number of words, and so it is impossible to store just all of them. Moreover, for any statistical NLP application, the relevant statistics for each word have to be computed. With a small vocabulary, there will be enough occurrences of each word in a reasonably large text corpus to reliably estimate all the necessary statistics. However, as vocabularies get larger, they include more and more words that occur very infrequently even in very large text corpora. Thus, the

statistical estimates with larger vocabularies get more and more imprecise. This is the problem of *sparsity* that lurks at every corner of the natural language processing field. Modeling morphological processes helps tackle sparsity by representing language in a more compact format that, rather than listing all possible word forms, describes how they can be constructed from morphological components by reusing the same components in many different words. For instance, even if a speech recognizer cannot recognize the word *disconnections*, it might be trained to recognize the individual pieces *dis*, *connect*, *ion* and *s*, each of which is relatively common.

Computational models of morphology can consider different aspects of the problem, leading to at least three different kinds of tasks:

- **Morphological segmentation**: splitting words into smallest syntactically or semantically meaningful units;
- **Part-of-speech (POS) tagging (clustering)**: dividing words into different classes based on their syntactic function, such as noun, verb, and adjective. Linguistically, this is not strictly part of morphology and rather belongs to the study of syntax. However, as words with different syntactic function behave morphologically differently, learning this clustering is important to morphology as well.
- **Morphological analysis**: assigning words an appropriate set of morphological attributes, such as gender, number and case, known to exist in the respective language.

All these tasks are closely interconnected but not completely overlapping, which makes them potentially informative for each other in a joint learning model. The benefits of joint learning manifest especially in the unsupervised or semi-supervised learning paradigm, where the model interacts more freely with the data, than in a supervised learning setting, which is constrained by the given set of correct labels. For instance, knowing that the suffix *-ion* in the word *disconnection* refers to a noun helps with the POS tagging. Conversely, knowing that all the words containing a hypothesized *-ion* suffix share a POS tag makes us more confident that this is, in fact, a real suffix. However, most of the previous models have treated those tasks as independent of each other, thus forgoing the opportunity to exploit the cross-referential cues present in the data. Although there are models that use information about another, related task, they usually acquire this information during preprocessing. For instance, a relatively popular scenario is to supply the POS induction system with morphological suffixes. These morphological suffixes, however, have been typically learned by an independent model. Hence, in this scenario the interdependence of the two tasks is only exploited partially, whereas during joint learning, which is the course taken in this dissertation, both tasks potentially benefit.

Although supervised morphological models are usually more accurate in terms of engineering, they are applicable only to a small set of well-studied languages that have large morphologically annotated corpora available. However, more than 99% of the world's languages are still considered less-studied (Ostler, 2003), which typically also means that they lack annotated resources. Therefore, developing models able to learn from unlabeled or partially labeled data is highly relevant for computational morphology.

This dissertation presents computational models of morphology that use unsupervised or weakly-supervised learning. This choice is motivated first by the reasons mentioned above, that unsupervised models can be applied to languages with no or little morphologically annotated data available. Secondly, there are many morphological cues in the plain unannotated text, manifested in the words' orthography and distributional context. Using unsupervised models enables studying methods of how those cues can be employed and related to each other in order to reveal the desired morphological patterns and structures.

In unsupervised and semi-supervised models, we need to use latent variables for representing the structures inferred by the model. Moreover, in joint models, the full underlying representation of the morphological system has to be expressed in a way that the information can be shared by the different parts of the model. For example, when jointly learning POS tags and morphological segmentations, the model has to know what kinds of segments are possible and how they affect the POS of the word. To build flexible latent-variable models, we adopt a Bayesian framework that naturally allows learning with incomplete or missing data.

Most of the previous work in computational morphology using latent-variable models has represented the task-defined latent variables only, such as morphemes or POS tags. However, in the Bayesian framework we must not be confined to those latent variables only, but can also model additional variables that capture some intermediary aspect of the learned process. For example, it will be possible to model latent sub- or superstructures of morphemes that have no meaning in linguistic terms and thus would be hard to define in supervised learning setting and, therefore, impossible to label. Empirically, however, modeling those latent structures proves to be highly beneficial for the task of morphological segmentation.

Non-parametric prior distributions that have been shown to be useful for unsupervised language learning (Teh, 2006a,b; Goldwater et al., 2011) add further flexibility to our models. The non-parametric Bayesian approach treats the model size as another parameter to be inferred from data and thus enables learning models with growing complexity as the amount of training data increases. For instance, the unsupervised POS induction task can be interpreted as a clustering task with unknown number of components. The

decision about the number of clusters can be delegated to the model, which learns it according to the evidence present in the data.

1.2 The claims and contributions

The central part of this dissertation consists of the following Claims:

A. For unsupervised or weakly-supervised learning of natural language structures, it is vital not only to model the known properties of those structures, but also some regularities or patterns that are latent, even if they have no specific meaning in linguistic terms.

B. Unsupervised learning can be improved by integrating different aspects of the same process into the joint model; this helps to resolve ambiguities, leading to overall better results. Of course, the interaction between different aspects of the data must be properly modeled in order for this claim to be valid.

In order to support these statements, we present three computational models defined in the non-parametric Bayesian framework, all modeling morphology from slightly different aspects and using different representations of data. The contributions related to each model are described in the following sections. The implementations of the models are available upon request.

1.2.1 Joint POS induction and morphological segmentation

In this part, we develop an unsupervised model for joint induction of POS tags and morphological segmentations. We use a hierarchical Dirichlet process to implement an infinite hidden Markov model over the latent tag trigrams conjoined with a simple unigram segmentation model. The aim is to model the interactions between the POS tags and morphological segmentations so that both structures would influence each other in an unsupervised learning setting. We focus on learning the clustering that roughly matches the set of coarse-grained POS tags, such as nouns, verbs, and adjectives, as this choice enables easier comparison with previous comparable systems. The main contributions of this work are:

1. State-of-the-art results in unsupervised POS induction over several languages;
2. Empirical evidence that morphological information and POS assignments influence each other in the joint learning setting (Claim B).

1.2.2 Weakly-supervised morphological segmentation

This part of the work uses the Adaptor Grammar (AG) framework (Johnson et al., 2007) that enables defining non-parametric Bayesian models using an extension of probabilistic context-free grammars (PCFG). We use AGs to

define models for inferring morphological segmentations. A small amount of labeled data is used in two different ways to guide the learning of suitable morphological grammars for different languages. The main results are:

1. State-of-the-art results in semi-supervised morphological segmentation across several languages;
2. Empirical evidence that the grammars jointly modeling additional latent sub- or superstructures of morphemes perform consistently better than the grammars generating flat morpheme sequences only (Claims A and B).

Considering that AG was originally defined as a general framework for unsupervised transductive learning, the further technical contributions are:

3. Two methods for using AGs for semi-supervised learning;
4. More generally, showing that AGs framework can be used on a full continuum from completely unsupervised to fully supervised learning;
5. Showing how to use AG's posterior grammar as a model for inductive learning.

1.2.3 Morphosyntactic clustering using morphological and distributional cues

In this work, we use an infinite mixture model for clustering words into fine-grained morphosyntactic clusters using both distributional and orthographic cues. The distributional information is represented by continuous vectors (word embeddings) obtained from an existing neural network language model. These vectors capture the information of all the contexts where the word has been observed. Orthography is included into the model via a non-parametric prior distribution (distance-dependent Chinese restaurant process (Blei and Frazier, 2011)) that encourages words with similar suffixes to be clustered together. The model induces a variable number of clusters, which makes using the standard evaluation methods and comparison with previous work in POS induction difficult. Therefore, we evaluate by comparing the model results with the K-means clustering on the same data with K set equal to the number of induced clusters. The contributions of this work are:

1. The current benchmark in unsupervised fine-grained morphosyntactic clustering (as there exist no directly comparable previous work);
2. Empirical evidence that the model using both sources of information learns better clusters than the one using distributional information only (Claim B);
3. Showing that the non-parametric model allowed to choose the number of morphosyntactic clusters freely makes a reasonable choice in English (Claim A).

As the model uses some familiar technical elements in novel situations, there are further technical contributions:

4. Defining a model-based similarity function for distance-dependent Chinese restaurant process and learning it during inference;
5. Using word embeddings as multivariate Gaussian random variables in the Gaussian mixture model.

1.3 Outline of the dissertation

Chapter 2 first provides an overview of the linguistic morphology followed by the relevant work in computational morphology focusing mostly on unsupervised morphological segmentation and POS induction tasks.

Chapter 3 gives the technical background necessary for understanding the non-parametric Bayesian models presented in the later chapters. In particular, we discuss the non-parametric Bayesian priors and inference with Markov chain Monte Carlo methods.

Chapter 4 presents the joint unsupervised POS induction and morphological segmentation model using hierarchical Dirichlet processes introduced in section 1.2.1. We provide experimental results in POS induction on 15 languages and segmentation results on four languages. The chapter is based on publication I (Sirts and Alumäe, 2012).

Chapter 5 focuses on the task of morphological segmentation. We present two weakly-supervised methods mentioned in section 1.2.2 for performing morphological segmentation using the Adaptor Grammars. The performance of these models is evaluated on five languages using various baselines. The basis of this chapter is publication II (Sirts and Goldwater, 2013).

Chapter 6 explains the unsupervised morphosyntactic clustering model based on the distance-dependent Chinese restaurant process (ddCRP) using distributional and morphological cues introduced in section 1.2.3. We experiment with three different ddCPR-based models and provide clustering results on 11 languages. The chapter is partly based on publication III (Sirts et al., 2014). It also contains unpublished experimental results on other languages than English and the analysis of those results.

Chapter 7 concludes the thesis. We take a joint look at all three models to view them as components of an unsupervised morphology induction system and discuss some issues about the typical assumptions about the relationships between POS tags and morphological suffixes. Finally, we assess the Claims and their validity according to the presented results.

Chapter 2

Morphology and POS tagging

This chapter provides the background information about morphology for understanding the context of this dissertation. We start by explaining the morphological processes from the linguistic point of view. Then we describe the previous work in computational morphology, putting the strongest emphasis on the tasks of morphological segmentation and unsupervised POS induction. Finally, we present the standard methods for evaluating both the morphological segmentation and POS induction results.

2.1 Morphology

In this section, we give a short overview of the field of linguistic morphology. The material is based on Haspelmath and Sims (2010).

Morphology is about the internal structure of words and operates with the subword units called *morphemes*. Morphemes can be either *roots* or *affixes*. Roots carry the basic indivisible meaning of the word. Affixes can be either *prefixes*, which appear before the root morpheme or *suffixes*, which occur in the word after the root. For instance, the word *connect* is a root and forms a monomorphemic word. The word *connect_ion* consists of a root and a suffix, whereas the word *dis_connect* is formed by a prefix and a root. Depending on their usage of affixes, languages are divided into primarily suffixing or primarily prefixing, the suffixing languages being prevalent among the world's languages[1].

Affixes can be either *inflectional* or *derivational*. Inflectional suffixes add appropriate syntactic properties to the word whereas derivational suffixes change either the semantic meaning or POS of the word. For instance, the suffix *-s* in the word *table_s* is inflectional because it marks the plurality

[1] `http://wals.info/chapter/26`

Table 2.1: Examples of different morphological concepts related to word forms.

Word	**Lexeme**	**Root**	**Stem**	**Lemma**	**Affix**
connections	`connection`	connect	connection	connection	-s
tables	`table`	table	table	table	-s
reader	`reader`	read	reader	reader	-
ate	`eat`	ate	ate	eat	-

of the word *table* while semantically the words *table* and *tables* share the same meaning. The suffix *-er* in the word *read_er* is derivational because it changes the meaning of the word—from an action to an agent. On the other hand, the suffix *-ion* in the word *connection* does not change the semantic meaning of the word *connect* a lot because both *connect* and *connection* denote roughly the same semantic concept. However, the suffix *-ion* is still considered derivational because it changes the POS of the word—from the verb to a noun.

A *stem* is the base of an inflected word. The stem of a word does not necessarily have to be indivisible and can consist of a root that has derivational suffixes attached to it. For example, the complex word *connection* is a stem (but not a root) and it can be used to derive the plural form *connection_s* by adding the inflectional suffix *-s*.

A *lexeme* is a kind of abstract word denoting a lexical concept. Each lexeme has a stem and all the words formed by applying different inflectional affixes to this stem belong to the same lexeme. A *lemma* is the base form of a lexeme; it is a specific word having a concrete form. In some languages lemma matches with the stem of the lexeme, but in other languages lemmas are also composed using inflectional affixes. For example, in English the lexeme `table` denotes the specific word forms *table* and *tables*. In most English nouns the lemma corresponds to the stem and in this example is thus *table*. Lexemes can also have several variants of the same stem, in case the application of inflectional processes brings along stem alternations. For instance, the English lexeme `eat` has two stems: *eat* used in the forms *eat*, *eaten* and *eating*, and *ate* that is used in the past tense form. Some examples of stems, roots, lemmas, affixes and lexemes are provided in Table 2.1.

In addition to inflectional and derivational processes, new words can be formed by using *compounding*. In compound words, several stems are joined together. For instance, the word *fireplace* is a compound consisting of stems *fire* and *place*.

The notion of roots and affixes allows defining morphology as "*... the study of the combination of morphemes to yield words*" (Haspelmath and Sims, 2010, p.3). This definition is valid when the morphology is expressed in

terms of distinct segments that are concatenated together to form derived or inflected words. The languages that mostly use this kind of process are called *concatenative* or *agglutinative*. In other languages, inflections and derivations are carried out by stem alternations, changes in phonology or stress patterns or some other non-concatenative process. In those languages, this simple definition of morphology is not adequate. Another definition of morphology (Haspelmath and Sims, 2010, p.2) says that "*Morphology is the study of systematic covariation in the form and meaning of words*". This definition implies that any systematic changes in the word that lead to the systematic change in the word's meaning either syntactically or semantically, is to be considered under morphology. Languages that mostly use non-concatenative morphological processes are called *fusional*.

Languages also differ in their morphological complexity. *Analytic* languages use little or no morphology at all; *synthetic* languages, on the other hand, make heavy use of morphology. The categories of concatenative vs. fusional and analytic vs. synthetic can be used to describe the properties of any world's language. However, there are no clear and distinct language classes. Rather, all languages lie on a continuum along the two axes, where one describes the extent of morphology present in the language (analytic vs. synthetic), and the other specifies the characteristics of the morphology (fusional vs. agglutinative).

2.1.1 Inflectional morphology

Inflectional morphology deals with applying inflectional affixes to a stem to form words that conform to specific syntactic or semantic requirements. During inflectional processes, the lexeme does not change, and the primary meaning of the word remains the same. The syntactic requirements come from the context in terms of *agreement*, such as the noun subject may demand the verb to agree in person and number. For instance, the subject *student* requires the verb *read* to appear in 3rd person singular form in the sentence "*The student reads his notes*".

There is a whole range of morphological categories or features used by the inflectional morphology. The specific categories used depend on the language. Also, each part-of-speech class uses a different set of morphological categories. The most common inflectional features across languages are number, case, gender, and person. Usually, all combinations of the applicable categories are valid, generating a whole set of different inflected word forms for each base form or lemma. These words retain the same basic meaning, i.e. they belong to the same lexeme, but behave differently syntactically or semantically. This set of words can be organized into a grid called *paradigm*. The structure of a paradigm is determined by the morphological categories used for the specific POS in the particular language. The grid has as many dimensions as there

are different morphological categories for that POS. For instance, the English nouns are inflected for number and case. The number can be either singular or plural and case either nominative or genitive. Thus, the noun paradigm is a two-dimensional grid, containing four word forms corresponding to all different combinations of both morphological features. It is an example of a very small paradigm as English is morphologically quite simple (e.g. analytic) language. An example of a noun paradigm of a synthetic language can be brought from Estonian, where nouns are inflected for number and 14 cases, making up to 28 different word forms for a single noun lexeme.

2.1.2 Derivational morphology

Derivational morphology deals with word creation. New concepts or words with novel meaning can be created by attaching derivational affixes to existing words. For instance, the English suffix *-er* can be used to turn a verb to an actor noun: *read/reader*, *play/player*, *view/viewer* etc. The change in POS is characteristic to derivational morphology, although it does not happen with every derivational affix. For instance, the English noun *child* does not change its POS when attached the affix *-hood* to derive the word *childhood*.

Derivational processes are usually carried out before attaching inflectional affixes and thus, the derivational affixes typically occur closer to the root than inflectional affixes. For instance, the English inflectional noun plural suffix *-s* is attached after the derivational suffix *-er* is added to the verbal root as in *read_er_s*.

Derivational morphology does not have such a well-defined structure as paradigms in inflectional morphology. In each language, there is a finite set of derivational affixes. Some of them are highly productive, combining with many words, and some are quite unproductive and can be used only on a finite list of roots. Therefore, it can be very hard to come up with a set of reliable rules describing which affixes can combine with which roots and stems. Usually, the native speakers recognize quite easily whether a derivation is grammatical or acceptable.[2] For the unproductive affixes, the full list of valid derived words could be listed in a dictionary.

Derived words are organized into *word families* where the head of each family is a basic concept, and the other members of the family constitute the words that are in some way derived from the root concept. Compounding is in this sense also considered a derivational process, and compound words belong to several word families simultaneously, one family for each root.

[2]There could be derivations that are grammatical but don't sound right to the native speaker.

2.1.3 Compounding

The process of compounding takes two or more stems and combines them together to form a new word. For instance, the English word *haystack* is a compound made of simple stems *hay* and *stack*. The meaning of the compound in this example is compositional, e.g. it is a stack made of hay. The meaning of a compound can also be non-compositional, in which case the meaning of the compound word cannot be deduced from the meanings of its parts. For example, the English compound *hot dog* means something else than a dog having high temperature. In many languages, the compound words are written together, but this is not necessarily the case. For example, in English many compounds, such as *hot dog*, are written as separate words. The most frequent compounding pattern combines two nouns, and usually only the last part of the compound is inflected.

2.1.4 Allomorphy

In a purely concatenative language, complex words are formed by adding derivational and inflectional affixes to the root morpheme. In many languages, however, this process is more complicated because the stem might undergo changes during affixation. Also, depending on the stem, the affixes might look (or sound) different. This phenomenon of having several variants of the same morpheme is called *allomorphy* and the set of morphemes carrying the same syntactic or semantic meaning are called *allomorphemes*.

Allomorphy occurring in stems is called *stem alternation*, which can occur both in inflectional and derivational morphology. For example, the stems of the nouns *leaf* and *knife* change when forming plurals: *leaf*/*leav_es*, *knife*/*knive_s*. As examples of stem alternation in derivational morphology consider the verb *destroy* and the adjective *broad*, and the respective derived nouns *destruc_tion* and *bread_th*.

Affix allomorphy occurs mainly in inflectional affixes. Common examples of suffix allomorphy in English are *-d* and *-ed* when forming past tense of regular verbs, as for example in *look_ed* and *arrive_d*, and *-s* and *-es* in present 3rd person form of verbs, as in *look_s* and *cross_es*.

2.2 Morphology in NLP applications

For many languages other than English, accounting for morphology is necessary in major NLP applications, such as automatic speech recognition (ASR), statistical machine translation (SMT), information retrieval (IR) and natural language generation (LG). Modeling morphology in those applications alleviates the problem of unknown words that inevitably occurs in systems using fixed size vocabularies, and unreliable parameter estimates caused by the long-tail distributions of natural language word frequencies.

There have been many attempts to incorporate morphology into different NLP applications; this section gives just some reference points illustrating the work done in this area.

IR systems use morphology primarily for text normalization. The goal of IR is to query documents containing the same or similar concepts. Performing word-based queries in morphologically complex languages requires that all the inflected word forms of the same lexeme, as well as the complex words derived from the same root, are listed in the query. A simple way to reduce the number of query words is to normalize the text by using stemming, lemmatization or compound splitting (Krovetz, 1993; Hollink et al., 2004). There have also been attempts to use fully morphologically segmented input in IR queries, but rather for the purpose of task-based evaluation of different segmentation systems (Kurimo et al., 2009) than for the sake of the IR task in itself.

Both ASR and SMT use statistical language models (LM) to assess the probability of a sentence. These LMs operate using fixed size lexicons and only those words present in the lexicon can appear in the output. This can pose serious problems for languages with productive morphology as the active vocabulary in those languages is typically much larger than any lexicon of reasonable size. A popular solution involves using a morpheme lexicon instead of words (Geutner, 1995; Ircing et al., 2001; Teemu Hirsimäki and Kurimo, 2005; Alumäe, 2006; Creutz et al., 2007; Mihajlik et al., 2007; Stas et al., 2012). The training corpus is preprocessed by a morphological segmentation system, and then the LM is trained using sequences of morphemes. Morpheme-based LM enables constructing words that did not fit into the word lexicon or were not even observed in the training corpus. This flexibility comes at a price though, as the combinatorial freedom introduces additional noise that can result in words that do not exist in the language. These observations have motivated research in choosing the appropriate granularity of the segmentation that would reduce the rate of out-of-vocabulary words while increasing the recognition accuracy or at least not decreasing it substantially (Geutner, 1995; Arisoy et al., 2006; Hirsimäki et al., 2009). In some languages, the full morphological segmentation of both inflectional and derivational affixes has been found to be too fine-grained, whereas splitting off inflectional affixes or separating compound words only has led to satisfactory results (Berton et al., 1996; Carter et al., 1996; Larson et al., 2000; Alumäe, 2012).

Morphology is also used more and more in SMT translation models, especially when the source and target languages have different morphological complexities. Here, the approaches can be roughly divided into two categories, using either morphological segmentations or morphological features.

The methods using morphological segmentation (see for example Niessen and Ney (2000); Lee (2004); El-Kahlout and Oflazer (2006); Virpioja et al.

(2007); Oflazer and El-Kahlout (2007); Nguyen and Vogel (2008); de Gispert et al. (2009); Luong et al. (2010); Clifton and Sarkar (2011); Eyigöz et al. (2013)) first process the parallel training text (either on the source side or the target side or both) into morphemic segments and then align the segmented text. In the result, both word-to-word, segment-to-segment and also word-to-segment alignments can occur. This strategy helps to reduce the number of many-to-one alignments where a whole phrase in one language is aligned with a single complex word in another. Those many-to-one alignments are problematic because the frequencies of such aligned components are usually quite low, and so the reliable estimation of their parameters may need more parallel training data than is available. Also, combining segments in the target language enables generating words that were never observed in the training corpus. The problems accompanying this approach are the same as in language modeling—increased flexibility increases the possibility of forming non-existent words that again degrades the overall translation accuracy.

The methods using morphological features proceed differently—they typically operate on words represented as base forms (lemmas or stems) enriched with morphological features obtained via morphological analysis. The translation procedure is different depending on whether the source language is more complex than the target language or vice versa. When the source language is morphologically complex, then the morphological information can be used in various ways for reducing its sparsity (Niessen and Ney, 2004; Goldwater and McClosky, 2005). The opposite scenario is more sophisticated and can be viewed as a two-step procedure. First translate the base forms and supply them with the appropriate set of morphological features and then generate the inflected word forms using a morphological generation system (Bojar, 2007; Minkov et al., 2007; Toutanova et al., 2008; Yeniterzi and Oflazer, 2010; Fraser et al., 2012). Morphology is employed in a similar fashion also in NLG systems. During sentence planning, relevant morphological features are attached to the word's base form and then the surface realization component generates the appropriate inflected word form (Gamon et al., 2002; Smets et al., 2003; de Novais and Paraboni, 2013). Recently, models using morphological features have also emerged in ASR language models (Kirchhoff et al., 2006; Arisoy et al., 2012).

All these systems have used features derived from the output of supervised or hand-crafted morphological analyzers. This hinders using such methods on languages where morphological analyzers are not available. Using unsupervised morphological induction systems could provide an alternative. However, there have been only single attempts in this direction so far (Clifton and Sarkar, 2011), mainly because the performance of unsupervised or weakly-supervised morphological analyzers is still far from the supervised

or hand-crafted ones both in terms of functionality and accuracy. Also, finding proper ways for exploiting the existing unsupervised morphology learning systems for any NLP task of interest is a non-trivial research problem on its own.

2.3 Computational morphology

The goal of computational morphology is to model morphological processes and simulate their behavior with computer programs. There is no single task of computational morphology, rather under this term several different tasks have been addressed, such as morphological segmentation, analysis, and tagging, stemming, lemmatization, and paradigm learning. The primary focus in this section is on morphological segmentation systems using unsupervised or semi-supervised learning as this is the task addressed in the models of this dissertation. We will also review other related tasks in computational morphology, such as stemming, lemmatization, morphological analysis, and paradigm learning and discuss how they are related to each other and the morphological segmentation task.

2.3.1 Hand-crafted and supervised systems

The simplest task one can imagine in computational morphology is stemming. The goal of stemming is to strip off suffixes from words so that only the stem remains. The most popular stemming algorithms, Lovins (Lovins, 1968), Porter (Porter, 1980) and Paice/Husk (Paice, 1990) operate according to similar principles. They utilize a list of valid suffixes and/or hand-crafted rules that describe in which circumstances a suffix can be stripped or not.

The development of stemming algorithms was led by the field of information retrieval (IR) because automatic stemming provides a simple and powerful way for normalizing textual data for the purpose of reducing the sparsity. The data normalization goal also implies that the stemming result can be noisy or erroneous in terms of linguistics, as long as it serves to be useful for IR. For example, the linguistically correct stem for the word *connections* would be the complex stem *connection* instead of *connect*, which would be the root of the respective word family. However, from an IR point of view, the stem *connect* is more useful as it clusters together more words with similar or close meaning.

Lemmatization, as stemming, can be used for text normalization, and it is part of the morphological analysis. The goal of lemmatization is to label each word with its base form (lemma). As opposed to the stem, which is always a substring of the word form and might not be a valid word itself, lemma is always a concrete word. Thus, for lemmatization the knowledge about how to form the specific words constituting lemmas in the particular language and for specific POS must be known. Techniques used

to learn lemmatization models include learning sets of transformation rules (Erjavec and Dcroski, 2004; Jusic et al., 2010), using memory-based learning (Clark, 2002), encoding transformations with finite state transducers (FSTs) (Clark, 2002; Dreyer et al., 2008; Toutanova and Cherry, 2009) or treating lemmatization as a tagging task, where each tag is the description of the transformations that must be carried out (Chrupala, 2006; Gesmundo and Samardžić, 2012).

Full morphological analysis includes labeling each word token with the lemma, POS tag and the sequence of morphological category values. For example, English nouns are inflected for number and case and so the morphological analysis for a noun would list the lemma and the appropriate values for both of those features.

The first successful computational approach for morphological analysis was two-level morphology (Kaplan and Kay, 1981; Koskenniemi, 1983). This method assumes the full morphological system description of the language that can be compiled into FSTs. Although the resulting models are highly accurate, building them is time-consuming and costly due to the amount of linguistic expertise needed. The morphological systems for European and bigger world's languages might be well studied and known, but for many world's languages, the descriptions of the morphological system still just do not exist.

Data-driven supervised morphological analyzers use either morphological lexicons or annotated corpora or both. Different methods have been adopted for learning morphological analyzers from such resources, such as memory- or analogy-based learning (van den Bosch and Daelemans, 1999; Stroppa and Yvon, 2005), learned rules (Erjavec and Dcroski, 2004), guided learning (Georgiev et al., 2012) and log-linear models (Hajič, 2000; Adler et al., 2008; Müller et al., 2013) to name a few examples.

The task of paradigm learning is essentially similar to the morphological analysis with the major difference of explicitly representing the paradigmatic structure of the morphology. The paradigm modeling systems cluster the words into lexemes and place each word into a proper slot of the respective lexeme's paradigm grid. The explicit modeling of orthographic relations between different forms of the same lexeme provides synthesis functionality, allowing the generation of word forms not observed in the training data. Supervised paradigm learning models (e.g. (Dreyer and Eisner, 2011; Durrett and DeNero, 2013; Ahlberg et al., 2014)) use wholly or partially observed inflection tables for bootstrapping the paradigmatic patterns.

2.3.2 Unsupervised systems

One of the most popular tasks tackled in unsupervised computational morphology is morphological segmentation. The goal of morphological segmentation is to split words into sequences of morphemes. For example, the English word *animations* could be segmented as *animat_ion_s* where the first morpheme is the root, the second morpheme *-ion* is derivational suffix, and the final segment is inflectional suffix. The popularity of this task is easy to understand when considering that, according to a narrow definition, morphology is the study about how words are composed of morphemes. Also, as concatenation is the most common morphological pattern in world's languages (Haspelmath and Sims, 2010, p. 40), focusing on morphological segmentation seems a reasonable thing to do.

The history of computational morphological segmentation starts with the work of Harris (1955) who, for each prefix in each word recorded the list of following characters, calling those lists *successor varieties* and the lengths of those lists *successor counts*. He proposed marking morpheme boundaries in places where the successor count is high while the small successor count suggests a morpheme-internal character sequence. Strictly speaking, Harris was dealing with phonemic utterances, so his task was to find both word and morpheme boundaries. However, the same idea can be applied to orthographic words as well. Several later studies have built on Harris' work using the idea of successor variety, such as Hafer and Weiss (1974); Dejean (1998); Bernhard (2006); Bordag (2006); Keshava and Pitler (2006); Dasgupta and Ng (2007); Bordag (2008). The main problem of those systems is the necessity to use different heuristics in the form of various thresholds and processing steps giving the whole approach an ad hoc flavor.

The next wave of work in morphological segmentation starting from Brent et al. (1995) and including the two most well-known unsupervised morphological segmentation systems—Linguistica (Goldsmith, 2001) and Morfessor (Creutz and Lagus, 2007)—adopted the minimum description length principle (MDL) (Rissanen, 1989). The idea of the MDL is based on the assumption that the model with the shortest description length is the most optimal. The MDL principle, similar to the successor count, is a heuristic objective. However, unlike the successor count that requires setting arbitrary thresholds, MDL provides a principled way of making decisions about the model.

Learning morphological segmentations in the MDL framework is defined as minimizing the description length of the morphological system—morpheme lexicon and the encoding of the segmented corpus. The optimal MDL solution finds a balance between the compactness of the lexicon and the length of the corpus encoding. For example, the most compact lexicon would list just

single characters. However, this model would not compress the corpus in any way. On the other hand, the most compact description of the corpus would encode each word with a single pointer to the lexicon, which would lead to a large lexicon listing every word type. The best solution lies somewhere in between those two extremes, and the morphological segmentation is then provided by looking up the sequence of lexicon elements used to encode each word.

The MDL principle has been applied to morphology learning in two ways: either explicitly computing the description lengths of the model and corpus in bits (Goldsmith, 2001, 2006; Creutz and Lagus, 2002) or defining a probabilistic Bayesian model and looking for the maximum a posteriori (MAP) solution (Snover and Brent, 2001; Snover et al., 2002; Snover and Brent, 2003; Creutz, 2003; Creutz and Lagus, 2004, 2005). In case of probabilistic interpretation, the negative logarithmic prior probability of the model corresponds to the length of the model and the negative logarithmic probability of the data given the model corresponds to the length of the encoded corpus. As global search over the full solution space is intractable in either formulation, different heuristic search strategies are adopted to minimize the description length or maximize the posterior probability.

Although MDL provides a general and principled framework for the model selection, it does not capture one of the most important properties of the natural language structures, that their distribution resembles power-law (Zipf, 1932). It may be that the compactness of the systems preferred by the MDL also reflects this property. However, MDL does not provide a straightforward way for explicitly expressing this preference in the model setup. The previous MDL-based models have used uniform priors over morphemes or simple distributions dependent on the length of the morphemes. The only exception is Creutz (2003) who develops a prior over morpheme frequencies to model the power-law behaviour.

Non-parametric Bayesian (NPB) methods, used for defining the segmentation models described in this dissertation, also prefer, similar to MDL, compact models but in addition to that also naturally generate power-law-like distributions over lexicon elements. Computing the full posterior in non-parametric Bayesian models is still intractable, but instead of devising ad hoc heuristic search strategies, we can use Markov chain Monte Carlo (MCMC) methods to simulate samples from the posterior. In addition, the heuristic search methods used in the previous MDL-based morphology learning systems operate greedily and are thus confined to the closest local optimum. On the other hand, MCMC methods permit occasionally also making low-probability choices, thus potentially enabling exploring larger parts of the whole sample space.

Most of the previous segmentation systems using NPB models study the task of unsupervised word segmentation (Goldwater et al., 2006a; Johnson, 2008b,a; Goldwater et al., 2009; Johnson and Goldwater, 2009; Mochihashi et al., 2009). The morphological segmentation models of this dissertation build on those works. The segmentation component in Chapter 4 is similar to the one described by Mochihashi et al. (2009). They present a hierarchical Pitman-Yor process (PYP) language model over latent words in a sentence while we build a Dirichlet process (DP) model over latent morphemes in a word. The unsupervised and weakly-supervised segmentation models in Chapter 5 use the same Adaptor Grammar formalism as Johnson (2008b,a) and Johnson and Goldwater (2009). However, instead of defining grammars over sentences and words as is appropriate in the word segmentation task, we describe grammars over morphological structures inside the words with the aim of learning the latent morphemes.

Another line of work has incorporated the task of morphological segmentation inside the NPB language model (Goldwater et al., 2006b). In this model, the word tokens are generated from a PYP and the base distribution generates word type segmentations. The segmentation model is pretty simple, generating each word as consisting of a stem and an optional suffix. The possible segments are modeled with a multinomial distribution with a Dirichlet prior over all possible substrings found in the corpus. The morphology component has been recently extended by Frank (2014) also to use non-parametric prior for inferring the morpheme lexicon of dynamic size. Another variant of Goldwater et al. (2006b) model is presented by Zhao and Marcus (2012) who, instead of NPB methods, use log-normal distributions for generating power-law behaviour.

Some of the unsupervised segmentation models also label the learned segments, in which case the task may also be referred to as unsupervised morphological analysis. The set of labels may include stems, prefixes and suffixes (Creutz and Lagus, 2005; Bernhard, 2008). Another approach has been to label segments with abstract morphemes with the goal of linking all allomorphic variants of the same segment to a single abstract morpheme (Goldsmith, 2006; Dasgupta and Ng, 2007; Bordag, 2008; Kohonen et al., 2009; Lignos, 2010). Although such segment labeling has not been the goal in this dissertation, some of the models in Chapter 5 also generate morphemes labeled as stems, prefixes or suffixes.

Some morphological segmentation systems specify the model as a collection of structures, typically expressed as sets of suffixes, which loosely correspond to inflectional classes (e.g. Goldsmith (2001); Snover and Brent (2003); Monson et al. (2007); Zeman (2008); Can and Manandhar (2012)). They also cluster stems that can be used together with all suffixes in some suffix set. For instance, in English a suffix set could consist of suffixes {*-s*,

-ed, *-ing*, \$[3]}, which would correspond to the regular verb paradigm. The stem set related to this suffix set would contain the roots of regular English verbs, such as {*work*, *laugh*, *play*, ...}. These systems provide more structure than the simple segmentation models and, therefore, seem very promising in terms of future research in unsupervised morphology induction. However, none of the models of this dissertation take this approach.

2.3.3 Semi-supervised systems

Most morphological segmentation systems have been using unsupervised learning techniques, which has been motivated by the fact that annotating training data is costly in terms of time and money. Perhaps this course of actions has also reflected the influence of Harris, who, as a real field linguist, assumed nothing about the language when describing the successor variety based method (Harris, 1955). However, frequently even the models declared as unsupervised need to tune some parameters on a small set of annotated data. This realization has recently led the research in morphological segmentation focus more on semi-supervised learning approaches. Semi-supervised models can exploit annotated data in different ways, either for hyperparameter tuning[4], for learning sufficient statistics or for performing model selection.[5]

The two morphological segmentation methods presented in Chapter 5 also use semi-supervised learning utilizing the annotated data in one case for model selection and in the other case for learning sufficient statistics. The amount of annotated data used by those models is small, with only 1000 gold-segmented word types providing substantial improvements over unsupervised models.

Next, we provide an overview of the other recent works in semi-supervised morphological segmentation. Snyder and Barzilay (2008a) present a non-parametric Bayesian model for cross-lingual segmentation, where the hand-annotated segmentations provided in one language are propagated to segment the words in the other language. Poon et al. (2009) and Ruokolainen et al. (2014) develop semi-supervised segmentation models using conditional random fields (CRF) (Lafferty et al., 2001). Also, several semi-supervised variants of Morfessor family models have been proposed (Kohonen et al., 2010a; Virpioja et al., 2013; Grönroos et al., 2014).

What is interesting about those models is that all of them have been demonstrated to be applicable using different amounts of supervision. For example, the cross-lingual model of Snyder and Barzilay (2008a) has been also demonstrated in unsupervised setting (Snyder and Barzilay, 2008b). Poon

[3] \$ denotes the empty suffix.

[4] Those models are usually regarded as unsupervised.

[5] This can be considered as a sophisticated version of the hyperparameter tuning.

et al. (2009) conducts experiments also in both unsupervised and supervised setting. Ruokolainen et al. (2014) first developed the model to perform fully supervised segmentation (Ruokolainen et al., 2013).[6] Semi-supervised Morfessor (Kohonen et al., 2010a; Virpioja et al., 2013) computes separate weights for the labeled and unlabeled data likelihoods and setting the weight of the labeled data likelihood to zero recovers the unsupervised Morfessor baseline (Creutz, 2003). Similarly, setting the weight of the unlabeled data to zero gives a supervised segmentation model. Morfessor FlatCat (Grönroos et al., 2014) is a mix of Morfessor baseline (Creutz, 2003) and Morfessor CatMap (Creutz and Lagus, 2005) whose usage is also demonstrated both in unsupervised and semi-supervised setting.

The semi-supervised Adaptor Grammar (AG) segmentation model (in Chapter 5) can be also used with different amounts of labeled data. When no labeled data is provided, the model performs unsupervised learning. Variable amounts of labeled data can be used for semi-supervised learning. Finally, when unlabeled data is omitted, fully supervised learning can be performed.[7] An essential difference between the AG-based models and other semi-supervised models discussed above is the possibility in AG models to adopt different annotation schemes dynamically. Other models implicitly assume that the training data is annotated with the morpheme boundaries. AGs allow flexibly defining various labeling schemes. For instance, the morphemes in the labeled data can be annotated as stems, prefixes, and suffixes. Another possibility is to label derivational and inflectional affixes. Also, multi-level annotations are possible, for example annotating compound parts and morphemes inside those, which is done in one of the models in Chapter 5.

There have also been a few examples addressing other computational morphology tasks in semi-supervised setting. Wicentowski (2002) uses a small amount of annotated data to perform cross-lingual lemmatization. The model projects the supervised lemmatization information in one language onto another language via a word-aligned text corpus. Kohonen et al. (2010a) perform semi-supervised morphological analysis. They first use the annotated data to extract a mapping between the segments and morphological labels. Then, the model segments the words and finally replaces each segment with a label according to the mapping. Yarowsky and Wicentowski (2000) bootstrap the learning of the partial English verb paradigm (relationships between present and past tense) from a small amount of annotated training data. Chater and Manning (2006) use LDA to reveal paradigms as latent classes in the stem-suffix matrix, assuming the correctly annotated morphological

[6] We are not aware of its adaptation into unsupervised setting.

[7] We do not conduct experiments with fully supervised learning though.

segmentations and POS information. Those works provide interesting bits and pieces to build on. However, the focus of the semi-supervised models in Chapter 5 is on morphological segmentation.

2.3.4 Evaluation methods

Different measures have been used to evaluate the results of unsupervised morphological segmentation or analysis systems; for an overview see Virpioja et al. (2011). Here we describe two measures used later in this dissertation: segment boundary F1-score and EMMA (Spiegler and Monson, 2010), which is specifically developed for evaluating the results of morphology learning systems.

Segmentation boundary F1-score

Segmentation boundary F1-score (SBF1) is one of the simplest and most widely used measures for evaluating morphological segmentations. It is the harmonic mean of the *precision* and *recall* of the correctly placed segment boundaries.

$$F1 = 2 \cdot \frac{P \cdot R}{P + R} \tag{2.1}$$

Precision P is the proportion of correctly placed segment boundaries with respect to all set segment boundaries and recall R is the percentage of correctly placed segment boundaries with respect to all correct segment boundaries. Precision and recall are expressed in terms of *true positives* (TP), *false positives* (FP) and *false negatives* (FN). True positives are the correctly set segment boundaries; false positives are the incorrectly placed segment boundaries, and false negatives are the missing correct segment boundaries.

$$P = \frac{TP}{TP + FP} \tag{2.2}$$

$$R = \frac{TP}{TP + FN} \tag{2.3}$$

Usually, the segment borders occurring at the beginning and the end of the word are ignored because they are given for free. The side effect of this is, however, that the correctly segmented monomorphemic words do not contribute positively to the overall score.

SBF1 can be computed using either *micro*-averaging (MI) or *macro*-averaging (MA). The MI score is the average of the individual word's F1 scores. The MA score uses the precision and recall computed over all words. In MI score, each word makes an equal contribution whereas the MA score is more influenced by the words with more segments. From another point of

Table 2.2: Examples of induced and correct segmentations of some English words.

Word	True	Induced
act	act	act
act	act_ed	acte_d
acting	act_ing	act_ing
looking	look_ing	look_ing
looked	look_ed	looke_d

Table 2.3: The best mapping between the induced and correct segments.

Induced		True
act	⇒	act
d	⇒	ed
ing	⇒	ing
look	⇒	look
looke	⇒	
acte	⇒	

view, each induced segment border makes a similar impact on the MA score while in the MI score, the induced borders in words with fewer segments have a stronger influence.

It may seem that MI alleviates the problem of evaluating the correctly segmented monomorphemic words because their contribution to the final score is equal to the contribution of any other word. However, in those words both TP, FP, and FN are 0 and the precision and recall would be effectively undefined. Of course, they could be defined to be 100% in these situations, and so also the SBF1 for those words would be 100%. However, the fact that the contribution of those words including no induced boundary is the same as that of the words with several correctly predicted boundaries makes this solution disputable, especially when considering that the monomorphemic words are usually short, and so their segmentations have fewer chances to contain incorrect boundaries.

EMMA

EMMA (Spiegler and Monson, 2010) addresses two of the issues the SBF1 has—correctly segmented single-morpheme words are reflected in the score, and, whereas SBF1 can only evaluate concatenative morphology, EMMA can evaluate non-concatenative morphology as well.

EMMA works by finding the best one-to-one mapping between the induced and reference segments. The induced segments are then replaced with their mapped actual segments and based on that F1-score is calculated. For instance, consider the induced and correct segmentations for some English words in Table 2.2. The best mapping between the reference and induced segments is given in Table 2.3. Because the number of different reference morphemes is smaller than the number of induced segments, two of the induced segments—*acte* and *looke*—will remain unmapped. Next, the induced segments are replaced with their mapped counterparts. Then, the number of TPs, FPs, and FNs can be counted and precision, recall and F-score computed as described in the previous section. F1 score can again

be computed by using either micro- or macro-averaging and depending on the choice, the result is slightly different.

EMMA mapping is found using Integer Linear Programming and its computational complexity is cubic in the number of segments in the evaluation set, making its computation quickly impractical when the evaluation set size increases. A modified measure EMMA-2 has been proposed in the literature (Virpioja et al., 2011) that, instead of 1-1 mapping, optimizes two many-to-one (M-1) mappings and combines them later. Computing M-1 mapping is much simpler problem reducing the overall complexity to $O(mn)$ where m and n are the induced and the reference segment set sizes respectively.

2.4 Part-of-speech tagging

Part-of-speech (POS) tagging is the task of labeling each word token in the corpus with its POS type, such as noun, verb or adjective. Linguistically, POS tagging is more related to syntactic processing and not considered as strictly part of morphology. However, the POS of a word determines what kind of prefixes or suffixes are possible, as well as the valid morphological categories and thus, POS tagging and morphology are very closely related. Morphological segmentation models frequently operate without any knowledge about the POS. However, when one wants to do morphological analysis, paradigm recovery, or word form generation from features, some knowledge about the POS is crucial.

There has been considerable work on supervised POS tagging, and the accuracies of those systems reach far over 90% (Toutanova et al., 2003; Shen et al., 2007). However, in this section we will concentrate on unsupervised approaches and elaborate on the main design decisions related to developing such systems.

2.4.1 POS disambiguation and POS induction

In unsupervised POS tagging, two settings are possible. The task is called *POS disambiguation* when a POS lexicon is available, (e.g. Merialdo (1994); Johnson (2007); Goldwater and Griffiths (2007)). A POS lexicon is a data structure that lists for each word type its possible synactic tags, and the task is then to choose the proper tag for each word in context. In this setting, the particular tagset is known, and there is no identifiability problem. POS tagging without a tag lexicon is called *POS induction*, (e.g. Van Gael et al. (2009); Ravi and Knight (2009); Ganchev et al. (2009); Berg-Kirkpatrick et al. (2010); Lee et al. (2010); Lamar et al. (2010); Christodoulopoulos et al. (2011); Blunsom and Cohn (2011)). The result of this task is a set of unnamed POS tags or clusters, and there is no straightforward way to map these clusters to linguistic POS tags without labeled data.

Both POS tagging models of this dissertation perform POS induction. The accuracies of POS induction are generally lower than those of POS disambiguation.[8] However, assuming the existence of a high-coverage POS lexicon that would list all possible POS tags for a large number of words is pretty ambitious even for a supervised learner. The lexicon coverage will especially be a problem for morphologically complex languages that have large active vocabularies. Some previous work have addressed this issue. For instance, Toutanova and Johnson (2008) use an existing lexicon to define ambiguity classes of POS tags and for each word missing from the lexicon, the model predicts the its ambiguity class, thus providing it a set of possible tags. This work was recently extended to POS induction by introducing an unsupervised lexicon containing ambiguity classes over latent tags (Dubbin and Blunsom, 2014).

2.4.2 Sequence models and clustering models

Models performing POS learning can be roughly divided into sequence models that attempt to find a likely sequence of POS tags for every sentence, and clustering models that organize words into hard or soft POS clusters.

Sequence models generally use a hidden Markov model (HMM) for generative modeling (Goldwater and Griffiths, 2007; Johnson, 2007; Van Gael et al., 2009), that may use features (Berg-Kirkpatrick et al., 2010; Lee et al., 2010) or log-linear modeling (Smith and Eisner, 2005; Haghighi and Klein, 2006). The advantage of the sequence models is that they have a direct access to the syntactic and lexical context of the words during learning. Also, they enable modeling ambiguous words because each word token in context can be treated separately.

The POS induction model in Chapter 4 is a sequence model using the structure of a trigram HMM incorporating morphological features. This model is type-based (see section 2.4.3) and hence does not resolve the ambiguities. However, adopting sequence modeling approach gives direct access to the token frequencies that can be used in estimating the model parameters.

Clustering models, (e.g., Clark (2003); Lamar et al. (2010); Christodoulopoulos et al. (2011)) on the other hand, usually do not have direct access to the contexts of the word tokens. In clustering models, the words are represented with distributional feature vectors that integrate the information about the different contexts each word type has occurred in. In previous work, these word representations have been compiled by counting the occurrences of context words. In the morphosyntactic clustering model in Chapter 6 we use a different word representation—the feature

[8] Comparing accuracies assumes that a mapping between the induced clusters and the linguistic tags can be constructed, see section 2.4.7.

vectors are trained with a neural network language model (Al-Rfou et al., 2013). These word embeddings have been shown to perform well as features for training a supervised POS tagger (Al-Rfou et al., 2013). Here we use them for clustering in an unsupervised POS induction model. Some recent work (Yatbaz et al., 2012; Yuret et al., 2014) also perform POS induction by clustering word embeddings. However, learning those embeddings is a crucial model-specific step in their algorithm while we use general-purpose pretrained word embeddings. Also, their POS induction performs simple K-means clustering while our model is a more flexible infinite Gaussian mixture model.

Hard clustering models, (e.g., Brown et al. (1992); Lamar et al. (2010)) assign to each word type a single label only. Soft clustering models or probabilistic clustering models (e.g., Clark (2000); Christodoulopoulos et al. (2011)) compute for each word type a full probability distribution over all clusters. The morphosyntactic clustering model presented in Chapter 6 is also a soft clustering model. Our tag clusters are Gaussian and for each word type, it is possible to compute the distribution of all clusters. However, during inference, we assign to each word type a single tag only.

2.4.3 Token-based and type-based POS tagging

POS tagging can be performed either *type-based* or *token-based*. The token-based approach assigns a tag to each word token in the running text while type-based models assume that every instance of the same word type has the same tag. A token-based approach allows different tags for ambiguous word types. For instance, the English word *book* might be used as a noun in one context and as a verb in another. However, allowing different word tokens of the same type to have a different tag comes with the cost of additional parameters in the model. On the other hand, type-based models are more constrained but do not enable solving ambiguous cases.

Sequence models performing POS disambiguation frequently operate by labeling tokens (Merialdo, 1994; Johnson, 2007; Goldwater and Griffiths, 2007), so do some sequence-based POS induction models (Van Gael et al., 2009; Ravi and Knight, 2009; Ganchev et al., 2009; Berg-Kirkpatrick et al., 2010), although the type-based approach for sequence-based POS induction has also become popular (Lee et al., 2010; Blunsom and Cohn, 2011). The type-based approach has been motivated by the fact that the accuracy of oracle experiments assigning to each word token its most frequent tag achieves over 90% across several languages (Lee et al., 2010), which is far above the current best results in unsupervised POS induction. We use the same motivation in Chapter 4 in the HMM-based POS induction model to constrain all word tokens of the same type with a single label only.

Clustering models typically operate on word types, but some models also cluster word tokens (Yuret et al., 2014). The morphosyntactic clustering model of Chapter 6 is type-based. We cluster distributional word vectors that encode information about all the token contexts where the word type has occurred in, but this token-based information is not explicitly available after the vectors have been trained.

2.4.4 Syntactic and morphosyntactic tags

The notion of POS tag is somewhat ambiguous. When looking at different corpora annotated with POS tags, the picture is quite incoherent. The Table 1 in (Petrov et al., 2012) shows the number of tags in various languages and corpora. A significant variation in these figures for some languages indicates that there are different labeling schemes encoding various levels of morphosyntax. In this dissertation, we work mainly with the Multext-East (MTE) corpus (Erjavec, 2004) that provides morphosyntactically labeled texts for several Eastern European languages. For instance, the MTE English part marks 11 syntactic tags and a total of 104 distinct sequences of morphological category values while the Wall Street Journal corpus (WSJ) (Marcus et al., 1993) tagset size is 45 and its coarser version contains 17 tags (Smith and Eisner, 2005). Also, for instance, the Czech CoNLL tagset[9] contains 63 tags, MTE again has 12 syntactic tags but almost 600 distinct sequences of morphological category values. These examples show that choosing the exact number of tags to model can be difficult, and the final choice may be somewhat arbitrary.

The approach taken in this dissertation is to differentiate between *coarse-grained* (syntactic) and *fine-grained* (morphosyntactic) POS tags. The coarse-grained tagset generally corresponds to the universal tagset described in (Petrov et al., 2012). In terms of the POS annotations in different corpora, the coarse-grained tag generally corresponds to the first character of the label, which determines the main syntactic class while the subsequent characters specify relevant morphological categories. The coarse-grained tagsets tend to consist of 10-14 tags depending on the language. The fine-grained tagsets may contain as few as 50 tags as the English WSJ corpus, or several hundreds of tags as the MTE corpus for morphologically complex Czech, Hungarian and Polish. For an example, the English coarse and fine-grained tags of MTE corpus are given in Table 2.4[10].

The models in this disseratation induce a variable number of POS tags and as such, they are not predetermined to learn a clustering conforming to either coarse-grained or fine-grained tagset. However, depending on the

[9] as of (Petrov et al., 2012)

[10] The descriptions of the fine-grained labels can be found in `http://nl.ijs.si/ME/Vault/V3/msd/html/msd.html`

Table 2.4: Universal tags of Petrov et al. (2012) and the English MTE tagsets.

Type	**Universal**	**Coarse**	**Fine**[a]
noun	NOUN	N	Nc, Nc-p, Nc-s, Ncfp, Ncfs, Ncmp, Ncms, Ncn, Ncnp, Ncns, Np, Np-p, Np-s, Npfs, Npms, Npns
verb	VERB	V	Vacs, Vaip-p, Vaip1s, Vaip2s,Vaip3s, Vais, Vais-p, Vais1s, Vais2s, Vais3s, Van, Vapp, Vaps, Vmcs, Vmip, Vmip-p, Vmip1s, Vmip2s, Vmip3s, Vmis, Vmis-p, Vmis1s, Vmis2s, Voip, Vmis3s, Vmn, Vmnp, Vmpp, Vmps
adjective	ADJ	A	Af, Afc, Afp, Afs
adverb	ADV	R	R-p-(3)q, Rmc, Rmp,Rmp-(3)q, Rmp-(3)r, Rsc, Rsp, Rss
pronoun	PRON	P	Pg3-(14)q, Pg3-(14)r, Pg3-p, Pg3-s, Pg3fs, Pg3ms, Pg3n, Pg3np, Pg3ns, Pp-(15)q, Pp-(15)r, Pp-(3)a-(11)q, Pp-(3)a-(11)r, Pp–pn, Pp–sn, Pp1-pa, Pp1-pn, Pp1-sn, Pp2, Pp2-p, Pp3-pa, Pp3fs, Pp3fsa, Pp3fsn, Pp3ms, Px1-p, Pp3msa, Pp3msn, Pp3ns, Ps3, Px3ns, Ps3-(14)q,Ps3-(14)r, Ps3-(3)p, Pt3, Ps3-(3)s, Ps3-(3)sf, Ps3-(3)sm, Px2-s, Px3-p, Px3-s, Px3fs, Px3ms
determiner	DET	D	Dd, Dg, Dg-(8)q, Dg-(8)r, Dg-(2)p, Dg-(2)s, Di, Ds-(8)q, Ds-(8)r, Ds1-(3)p, Ds1-(3)s, Ds2, Ds3-(3)p, Ds3-(3)sf, Ds3-(3)sm, Ds3-(3)sn
adposition[b]	ADP	S	Sp, St
numerals	NUM	M	Mc, Mo
conjunctions	CONJ	C	Cc, Cc-i, Cc-n, Cs
punctuation	.	X	X
others[c]	X	I[d]	I

[a]The number in the parentheses indicates the number of dashes in the label at that position; only one dash is spelled out for brevity, for instance Dg-(2)s stands for Dg--s.

[b]prepositions and postpositions

[c]Includes abbreviations, foreign words etc.

[d]interjections

model specifics, the number of clusters is either smaller or larger and thus the comparison with either coarse- or fine-grained tagset is more appropriate. The clusters of the POS induction model in Chapter 4 correspond more readily to the coarse-grained syntactic tags and some of the clusterings induced by the model in Chapter 6 are better interpreted as fine-grained morphosyntactic clusters.

2.4.5 Finite and infinite models

Finally, one must consider whether to perform POS induction with a finite (parametric) or an infinite (non-parametric) model. In the finite case, the number of desired POS labels or clusters is assumed be known. On the other hand, infinite models (Biemann, 2006; Van Gael et al., 2009) treat the number of clusters as a model parameter to be learned from the data.

As argued in section 2.4.4, the number of gold-standard tags can vary in different annotated reference corpora. Therefore, it is hard to state what the correct number of tags for each language should be, especially when attempting to learn fine-grained morphosyntactic tags. However, in previous work the common choice has been to learn finite models with the same number of POS tags as in that particular gold-standard corpus with reference annotations that was used for evaluation. Although the POS clusters found in an unsupervised manner indeed correlate with the linguistic syntactic clusters, they also capture other latent phenomena such as semantics and thus, perfect overlap with linguistic tags is probably impossible, even when the numbers of tags match.

The both POS induction models presented in this dissertation are infinite, allowing the model to infer the number of clusters from the data. This makes evaluation against the linguistic reference annotations harder due to the highly probable mismatch between the number of induced clusters and the number of tags in the reference corpus. However, the infinite models allow the emergence of the patterns inherent in the data more naturally and as demonstrated by Van Gael et al. (2009) and Biemann et al. (2007), these patterns can be validated through task-based evaluation.

2.4.6 Morphology in POS induction

The idea that morphology can constrain and guide the POS learning is intuitive. For instance, the English suffix *-ed* strongly suggests that the word is a past tense verb. Thus, a POS induction system can exploit such information to cluster together morphologically similar words.

Several methods have been proposed for incorporating morphology into POS induction systems. Clark (2003) modeled the morphological information with a class-based character HMM. His approach motivated also other researchers to use character n-gram features (Berg-Kirkpatrick et al., 2010;

Lee et al., 2010; Blunsom and Cohn, 2011) for predicting the POS. Hasan and Ng (2009) and Christodoulopoulos et al. (2011) used suffix features learned with an unsupervised morphological segmentation system during preprocessing.

We also use suffix features in our POS induction systems. However, these suffixes are learned jointly with the POS tags by the same model—in the joint POS induction and segmentation model of Chapter 4 via the full morphological segmentation and in the morphosyntactic clustering model in Chapter 6 as a log-linear model using suffix features of different length.

2.4.7 Evaluation methods

POS tagging is a standard component in NLP pipelines and rarely a target in itself. Therefore, it would be reasonable to evaluate its results via the respective downstream task. Especially in unsupervised POS induction this kind of *extrinsic* evaluation makes sense because there is no direct mapping between the induced labels and the reference POS tags. There have been attempts to evaluate POS induction results using shallow parsing task (Van Gael et al., 2009). Unsupervised POS induction has been used in word sense disambiguation and named entity recognition systems (Biemann et al., 2007) and for dependency parsing (Spitkovsky et al., 2011), so also these tasks could potentially be used for evaluating POS induction results.

The downside of the extrinsic evaluation is that the procedure itself may be rather complex and in some cases also quite time-consuming. Therefore, most of the times *intrinsic* assessment methods are used, which compare the induced labels with the annotated reference POS tags. Computing intrinsic measures is simple and quick compared to the extrinsic evaluation. However, those measures do not necessarily correlate with the results of the respective downstream task. For a further discussion of the issue see Vlachos (2011).

Intrinsic evaluation measures

Next, we describe four measures that have been commonly used for evaluating unsupervised POS induction results: one-to-one accuracy, many-to-one accuracy, V-measure and variation of information. These measures are used later in this dissertation to evaluate the POS induction results.

One-to-one accuracy (1-1) is based on the solution of the assignment problem (Munkres, 1957) by finding a 1-1 mapping between induced labels and true tags. Although the globally optimal mapping can be found in polynomial time, it has been more common in unsupervised POS evaluation to use the greedy 1-1 accuracy (Haghighi and Klein, 2006). Greedy mapping first maps the cluster and POS class having the largest intersection and then removes this cluster and class from the procedure. This process is continued until all the labels or tags, whichever number is smaller, are mapped. The

greedy nature of the algorithm can lead to unfavourable mappings in the last stages of the procedure when there are no better decisions available anymore. For instance, an induced cluster may be mapped to a real POS class that it shares no common elements with because all the other more suitable POS classes have been already mapped.

1-1 accuracy is very sensitive to the number of induced clusters. If the number of induced and reference clusters do not match then some clusters or POS classes will remain unmapped, and items belonging to those clusters or classes will contribute negatively to the accuracy score.

The POS induction models presented in this dissertation induce a variable number of clusters. Thus, the number of learned clusters cannot be ensured to be equal or even in the close range with the reference number of classes. However, 1-1 measure is still informative because in this situation it measures mainly the accuracy of the big clusters that are expected to correspond to the POS tags of the open-class words.

Many-to-one accuracy (M-1) (Johnson, 2007) is one of the most frequently reported measures in previous work on unsupervised POS induction. It computes the accuracy using the mapping between each induced cluster and its most overlapping gold standard class, whereas several clusters can be mapped to the same class. This measure is particularly sensitive to the number of induced clusters, reaching 100% accuracy when each word is put into a separate cluster.

M-1 accuracy usually scores higher than 1-1 and may do so even with the same number of induced and reference clusters. This behaviour indicates that some gold standard classes remain unmapped.

Due to the sensitivity to the number of clusters, M-1 is not the preferred measure for evaluating systems inducing a variable number of POS clusters as it is the case in this work. However, it will be reported as a secondary measure to enable some comparison with previously published systems.

V-measure (V-m) (Rosenberg and Hirschberg, 2007) is an information-theoretic score that compares two clusterings. It is the harmonic mean of *homogeneity* (h) and *completeness* (c) and varies between 0 and 100%.

$$\text{V-m} = \frac{2 \cdot h \cdot c}{h + c} \tag{2.4}$$

In the POS induction context, the induced clustering is compared to the gold standard POS classes. Homogeneity measures how many of those elements that are assigned to a cluster belong to a single POS class, and it reaches its maximum when the items in a cluster belong to a single reference class only. Homogeneity is computed using the normalized conditional entropy of the reference classes (C) given the induced clustering (K):

$$h = \begin{cases} 1 - \frac{H(C|K)}{H(C)} & \text{if } H(C) \neq 0 \\ 1 & \text{if } H(C) = 0 \end{cases} \tag{2.5}$$

Completeness is a symmetrical measure—it measures how many elements from the same POS class were assigned to the same induced cluster:

$$c = \begin{cases} 1 - \frac{H(K|C)}{H(K)} & \text{if } H(K) \neq 0 \\ 1 & \text{if } H(K) = 0 \end{cases} \tag{2.6}$$

Although V-m does not compute any explicit mapping between the induced and the reference clusters, it can still be somewhat sensitive to the number of clusters (Reichart and Rappoport, 2009) but is much less so than M-1 (Christodoulopoulos et al., 2010).

Variation of information (VI) (Meilă, 2002) is another information-theoretic measure that has been proposed for evaluating unsupervised POS induction results (Goldwater and Griffiths, 2007). It is the sum of the conditional entropies in both ways:

$$VI(C, K) = H(C|K) + H(K|C) \tag{2.7}$$

In case the two clusterings are equal the conditional entropies are zero because the conditioning clustering already tells everything about the other one. VI is upper bounded by the sum of the entropies of both clusterings because when the two clusterings are maximally dissimilar then $H(C|K) = H(C)$ and $H(K|C) = H(K)$. As the VI values do not lie between 0% and 100%, this measure is harder to interpret. Therefore, several normalization schemes have been proposed. Normalized VI (NVI) (Reichart and Rappoport, 2009) divides VI by the entropy of the reference clustering:

$$NVI(C, K) = \begin{cases} \frac{H(C|K)+H(K|C)}{H(C)} & \text{if } H(C) \neq 0 \\ H(K) & \text{if } H(C) = 0 \end{cases} \tag{2.8}$$

The idea is that the gold standard clustering is fixed, and thus its entropy provides a stable normalization constant that does not depend on different induced clusterings. The values of NVI do not lie strictly in the range of 0 and 1. However, "good" clusterings should have NVI less than one (Reichart and Rappoport, 2009) and in the case of matching clusterings $NVI = 0$.

Evaluating tokens vs types

All described measures can be used for both *token-based* and *type-based* evaluation. In token-based evaluation, the assigned label of each word in the running text will be compared to the true POS tag of the same token

in the reference corpus. This kind of evaluation penalizes the type-based models that label each word type with a single tag only, ignoring the possible ambiguities.

For performing type-based evaluation, a type-based reference lexicon has to be created. This is typically done by assigning each word type its majority label in the reference corpus. The hypothesized label of each word type is then compared to the reference label in the lexicon and based on that, the respective scores are computed.

Token-based evaluation measures take the token frequency into account. Thus, they mainly evaluate the accuracy of the frequent word types (e.g., the closed class function words). Type-based evaluation, on the other hand, gives equal weight to all word types, regardless of their token frequency. Thus, its scores reflect mostly the accuracy of the low-frequency words. Therefore, the type- and token-based measures reflect the accuracy of different parts of the vocabulary; for a further discussion see Reichart et al. (2010).

2.5 Conclusion

In this chapter we explained the complexity of the natural language morphology both from linguistic and computational point of view. We reviewed the state-of-the-art methods in both unsupervised and weakly-supervised morphological segmentation and unsupervised POS induction. This background will serve as a reference point for developing the segmentation and POS induction models presented in the next chapters.

Chapter 3

Non-parametric Bayesian modeling

This chapter is a short tutorial on the methods and techniques used in non-parametric Bayesian modeling. First, we give an overview of the Dirichlet process and the Pitman-Yor process—non-parametric prior distributions that can be used to generate power-law-shaped distributions similar to those found in natural language. Then we explain the Markov chain Monte Carlo methods used in this dissertation for posterior inference as exact inference in non-parametric Bayesian models is usually intractable.

3.1 Non-parametric prior distributions

The material in this section is based on (Teh et al., 2006; Teh, 2010).

When building models of natural language the exact size of the lexicon is often unknown. For instance, when learning morphological segmentations with an unsupervised model, the number of morphemes and their lexical forms are not known a priori. If Σ is the alphabet of a language, then we need to define a prior distribution over all strings in Σ^*. A parametric solution would be to define, for example, a geometric distribution dependent on the string length or a finite discrete distribution over all substrings occurring in the unlabeled training data. However, it is highly unlikely that all substrings are valid morphemes. Thus, care should be taken to ensure that the probability of most of those substrings would be close to zero.

Using a non-parametric prior distribution provides a more compact solution. Non-parametric distributions can be used to represent explicitly only those lexicon elements that have a non-zero probability in the posterior distribution. Hence, the support of the distribution is dynamically adapted according to the data. In what follows, we describe the Dirichlet process

and the Pitman-Yor process that can be used to model discrete distributions with varying support.

3.1.1 Dirichlet process

The Dirichlet process (DP) (Ferguson, 1973) is a stochastic process that generates random discrete distributions. DP is parameterized by a concentration parameter $\alpha > 0$ and a base distribution H:

$$G \sim \mathrm{DP}(\alpha, H) \tag{3.1}$$

The base distribution H may be either continuous or discrete, finite or infinite. In any case, the support of a distribution G generated from a DP is a subset of the support of H. The base distribution H itself can be viewed as the mean distribution of the DP. The concentration parameter α is inversely related to the variance. The larger is α the closer the generated distributions are to the base distribution while with small α-s sparse distributions arise.

Let $G \sim \mathrm{DP}(\alpha, H)$ and draw n data points $X = x_1, \ldots, x_n \sim G$. Since G is discrete, several x_i-s can have the same value, forming a partition (clustering) over X. We can say that DP generates random distributions over the partitions of X, and those distributions are Dirichlet-distributed.

The posterior distribution of G given the observations X is also a DP with updated parameters:

$$G|x_1, \ldots, x_n \sim DP\left(\alpha + n, \frac{\sum_{i=1}^{n} \delta_{x_i} + \alpha H}{\alpha + n}\right), \tag{3.2}$$

where δ_{x_i} is a point mass located at x_i.

If the base distribution H is discrete then the posterior predictive probability for a new point x_{n+1} having a value ϕ_k, with G marginalized out is:

$$P(x_{n+1} = \phi_k | x_1, \ldots, x_n; \alpha, H) = \frac{n_{\phi_k} + \alpha H(\phi_k)}{\alpha + n}, \tag{3.3}$$

where n_{ϕ_k} is the number of points in $x_1, \ldots, x_n$ having the value ϕ_k and $H(\phi_k)$ is the probability of the value ϕ_k according to the base measure. The derivations of (3.2) and (3.3) can be found in (Teh, 2010). In the case of continuous base distribution, ϕ_k is a parameter related to the kth cluster. Then we are computing the probability of x_{n+1} belonging to the cluster parameterized by ϕ_k and n_k is the number of points previously assigned to that cluster.

The posterior predictive distribution of DP provides equations for working with the partitions over the data points X having the same values. However, sometimes it is desirable to view the clustering of the data and the

values of the data points separately. For instance, in an infinite Gaussian mixture model (Rasmussen, 2000) the infinite prior is over the clusterings of the data and the likelihood deals with the particular values of each data point. Another stochastic process introduced in the next section—Chinese restaurant process—provides an alternative view of the DP and enables working with the clustering part only.

3.1.2 Chinese restaurant process

The Chinese restaurant process (CRP) (Aldous, 1985) is a stochastic process that can be described with the following metaphor. Imagine an infinitely big Chinese restaurant with infinitely many tables with each table having a capacity for infinitely many customers. In the beginning, the restaurant is empty. Then the customers, corresponding to data points, start entering one after another. The first customer chooses an empty table to sit at. Next customers choose an empty table with probability proportional to the concentration parameter α or sit into one of the already occupied tables with probability proportional to the number of customers already sitting there. The probability of the customer z_{n+1} sitting at the table t given the seating assignments of the previous customers $z_1, \ldots, z_n$ is:

$$P(z_{n+1} = t | z_1, \ldots, z_n; \alpha) = \begin{cases} \frac{n_t}{n+\alpha} & \text{if } t\text{th table is occupied} \\ \frac{\alpha}{n+\alpha} & \text{if } t \text{ is an empty table} \end{cases} \tag{3.4}$$

where n_t is the number of customers already sitting at the tth table.

The relation between the DP and the CRP is that the table arrangement $z_i, \ldots, z_n$ generated by a CRP corresponds to the partition over the indices of the data points $x_1, \ldots, x_n$ generated by a $G \sim$ DP. However, in DP each cluster ("table" in CRP) also has a value ϕ ("dish" served on that table) generated from the base distribution H, which is sampled whenever a customer chooses an empty table.[1] Thus, all customers subsequently sitting at that table will have the same value ϕ (in the discrete case) or are parameterized by the same ϕ (when the base distribution is continuous).

3.1.3 Pitman-Yor Chinese restaurant process

The Pitman-Yor process (PYP) (Pitman and Yor, 1997) is a two-parameter generalization of the DP that generates distributions with longer tails than DP. In PYP, H is the base distribution, $b > 0$ is the concentration parameter

[1]The same dish can be served on several tables only when the base distribution is discrete and finite because only then repeated draws of the same element can occur. In case the base distribution is continuous or infinite, the probability of generating the same element from the base distribution more than once, is zero. In such a case, each table in the restaurant serves a different dish.

as before, and $1 > a \geq 0$ is the smoothing parameter:

$$G \sim \text{PYP}(a, b, H) \tag{3.5}$$

The clusterings generated by a $G \sim$ PYP can be viewed again using the CRP metaphor. According to the PYP CRP, the probability of a customer z_{n+1} sitting at a table t, given the seating arrangements of the customers $z_1, \ldots, z_n$, is (Goldwater et al., 2011):

$$P(z_{n+1} = t | z_1, \ldots, z_n; a, b) = \begin{cases} \frac{n_t - a}{n+b} & \text{if } t \text{ is an occupied table} \\ \frac{ma+b}{n+b} & \text{if a new table is chosen} \end{cases} \tag{3.6}$$

Here, n_t is the number of customers sitting in the tth table as before and m is the total number of occupied tables. When $b = \alpha$ and $a = 0$, the original DP CRP is recovered.

As seen from (3.6), the generation of longer tail distributions is achieved by systematically removing some probability mass from the occupied tables and moving it to the pool of unoccupied tables. It has been shown (Teh, 2006a; Goldwater et al., 2011) that language models based on the PYP provide nearly equivalent estimates of word probabilities smoothing as modified Kneser-Ney (Chen and Goodman, 1998), which is considered nowadays to be one of the most successful smoothing techniques. This observation has made PYP attractive for using in different NLP tasks. The PYP has been applied for example to language modeling (Teh, 2006b; Goldwater et al., 2006b), word segmentation (Mochihashi et al., 2009), POS induction (Van Gael et al., 2009; Blunsom and Cohn, 2011), and grammar induction (Cohn et al., 2009).

3.1.4 Hierarchical models

The base distribution of a DP can itself be a DP, resulting in a hierarchical DP (HDP) (Teh et al., 2006).[2]

$$\begin{aligned} G_0 &\sim DP(\beta, H) \\ G &\sim DP(\alpha, G_0) \end{aligned} \tag{3.7}$$

The hierarchy need not stop with two levels. In principle, HDP-s with arbitrary depths can be constructed. In Chapter 4 we use an HDP to define an infinite trigram HMM for POS induction. Hierarchical models (using PYP) have been previously used for example in language modeling to represent the n-gram contexts of varying lengths (Teh, 2006a; Goldwater et al., 2006b; Wood et al., 2009) as well as for unsupervised POS induction (Blunsom and Cohn, 2011).

[2]Similarly, hierarchical models using PYP can be defined.

3.2 Inference with MCMC

The material in this section is based on (MacKay, 2003).

Inference in Bayesian models involves combining a prior distribution over the model parameters θ and the likelihood of the data $\mathcal{D}$ into the posterior distribution over the parameters according to the Bayes rule:

$$P(\theta|\mathcal{D}) = \frac{P(\mathcal{D}|\theta)P(\theta)}{P(\mathcal{D})} \tag{3.8}$$

$$P(\mathcal{D}) = \int_{\theta'} P(\mathcal{D}|\theta')P(\theta')d\theta' \tag{3.9}$$

Working with this posterior requires overcoming two main problems:

1. We usually can compute the terms in the numerator quite easily but calculating the marginal data likelihood $P(\mathcal{D})$ in the denominator is generally hard or intractable.

2. Often we do not know how to sample from the posterior analytically.

Markov chain Monte Carlo (MCMC) methods provide principled ways of working with posteriors suffering from those two problems. The general idea is to construct a Markov chain over a large number of states, each of which is a possible value of θ. We will use x to refer to a state of the Markov chain. Also, we need to specify a transition distribution $Q(x, x')$ that can be used to go from any state to any other state in the chain. If the chain is properly constructed, then its stationary distribution is guaranteed to be the desired posterior distribution. Then we can start from an arbitrary state and, after taking a random walk in the chain using the transition distribution long enough (*burn-in*), the chain will finally start to generate samples from the true posterior $P(x|\mathcal{D})$.

In order to construct a proper Markov chain, few conditions must be satisfied. First, the chain must be *ergodic*, which means that it is possible to reach any state from any other state and, in addition, the expected time for reaching any state from any other state is finite. This guarantees that a random walk in the Markov chain will eventually explore the whole state space (support) of the distribution. Second, the transition probabilities must satisfy the *detailed balance* property:

$$P(x|\mathcal{D})Q(x, x') = P(x'|\mathcal{D})Q(x', x) \tag{3.10}$$

This means that the probability of choosing a value x according to the target distribution P and then using the transition distribution Q to move to the

state with the value x' must be the same as choosing the value x' according to P and then moving to the state x via the transition distribution Q.

Each move in the Markov chain provides a sample from the posterior distribution. Of course, the samples generated in such a way are not independent as each sample is conditioned on the previous sample. For obtaining approximately independent samples, a number of interleaving sampling steps must be taken in order to limit the dependence of the previous sample.

There are a number of different MCMC methods. Here we describe two methods we use in our models: Gibbs sampling and Metropolis-Hastings sampling. These are widely applicable and often used methods in Bayesian inference. A thorough overview of other sampling methods can be found in (MacKay, 2003).

3.2.1 Gibbs sampling

Gibbs sampling is one of the most widely used MCMC methods. It can be used to generate samples from joint distributions of multiple random variables when the marginal conditional distribution for each variable given the values of all other variables can be computed.

For example, we may want to generate samples from the joint distribution over random variables x, y and z. Assume that we do not know how to do it analytically, but we know how to compute the conditional probabilities $P(x|y,z)$, $P(y|x,z)$ and $P(z|x,y)$. These conditional probabilities provide the transition distribution for the Markov chain whose states are all the possible combinations of the x, y and z values. We initialize the variable values randomly and start the random walk in the Markov chain. We sample a value for each random variable in turn from its conditional distribution, given the latest sampled values of other variables. One *Gibbs sweep* or iteration is performed when the values of all random variables are resampled. With the toy example introduced above and starting with the values $(x^{(0)}, y^{(0)}, z^{(0)})$ this means taking three sampling steps:

$$
\begin{aligned}
x^{(1)} &\sim P(x^{(1)}|y^{(0)}, z^{(0)}) \\
y^{(1)} &\sim P(y^{(1)}|x^{(1)}, z^{(0)}) \\
z^{(1)} &\sim P(z^{(1)}|x^{(1)}, y^{(1)})
\end{aligned}
$$

After every sampling step, there is a new configuration of the random variable values and thus a new sample from the joint distribution. Consecutive samples are apparently not independent because they are generated from the conditional distributions. However, by traversing the chain sufficiently long for burn-in and then taking samples by interleaving sufficient amount

of dependent samples, it is possible to simulate the behavior of taking independent samples from the true joint distribution.

3.2.2 Metropolis-Hastings sampling

Gibbs sampling is suitable when the conditional distributions are easy to construct—they either have an analytic form or they are discrete distributions with relatively small number of values that makes the normalization constant easy to compute. When these properties are not satisfied, then Gibbs sampling is not easily applicable. However, it is generally possible to use Metropolis-Hastings (MH) sampler instead.

MH sampling is another MCMC method employing a random walk in the sample space. However, instead of using conditional distributions for proceeding in the chain, MH sampler uses a *proposal distribution* as a transition distribution of the chain. The proposal distribution is constructed at each random walk step, and it depends on the current value of the chain. It is utilized for proposing samples, which are then accepted or rejected using the target distribution. The proposal distribution must have the same support as the target distribution and sampling from it should be relatively simple. The only requirement for the MH sampling to be applicable is the ability to compute unnormalized probabilities according to the target distribution for any sample drawn from the proposal distribution.

The general procedure is similar to Gibbs sampling—values of the random variables are resampled in turn. First a sample x' is generated from the proposal distribution, and then a threshold value is computed using the formula:

$$a = \frac{P^*(x')}{P^*(x^{(t)})} \frac{Q(x,' x^{(t)})}{Q(x^{(t)}, x')} \tag{3.11}$$

Here P^* is the unnormalized target probability, $x^{(t)}$ is the current value of the random variable, and Q is the transition probability according to the proposal distribution. If the threshold $a \geq 1$ then the proposed value is more probable than the current one and the current value is set equal to the proposed value: $x^{(t+1)} = x'$. If $a < 1$ then with probability a the proposed value is accepted, otherwise it is rejected and $x^{(t+1)} = x^{(t)}$.

Chapter 4

Joint POS induction and morphological segmentation

This chapter presents the joint unsupervised POS induction and morphological segmentation model using hierarchical Dirichlet processes. We provide experimental results in POS induction on 15 languages and segmentation results on four languages. The chapter is based on publication I (Sirts and Alumäe, 2012).

4.1 Introduction

Part-of-speech (POS) tagging and morphological segmentation are intuitively very closely related tasks—POS type determines which affixes are legal and affixes (or their lack) suggest the valid POS tag. Thus, it seems only natural to learn those tasks jointly so that the decisions of both tasks could influence each other. However, there have not been many attempts so far to implement this intuition into an unsupervised model. Most of the previous works have either treated those tasks separately or used one for merely guiding the other. For an overview about using morphology in POS induction see section 2.4.6.

In this chapter, we try to fill this gap by defining a joint model for unsupervised POS induction and morphological segmentation. We propose a model that constrains each word type with a single POS tag and segmentation, and infers the number of POS tags dynamically. The model has the structure of a hidden Markov model, which tags each word in context, but the type-based nature of the model constrains all word tokens of the same type to have the same tag. The type constraint prohibits certain kind of information flowing between POS tags and morphology. For instance, it ignores the fact that words can be ambiguous and belong simultaneously to several syntactic classes, having different morphological segmentations depending on the POS class. However, the primary goal of achieving information flow

between syntactic tags and morphology should still be obtainable without considering token ambiguity. Rather, taking it into account could make the task unnecessarily complicated.

Although the model learns full morphological segmentations of words, the tagging component in our model exploits only the suffix part of the segmentation. This is the standard approach in light of previous work, such as Lee et al. (2010) who use suffixes of fixed length and Christodoulopoulos et al. (2011) who obtain suffixes by pre-processing the words with Morfessor Categories-MAP (Creutz and Lagus, 2005), an unsupervised morphology induction system.

In the segmentation component of our model, each tag has a distribution over morphemes, whereas morphemes are conditionally independent given the tag. These distributions are smoothed with a common tag-independent distribution over segments. The smoothing distribution encourages reusing segments across different tags. Previous segmentation models using a similar setup include Goldwater et al. (2006b) and Lee et al. (2011). In both of these works, the number of tags or classes is predefined and small (6 and 5 respectively) and they only served to improve the segmentation and not evaluated as POS tags. The inference in those models is performed jointly over syntactic classes and segmentations, which is feasible due to the small number of tag classes and fixed segmentation structure. Goldwater et al. (2006b) splits words into stems and suffixes, and Lee et al. (2011) allows maximum five segments and in addition, applies several heuristics to reduce the sample space. In our model, the number of tags is unrestricted and can change during inference. The segmentation component allows learning segmentations with arbitrary number of splits. The joint search space can be thus very large and therefore we perform inference on tags and segmentations separately.

The only joint model with the goal of learning both POS tags and morphological segmentations we are aware of is Can (2011). This model is in several respects similar to what we propose. It learns both tags and segmentations jointly, and it induces a realistic (although predefined) number of tags. However, their tagging component does not make use of the morphological information, and the segmentation model is simple, splitting words into stems and suffixes only. Another main difference, aside from the fact that our model is non-parametric in terms of the number of tags, is that Can's model is token-based, enabling each word token of the same type to have a different tag and segmentation. The results are presented on English only, giving no indication of how the model would work on other languages. Also, Can's POS induction results are low on the WSJ corpus.

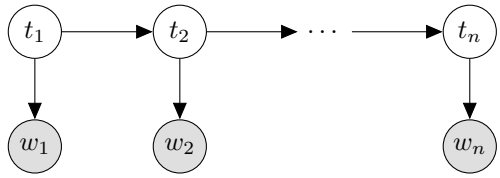

Figure 4.1: Standard hidden Markov model.

Our experimental results, conducted on 15 languages for the tagging component, compare favourably to the previously published results on unsupervised POS induction. The segmentation component is tested on four languages, and these results are unfortunately rather modest. However, more interesting than the numerical results, is assessing the joint learning aspect. We show via a set of semi-supervised experiments that there is interaction going on between the tagging and segmentation components. However, the experiments provide several cues, discussed later, indicating the model's failure to capture the interdependencies at the desired level and thus the main goal of this chapter will remain unsatisfied.

The structure of this chapter is as follows. We first describe the infinite hidden Markov model in section 4.2. Section 4.3 describes the model as it was presented by Sirts and Alumäe (2012), followed by the description of the experimental setup in section 4.4. Results of the experiments are presented in section 4.5. In the discussion section 4.6 we take a critical look at the model setup identifying several possible points of failure and propose potential fixes to these problems. Finally, in the conclusion, section 4.7, we review how the main results support the Claims of this dissertation.

4.2 Infinite hidden Markov model

This section first recaps the well-known hidden Markov model, which is one of the standard models for POS induction, then extends it into Bayesian domain by introducing prior distributions to the parameters (Goldwater and Griffiths, 2007) and finally turns it into an infinite HMM by replacing finite priors with infinite ones (Teh et al., 2006).

4.2.1 Hidden Markov model

The hidden Markov model (HMM) (Rabiner, 1989) is a graphical model for sequential data, see Figure 4.1. The Markov chain goes over latent tag variables t_i, each of which is dependent only on the previous latent variable t_{i-1}, and each tag emits an observable word w_i. If the number of latent tag values is T and the number of possible words is W, then the model parameters are: W-dimensional emission probability vectors $\psi_{k=1...T}$ and

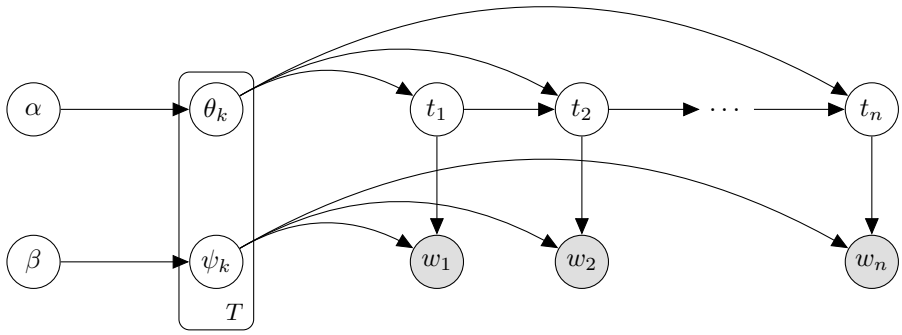

Figure 4.2: Bayesian hidden Markov model with symmetric Dirichlet priors.

T-dimensional transition probability vectors $\theta_{k=0\ldots T}$, where θ_0 covers the probabilities of choosing the value for the first latent tag in the sequence and for $k = 1 \ldots T$, θ_k is the transition distribution over tag values given that the previous tag is k. The sequence generation proceeds as follows:

$$\begin{aligned} t_1 &\sim \text{Multi}(\theta_0) \\ t_i | t_{i-1} = k &\sim \text{Multi}(\theta_k),\ i > 1 \\ w_i | t_i = k &\sim \text{Multi}(\psi_k), \end{aligned} \tag{4.1}$$

where Multi stands for the multinomial distribution.

The standard inference procedure for learning the HMM parameters is the forward-backward algorithm (Rabiner, 1989).

4.2.2 Bayesian hidden Markov model

Bayesian HMM extends the standard HMM by adding prior distributions to the model parameters θ_k and ψ_k, see Figure 4.2. The standard HMM with the forward-backward training treats each latent tag equally likely a priori and thus tends to learn tag clusters with relatively equal size. The Bayesian treatment, on the other hand, enables setting priors that prefer sparse posterior distributions, so that the transition and also the emission distributions are peaky, placing most of the probability mass on a few elements—a behaviour that more closely corresponds to the real properties of the language. A standard choice is to place conjugate symmetric Dirichlet priors over the multinomial parameters so that during inference the parameters can be integrated out (Goldwater and Griffiths, 2007). The model becomes:

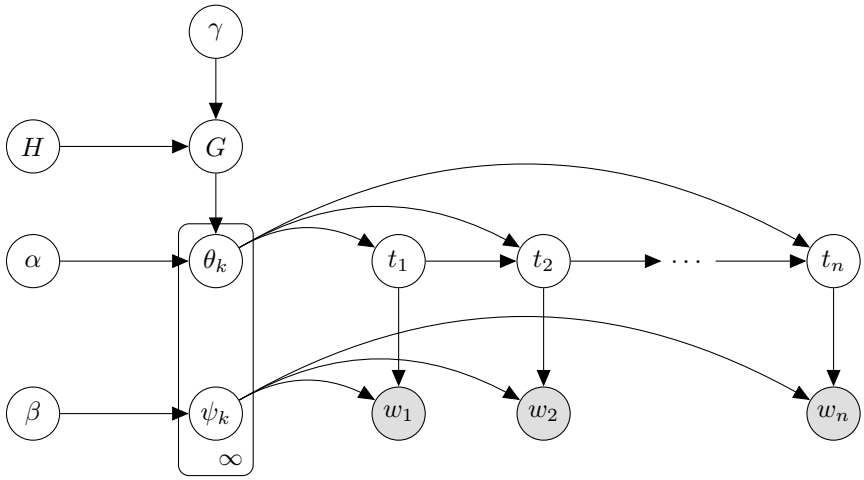

Figure 4.3: Infinite HMM with Dirichlet process prior distributions.

$$\begin{aligned} \theta_k &\sim \text{Dir}(\alpha), k = 0, \ldots, T \\ \psi_k &\sim \text{Dir}(\beta), k = 1, \ldots, T \\ t_1 &\sim \text{Multi}(\theta_0) \\ t_i | t_{i-1} = k &\sim \text{Multi}(\theta_k),\ i > 1 \\ w_i | t_i = k &\sim \text{Multi}(\psi_k), \end{aligned} \tag{4.2}$$

where Dir denotes the Dirichlet distribution, α and β are the respective Dirichet hyperparameters.

Inference with the Bayesian HMM can be done with Gibbs sampling (Goldwater and Griffiths, 2007) (see also section 3.2.1) or variational methods (Johnson, 2007).

4.2.3 Infinite hidden Markov model

Finally, the infinite HMM (Beal et al., 2002) can be constructed by replacing the Dirichlet priors of the transition and/or emission distributions with the Dirichlet process or Pitman-Yor process priors (described in sections 3.1.1 and 3.1.3 respectively), see Figure 4.3. The infinite HMM has two main advantages over the finite one. First, the non-parametric priors can be used hierarchically to implement complex smoothing schemes over a finite tagset (Blunsom and Cohn, 2011). Second, they can be used with a base distribution with infinite support, thus allowing the model to choose the number of latent values or tags dynamically based on the data.

The transition distributions must be coupled via the shared base distribution G. Otherwise, it would not be possible to ensure the transition distributions sharing the same support. The base distribution G is again generated by a DP with its own concentration parameter γ and with the base distribution H that can be a uniform distribution over infinitely many tags. In terms of the mathematical model, the parameter vectors are infinite-dimensional. However, in practice we always use CRPs (see section 3.1.2) to integrate over the infinite distributions G and thus, when performing inference using a finite amount of data we only have to deal with finite parameter vectors.

Mathematically, a model with infinite transition distributions and finite emission distributions is:

$$\begin{aligned} G &\sim \mathrm{DP}(\gamma, H) \\ \theta_k &\sim \mathrm{DP}(\alpha, G), k = 0, \ldots, \infty \\ \psi_k &\sim \mathrm{Dir}(\beta), k = 1, \ldots, \infty \\ t_1 &\sim \mathrm{Multi}(\theta_0) \\ t_i | t_{i-1} = k &\sim \mathrm{Multi}(\theta_k),\ i > 1 \\ w_i | t_i = k &\sim \mathrm{Multi}(\psi_k) \end{aligned} \tag{4.3}$$

4.3 Joint model for POS induction and morphological segmentation

We consider the problem of unsupervised POS induction and morphological segmentation in a joint model. The model is type-based, allowing each word type a single POS tag and segmentation only. Unlike most of the recent POS induction models, we do not assume any prior information about the number of POS tags. Rather, we let the model infer the number of tags from data using non-parametric prior distributions. Other than that, it has a familiar HMM structure, generating latent sequences of tags in order and emitting observable words as well as latent morphological segmentations. Previous work using non-parametric prior distributions for POS induction includes Van Gael et al. (2009); however their model is token-based. Also, their inference strategy is different—rather than resampling the tags from the collapsed posteriors using CRPs as we do, they explicitly resample model parameters as well. Another unsupervised POS induction model using hierarchical PYP priors is Blunsom and Cohn (2011). However, they operate with a fixed number of tags by using a finite-dimensional base distribution and use the PYP for smoothing purposes only.

4.3.1 Model formulation

The model (Sirts and Alumäe, 2012)[1] has four components:

1. A lexicon, which for each word type specifies its tag and morphological segmentation;
2. A trigram transition component using an infinite HMM;
3. An emission component emitting indices to the lexicon;
4. A segmentation component generating segmentations for word types.

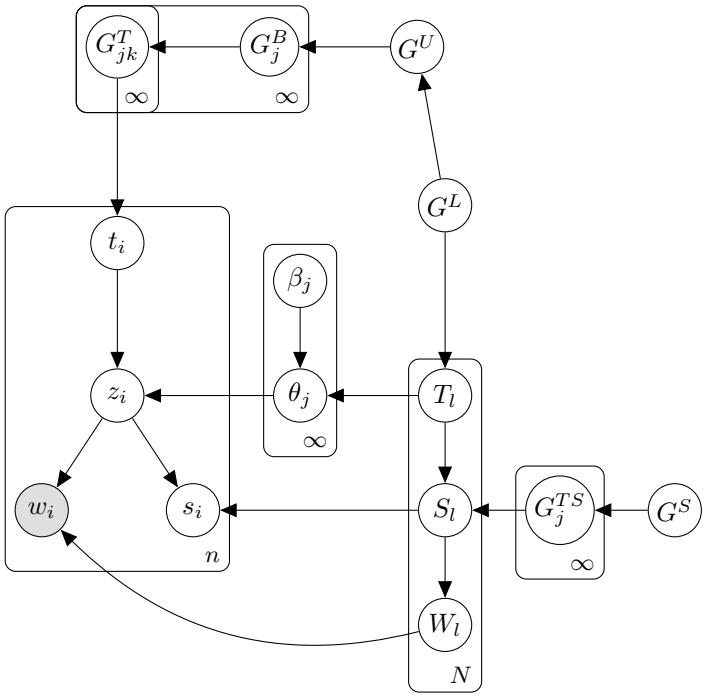

Figure 4.4: Plate diagram of the model. T_l, W_l and S_l denote the tag, the word and the segmentation of the lth lexical item. t_i, w_i and s_i denote, respectively, the tag, the word and the segmentation of each word token. z_i is the index of the generated token to the lexicon. Gs are the distributions generated from the various DPs, θ_j and β_j specify the tag-specific emission distribution and its Dirichlet prior hyperparameter. DP concentration parameters and the non-DP base distributions are omitted. Tag tokens t_i are presented compactly on a plate, but there are actually trigram dependencies between the consecutive t_is.

A graphical depiction of the model is given in Figure 4.4 and the mathematical formulation is given in Figure 4.5. The notations are explained in Figure 4.6.

[1]The presentation of the model has been slightly changed for the sake of clarity.

Lexicon generation	
Tags	**Segments**
$\alpha^L \sim \text{Gamma}(a, b)$	$p_l \sim \text{Beta}(1, 1)$
$H^L = \text{Uniform}$	$p_\# \sim \text{Beta}(1, 1)$
$G^L \sim \text{DP}(\alpha^L, H^L)$	$\beta_c \sim \text{Gamma}(a, b)$
$T_l \sim G^L,\ l = 1 \dots N$	$l_s \vert B \sim \text{Geometric}(p_\#)$
	$\theta_c \sim \text{Dir}(\beta_c)$
	$c \vert C \sim \text{Multi}(\theta_c)$
	$P_S(s) = \prod_{k=1}^{l_s} P_C(s_k) \times P_B(s_l)$
	$\alpha^S \sim \text{Gamma}(a, b)$
	$\alpha^{ST} \sim \text{Gamma}(a, b)$
	$G^S \sim \text{DP}(\alpha^S, S)$
	$G_k^{ST} \sim \text{DP}(\alpha^{ST}, G^S),\ k = 1 \dots \infty$
	$m_l \sim \text{Geometric}(p_l),\ l = 1 \dots N$
	$S_{lj} \vert T_l = k \sim G_k^{ST},\ \forall l : j = 1 \dots m_l$

Corpus generation	
Transitions	**Emissions**
$\alpha^U \sim \text{Gamma}(a, b)$	$\beta_j \sim \text{Gamma}(a, b),\ j = 1 \dots \infty$
$\alpha^B \sim \text{Gamma}(a, b)$	$\theta_j \sim \text{Dir}(\beta_j),\ \forall j$
$\alpha^T \sim \text{Gamma}(a, b)$	$z_i \vert t_i = j \sim \text{Multi}(\theta_j), i = 1 \dots n$
$G^U \sim \text{DP}(\alpha^U, G^L)$	$z_i = l \Rightarrow w_i = W_l$
$G_j^B \sim \text{DP}(\alpha^B, G^U),\ j = 1 \dots \infty$	$z_i = l \Rightarrow s_i = S_l$
$G_{jk}^T \sim \text{DP}(\alpha^T, G_j^B),\ \forall j : k = 1 \dots \infty$	
$t_i \vert t_{i-1} = j, t_{i-2} = k \sim G_{jk}^T,\ i = 1 \dots n$	

Figure 4.5: Generative story of the model.

The lexicon generation is pretty straightforward. The tags T_l are generated from G^L, which is drawn from a DP with a uniform base measure H^L over natural numbers. H^L is improper because the probability of any specific value will be zero. However, it can still be used to generate unique cluster labels denoting each tag.

$$G^L \sim \text{DP}(\alpha^L, H^L) \tag{4.4}$$

$$T_l \sim G^L, \quad l = 1 \dots N, \tag{4.5}$$

where N is the total number of word types in the lexicon.

Then, the number of segments m_l for each word type from a geometric distribution is generated and then the segments themselves are generated from the tag-dependent distributions G_k^{ST}.[2] Finally, words are obtained by simply concatenating the segments.

[2]Formally, there are infinitely many distributions G_k^{TS}, one for each of the infinitely many tags. However, the maximum number of tag-dependent segmentation distributions ever used is N, in which case each word type in the lexicon has a different tag.

α^L	Lexicon DP concentration parameter
H^L	Uniform distribution over infinite tagset
G^L	Distribution over tags in the lexicon
p_l	Parameter controlling the number of segments in words
m_l	Number of segments in the lth word type
$p_\#$	Parameter controlling the length of the segments
l_s	Length of the segment s
B	Geometric distribution over segment lengths
β_c	Concentration parameter for character Dirichlet prior
θ_c	Parameters of the multinomial distribution over characters
C	Multinomial distribution over characters
S	Base distribution over segments
α^S	Segment DP concentration parameter
α^{ST}	Tag-conditioned segment DP concentration parameter
G^S	Distribution over segments
G_k^{ST}	Distribution over segments conditioned on the kth tag
T_l	Tag of the lth word type
S_l	Segmentation of the lth word type
S_{lj}	jth segment of the lth word type
W_l	lth word type
α^U	Unigram transition DP concentration parameter
α^B	Bigram transition DP concentration parameter
α^T	Trigram transition DP concentration parameter
G^U	Unigram transition distribution
G_j^B	Bigram transition distribution conditioned on the tag j
G_{jk}^T	Trigram transition distribution conditioned on tags j and k
t_i	Tag of the ith word token
z_i	Word type index in the lexicon of the ith token
w_i	ith word token
s_i	Segmentation of the ith word token
β_j	Concentration parameter for the Dirichlet prior of the emission distribution of the tag j
θ_j	Parameters for the emission distribution of the tag j
N	Number of word types
n	Number of word tokens

Figure 4.6: Notation explanations.

$$G_k^{ST} \sim \mathrm{DP}(\alpha^{ST}, G^S), \quad k = 1, \ldots, \infty \tag{4.6}$$

$$m_l \sim \mathrm{Geometric}(p_l), \quad l = 1, \ldots, N \tag{4.7}$$

$$S_{lj}|T_l = k \sim G_k^{ST}, \quad l = 1, \ldots, N, \quad j = 1, \ldots, m_l \tag{4.8}$$

$$W_l = S_{l1} \ldots S_{lm_l} \tag{4.9}$$

The tag-dependent segment distributions G_k^{ST} have a common unigram segment distribution $G^S \sim \mathrm{DP}(\alpha^S, S)$ as base distribution. Its base distribution S combines the geometric distribution over segment lengths l_s and the unigram character probabilities according to a multinomial distribution with symmetric Dirichlet prior:

$$l_s|B \sim \mathrm{Geometric}(p_\#) \tag{4.10}$$
$$\theta_c \sim \mathrm{Dir}(\beta_c) \tag{4.11}$$
$$c|C \sim \mathrm{Multi}(\theta_c) \tag{4.12}$$
$$P_S(s) = P_B(l_s) \prod_{i=1}^{l_s} P_C(s_i) \tag{4.13}$$
$$G^S \sim \mathrm{DP}(\alpha^S, S) \tag{4.14}$$

The corpus is generated according to the standard HMM generation procedure from the hierarchical DP implementing an infinite trigram HMM:

$$G^U \sim \mathrm{DP}(\alpha^U, G^L) \tag{4.15}$$
$$G_j^B \sim \mathrm{DP}(\alpha^B, G^U), \quad j = 1, \ldots, \infty \tag{4.16}$$
$$G_{jk}^T \sim \mathrm{DP}(\alpha^T, G_j^B), \quad \forall j : k = 1, \ldots, \infty \tag{4.17}$$
$$t_i|t_{i-1} = j, t_{i-2} = k \sim G_{jk}^T, \quad i = 1, \ldots, n, \tag{4.18}$$

where n is the total number of word tokens. The base distribution of the unigram transition distribution G^U is the distribution over tags in the lexicon G^L because this enables the HMM procedure to generate tags from the same set that was already generated in the lexicon.

Each tag token t_i emits an index z_i to the lexicon from the respective tag-dependent emission distribution, which are multinomials with Dirichlet prior distributions. The required sparsity varies for different tags and hence each emission distribution has its own symmetric Dirichlet hyperparameter β_j.

$$\theta_j \sim \mathrm{Dir}(\beta_j), \quad j = 1, \ldots, \infty \tag{4.19}$$
$$z_i|t_i = j \sim \mathrm{Multi}(\theta_j) \tag{4.20}$$

As the model is type-based, the emission distributions should place probability mass only on those word types that were assigned the respective tag in the lexicon. However, the N-dimensional distributions generated from the symmetric Dirichlet priors place some probability mass on every word

type in the lexicon. In order to prevent generating illegal word tokens, the emission distributions are conditioned on the lexicon. The lexicon informs which tags can generate which words and forces the probabilities of all other word types to zero.[3]

$$P(z_i = l | t_i = j) = \begin{cases} \theta_{jl} & \text{if } T_l = j \\ 0 & \text{if } T_l \neq j \end{cases} \tag{4.21}$$

This solution is by no means elegant[4] but causes no major problems during inference, except that the emission probabilities are slightly underestimated. A formally correct solution would be the one adopted by Lee et al. (2010) who use emission distributions with different dimensionality for each tag, the dimensions corresponding only to those words in the lexicon that have the respective tag. Setting the probabilities of the illegal positions in θ_js to zero and renormalizing leads essentially to the same solution. This solution works well with the finite models but causes certain problems with infinite ones, the reasons of which we explain later in section 4.6.

Finally, the word tokens w_i and segmentations s_i are emitted deterministically via the lexicon lookup.

$$z_i = l \Rightarrow \begin{cases} w_i = W_l \\ s_i = S_l \end{cases} \tag{4.22}$$

All the DP and Dirichlet concentration parameters are generated from vague Gamma priors. Parameters for geometric distributions controlling the number and length of the segments are generated from uniform Beta priors.

4.3.2 Inference

Inference involves computing the posterior:

$$P(\mathbf{T}, \mathbf{S}, \mathbf{t}, \mathbf{s}, \boldsymbol{\Theta} | \mathbf{W}, \mathbf{w}, \boldsymbol{\Psi}), \tag{4.23}$$

where $\boldsymbol{\Theta}$ includes all model parameters and $\boldsymbol{\Psi}$ encompasses all hyperparameters. As usual, computing this posterior is intractable, so we use a Gibbs sampler to perform inference.

We resample tags and segmentations separately in turn. When resampling the tag of a word type, its segmentation is kept fixed, and during segmentation resampling the tag is fixed. Computing the joint posterior over the tag and

[3] We can imagine an iterative stochastic process generating the word tokens given the tag. Illegal words are discarded and the generation proceeds until a suitable word is sampled.

[4] The resulting emission distributions are improper because the probability mass does not sum to one.

the segmentation of a word type is not totally infeasible because at each moment the number of tags in the state space is finite, and the words have a fixed length. However, separate sampling reduces the amount of computation considerably, although it probably comes with some cost to the sampler mixing speed.

The model is type-based, and hence we use a blocked sampler, resampling the tags and the segmentations of the word tokens related to each word type at once. The sampler is collapsed, integrating out the random measures generated from the DPs by using the CRP representation. The whole inference procedure alternates between three sampling steps—tags, segmentations and non-collapsed parameters—which are described in detail in the following sections.

4.3.3 Tag resampling

Tags are sampled from the posterior:

$$P(\mathbf{T}, \mathbf{t}|\mathbf{W}, \mathbf{w}, \mathbf{S}, \mathbf{s}, \boldsymbol{\Theta}_{nc}), \tag{4.24}$$

where $\boldsymbol{\Theta}_{nc}$ denotes the set of non-collapsed model parameters. For a single word type, this posterior can be factored as follows:

$$\begin{aligned}
&P(T_l = k, \mathbf{t}_{\mathbf{i}_l}|\mathbf{T}_{-l}, \mathbf{t}_{-\mathbf{i}_l}, \mathbf{S}, \mathbf{s}, \mathbf{W}, \mathbf{w}, \boldsymbol{\Theta}_{nc}) && (4.25)\\
&\quad \propto P(\mathbf{S}, \mathbf{s}, \mathbf{W}, \mathbf{w}|T_l = k, \mathbf{T}_{-l}, \mathbf{t}_{\mathbf{i}_l}, \mathbf{t}_{-\mathbf{i}_l}, \boldsymbol{\Theta}_{nc}) && (4.26)\\
&\quad \times P(T_l = k, \mathbf{t}_{\mathbf{i}_l}|\mathbf{T}_{-l}, \mathbf{t}_{-\mathbf{i}_l}, \boldsymbol{\Theta}_{nc}) && (4.27)\\
&\quad \propto P(S_l, \mathbf{s}_{\mathbf{i}_l}|T_l = k, \mathbf{T}_{-l}, \mathbf{S}_{-l}, \mathbf{s}_{-\mathbf{i}_l}, \boldsymbol{\Theta}_{nc}) && (4.28)\\
&\quad \times P(\mathbf{w}_{\mathbf{i}_l}|T_l = k, \mathbf{T}_{-l}, \mathbf{W}, \mathbf{w}_{-\mathbf{i}_l}, \boldsymbol{\Theta}_{nc}) && (4.29)\\
&\quad \times P(\mathbf{t}_{\mathbf{i}_l}|T_l = k, \mathbf{t}_{-\mathbf{i}_l}, \boldsymbol{\Theta}_{nc}) && (4.30)\\
&\quad \times P(T_l = k|\mathbf{T}_{-l}, \boldsymbol{\Theta}_{nc}) && (4.31)
\end{aligned}$$

The subscript $-l$ denotes the lexicon with the lth word type excluded. The subscript $\mathbf{i}_l = \{i : z_i = l\}$ denotes the set of token indices for which the lexicon index variable $z_i = l$. Minus before the latter subscript means that this set of tokens is excluded from the set of all tokens.

Segmentation probabilities

The term in (4.28) is the segmentation likelihood and can be computed according to the CRP formula. Instead of using the full segmentation we compute the suffix likelihood only, taking as suffix the last segment in the segmentation.[5] If the number of tokens in the corpus corresponding to the

[5]If the segmentation consists of a single morpheme only then the likelihood of this single segment is computed.

lth word type is n_l, F_l is the suffix of the lth word type and $\mathbf{f}_{\mathbf{i}_l}$ are the suffixes of the corresponding tokens, the segmentation likelihood is:

$$\begin{aligned}
&P(S_l, \mathbf{s}_{\mathbf{i}_l} | T_l = k, \mathbf{T}_{-l}, \mathbf{S}_{-l}, \mathbf{s}_{-\mathbf{i}_l}, \boldsymbol{\Theta}_{nc}) \\
&\propto P(F_l, \mathbf{f}_{\mathbf{i}_l} | T_l = k, \mathbf{T}_{-l}, \mathbf{S}_{-l}, \mathbf{s}_{-\mathbf{i}_l}, \boldsymbol{\Theta}_{nc}) \\
&= \prod_{j=0}^{n_l - 1} \left(\frac{n_{kF_l}^{-\mathbf{s}_{\mathbf{i}_l}} + j}{n_{k\cdot}^{-\mathbf{s}_{\mathbf{i}_l}} + j + \alpha^{ST}} + \frac{\alpha^{ST}}{(n_{k\cdot}^{-\mathbf{s}_{\mathbf{i}_l}} + j + \alpha^{ST})} \frac{(m_{F_l}^{-\mathbf{s}_{\mathbf{i}_l}} + \alpha^S P_S(F_l))}{(m_{\cdot}^{-\mathbf{s}_{\mathbf{i}_l}} + \alpha^S)} \right),
\end{aligned} \tag{4.32}$$

where the product is over the token count, n_{kF_l} and m_{F_l} denote the number of customers "eating" the suffix F_l under the tag k and the number of tables "serving" the suffix F_l across all restaurants respectively, dot represents the marginal counts and $P_S(F_l)$ is the suffix probability according to the base distribution as described in the previous section. $-\mathbf{s}_{\mathbf{i}_l}$ in the upper index means that the segments of the word tokens with indices $\{i : z_i = l\}$ have been excluded.

Word emission probabilities

The term in (4.29) models the word token emissions. Due to the blocked sampler, the word token count $n_l^{-\mathbf{w}_{\mathbf{i}_l}}$ is zero after removing the word type from the state space. Therefore, the collapsed multinomial-Dirichlet probability takes the form:

$$P(\mathbf{w}_{\mathbf{i}_l} | T_l = k, \mathbf{T}_{-l}, \mathbf{W}, \mathbf{w}_{-\mathbf{i}_l}, \boldsymbol{\Theta}_{nc}) = \prod_{j=0}^{n_l - 1} \frac{j + \beta_k}{n_{k\cdot}^{-\mathbf{w}_{\mathbf{i}_l}} + j + N\beta_k} \tag{4.33}$$

Here, $n_{k\cdot}^{-\mathbf{w}_{\mathbf{i}_l}} = n_{k\cdot} - n_l$ denotes the number of word tokens having the tag k excluding the n_l word tokens corresponding to the l-th word type.

Transition probabilities

The term in (4.30) covers trigram transition probabilities. Trigrams relevant to a word type are those three trigrams associated with the particular word type in all token contexts, excluding any duplications.[6]

$$\begin{aligned}
P(\mathbf{t}_{\mathbf{i}_l} | T_l = k, \mathbf{t}_{-\mathbf{i}_l}, \boldsymbol{\Theta}_{nc}) = \prod_{i \in \mathbf{i}_l} (&P(t_i = k | t_{i-1}, t_{i-2}, \mathbf{t}_{-\mathbf{i}_l}, \boldsymbol{\Theta}_{nc}) \\
&\cdot P(t_{i+1} | t_i = k, t_{i-1}, \mathbf{t}_{-\mathbf{i}_l}, \boldsymbol{\Theta}_{nc}) P(t_{i+2} | t_{i+1}, t_i = k, \mathbf{t}_{-\mathbf{i}_l}, \boldsymbol{\Theta}_{nc}))
\end{aligned} \tag{4.34}$$

[6]For instance, if two consecutive tokens are of the same type then they share two common context trigrams.

All these terms can be calculated using CRP formulas. The probability of a single tag trigram r, s, k is:

$$P(t_i = k | t_{i-1} = s, t_{i-2} = r) = \frac{n^T_{rsk}}{n^T_{rs\cdot} + \alpha^T} + \frac{\alpha^T}{n^T_{rs\cdot} + \alpha^T} \left(\frac{n^B_{sk}}{n^B_{s\cdot} + \alpha^B} + \frac{\alpha^B}{n^B_{s\cdot} + \alpha^B} \left(\frac{n^U_k + \alpha^U G^L(k)}{n^U_\cdot + \alpha^U} \right) \right) \tag{4.35}$$

Here, n^T, n^B, and n^U are the counts of trigrams, bigrams and unigrams in respective CRPs.

The subscript $\mathbf{i}_l$ in equation (4.34) again denotes the set of token indices corresponding to the lth lexicon type. However, the subscript $-\mathbf{i}_l$, which denotes the exclusion of this set of tokens from the set of all tokens is slightly abused. The product in (4.34) goes over all tokens of the lth type and over all three context trigrams for each token. Each transition probability in the product is computed according to the equation (4.35) and is dependent on the tag counts of all the other tokens in the state space. When we start to compute (4.34) then the first trigram probability is dependent on the tag counts of the tokens $\mathbf{t}_{-\mathbf{i}_l}$. However, the probabilities of the next trigrams are dependent on the tag counts of the tokens $\mathbf{t}_{-\mathbf{i}_l}$ plus the counts of all the trigrams whose probabilities have been already computed. So we must on the fly cache the counts of all processed trigrams. The problem is that in terms of CRP terminology, we do not know which table those trigrams sit at. Thus, we would have to sum over all possible table arrangements of the trigrams (and their smoothing bigrams and unigrams), which is intractable.

The approximate solution we use is simple. After computing the transition probability for a trigram, we temporarily add its count to the respective CRP cache. While doing so, we decide randomly based on the previous counts in that CRP and the concentration parameter, whether the customer will stay on that HDP level or sit at an empty table causing a customer entering also into the smoothing CRP. So, instead of summing over all possible table configurations we randomly pick just one, which makes the Gibbs sampler biased. Another biased solution was presented by Blunsom and Cohn (2011) who proposed a scheme for approximating the expected table counts of each restaurant in the hierarchy. Their analysis showed that their approximation is relatively accurate when the number of word tokens sampled together is small. However, with increasing block sizes the approximation tends to underestimate the number of tables by quite a large margin. Their model uses a PYP hierarchy that is more sensitive to the number of tables than the HDP used in this work. Although we have not done a comparative analysis with the approximation we used, we expect its influence to the final result to be minor. However, we return to this issue

later in section 4.6 and sketch a Metropolis-Hastings step within the Gibbs sampler to correct the sampler bias.

Tag prior probability

The term in (4.31) covers the prior probability of the word type tag in the lexicon. The influence of this term is minor compared to the magnitude of the token transition probabilities, and its main role is to determine the set of tags that can be used in token sequence generation. Thus, in the implementation we will omit this term and use the improper uniform lexicon base measure H^L instead of G^L as the base distribution for the unigram DP.

4.3.4 Segmentation resampling

The number of different segmentations is exponential in the word length. As all words have a finite length, and most words are relatively short, it would not be completely infeasible to enumerate all possible segmentations. However, as the number of possible segmentations in longer words can still be quite large, we adopt a blocked sampler based on dynamic programming, similar to the one used by Mochihashi et al. (2009) for word segmentation. The dynamic programming approach serves as a proposal distribution for a Metropolis-Hastings sampler (see section 3.2.2). It is used to propose the next sample, which will be accepted or rejected using the true segmentation distribution.

For proposing a segmentation for a word type, we construct a forward table containing variables $\alpha[t][k]$ that present the probabilities of the last k characters of a t-character string constituting a segment. Define:

$$\alpha[0][0] = 1 \tag{4.36}$$

$$\alpha[t][0] = 0, \quad t > 0 \tag{4.37}$$

Then the forward variables can be computed recursively:

$$\alpha[t][k] = p(c_{t-k}^{t}) \sum_{j=0}^{t-k} \alpha[t-k][j], \quad t = 1 \dots l_w, \tag{4.38}$$

where c_m^n denotes the characters $c_m \dots c_n$ of a string c and l_w is the length of the word. The probabilities of the potential segments $p(s)$, conditioned on the tag currently assigned to the word type, are calculated according to the formula (4.32), by replacing the suffix F_l with the segment $s = c_{t-k}^{t}$.

Sampling starts from the end of the word because it is known for certain that the word end coincides with the end of a segment. Sample the beginning position k of the last segment from the forward variables $\alpha[t][k]$, where t

is the length of the word. Then set $t := t - k$ and continue to sample the beginning index of the previous to the last segment. This process continues until $t = 0$.

The dynamic programming approach assumes that the segments are conditionally independent given the tag. One way of constructing such a distribution would be to take a snapshot of the CRP caches with the counts related to the current word's segmentation excluded. This approach would be suitable when the segmentation probabilities would be computed using the word type counts because then the proposal distribution would not be that different from the true one. However, taking the token frequencies into account, this proposal can deviate a lot from the true probability, especially when the token count is high. Therefore, we change the proposal distribution during forward table construction by temporarily caching all counts of the encountered segment tokens. Although this solution is not very clean, we believe the resulting probabilities are closer to the true distribution, especially for high-frequency words. If $P(S_{\text{prop}})$ is the true probability of the proposed segmentation and $P(S_{\text{old}})$ is the true probability of the old segmentation, then the proposal will be accepted with the probability $\min(1, \frac{P(S_{\text{prop}})}{P(S_{\text{old}})})$, with the acceptance rate varying during experiments between 94-98%.

Note that according to the model, the segmentations are generated in the lexicon because of the type constraint, but the segment distributions themselves are estimated based on the segmentations of the word tokens. Estimating type-based segmentations using token frequencies is common in morphological segmentation systems. For instance, Morfessor systems (Creutz and Lagus, 2005) are all type-based but use token frequencies.

4.3.5 Hyperparameter resampling

All DP and Dirichlet concentration parameters are given vague Gamma(10, 0.1) priors, and their values are resampled by using the auxiliary variable sampling scheme described by Escobar and West (1995) and the extended version for HDPs described by Teh et al. (2006). This procedure uses the customer and table counts to update the Gamma prior into Gamma posterior, which is easy to sample from. The geometric distribution parameters are given uniform Beta priors, and their values are resampled from the posteriors, which are also Beta distributions.

4.4 Experiments

In this section, we describe the experimental setup, including the text corpora used for conducting experiments and evaluation methods for assessing the results.

4.4.1 Data

We test the POS induction part of the model on all languages in the Multext-East corpora (Erjavec, 2004) as well as on the free corpora from the CONLL-X Shared Task[7] for Dutch, Danish, Swedish and Portuguese. The evaluation of morphological segmentations uses the annotated word lists from the Morpho Challenge competition for English, Finnish and Turkish.[8] We gathered the sentences from Europarl corpus[9] for English and Finnish, and use the Turkish text data from the Morpho Challenge 2009 competition.[10] Estonian annotated segmentations are obtained from the Estonian morphologically disambiguated corpus.[11]

4.4.2 Evaluation

We report three accuracy measures for POS induction results: greedy 1-1, M-1 and V-m (see section 2.4.7). Segmentation is evaluated using the segment boundary F1-score (see section 2.3.4).

4.4.3 Experimental setup

For each experiment, we report the median result of five randomly initialized samples. The sampler was first run for 200 iterations for burnin, after which we collected the five samples, letting the sampler run for another 200 iterations between each two samples. We started with 15 segmenting iterations during each Gibbs iteration to enable the segmentation sampler to burnin to the current tagging state, and gradually reduced this number to one.

During preliminary experiments, we observed that the different components making up the log-posterior have different magnitudes. In particular, we found that the emission component is roughly four times smaller in the log-scale than the transition probability. This observation motivated introducing an additional parameter a into the model to scale the emission probability up. If the original word emission probability is $P(\mathbf{w})$ then the scaled emission probability is $P(\mathbf{w})^{\frac{1}{a}}$. We tried a couple of different scaling values on the MTE English corpus and then set its value to 4 for all languages for the rest of the experiments. This improved the tagging results consistently across all languages.

[7] http://ilk.uvt.nl/conll/free_data.html

[8] http://research.ics.tkk.fi/events/morphochallenge2010/datasets.shtml

[9] http://www.statmt.org/europarl/

[10] http://research.ics.tkk.fi/events/morphochallenge2009/datasets.shtml

[11] http://www.cl.ut.ee/korpused/morfkorpus/index.php?lang=eng

Table 4.1: POS induction results for different languages. For each language, the median one-to-one (1-1), many-to-one (M-1) and V-measure (V-m) together with the standard deviation from five runs is reported where the median is taken over V-measure. **Types** is the number of word types in each corpus, **True** is the number of tags in the reference corpus and **Induced** is the median number of induced tags together with the standard deviation. **Best Pub.** lists the best published results so far (also 1-1, M-1 and V-m) in (Christodoulopoulos et al., 2011)*, (Blunsom and Cohn, 2011)⋆ and (Lee et al., 2010)†.

	Types	**1-1**	**M-1**	**V-m**	**Induced**	**True**	**Best Pub.**		
Bulgarian	15103	**50.3 (0.9)**	**71.9 (3.8)**	54.9 (2.2)	13 (1.6)	12	-	66.5*	**55.6***
Czech	17607	**46.0 (1.0)**	60.7 (1.6)	46.2 (0.7)	12 (0.8)	12	-	**64.2***	**53.9***
Danish	17157	**53.2 (0.2)**	69.5 (0.1)	52.7 (0.4)	14 (0.0)	25	43.2†	**76.2⋆**	**59.0***
Dutch	27313	**60.5 (1.9)**	**74.0 (1.6)**	**59.1 (1.1)**	22 (0.0)	13	55.1†	71.1*	54.7*
English	9196	**67.4 (0.1)**	**79.8 (0.1)**	**66.7 (0.1**	13 (0.0)	12	-	73.3*	63.3*
Estonian	16820	**47.6 (0.9)**	**64.5 (1.9)**	45.6 (1.4)	14 (0.5)	11	-	64.4*	**53.3***
Farsi	11319	**54.9 (0.1)**	**65.3 (0.1)**	**52.1 (0.1)**	13 (0.5)	12	-	-	-
Hungarian	19191	**62.1 (0.7)**	**71.4 (0.3)**	**56.0 (0.6)**	11 (0.9)	12	-	68.2*	54.8*
Polish	19542	**48.5 (1.8)**	**59.6 (1.9)**	**45.4 (1.0)**	13 (0.8)	12	-	-	-
Portuguese	27250	45.4 (1.1)	71.3 (0.3)	55.4 (0.3)	21 (1.1)	16	**56.5†**	**78.5⋆**	**63.9***
Romanian	13822	**44.3 (0.5)**	60.5 (1.7)	46.7 (0.5)	14 (0.8)	14	-	**61.1***	**52.3***
Serbian	16813	**40.1 (0.2)**	60.1 (0.2)	43.5 (0.2)	13 (0.0)	12	-	**64.1***	**51.1***
Slovak	18793	**44.1 (1.5)**	**56.2 (0.8)**	**41.2 (0.6)**	14 (1.1)	12	-	-	-
Slovene	16420	**51.6 (1.5)**	66.8 (0.6)	51.6 (1.0)	12 (0.7)	12	-	**67.9***	**56.7***
Swedish	18473	**50.6 (0.1)**	60.3 (0.1)	55.8 (0.1)	17 (0.0)	41	38.5†	**68.7***	**58.9***

Table 4.2: Segmentation results on different languages calculated on word types. For each language the precision, recall and F1 measure, the number of word types in the corpus and the number of word types with reference segmentations available is reported. We present two results for each language—trained with and without the emission scaling (no ES and ES respectively).

Language		Precision	Recall	F1	Types	Eval
Estonian	no ES	43.5	59.4	50.3	16820	16820
	ES	42.8	54.6	48.0		
English	no ES	69.0	37.3	48.5	20628	399
	ES	59.8	29.0	39.1		
Finnish	no ES	56.2	29.5	38.7	25364	292
	ES	56.0	28.0	37.4		
Turkish	no ES	65.4	44.8	53.2	18459	293
	ES	68.9	39.2	50.0		

4.5 Results

In this section, we first describe the POS induction and morphological segmentation results. Then we describe some further experiments to assess whether the POS tags and segmentations influenced each other during learning and also evaluate the convergence of the model.

4.5.1 POS induction results

POS induction results for all languages are given in Table 4.1. When comparing these numbers with recently published results on the same corpora (Christodoulopoulos et al., 2011; Blunsom and Cohn, 2011; Lee et al., 2010), we can see that the numbers compare favourably with the state-of-the-art, beating the best-published results on many occasions. The tagging decisions were conditioned on morphological suffixes and thus we expected to learn clusters corresponding more to the morphosyntactic function of words. However, the number of tag clusters induced by the model corresponds surprisingly well to the number of coarse-grained tags in the reference corpus across all languages, which is at least partly the effect of using the emission scaling heuristic.

4.5.2 Segmentation results

Segmentation results are presented in Table 4.2. For each language, we report type-based precision, recall and F-measure, the number of word types in the corpus and the number of word types with reference segmentations available. We give the segmentation results both with and without the emission scaling heuristic and note that while the emission scaling improves the tagging accuracy, it degrades the segmentation results.

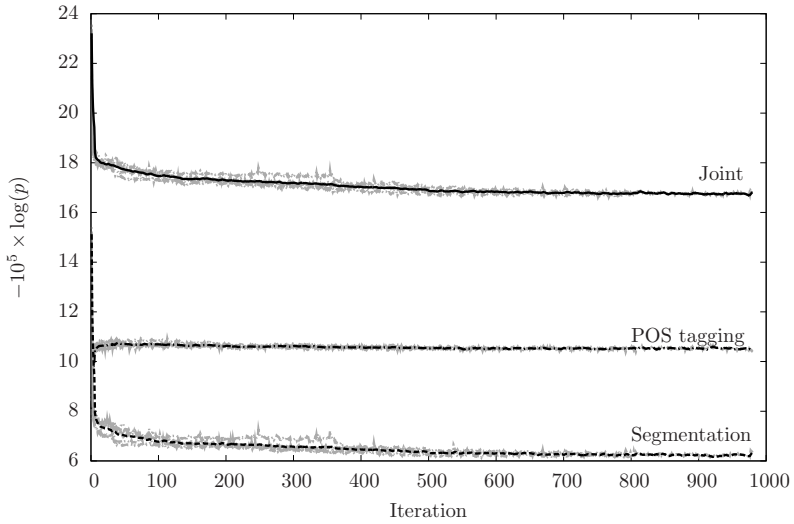

Figure 4.7: Log-posterior of samples plotted against iterations. Dark lines show the average over five runs, gray lines on the back show the real samples.

In general, the precision score is better but for Estonian, recall is higher. This can be explained by the characteristics of the reference annotations. For English, Finnish and Turkish, we use the Morpho Challenge competition annotated word lists where the reference segmentations are very fine-grained, separating both inflectional and derivational morphemes. These reference annotations sometimes contain very short morphemes in the middle of the word that are hard to learn with unsupervised data-driven methods. On the other hand, the Estonian corpus uses much coarser annotation scheme, mainly separating inflectional morphemes, which leads to higher recall. Some difference can also stem from the fact that the sets of gold-segmented word types for other languages are much smaller than in Estonian. Thus, it would be interesting to see whether and how the results would change if the evaluation could be done on all word types in the corpus for other languages as well. Under-segmentation is, in general, more acceptable than over-segmentation, especially when the aim is to use the resulting segmentations in some downstream NLP application.

4.5.3 Further experiments

Next, we studied the convergence characteristics of the model. For these experiments, we performed five runs with random initializations on Estonian corpus and let the sampler run up to 1000 iterations. Samples were taken after each ten iterations. Figure 4.7 shows the log-posterior of the samples plotted against the iteration number. Dark lines show the averages over five

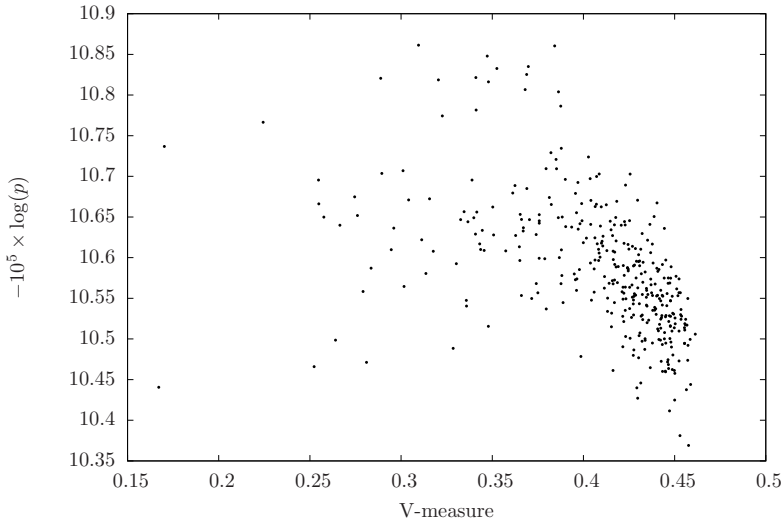

Figure 4.8: V-measure plotted against the tagging part of log-posterior.

runs and gray lines in the background are the probabilities of real samples also showing the variance.

We first calculated the full posterior of the samples (the solid line) that showed a quick improvement during the first few iterations and then stabilized by continuing with only slow improvements over time. We then divided the posterior into different factors in order to see the contribution of each component separately, plotting on the figure the log-posterior corresponding to the tag transition and segmentation components. The separated lines show that most of the improvement in the log-posterior is caused by the increasing probability of the segmentation component. The probability of the tag transitions improves as well but much less.

For zooming-in into the tagging component of the model, we next plotted in Figure 4.8 the V-measure of all samples against the log-posterior of the tag transitions. It reveals that the lower V-measure values are more spread out in terms of probability. These points correspond to the earlier samples of the runs. The samples taken later during the runs are on the right in the figure and the positive correlation between the V-measure and the log-posterior values can be seen.

Finally, we tried to assess whether the morphological segmentations and POS tags help each other in the learning process. For that, we conducted two semi-supervised experiments using the Estonian corpus. First, we fixed the segmentations to the reference annotations and only learned the tags. Then, we gave the model reference POS tags and only learned the segmentations.

Table 4.3: Tagging and segmentation results on the MTE Estonian corpus (Unsupervised) compared to the semi-supervised setting with the fixed reference segmentations or tags (Semi-supervised). Finally, the segmentation results of the Morfessor system for comparison are presented.

Tagging	**1-1**	**M-1**	**V-m**
Semi-supervised	40.5	53.4	37.5
Unsupervised	47.6	64.5	45.6
Segmentation	**Precision**	**Recall**	**F1**
Semi-supervised	36.7	56.4	44.5
Unsupervised	42.8	54.6	48.0
Morfessor	51.3	52.6	51.9

The results are given in Table 4.3. The joint unsupervised learning results are also added for easier comparison. Unfortunately, we could not perform this experiment on other languages because Estonian was the only language for which we could obtain both reference tags and segmentations for all the words in the MTE corpus.

The table shows that the unsupervised joint learning results for both POS induction and segmentation are better than the results of semi-supervised learning. This is surprising because one would assume that providing reference annotations would lead to better results. On the other hand, these results are encouraging, showing that learning two dependent tasks in an unsupervised joint model can be as good or even better than learning the same tasks separately and providing the gold standard data as features.

Finally, we also learned the morphological segmentations with the unsupervised morphology induction system Morfessor baseline[12] (Creutz and Lagus, 2005) and report the results in the last row of Table 4.3. Apparently, our joint model cannot beat Morfessor in morphological segmentation, and when applying the emission scaling that influences the tagging results favourably, the segmentation results get even worse. Although the semi-supervised experiments show that the model to some extent captures dependencies between tags and segmentations, the segmentation results themselves are relatively low, indicating that the syntactic clusters fail to influence the segmentation decisions strongly and in the desired direction.

[12] `http://www.cis.hut.fi/projects/morpho/`

4.6 Discussion

The experimental results presented in the previous section demonstrated good results in POS induction but mediocre results in morphological segmentation. This behaviour suggests that the model setup did not capture the relationships between POS tags and suffixes as well as expected. In this section, we review the possible causes of this failure. There were also some technical problems related to the generative story and inference, described in sections 4.3 and 4.3.2. We also return to those problems and sketch possible solutions.

The first problem with the model is that the lexicon dictates, which words each tag can emit, but the emission distributions have the dimensionality of the vocabulary and thus in principle can emit any word from the vocabulary. One possible solution to this problem is that of Lee et al. (2010), where the different emission distributions have different dimensionalities depending on the number of words having the respective tag in the lexicon. This solution is easy to implement with the finite model, but with the infinite model the following problem arises. Whenever a new tag is considered by the sampler, a new emission distribution is created. At this very moment, the probability of emitting the word under consideration from this emission distribution is one because it is currently the only word emitted by this tag. Such behaviour has the undesirable effect of instantiating too many tags. Perhaps a better and easier solution would be the one used by Blunsom and Cohn (2011), who use hierarchical CRPs instead of Dirichlet-multinomials also for word emission. The emission distributions have the same coupled base distribution over the whole lexicon, but each tag-dependent distribution chooses only a specific subset of the lexicon as its support.

The second issue that makes the model description and also inference too complicated is using the distributions that generate segments proportional to their token frequencies. This creates a kind of circularity in the generative story, where the segmentations are generated in the lexicon but generating the segmentation distributions is conditioned on the token sequences, which at that point are not generated yet. Using segment token frequencies also makes updating the segmentation proposal distribution during the forward table computation necessary, because otherwise the proposal might deviate too much from the true segment distribution. Both of those problems can be solved by keeping the segmentations strictly type-based and not emitting them in the HMM chain. Although several popular unsupervised morphology induction systems have used word tokens for learning type segmentations, some of the recent results (Virpioja et al., 2013) showed better segmentation results when omitting token frequencies. Using segmentations in the lexicon only enables constructing the proposal distribution from the snapshot of

the CRP caches, rendering segment probabilities conditionally independent of each other and thus making the construction of the forward table more sound mathematically.

The next issue is the sampler bias also caused by working with word tokens. When computing the probability of a word type having a particular tag, it is unrealistic to try to marginalize over all table configurations of the different contexts where the word type occurs in. Here we sketch a Metropolis-Hastings sampler that can be used to correct this bias. The general idea is to store for each trigram context its seating configuration. If a certain word trigram occurs n times, then its seating configuration is a list of n table indices indicating which table the specific token sits at. For this seating information to be meaningful we have to assign index for each table serving the same dish (tag) and keep those indices fixed. Also, each table has to know in which table it sits at in the base distribution.

We have to store explicitly for each customer in which table it sits at. Then we can use a proposal distribution to propose a new tag. For instance, a relatively simple proposal distribution would be to multiply together the posterior predictive probabilities of each relevant trigrams but ignoring the trigram counts. For example, if a word type occurs in three different word contexts and in one of those it occurs twice, so that altogether there are 4 tokens, then the proposal would compute the probability based on those three different contexts only ignoring the fact that one of those contexts appeared twice. Before performing the acceptance test, we have to generate a specific seating configuration for all tokens for the proposed tag. The proposed tag together with its token seatings will be accepted according to the true posterior probability of this particular seating configuration. It might happen that the proposed tag is same as the old tag. In this case, it does not necessarily mean that the state will be unchanged because the proposed seating might be different from the old one, and it still has to be either accepted or rejected. A similar approach can be adopted with the segmentations where we can store for each word type its seating configuration related to its segmentation.

Finally, we would like to discuss the issues related to the segmentation performance. The segmentation results presented in this chapter are moderate and below the Morfessor baseline. One of the reasons for that is the simplicity of the segmentation component, which is just a unigram model over segments. In addition, all segments are conditioned on the tag, which poses an incorrect assumption that not only the suffixes are tag-dependent but so are also the stems. A simple solution would be to model only suffixes as tag-dependent and have a tag independent distribution over the rest of the morphemes like in (Can, 2011; Lee et al., 2011). However, the unigram segmentation model does not know which segments correspond to suffixes and

which correspond to stems. For instance, the segmentation of a compound word can consist of several segments, none of which is a suffix. Another useful cue, as showed in (Lee et al., 2011) would be to model agreement between the suffixes of adjacent word tokens in case they share the same sequence of final characters. In general, morphological segmentation is a complex task on its own and developing more successful segmentation models is the topic of the next chapter.

4.7 Conclusion

In this chapter, we presented a joint unsupervised model for learning POS tags and morphological segmentations with a hierarchical Dirichlet process model. The model induces the number of POS clusters from data and does not contain any hand-tuned parameters. We tested the model on several languages and showed that it produces state-of-the-art POS induction results, although the segmentation results were hardly impressive.

We demonstrated with a set of semi-supervised experiments, where either POS tags or morphological segmentations were fixed to their reference annotations, that the unsupervised joint learning of both tasks leads to better results, which provides some evidence that the joint learning is beneficial, and the two learning tasks influence each other.

The results also showed that the information flow in both directions was not as strong or informative as hoped. First, the tagging decisions were dependent on the suffix of the word type, which according to our expectations should have given rise to clusters corresponding to morphosyntactic classes. However, the model learned clusterings closer to the coarse-grained syntactic tagset in all languages, which means that the suffixes did not have a strong enough influence on the tagging decisions. Second, the morphological segmentations learned were mediocre at best, which suggests that although the segmentation decisions were dependent on the tag of the word type, the dependency structure did not accurately capture the relevant cues. In that sense these results support the Claim B of this dissertation with a negative example—that although the interaction between morphology and syntactic tags clearly exists, their relationship, especially in morphologically rich languages, is more complex than just a one-to-one relation between a suffix and a syntactic class. However, experimenting with a model such as the one proposed in this chapter, enables studying the characteristics of this relationship.

Chapter 5

Weakly-supervised morphological segmentation

In this chapter we present two weakly-supervised methods for performing morphological segmentation using Adaptor Grammars. The performance of these models is evaluated on five languages using various baselines. The basis of this chapter is publication II (Sirts and Goldwater, 2013).

5.1 Introduction

Morphological segmentation has been a highlighted task in unsupervised computational morphology for the last two decades, during which numerous approaches have been proposed; for an overview see section 2.3.2. In the previous work two main shortcomings can be identified: 1) many proposed methods do not use a well-defined model but rather execute a sequence of ad hoc learning procedures; 2) most only exploit the flat structure of the segmentation and do not model the interdependencies between morphemes or their substructures. Of course, not all previous work has both of those shortcomings, although many of them do. In the work described in this chapter, we attempt to address both of these issues.

One can claim that the requirement for a well-defined model is an artificial one as long as the method does well what it is supposed to do. However, a well-defined model provides a sound description of the problem and enables developing well-grounded hypotheses about its expected behaviour. Also, a well-defined probabilistic model is in itself a modular component that can be, although usually not entirely effortlessly, integrated into a larger probabilistic model. The minimum description length (MDL) principle (Rissanen, 1989) that has been popular for developing morphological segmentation models (e.g. Goldsmith's (2001) Linguistica and the models of Morfessor family (Creutz and Lagus, 2007)) provides a well-defined modeling framework. However, the

inference procedures of those models are full of heuristic steps that cannot be easily integrated with any other model. The non-parametric Bayesian models presented in this chapter, on the other hand, are defined in the generative probabilistic framework that, in addition to enabling using mathematically well-founded and principled inference methods, also provide clear semantics for the model in the form of a generative story. In addition, non-parametric Bayesian models naturally incorporate Zipfian behaviour (Zipf, 1932) into the model.

The second issue involves modeling the flat morphological structure only. For example, a flat unigram segmentation model might not be able to learn to separate morphemes in affix sequences, as for instance in *connect_ions* if the sequence *ion_s* occurs frequently enough. Modeling the hierarchical structure of morphemes can help to overcome such problems as learning that *ions* is a collocation that consists of morphemes *-ion* and *-s* will be made possible. In previous morphological segmentation systems, the hierarchical structure of morphemes has been used in Morfessor Categories-Map (Creutz and Lagus, 2005) (see section 5.4.3). Linguistica system (Goldsmith, 2001) also represents the lexicon recursively. Here, the stem of one word can be again segmented into a stem and a suffix. For instance, the complex word *workings* is first analyzed as *working_s* and then its stem is again analyzed as *work_ing*. However, the recursion is always applied to the stem part only, and the splitting stops at the morpheme level without attempting to model the morpheme substructures. The models in this chapter are defined in the Adaptor Grammar framework (Johnson et al., 2007), which allows defining models with hierarchical structure, enabling additional structures that naturally extend both above and below the target structures (morphemes in this case).

Adaptor Grammars (AGs) are a non-parametric Bayesian modeling framework that can learn latent tree structures over an input corpus of strings. For example, they can be used to define a morphological grammar where each word consists of zero or more prefixes, a stem, and zero or more suffixes; the actual forms of these morphs (and the segmentation of words into morphs) are learned from the data. In this general approach, AGs are similar to many other unsupervised morphological segmentation systems (e.g. the aforementioned Linguistica and Morfessor systems). A major difference, however, is that the morphological grammar is specified as an input to the program, rather than hard-coded, which allows different grammars to be explored easily. For the task of segmenting utterances into words, for example, Johnson and colleagues have experimented with grammars encoding different kinds of sub-word and super-word structure (e.g., syllables and collocations), showing that the best grammars far outperform other systems on the same corpora (Johnson, 2008a; Johnson and Goldwater, 2009; Johnson

and Demuth, 2010). These word segmentation papers demonstrated both the power of the AG approach and the synergistic behavior that occurs when learning multiple levels of structure simultaneously. However, turning back to morphology, we know that different languages can have very different morphological properties, so using a single unsupervised grammar for all languages may not be the best approach. Though AGs make it easy to try many different possible grammars, the process of proposing and testing plausible options for different languages can still be time-consuming.

The problem of grammar selection can be tackled by adopting a weakly-supervised approach instead of an entirely unsupervised one. In this chapter, we present two models for weakly-supervised morphological segmentation that use the AG framework as their basis. First, we propose a novel method, AG Select, for automatically selecting good morphological grammars for different languages using a small amount of annotated data. AG Select uses the AG framework for specifying a very general binary-branching grammar that is used to learn a parse tree of each word that contains many possible segmentation splits for the word. Then, we use the annotated data to determine, for each language, which of the proposed splits from the original grammar should be used in order to best segment that language. The other method—semi-supervised AG—uses the annotated data for accumulating rule statistics. We train the semi-supervised AG with hierarchical grammars that in addition to morphemes also define, e.g., submorphemes or morpheme collocations. The variables associated with the annotated structures in the supervised data are kept fixed, and all the structures of the unannotated data plus the latent sub- or super-structures of the annotated data are inferred during learning.

We evaluate both approaches on several languages (English, Finnish, Turkish, Estonian and German) using both a small development set and the full Morpho Challenge[1] test set—up to three million word types. In doing so, we demonstrate that using the posterior grammar of an AG model to decode unseen data is a feasible way to scale these models to large datasets. We compare to several baselines that use the annotated data to different degrees: unsupervised and semi-supervised Morfessor (Creutz and Lagus, 2007; Kohonen et al., 2010a), unsupervised Morsel (Lignos, 2010), and unsupervised AGs. Both proposed methods yield comparable results to the best of these other approaches. We will contrast a flat morphological grammar with grammars that also model the latent substructures of morphemes. The results of these experiments provide empirical evidence that modeling the hierarchical latent structures improves the segmentation results considerably.

[1] `http://research.ics.aalto.fi/events/morphochallenge/`

To summarize, the contributions of this chapter are: 1) introducing a novel grammar selection method for AG models that achieves morphological segmentation results competitive with the best existing systems; 2) demonstrating how to train semi-supervised AG models, and showing that this improves morphological segmentation over unsupervised training; and 3) scaling AGs to large data sets by using the posterior grammar for decoding.

The structure of this chapter is as follows. We start in section 5.2 with the description of AG as introduced by Johnson et al. (2007). Then, in section 5.3, we describe the two novel methods for weakly-supervised AG learning and explain how to use the posterior grammar as an inductive model. Section 5.4 describes the experimental setup, including the datasets and descriptions of the baseline models. Results of the experiments are presented in section 5.5, followed by the discussion in section 5.6 that also offers examples of correctly and incorrectly segmented words as well as the snippets of the posterior grammars learned. Finally, in section 5.7 we will look how the presented results support the thesis Claims and propose some possibilities for further work.

5.2 Adaptor Grammars

Adaptor Grammars (AG) (Johnson et al., 2007) are a framework for specifying probabilistic models that can be used to learn latent tree structures from a corpus of strings. An AG model has two components: a *base distribution*, which is just a probabilistic context-free grammar (PCFG), and an *adaptor* that changes the probabilities of the subtrees, such that the adapted probability of a subtree may be substantially different from the product of probabilities of the PCFG rules required to construct it.

Adaptor Grammars have been applied to a wide variety of tasks, including segmenting utterances into words (Johnson, 2008b,a; Börschinger and Johnson, 2014), named entity clustering (Elsner et al., 2009), classifying documents according to perspective (Hardisty et al., 2010), unsupervised dependency parsing (Cohen et al., 2010), LDA topic modeling (Johnson, 2010), Chinese word segmentation (Johnson and Demuth, 2010), machine transliteration of names (Huang et al., 2011), native language identification (Wong et al., 2012), and joint learning of words and their referents (Johnson et al., 2010, 2012). There have also been AG experiments with morphological segmentation, but more as a proof of concept than an attempt to achieve state-of-the-art results (Johnson et al., 2007; Johnson, 2008b). Recently, AGs have been also used to define grammars for learning non-concatenative morphology (Botha and Blunsom, 2013).

5.2.1 Model

The description of the AG model in this section closely follows that of Johnson et al. (2007). The material assumes familiarity with the Pitman-Yor process (see section 3.1.3).

Probabilistic context-free grammars

A probabilistic context-free grammar (PCFG) is a quintuple $(N, T, R, S, \boldsymbol{\theta})$, where N is a finite set of non-terminal symbols, T is a finite set of terminal symbols, R is a finite set of rewrite rules of the form $A \to \alpha$, where $A \in N$ and $\alpha \in (N \cup T)^*$. $S \in N$ is a distinguished start symbol and $\boldsymbol{\theta}$ is a vector of model parameters such that the parameters of the set of rules R_A sharing the same left-hand side A form a probability distribution.

$$\sum_{A \to \alpha \in R_A} \theta_{A \to \alpha} = 1, \tag{5.1}$$

where $\theta_{A \to \alpha}$ is the probability of the rule $A \to \alpha$.

The probability of a tree under a PCFG is the product of all rules necessary to construct that tree. If $R_{\mathcal{T}}$ is the set of rules used to construct the tree $\mathcal{T}$ then the probability of that tree is:

$$P(\mathcal{T}) = \prod_{A \to \alpha \in R_{\mathcal{T}}} \theta_{A \to \alpha} \tag{5.2}$$

Denote by $\mathcal{T}_A$ a tree rooted in a non-terminal symbol $A \in N$. Also, denote by $s = \text{yield}(\mathcal{T}_A)$ the yield (sequence of terminal symbols) of this tree. Then the probability of the non-terminal symbol A generating the terminal symbol sequence s is the sum of the probabilities of all trees rooted in A and yielding the sequence s:

$$P_A(s) = \sum_{\mathcal{T}_A : \text{yield}(\mathcal{T}_A) = s} P(\mathcal{T}_A) \tag{5.3}$$

Adaptor Grammars

An Adaptor Grammar is a sextuple $(N, T, R, S, \boldsymbol{\theta}, \mathbf{C})$, where $(N, T, R, S, \boldsymbol{\theta})$ is a PCFG and $\mathbf{C}$ is a vector of adaptors indexed by distinct non-terminal symbols. Each component in $\mathbf{C}$ contains the parameters of an adaptor function that maps the distribution over trees rooted in a non-terminal symbol $A \in N$ to an adapted distribution over the same set of trees. Thus, if G_A is the distribution over trees rooted in A according to the PCFG then the adapted distribution H_A under the AG model is:

$$H_A \sim C_A(G_A) \tag{5.4}$$

Although theoretically the adaptor can be any function that maps one distribution onto another, Johnson et al. (2007) use the Pitman-Yor (PY) CRP as an adaptor because it acts as a caching model. Under the PY AG model, the posterior probability of a particular subtree is roughly proportional to the number of times that subtree occurs in the current analysis of the data. The probabilities of the unseen subtrees are computed using the base PCFG distribution. In PY AG the adaptor of a non-terminal A is represented by a tuple $(a_A, b_A, \mathbf{x}_A, \mathbf{n}_A)$, where a and b are the PYP hyperparameters, $\mathbf{x}_A$ is the sequence of cached subtrees rooted in A and $\mathbf{n}_A$ is the vector of counts of those subtrees. If the smoothing parameter $a = 1$ then the PYP forces each new customer (tree) to sit at a new table and its probability is equal to the base probability according to the PCFG. The latter is equivalent to the adaptor being the identity function.

The analysis of a string s under PY AG model is a pair $u = (\mathcal{T}_A, l)$, where $\mathcal{T}_A$ is the tree rooted in A yielding the string s and $l(\cdot)$ is an index function, such that:

$$l(\mathcal{T}_A) = i \iff \mathbf{x}_{A,i} = \mathcal{T}_A \tag{5.5}$$

After observing the analyses $\mathbf{u} = u_1, \ldots, u_n$ the predictive posterior probability of a new analysis u can be computed:

$$\begin{aligned} &P(u = (\mathcal{T}_A, l)|\mathbf{C}(\mathbf{u})) \\ &\quad = \begin{cases} \dfrac{\mathbf{n}_{A,i} - a_A}{n_A + b_A}, \text{ if } i \leq m_A \\ \dfrac{m_A a_A + b_A}{n_A + b_A} \cdot \theta_{A\to\alpha} \displaystyle\prod_{A' \in \alpha} P(u' = (\mathcal{T}_{A'}, l)|\mathbf{C}(\mathbf{u})), \quad \text{otherwise} \end{cases} \end{aligned} \tag{5.6}$$

$\mathbf{C}(\mathbf{u})$ is the adaptor state computed based on the analyses $\mathbf{u}$, $i = l(\mathcal{T}_A)$ is the index related to the subtree $\mathcal{T}_A$ in the adaptor vectors $\mathbf{x}_A$ and $\mathbf{n}_A$, $\mathbf{n}_{A,i}$ is the number of times the subtree $\mathcal{T}_A$ was previously cached with the index i[2], m_A is the number of elements in $\mathbf{x}_A$, $n_A = \sum_{i=1}^{m_A} \mathbf{x}_{A,i}$ is the total number of (sub)trees rooted in A, $\theta_{A\to\alpha}$ is the probability of the rule $A \to \alpha$ according to the PCFG and u' are the analyses of the subtrees rooted at the children of the root node of $\mathcal{T}_A$.

5.2.2 Inference with PY Adaptor Grammars

This section assumes familiarity with the Markov chain Monte Carlo methods explained in section 3.2.

[2]There can be several indices in $\mathbf{x}_A$ and $\mathbf{n}_A$ referring to the similar subtree. In CRP notation they are different tables but serving the same dish (subtree).

An AG model can be specified by writing down the CFG rules (the support for the base distribution) and indicating which non-terminals are adapted, i.e., can serve as the root of a cached subtree. Given this specification and an input corpus of strings, Markov chain Monte Carlo samplers can be used to infer the posterior distribution of trees and all parameters of the model. Any frequently recurring substring will tend to be parsed consistently, as this permits the model to treat the subtree spanning that string as a cached subtree, assigning it higher probability than would be the case under the PCFG distribution.

Our experiments are based on Mark Johnson's implementation[3] that uses a Gibbs sampler to resample the latent parse trees for each input string in turn. Within each Gibbs step a Metropolis-Hastings (MH) sampler is used to propose a new parse tree for each string. The MH proposal distribution is a PCFG constructed from the snapshot of the present AG state including the parses of all other input strings except the one currently being resampled. Under this distribution, all sampling decisions are independent of each other and thus, the standard inside table for PCFG parsing can be constructed using dynamic programming. The rules used to construct the inside table include not only the PCFG rules but also all cached subtrees. A proposal tree can then be generated starting from the root node and sampling each expansion from the top down. Finally, the proposed parse is accepted according to the Metropolis acceptance distribution constructed using the true probability of the proposed tree.

There are a few other relevant technical details in Mark Johnson's PY AG sampler described in Johnson and Goldwater (2009). The first is the PYP hyperparameter resampling that uses a slice sampler (Neal, 2003). Each adapted non-terminal is associated with separate concentration and smoothing parameters (a and b hyperparameters) which are given vague gamma and beta priors respectively. They are resampled from the posterior $p(\theta|D) \propto P(D|\theta)P(\theta)$, where θ is the parameter resampled and D is the current AG state.

Another procedure, which speeds the mixing, is *table label resampling* which is done after every Gibbs sweep over the data. In CRP terms, each cached parse is a dish on some table and during table label resampling, each table is processed in turn and the parse on that table is resampled using the same MH sampling scheme described above. If the resampled parse is accepted, then it changes the parses of all the customers sitting at that table at once, i.e. all subtrees sharing that analysis, allowing bigger jumps in the state space.

[3] `http://web.science.mq.edu.au/~mjohnson/Software.htm`

Besides MCMC sampling also other methods have been proposed for AG inference: Cohen et al. (2010) implemented variational inference for AG learning and Zhai et al. (2014) proposed a hybrid algorithm combining both MCMC sampler and variational inference for online learning.

5.3 Morphological segmentation with Adaptor Grammars

Initially, the AG framework was designed for unsupervised learning. This section first describes how AGs can be used for unsupervised morphological segmentation, and then introduces two ways to use a small labeled dataset to improve performance: semi-supervised learning and grammar selection. Although both of these methods are presented in the context of morphological segmentation, they are general in principle and can be used for other tasks as well. The grammar selection method AG Select is clearly weakly-supervised—the AG is trained unsupervised and the labeled data is used to select the best grammar from the trained AG. Increasing the amount of labeled data in this context would probably not improve the results. On the other hand, the semi-supervised AG can exploit different amounts of labeled data—from as little as possible to as much as available.

5.3.1 Unsupervised Adaptor Grammars

We define three AG models to use as unsupervised baselines in our segmentation experiments. The first of these is very simple:

$$\begin{aligned} \text{Word} &\rightarrow \underline{\text{Morph}}^{+} \\ \underline{\text{Morph}} &\rightarrow \text{Char}^{+} \end{aligned} \tag{5.7}$$

The underline notation indicates an adapted non-terminal, and $^{+}$ abbreviates a set of recursive rules, e.g., Word $\rightarrow \underline{\text{Morph}}^{+}$ is short for

$$\begin{aligned} \text{Word} &\rightarrow \text{Morphs} \\ \text{Morphs} &\rightarrow \underline{\text{Morph}}\ \text{Morphs} \\ \text{Morphs} &\rightarrow \underline{\text{Morph}} \end{aligned}$$

Grammar (5.7) (**MorphSeq**) is just a unigram model over morphemes. The Morph symbol is adapted, so the probability of each Morph will be roughly proportional to its (inferred) frequency in the corpus. The grammar does not impose any constraints on the number of morphemes, the relationships between different morphemes, or the structure inside the morphemes (other than a geometric distribution over morpheme lengths).

Experiments with AGs for unsupervised word segmentation suggest that adding further latent structure can help with learning. Here, we add another

layer of structure below the morphemes, calling the resulting grammar **SubMorphs**. Because the non-terminal labels are arbitrary, this grammar can also be interpreted as adding another layer on top of morphemes, allowing the model to learn morpheme *collocations* that encode dependencies between morphemes (which themselves have no substructure). However, preliminary experiments showed that the morpheme/submorpheme interpretation performs better than the collocation/morpheme interpretation. Hence, we chose the corresponding non-terminal names.

$$\begin{aligned} \text{Word} &\rightarrow \underline{\text{Morph}}^{+} \\ \underline{\text{Morph}} &\rightarrow \underline{\text{SubMorph}}^{+} \\ \underline{\text{SubMorph}} &\rightarrow \text{Char}^{+} \end{aligned} \tag{5.8}$$

For capturing the morphotactic rules, a grammar with linguistically motivated non-terminals can be created. There are many plausible options, and the best-performing grammar may be somewhat language-dependent. Rather than experimenting extensively, we designed a grammar to replicate as closely as possible the grammar that is implicitly implemented in the Morfessor system. This grammar distinguishes between prefixes, stems and suffixes, allows compounding, defines the order in which the morphemes can occur and also allows the morphemes to have inner latent structure. We will refer to this grammar in the following as **Compounding**:

$$\begin{aligned} \text{Word} &\rightarrow \underline{\text{Compound}}^{+} \\ \underline{\text{Compound}} &\rightarrow \underline{\text{Prefix}}^{*}\ \underline{\text{Stem}}\ \underline{\text{Suffix}}^{*} \\ \underline{\text{Prefix}} &\rightarrow \underline{\text{SubMorph}}^{+} \\ \underline{\text{Stem}} &\rightarrow \underline{\text{SubMorph}}^{+} \\ \underline{\text{Suffix}} &\rightarrow \underline{\text{SubMorph}}^{+} \\ \underline{\text{SubMorph}} &\rightarrow \text{Char}^{+} \end{aligned} \tag{5.9}$$

The above grammars can be used for unsupervised learning when no amount of labeled data is available. However, often at least a small amount of labeled data is available, and the following sections describe two methods of exploiting such data in AG learning.

5.3.2 Semi-supervised Adaptor Grammars

The first new use of AGs we introduce is the *semi-supervised AG*, where we use the labeled data to extract counts of different rules and subtrees present in the annotated analyses. We then run the MCMC sampler as before over both the unlabeled and labeled data, treating the counts from the labeled data as fixed.

We assume that the labeled data provides a consistent bracketing (no two spans in the bracketing can partially overlap) and the labels of the spans must be compatible with the grammar. However, the bracketing may not specify all levels of structure in the grammar. In our case, we have morpheme bracketings but not, e.g., submorphemes. Thus, using the SubMorphs grammar in semi-supervised learning will constrain the sampler so that Morph spans in the labeled data will remain fixed while the SubMorphs inside those Morphs will be resampled.

The main change made to the AG inference process to adopt semi-supervised learning was to introduce pruning for the inside table constructed for the proposal distribution. We prune any non-terminals that are inconsistent with the spans/labels in the given labeling.

In our semi-supervised experiments, we use the same grammars as for the unsupervised AG. We provide the system with a small set (1000 word types in all our experiments) of labeled data and train on unlabeled data. The labeled data must be formatted as bracketed parse trees consistent with the training grammar. For example, the segmented word *liv_ed* according to the MorphSeq grammar (5.7) should be bracketed as:

```
(Word   (Morphs   (Morph   (Chars (Char (l))
                             (Chars (Char (i))
                             (Chars (Char (v)))))
                  (Morphs  (Morph   (Chars (Char (e))
                                    (Chars (Char (d)))))))))
```

5.3.3 AG Select

Both the unsupervised and semi-supervised methods described above assume the definition of a grammar that adequately captures the phenomena being modeled. Although the AG framework makes it easy to experiment with different grammars, these experiments can be time-consuming and require some good guesses as to what a plausible grammar might be. These problems can be overcome by automating the grammar development process to evaluate different grammars systematically and find the best one.

We propose a weakly-supervised model selection method, AG Select, which uses the AG framework to identify the best grammar for different languages and datasets automatically. We first define a very general binary-branching CFG for AG training that we call the *metagrammar*. The metagrammar learns a parse tree for each word where each branch contains a different structure in the word. The granularity of these structures is determined by the depth of the tree. For example, Grammar (5.10) generates binary trees of depth two and can learn segmentations of up to four segments.

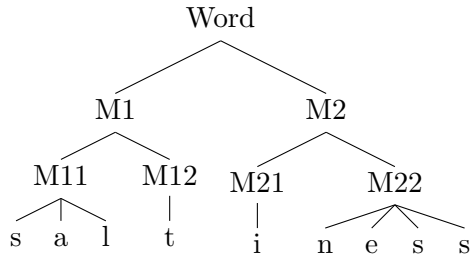

Figure 5.1: A parse tree generated by the metagrammar of depth 2 for the word *saltiness.*

Next we introduce the notion of a *morphological template*, which is an ordered sequence of non-terminals whose concatenated yields constitute a word and which is used to parse out a particular segmentation of that word. For example, using Grammar (5.10) a parse tree of the word *saltiness* is shown in Figure 5.1. There are four possible templates with four different segmentations: **M1 M2** (*salt_iness*), **M11 M12 M2** (*sal_t_iness*), **M1 M21 M22** (*salt_i_ness*), and **M11 M12 M21 M22** (*sal_t_i_ness*).

$$
\begin{aligned}
&\text{Word} \rightarrow \underline{\text{M1}} && \\
&\text{Word} \rightarrow \underline{\text{M1}}\ \underline{\text{M2}} && \underline{\text{M11}} \rightarrow \text{Chars}^{+} \\
&\quad \underline{\text{M1}} \rightarrow \underline{\text{M11}} && \underline{\text{M12}} \rightarrow \text{Chars}^{+} \\
&\quad \underline{\text{M1}} \rightarrow \underline{\text{M11}}\ \underline{\text{M12}} && \underline{\text{M21}} \rightarrow \text{Chars}^{+} \\
&\quad \underline{\text{M2}} \rightarrow \underline{\text{M21}} && \underline{\text{M22}} \rightarrow \text{Chars}^{+} \\
&\quad \underline{\text{M2}} \rightarrow \underline{\text{M21}}\ \underline{\text{M22}} &&
\end{aligned}
\tag{5.10}
$$

The morphological template consisting only of the non-terminals from the lowest cached level of the parse tree is expected to have a high recall, whereas the template containing the non-terminals just below the Word is expected to have a high precision. Our goal is to find the optimal template by using a small labeled dataset. The grammar selection process iterates over the set of all templates. For each template, we extract the segmentations of the words in the labeled set and compute the value of the desired evaluation metric. We choose the template that obtained the highest score.

For each language, we use a single template to segment all words in that language. However, even using (say) a four-morpheme template such as **M11 M12 M21 M22**, some words may contain fewer morphemes because the metagrammar permits either unary or binary branching rules. Therefore, some parses may not contain **M12** or **M2** (and thus **M21 M22**) spans. Thus, we can represent segmentations of different lengths (from 1 to 2^n,

where n is the depth of the metagrammar) with a single template. We also experimented with selecting different templates for words of different length but observed no improvements over the single template approach.

For our experiments, we use a metagrammar of depth four. This grammar allows words to consist of up to 16 segments, which we felt would be enough for any word in the training data. Also, iterating over all the templates of a deeper grammar would not be feasible as the number of different templates increases very rapidly. The number of templates of the depth i can be expressed recursively as $N_i = (N_{i-1} + 1)^2$, where N_{i-1} is the number of templates in the grammar of depth one less and $N_0 = 0$.

5.3.4 Inductive learning

Previous work on AGs has used relatively small datasets and run the sampler on the entire input corpus (some or all of which is also used for evaluation)—a *transductive* learning scenario. However, our larger datasets contain millions of word types, where sampling over the whole set is not feasible. For example, 1000 training iterations on 50k word types took about a week on one 2.67 GHz CPU. To solve this problem, we need an *inductive* learner that can be trained on a smaller set of data and then used to process a different larger set.

To create such a learner, we run the sampler on up to 50k word types, and then extract the *posterior grammar* as a PCFG. This grammar contains all the initial CFG rules, plus rules to generate each of the cached subtrees inferred by the sampler. We sum the counts of all the identical cached rules to obtain an unnormalized PCFG. Then we can use a standard CKY parser to decode the remaining data using this PCFG.

The experiments presented later show that the morphological segmentations decoded in such a way do not suffer in accuracy when compared to the transductive learning. Therefore, this inductive learning scheme enables pretraining and storing the expensive AG models in the form of posterior grammars and later reusing the stored models for new data with much smaller computational cost.

5.4 Experiments

In this section, we describe the experimental setup including the datasets used for conducting experiments and the methods for evaluating results. Also, the baseline models that are used to put our results into context are described in this section.

5.4.1 Data

We test on languages with a range of morphological complexity: English, Finnish, Turkish, German and Estonian. For each language, we use two

Table 5.1: Number of word types in our datasets. Test set words (except in Estonian) are an unknown subset of the unlabeled words.

	Unlabelled	**Train**	**Dev**	**Test**
English	0.9M	1000	1212	16K
Finnish	2.9M	1000	1494	225K
Turkish	0.6M	1000	1531	64K
German	2.3M	1000	785	62K
Estonian	2.1M	1000	1500	74K

small sets of annotated data—a *training set* for semi-supervised training or model selection and a *dev set* for development results—and one larger annotated dataset for final tests. We also have a large *unlabeled* set for each language. Table 5.1 gives statistics.

The datasets for English, Finnish, Turkish and German are from the Morpho Challenge 2010 competition[4] (MC2010). We use the MC2010 training set of 1000 annotated word types as our training data, and for our dev sets we collate together the development data from all years of the MC competition. The final evaluation is done on the official MC2010 test sets, which are not public, so we rely on the MC organizers to perform the evaluation. The words in each test set are an unknown subset of the words in the unlabeled corpus, so to evaluate we segmented the entire unlabeled corpus and sent the results to the MC team, who then computed scores on the test words.

The Estonian wordlist is gathered from the newspaper texts of a mixed corpus of Estonian.[5] Hand-annotated segmentations of some of these words are available from the Estonian morphologically disambiguated corpus;[6] we used these for the test set, with small subsets selected randomly for the training and dev sets.

For semi-supervised tests of the AG Compounding grammar we annotated the morphemes in the English, Finnish and Estonian labeled sets as prefixes, stems or suffixes. This annotation could not be done for Turkish because we did not know anybody who would know Turkish.

5.4.2 Evaluation

We evaluate the results with segment border F1-score (SBF1) and EMMA measure (see section 2.3.4). For our dev results, we computed both scores using the entire dev set, but for the large test sets, the evaluation is done in batches of 1000 word types selected randomly from the test set. This

[4] `http://research.ics.aalto.fi/events/morphochallenge2010/datasets.shtml`

[5] `http://www.cl.ut.ee/korpused/segakorpus/epl`

[6] `http://www.cl.ut.ee/korpused/morfkorpus/`

procedure is repeated ten times and the average is reported, just as in the MC2010 competition (Kohonen et al., 2010a).

5.4.3 Baseline models

We compare our AG models to several other morphology learning systems. We were able to obtain implementations of two of the best unsupervised systems from MC2010, Morfessor (Creutz and Lagus, 2007) and Morsel (Lignos, 2010), and we use these for comparisons on both the dev and test sets. We also report test results from MC2010 for the only semi-supervised system in the competition, semi-supervised Morfessor (Kohonen et al., 2010a,b). No dev results are reported on this system since for the time of conducting the experiments we were unable to obtain an implementation.[7] This section briefly reviews the systems.

Morfessor Categories-MAP

Morfessor Categories-MAP (CatMAP) (Creutz and Lagus, 2005) is a state-of-the-art unsupervised morphology learning system. Its implementation is freely available[8] and so it is widely used both in tasks that require morphological preprocessing, and as a baseline for evaluating morphology learning systems. CatMAP uses the MDL principle to choose the optimal segment lexicon and the corpus segmentation. Each morpheme in the segment lexicon is labeled as a stem, prefix, suffix or non-morph. The morphotactic rules are encoded as an HMM, which specifies the allowed morpheme sequences with respect to the labels (e.g., a suffix cannot directly follow a prefix).

The morphemes in the segment lexicon can have a hierarchical structure, containing submorphemes that themselves can consist of submorphemes. We hypothesize that this hierarchical structure is one of the key reasons why Morfessor has been so successful. The experiments conducted in this chapter with different grammars also show that the ability to learn latent structures is crucial to learning good segmentations. One essential difference between Morfessor and the proposed AG Select is that while we use the labeled data to choose which levels of the hierarchy are to be used as morphemes, Morfessor makes this decision based on the labels of the segments, choosing the most fine-grained morpheme sequence that does not contain the non-morph label.

Morfessor includes a free parameter, perplexity threshold, which we found can affect the SBF1 score considerably (7 points or more). The best value for this parameter depends on the size of the training set, characteristics of the language being learned, and also the evaluation metric being used, as in some cases the best SBF1 and EMMA scores are obtained with completely

[7] However, the implementation has recently become publicly available.

[8] `http://www.cis.hut.fi/projects/morpho/morfessorcatmapdownloadform.shtml`

different values. Thus, we tuned the value of the perplexity threshold on the labeled set for each language and evaluation metric for different unlabeled training set sizes. This parameter tuning can be regarded similar to the template selection in AG Select. In this respect, Morfessor can be considered, similar to AG Select, a weakly-supervised system.

Semi-supervised Morfessor

Recently, the Morfessor system has been adapted to allow semi-supervised training. Semi-supervised Morfessor (Kohonen et al., 2010a) is based on Morfessor baseline (Creutz and Lagus, 2002, 2007), which also learns using MDL principle. The model consists of a morpheme inventory (lexicon) prior and the data likelihood term, which are combined into posterior that is minimized using heuristic search. Unlike Morfessor CatMAP, the lexicon here is flat—each lexicon element is just a character string. The semi-supervised version maintains separate likelihoods for the labeled and unlabeled data and uses a development set to tune two parameters that weigh these terms with respect to each other and the prior. The following function is minimized:

$$\begin{aligned} L(\boldsymbol{\theta}, \mathbf{z}, \mathbf{D}, \mathbf{D}_A) = &- \log P(\boldsymbol{\theta}) - \alpha \times \log P(\mathbf{D}|\mathbf{z}, \boldsymbol{\theta}) \\ &- \beta \times \log P(\mathbf{D}_A|\mathbf{z}, \boldsymbol{\theta}), \end{aligned} \tag{5.11}$$

where $\boldsymbol{\theta}$ are the morpheme lexicon parameters, $\mathbf{z}$ are the latent segmentations, $\mathbf{D}$ and $\mathbf{D}_A$ are the unlabeled and labeled data respectively, and α and β are the weights of both likelihood components that can be tuned on the development set.

Four versions of the system were evaluated in MC2010, using different degrees of supervision. Results reported here are from the Morfessor S+W system, which performed best of those that use the same kind of labeled data as we do.[9]

Morsel

Morsel (Lignos, 2010) is an unsupervised morphology learning system introduced in MC2010[10]. It is a rule-based system that iteratively processes pairs of words in the corpus and proposes affix transformations based on scores aggregating the affix frequency, transformation frequency and the amount of change effected by the transformation. For instance, the transformation (\$, *ing*) can model the word pairs *help*/*helping*, *learn*/*learning* etc. Morsel learns morphological analyses rather than segmentations; i.e., it might produce *available+ity* as the analysis of *availability*, so it can be evaluated only using EMMA. Morsel implements a special treatment with two different strategies

[9] Morfessor S+W+L performs better but uses training data with morpheme analyses rather than surface segmentations.

[10] `https://github.com/ConstantineLignos/MORSEL`

for compound splitting. Aggressive strategy proposes more compound part candidates during learning while conservative approach delays some of the compound splitting for postprocessing. We used the dev set to choose the best setting in each experimental case.

The MC datasets contain in addition to annotated segmentations also morphological analyses so we could compute Morsel's EMMA scores using the analyses. However, we found that Morsel obtains higher EMMA scores when evaluated against annotated segmentations and thus we used this option in all experiments.[11]

5.4.4 Method

We conducted experiments in two parts. First, we evaluated different aspects of the AG models and compared to all baseline models on the dev set. Then we evaluated the most competitive models on the final test data.

For the development experiments, we compiled unlabeled training sets with sizes ranging from 10k to 50k word types, using the most frequent word forms in each case. For the AG results, we report the average of five different runs made on the same training set. We let the sampler run for 1000 iterations and then collected a single sample for each experiment. No annealing was used as it did not seem to help. The table label resampling option was turned on, and the hyperparameter values were inferred.

We trained all AG and baseline models on each of these training sets. For AG Select, the words from the labeled training set were added to the unlabeled set to allow for template selection. We evaluated all results on the dev set using both transductive and inductive training. Assessing the difference between those learning modes is especially important for AG models where transductive training is very costly compared to the inductive decoding. For evaluating the results in transductive mode, the words from the dev set were also added to the training data. In inductive mode for AG models, the dev set was instead parsed with a CKY parser using the trained posterior grammar.

Preliminary experiments showed that the performance of unsupervised AG and AG Select improved with larger training sets, though the effect is small (see Figure 5.2 left for results of AG Select in transductive mode; the trend in inductive mode is similar). Based on these and similar results with other baseline systems, all results reported later for unsupervised models (AG and baseline) and AG Select were trained using 50k words.

In contrast to the above models, the semi-supervised AG does not always improve with more unlabeled data (see Figure 5.2 right). In the limit, it will match the performance of the same grammar in the unsupervised setting.

[11] EMMA scores for other systems were also computed using the segmentations.

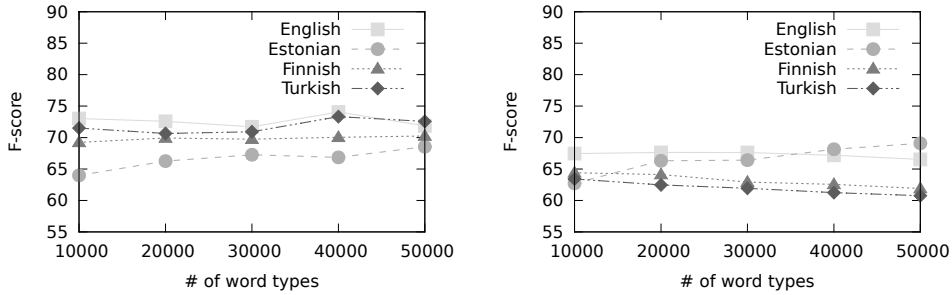

Figure 5.2: Effect of training data size on the dev set SBF1 for AG Select (left) and the semi-supervised SubMorphs grammar (right) in transductive mode.

Other semi-supervised approaches often solve this problem by weighting the labeled data more heavily when estimating model parameters, assuming that each labeled item has been observed more than once. However, duplicating the labeled data does not make sense in the AG framework, because duplicate items will in most cases just be cached at the root (Word) node, providing no additional counts of Morphs (which are where the useful information is). It might be possible to come up with a different way to weight the labeled data more heavily when larger unlabeled sets are used. However for now we instead kept the labeled data the same and tuned the amount of unlabeled data. We used the dev set to choose the amount of unlabeled data (in the range from 10k to 50k types); results for semi-supervised AG are reported using the optimal amount of unlabeled data for each experiment.

5.5 Results

We present the dev set results in Table 5.2(a) for transductive and in Table 5.2(b) for inductive learning. In each table, unsupervised models are shown in the upper section, and the semi-supervised models and AG Select below. Morsel appears only in Table 5.2(a) since it only works transductively. Semi-supervised grammars cannot be trained on German since the MC2010 dataset only provides annotated analyses, not segmentations.

The SubMorphs grammar performs the best of the unsupervised AG models, with the Compounding grammar being only slightly worse. We also tried the Compounding grammar without the submorpheme structures, but the results were even worse than those of MorphSeq. These findings show that the latent structures are important for learning good segmentations.

In all cases, the semi-supervised AGs perform better (often much better) than the corresponding unsupervised grammars. Even though their average scores are not as high as AG Select's, they give the best dev results in many cases. It means that although the grammar for semi-supervised AG must be

Table 5.2: Dev set results for all models in (a) transductive and (b) inductive mode. Unsupervised AG models and baselines are shown in the top part of each table; semi-supervised AG models and grammar selection method are below.

(a) Transductive mode	Border F1-score					EMMA					
	Eng	**Est**	**Fin**	**Tur**	**Avg**	**Eng**	**Est**	**Fin**	**Tur**	**Ger**	**Avg**
AG MorphSeq	61.5	54.0	56.9	59.5	58.0	74.7	74.1	63.7	53.5	59.4	65.1
AG SubMorphs	66.2	66.9	60.5	59.5	63.3	79.1	83.4	66.8	53.4	57.4	68.0
AG Compounding	63.0	64.8	60.9	60.9	62.4	75.4	81.6	65.5	53.7	62.2	67.7
Morfessor	69.5	55.7	65.0	69.3	64.9	**81.3**	75.3	67.8	62.2	**62.7**	69.9
Morsel	-	-	-	-	-	76.8	74.4	66.1	50.1	55.9	64.7
AG ssv MorphSeq	64.4	57.3	63.0	68.9	63.4	74.4	75.9	65.6	59.6	-	-
AG ssv SubMorphs	67.6	**69.1**	64.4	63.4	66.1	79.5	**84.4**	69.2	56.1	-	-
AG ssv Compounding	70.0	67.5	**71.8**	-	-	79.5	82.8	**74.0**	-	-	-
AG Select	**71.9**	68.5	70.2	**72.6**	**70.8**	77.5	81.8	73.2	**63.0**	62.4	**71.6**

(b) Inductive mode	Border F1-score					EMMA					
	Eng	**Est**	**Fin**	**Tur**	**Avg**	**Eng**	**Est**	**Fin**	**Tur**	**Ger**	**Avg**
AG MorphSeq	57.6	54.0	55.4	59.8	56.7	72.0	73.8	62.6	53.7	58.9	64.2
AG SubMorphs	66.1	67.5	61.6	59.8	63.7	78.6	**83.7**	67.4	53.4	56.0	67.8
AG Compounding	62.0	64.8	57.4	61.1	61.3	73.5	81.1	61.9	53.2	61.0	66.2
Morfessor	68.9	51.1	63.5	68.2	62.9	**80.9**	72.0	68.1	60.6	60.8	68.5
AG ssv MorphSeq	64.6	56.9	63.1	**70.3**	63.8	72.7	73.3	65.9	**61.2**	-	-
AG ssv SubMorphs	70.1	**69.7**	66.3	67.9	68.4	80.4	**83.7**	70.5	59.0	-	-
AG ssv Compounding	**70.5**	67.2	**70.0**	-	-	77.3	81.9	70.5	-	-	-
AG Select	69.8	68.8	67.5	70.1	**69.1**	77.3	81.9	**71.1**	59.7	**62.6**	**70.5**

Table 5.3: Test set results for unsupervised baselines Morfessor CatMAP (in transductive and inductive mode) and Morsel; semi-supervised Morfessor; and AG semi-supervised ($\star$ marks the Compounding grammar, † denotes SubMorphs grammar, and $*$ is the MorphSeq grammar) and grammar selection methods. Results are shown for each language, averaged over all languages (when possible: Avg), and averaged over just the languages where scores are available for all systems (-Est, -Est/Ger).

	Border F1-score						**EMMA**						
	Eng	**Est**	**Fin**	**Tur**	**Avg**	**-Est**	**Eng**	**Est**	**Fin**	**Tur**	**Ger**	**Avg**	**-Est/Ger**
Morf. trans	67.3	**73.9**	61.2	57.1	64.9	61.9	78.4	78.8	61.8	49.8	**65.2**	66.8	63.3
Morf. ind	65.7	57.7	60.8	60.1	61.1	62.2	76.5	70.5	59.6	47.0	64.1	63.5	61.0
Morsel	-	-	-	-	-	-	**81.9**	77.2	63.3	47.8	59.0	65.8	**64.3**
Morf. ssv	**77.8**	-	**71.7**	**68.9**	-	**72.8**	80.6	-	62.1	**49.9**	-	-	64.2
AG ssv best	70.3*	68.6†	64.9*	58.2*	65.5	64.5	75.9*	80.3†	61.3*	46.1*	-	-	61.1
AG Select	74.4	71.7	70.0	61.4	**69.4**	68.6	81.3	**81.0**	**64.0**	47.5	63.8	**67.5**	**64.3**

chosen manually, with a suitable choice of the grammar and only a small set of labeled data it can improve considerably over unsupervised AG.

In transductive mode, the AG Select performs the best in several cases. In both transductive and inductive mode, the results of AG Select are close to the best results obtained and are consistently good across all languages. It achieves the best average scores of all models suggesting that the model selection method is robust to different types of morphology and annotation schemes.

Table 5.3 presents the test set results. We include scores for unsupervised Morfessor using both transductive and inductive training, where transductive model trains on the entire unlabeled corpus and inductive model trains on the 50k subset. We took the semi-supervised Morfessor scores from the MC results page[12] after verifying that the evaluation methodology and labeled data used is the same as ours.[13]

There is a good deal of variation between development and test results, indicating that the dev sets may not be a representative sample. The most notable differences are in Turkish, where all models perform far worse on the test than dev set. However, AG Select performs slightly better on the test set for the other languages. Thus, its average SBF1 score improves on the test set and is not much worse than semi-supervised Morfessor. While its average performance drops somewhat on test set EMMA, it is still as good as any other model on that measure. Again, these results support the idea that AG Select is robust to variations in language and dataset.

We also note the surprisingly good performance of Morfessor in transductive mode on Estonian. It could be due to the larger amount of training data used for the test set results, but it is not clear why this would improve performance so much on Estonian and not in the other languages.

5.6 Discussion

To give a sense of what the AG Select model is learning, we provide some examples of both correctly and incorrectly induced segmentations in Table 5.4. These examples suggest that for example in English, M1 is used to model the stem, M21 is for the suffix or the second stem in the compound word, and the rest of the elements in the template are for the remaining suffixes (if any).

Table 5.5 presents examples of some of the most frequently used meta-grammar rules and cached rules for English, together with their relative frequencies. It shows that at the Word level, the binary rule is selected more than three times more frequently than the unary rule. Also, most of the

[12] `http://research.ics.aalto.fi/events/morphochallenge/`

[13] Sami Virpioja, personal communication.

Table 5.4: Examples of segmented words in English (top), Estonian (middle) and Finnish (bottom). Correctly segmented words are in the left part of the table. The identified segments are in brackets indexed by the respective template non-terminal; dots separate the metagrammar generated parse tree leaves. Examples of incorrectly segmented words together with the correct segmentation are on the right.

	Correct Segmentations	**Incorrect Segmentations**	
Word	**Segmentation**	**Induced**	**Correct**
treatable	$[\text{tr.ea.t}]_{\text{M1}}$ $[\text{a.b.le}]_{\text{M21}}$	disagree_s	dis_agree_s
disciplined	$[\text{dis.cip.l.i.n}]_{\text{M1}}$ $[\text{e.d}]_{\text{M21}}$	reduc_e	reduce
monogamous	$[\text{mon.o.g.a.m}]_{\text{M1}}$ $[\text{o.u.s}]_{\text{M21}}$	revalu_e	re_value
streakers	$[\text{st.r.e.a.k}]_{\text{M1}}$ $[\text{e.r}]_{\text{M21}}$ $[\text{s}]_{\text{M2211}}$	derid_e	deride
tollgate	$[\text{t.o.l.l.}]_{\text{M1}}$ $[\text{g.a.t.e}]_{\text{M21}}$ $[\text{s}]_{\text{M2211}}$	accompani_ed	ac_compani_ed
foxhunting	$[\text{f.o.x}]_{\text{M1}}$ $[\text{h.u.n.t}]_{\text{M21}}$ $[\text{ing}]_{\text{M2211}}$	war_y	wary
muscovites	$[\text{m.u.sc.o.v}]_{\text{M1}}$ $[\text{i.t.e}]_{\text{M21}}$ $[\text{s}]_{\text{M2211}}$	indescrib_able	in_describ_able
standardizes	$[\text{st.a.n.d.a.rd}]_{\text{M1}}$ $[\text{i.z.e}]_{\text{M21}}$ $[\text{s}]_{\text{M2211}}$	orat_es	orate_s
slavers'	$[\text{sl.a.v}]_{\text{M1}}$ $[\text{e.r}]_{\text{M21}}$ $[\text{s}]_{\text{M2211}}$ $[\text{'}]_{\text{M2212}}$	alger_ian_s	algeri_an_s
earthiness'	$[\text{e.ar.th}]_{\text{M1}}$ $[\text{i}]_{\text{M2111}}$ $[\text{ness}]_{\text{M2211}}$ $[\text{'}]_{\text{M2212}}$	disput_e_s	dispute_s
instinkt	$[\text{in.st.in.kt}]_{\text{M1}}$	meister_likkust	meisterlikkus_t
rebis	$[\text{re.b.i}]_{\text{M1}}$ $[\text{s}]_{\text{M2}}$	min_a	mina
toitsid	$[\text{to.it}]_{\text{M1}}$ $[\text{s.id}]_{\text{M2}}$	teiste	teis_te
armuavaldus	$[\text{a.rm.u}]_{\text{M11}}$ $[\text{ava.ld.u.s}]_{\text{M12}}$	kuritegu_de_sse	kuri_tegu_desse
määgivale	$[\text{mää.g.i}]_{\text{M11}}$ $[\text{v.a}]_{\text{M12}}$ $[\text{l.e}]_{\text{M2}}$	liharoa_ga	liha_roa_ga
keskuskoulussa	$[\text{kesk.us}]_{\text{M11}}$ $[\text{koul.u}]_{\text{M12}}$ $[\text{s.sa}]_{\text{M2}}$	polte_tti_in	polte_tt_i_in
peruslähteille	$[\text{per.u.s}]_{\text{M11}}$ $[\text{l.ä.ht.e}]_{\text{M12}}$ $[\text{i}]_{\text{M211}}$ $[\text{ll.e}]_{\text{M212}}$	kulttuuri_se_lt_a_kin	kulttuurise_lta_kin
perunakaupoista	$[\text{per.u.n.a}]_{\text{M11}}$ $[\text{k.au.p.o}]_{\text{M12}}$ $[\text{i}]_{\text{M211}}$ $[\text{st.a}]_{\text{M212}}$	tuote_palki_ntoja	tuote_palkinto_j_a
yöpaikkaani	$[\text{yö}]_{\text{M11}}$ $[\text{p.ai.kk.a}]_{\text{M12}}$ $[\text{a}]_{\text{M21}}$ $[\text{n.i}]_{\text{M22}}$	veli_puo_lt_a	veli_puol_ta
nimettäköön	$[\text{ni.m.e}]_{\text{M11}}$ $[\text{tt.ä}]_{\text{M12}}$ $[\text{k.ö}]_{\text{M21}}$ $[\text{ö.n}]_{\text{M22}}$	ota_ttava	otatta_va

Table 5.5: Examples of the most frequently used metagrammar rules and cached rules for English together with their relative frequencies (in percentages).

(a) PCFG rules	
Freq (%)	**Rule**
9.9	Word → M1 M2
5.7	M1 → M11 M12
3.1	Word → M1
2.5	M11 → M111
1.8	M2 → M21
1.4	M12→ M121 M122
1.4	M111 → M1111 M1112
0.9	M12 → M121
0.9	M11 → M111 M112
(b) Cached Subtrees	
Freq (%)	**Cached Rule**
1.2	(M2 (M21 (M211 (M2111 s)))))
0.9	(M2 (M21 (M211 (M2111 e)) (M212 (M2121 d))))
0.7	(M2 (M21 (M211 (M2111 i)) (M212 (M2121 n g))))
0.6	(M2 (M21 (M211 (M2111 e)))
0.4	(M2 (M21 (M211 (M2111 ’))) (M22 (M221 (M2211 s))))
0.3	(M1112 a)
0.3	(M2 (M21 (M211 (M2111 y))))
0.3	(M2 (M21 (M211 (M2111 e))) (M212 (M2121 r)))
0.2	(M2 (M21 (M211 (M2111 a))))

more frequent grammar rules expand the first branch (rooted in M1) into more fine-grained structures. The second branch (M2) is mostly modeled with the unary rule. Among the frequently cached rules we see the common English prefixes and suffixes. One of the most frequent cached rule stores the single letter *e* at the end of a word, which often causes over-segmentation of words ending in *e* (as seen in the incorrect examples in Table 5.4). This problem is common in unsupervised morphological segmentation of English (Goldwater et al., 2006b; Goldsmith, 2001).

We also took a look at the most frequent cached rules learned by the semi-supervised AG with the SubMorphs grammar and observed that Morphs tended to contain only a single SubMorph. This helps to explain why the SubMorphs grammar in semi-supervised AG improved less over the unsupervised AG as compared to the MorphSeq grammar—the rules with only a single SubMorph under the Morph are essentially the same as they would be in the MorphSeq grammar.

Table 5.6: Majority templates for each language. Note that the Estonian annotations contain less fine-grained segmentations than some of the other languages.

	Majority template
English	M1 M21 M2211 M2212 M222
Finnish	M11 M12 M211 M212 M22
Turkish	M11 M12 M211 M212 M22
German	M11 M121 M122 M21 M221 M222
Estonian	M11 M12 M2

Although the semi-supervised AG improved over the unsupervised AG, it did not on average perform as well as AG Select. However, the preliminary experiments with the semi-supervised AG using a three-layer grammar, modeling also morpheme collocations, have shown the potential for further improvements. Perhaps the submorphemes layer alone is too little to exploit the latent inter- and intra-morpheme dependencies fully. Also, using latent layers both above and below the morphemes, the labeled data has the potential to perform a calibrating function for each language, providing examples of the granularity of structures that have to be modeled at the morpheme level.

Finally, we examined the consistency of the templates chosen for each of the five samples during model selection for the test set (Section 5.4.4). We found that there was some variability in the templates, but in most experiments the same template was chosen for the majority of the samples (see Table 5.6). While this majority template is not always the optimal one on the dev set, we observed that it does produce consistently good results. It is possible that using the majority template, rather than the optimal template for the dev set, would actually produce better results on the test set, especially if (as appears to be the case here, and may often be the case in real applications) the dev and test sets are from somewhat different distributions.

It must be noted that both AG Select and semi-supervised AG are computationally more demanding than the comparison systems. Since we do inference over tree structures, the complexity is cubic in the input word length while most segmentation systems are quadratic or linear. Even compared to the unsupervised AG, AG Select is more expensive, because of the larger grammar and number of cached symbols. Nevertheless, our systems can feasibly be run on the large Morpho Challenge datasets by using inductive decoding with posterior grammar.

Other recent unsupervised or semi-supervised systems have reported state-of-the-art results by incorporating additional information from surrounding

words (Lee et al., 2011), multilingual alignments (Snyder and Barzilay, 2008a), or overlapping context features in a log-linear model (Poon et al., 2009), but they have only been run on Semitic languages and English (and in the latter case, a very small corpus). Since they explicitly enumerate and sample from all possible segmentations of each word (often with some heuristic constraints), they could have trouble with the much longer words of the agglutinative languages tested here. In any case, the results are not directly comparable to ours. Recently, another semi-supervised log-linear segmentation model labeling each character as being at the start, end or in the middle of a morpheme was presented in (Ruokolainen et al., 2014).[14] Their results are comparable to ours and in fact they outperform our results in all three languages tested (English, Finnish and Turkish).

The results in this chapter were evaluated using intrinsic evaluation measures that assume that the goal is to learn linguistically correct segmentations. This goal is perhaps tautological and has been mostly motivated by the need for quick and easy evaluation. Recall, that automatic stemmers usually do not strive for linguistic correctness because their performance can be relatively easily evaluated using an information retrieval (IR) task. Until now, there is no such standard task for evaluating different morphological segmentation systems, although the IR task has been tried in Morpho Challenge competition (Kurimo et al., 2009). Some recent work using morphological segmentations in statistical machine translation (SMT) (Salameh et al., 2014) (although focusing on desegmentation task) seems to indicate that SMT could provide mature enough environment to be used as the test-bed for morphological segmentation systems thus eliminating the need to learn linguistically correct segmentations. However, the potential problem with this approach is the same as demonstrated in comparing different morphological segmentation systems in ASR training (Arısoy et al., 2008). Namely, that the internal factors of the downstream task play a much bigger role in the final result than the differences in segmentations, rendering the differences in different segmentation systems negligible.

5.7 Conclusion

In this chapter, we introduced three new methods for Adaptor Grammars and demonstrated their usefulness for morphological segmentation. First, we showed that AG models can be scaled to large data sets by using the posterior grammar for defining an inductive model, that on average results in the same accuracy as full transductive training.

[14] Their paper was published later than (Sirts and Goldwater, 2013), which is the basis for this chapter.

Second, we implemented semi-supervised AG inference, which uses labeled data to constrain the sampler and showed that it performs much better than the unsupervised AG on the same grammar in all cases. Although we used semi-supervised AG only for morphological segmentation in this work, the implementation is general and could also be used for semi-supervised learning of other tasks. Also, it can be employed for fully supervised learning without using any unlabeled data at all. Semi-supervised AG could benefit from labeled data reweighting techniques frequently used in semi-supervised learning. Studying the proper ways of doing so within the AG framework would be a potential topic for future research.

The final contribution is the AG Select method, where the initial model is trained using a very general grammar that oversegments the data, and the labeled data is used to select the optimal granularity of segments. Unlike other morphological segmentation models, this method can adapt its grammar to languages with different structures, rather than having to use the same grammar for every language. Indeed, we found that AG Select performs well across a range of languages and also seems to be less sensitive to differences between datasets (here, dev vs. test). In addition, it can be trained on either morphological analyses or segmentations. Although we tuned all results to optimize the segment boundary F1-score, in principle, the same method could be used to optimize other measures, including extrinsic measures on downstream applications such as machine translation or information retrieval. It can also potentially be useful for related segmentation tasks such as stemming or syllabification. Also, the method itself could potentially be improved by designing a classifier to determine the best template for each word based on a set of features, rather than using a single template for all words in the language.

The presented results provide support for Claim A as the latent morpheme substructures improve the results over the grammar modeling flat morpheme sequences only. Both target morpheme structures and auxiliary sub- and super-structures are learned jointly and so the superiority of the hierarchical grammars also supports Claim B as the different grammar layers learned jointly influence each other. Finally, in terms of exploratory linguistic research it would be interesting to study which of the latent sub- or superstructures help most to improve the segmentation and why.

Chapter 6

Morphosyntactic clustering

This chapter presents an unsupervised morphosyntactic clustering model based on the distance-dependent Chinese restaurant process (ddCRP) using distributional and morphological cues. We experiment with three different ddCPR-based models and provide clustering results on 11 languages. The chapter is partly based on publication III (Sirts et al., 2014). It also contains unpublished experimental results on other languages than English and the analysis of those results.

6.1 Introduction

Morphological analysis is the task of labeling each word token with detailed morphosyntactic information. Proper morphological analysis can only be done either rule-based or using supervised learning techniques because the morphological categories and their possible values are different in each language. Morphosyntactic clustering can be considered an unsupervised alternative to the morphological analysis. The goal of this task is to cluster together words sharing the same morphosyntactic function. The correct labels identifying the common morphosyntactic function are, in this case, of course, unknown. However, in the potential downstream tasks, such a dependency parsing or statistical machine translation, it is not the specific label that is important, but rather that words with different labels can be distinguished.

The morphosyntactic function of words is reflected in both their distributional properties as well as their morphological structure. These information sources display complementary aspects of the morphosyntax. Distributional similarity varies smoothly with syntactic function so that words with similar syntactic functions occur in similar distributional contexts. However, distributional information may not be sufficient for differentiating between words in the same syntactic class performing a different morphosyntactic function.

For instance, English verbs in the present and past tense can occur in similar distributional context (e.g. I *look* at the picture vs. I *looked* at the picture). On the other hand, the present and past tense verbs (at least the regular ones) can be easily differentiated by looking at their morphological suffixes. Also, learning the accurate distributional properties requires observing each word type in a large number of contexts, and this may not be feasible for rare words even when using very large corpora. On the other hand, once the morphological regularities have been learned, they can be applied to any word regardless of how often it is used. Moreover, morphologically irregular words tend to be more frequent and thus their distributional properties can be learned more reliably, whereas rare words are usually morphologically regular and thus can be more reliably analyzed using morphological rules.

These observations suggest that a general approach to the induction of morphosyntactic categories should exploit both distributional and morphological features (Clark, 2003; Christodoulopoulos et al., 2010). However, these features may have disparate representations and thus combining them can be difficult. Distributional information is typically represented in numerical vectors, and recent work has demonstrated the utility of continuous vector representations, or *embeddings* (Mikolov et al., 2013; Luong et al., 2013; Kim and de Marneffe, 2013; Turian et al., 2010). In contrast, morphology is often represented in terms of sparse, discrete features (such as morphemes), or via pairwise measures such as string edit distance. However, morphological processes generating surface forms are complex, so that simple metrics such as edit distance will only weakly approximate morphological similarity.

The classic HMM approach represents distributional information with n-grams over syntactic categories, and several ways have been proposed to combine it with morphological features (Clark, 2003; Hasan and Ng, 2009; Lee et al., 2010; Berg-Kirkpatrick et al., 2010; Blunsom and Cohn, 2011; Sirts and Alumäe, 2012). However, as argued above, learning accurate distributional representations requires large amounts of data, whereas unsupervised HMMs can be trained on corpora of limited size due to the computational requirements. Continuous word embeddings, on the other hand, can be pretrained on large corpora and the same pretrained embeddings can be reused over and over again for different models and tasks.

In this chapter we present a new approach for inducing fine-grained part-of-speech (POS) classes, combining morphological and distributional information in a non-parametric Bayesian model based on the *distance-dependent Chinese restaurant process* (ddCRP; Blei and Frazier, 2011). In the ddCRP, each data point (word type) selects another point to "follow"; this chain of following links corresponds to a partition of the data points into clusters. The probability of a word w_1 following w_2 depends on two factors: 1) the *distributional* similarity between all words in the proposed partition

containing w_1 and w_2, which is encoded using a Gaussian likelihood function over the word embeddings; and 2) the *morphological* similarity between w_1 and w_2, which acts as a prior distribution on the induced clustering. We use a log-linear model in the ddCRP prior to capture suffix similarities between words, and learn the feature weights by iterating between sampling and weight learning.

We apply the model to the languages in the Multext-East (MTE) corpus (Erjavec, 2004), whose annotations contain both coarse-grained syntactic tags and fine-grained morphosyntactic tags. The model infers a variable number of clusters from the data that in all cases is somewhere between the sizes of the coarse and fine reference tagsets. Therefore, we evaluate against both although because of the model semantics the main attention is on the fine-grained tagset. We compare several variants of the ddCRP model against K-means clustering and an infinite Gaussian mixture model. We find that in English, the model effectively combines morphological features with distributional similarity, outperforming the baseline models. The results on other, morphologically more complex MTE languages are not as good. We suggest that this is mainly because the log-linear model in the ddCRP prior was too simple for more complex languages, and the corpora on which the word embeddings were trained were not large enough.

The structure of this chapter is as follows: we start with a short overview about the word embeddings in section 6.2. In section 6.3, we first explain the finite Gaussian mixture model (GMM), and then extend it into Bayesian GMM and finally describe the non-parametric version of it—the infinite GMM. Section 6.4 first explains the distance-dependent CRP prior and then describes the proposed model, which is essentially an infinite GMM with distance-dependent CRP prior distribution. The experimental setup is detailed in section 6.5, the results of which are given in section 6.6. Then, in section 6.7, we discuss the weaknesses of the model and propose several reasons for why the model performed well on English but failed on other languages. Finally, we conclude the chapter in section 6.8 by reviewing how the experimental results, although partly negative, support the claims of this dissertation.

6.2 Word embeddings

The idea of using distributional word embeddings has been around for quite some time already, and many training methods for different purposes have been proposed, e.g. LSA (Deerwester et al., 1990; Hofmann, 1999), Brown embeddings (Brown et al., 1992), LDA (Blei et al., 2003). More recently, neural network models have become popular for training word embeddings (Turian et al., 2010; Mikolov et al., 2013; Al-Rfou et al., 2013; Huang et al., 2012; Levy and Goldberg, 2014).

The choice of particular embeddings, as well as their quality, is crucial to the task at hand. However, the space of different options is huge, and so far there are no well-established best practices to rely on. So we first just tried a few different options during preliminary experiments[1] and then chose to proceed with the pretrained Polyglot embeddings (Al-Rfou et al., 2013). The dimensionality of the embeddings is potentially significant as well, but as we did not have much wisdom in this matter we just accepted the dimensionality of the pretrained embeddings, which is 64. These somewhat arbitrary choices seem to suit well for English as shown later in the results section. However, for other, morphologically more complex languages, more research is needed to understand how to train embeddings that capture the desired morphosyntactic properties.

6.3 Infinite Gaussian mixture model

A typical scenario for exploiting the embeddings involves using them as features in a supervised learning task, such as chunking, named entity recognition (Turian et al., 2010) or POS tagging (Al-Rfou et al., 2013). In this work, we utilize the embeddings differently—instead of viewing them as latent feature vectors we treat them as multivariate Gaussian random variables in a generative probabilistic model. We assume that the distribution over the word embeddings is a mixture of multivariate Gaussians with the number of mixture components corresponding to the number of morphosyntactic clusters in the data, but the exact number of clusters being unknown. This setup motivates fitting an infinite Gaussian mixture model to the data that enables clusters with different parameters and infers the number of components dynamically.

The infinite Gaussian mixture model (IGMM) (Rasmussen, 2000) is a mixture model with Gaussian component distributions and a Dirichlet Process (DP) prior over the mixing distribution. In this section, we will first introduce the Gaussian mixture model. Then we describe its extension into Bayesian statistics that places prior distributions over parameters. Finally, we extend it into non-parametric Bayesian domain by introducing the infinite DP prior over mixture parameters.

6.3.1 Gaussian mixture model

The Gaussian mixture model (GMM) (Reynolds, 2009) is a very common model for expressing complicated multimodal distributions over continuous random variables. For example, we might have a non-symmetric continuous distribution with three modes. In such a case, a GMM with three components would be a reasonable modeling choice for approximating this data.

[1] First LSA and then pretrained neural embeddings of Mikolov et al. (2011)

$$P(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{i=1}^{n} P(\mathbf{x}_i|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{6.1}$$

$$= \prod_{i=1}^{n} \sum_{k=1}^{K} P(\mathbf{x}_i, z_i = k|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{6.2}$$

$$= \prod_{i=1}^{n} \sum_{k=1}^{K} P(\mathbf{x}_i|z_i = k, \boldsymbol{\mu}, \boldsymbol{\Sigma}) P(z_i = k|\boldsymbol{\pi}) \tag{6.3}$$

$$= \prod_{i=1}^{n} \sum_{k=1}^{K} \mathcal{N}(\mathbf{x}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\pi_k \tag{6.4}$$

$$P(\mathbf{X}, \mathbf{z}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{i=1}^{n} P(\mathbf{x}_i|z_i = k, \boldsymbol{\mu}, \boldsymbol{\Sigma}) P(z_i = k|\boldsymbol{\pi}) \tag{6.5}$$

$$= \prod_{i=1}^{n} \mathcal{N}(\mathbf{x}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\pi_i \tag{6.6}$$

Figure 6.1: Gaussian mixture model

In a GMM with K mixture components, there are two types of distributions: a discrete mixing distribution and K Gaussian distributions that generate data points from specific mixture components. The relevant equations are given in Figure 6.1.

In a GMM, each data point $\mathbf{x}_i$ is assumed to be conditionally independent of all other data points given the responsible mixture component z_i. This conditional independence assumption allows the complicated joint distribution over the full dataset $\mathbf{X}$ to be expressed in relatively simple terms (6.2). On the right-hand side of the conditioning bar are the mixing distribution $\boldsymbol{\pi}$, the collections of component means $\boldsymbol{\mu}$ and covariances $\boldsymbol{\Sigma}$.

The joint probabilities can be factored further as in (6.3). In (6.4), π_k is just the kth component in the discrete mixing distribution and for the multivariate Gaussian probability density function see Appendix A.1. When performing clustering, we do not want to marginalize out the cluster indicator variables. Rather, we are interested in the joint distribution over the observed variables $\mathbf{X}$ and the latent mixture indicators $\mathbf{z}$ as in (6.5) and (6.6).

The maximum-likelihood parameters for the GMM model are typically estimated with the Expectation-Maximization algorithm (Dempster et al., 1977; Reynolds, 2009).

$$P(\mathbf{X}, \mathbf{z}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}|\Phi) = P(\mathbf{X}, \mathbf{z}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi})P(\boldsymbol{\pi}|\Phi)P(\boldsymbol{\mu}, \boldsymbol{\Sigma}|\Phi) \tag{6.7}$$

$$\boldsymbol{\pi}|\Phi \sim \text{Dirichlet}(\alpha) \tag{6.8}$$

$$\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k|\Phi \sim \mathcal{NIW}(\mathbf{m}_0, \kappa_0, \nu_0, \boldsymbol{\Lambda}_0), \quad k = 1 \dots K \tag{6.9}$$

Figure 6.2: Bayesian Gaussian mixture model

6.3.2 Bayesian GMM

The GMM can be extended into Bayesian setting by adding priors to the component distribution parameters—Gaussian means and covariances—and mixing distribution parameters. The priors for component parameters and mixing parameters are independent of each other, and they can be treated separately. The model is given formally in Figure 6.2.

The first term in (6.7) is the same as equation (6.6), except that $\boldsymbol{\pi}$, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are now random variables. The other two terms represent the prior probabilities for the model parameters. Φ includes all hyperparameters of the model. The discrete mixing distribution $\boldsymbol{\pi}$ given a symmetric Dirichlet prior with the concentration parameter α. The mean and covariance pairs for each cluster are given normal-inverse-Wishart prior distributions (for more information and an explanation of the hyperparameters, see Appendix A.4).

Inference

Exact inference in the Bayesian GMM (BGMM) is, in general, intractable. A standard approach is to simulate samples from the posterior distribution using Gibbs sampling where the value of each latent variable is sampled in turn from its conditional posterior distribution. For the BGMM, this means sampling the latent cluster assignments and the model parameters in turn:

$$P(z_i = k|\mathbf{X}, \mathbf{z}_{-i}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}, \Phi) \propto P(\mathbf{x}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)P(z_i = k|\boldsymbol{\pi}), i = 1, \dots, n \tag{6.10}$$

$$P(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k|\mathbf{X}, \mathbf{z}, \boldsymbol{\pi}, \Phi) \propto P(\mathbf{X}_k|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)P(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k|\Phi), k = 1 \dots K \tag{6.11}$$

$$P(\boldsymbol{\pi}|\mathbf{X}, \mathbf{z}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \Phi) \propto P(\mathbf{z}|\boldsymbol{\pi})P(\boldsymbol{\pi}|\Phi) \tag{6.12}$$

The subscript $-i$ in (6.10) excludes the i-th variable from the respective variable collection and $\mathbf{X}_k$ in (6.11) denotes the set of points belonging to the k-th cluster. The posteriors in (6.11) and (6.12) can be sampled analytically, because the Dirichlet prior is conjugate to the discrete likelihood and the normal-inverse-Wishart prior is conjugate to the normal likelihood

leading to Dirichlet and normal-inverse-Wishart posteriors respectively (see Appendix A.5). However, if the model parameters themselves are not of interest it is common in Bayesian inference to integrate them out altogether:

$$P(z_i = k|\mathbf{X}, \mathbf{z}_{-i}, \Phi) \propto \int_{\boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}} P(\mathbf{x}_i|\mathbf{X}_k^{-i}, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) P(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k|\Phi) d\boldsymbol{\mu}_k d\boldsymbol{\Sigma}_k \times \int_{\boldsymbol{\pi}} P(z_i = k|\mathbf{z}_{-i}, \boldsymbol{\pi}) P(\boldsymbol{\pi}|\alpha) d\boldsymbol{\pi} \tag{6.13}$$

Here the notation $\mathbf{X}_k^{-i}$ denotes the set of points currently belonging to the kth cluster but excluding the ith point.

Integrating out the Gaussian mean and covariance parameters leads to a multivariate Student-T distribution (see Appendix A.6). The prior probability for the cluster assignment also has a closed form solution (see, for example, Goldwater (2006) section 2.2.3 for derivation):

$$\int_{\boldsymbol{\pi}} P(z_i = k|\mathbf{z}_{-i}, \boldsymbol{\pi}) P(\boldsymbol{\pi}|\alpha) d\boldsymbol{\pi} = P(z_i = k|\mathbf{z}_{-i}, \alpha) = \frac{n_k^{-i} + \alpha}{n - 1 + K\alpha}, \tag{6.14}$$

where n is the total number of data points and n_k^{-i} is the number of points in the kth cluster excluding the ith point.

6.3.3 Bayesian GMM with infinite prior

Although the Bayesian GMM is more flexible than the regular one, it still poses a strict limitation to the model, namely that the number of components (clusters) K must be fixed in advance. In many NLP applications and also in the fine-grained morphosyntactic induction, in particular, there are not enough justifications to build such a strong bias into the model. A more reasonable option would be to set a non-parametric prior distribution over the mixing distribution that enables emerging posterior distributions with arbitrary dimensionality. The particular number of clusters chosen at the inference time would then be determined by the data.

The GMM can be made infinite by replacing the finite Dirichlet prior over the mixing distribution with a Dirichlet Process prior (Rasmussen, 2000) (see Figure 6.3). H is a base distribution over clusters, which is typically a uniform distribution.

Inference

Inference in the IGMM model with a Gibbs sampler is essentially the same as in the finite Bayesian GMM. The only significant difference stems from the fact that in the IGMM, each resampled point can consider starting a new

$$P(\mathbf{X}, \mathbf{z}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}|\Phi) = P(\mathbf{X}, \mathbf{z}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi})P(\boldsymbol{\pi}|\Phi)P(\boldsymbol{\mu}, \boldsymbol{\Sigma}|\Phi) \tag{6.15}$$

$$\boldsymbol{\pi}|\Phi \sim \mathrm{DP}(\alpha, H) \tag{6.16}$$

$$\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k|\Phi \sim \mathcal{NIW}(\mathbf{m}_0, \kappa_0, \nu_0, \boldsymbol{\Lambda}_0), \quad k = 1 \ldots \infty \tag{6.17}$$

Figure 6.3: Infinite Gaussian mixture model

cluster. Thus, if the number of instantiated clusters in the state space is K then the sampling distribution has $K+1$ dimensions containing a probability for each of the existing K clusters plus an additional probability for a new cluster. The prior probabilities for the cluster assignments will be computed according to the CRP formulas (see section 3.1.2). The prior probability of the ith point belonging to an existing cluster k is:

$$P(z_i = k|\mathbf{z}_{-i}, \Phi) = \frac{n_k^{-i}}{n - 1 + \alpha}, \tag{6.18}$$

where n_k^{-i} is the number of points in the kth cluster excluding the ith point, n is the total number of points and α is the concentration parameter of the DP prior. The probability of starting a new cluster is:

$$P(z_i = k_{new}|\mathbf{z}_{-i}, \Phi) = \frac{\alpha}{n - 1 + \alpha}, \tag{6.19}$$

The likelihood terms will be computed exactly as in the finite Bayesian GMM.

6.3.4 Morphosyntactic clustering with IGMM

We can use the IGMM to perform clustering on word embeddings. In addition to the information encoded in the embeddings, the DP prior in the IGMM generates a power-law-like distribution over clusters. However, the success of the clustering relies solely on the properties of the embeddings. If the embeddings encode the morphosyntactic information and if the distributions over the words belonging to the same morphosyntactic class approximately resemble Gaussians then we can expect to learn sensible clusters, otherwise our attempts will be fruitless. Evidence that the word embeddings indeed contain some amount of syntactic information has been provided by Mikolov et al. (2013) and Al-Rfou et al. (2013). Although the word embeddings distributions' resemblance to the Gaussians is hard to check due to the high dimensionality, we make this assumption here.

6.4 IGMM with distance-dependent CRP prior

Next we change the IGMM model by replacing the DP prior with a distance-dependent CRP (ddCRP) (Blei and Frazier, 2011). The distance-dependent CRP uses distances (or similarities) between data points to define probabilities, and this allows bringing in another source of information. We will first describe the ddCRP formally, and then explain how to do inference with this model and finally present details of the proposed ddCRP model for the morphosyntactic clustering.

6.4.1 Distance-dependent CRP

The ddCRP (Blei and Frazier, 2011) is an extension of the regular CRP; like the CRP, it defines a distribution over partitions ("table assignments") of data points ("customers"). Whereas in the regular CRP each customer chooses a table with probability proportional to the number of customers already sitting there, in the ddCRP each customer chooses another customer to follow and sits at the same table with that customer. By identifying the connected components in the resulting follower graph, ddCRP equivalently defines a prior over clusterings.

If c_i is the index of the customer followed by the ith customer, then the ddCRP prior can be written as:

$$P(c_i = j|, f, d, \alpha) \propto \begin{cases} f(d_{ij}) & \text{if } i \neq j \\ \alpha & \text{if } i = j, \end{cases} \tag{6.20}$$

where d_{ij} is the distance between customers i and j and f is a decay function that transforms the distances to unnormalized probabilities such that the probabilities decrease monotonically with the increasing distance. A ddCRP is *sequential* if customers can only follow previous customers, i.e., $d_{ij} = \infty$ when $i > j$ and $f(\infty) = 0$. In this case, if $d_{ij} = 1$ for all $i < j$ then the ddCRP reduces to the regular CRP.

Separating the distance and decay function makes sense for "natural" distances (e.g., the number of words between the ith and jth word in a document, or the time between two events), but they can also be collapsed into a single similarity function. In the context of morphosyntactic clustering, we wish to assign higher similarities to pairs of words that share meaningful suffixes. Because we do not know which suffixes are meaningful *a priori*, we use a maximum entropy model[2] whose features include all suffixes up to length three that are shared by at least one pair of words. The prior is then:

[2] Also known as the multiclass logistic regression model or log-linear model.

$$P(c_i = j|\mathbf{w}, \alpha) \propto \begin{cases} e^{\mathbf{w}^\top \mathbf{g}(i,j)} & \text{if } i \neq j \\ \alpha & \text{if } i = j, \end{cases} \tag{6.21}$$

where $\mathbf{g}$ is the feature function assigning $g_s(i, j)$ to 1 if suffix s is shared by ith and jth words, and 0 otherwise, and $\mathbf{w}$ is the weight vector.

The ddCRP has been used before in NLP. In the introducing paper, Blei and Frazier (2011) used it for language modeling and document clustering. In both of these tasks, the distances were derived from the spatial distances between words in text or textual documents in time respectively. Socher et al. (2011) used the ddCRP for detecting hand-written digits and clustering newsgroups articles. In their model, they again derived the distances from the data itself by projecting it into lower-dimensional spectral space and then computing the Euclidean distances between the projections. In the work of Titov and Klementiev (2012), the ddCRP model was used for unsupervised semantic role labeling. Different from the previous work and similar to our model their distance function is learned. However, the learned parameters in their model are the distances themselves while we use a more flexible feature-based model and learn the feature weights. Models using linkage structure resembling ddCRP have been used for coreference resolution (Haghighi and Klein, 2010; Durrett and Klein, 2013; Andrews et al., 2014) where the links point to the mentions of the same entity in the text.

6.4.2 Inference with the ddCRP

The essential component in the ddCRP model is the latent follower structure: a graph consisting of directed links between points. The inference with Gibbs sampling involves repeatedly resampling those links from the respective conditional posterior distributions. To explain it, consider first the familiar Gibbs sampling with the IGMM model, where the sampling of the cluster assignments proceeds as follows. Choose a point, remove it from the corresponding cluster and pretend it has not been generated yet. Then compute the probabilities of adding this point into each possible cluster, normalize and sample the new cluster assignment. With the ddCRP, the inference is conceptually a bit different. We again choose a point $\mathbf{x}_i$, linked to the point $\mathbf{x}_j$ and belonging to a cluster k, but only *remove the link* $c_i = j$ from the configuration while the point $\mathbf{x}_i$ itself *stays in the state space*. Then we compute the probabilities of the links from $\mathbf{x}_i$ to every possible point including itself, normalize and resample the new link.

When resampling the link probabilities we have to take into account the changes in the clustering caused by the removal of the old link and the addition of the new link. First, denote by $C_{\mathbf{x}}$ the cluster that contains the point $\mathbf{x}$. By removing the old link $c_i = j$ two scenarios can occur: 1) if

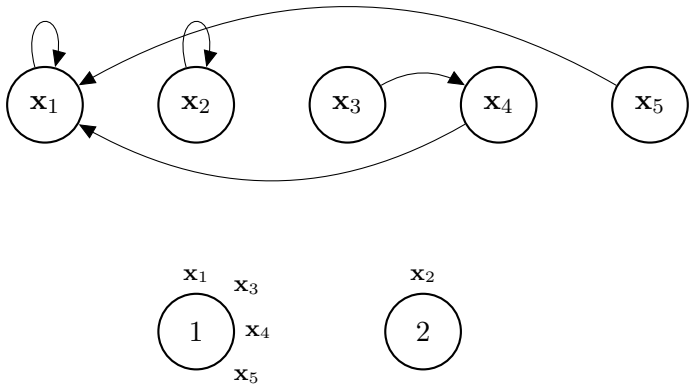

Figure 6.4: An example follower structure and the clustering induced by this follower structure.

after the link removal the points $\mathbf{x}_i$ and $\mathbf{x}_j$ are still in the same cluster, i.e. $k = C_{\mathbf{x}_i} = C_{\mathbf{x}_j}$ then the clustering remains the same;[3] 2) removing the link breaks the cluster k into two, such that $k = C_{\mathbf{x}_j}$, $C_{\mathbf{x}_i} \neq C_{\mathbf{x}_j}$. In either case, after removing the link the cluster $C_{\mathbf{x}_i}$ consists of the block of points fol(i) who follow the point $\mathbf{x}_i$, including $\mathbf{x}_i$ itself. When adding a new link $c_i = l$ from the point $\mathbf{x}_i$ to a point $\mathbf{x}_l$, there are again two possible scenarios: 1) the point $\mathbf{x}_l \in \text{fol}(i)$, in which case the link forms a cycle, and the clustering remains the same; 2) the link is directed to a point $\mathbf{x}_l \notin \text{fol}(i)$, in which case the clusters $C_{\mathbf{x}_i}$ and $C_{\mathbf{x}_l}$ will be merged.

We can compute the probability of a new clustering by starting with the likelihood of the old clustering, dividing out the changes caused by the old link removal and multiplying in the changes effected by the new link addition. Because the old likelihood as well as the probability caused by removing the old link are the same, regardless of where the new link is directed to, we can omit these two terms. Hence, we only consider the changes caused by creating the new link $c_i = l$, which is equal to the probability of adding the points fol(i) to the cluster $C_{\mathbf{x}_l}$.

For an example consider the initial situation pictured in Figure 6.4. Suppose we are resampling the link emanating from the point $\mathbf{x}_4$ belonging to the cluster 1 and with the old link $c_4 = 1$. Removing this link splits the cluster 1 into two. The new cluster 1 only contains the points $\mathbf{x}_1$ and $\mathbf{x}_5$ and the new tentative cluster $C_{\mathbf{x}_4}$ contains the points $\mathbf{x}_3$ and $\mathbf{x}_4$, which form the set of points fol(4) that follow $\mathbf{x}_4$ including $\mathbf{x}_4$ itself. We start from this situation when constructing the sampling distribution of the links to each

[3]This scenario always occurs with the self-links when $c_i = i$ but also, when the links making up the cluster form a cycle.

of the five points. Directing the new link to the points $\mathbf{x}_1$ or $\mathbf{x}_5$ causes the tentative cluster $C_{\mathbf{x}_4}$ to merge again with the cluster 1. Choosing to follow $\mathbf{x}_2$ merges $C_{\mathbf{x}_4}$ with the cluster 2. When the new link is directed to either $\mathbf{x}_3$ or $\mathbf{x}_4$, then the tentative cluster becomes a real cluster 3, and we compute the probability of forming such a cluster. By denoting the current follower structure with $\mathcal{F}$, with the subscript $-i$ meaning that the old link $c_i = j$ has been excluded, and including the prior probability of the new link, the unnormalized sampling distribution can be computed with the following set of formulas:

$$P(c_4 = j|\mathbf{X}, \mathcal{F}_{-4}, \Phi, \Theta)_{j\in\{1,5\}} \propto P(\text{fol}(4)|\mathbf{x}_1, \mathbf{x}_5, \Phi) \times P(c_4 = j|\Theta) \quad (6.22)$$

$$P(c_4 = j|\mathbf{X}, \mathcal{F}_{-4}, \Phi, \Theta)_{j\in\{2\}} \propto P(\text{fol}(4)|\mathbf{x}_2, \Phi) \times P(c_4 = j|\Theta) \quad (6.23)$$

$$P(c_4 = j|\mathbf{X}, \mathcal{F}_{-4}, \Phi, \Theta)_{j\in\text{fol}(4)} \propto P(\text{fol}(4)|\Phi) \times P(c_4 = j|\Theta), \quad (6.24)$$

where $\mathbf{X}$ is the set of all points $\mathbf{x}_1, \ldots, \mathbf{x}_5$, Φ denotes the hyperparameters of the Gaussian clusters and Θ includes the parameters and hyperparameters of the ddCRP prior.

6.4.3 Morphosyntactic clustering with ddCRP

The ddCRP model enables combining several sources of data by using different features in the likelihood and the prior. We use the Gaussian likelihood to cluster the real-valued word embeddings and the features derived from suffixes for expressing the similarities between words in the ddCRP prior. This approach is motivated by three main assumptions: 1) The word embeddings encode the morphosyntactic information at least to some extent; 2) The distribution over word embeddings within a morphosyntactic class is approximately Gaussian; 3) The suffixes encode information about the word's morphosyntactic role. The first two assumptions were already mentioned in the context of the IGMM in section 6.3.4. The third assumption seems intuitively to make sense, although later we will argue that matters might be more complicated than they initially seem.

We model the Gaussian likelihood as described in Appendix A. The ddCRP prior using the suffix features as described in section 6.4.1 requires setting the weight vector that we learn iteratively during inference as will be explained in the next section. In this work, we only use suffix features up to three characters plus a zero suffix for the intercept term. The features are extracted from pairs of words, and a feature will be fired if both words in the pair share the suffix encoding that feature. For instance the pair of words *reading-singing* fires four features: the empty suffix and the suffixes *-g*, *-ng* and *-ing*. The pair *big-cling* fires in addition to the null suffix only the feature corresponding to the suffix *-g* and the pair *table-book* only activates the empty suffix feature.

We implemented three different ddCRP priors: *ddCRP uniform* that uses the uniform similarity function, *ddCRP learned* that learns the weights for the log-linear similarity function, and *ddCRP exp* that adds an additional exponentiating parameter to the *ddCRP learned* model. All these ddCRP models are non-sequential (Socher et al., 2011), allowing cycles to be formed in the follower link structure.

ddCRP uniform

In the ddCRP uniform model, the similarities between each two data points are set equal to one. Thus, the probability for any customer i to follow any other customer j is in sequential case equal to:

$$P(c_i = j)_{i>j} = \frac{1}{i - 1 + \alpha} \tag{6.25}$$

and in the non-sequential case:

$$P(c_i = j) = \frac{1}{N - 1 + \alpha} \tag{6.26}$$

It is easy to see that the sequential case recovers the original IGMM with the standard CRP prior. The prior probability of the ith customer belonging to a particular cluster (sitting at a particular table) is computed by summing the probabilities of following any data point in that cluster:

$$\begin{aligned} P(z_i = k) &= \sum_{j:z_j=k,i>j} P(c_i = j) = \sum_{j:z_j=k,i>j} \frac{1}{i - 1 + \alpha} \\ &= \sum_{j=1}^{n_k} \frac{1}{i - 1 + \alpha} = \frac{n_k}{i - 1 + \alpha}, \end{aligned} \tag{6.27}$$

where n_k is the number of points in $x_1, \ldots, x_{i-1}$ belonging to the cluster k. Although the non-sequential model is not exactly equivalent to the IGMM, we nevertheless expect it to perform similarly.

ddCRP learned

In the ddCRP learned model, the similarity between each two data points is expressed with a log-linear model. Using this kind of similarity function requires setting the weights of the features. As we do not know a priori the weight vector nor which features are relevant, we learn the weight vector iteratively during inference as a supervised learning task. We minimize the negative log-likelihood of the follower structure $\mathcal{F} = \{(i, j) : c_i = j, i = 1 \ldots n\}$ which, assuming a fixed $\mathcal{F}$, leads to a convex objective function:

$$
\begin{aligned}
J(\mathcal{F}, \mathbf{w}) &= -\log \prod_{\substack{i,j \in \mathcal{F} \\ i \neq j}} P(c_i = j|\mathbf{w}) = -\sum_{\substack{i,j \in \mathcal{F} \\ i \neq j}} \log P(c_i = j|\mathbf{w}) \\
&= -\sum_{\substack{i,j \in \mathcal{F} \\ i \neq j}} \log \frac{\exp\left(\mathbf{w}^T \mathbf{g}(i,j)\right)}{\sum_{\substack{l=1 \\ l \neq i}}^{n} \exp\left(\mathbf{w}^T \mathbf{g}(i,l)\right)} \qquad (6.28) \\
&= -\sum_{\substack{i,j \in \mathcal{F} \\ i \neq j}} \left(\log \exp\left(\mathbf{w}^T \mathbf{g}(i,j)\right) - \log \sum_{\substack{l=1 \\ l \neq i}}^{n} \exp\left(\mathbf{w}^T \mathbf{g}(i,l)\right) \right) \\
&= -\sum_{\substack{i,j \in \mathcal{F} \\ i \neq j}} \mathbf{w}^T \mathbf{g}(i,j) + \sum_{\substack{i=1 \\ c_i \neq i}}^{n} \log \sum_{\substack{l=1 \\ l \neq i}}^{n} \exp\left(\mathbf{w}^T \mathbf{g}(i,l)\right)
\end{aligned}
$$

For minimizing this function with some off-the-shelf optimization routine we need to be able to calculate the gradients with respect to the weight vector components w_k:

$$
\begin{aligned}
\frac{\partial J(\mathcal{F}, \mathbf{w})}{\partial w_k} &= -\sum_{\substack{i,j \in \mathcal{F} \\ i \neq j}} \mathbf{g}(i,j)_k + \sum_{\substack{i=1 \\ c_i \neq i}}^{n} \frac{\frac{\partial}{\partial w_k} \sum_{\substack{l=1 \\ l \neq i}}^{n} \exp\left(\mathbf{w}^T \mathbf{g}(i,l)\right)}{\sum_{\substack{l=1 \\ l \neq i}}^{n} \exp\left(\mathbf{w}^T \mathbf{g}(i,l)\right)} \\
&= -\sum_{\substack{i,j \in \mathcal{F} \\ i \neq j}} \mathbf{g}(i,j)_k + \sum_{\substack{i=1 \\ c_i \neq i}}^{n} \frac{\sum_{\substack{l=1 \\ l \neq i}}^{n} \left(\exp\left(\mathbf{w}^T \mathbf{g}(i,l)\right) \frac{\partial \mathbf{w}^T \mathbf{g}(i,l)}{\partial w_k} \right)}{\sum_{\substack{l=1 \\ l \neq i}}^{n} \exp\left(\mathbf{w}^T \mathbf{g}(i,l)\right)} \qquad (6.29) \\
&= -\sum_{\substack{i,j \in \mathcal{F} \\ i \neq j}} \mathbf{g}(i,j)_k + \sum_{\substack{i=1 \\ c_i \neq i}}^{n} \frac{\sum_{\substack{l=1 \\ l \neq i}}^{n} \mathbf{g}(i,l)_k \exp\left(\mathbf{w}^T \mathbf{g}(i,l)\right)}{\sum_{\substack{l=1 \\ l \neq i}}^{n} \exp\left(\mathbf{w}^T \mathbf{g}(i,l)\right)},
\end{aligned}
$$

where the subscript k in $\mathbf{g}(i,j)_k$ chooses the kth component of the feature vector.

We used the L-BFGS optimization method provided in the Python scipy package to learn the weights. We perform Monte Carlo Expectation-Maximization (Wei and Tanner, 1990) by intermittently optimizing the weights and resampling the follower structure. Therefore, each time the weights are learned using the follower structure resulting from the last Gibbs iteration.

ddCRP exp

The ddCRP Gibbs sampler is a blocked sampler by nature because each time a point moves to another cluster all the points following that point move along. This block of points can be potentially very large, and this can cause the likelihood to swamp the prior which takes into account a single link probability only. A simple solution for overcoming this problem would be to introduce an additional parameter a to exponentiate the prior:

$$P(c_i = j|\text{fol}(i)) = P(\text{fol}(i)|c_i = j)P(c_i = j)^a \tag{6.30}$$

This technique has been previously also used for example by Titov and Klementiev (2012) and Elsner et al. (2012). However, sometimes in our model this simple method is not sufficient.

For instance, consider a situation with two clusters *NOUN* and *VERB*. Suppose the *VERB* cluster consists of verbs with the *-ing* suffix. Suppose also that the *NOUN* cluster contains mainly nouns but it also contains some (probably ambiguous) words ending with *-ing* suffix and we would like to relocate those words to the *VERB* cluster. Imagine now resampling a link from one of those *-ing*-suffixing words in the *NOUN* cluster. Assume that the ddCRP prior is trained to have a high probability of linking words with the *-ing* suffix and lower probability of linking words with just *-ng* or *-g* suffix. Then the model prefers creating a link to a word with the same *-ing* suffix. However, the probability of following a word either from the *VERB* cluster or the *NOUN* cluster relies solely on the likelihood component because the prior part is the same for both clusters. Now, if this word is a leaf in the *NOUN* cluster follower structure, then it might be possible to make the switch to the *VERB* cluster because the likelihood of this word may prefer both clusters roughly equally. However, the more followers the word has, the more improbable the cluster switch becomes because, in case of many followers, part of them are nouns that strongly prefer staying in the *NOUN* cluster. Exponentiating the prior with an additional parameter in this situation does not help because the proportions of the posteriors of the words in each cluster will remain the same due to the prior being the same.

We tackle this problem by splitting the sampling process into two separate steps. The first step samples the cluster assignment with the probability proportional to the sum of the prior probabilities of the words in that cluster:

$$P(z_i = k|\mathbf{z}_{-i}, \mathbf{w}) = \frac{\sum_{j':z_{j'}=k} \exp\left(\mathbf{w}^T \mathbf{g}(i, j')\right)}{\sum_{j=1}^{n} \exp\left(\mathbf{w}^T \mathbf{g}(i, j)\right)} \tag{6.31}$$

The second step chooses a particular point to follow in the selected cluster:

$$P(c_i = j|z_j = k, \mathbf{z}_{-i}, \mathbf{w}) = \frac{\exp\left(\mathbf{w}^T \mathbf{g}(i,j)\right)}{\sum_{j':z_{j'}=k} \exp\left(\mathbf{w}^T \mathbf{g}(i,j')\right)} \tag{6.32}$$

Multiplying those quantities together will cause the cluster-specific sums to cancel, and we are left with the original prior probability:

$$\begin{aligned} P(c_i = j|\mathbf{z}_{-i}, \mathbf{w}) &= \frac{\cancel{\sum_{j':z_{j'}=k} \exp\left(\mathbf{w}^T \mathbf{g}(i,j')\right)}}{\sum_{j=1}^{n} \exp\left(\mathbf{w}^T \mathbf{g}(i,j)\right)} \frac{\exp\left(\mathbf{w}^T \mathbf{g}(i,j)\right)}{\cancel{\sum_{j':z_{j'}=k} \exp\left(\mathbf{w}^T \mathbf{g}(i,j')\right)}} \\ &= \frac{\exp\left(\mathbf{w}^T \mathbf{g}(i,j)\right)}{\sum_{j=1}^{n} \exp\left(\mathbf{w}^T \mathbf{g}(i,j)\right)} \end{aligned} \tag{6.33}$$

We can exponentiate both parts separately, and it is still equivalent to exponentiating just the initial prior probability, thus in doing so we do not change anything in the posterior. However, performing the sampling in two steps and exponentiating the two parts of the prior separately does affect the mixing properties of the sampler. So we first sample the cluster assignment according to:

$$P(z_i = k|\text{fol}(i), \mathbf{z}_{-i}, \mathbf{w}) \propto P(\text{fol}(i)|z_i = k) P(z_i = k|\mathbf{z}_{-i}, \mathbf{w})^a \tag{6.34}$$

and then the specific point in the chosen cluster according to $P(c_i = j|z_j = k, \mathbf{z}_{-i}, \mathbf{w})^a$.

To understand why this two-step sampling procedure affects the results consider again the situation with the *VERB* and *NOUN* clusters explained above. The prior in the first sampling step now takes into account *all* words in the respective cluster. Although the *NOUN* cluster may be larger, and thus the summation in (6.31) contains more elements, the suffixes of most words are different from *-ing*, and thus the summed probabilities are smaller. On the other hand, the *VERB* cluster may be smaller but if it contains mostly or only words with suffix *-ing* then we are summing over high probability links. As a result, the priors for different clusters are different, and so the exponentiation will have an effect that can change the odds for one or another cluster. After the cluster assignment has been fixed, we are relatively safe to sample the link to a word with a similar suffix inside the chosen cluster.

6.4.4 Initialization

Initializing the Gibbs sampler becomes a separate question because before the first Gibbs iteration we already need to have a weight vector, and it

would better not be arbitrary. The high-probability areas of the posterior distribution depend on the weight vector values. Although the EM approach will guarantee that the posterior probability will improve with every iteration, starting with an arbitrary weight vector can make the convergence unreasonably slow. Therefore, we begin the inference with optimizing the weight vector on a heuristically initialized follower structure.

The heuristic initialization works as follows. For each word, we select the set of words with whom it shares the longest suffix (but no longer than three characters) and then choose uniformly at random one of them to follow. For instance, a word ending with the suffix *-ing* will choose randomly another word with the same suffix to follow. Even though the longest suffix the words *learning* and *running* share is *-ning*, the set of candidate words is still formed using the 3-character suffix *-ing* only. We set this constraint because the longer suffixes are less probably valid suffixes and they also unnecessarily constrain the set of potential candidates to follow. When a word does not share a 3-character common suffix with any other word, then we consider the set of words sharing 2-character common suffixes. If this fails as well, then we look at the last character only. If a word does not share even a single-character suffix with any other word, then we pick a word to follow uniformly at random from the set of all words.

Initializing in such a way leads to about 1000 clusters, which in most languages is more than the number of morphosyntactic clusters we expect to find. The computational performance of the sampler is directly related to the number of clusters in the state space, and so we do not want to start with too many clusters. Fortunately, the sampler reduces the number of clusters quickly within the first iterations and so only the first few sampling iterations take longer than the rest.

6.5 Experiments

In this section we describe the experimental setup including the datasets used for conducting experiments and the methods used for evaluation.

6.5.1 Data

For the experiments we used the pre-trained word embeddings from the Polyglot project (Al-Rfou et al., 2013)[4], which provides embeddings trained on Wikipedia texts for the 100,000 most frequent words in many languages. The dimensionality of the data vectors is 64.

We developed the models on the English part of the Multext-East (MTE) corpus (Erjavec, 2004) and also evaluated on the other languages. MTE provides both coarse-grained and fine-grained POS labels for the text of

[4] `https://sites.google.com/site/rmyeid/projects/polyglot`

Table 6.1: Statistics for the Polyglot word embeddings and MTE: number of Wikipedia tokens used to train the embeddings, number of tokens/types in MTE, token and type numbers shared by the both datasets, and the number of fine-grained morphosyntactic tags.

Language	**Wiki**	**MTE**		**MTE & Wiki**		**Fine**
	tokens	**tokens**	**types**	**tokens**	**types**	**tags**
English	1843M	118K	9193	115K	7540	104
Bulgarian	45M	101K	15103	91K	9543	104
Czech	91M	100K	17606	91K	11130	573
Estonian	23M	95K	16821	84K	9349	316
Farsi	50M	108K	11322	85K	4197	207
Hungarian	104M	98K	19192	89K	11976	429
Polish	224M	97K	19542	91K	14752	588
Serbian	51M	109K	16818	87K	5020	456
Slovak	41M	103K	18794	92K	11026	581
Slovene	39M	112K	16413	104K	10899	610

Orwell's "1984". Coarse labels consist of 11 main word classes while the fine-grained tags (104 for English) are sequences of detailed morphological attributes. Some of these attributes are not well-attested in English (e.g. gender), and some are distinguishable syntactically but have little or no morphological marking (e.g. 1st and 2nd person verbs). Many tags are assigned only to one or a few words.

Since Wikipedia and MTE are from different domains their lexicons do not entirely overlap; we take the intersection of the two data sets for training and evaluation. The statistical information about the data is given in Table 6.1.

6.5.2 Evaluation

With a few exceptions (Biemann, 2006; Van Gael et al., 2009; Sirts and Alumäe, 2012), POS induction systems typically require the user to specify the number of desired clusters. Usually, this number is set equal to the actual number of tags in the reference corpus. Previously published experiments on MTE corpora (Christodoulopoulos et al., 2010, 2011) have been evaluated against the coarse-grained tagset, which contains 11-12 syntactic tags depending on the language. However, the models presented in this chapter use morphological information to encourage the emergence of a variable number of the more fine-grained morphosyntactic clusters. We expect the models to learn a number of clusters corresponding more closely to the morphosyntactic annotations of the MTE corpora, and therefore we focus on evaluating against the fine-grained tagsets.

For better comparison with the previous work, we also evaluate against the coarse-grained tags. However, these numbers are not strictly comparable to other scores reported on MTE because we are only using a subset of MTE words that also have pretrained Polyglot embeddings. To provide some measure of the difficulty of the task, we report baseline scores using K-means clustering, which is a relatively strong baseline in this task (Christodoulopoulos et al., 2011).

We report four different scores: 1-1 and M-1 accuracy, V-measure and normalized VI (see section 2.4.7). In unsupervised POS induction, it is standard to report accuracy on tokens even when the model itself works on types. Here we also report type-based measures because these can reveal differences in model behavior even when the token-based measures are similar (Reichart et al., 2010).

6.5.3 Experimental setup

For baselines, we use K-means and the IGMM, which both only learn from the word embeddings. K-means assumes fixed covariances and a uniform prior over the clusters. IGMM adds the possibility to model the covariance of each cluster separately, and the CRP prior induces power-law-like distributions. The CRP prior in the IGMM has one hyperparameter—concentration parameter α—that controls the number of clusters induced. We report the results for $\alpha = 5$ and 20.[5]

Both the IGMM and ddCRP models have four hyperparameters controlling the Normal-inverse-Wishart prior for Gaussians: Λ_0, $\mathbf{m}_0$, ν_0 and κ_0. We set the prior scale matrix Λ_0 by using the average covariance obtained from a K-means run with $K = 200$. When setting the average covariance as the expected value of the inverse-Wishart distribution the suitable scale matrix can be computed as:

$$\Lambda_0 = E\left[X\right](\nu_0 - d - 1), \tag{6.35}$$

where ν_0 is the prior degrees of freedom that we set to $d + 10$ and d is the data dimensionality. The rationale behind this procedure is to encourage learning small covariances. K-means assumes a uniform distribution over clusters and the learned clusters tend to be of equal sizes. Thus, setting $K = 200$ and taking the average of the covariances will give a relatively small expected prior covariance, which is smaller than the posterior covariance of the largest induced clusters. This ensures that the posterior covariance is large only when the data clearly requires so and avoids learning overly large clusters that may result from a greedy iterative process where the

[5] Although hyperparameter inference is possible with slice sampling (Johnson and Goldwater, 2009) or Gamma posterior (Escobar and West, 1995), we do not do it here.

large covariance causes more points to be included in that cluster and the resulting even bigger cluster in turn enlarges the covariance even more. We set the prior mean $\mathbf{m}_0$ equal to the sample mean of the data, and the pseudo counts k_0 to 0.01. The concentration parameter for the ddCRP prior is set to 1 in all experiments because preliminary experiments showed the results to be insensitive to this value. For *ddCRP exp*, we report results with the exponent a set to 5. We tuned the hyperparameters on the English corpus and applied them without any changes to other languages.

Each experiment is initialized randomly using the procedure described in section 6.4.4. The sampler is run for 500 iterations, after which we collect a single sample to evaluate.

6.6 Results

We present the results in two sets. First we give the results for English that was used to develop the model. Then we show the results for other MTE languages. Those results and their analysis have not been published previously.

6.6.1 English

We first present the results on English, which was used to develop the models. Tables 6.2 and 6.3 present the results using fine-grained type- and token-based evaluation respectively. Each number is an average of 5 experiments with different random initializations. For each evaluation setting, we provide two sets of scores. First we give the scores of the given model; second we present the comparable scores of the K-means runs with the same number of clusters as induced on average by the non-parametric model.

These results show that all non-parametric models perform better than K-means according to most evaluation measures and schemes. The poor performance of the K-means can be explained by its tendency to find clusters of relatively equal size. This behavior is characteristic of the non-Bayesian algorithms using EM (Johnson, 2007), although the POS clusters are rarely of similar size. The common noun singular class is by far the largest in English, containing roughly a quarter of the word types. Non-parametric models can produce clusters of different sizes when the evidence indicates so, and this is clearly the case here. An exception is the M-1 measure evaluated on word types using fine-grained tags, where K-means always performs better. This behavior can again be attributed to the property of the K-means algorithm to induce relatively small clusters of equal size, as K is quite large. It is easier for the M-1 mapper to produce a more accurate mapping by combining small clusters than by using the clusters of variable size induced by the non-parametric models.

Table 6.2: Results of baseline and ddCRP models evaluated on English **word types** using **fine-grained** tags. For each model we present the number of induced clusters K (or fixed K for K-means) and different scores. The second column under each evaluation setting gives the scores of the K-means with K set equal to the number of clusters induced by the model in that row. For definitions of the measures, see section 2.4.7.

Model	**K**	**1-1**	**V-M**	**M-1**	**NVI**
K-means	104	16.1	47.3	75.6	1.53
IGMM, $\alpha = 5$	55.6	41.0/23.1	45.9/49.5	59.2/74.2	1.18/1.34
IGMM, $\alpha = 20$	121.2	35.0/14.7	47.1/46.9	65.7/**76.2**	1.34/1.58
ddCRP uniform	80.4	50.5/18.6	52.9/48.2	70.4/75.4	1.10/1.45
ddCRP learned	89.6	50.1/17.6	55.1/48.0	73.4/75.5	1.08/1.47
ddCRP exp	47.2	**64.0**/25.0	**60.3**/50.3	69.1/74.6	**0.73**/1.28

Table 6.3: Results of baseline and ddCRP models evaluated on English **word tokens** using **fine-grained** tags. For each model we present the number of induced clusters K (or fixed K for K-means) and different scores. The second column under each evaluation setting gives the scores for K-means with K equal to the number of clusters induced by the model in that row. For definitions of the measures, see section 2.4.7.

Model	**K**	**1-1**	**V-M**	**M-1**	**NVI**
K-means	104	39.2	62.0	59.1	0.76
IGMM, $\alpha = 5$	55.6	48.0/37.2	64.8/61.0	58.0/53.2	0.66/0.73
IGMM, $\alpha = 20$	121.2	50.6/44.7	67.8/65.5	65.4/64.8	0.66/0.71
ddCRP uniform	80.4	52.4/35.1	68.7/60.3	64.2/53.2	0.62/0.77
ddCRP learned	89.6	51.1/39.0	**69.7**/63.2	**65.6**/58.9	0.61/0.73
ddCRP exp	47.2	**55.1**/33.0	66.4/59.1	59.5/48.0	**0.58**/0.75

From the token-based evaluation it is hard to say which IGMM hyperparameter value is better even though the number of induced clusters differs by a factor of 2. The type-base evaluation, however, prefers the smaller value with fewer clusters. We can see similar effects when comparing IGMM and ddCRP uniform. We expected these two models perform on the same level, and their token-based scores are similar, but on the type-based evaluation the ddCRP is clearly superior. The difference could be due to the non-sequentiality, or because the samplers are different—IGMM resampling one item at a time while ddCRP performs blocked sampling.

Further, we can see that ddCRP uniform and learned perform roughly the same. Although the prior in those models is different, they work mainly using the likelihood. The ddCRP with learned prior does produce nice

follower structures within each cluster, but the prior is in general too weak to influence the clustering decisions compared to the likelihood. Exponentiating the prior reduces the number of induced clusters and improves the results, as it can change the cluster assignment for some words where the likelihood strongly prefers one cluster, but the prior clearly indicates another.

Although we also report M-1 measures with each evaluation setting, we do not consider this score as informative as, for example, 1-1 or V-m. M-1 accuracy is sensitive to the number of clusters—in general, the more clusters are induced, the higher the M-1 accuracy. This behavior would speak against using M-1 measure when inducing variable number of clusters as it might create the temptation to bias the model to induce more clusters for the sake of higher accuracy.

Although NVI seems to correlate with the other measures most of the times, it is most difficult to interpret. Smaller numbers indicate better clusterings and a score > 1 refers to a clustering worse than the single cluster solution. The only system having NVI score less than one on word types using the fine-grained tagset is ddCRP exp and in that sense, NVI score agrees with all other measures. However, all other models have an NVI score greater than 1, meaning that according to this measure the single cluster solution would be better than the clusterings induced by these systems. On the other hand, the token-based NVI scores are smaller than 1 for all systems while the clusterings are the same. This exemplifies that type- and token-based evaluation reveal different aspects of the same clustering (Reichart and Rappoport, 2009). Also, perhaps NVI is not as suitable for type-based evaluation, as the other measures do not agree that the clusterings having NVI > 1 would be worse than the single cluster solution.[6]

6.6.2 Other languages

Based on the English results we chose the best ddCRP model setting, fixed the IGMM hyperparameter and ran the experiments on all MTE languages.

Surprisingly, the number of clusters induced by the non-parametric models is in all cases much smaller than the number of fine-grained tags in the reference corpus. In fact, the number of induced clusters is in most instances much closer to the size of the coarse-grained syntactic tagset. Thus, we first provide in Tables 6.4 and 6.5 both type- and token-based 1-1 and V-m scores evaluated on the coarse tagsets. First of all, when looking at the 1-1 type-based accuracies, in most cases the ddCRP model improves over the IGMM and the both non-parametric models improve over the K-means baseline, both when K-means learns the reference number of clusters and when K is set equal to the number of clusters induced by the respective

[6]The single-cluster solution has a type-based accuracy of 29.8%.

non-parametric model. However, when evaluating on tokens, the IGMM is the best in most cases while ddCRP is still better than both K-means baselines. V-m mostly agrees with the 1-1 scores on tokens. Curiously, the type-based V-m scores are in many cases very low on non-parametric models, and the K-means algorithm produces the best results. Although the absolute numbers themselves are not too impressive, there are also some positive highlights. For instance, Slovak ddCRP and Slovene IGMM both score over 60% 1-1 accuracy on types and also have reasonable or at least mediocre V-m and token-based 1-1 scores.

Looking more carefully at the Slovak results reveals that the induced clusterings contain several large pretty clean clusters corresponding to the open class words (nouns, verbs, and adjectives). However, the largest cluster tends to be a mix of all word types. It is mapped to the noun class by 1-1, and its type-based accuracy is 77.9% while its token-based accuracy is only 43.8%. So this cluster is mostly composed of nouns but additionally also accommodates frequently occurring words from all other classes. For example, the verbs in this noun cluster are on average almost three times more frequent than the verbs in the cluster mapped to the verb class. Another problem is that the closed class high-frequency words do not form separate clean clusters themselves but most of them are in the biggest cluster, and the rest are divided into different clusters. Also, there are several clusters corresponding to nouns, verbs, and adjectives, but each of them typically contains words from several morphosyntactic classes. For istance, there is a smaller noun cluster that mostly contains words in the instrumental case, which are expressed via a small set of distinctive suffixes (e.g. *-ou*, *om*, *am*, *ami*). In addition to nouns, it also contains a set of adjectives with similar morphological suffixes, which indicates that the ddCRP model morphological component was effective in this situation. However, the embeddings did not provide enough discriminating information to separate nouns from adjectives.

All in all, these results suggest that the word embeddings at least in some languages contain syntactic rather than morphosyntactic information. Although these numbers are not directly comparable with the previous results, the scores in several languages are in the similar range to the best-published results in Christodoulopoulos et al. (2010) and Sirts and Alumäe (2012).

Next, we look at the results in Table 6.6 evaluating word types on the fine-grained tagset. In general, the 1-1 accuracy scores are pretty low due to the too small number of clusters. The only exception is English, where the difference between the number of reference and induced clusters is the smallest and whose results were discussed in the previous section. However, in most languages we can again see the improvement of the ddCRP model over the IGMM in terms of both 1-1 and V-m, although in many

cases these improvements are minor and in general, the performance of the non-parametric models falls below the baseline K-means with the same number of clusters. The only language besides English that shows some visible improvement over all baselines is Farsi reaching 47.5% type-based 1-1 accuracy with the ddCRP model. However, the number of clusters induced in Farsi is on average only 14.6 while the gold number of clusters is over 200 and so the clustering corresponds more closely to the coarse-grained tagset. Regarding V-m, we can again see the similar effect we observed in Table 6.4 for coarse-grained scores, that the K-means with the reference number of tags has the largest V-m scores, which here is probably caused by the large number of small clusters learned by the K-means.

The token-based scores evaluated against the fine-grained tagsets in Table 6.7 show less variation between different models than the type-based scores. In almost a half of the languages, ddCRP is better than IGMM and in most cases IGMM is better than K-means with K equal to the reference number of clusters according to 1-1 accuracy. Also, the ddCRP 1-1 scores are in most cases slightly better than the K-means scores with the same number of clusters. For IGMM, both 1-1 and V-m scores are in most cases better than the K-means baselines. In terms of V-m, we again see the already familiar pattern that the K-means clustering using the reference number of clusters performs the best.

There are few points to highlight about these results. First, consider that higher type-based accuracies indicate better performance on low-frequency words while the token-based measures assess the performance of the more frequent words. In many cases, the differences between the models using token-based evaluation are small, regardless of the different number of induced clusters, which suggests that all models perform in a similar range in terms of frequent words. However, the message is different when we look at the type-based results where there are much clearer differences between the various model, indicating that different models handle low-frequency words differently.

Secondly, we want to draw attention to the fact that 1-1 and V-m scores do not always agree, and the discrepancies between those two measures are especially visible when performing type-based evaluation. The type-based results on K-means using the reference number of clusters are highest in terms of V-m but lowest when looking at the 1-1 accuracies. Which measure should we take more seriously? Christodoulopoulos et al. (2010) argues that V-m is perhaps the most suitable measure when comparing clusterings with the varying number of clusters. According to that, K-means is better. However, considering that K-means did learn exactly the same number of clusters as there are in the reference corpus and the 1-1 accuracies are the lowest, we cannot accept this judgment so quickly. Imagine we had used V-m

only as suggested by Christodoulopoulos et al. (2010) and never computed the 1-1 accuracies. Then we would have the impression that K-means indeed performed the best. Moreover, if we had only computed the token-based scores, as has been standard for unsupervised POS induction, we would also think that, in general, there are no major differences between the clusterings produced by different systems. For instance, the token-based V-m alone does not reveal any significant differences between English K-means and ddCRP, while the type-based 1-1 accuracy shows that ddCRP is much better. Also in Farsi, the token-based V-m suggests that K-means is much better than ddCRP (62.4 vs. 39.5). However, the ddCRP type-based 1-1 accuracy in Farsi is 47.5% while K-means scores only 10.5%.

Based on the above, we propose that the token-based V-m alone is not sufficient to describe the differences between different clusterings, especially in the context of variable number of clusters. We propose that if the clustering corresponds well to the reference clustering then it should be reflected in various measures, including 1-1 accuracy, and as evaluated both on types and tokens assessing the performance of the low- and high-frequency words respectively. To provide further arguments to our proposition, we evaluated the results of a random baseline. For each word type, we randomly generated a cluster assignment from the K number of clusters with K equal to the reference number of clusters. We then evaluated these results on tokens, which gave V-m scores from 43.3% in Bulgarian up to 64.3% in Serbian. Although the token-based K-means results are in all cases better than the random baseline, the difference for instance in Serbian scores is only less than 3%.

Assessing our results in the light of this proposition, we conclude that the pattern of the non-parametric models being better than K-means and ddCPR being better than IGMM is visible. This is expected because the non-parametric priors enable inferring clusters with different sizes which is also the case in reference clusterings while K-means tends to learn clusters with roughly equal sizes. Also, the morphological similarity function in the ddCRP prior was specially tailored for modeling infrequent yet morphologically regular words for which, due to the low frequency, the distributional cues might not be informative enough. However, in terms of absolute scores, only English results can be considered as positive. As a negative example consider for instance Serbian whose token-based V-m score on the ddCRP model is almost 60%[7] although the type-based 1-1 accuracy is one of the lowest, below 10%.

[7]It may seem that this score is lower than the random baseline mentioned above but this is not the case because the number of clusters is different—random baseline for Serbian was induced with 456 clusters while the ddCRP model mentioned here found 48 clusters. The random baseline score with 48 clusters would be only ca 45%.

Table 6.4: Results of the K-means and IGMM baselines and the best ddCRP model on all MTE languages, **type-based** evaluation against the MTE **coarse-grained** tagset. For each method, we report 1-1 and V-m scores and the average number of induced clusters. The second column under IGMM and ddCRP presents the 1-1 and V-m scores of the K-means run with the K set equal to the number of induced clusters.

Language	**K-means**		**IGMM, $\alpha = 5$**			**ddCRP exp, $\alpha = 1$, a $= 5$**		
	#C	**1-1 / V-m**	**#C**	**1-1 / V-m**	**K-means K=#C**	**#C**	**1-1 / V-m**	**K-means K=#C**
English	11	39.2 / **41.8**	55.6	40.6 / 40.2	14.1 / 35.7	47.2	**50.2** / 40.6	15.1 / 36.4
Polish	12	35.4 / 45.0	51.8	37.5 / 44.8	12.7 / 39.3	65.6	**48.4** / **48.0**	11.7 / 38.3
Hungarian	12	35.0 / **43.6**	46.2	44.1 / 41.0	14.3 / 37.0	50.8	**53.8** / 30.3	13.7 / 36.6
Czech	12	40.7 / 53.6	43.0	52.4 / **56.2**	16.7 / 45.4	57.6	**53.6** / 47.7	13.2 / 43.5
Serbian	12	34.8 / **39.9**	32.2	33.1 / 14.3	20.9 / 37.1	48.0	**45.4** / 20.2	16.3 / 36.2
Farsi	12	27.6 / 28.2	33.8	40.6 / 18.6	16.0 / 27.6	14.6	**48.7** / 11.6	24.2 / **28.7**
Bulgarian	11	39.8 / **47.6**	37.2	**51.4** / 41.8	16.9 / 40.5	12.0	37.5 / 10.7	36.6 / 46.1
Slovak	12	39.1 / **48.5**	44.0	40.5 / 46.5	15.6 / 42.3	60.8	**60.2** / 46.4	12.5 / 40.8
Slovene	12	38.4 / 46.9	30.4	**61.9** / **49.6**	19.2 / 41.2	20.6	36.2 / 16.2	23.5 / 42.6
Estonian	11	32.3 / 33.0	39.2	33.9 / 16.9	15.6 / 33.1	37.2	**52.6** / 16.0	16.5 / **33.3**
Average	11.7	36.2 / **42.8**	41.3	43.6 / 37.0	16.2 / 37.9	41.4	**48.7** / 28.8	18.3 / 38.2

Table 6.5: Results of the K-means and IGMM baselines and the best ddCRP model on all MTE languages, **token-based** evaluation against the MTE **coarse-grained** tagset. For each method, we report 1-1 and V-m scores and the average number of induced clusters. The second column under IGMM and ddCRP presents the 1-1 and V-m scores of the K-means run with the K set equal to the number of induced clusters.

Language	**K-means**		**IGMM, $\alpha = 5$**			**ddCRP exp, $\alpha = 1$, a $= 5$**		
	#C	**1-1 / V-m**	**#C**	**1-1 / V-m**	**K-means K=#C**	**#C**	**1-1 / V-m**	**K-means K=#C**
English	11	44.4 / 45.5	55.6	**48.3** / **58.3**	40.8 / 55.0	47.2	47.8 / 55.1	36.9 / 53.1
Polish	12	35.3 / 43.3	51.8	44.9 / **60.1**	30.7 / 49.9	65.6	**48.6** / 58.7	33.8 / 52.1
Hungarian	12	39.6 / 50.1	46.2	**45.7** / **58.4**	39.2 / 52.4	50.8	44.5 / 53.9	37.3 / 52.1
Czech	12	37.2 / 46.9	43.0	**52.7** / **62.6**	35.7 / 51.6	57.6	44.3 / 50.0	33.1 / 49.9
Serbian	12	30.6 / 39.2	32.2	43.3 / 53.2	34.1 / 48.1	48.0	**44.1** / 53.4	42.7 / **55.0**
Farsi	12	37.8 / 43.7	33.8	**50.0** / **50.9**	36.6 / 47.6	14.6	38.6 / 37.6	36.7 / 44.2
Bulgarian	11	38.5 / 45.4	37.2	**52.9** / **62.1**	37.6 / 53.2	12.0	42.7 / 43.8	38.7 / 46.9
Slovak	12	30.5 / 39.1	44.0	**45.3** / **56.5**	31.9 / 48.6	60.8	44.8 / 51.9	31.2 / 48.7
Slovene	12	34.7 / 44.5	30.4	**50.0** / **60.2**	32.3 / 44.5	20.6	33.2 / 31.0	34.0 / 39.7
Estonian	11	34.0 / 32.9	39.2	35.7 / 48.1	31.4 / 41.9	37.2	**42.9** / **48.4**	35.0 / 44.9
Average	11.7	36.3 / 43.1	41.3	**46.9** / **57.0**	35.0 / 49.3	41.4	43.2 / 48.4	35.9 / 48.7

Table 6.6: Results of the K-means and IGMM baselines and the best ddCRP model on all MTE languages, **type-based** evaluation against the MTE **fine-grained** tagset. For each method, we report 1-1 and V-m scores and the average number of induced clusters. The second column under IGMM and ddCRP presents the 1-1 and V-m scores of the K-means run with the K set equal to the number of induced clusters.

Language	**K-means**		**IGMM, $\alpha = 5$**			**ddCRP exp, $\alpha = 1$, a $= 5$**		
	#C	**1-1 / V-m**	**#C**	**1-1 / V-m**	**K-means K=#C**	**#C**	**1-1 / V-m**	**K-means K=#C**
English	104	16.1 / 47.3	55.6	41.0 / 45.9	23.1 / 49.5	47.2	**64.0** / **60.3**	25.0 / 50.3
Polish	588	14.5 / **58.8**	51.8	16.0 / 43.0	25.9 / 56.1	65.6	22.0 / 58.2	**26.0** / 56.8
Hungarian	429	11.5 / **52.5**	46.2	25.8 / 37.4	24.2 / 49.7	50.8	**26.1** / 43.3	23.1 / 50.0
Czech	573	14.5 / 60.3	43.0	19.2 / 44.7	**33.5** / 60.0	57.6	24.7 / 55.4	32.8 / **60.5**
Serbian	456	15.0 / **57.1**	32.2	7.0 / 15.4	19.3 / 47.7	48.0	8.8 / 21.4	**20.0** / 49.7
Farsi	207	10.5 / **41.2**	33.8	36.3 / 25.2	19.5 / 38.5	14.6	**47.5** / 24.8	27.6 / 35.8
Bulgarian	104	24.5 / 59.0	37.2	25.7 / 41.5	37.8 / **60.8**	12.0	20.4 / 40.2	**44.1** / 58.5
Slovak	581	13.7 / **58.7**	44.0	11.5 / 33.5	**27.4** / 57.6	60.8	16.5 / 48.5	26.9 / 58.3
Slovene	610	16.1 / 61.5	30.4	14.3 / 41.3	**35.4** / **63.7**	20.6	14.5 / 44.6	33.4 / 62.6
Estonian	316	12.2 / **48.2**	39.2	12.4 / 16.7	23.4 / 42.7	37.2	14.6 / 23.7	**23.8** / 42.8
Average	396.8	14.8 / **54.5**	41.3	20.9 / 34.5	27.0 / 52.6	41.4	25.9 / 42.0	**28.3** / 52.5

Table 6.7: Results of the K-means and IGMM baselines and the best ddCRP model on all MTE languages, **token-based** evaluation against the MTE **fine-grained** tagset. For each method, we report 1-1 and V-m scores and the average number of induced clusters. The second column under IGMM and ddCRP presents the 1-1 and V-m scores of the K-means run with the K set equal to the number of induced clusters.

Language	**K-means**		**IGMM, $\alpha = 5$**			**ddCRP exp, $\alpha = 1$, a $= 5$**		
	#C	**1-1 / V-m**	**#C**	**1-1 / V-m**	**K-means K=#C**	**#C**	**1-1 / V-m**	**K-means K=#C**
English	104	39.8 / 62.0	55.6	48.0 / 64.8	37.2 / 61.0	47.2	**55.1** / **66.4**	33.0 / 59.1
Polish	588	33.3 / **66.2**	51.8	**33.7** / 60.2	32.1 / 57.8	65.6	32.2 / 62.5	32.7 / 60.1
Hungarian	429	34.9 / **62.3**	46.2	**43.7** / 59.1	42.5 / 59.1	50.8	42.3 / 58.8	40.8 / 59.0
Czech	573	34.0 / **68.4**	43.0	**36.8** / 62.0	33.7 / 59.8	57.6	33.1 / 57.2	34.0 / 60.3
Serbian	456	**37.0** / **67.2**	32.2	36.9 / 54.1	31.9 / 53.3	48.0	**37.0** / 58.2	36.2 / 59.2
Farsi	207	39.2 / **62.4**	33.8	**40.0** / 52.4	28.5 / 50.8	14.6	33.5 / 39.5	29.7 / 45.5
Bulgarian	104	39.8 / **64.3**	37.2	**43.5** / 63.7	41.1 / 61.3	12.0	30.9 / 51.3	37.4 / 51.8
Slovak	581	**33.6** / **66.3**	44.0	30.3 / 55.0	29.8 / 56.9	60.8	32.0 / 57.5	30.5 / 58.6
Slovene	610	**39.1** / **69.9**	30.4	34.2 / 59.0	35.4 / 57.8	20.6	24.7 / 43.7	33.4 / 52.5
Estonian	316	28.0 / **60.7**	39.2	31.3 / 49.9	30.9 / 49.8	37.2	**38.3** / 52.7	34.4 / 50.9
Average	396.8	35.9 / **65.0**	41.3	**37.8** / 58.0	34.3 / 56.8	41.4	35.9 / 54.8	34.2 / 55.7

6.7 Discussion

Experimental results using the ddCRP model showed reliably good performance only in English, which is morphologically the simplest language in the MTE corpora, and that was also the language the model was developed on. In other languages, the experiments mostly produced negative results, which suggests that the model was overfitted to English. The model was complex enough to capture the morphological regularities of English language but too simple to learn the regularities of the other, morphologically more complex languages. Next we will review some possible reasons of this failure that can suggest directions for further research.

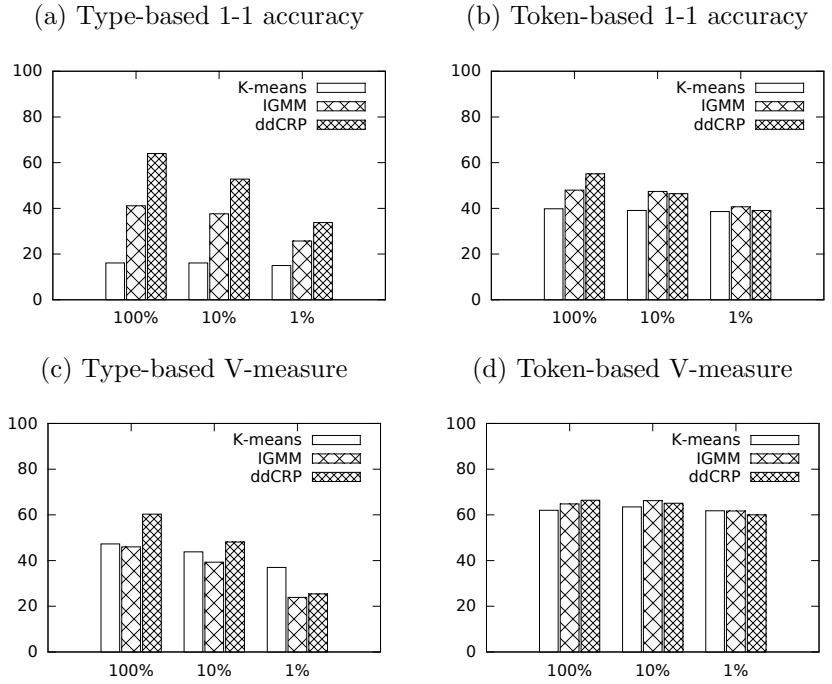

Figure 6.5: Comparative results of different models using embeddings trained on corpora of different sizes: 100% denotes the full corpus, 10% and 1% denote the respective smaller fractions.

The first aspect is the quality of the word embeddings. The embeddings used in this work were trained on Wikipedia, and the Wikipedia in English is substantially larger than in any other language. The next largest of the MTE languages is Czech, being ten times smaller than English, and the smallest corpus in Estonian is over 100 times smaller. To verify how much the training corpus size affects the quality of the embeddings, we composed two subsets of the English Wikipedia corpus containing 10% and 1% of the

original corpus respectively. We used the code from the Polyglot project page[8] to train the embeddings and then trained all the models on those corpora. Figure 6.5 provides the comparison between different models using the embeddings trained on the corpora of different sizes. The evaluation is performed against the fine-grained tagset, and for each experiment we report both type- and token-based 1-1 accuracy and V-m.

First, notice that the ddCRP scores drop the most when the training corpus size decreases. IGMM also drops quite a lot but not as much as ddCRP while K-means scores almost do not change. Second, type-based scores fall quite a lot more than the token-based scores, which indicates that the performance of the low-frequency words is more affected, and 1-1 accuracy decreases more than V-m. In fact, token-based V-m is the most indifferent to the size of the training corpus as well as to the clustering model. It is not clear why the morphological prior in the ddCRP was not able to compensate for the decreasing quality of the embeddings of the low-frequency words. One reason might lie in the choice of the hyperparameters that might have to be set differently for a different set of embeddings. In general, these results provide evidence that the embeddings quality is dependent on the amount of training data which again influences the clustering results. We conclude that for languages other than English, the amount of data used to train the embeddings may not have been sufficient for encoding the desired morphosyntactic regularities. Also, for morphologically rich languages, the training corpus size might have to be even larger for learning good quality embeddings because, in those languages, different inflected word forms are less frequent generally and more data might be needed to observe them in as many contexts as words with the same frequency in English. In addition, these results also support our conclusion from section 6.6.2 that the token-based V-m alone is not sufficient for evaluating the performance of a POS induction model.

The embeddings were trained using a context window of 5 words, which might be appropriate for capturing the local syntactic properties of English that has a fixed word order. However, some other languages, Estonian, for example, exhibit relatively free word order. Thus, syntactic patterns may have long range behavior, in which case a larger window size might be necessary to learn the relevant information. However, larger window sizes have been associated with more topical-like embeddings (Levy and Goldberg, 2014), which is not what is desired for syntactic clustering. This points to a necessary research direction of studying the effect of the context window size for languages other than English, that might be morphologically complex,

[8] `https://sites.google.com/site/rmyeid/projects/polyglot`

have free word order and long range syntactic dependencies, in order to understand how to learn proper embeddings for different tasks.

The second major issue is the simplicity of the similarity function used in the ddCRP model that only considers the shared suffix features with the maximum length of three characters between each pair of words. There are two problems related to this similarity function. The first is related to the optimization goal of the log-linear model. Currently, for learning the weights for the log-linear model, only the links in the current follower structure are considered as positive examples whereas all the other word pairs are viewed as negative training examples. This view is most probably too constraining because essentially according to the reference clustering, any link within the cluster should be permitted and viewed as a positive example. Correcting this issue would require summing over all possible follower structures within each cluster and treating all those follower structures that cross the cluster borders as negative examples. However, considering that clusters can contain several thousands of words, this exhaustive enumeration is clearly intractable. A possible compromise would be to treat all links pointing to the same cluster as positive instead of just the one that is represented in the follower structure.

Another problem related to the similarity function is the simplistic assumption that words in each morphosyntactic cluster are mostly realized with a single suffix or in case of allomorphy, the distributions over suffixes within clusters are very peaky so that there is a single most frequent suffix while others are relatively infrequent. This assumption holds for English, but it is probably not true for the morphologically more complex languages where a single morphosyntactic role can be realized by several suffixes of roughly equal frequencies. Our similarity function is currently unable to learn these kinds of regularities, and this is probably also one of the reasons, why the model did not perform so well on other languages.

The advantage of the ddCRP prior is that it enables plugging in any distance or similarity function. The distance does not have to be a metric as the links are directed, and one can easily imagine having non-symmetric distances (or similarities). Also, any prior linguistic knowledge about a language or a language group could be incorporated into the similarity function. Defining more complex similarity functions could potentially overcome the problem with allomorphy and enable learning effective priors for morphologically complex languages. For instance, one option for modeling allomorphemes would be to encode similarities with finite state transducers that would allow several suffixes to realize the same morphosyntactic cluster. Regardless of the choice, no similarity function can fully express the whole set of morphological relations unambiguously and, therefore, the word embeddings must contain enough syntactic information to support the linking decisions.

Lastly, different languages may require different parameterizations of the model. In this work, the hyperparameters of the prior for the Gaussian and the ddCRP prior were tuned and developed on English language, and the hope was that the different languages are insensitive to these decisions. However, more work is necessary to verify or disprove this assumption.

6.8 Conclusion

The work in this chapter demonstrates that morphological and distributional features can be combined in a flexible, joint probabilistic model, using the distance-dependent Chinese restaurant process. A key advantage of this framework is the ability to include arbitrary features in the prior distribution. We were able to demonstrate the usefulness of this approach on English where the ddCRP model using both types of features clearly outperformed the baseline models using distributional information only—a result that supports the Claim B.

The number of clusters induced by the best ddCRP model for English was much smaller than the number of different annotated tags in the reference corpus. However, the type-based 1-1 accuracy measure of the baseline K-means clustering is the highest using the induced number of clustering. Although the K-means type-based 1-1 accuracies are pretty low in all settings tried, the results with the reference number of tags are almost twice as low than when using the number of clusters induced by the ddCRP model. These results confirm the utility of letting the model choose the number of morphosyntactic clusters dynamically and thus provide support for the Claim A.

In the other, morphologically more complex languages, we were able to demonstrate the superiority of the ddCRP model over the other models. However, the absolute scores in general were quite low, and we reviewed the possible reasons for that. One important problem was the insufficient quality of the word embeddings that were trained on too small corpora and thus did not contain enough syntactic information. The second problem involved the too simplistic nature of the similarity function used in the ddCRP prior. The similarity function was sufficient to capture the morphological regularities of the relatively simple English language but was not enough for learning the morphosyntactic patterns of the morphologically more complex languages. Thus, a potential avenue for future work would be to develop more sophisticated similarity functions that would be able to capture the morphological regularities of other languages than English. For instance, possible options include using features that incorporate prior knowledge of the language's morphological structure or modeling more explicitly the allomorphic processes.

Chapter 7

Conclusions and future work

In this dissertation, we have presented three computational models to learn different morphology related tasks. All models were defined in the non-parametric Bayesian framework and used unsupervised or weakly-supervised learning techniques. Also, they all are joint learning models, inferring several sets of random variables simultaneously.

The first model in Chapter 4 performed joint POS induction and morphological segmentation. The POS induction part was demonstrated to perform well in comparison with the other state-of-the-art unsupervised POS induction systems. However, the segmentation results were rather poor indicating that the used unigram segmentation model was too simple. Nevertheless, we were able to demonstrate that the joint learning produced better results than the non-joint learning in a setting where the labels of one task (POS tags and segmentations in turn) were fixed according to the reference corpus.

The next models in Chapter 5 focused solely on the morphological segmentation task. We showed that learning morphemes jointly with latent morpheme sub- and/or superstructures leads to better results than learning flat morpheme sequences only. This effect was demonstrated both in unsupervised and weakly-supervised learning settings. The contributions of this chapter include two state-of-the-art weakly-supervised morphological segmentation systems, one of which can also be used in unsupervised or fully supervised setting. Both systems are scalable to process millions of word types.

The third model in Chapter 6 was again about the joint learning of words' syntactic functions and morphological features. The target goal of this model was to perform unsupervised morphosyntactic clustering while the learned suffix similarity model played the supportive role. We demonstrated the superiority of the model using both distributional and morphological

features over the model using only distributional information. Our contributions in this chapter include setting the current benchmark in fine-grained unsupervised morphosyntactic clustering for languages in the Multext-East corpus, showing how to use a model-based similarity function in the distance-dependent Chinese restaurant process model and demonstrating that the word embeddings trained by a neural network can be used as Gaussian random variables in an infinite Gaussian mixture model.

In the rest of this chapter, we consider two additional topics that arise from the results presented in previous chapters and provide directions for future research. First, we will look at the idea of how those three models together form an unsupervised morphology induction system. Secondly, we will discuss the assumptions usually made in unsupervised models about the relationships between morphological suffixes and POS tags. Finally, we will turn to the Claims stated in the introduction and assess their validity in the light of the results presented in this dissertation.

7.1 Unsupervised morphology induction system

All models presented in this dissertation address slightly different aspects of morphological processes. However, looking at them together in a bundle we can see the outlines of a full unsupervised morphology induction system, with some logical juncture points for connecting the models.

First of all, the Adaptor Grammar segmentation models from Chapter 5 could be used as a segmentation component in the joint model of Chapter 4. One could define a segmentation grammar augmented with POS tags, analogous to the grammars used to train topic models with AGs (Johnson, 2010), which could be updated as tag assignments change. Next, unsupervised POS induction could be tackled in two steps, by first grouping words into coarse-grained syntactic clusters and within each such cluster performing another clustering according to the more fine-grained morphosyntactic function. Thus, we could imagine learning the syntactic tags together with their distributions over morphemic suffixes with a joint POS and segmentation model combining the models from Chapters 4 and 5. Then, inside each syntactic cluster learn the fine-grained morphosyntactic clusters with the model presented in Chapter 6, where the morphological similarity function could use the suffixes inferred by the segmentation model. The resulting model would tag each word with a morphological analysis consisting of the morphological segmentation, syntactic and morphosyntactic cluster labels.

Put together, the presented models cover almost all relevant aspects of morphology from the unsupervised computational point of view. The only essential part missing from the full picture is the paradigmatic view. However, considering the existing components, it would not be too hard to imagine

a model that, based on the segmentations and syntactic and morphosyntactic cluster assignments, attempts to organize words into paradigm-like structures. Those structures would provide another clustering, orthogonal to the syntactic ones, organizing together word forms belonging to the same lexeme.

7.2 Assumptions about POS and morphology

Previous work in unsupervised POS induction has claimed and shown the utility of using morphological information (Clark, 2003; Lee et al., 2010; Christodoulopoulos et al., 2011; Blunsom and Cohn, 2011), mostly in the form of suffixes. Although the improvements using morphological information have been relatively small in most cases (see, for example, Table 3 in Christodoulopoulos et al. (2011)), we have in our work relied on those previous results. The implicit assumptions made in these works is that there are systematic dependencies between POS tags and suffixes whereby the conditional distributions in either direction—$P(\text{tag}|\text{suffix})$ and $P(\text{suffix}|\text{tag})$—are peaky.

Ideally, there is a single suffix realizing each tag and the suffixes realizing different tags are distinct. The tagset of 45 tags used in English WSJ corpus (Marcus et al., 1993) seems pretty close to this ideal situation because it has a separate POS class for every major suffix in English. However, there are also other tagsets for English; for instance, the one used to label the MTE corpus employed in our work. MTE English part has 12 coarse and 104 fine-grained tags, and neither of those sets corresponds to the mostly one suffix per tag ideal. Also in many other languages the situation is fuzzier. In morphologically rich languages, there can be several different inflection classes realized by different sets of suffixes. Also, various POS tags may reuse the same suffixes. Therefore, in those languages it might not be even possible to devise a reasonable tagset that would be close to the ideal.

We propose that the assumptions about the relationships between POS tags and suffixes have been biased by the properties of English language and the WSJ tagset, because most unsupervised POS induction systems, including those using morphological features, have been developed on WSJ corpus. The results presented in Chapter 4 and especially in Chapter 6 support our proposition. The morphosyntactic clustering model in Chapter 6 performed well on English but did much worse on other languages. Of course, there are various reasons for that (as discussed in section 6.8), but one of them might lie in the idealized assumption that the distributions relating POS tags and morphological suffixes are peaky, even close to one-to-one mapping.

Nevertheless, in our work we assumed the peaky distributions between tags and suffixes but our models in most cases learned much coarser tagsets

than we expected. This also indicates that the assumptions encoded in the models and the true regularities found in the language did not match. Based on our results, different methods may be needed to model the relationships between morphological suffixes and POS tags, depending on whether the aim is to learn coarse-grained syntactic or fine-grained morphosyntactic POS classes. In addition, further research is needed to find appropriate ways of modeling the relationship between morphology and POS tags in morphologically rich languages where the distributions over tags conditioned on suffixes and over suffixes conditioned on tags are probably flatter than in English.

7.3 Validation of the Claims

Claim A stated that learning some latent aspects of the natural language structures together with the target structures would improve the results. The morphological segmentation results in Chapter 5 clearly support this Claim. We demonstrated that the morphological grammars generating submorpheme structures inside the morphemes and/or compound structures above the morphemes were superior to the grammars generating flat morpheme sequences only.

In the POS induction models of Chapters 4 and 6, we treated the number of learned POS clusters as a latent random variable. The number of clusters learned by the model in Chapter 4 was close to the size of the coarse-grained reference tagset, which was desirable in the experimental context of Chapter 4. In Chapter 6, the tagsets induced for English were roughly half as large as the number of fine-grained morphosyntactic tags in the reference corpus. However, according to our subjective judgment, the reference tagset is too fine-grained, and the number of clusters learned by our non-parametric model is much closer to the relevant partitioning of the data. Thus, these results, too, provide some preliminary support for Claim A. To more fully support this Claim, further experiments should be conducted comparing a clustering with the reference number of tags and a clustering with a variable number of tags in some downstream task.

Claim B proposed that the unsupervised joint learning of related aspects improves the results over non-joint learning. This Claim turned out to be somewhat difficult to assess because of two reasons. First, if a joint model does not perform as well as expected, then there can be many reasons for that aside from the option that the joint learning is not beneficial. The most likely reason is that the overall model structure fails to capture relevant relationships between modeled tasks sufficiently. This seems to be the case in Chapter 4 where the poor segmentation results suggest that, besides the over-simplicity of the segmentation component itself, the information between tags and segments was not flowing as well as expected. However,

these results do not imply that the segmentation learning in principle could not gain from the joint learning with POS tags. Rather, the current model structure was not able to exploit the potential advantages properly. However, the experiments comparing semi-supervised learning, where either tags or segmentations were fixed to the reference annotations, and joint unsupervised learning revealed that the joint learning performs better, which provides some support for the stated Claim.

The second issue is that it is not always possible to evaluate both of the jointly learned parts and sometimes it is even hard to imagine how to learn both parts separately. In Chapter 5, we performed joint learning over different subword structures beyond morphemes, e.g., submorphemes. The target structures here were morphemes, and all other structures played a helping role. It is easy to evaluate morphological segmentations learned both with and without submorphemes, but assessing the accuracy of the submorphemes themselves, with or without the morphemes, makes no sense. In a similar fashion, it is possible to evaluate the morphosyntactic clusters learned in Chapter 6 with and without the jointly learned morphological similarity function. However, it is not obvious how to learn the similarity function without the clustering and also, how to evaluate it. If we ignore those problems and evaluate what is possible—morphological segmentations in Chapter 5 and morphosyntactic clusterings in Chapter 6—then both tasks improve when using joint learning and thus support our Claim.

7.4 Conclusion

Although not all results presented supported our Claims, we were able to demonstrate that the joint learning of related morphological aspects in an unsupervised or weakly-supervised model is beneficial. On the other hand, those experiments that failed to support our hypotheses provided useful insights into the relationships between syntactic tags and morphological suffixes that refer to possible directions for future research.

Bibliography

Adler, M., Goldberg, Y., Gabay, D., and Elhadad, M. (2008). Unsupervised lexicon-based resolution of unknown words for full morphological analysis. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 728–736.

Ahlberg, M., Forsberg, M., and Hulden, M. (2014). Semi-supervised learning of morphological paradigms and lexicons. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 569–578.

Al-Rfou, R., Perozzi, B., and Skiena, S. (2013). Polyglot: Distributed word representations for multilingual NLP. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 183–192.

Aldous, D. (1985). Exchangeability and related topics. In *École d'été de probabilités de Saint-Flour, XIII—1983*, pages 1–198. Springer, Berlin.

Alumäe, T. (2012). Transcription system for semi-spontaneous Estonian speech. In *Proceedings of the Fifth International Conference of Human Language Technologies - The Baltic Perspective*, pages 10–17.

Alumäe, T. (2006). *Methods for Estonian Large Vocabulary Speech Recognition.* PhD thesis, Tallinn University of Technology.

Andrews, N., Eisner, J., and Dredze, M. (2014). Robust entity clustering via phylogenetic inference. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 775–785.

Arisoy, E., Dutağaci, H., and Arslan, L. M. (2006). A unified language model for large vocabulary continuous speech recognition of Turkish. *Signal Processing*, 86(10):2844–2862.

Arisoy, E., Saraclar, M., Roark, B., and Shafran, I. (2012). Discriminative language modeling with linguistic and statistically derived features. *IEEE Transactions on Audio, Speech & Language Processing*, 20(2):540–550.

Arısoy, E., Kurimo, M., Saraçlar, M., Hirsimäki, T., Pylkkönen, J., Alumäe, T., and Sak, H. (2008). Statistical language modeling for automatic speech recognition of agglutinative languages. *Speech Recognition : Technologies and Applications*, 10:193–204.

Beal, M., Ghahramani, Z., and Rasmussen, C. (2002). The infinite hidden Markov model. In *Advances in Neural Information Processing Systems 14*, pages 577–584. MIT Press.

Berg-Kirkpatrick, T., Côté, A. B., DeNero, J., and Klein, D. (2010). Painless unsupervised learning with features. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 582–590.

Bernhard, D. (2006). Unsupervised morphological segmentation based on segment predictability and word segments alignment. In *Proceedings of the PASCAL Challenge Workshop on Unsupervised segmentation of words into morphemes*, pages 19–23.

Bernhard, D. (2008). Simple morpheme labelling in unsupervised morpheme analysis. In *Advances in Multilingual and Multimodal Information Retrieval*, volume 5152 of *Lecture Notes in Computer Science*, pages 873–880.

Berton, A., Fetter, P., and Regel-Brietzmann, P. (1996). Compound words in large-vocabulary German speech recognition systems. In *The 4th International Conference on Spoken Language Processing*, volume 2, pages 1165–1168.

Biemann, C. (2006). Unsupervised part-of-speech tagging employing efficient graph clustering. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 7–12.

Biemann, C., Giuliano, C., and Gliozzo, A. (2007). Unsupervised part-of-speech tagging supporting supervised methods. In *Proceedings of Recent Advances in Natural Language Processing*.

Blei, D., Ng, A., and Jordan, M. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Blei, D. M. and Frazier, P. I. (2011). Distance dependent Chinese restaurant processes. *Journal of Machine Learning Research*, 12:2461–2488.

Blunsom, P. and Cohn, T. (2011). A hierarchical Pitman-Yor process HMM for unsupervised part of speech induction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 865–874.

Bojar, O. (2007). English-to-Czech factored machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 232–239.

Bordag, S. (2006). Two-step approach to unsupervised morpheme segmentation. In *Proceedings of the PASCAL Challenge Workshop on Unsupervised segmentation of words into morphemes*, pages 25–29.

Bordag, S. (2008). Unsupervised and knowledge-free morpheme segmentation and analysis. In *Advances in Multilingual and Multimodal Information Retrieval*, volume 5152 of *Lecture Notes in Computer Science*, pages 881–891.

Börschinger, B. and Johnson, M. (2014). Exploring the role of stress in Bayesian word segmentation using Adaptor Grammars. *Transactions of the Association for Computational Linguistics*, 2:93–104.

Botha, J. A. and Blunsom, P. (2013). Adaptor Grammars for learning non-concatenative morphology. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 345–356.

Brent, M. R., Murthy, S. K., and Lundberg, A. (1995). Discovering morphemic suffixes: A case study in MDL induction. In *The Fifth International Workshop on Artificial Intelligence and Statistics*, pages 264–271.

Brown, P., Della Pietra, V., de Souza, V., Lai, J., and Mercer, R. (1992). Class-based n-gram models of natural language. *Computational Linguistics*, 18:467–479.

Can, B. (2011). *Statistical Models for Unsupervised Learning of Morphology and POS Tagging*. PhD thesis, The University of York.

Can, B. and Manandhar, S. (2012). Probabilistic hierarchical clustering of morphological paradigms. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 654–663.

Carter, D. M., Kaja, J., Neumeyer, L., Rayner, M., Weng, F., and Wirén, M. (1996). Handling compound nouns in a Swedish speech-understanding system. In *The 4th International Conference on Spoken Language Processing*, pages 26–29.

Chater, N. and Manning, C. D. (2006). Probabilistic models of language processing and acquisition. *Trends in Cognitive Sciences*, 10(7):335–344.

Chen, S. F. and Goodman, J. (1998). An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Center for Research in Computing Technology, Harvard University.

Christodoulopoulos, C., Goldwater, S., and Steedman, M. (2010). Two decades of unsupervised POS induction: How far have we come? In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 575–584.

Christodoulopoulos, C., Goldwater, S., and Steedman, M. (2011). A Bayesian mixture model for part-of-speech induction using multiple features. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 638–647.

Chrupala, G. (2006). Simple data-driven context-sensitive lemmatization. *Procesamiento del Lenguaje Natural*, 37:121–130.

Clark, A. (2000). Inducing syntactic categories by context distribution clustering. In *Proceedings of the Conference on Natural Language Learning*, pages 91–94.

Clark, A. (2002). Memory-based learning of morphology with stochastic transducers. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 513–520.

Clark, A. (2003). Combining distributional and morphological information for part of speech induction. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics*, pages 59–66.

Clifton, A. and Sarkar, A. (2011). Combining morpheme-based machine translation with post-processing morpheme prediction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 32–42.

Cohen, S. B., Blei, D. M., and Smith, N. A. (2010). Variational inference for Adaptor Grammars. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 564–572.

Cohn, T., Goldwater, S., and Blunsom, P. (2009). Inducing compact but accurate tree-substitution grammars. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 548–556.

Creutz, M. (2003). Unsupervised segmentation of words using prior distributions of morph length and frequency. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 280–287.

Creutz, M., Hirsimäki, T., Kurimo, M., Puurula, A., Pylkkönen, J., Siivola, V., Varjokallio, M., Arisoy, E., Saraclar, M., and Stolcke, A. (2007). Morph-based speech recognition and modeling of out-of-vocabulary words across languages. *ACM Transactions on Speech and Language Processing*, 5(1):3:1–3:29.

Creutz, M. and Lagus, K. (2002). Unsupervised discovery of morphemes. In *Proceedings of the ACL-02 Workshop on Morphological and Phonological Learning*, pages 21–30.

Creutz, M. and Lagus, K. (2004). Induction of a simple morphology for highly-inflecting languages. In *Proceedings of the 7th Meeting of the ACL Special Interest Group in Computational Phonology: Current Themes in Computational Phonology and Morphology*, pages 43–51.

Creutz, M. and Lagus, K. (2005). Inducing the morphological lexicon of a natural language from unannotated text. In *Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning*, pages 51–59.

Creutz, M. and Lagus, K. (2007). Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions on Speech and Language Processing*, 4(1):3:1–3:34.

Dasgupta, S. and Ng, V. (2007). High-performance, language-independent morphological segmentation. In *Proceedings of Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 155–163.

de Gispert, A., Virpioja, S., Kurimo, M., and Byrne, W. (2009). Minimum Bayes risk combination of translation hypotheses from alternative morphological decompositions. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 73–76.

de Novais, E. M. and Paraboni, I. (2013). Portuguese text generation using factored language models. *Journal of the Brazilian Computer Society*, 19(2):135–146.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41:391–407.

Dejean, H. (1998). Morphemes as necessary concept for structures discovery from untagged corpora. In *New Methods in Language Processing and Computational Natural Language Learning*, pages 295–298.

Dempster, A., Laird, N., and Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical Society B*, 39:1–38.

Dreyer, M. and Eisner, J. (2011). Discovering morphological paradigms from plain text using a Dirichlet Process mixture model. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 616–627.

Dreyer, M., Smith, J. R., and Eisner, J. (2008). Latent-variable modeling of string transductions with finite-state methods. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1080–1089.

Dubbin, G. and Blunsom, P. (2014). Modelling the lexicon in unsupervised part of speech induction. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 116–125.

Durrett, G. and DeNero, J. (2013). Supervised learning of complete morphological paradigms. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1185–1195.

Durrett, G. and Klein, D. (2013). Easy victories and uphill battles in coreference resolution. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1971–1982.

El-Kahlout, I. D. and Oflazer, K. (2006). Initial explorations in English to Turkish statistical machine translation. In *Proceedings of the Workshop on Statistical Machine Translation*, pages 7–14.

Elsner, M., Charniak, E., and Johnson, M. (2009). Structured generative models for unsupervised named-entity clustering. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 164–172.

Elsner, M., Goldwater, S., and Eisenstein, J. (2012). Bootstrapping a unified model of lexical and phonetic acquisition. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 184–193.

Erjavec, T. (2004). MULTEXT-East version 3: Multilingual morphosyntactic specifications, lexicons and corpora. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, pages 1535–1538.

Erjavec, T. and Dcroski, S. (2004). Machine learning of morphosyntactic structure: Lemmatizing unknown Slovene words. *Applied Artificial Intelligence*, 18(1):17–41.

Escobar, M. D. and West, M. (1995). Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, 90(430):577–588.

Eyigöz, E., Gildea, D., and Oflazer, K. (2013). Simultaneous word-morpheme alignment for statistical machine translation. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 32–40.

Ferguson, T. S. (1973). A Bayesian analysis of some nonparametric problems. *Annals of Statistics*, 1:209–230.

Frank, S. (2014). Learning the hyperparameters to learn morphology. In *Proceedings of the 5th Workshop on Cognitive Aspects of Computational Language Learning (CogACLL)*, pages 14–18.

Fraser, A., Weller, M., Cahill, A., and Cap, F. (2012). Modeling inflection and word-formation in SMT. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 664–674.

Gamon, M., Ringger, E., Corston-Oliver, S., and Moore, R. (2002). Machine-learned contexts for linguistic operations in German sentence realization. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 25–32.

Ganchev, K., Taskar, B., Pereira, F., and Gama, J. (2009). Posterior vs parameter sparsity in latent variable models. In *Advances in Neural Information Processing Systems 22*, pages 664–672.

Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2004). *Bayesian Data Analysis*. Chapman & Hall/CRC, New York.

Georgiev, G., Zhikov, V., Osenova, P., Simov, K., and Nakov, P. (2012). Feature-rich part-of-speech tagging for morphologically complex languages: Application to Bulgarian. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 492–502.

Gesmundo, A. and Samardžić, T. (2012). Lemmatisation as a tagging task. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 368–372.

Geutner, P. (1995). Using morphology towards better large vocabulary speech-recognition systems. In *Proceedings of International Conference of Acoustics, Speech, and Signal Processing, 1995*, pages 445–448.

Goldsmith, J. (2001). Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, 27:153–198.

Goldsmith, J. (2006). An algorithm for the unsupervised learning of morphology. *Journal of Natural Language Engineering*, 12:353–371.

Goldwater, S. (2006). *Nonparametric Bayesian Models of Lexical Acquisition.* PhD thesis, Brown University.

Goldwater, S. and Griffiths, T. L. (2007). A fully Bayesian approach to unsupervised part-of-speech tagging. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 744–751.

Goldwater, S., Griffiths, T. L., and Johnson, M. (2006a). Contextual dependencies in unsupervised word segmentation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 673–680.

Goldwater, S., Griffiths, T. L., and Johnson, M. (2006b). Interpolating between types and tokens by estimating power-law generators. In *Advances in Neural Information Processing Systems 18*, pages 459–466.

Goldwater, S., Griffiths, T. L., and Johnson, M. (2009). A Bayesian framework for word segmentation: Exploring the effects of context. *Cognition*, 112(1):21–54.

Goldwater, S., Griffiths, T. L., and Johnson, M. (2011). Producing power-law distributions and damping word frequencies with two-stage language models. *Journal of Machine Learning Research*, 12(Jul):2335–2382.

Goldwater, S. and McClosky, D. (2005). Improving statistical MT through morphological analysis. In *Proceedings of the 2005 Conference on Empirical Methods in Natural Language Processing*, pages 676–683.

Grönroos, S.-A., Virpioja, S., Smit, P., and Kurimo, M. (2014). Morfessor FlatCat: An HMM-based method for unsupervised and semi-supervised learning of morphology. In *Proceedings of the 25th International Conference on Computational Linguistics*, pages 1177–1185.

Hafer, M. A. and Weiss, S. F. (1974). Word segmentation by letter successor varieties. *Information Storage and Retrieval*, 10(11-12):371–385.

Haghighi, A. and Klein, D. (2006). Prototype-driven learning for sequence models. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 320–327.

Haghighi, A. and Klein, D. (2010). Coreference resolution in a modular, entity-centered model. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 385–393.

Hajič, J. (2000). Morphological tagging: Data vs. dictionaries. In *Proceedings of 1st Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 94–101.

Hardisty, E. A., Boyd-Graber, J., and Resnik, P. (2010). Modeling perspective using Adaptor Grammars. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 284–292.

Harris, Z. (1955). From phoneme to morpheme. *Language*, 31:190–222.

Hasan, K. S. and Ng, V. (2009). Weakly supervised part-of-speech tagging for morphologically-rich, resource-scarce languages. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 363–371.

Haspelmath, M. and Sims, A. D. (2010). *Understanding Morphology: 2nd edition*. Hodder Education, London.

Hirsimäki, T., Pylkkönen, J., and Kurimo, M. (2009). Importance of high-order n-gram models in morph-based speech recognition. *IEEE Transactions on Audio, Speech & Language Processing*, 17(4):724–732.

Hofmann, T. (1999). Probabilistic latent semantic analysis. In *Proceedings of Uncertainty in Artificial Intelligence*, pages 289–296.

Hollink, V., Kamps, J., Monz, C., and de Rijke, M. (2004). Monolingual document retrieval for European languages. *Information Retrieval*, 7(1-2):33–52.

Huang, E. H., Socher, R., Manning, C. D., and Ng, A. Y. (2012). Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Huang, Y., Zhang, M., and Tan, C. L. (2011). Nonparametric Bayesian machine transliteration with synchronous Adaptor Grammars. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 534–539.

Ircing, P., Krbec, P., Hajic, J., Psutka, J., Khudanpur, S., Jelinek, F., and Byrne, W. (2001). On large vocabulary continuous speech recognition of highly inflectional language - Czech. In *Proceedings of the 7th European Conference on Speech Communication and Technology*, pages 487–490.

Johnson, M. (2007). Why doesn't EM find good HMM POS-taggers? In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 296–305.

Johnson, M. (2008a). Unsupervised word segmentation for Sesotho using Adaptor Grammars. In *Proceedings of the Tenth Meeting of ACL Special Interest Group on Computational Morphology and Phonology*.

Johnson, M. (2008b). Using Adaptor Grammars to identify synergies in the unsupervised acquisition of linguistic structure. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 398–406.

Johnson, M. (2010). PCFGs, topic models, Adaptor Grammars and learning topical collocations and the structure of proper names. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1148–1157.

Johnson, M. and Demuth, K. (2010). Unsupervised phonemic Chinese word segmentation using Adaptor Grammars. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 528–536.

Johnson, M., Demuth, K., and Frank, M. (2012). Exploiting social information in grounded language learning via grammatical reductions. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, pages 883–891.

Johnson, M., Demuth, K., Jones, B., and Black, M. J. (2010). Synergies in learning words and their referents. In *Advances in Neural Information Processing Systems 23*, pages 1018–1026.

Johnson, M. and Goldwater, S. (2009). Improving nonparametric Bayesian inference: Experiments on unsupervised word segmentation with Adaptor Grammars. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 317–325.

Johnson, M., Griffiths, T. L., and Goldwater, S. (2007). Adaptor Grammars: a framework for specifying compositional nonparametric Bayesian models. In *Advances in Neural Information Processing Systems 19*, pages 641–648.

Jusic, M., Mozetic, I., Erjavec, T., and Lavrac, N. (2010). LemmaGen: Multilingual lemmatisation with induced ripple-down rules. *Journal of Universal Computer Science*, 16(9):1190–1214.

Kamper, H. (2013). Gibbs sampling for fitting finite and infinite Gaussian mixture models. Available at `http://www.kamperh.com/notes/kamper_bayesgmm13.pdf`.

Kaplan, R. and Kay, M. (1981). Phonological rules and finite-state transducers. In *Linguistic Society of America Meeting Handbook*. 56th Annual Meeting.

Keshava, S. and Pitler, E. (2006). A simpler, intuitive approach to morpheme induction. In *Proceedings of the PASCAL Challenge Workshop on Unsupervised segmentation of words into morphemes*, pages 31–35.

Kim, J.-K. and de Marneffe, M.-C. (2013). Deriving adjectival scales from continuous space word representations. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1625–1630.

Kirchhoff, K., Vergyri, D., Bilmes, J., Duh, K., and Stolcke, A. (2006). Morphology-based language modeling for conversational Arabic speech recognition. *Computer Speech & Language*, 20(4):589–608.

Kohonen, O., Virpioja, S., and Klami, M. (2009). Allomorfessor: Towards unsupervised morpheme analysis. In *Evaluating Systems for Multilingual and Multimodal Information Access*, volume 5706 of *Lecture Notes in Computer Science*, pages 975–982.

Kohonen, O., Virpioja, S., and Lagus, K. (2010a). Semi-supervised learning of concatenative morphology. In *Proceedings of the 11th Meeting of the ACL Special Interest Group on Computational Morphology and Phonology*, pages 78–86.

Kohonen, O., Virpioja, S., Leppänen, L., and Lagus, K. (2010b). Semi-supervised extensions to Morfessor baseline. In *Proceedings of the Morpho Challenge 2010 Workshop*, pages 30–34. Aalto University School of Science and Technology.

Koskenniemi, K. (1983). Two-level morphology: A general computational model for word-form recognition and production. Technical Report 11, University of Helsinki, Department of General Linguistics.

Krovetz, R. (1993). Viewing morphology as an inference process. In *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 191–203.

Kurimo, M., Creutz, M., and Turunen, V. (2009). Morpho Challenge evaluation by information retrieval experiments. In *Evaluating Systems for Multilingual and Multimodal Information Access*, volume 5706 of *Lecture Notes in Computer Science*, pages 991–998.

Lafferty, J. D., McCallum, A., and Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289.

Lamar, M., Maron, Y., Johnson, M., and Bienenstock, E. (2010). SVD and clustering for unsupervised POS tagging. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics Short Papers*, pages 215–219.

Larson, M., Willett, D., Köhler, J., and Rigoll, G. (2000). Compound splitting and lexical unit recombination for improved performance of a speech recognition system for German parliamentary speeches. In *Sixth International Conference on Spoken Language Processing*, pages 945–948.

Lee, Y. (2004). Morphological analysis for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 57–60.

Lee, Y. K., Haghighi, A., and Barzilay, R. (2010). Simple type-level unsupervised POS tagging. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 853–861.

Lee, Y. K., Haghighi, A., and Barzilay, R. (2011). Modeling syntactic context improves morphological segmentation. In *Proceedings of the Fifteenth Conference on Natural Language Learning*, pages 1–9.

Levy, O. and Goldberg, Y. (2014). Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308.

Lignos, C. (2010). Learning from Unseen Data. In *Proceedings of the Morpho Challenge 2010 Workshop*, pages 35–38. Aalto University School of Science and Technology.

Lovins, J. B. (1968). Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics*, 11:22–31.

Luong, M.-T., Nakov, P., and Kan, M.-Y. (2010). A hybrid morpheme-word representation for machine translation of morphologically rich languages. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 148–157.

Luong, M.-T., Socher, R., and Manning, C. D. (2013). Better word representations with recursive neural networks for morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 104–113.

MacKay, D. (2003). *Information Theory, Inference and Learning Algorithms*. Cambridge University Press.

Marcus, M. P., Marcinkiewicz, M. A., and Santorini, B. (1993). Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, 19(2):313–330.

Meilă, M. (2002). Comparing clusterings. Technical Report 418, University of Washington Statistics Department.

Merialdo, B. (1994). Tagging English text with a probabilistic model. *Computational Linguistics*, 20(2):155–172.

Mihajlik, P., Fegyó, T., Tüske, Z., and Ircing, P. (2007). A morpho-graphemic approach for the recognition of spontaneous speech in agglutinative languages - like Hungarian. In *Proceedings of the 8th Annual Conference of the International Speech Communication Association (Interspeech 2007)*, pages 1497–1500.

Mikolov, T., Kombrink, S., Deoras, A., Burget, L., and Cernocky, J. (2011). RNNLM - recurrent neural network language modeling toolkit. In *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding*.

Mikolov, T., Yih, W.-t., and Zweig, G. (2013). Linguistic regularities in continuous space word representations. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 746–751.

Minkov, E., Toutanova, K., and Suzuki, H. (2007). Generating complex morphology for machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 128–135.

Mochihashi, D., Yamada, T., and Ueda, N. (2009). Bayesian unsupervised word segmentation with nested Pitman-Yor language modeling. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 100–108.

Monson, C., Carbonell, J., Lavie, A., and Levin, L. (2007). Paramor: Minimally supervised induction of paradigm structure and morphological analysis. In *Proceedings of Ninth Meeting of the ACL Special Interest Group in Computational Morphology and Phonology*, pages 117–125.

Müller, T., Schmid, H., and Schütze, H. (2013). Efficient higher-order CRFs for morphological tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 322–332.

Munkres, J. (1957). Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics*, 5(1):32–38.

Murphy, K. P. (2012). *Machine learning: a Probabilistic Perspective*. The MIT Press.

Neal, R. M. (2003). Slice sampling. *Annals of Statistics*, 31:705–767.

Nguyen, T. and Vogel, S. (2008). Context-based Arabic morphological analysis for machine translation. In *Proceedings of the Twelfth Conference on Natural Language Learning*, pages 135–142.

Niessen, S. and Ney, H. (2000). Improving SMT quality with morpho-syntactic analysis. In *Proceedings of the 18th International Conference on Computational Linguistics*, pages 1081–1085.

Niessen, S. and Ney, H. (2004). Statistical machine translation with scarce resources using morpho-syntactic analysis. *Computational Linguistics*, 30(2):181–204.

Oflazer, K. and El-Kahlout, I. D. (2007). Exploring different representational units in English-to-Turkish statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 25–32.

Ostler, N. (2003). Introduction to language engineering for lesser studied languages - linguistic aspects. In *Language engineering for lesser studied languages*, pages 1–20.

Paice, C. D. (1990). Another stemmer. *SIGIR Forum*, 24(3):56–61.

Petrov, S., Das, D., and McDonald, R. (2012). A universal part-of-speech tagset. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation*, pages 2089–2096.

Pitman, J. and Yor, M. (1997). The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *Annals of Probability*, 25:855–900.

Poon, H., Cherry, C., and Toutanova, K. (2009). Unsupervised morphological segmentation with log-linear models. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 209–217.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program*, 14(3):130–137.

Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, volume 77, pages 257–286.

Rasmussen, C. (2000). The infinite Gaussian mixture model. In *Advances in Neural Information Processing Systems 12*, pages 554–560.

Ravi, S. and Knight, K. (2009). Minimized models for unsupervised part-of-speech tagging. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 504–512.

Reichart, R., Abend, O., and Rappoport, A. (2010). Type level clustering evaluation: New measures and a POS induction case study. In *Proceedings of the Fourteenth Conference on Natural Language Learning*, pages 77–87.

Reichart, R. and Rappoport, A. (2009). The NVI clustering evaluation measure. In *Proceedings of the Thirteenth Conference on Natural Language Learning*, pages 165–173.

Reynolds, D. (2009). Gaussian mixture models. In Li, S. and Jain, A., editors, *Encyclopedia of Biometrics*, pages 659–663. Springer US.

Rissanen, J. (1989). *Stochastic Complexity and Statistical Inquiry*. World Scientific Publishing Co., Inc., River Edge, NJ, USA.

Rosenberg, A. and Hirschberg, J. (2007). V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 410–42.

Ruokolainen, T., Kohonen, O., Virpioja, S., and Kurimo, M. (2013). Supervised morphological segmentation in a low-resource learning setting using conditional random fields. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 29–37.

Ruokolainen, T., Kohonen, O., Virpioja, S., and Kurimo, M. (2014). Painless semi-supervised morphological segmentation using conditional random fields. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, volume 2: Short Papers*, pages 84–89.

Salameh, M., Cherry, C., and Kondrak, G. (2014). Lattice desegmentation for statistical machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 100–110.

Shen, L., Satta, G., and Joshi, A. (2007). Guided learning for bidirectional sequence classification. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 760–767.

Sirts, K. and Alumäe, T. (2012). A hierarchical Dirichlet process model for joint part-of-speech and morphology induction. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 407–416.

Sirts, K., Eisenstein, J., Elsner, M., and Goldwater, S. (2014). POS induction with distributional and morphological information using a distance-dependent Chinese restaurant process. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 265–271.

Sirts, K. and Goldwater, S. (2013). Minimally-supervised morphological segmentation using adaptor grammars. *Transactions of the Association for Computational Linguistics*, 1(May):231–242.

Smets, M., Gamon, M., Corston-Oliver, S., and Ringger, E. K. (2003). French amalgam: a quick adaptation of a sentence realization system to French. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics*, pages 323–330.

Smith, N. and Eisner, J. (2005). Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 354–362.

Snover, M. and Brent, M. (2003). A probabilistic model for learning concatenative morphology. In *Advances in Neural Information Processing Systems 15*, pages 1537–1544.

Snover, M., Jarosz, G., and Brent, M. (2002). Unsupervised learning of morphology using a novel directed search algorithm: Taking the first step. In *Proceedings of the ACL-02 Workshop on Morphological and Phonological Learning*, pages 11–20.

Snover, M. G. and Brent, M. R. (2001). A Bayesian model for morpheme and paradigm identification. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pages 490–498.

Snyder, B. and Barzilay, R. (2008a). Cross-lingual propagation for morphological analysis. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2*, pages 848–854.

Snyder, B. and Barzilay, R. (2008b). Unsupervised multilingual learning for morphological segmentation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 737–745.

Socher, R., Maas, A. L., and Manning, C. D. (2011). Spectral Chinese restaurant processes: Nonparametric clustering based on similarities. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, pages 698–706.

Spiegler, S. and Monson, C. (2010). EMMA: A novel evaluation metric for morphological analysis. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1029–1037.

Spitkovsky, V. I., Alshawi, H., Chang, A. X., and Jurafsky, D. (2011). Unsupervised dependency parsing without gold part-of-speech tags. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1281–1290.

Stas, J., Hladek, D., Juhar, J., and Zlacky, D. (2012). Analysis of morph-based language modeling and speech recognition in Slovak. *Advances in Electrical and Electronic Engineering*, 10(4):291–296.

Stroppa, N. and Yvon, F. (2005). An analogical learner for morphological analysis. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, pages 120–127.

Sudderth, E. B. (2006). *Graphical Models for Visual Object Recognition and Tracking*. PhD thesis, MIT.

Teemu Hirsimäki, Mathias Creutz, V. S. and Kurimo, M. (2005). Morphologically motivated language models in speech recognition. In *Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning*, pages 121–126.

Teh, Y. W. (2006a). A Bayesian interpretation of interpolated Kneser-Ney. Technical Report TRA2/06, National University of Singapore, School of Computing.

Teh, Y. W. (2006b). A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 985–992.

Teh, Y. W. (2007). Exponential families: Gaussian, Gaussian-Gamma, Gaussian-Wishart, multinomial. Available at `http://www.stats.ox.ac.uk/~teh/research/notes/GaussianInverseWishart.pdf`.

Teh, Y. W. (2010). Dirichlet process. In *Encyclopedia of Machine Learning*, pages 280–287. Springer US.

Teh, Y. W., Jordan, M. I., Beal, M. J., and Blei, D. M. (2006). Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.

Titov, I. and Klementiev, A. (2012). A Bayesian approach to unsupervised semantic role induction. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 12–22.

Toutanova, K. and Cherry, C. (2009). A global model for joint lemmatization and part-of-speech prediction. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 486–494.

Toutanova, K. and Johnson, M. (2008). A Bayesian LDA-based model for semi-supervised part-of-speech tagging. In *Advances in Neural Information Processing Systems 20*, pages 1521–1528.

Toutanova, K., Klein, D., Manning, C. D., and Singer, Y. (2003). Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 173–180.

Toutanova, K., Suzuki, H., and Ruopp, A. (2008). Applying morphology generation models to machine translation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 514–522.

Turian, J., Ratinov, L.-A., and Bengio, Y. (2010). Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394.

van den Bosch, A. and Daelemans, W. (1999). Memory-based morphological analysis. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 285–292.

Van Gael, J., Vlachos, A., and Ghahramani, Z. (2009). The infinite HMM for unsupervised PoS tagging. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 678–687.

Virpioja, S., Smit, P., Grönroos, S.-A., and Kurimo, M. (2013). Morfessor 2.0: Python implementation and extensions for Morfessor Baseline. Technical Report 25/2013 in Aalto University publication series SCIENCE + TECHNOLOGY, Department of Signal Processing and Acoustics, Aalto University.

Virpioja, S., Turunen, V. T., Spiegler, S., Kohonen, O., and Kurimo, M. (2011). Empirical comparison of evaluation methods for unsupervised learning of morphology. *Traitement Automatique des Langues*, 52(2):45–90.

Virpioja, S., Väyrynen, J. J., Creutz, M., and Sadeniemi, M. (2007). Morphology-aware statistical machine translation based on morphs induced in an unsupervised manner. In *Proceedings of the Machine Translation Summit XI*, pages 491–498.

Vlachos, A. (2011). Evaluating unsupervised learning for natural language processing tasks. In *Proceedings of the First workshop on Unsupervised Learning in NLP*, pages 35–42.

Wei, G. C. and Tanner, M. A. (1990). A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms. *Journal of the American Statistical Association*, 85(411):699–704.

Wicentowski, R. (2002). *Modeling and Learning Multilingual Inflectional Morphology in a Minimally Supervised Framework*. PhD thesis, The Johns Hopkins University.

Wong, S.-M. J., Dras, M., and Johnson, M. (2012). Exploring Adaptor Grammars for native language identification. In *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing*, pages 699–709.

Wood, F., Archambeau, C., Gasthaus, J., James, L., and Teh, Y. W. (2009). A stochastic memoizer for sequence data. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1129–1136.

Yarowsky, D. and Wicentowski, R. (2000). Minimally supervised morphological analysis by multimodal alignment. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 207–216.

Yatbaz, M. A., Sert, E., and Yuret, D. (2012). Learning syntactic categories using paradigmatic representations of word context. In *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing*, pages 940–951.

Yeniterzi, R. and Oflazer, K. (2010). Syntax-to-morphology mapping in factored phrase-based statistical machine translation from English to Turkish. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 454–464.

Yuret, D., Yatbaz, M. A., and Sert, E. (2014). Unsupervised instance-based part of speech induction using probable substitutes. In *Proceedings of the 25th International Conference on Computational Linguistics*, pages 2303–2313.

Zeman, D. (2008). Unsupervised acquiring of morphological paradigms from tokenized text. In *Advances in Multilingual and Multimodal Information Retrieval*, volume 5152 of *Lecture Notes in Computer Science*, pages 892–899.

Zhai, K., Boyd-Graber, J., and Cohen, S. B. (2014). Online Adaptor Grammars with hybrid inference. 2:465–476.

Zhao, Q. and Marcus, M. (2012). Long-tail distributions and unsupervised learning of morphology. In *Proceedings of the 24th International Conference on Computational Linguistics*, pages 3121–3136.

Zipf, G. (1932). *Selective Studies and the Principle of Relative Frequency in Language*. Harvard University Press, Cambridge, MA.

Appendix A

Bayesian inference for multivariate normal distribution

The normal distribution is a common choice when dealing with continuous random variables. Using normal distribution in mixture models enables representing complicated probability distributions with relatively simple models. Often, there are no reasons to make strong assumptions about the parameters of the normal model, especially in the mixture model setting where the distribution over data might be very complicated. In this situation the Bayesian treatment is a good option, which means setting priors over the mean and covariance parameters and computing the posterior distribution of the parameters using the Bayes rule.

The Bayesian inference for normal models is discussed in various sources. However, the parameterization of prior distributions is slightly different in different sources. Also, there are several possible ways for computing the posterior predictive distribution. This Appendix attempts to give a full description of the Bayesian inference for the multivariate normal distribution that could be used as a reference material. The univariate case is omitted because it can be readily derived from the multivariate case, and it has been fully described in (Rasmussen, 2000). The material follows (Murphy, 2012) sections 4.5 and 4.6 and (Gelman et al., 2004) section 3.6. The form of the prior distribution parameterization follows (Gelman et al., 2004)[1].

[1]Different sources parameterize inverse-Wishart distribution differently that can lead to some confusion. For example, (Murphy, 2012) uses $\mathcal{IW}(\mathbf{\Sigma}|\mathbf{\Lambda}, \nu)$, wherease (Gelman et al., 2004) use the notation $\mathcal{IW}(\mathbf{\Sigma}|\mathbf{\Lambda}^{-1}, \nu)$ which raises the question of why the scale parameter is inverted. Here we use the form of density given in (Gelman et al., 2004) but present it with non-inverted scale: $\mathcal{IW}(\mathbf{\Sigma}|\mathbf{\Lambda}, \nu)$. The confusion even increases when one attempts to

A.1 Data likelihood

If a point $\mathbf{x}$ is a multivariate Gaussian random variable with a mean vector $\boldsymbol{\mu}$ and a covariance matrix $\boldsymbol{\Sigma}$

$$\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{A.1}$$

then its likelihood is given with the multivariate Gaussian density:

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right), \tag{A.2}$$

where d is the dimensionality of the data vector. When multiplying together the densities of several independent points $\boldsymbol{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, the multiplication can be pushed into the exponent as a summation:

$$\mathcal{N}(\boldsymbol{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{i=1}^{n} \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}_i-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}_i-\boldsymbol{\mu})\right) \tag{A.3}$$

$$= \frac{1}{(2\pi)^{nd/2}|\boldsymbol{\Sigma}|^{n/2}} \exp\left(-\frac{1}{2}\sum_{i=1}^{n}(\mathbf{x}_i-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}_i-\boldsymbol{\mu})\right) \tag{A.4}$$

$$= \frac{1}{(2\pi)^{nd/2}|\boldsymbol{\Sigma}|^{n/2}} \exp\left(-\frac{n}{2}(\boldsymbol{\mu}-\bar{\mathbf{x}})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}-\bar{\mathbf{x}}) - \frac{1}{2}\mathrm{tr}(\boldsymbol{\Sigma}^{-1}\mathbf{S}_{\bar{\mathbf{x}}})\right), \tag{A.5}$$

where $\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$ is the sample mean, $\mathrm{tr}(A) = \sum_{i=1}^{n} A_{ii}$ is the matrix trace operator and

$$\mathbf{S}_{\bar{\mathbf{x}}} = \sum_{i=1}^{n}(\mathbf{x}_i-\bar{\mathbf{x}})(\mathbf{x}_i-\bar{\mathbf{x}})^T \tag{A.6}$$

is the scatter matrix that can be viewed as the unnormalized version of the covariance matrix. (A.5) is the quadratic form of the likelihood that will be useful later in section A.7 for computing the joint posterior distribution.

A.2 Prior for the covariance

A popular choice for the covariance matrix prior distribution is the *inverse-Wishart* ($\mathcal{IW}$) distribution. $\mathcal{IW}$ is a distribution over symmetric positive-definite matrices, parameterized by the degrees of freedom ν and a positive-definite scale matrix $\boldsymbol{\Lambda}$. The probability density of an $\mathcal{IW}$-distributed random variable $\boldsymbol{\Sigma}$ is:

understand the exact relation between Wishart and inverse-Wishart distributions but this question is beyond the scope of this Appendix.

$$\mathcal{IW}(\boldsymbol{\Sigma}|\boldsymbol{\Lambda},\nu) = \frac{|\boldsymbol{\Lambda}|^{\nu/2}|\boldsymbol{\Sigma}|^{-(\nu+d+1)/2}}{2^{\nu d/2}\Gamma_d(\nu/2)} \exp\left(-\frac{1}{2}\mathrm{tr}(\boldsymbol{\Lambda}\boldsymbol{\Sigma}^{-1})\right), \tag{A.7}$$

where $\Gamma_d(\nu/2)$ is the multivariate gamma function and can be computed as:

$$\Gamma_d(\nu/2) = \pi^{d(d-1)/4} \prod_{i=1}^{d} \Gamma\left(\frac{\nu+1-i}{2}\right) \tag{A.8}$$

The multivariate gamma function can only be computed when $\nu > d-1$ and hence, this is the requirement for the existence of the proper $\mathcal{IW}$ probability density. The mean of the $\mathcal{IW}$ distribution is:

$$E\left[\boldsymbol{\Sigma}\right] = \frac{\boldsymbol{\Lambda}}{\nu-d-1} \tag{A.9}$$

and it exists only when $\nu > d+1$.

In the context of the multivariate Gaussian inference, the scale matrix $\boldsymbol{\Lambda}$ of the $\mathcal{IW}$ prior is proportional to the prior mean for $\boldsymbol{\Sigma}$ and the degrees of freedom ν indicates the strength of the belief into this mean.

The $\mathcal{IW}$ distribution is not the only choice possible for the covariance matrix prior distribution. However, it is a convenient one because it leads to a conjugate[2] model as will be clear later.

A.3 Prior for the mean

The conjugate prior distribution for the Gaussian mean vector is again a multivariate Gaussian:

$$\boldsymbol{\mu} \sim \mathcal{N}\left(\mathbf{m}, \frac{\boldsymbol{\Sigma}}{\kappa}\right), \tag{A.10}$$

where the covariance $\boldsymbol{\Sigma}$ is $\mathcal{IW}$-distributed, $\mathbf{m}$ is the prior mean for $\boldsymbol{\mu}$ and κ is the pseudo-count expressing the strength of the belief into $\mathbf{m}$.

A.4 Joint prior for the Gaussian parameters

The priors described in the previous two sections are coupled—the prior for the Gaussian mean is dependent on the covariance. This is appropriate because the mean and covariance are also coupled in the Gaussian likelihood. Choosing the priors in this way leads to a fully conjugate joint prior distribution over Gaussian parameters and is called the *Normal-inverse-Wishart*

[2]Conjugation in Bayesian terms means that the prior and posterior distributions have the same form, for example normal distribution is conjugate prior for the normal likelihood because multiplying them together leads to the normal posterior distribution.

($\mathcal{NIW}$) distribution. $\mathcal{NIW}$ is a four-parameter distribution over pairs of normally distributed vectors and $\mathcal{IW}$-distributed matrices parameterized by a prior mean vector $\mathbf{m}_0$, pseudo-count κ_0, degrees of freedom ν_0 and a prior scale matrix $\mathbf{\Lambda}_0$:

$$\boldsymbol{\mu}, \mathbf{\Sigma} \sim \mathcal{NIW}(\mathbf{m}_0, \kappa_0, \nu_0, \mathbf{\Lambda}_0) \tag{A.11}$$

$\mathcal{NIW}$ is obtained by combining the prior probabilities of the mean vector and the covariance matrix:

$$\mathcal{NIW}(\boldsymbol{\mu}, \mathbf{\Sigma}|\mathbf{m}_0, \kappa_0, \nu_0, \mathbf{\Lambda}_0) = \mathcal{N}\left(\boldsymbol{\mu}|\mathbf{m}_0, \frac{\mathbf{\Sigma}}{\kappa_0}\right) \times \mathcal{IW}(\mathbf{\Sigma}|\mathbf{\Lambda}_0, \nu_0) \tag{A.12}$$

The hyperparameters are indexed with 0 to emphasize the fact that they are prior hyperparameters. The probability density function after gathering some terms and joining the partition functions of both distributions is:

$$\begin{aligned}
&\mathcal{NIW}(\boldsymbol{\mu}, \mathbf{\Sigma}|\mathbf{m}_0, \kappa_0, \nu_0, \mathbf{\Lambda}_0) = \\
&= \frac{|\mathbf{\Sigma}|^{-(\nu_0+d+2)/2}}{2^{\nu_0 d/2}\Gamma_d(\nu_0/2)(2\pi/\kappa_0)^{d/2}|\mathbf{\Lambda}_0|^{-\nu_0/2}} \\
&\times \exp\left(-\frac{\kappa_0}{2}(\boldsymbol{\mu}-\mathbf{m}_0)^T\mathbf{\Sigma}^{-1}(\boldsymbol{\mu}-\mathbf{m}_0) - \frac{1}{2}\mathrm{tr}(\mathbf{\Sigma}^{-1}\mathbf{\Lambda}_0)\right)
\end{aligned} \tag{A.13}$$

Note that the form of the exponent in (A.13) is similar to (A.5).

A.5 Joint posterior distribution

The $\mathcal{NIW}$ conjugacy to the Gaussian likelihood leads to an $\mathcal{NIW}$ posterior distribution over Gaussian parameters:

$$P(\boldsymbol{\mu}, \mathbf{\Sigma}|\mathbf{X}; \mathbf{m}_0, \kappa_0, \nu_0, \mathbf{\Lambda}_0) = \mathcal{NIW}(\boldsymbol{\mu}, \mathbf{\Sigma}|\mathbf{m}_n, \kappa_n, \nu_n, \mathbf{\Lambda}_n), \tag{A.14}$$

where n is the size of the sample $\mathbf{X}$, and the posterior hyperparameters are computed as follows:

$$\kappa_n = \kappa_0 + n \tag{A.15}$$

$$\nu_n = \nu_0 + n \tag{A.16}$$

$$\mathbf{m}_n = \frac{\kappa_0\mathbf{m}_0 + n\bar{\mathbf{x}}}{\kappa_n} \tag{A.17}$$

$$\mathbf{\Lambda}_n = \mathbf{\Lambda}_0 + \mathbf{S}_{\bar{\mathbf{x}}} + \frac{\kappa_0 n}{\kappa_0 + n}(\bar{\mathbf{x}} - \mathbf{m}_0)(\bar{\mathbf{x}} - \mathbf{m}_0)^T \tag{A.18}$$

$$= \mathbf{\Lambda}_0 + \mathbf{S}_0 + \kappa_0\mathbf{m}_0\mathbf{m}_0^T - \kappa_n\mathbf{m}_n\mathbf{m}_n^T \tag{A.19}$$

The posterior mean $\mathbf{m}_n$ is just the weighted average of the prior and the sample means. $\mathbf{S}_0 = \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^T$ is the uncentered scatter matrix of evidence. The posterior scatter matrix $\mathbf{\Lambda}_n$ is just the evidence added to the prior plus an extra term that represents the uncertainty in the mean.

A.6 Posterior predictive distribution

Typically in Bayesian modeling one wants to make predictions for new data given some evidence. This is done by using the posterior predictive distribution where the actual parameter values are integrated out. Posterior predictive distribution is also employed in Bayesian inference where it is used to compute the likelihood of a point given the model.

$$\begin{aligned} P(\mathbf{x}|\mathbf{X}; \mathbf{m}_0, \kappa_0, \nu_0, \mathbf{\Lambda}_0) = \int_{\boldsymbol{\mu}} \int_{\mathbf{\Sigma}} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \mathbf{\Sigma}) \\ \times \mathcal{NIW}(\boldsymbol{\mu}, \mathbf{\Sigma}|\mathbf{m}_n, \kappa_n, \nu_n, \mathbf{\Lambda}_n) d\boldsymbol{\mu} d\mathbf{\Sigma} \end{aligned} \tag{A.20}$$

This integral has the form of a multivariate Student-T distribution with parameters derived from the $\mathcal{NIW}$ posterior hyperparameters:

$$\mathbf{x}|\mathbf{X}; \mathbf{m}_0, \kappa_0, \nu_0, \mathbf{\Lambda}_0 \sim \mathcal{T}_d(\mathbf{m}_n, \frac{\kappa_n + 1}{\kappa_n(\nu_n - d + 1)} \mathbf{\Lambda}_n, \nu_n - d + 1) \tag{A.21}$$

A multivariate Student-T distribution is parameterized by a location $\boldsymbol{\mu}$, a $d \times d$ positive-definite scale matrix $\mathbf{\Lambda}$ and the degrees of freedom ν and its density is given by the formula:

$$\mathcal{T}_d(\mathbf{x}|\boldsymbol{\mu}, \mathbf{\Lambda}, \nu) = \frac{\Gamma((\nu + d)/2)|\mathbf{\Lambda}|^{-1/2}}{\Gamma(\nu/2)\nu^{d/2}\pi^{d/2}} \left(1 + \frac{1}{\nu}(\mathbf{x} - \boldsymbol{\mu})^T \mathbf{\Lambda}^{-1}(\mathbf{x} - \boldsymbol{\mu})\right)^{-\frac{\nu+d}{2}} \tag{A.22}$$

A Student-T distribution has heavier tails than a Gaussian and thus it is well-suited in situations where there is uncertainty in the covariance. However, when the dimensionality d is small and the $\mathcal{NIW}$ degrees of freedom $\nu_n > d - 1$ this can be accurately approximated with a moment-matched Gaussian (Sudderth, 2006):

$$\mathbf{x}|\mathbf{X}; \mathbf{m}_0, \kappa_0, \nu_0, \mathbf{\Lambda}_0 \sim \mathcal{N}\left(\mathbf{m}_n, \frac{(\kappa_n + 1)\nu_n}{\kappa_n(\nu_n - d - 1)} \mathbf{\Lambda}_n\right) \tag{A.23}$$

Using the posterior predictive given in (A.21) is convenient when it has to be computed for a single point only. In case of a set of several points the posterior hyperparameters have to be updated and a new Student-T distribution constructed after the probability calculation of each point,

which can become computationally burdensome with large sets of points. Fortunately, the same posterior predictive can be expressed also in terms of the $\mathcal{NIW}$ normalization constants (Teh, 2007; Kamper, 2013), which only requires constructing two distributions—first the posterior based on the evidence provided by $\mathbf{X}$ and then the posterior including also the new set of points. This expression is derived from the marginal likelihood, which takes the the joint distribution over parameters and data and integrates over the parameters. In order to get there we first derive the full joint distribution.

A.7 Full joint distribution

The full joint distribution is necessary for computing the marginal data likelihood, which in turn is used to derive the computationally efficient version of the posterior predictive, which is the formula actually implemented for inference in this dissertation. The full joint probability of the data and the parameters is obtained by multiplying the Gaussian likelihood (we use the quadratic form given in (A.5)) with the $\mathcal{NIW}$ prior.

$$\begin{aligned}
P(\mathbf{X}, \boldsymbol{\mu},\boldsymbol{\Sigma}|\mathbf{m}_0, \kappa_0, \nu_0, \boldsymbol{\Lambda}_0) &= \mathcal{N}(\mathbf{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma})\mathcal{NIW}(\boldsymbol{\mu}, \boldsymbol{\Sigma}|\mathbf{m}_0, \kappa_0, \nu_0, \boldsymbol{\Lambda}_0) \\
&= \frac{1}{(2\pi)^{nd/2}|\boldsymbol{\Sigma}|^{n/2}} \exp\left(-\frac{n}{2}(\boldsymbol{\mu} - \bar{\mathbf{x}})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - \bar{\mathbf{x}}) - \frac{1}{2}\mathrm{tr}(\boldsymbol{\Sigma}^{-1}\mathbf{S}_{\bar{\mathbf{x}}})\right) \\
&\times \frac{|\boldsymbol{\Sigma}|^{-(\nu_0+d+2)/2}}{2^{\nu_0 d/2}\Gamma_d(\nu_0/2)(2\pi/\kappa_0)^{d/2}|\boldsymbol{\Lambda}_0|^{-\nu_0/2}} \\
&\times \exp\left(-\frac{\kappa_0}{2}(\boldsymbol{\mu} - \mathbf{m}_0)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - \mathbf{m}_0) - \frac{1}{2}\mathrm{tr}(\boldsymbol{\Sigma}^{-1}\boldsymbol{\Lambda}_0)\right)
\end{aligned} \tag{A.24}$$

After gathering similar terms and expressing the $\mathcal{NIW}$ normalization constant as

$$Z_{\mathcal{NIW}}(d, \kappa_0, \nu_0, \boldsymbol{\Lambda}_0) = 2^{\nu_0 d/2}\Gamma_d(\nu_0/2)(2\pi/\kappa_0)^{d/2}|\boldsymbol{\Lambda}_0|^{-\nu_0/2} \tag{A.25}$$

one obtains:

$$\begin{aligned}
P(\mathbf{X}, \boldsymbol{\mu},\boldsymbol{\Sigma}|\mathbf{m}_0, \kappa_0, \nu_0, \boldsymbol{\Lambda}_0) &= \frac{(2\pi)^{-nd/2}|\boldsymbol{\Sigma}|^{-(\nu_0+n+d+2)/2}}{Z_{\mathcal{NIW}}(d, \kappa_0, \nu_0, \boldsymbol{\Lambda}_0)} \\
&\times \exp\left(-\frac{n}{2}(\boldsymbol{\mu} - \bar{\mathbf{x}})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - \bar{\mathbf{x}}) - \frac{\kappa_0}{2}(\boldsymbol{\mu} - \mathbf{m}_0)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - \mathbf{m}_0)\right) \\
&\times \exp\left(-\frac{1}{2}\mathrm{tr}(\boldsymbol{\Sigma}^{-1}\mathbf{S}_{\bar{\mathbf{x}}}) - \frac{1}{2}\mathrm{tr}(\boldsymbol{\Sigma}^{-1}\boldsymbol{\Lambda}_0)\right)
\end{aligned} \tag{A.26}$$

Both exponents can be brought into a form where the posterior $\mathcal{NIW}$ hyperparameters emerge explicitly:

$$
\begin{aligned}
&P(\mathbf{X}, \boldsymbol{\mu}, \boldsymbol{\Sigma} | \mathbf{m}_0, \kappa_0, \nu_0, \boldsymbol{\Lambda}_0) \\
&\quad = \frac{(2\pi)^{-nd/2} |\boldsymbol{\Sigma}|^{-(\nu_0+n+d+2)/2}}{Z_{\mathcal{NIW}}(d, \kappa_0, \nu_0, \boldsymbol{\Lambda}_0)} \\
&\quad \times \exp\left[-\frac{\kappa_0 + n}{2} \left(\boldsymbol{\mu} - \frac{\kappa_0 \mathbf{m}_0 + n\bar{\mathbf{x}}}{\kappa_0 + n} \right)^T \boldsymbol{\Sigma}^{-1} \left(\boldsymbol{\mu} - \frac{\kappa_0 \mathbf{m}_0 + n\bar{\mathbf{x}}}{\kappa_0 + n} \right) \right] \\
&\quad \times \exp\left\{ -\frac{1}{2} \mathrm{tr}\left[\boldsymbol{\Sigma}^{-1} \left(\boldsymbol{\Lambda}_0 + \mathbf{S}_{\bar{\mathbf{x}}} + \frac{\kappa_0 n}{\kappa_0 + n} (\bar{\mathbf{x}} - \mathbf{m}_0)(\bar{\mathbf{x}} - \mathbf{m}_0)^T \right) \right] \right\} \\
&\quad = \frac{(2\pi)^{-nd/2}}{Z_{\mathcal{NIW}}(d, \kappa_0, \nu_0, \boldsymbol{\Lambda}_0)} |\boldsymbol{\Sigma}|^{-(\nu_n+d+2)/2} \\
&\quad \times \exp\left(-\frac{\kappa_n}{2} (\boldsymbol{\mu} - \mathbf{m}_n)^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu} - \mathbf{m}_n) - \frac{1}{2} \mathrm{tr}(\boldsymbol{\Sigma}^{-1} \boldsymbol{\Lambda}_n) \right)
\end{aligned}
\tag{A.27}
$$

A.8 Marginal likelihood

Marginal likelihood of the data is obtained by taking the full joint distribution and integrating over the parameters. The end result can be expressed in terms of $\mathcal{NIW}$ normalization constants and will be used in the next section to derive the formula for efficient computation of the posterior predictive probability of a set of several points.

$$
P(\mathbf{X} | \mathbf{m}_0, \kappa_0, \nu_0, \boldsymbol{\Lambda}_0) = \int_{\boldsymbol{\mu}} \int_{\boldsymbol{\Sigma}} P(\mathbf{X}, \boldsymbol{\mu}, \boldsymbol{\Sigma} | \mathbf{m}_0, \kappa_0, \nu_0, \boldsymbol{\Lambda}_0) d\boldsymbol{\mu} d\boldsymbol{\Sigma} \tag{A.28}
$$

$$
= \frac{(2\pi)^{-nd/2}}{Z_{\mathcal{NIW}}(d, \kappa_0, \nu_0, \boldsymbol{\Lambda}_0)} \int_{\boldsymbol{\mu}} \int_{\boldsymbol{\Sigma}} |\boldsymbol{\Sigma}|^{-(\nu_n+d+2)/2} \tag{A.29}
$$

$$
\times \exp\left(-\frac{\kappa_n}{2} (\boldsymbol{\mu} - \mathbf{m}_n)^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu} - \mathbf{m}_n) - \frac{1}{2} \mathrm{tr}(\boldsymbol{\Sigma}^{-1} \boldsymbol{\Lambda}_n) \right) d\boldsymbol{\mu} d\boldsymbol{\Sigma} \tag{A.30}
$$

Inside the integral there is an unnormalized posterior $\mathcal{NIW}$ and so the integration result is just the respective normalization constant:

$$
P(\mathbf{X} | \mathbf{m}_0, \kappa_0, \nu_0, \boldsymbol{\Lambda}_0) = (2\pi)^{-nd/2} \frac{Z_{\mathcal{NIW}}(d, \kappa_n, \nu_n, \boldsymbol{\Lambda}_n)}{Z_{\mathcal{NIW}}(d, \kappa_0, \nu_0, \boldsymbol{\Lambda}_0)} \tag{A.31}
$$

A.9 Posterior predictive from marginal likelihood

Now we can use (A.31) to derive the second formula for the posterior predictive probability. It is computationally more efficient than (A.21) when

computing it for a set of points and so in our implementation we use this formula.

$$\begin{aligned} P(\mathbf{x}|\mathbf{X};\mathbf{m}_0,\kappa_0,\nu_0,\mathbf{\Lambda}_0) &= \frac{P(\mathbf{x},\mathbf{X}|\mathbf{m}_0,\kappa_0,\nu_0,\mathbf{\Lambda}_0)}{P(\mathbf{X}|\mathbf{m}_0,\kappa_0,\nu_0,\mathbf{\Lambda}_0)} \\ &= \frac{(2\pi)^{nd/2}}{(2\pi)^{(n+1)d/2}} \frac{\cancel{Z_{\mathcal{NIW}}(d,\kappa_0,\nu_0,\mathbf{\Lambda}_0)}}{Z_{\mathcal{NIW}}(d,\kappa_n,\nu_n,\mathbf{\Lambda}_n)} \frac{Z_{\mathcal{NIW}}(d,\kappa_{n+1},\nu_{n+1},\mathbf{\Lambda}_{n+1})}{\cancel{Z_{\mathcal{NIW}}(d,\kappa_0,\nu_0,\mathbf{\Lambda}_0)}} \\ &= \pi^{-\frac{d}{2}} \frac{Z_{\mathcal{NIW}}(d,\kappa_{n+1},\nu_{n+1},\mathbf{\Lambda}_{n+1})}{Z_{\mathcal{NIW}}(d,\kappa_n,\nu_n,\mathbf{\Lambda}_n)} \end{aligned} \tag{A.32}$$

More generally, the joint posterior predictive probability for k points $\{\mathbf{x}\}_k$ takes the form:

$$P(\{\mathbf{x}\}_k|\mathbf{X};\mathbf{m}_0,\kappa_0,\nu_0,\mathbf{\Lambda}_0) = \pi^{-\frac{kd}{2}} \frac{Z_{\mathcal{NIW}}(d,\kappa_{n+k},\nu_{n+k},\mathbf{\Lambda}_{n+k})}{Z_{\mathcal{NIW}}(d,\kappa_n,\nu_n,\mathbf{\Lambda}_n)}, \tag{A.33}$$

where the $n+k$ subscript denotes the posterior hyperparameters computed using points both from $\mathbf{X}$ and $\{\mathbf{x}\}_k$.

Appendix B

Publication I

Sirts, K. and Alumäe, T. (2012). A hierarchical Dirichlet process model for joint part-of-speech and morphology induction. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 407–416.

A Hierarchical Dirichlet Process Model for Joint Part-of-Speech and Morphology Induction

Kairit Sirts
Institute of Cybernetics at
Tallinn University of Technology
kairit.sirts@phon.ioc.ee

Tanel Alumäe
Institute of Cybernetics at
Tallinn University of Technology
tanel.alumae@phon.ioc.ee

Abstract

In this paper we present a fully unsupervised nonparametric Bayesian model that jointly induces POS tags and morphological segmentations. The model is essentially an infinite HMM that infers the number of states from data. Incorporating segmentation into the same model provides the morphological features to the system and eliminates the need to find them during preprocessing step. We show that learning both tasks jointly actually leads to better results than learning either task with gold standard data from the other task provided. The evaluation on multilingual data shows that the model produces state-of-the-art results on POS induction.

1 Introduction

Nonparametric Bayesian modeling has recently become very popular in natural language processing (NLP), mostly because of its ability to provide priors that are especially suitable for tasks in NLP (Teh, 2006). Using nonparametric priors enables to treat the size of the model as a random variable with its value to be induced during inference which makes its use very appealing in models that need to decide upon the number of states.

The task of unsupervised parts-of-speech (POS) tagging has been under research in numerous papers, for overview see (Christodoulopoulos et al., 2010). Most of the POS induction models use the structure of hidden Markov model (HMM) (Rabiner, 1989) that requires the knowledge about the number of hidden states (corresponding to the number of tags) in advance. According to our considerations, supplying this information is not desirable for two opposing reasons: 1) it injects into the system a piece of knowledge which in a truly unsupervised setting would be unavailable; and 2) the number of POS tags used is somewhat arbitrary anyway because there is no common consensus of what should be the true number of tags in each language and therefore it seems unreasonable to constrain the model with such a number instead of learning it from the data.

Unsupervised morphology learning is another popular task that has been extensively studied by many authors. Here we are interested in learning concatenative morphology of words, meaning the substrings of the word corresponding to morphemes that, when concatenated, will give the lexical representation of the word type. For the rest of the paper we will refer to this task as (morphological) segmentation.

Several unsupervised POS induction systems make use of morphological features (Blunsom and Cohn, 2011; Lee et al., 2010; Berg-Kirkpatrick et al., 2010; Clark, 2003; Christodoulopoulos et al., 2011) and this approach has been empirically proved to be helpful (Christodoulopoulos et al., 2010). In a similar fashion one could think that knowing POS tags could be useful for learning morphological segmentations and in this paper we will study this hypothesis.

In this paper we will build a model that combines POS induction and morphological segmentation into one learning problem. We will show that the unsupervised learning of both of these tasks in the same

2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 407–416,
Montréal, Canada, June 3-8, 2012.

model will lead to better results than learning both tasks separately with the gold standard data of the other task provided. We will also demonstrate that our model produces state-of-the-art results on POS tagging. As opposed to the compared methods, our model also induces the number of tags from data.

In the following, section 2 gives the overview of the Dirichlet Processes, section 3 describes the model setup followed by the description of inference procedures in section 4, experimental results are presented in section 5, section 6 summarizes the previous work and last section concludes the paper.

2 Background

2.1 Dirichlet Process

Let H be a distribution called base measure. Dirichlet process (DP) (Ferguson, 1973) is a probability distribution over distributions whose support is the subset of the support of H:

$$G \sim DP(\alpha, H), \tag{1}$$

where α is the concentration parameter that controls the number of values instantiated by G.

DP has no analytic form and therefore other representations must be developed for sampling. In the next section we describe Chinese Restaurant Process that enables to obtain samples from DP.

2.2 Chinese Restaurant Process

Chinese Restaurant Process (CRP) (Aldous, 1985) enables to calculate the marginal probabilities of the elements conditioned on the values given to all previously seen items and integrating over possible DP prior values.

Imagine an infinitely big Chinese restaurant with infinitely many tables with each table having capacity for infinitely many customers. In the beginning the restaurant is empty. Then customers, corresponding to data points, start entering one after another. The first customer chooses an empty table to sit at. Next customers choose a new table with probability proportional to the concentration parameter α or sit into one of the already occupied tables with probability proportional to the number of customers already sitting there. Whenever a customer chooses an empty table, he will also pick a dish from H to be served on that table. The predictive probability distribution over dishes for the i-th customer is:

$$P(x_i = \phi_k | \mathbf{x}_{-i}, \alpha, H) = \frac{n_{\phi_k} + \alpha}{i - 1 + \alpha} p_H(\phi_k), \tag{2}$$

where $\mathbf{x}_{-i}$ is the seating arrangement of customers excluding the i-th customer and n_{ϕ_k} is the number of customers eating dish ϕ_k and $p_H(\cdot)$ is the probability according to H.

2.3 Hierarchical Dirichlet Process

The notion of hierarchical Dirichlet Process (HDP) (Teh et al., 2006) can be derived by letting the base measure itself to be a draw from a DP:

$$G_0 | \alpha_0, H \sim DP(\alpha_0, H) \tag{3}$$

$$G_j | \alpha, G_0 \sim DP(\alpha, G_0) \quad j = 1 \cdots J \tag{4}$$

Under HDP, CRP becomes Chinese Restaurant Franchise (Teh et al., 2006) with several restaurants sharing the same franchise-wide menu G_0. When a customer sits at an empty table in one of the G_j-th restaurants, the event of a new customer entering the restaurant G_0 will be triggered. Analogously, when a table becomes empty in one of the G_j-th restaurants, it causes one of the customers leaving from restaurant G_0.

3 Model

We consider the problem of unsupervised learning of POS tags and morphological segmentations in a joint model. Similarly to some recent successful attempts (Lee et al., 2010; Christodoulopoulos et al., 2011; Blunsom and Cohn, 2011), our model is type-based, arranging word types into hard clusters. Unlike many recent POS tagging models, our model does not assume any prior information about the number of POS tags. We will define the model as a generative sequence model using the HMM structure. Graphical depiction of the model is given in Figure 1.

3.1 Generative story

We assume the presence of a fixed length vocabulary W. The process starts with generating the lexicon that stores for each word type its POS tag and morphological segmentation.

- Draw a unigram tag distribution from the respective DP;
- Draw a segment distribution from the respective DP;
- For each tag, draw a tag-specific segment distribution from HDP with the segment distribution as base measure;
- For each word type, draw a tag from the unigram tag distribution;
- For each word type, draw a segmentation from the respective tag-specific segment distribution.

Next we proceed to generate the HMM parameters:

- For each tag, draw a bigram distribution from HDP with the unigram tag distribution as base measure;
- For each tag bigram, draw a trigram distribution from HDP with the respective bigram distribution as base measure;
- For each tag, draw a Dirichlet concentration parameter from Gamma distribution and an emission distribution from the symmetric Dirichlet.

Finally the standard HMM procedure for generating the data sequence follows. At each time step:

- Generate the next tag conditioned on the last two tags from the respective trigram HDP;
- Generate the word from the respective emission distribution conditioned on the tag just drawn;
- Generate the segmentation of the word deterministically by looking it up from the lexicon.

3.2 Model setup

The trigram transition hierarchy is a HDP:

$$G^U \sim DP(\alpha^U, H) \tag{5}$$

$$G^B_j \sim DP(\alpha^B, G^U) \quad j = 1 \cdots \infty \tag{6}$$

$$G^T_{jk} \sim DP(\alpha^T, G^B_j) \quad j, k = 1 \cdots \infty, \tag{7}$$

where G^U, G^B and G^T denote the unigram, bigram and trigram context DP-s respectively, α-s are the respective concentration parameters coupled for DP-s of the same hierarchy level. Emission parameters are drawn from multinomials with symmetric Dirichlet priors:

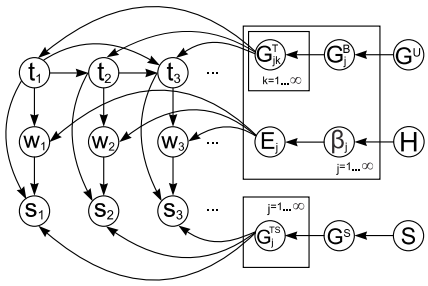

Figure 1: Plate diagram representation of the model. t_i-s, w_i-s and s_i-s denote the tags, words and segmentations respectively. G-s are various DP-s in the model, E_j-s and β_j-s are the tag-specific emission distributions and their respective Dirichlet prior parameters. H is Gamma base distribution. S is the base distribution over segments. Coupled DP concetrations parameters have been omitted for clarity.

$$E_j|\beta_j, H \sim \int Mult(\theta) Dir(\beta_j) d\theta \quad j = 1 \cdots \infty, \tag{8}$$

where each emission distribution has its own Dirichlet concentration parameter β_j drawn from H.

Morphological segments are modelled with another HDP where the groups are formed on the basis of tags:

$$G^S \sim DP(\alpha^S, S) \tag{9}$$

$$G^{TS}_j \sim DP(\alpha^{TS}, G^S) \quad j = 1 \cdots \infty, \tag{10}$$

where G^{TS}_j are the tag-specific segment DP-s and G^S is their common base distribution with S as base measure over all possible strings. S consists of two components: a geometric distribution over the segment lengths and collapsed Dirichlet-multinomial over character unigrams.

4 Inference

We implemented Gibbs sampler to draw new values for tags and Metropolis-Hastings sampler for resampling segmentations. We use a type-based col-

lapsed sampler that draws the tagging and segmentation values for all tokens of a word type in one step and integrates out the random DP measures by using the CRP representation. The whole procedure alternates between three sampling steps:

- Sampling new tag value for each word type;
- Resampling the segmentation for each type;
- Sampling new values for all parameters.

4.1 Tag sampling

The tags will be sampled from the posterior:

$$P(\mathbf{T}|\mathbf{W}, \mathbf{S}, \mathbf{w}, \mathbf{\Theta}), \tag{11}$$

where $\mathbf{W}$ is the set of words in the vocabulary, $\mathbf{T}$ and $\mathbf{S}$ are tags and segmentations assigned to each word type, $\mathbf{w}$ is the actual word sequence, and $\mathbf{\Theta}$ denotes the set of all parameters relevant for tag sampling. For brevity, we will omit $\mathbf{\Theta}$ notation in the formulas below. For a single word type, this posterior can be factored as follows:

$$\begin{aligned} P(T_i = t|\mathbf{T}_{-i}, \mathbf{S}, \mathbf{W}, \mathbf{w}) \sim \\ P(S_i|T_i = t, \mathbf{T}_{-i}, \mathbf{S}_{-i}) \times \\ P(W_i|T_i = t, \mathbf{T}_{-i}, \mathbf{W}_{-i}) \times \\ P(\mathbf{w}|T_i = t, \mathbf{T}_{-i}, \mathbf{W}), \end{aligned} \tag{12}$$

where $-i$ in the subscript denotes the observations with the i-th word type excluded.

The first term is the segmentation likelihood and can be computed according to the CRP formula:

$$P(S_i|T_i = t, \mathbf{T}_{-i}, \mathbf{S}_{-i}) = \prod_{j=1}^{|W_i|} \prod_{s \in S_i} \left(\frac{n_{ts}^{-S_i}}{n_{t\cdot}^{-S_i} + \alpha} + \frac{\alpha(m_s^{-S_i} + \beta P_0(s))}{(n_{t\cdot}^{-S_i} + \alpha)(m_{\cdot}^{-S_i} + \beta)} \right), \tag{13}$$

where the outer product is over the word type count, n_{ts} and m_s denote the number of customers "eating" the segment s under tag t and the number of tables "serving" the segment s across all restaurants respectively, dot represents the marginal counts and α and β are the concentration parameters of the respective DP-s. $-S_i$ in upper index means that the segments belonging to the segmentation of the i-th word type and not calculated into likelihood term yet have been excluded.

The word type likelihood is calculated according to the collapsed Dirichlet-multinomial likelihood formula:

$$P(W_i|T_i = t, \mathbf{T}_{-i}, \mathbf{W}_{-i}, \mathbf{w}) = \prod_{j=0}^{|W_i|-1} \frac{n_{tW_i} + j + \alpha}{n_{t\cdot} + j + \alpha N} \tag{14}$$

where n_{tW_i} is the number of times the word W_i has been tagged with tag t so far, $n_{t\cdot}$ is the number of total word tokens tagged with the tag t and N is the total number of words in the vocabulary.

The last factor is the word sequence likelihood and covers the transition probabilities. Relevant trigrams are those three containing the current word, and in all contexts where the word token appears in:

$$\begin{aligned} P(\mathbf{w}|T_i = t, \mathbf{T}_{-i}, \mathbf{W}) \sim \\ \prod_{c \in C_{W_i}} P(t|t(c_{-2}), t(c_{-1})) \cdot \\ P(t(c_{+1})|t(c_{-1}), t) \cdot \\ P(t(c_{+2})|t, t(c_{+1})) \end{aligned} \tag{15}$$

where C_{W_i} denotes all the contexts where the word type W_i appears in, $t(c)$ are the tags assigned to the context words. All these terms can be calculated with CRP formulas.

4.2 Segmentation sampling

We sample the whole segmentation of a word type as a block with forward-filtering backward-sampling scheme as described in (Mochihashi et al., 2009).

As we cannot sample from the exact marginal conditional distribution due to the dependencies between segments induced by the CRP, we use the Metropolis-Hastings sampler that draws a new proposal with forward-filtering backward-sampling scheme and accepts it with probability $min(1, \frac{P(S_{prop})}{P(S_{old})})$, where S_{prop} is the proposed segmentation and S_{old} is the current segmentation of a word type. The acceptance rate during experiments varied between 94-98%.

For each word type, we build a forward filtering table where we maintain the forward variables $\alpha[t][k]$ that present the probabilities of the last k characters of a t-character string constituting a segment. Define:

$$\alpha[0][0] = 1 \tag{16}$$

$$\alpha[t][0] = 0, \quad t > 0 \tag{17}$$

Then the forward variables can be computed recursively by using dynamic programming algorithm:

$$\alpha[t][k] = p(c_{t-k}^{t}) \sum_{j=0}^{t-k} \alpha[t-k][j], \quad t = 1 \cdots L, \tag{18}$$

where c_m^n denotes the characters $c_m \cdots c_n$ of a string c and L is the length of the word.

Sampling starts from the end of the word because it is known for certain that the word end coincides with the end of a segment. We sample the beginning position k of the last segment from the forward variables $\alpha[t][k]$, where t is the length of the word. Then we set $t = t - k$ and continue to sample the start of the previous to the last segment. This process continues until $t = 0$. The segment probabilities, conditioned on the tag currently assigned to the word type, will be calculated according to the segmentation likelihood formula (13).

4.3 Hyperparameter sampling

All DP and Dirichlet concentration parameters are given vague Gamma(10, 0.1) priors and new values are sampled by using the auxiliary variable sampling scheme described in (Escobar and West, 1995) and the extended version for HDP-s described in (Teh et al., 2006). The segment length control parameter is given uniform Beta prior and its new values are sampled from the posterior which is also a Beta distribution.

5 Results

5.1 Evaluation

We test the POS induction part of the model on all languages in the Multext-East corpora (Erjavec, 2010) as well as on the free corpora from CONLL-X Shared Task[1] for Dutch, Danish, Swedish and Portuguese. The evaluation of morphological segmentations is based on the Morpho Challenge gold segmented wordlists for English, Finnish and Turkish[2]. We gathered the sentences from Europarl corpus[3] for English and Finnish, and use the Turkish text data from the Morpho Challenge 2009[4]. Estonian gold standard segmentations have been obtained from the Estonian morphologically annotated corpus[5].

We report three accuracy measures for tagging: greedy one-to-one mapping (**1-1**) (Haghighi and Klein, 2006), many-to-one mapping (**m-1**) and V-measure (**V-m**) (Rosenberg and Hirschberg, 2007).

Segmentation is evaluated on the basis of standard F-score which is the harmonic mean of precision and recall.

5.2 Experimental results

For each experiment, we made five runs with random initializations and report the results of the median. The sampler was run 200 iterations for burnin, after which we collected 5 samples, letting the sampler to run for another 200 iterations between each two sample. We start with 15 segmenting iterations during each Gibbs iteration to enable the segmentation sampler to burnin to the current tagging state, and gradually reduce this number to one. Segmentation likelihood term for tagging is calculated on the basis of the last segment only because this setting gave the best results in preliminary experiments and it also makes the whole computation less expensive.

The first set of experiments was conducted to test the model tagging accuracy on different languages mentioned above. The results obtained were in general slightly lower than the current state-of-the-art and the number of tags learned was generally bigger than the number of gold standard tags. We observed that different components making up the corpus logarithmic probability have different magnitudes. In particular, we found that the emission probability component in log-scale is roughly four times smaller than the transition probability. This observation motivated introducing the likelihood scaling heuristic into the model to scale the emission probability up. We tried a couple of different scaling factors on Multext-East English corpus and then set its value to 4 for all languages for the rest of the experiments. This improved the tagging results consistently across all languages.

[1] http://ilk.uvt.nl/conll/free_data.html

[2] http://research.ics.tkk.fi/events/morphochallenge2010/datasets.shtml

[3] http://www.statmt.org/europarl/

[4] http://research.ics.tkk.fi/events/morphochallenge2009/datasets.shtml

[5] http://www.cl.ut.ee/korpused/morfkorpus/index.php?lang=eng

POS induction results are given in **Table 1**. When comparing these results with the recently published results on the same corpora (Christodoulopoulos et al., 2011; Blunsom and Cohn, 2011; Lee et al., 2010) we can see that our results compare favorably with the state-of-the-art, resulting with the best published results in many occasions. The number of tag clusters learned by the model corresponds surprisingly well to the number of true coarse-grained gold standard tags across all languages. There are two things to note here: 1) the tag distributions learned are influenced by the likelihood scaling heuristic and more experiments are needed in order to fully understand the characteristics and influence of this heuristic; 2) as the model is learning the coarse-grained tagset consistently in all languages, it might as well be that the POS tags are not as dependent on the morphology as we assumed, especially in inflectional languages with many derivational and inflectional suffixes, because otherwise the model should have learned a more fine-grained tagset.

Segmentation results are presented in **Table 2**. For each language, we report the lexicon-based precision, recall and F-measure, the number of word types in the corpus and and number of word types with gold segmentation available. The reported standard deviations show that the segmentations obtained are stable across different runs which is probably due to the blocked sampler. We give the segmentation results both with and without likelihood scaling heuristic and denote that while the emission likelihood scaling improves the tagging accuracy, it actually degrades the segmentation results.

It can also be seen that in general precision score is better but for Estonian recall is higher. This can be explained by the characteristics of the evaluation data sets. For English, Finnish and Turkish we use the Morpho Challenge wordlists where the gold standard segmentations are fine-grained, separating both inflectional and derivational morphemes. Especially derivational morphemes are hard to learn with pure data-driven methods with no knowledge about semantics and thus it can result in undersegmentation. On the other hand, Estonian corpus separates only inflectional morphemes which thus leads to higher recall. Some difference can also come from the fact that the sets of gold-segmented word types for other languages are much smaller than in Estonian and thus it would be interesting to see whether and how the results would change if the evaluation could be done on all word types in the corpus for other languages as well. In general, undersegmentation is more acceptable than oversegmentation, especially when the aim is to use the resulting segmentations in some NLP application.

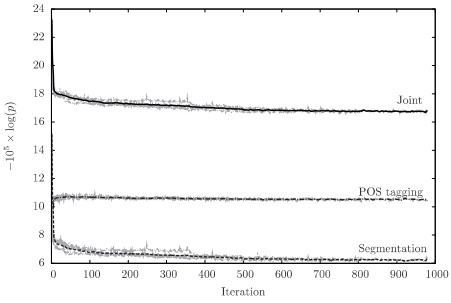

Figure 2: Log-likelihood of samples plotted against iterations. Dark lines show the average over five runs, grey lines in the back show the real samples.

Next, we studied the convergence characteristics of our model. For these experiments we made five runs with random initializations on Estonian corpus and let the sampler run up to 1100 iterations. Samples were taken after each ten iterations. **Figure 2** shows the log-likelihood of the samples plotted against iteration number. Dark lines show the averages over five runs and gray lines in the background are the likelihoods of real samples showing also the variance. We first calculated the full likelihood of the samples (the solid line) that showed a quick improvement during the first few iterations and then stabilized by continuing with only slow improvements over time. We then divided the full likelihood into two factors in order to see the contribution of both tagging and segmentation parts separately. The results are quite surprising. It turned out that the random tagging initializations are very good in terms of probability and as a matter of fact much better than the data can support and thus the tagging likelihood drops quite significantly after the first iteration and then continues with very slow improvements. The matters are totally different with segmentations where the initial random segmentations result in a low likelihood that improves heavily

	Types	1-1	m-1	V-m	Induced	True	Best Pub.		
Bulgarian	15103	**50.3 (0.9)**	**71.9 (3.8)**	54.9 (2.2)	13 (1.6)	12	-	66.5*	**55.6***
Czech	17607	**46.0 (1.0)**	60.7 (1.6)	46.2 (0.7)	12 (0.8)	12	-	**64.2***	**53.9***
Danish	17157	**53.2 (0.2)**	69.5 (0.1)	52.7 (0.4)	14 (0.0)	25	43.2†	**76.2⋆**	**59.0***
Dutch	27313	**60.5 (1.9)**	**74.0 (1.6)**	**59.1 (1.1)**	22 (0.0)	13	55.1†	71.1*	54.7*
English	9196	**67.4 (0.1)**	**79.8 (0.1)**	**66.7 (0.1**	13 (0.0)	12	-	73.3*	63.3*
Estonian	16820	**47.6 (0.9)**	**64.5 (1.9)**	45.6 (1.4)	14 (0.5)	11	-	64.4*	**53.3***
Farsi	11319	**54.9 (0.1)**	**65.3 (0.1)**	**52.1 (0.1)**	13 (0.5)	12	-	-	-
Hungarian	19191	**62.1 (0.7)**	**71.4 (0.3)**	**56.0 (0.6)**	11 (0.9)	12	-	68.2*	54.8*
Polish	19542	**48.5 (1.8)**	**59.6 (1.9)**	**45.4 (1.0)**	13 (0.8)	12	-	-	-
Portuguese	27250	45.4 (1.1)	71.3 (0.3)	55.4 (0.3)	21 (1.1)	16	**56.5†**	**78.5⋆**	**63.9***
Romanian	13822	**44.3 (0.5)**	60.5 (1.7)	46.7 (0.5)	14 (0.8)	14	-	**61.1***	**52.3***
Serbian	16813	**40.1 (0.2)**	60.1 (0.2)	43.5 (0.2)	13 (0.0)	12	-	**64.1***	**51.1***
Slovak	18793	**44.1 (1.5)**	**56.2 (0.8)**	**41.2 (0.6)**	14 (1.1)	12	-	-	-
Slovene	16420	**51.6 (1.5)**	66.8 (0.6)	51.6 (1.0)	12 (0.7)	12	-	**67.9***	**56.7***
Swedish	18473	**50.6 (0.1)**	60.3 (0.1)	55.8 (0.1)	17 (0.0)	41	38.5†	**68.7***	**58.9***

Table 1: Tagging results for different languages. For each language we report median one-to-one (1-1), many-to-one (m-1) and V-measure (V-m) together with standard deviation from five runs where median is taken over V-measure. **Types** is the number of word types in each corpus, **True** is the number of gold tags and **Induced** reports the median number of tags induced by the model together with standard deviation. **Best Pub.** lists the best published results so far (also 1-1, m-1 and V-m) in (Christodoulopoulos et al., 2011)*, (Blunsom and Cohn, 2011)⋆ and (Lee et al., 2010)†.

		Precision	Recall	F1	Types	Segmented
Estonian	without LLS	43.5 (0.8)	59.4 (0.6)	50.3 (0.7)	16820	16820
	with LLS	42.8 (1.1)	54.6 (0.7)	48.0 (0.9)		
English	without LLS	69.0 (1.3)	37.3 (1.5)	48.5 (1.1)	20628	399
	with LLS	59.8 (1.8)	29.0 (1.0)	39.1 (1.3)		
Finnish	without LLS	56.2 (2.5)	29.5 (1.7)	38.7 (2.0)	25364	292
	with LLS	56.0 (1.1)	28.0 (0.6)	37.4 (0.7)		
Turkish	without LLS	65.4 (1.8)	44.8 (1.8)	53.2 (1.7)	18459	293
	with LLS	68.9 (0.8)	39.2 (1.0)	50.0 (0.6)		

Table 2: Segmentation results on different languages. Results are calculated based on word types. For each language we report precision, recall and F1 measure, number of word types in the corpus and number of word types with gold standard segmentation available. For each language we report the segmentation result without and with emission likelihood scaling (without LLS and with LLS respectively).

with the first few iterations and then stabilizes but still continues to improve over time. The explanation for this kind of model behaviour needs further studies and we leave it for future work.

Figure 3 plots the V-measure against the tagging factor of the log-likelihood for all samples. It can be seen that the lower V-measure values are more spread out in terms of likelihood. These points correspond to the early samples of the runs. The samples taken later during the runs are on the right in the figure and the positive correlation between the V-measure and likelihood values can be seen.

Next we studied whether the morphological seg-

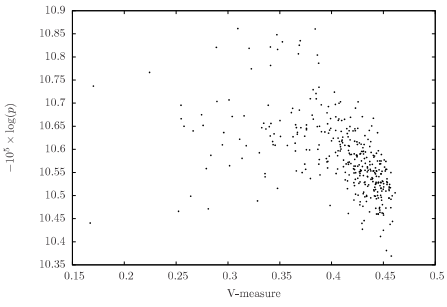

Figure 3: Tagging part of log-likelihood plotted against V-measure

	1-to-1	m-to-1	V-m
Fixed seg	40.5 (1.5)	53.4 (1.0)	37.5 (1.3)
Learned seg	47.6 (0.4)	64.5 (1.9)	45.6 (1.4)
	Precision	**Recall**	**F1**
Fixed tag	36.7 (0.3)	56.4 (0.2)	44.5 (0.3)
Learned tag	42.8 (1.1)	54.6 (0.7)	48.0 (0.9)
Morfessor	51.29	52.59	51.94

Table 3: Tagging and segmentation results on Estonian Multext-East corpus (Learned seg and Learned tag) compared to the semisupervised setting where segmentations are fixed to gold standard (Fixed seg) and tags are fixed to gold standard (Fixed tag). Finally the segmentatation results from Morfessor system for comparison are presented.

mentations and POS tags help each other in the learning process. For that we conducted two semisupervised experiments on Estonian corpus. First we provided gold standard segmentations to the model and let it only learn the tags. Then, we gave the model gold standard POS tags and only learned the segmentations. The results are given in **Table 3**. We also added the results from joint unsupervised learning for easier comparison. Unfortunately we cannot repeat this experiment on other languages to see whether the results are stable across different languages because to our knowledge there is no other free corpus with both gold standard POS tags and morphological segmentations available.

From the results it can be seen that the unsupervised learning results for both tagging and segmentation are better than the results obtained from semisupervised learning. This is surprising because one would assume that providing gold standard data would lead to better results. On the other hand, these results are encouraging, showing that learning two dependent tasks in a joint model by unsupervised manner can be as good or even better than learning the same tasks separately and providing the gold standard data as features.

Finally, we learned the morphological segmentations with the state-of-the-art morphology induction system Morfessor baseline[6] (Creutz and Lagus, 2005) and report the best results in the last row of **Table 3**. Apparently, our joint model cannot beat Morfessor in morphological segmentation and when using the emission likelihood scaling that influences the tagging results favorably, the segmentation results get even worse. Altough the semisupervised experiments showed that there are dependencies between tags and segmentations, the conducted experiments do not reveal of how to use these dependencies for helping the POS tags to learn better morphological segmentations.

[6] http://www.cis.hut.fi/projects/morpho/

6 Related Work

We will review some of the recent works related to Bayesian POS induction and morphological segmentation.

One of the first Bayesian POS taggers is described in (Goldwater and Griffiths, 2007). The model presented is a classical HMM with multinomial transition and emission distributions with Dirichlet priors. Inference is done using a collapsed Gibbs sampler and concentration parameter values are learned during inference. The model is token-based, allowing different words of the same type in different locations to have a different tag. This model can actually be classified as semi-supervised as it assumes the presence of a tagging dictionary that contains the list of possible POS tags for each word type - an assumption that is clearly not realistic in an unsupervised setting.

Models presented in (Christodoulopoulos et al., 2011) and (Lee et al., 2010) are also built on Dirichlet-multinomials and, rather than defining a sequence model, present a clustering model based on features. Both report good results on type basis and use (among others) also morphological features, with (Lee et al., 2010) making use of fixed length suffixes and (Christodoulopoulos et al., 2011) using the suffixes obtained from an unsupervised morphology induction system.

Nonparametric Bayesian POS induction has been studied in (Blunsom and Cohn, 2011) and (Gael et al., 2009). The model in (Blunsom and Cohn, 2011) uses Pitman-Yor Process (PYP) prior but the model itself is finite in the sense that the size of the tagset is fixed. Their model also captures morphological regularities by modeling the generation of words with character n-grams. The model in (Gael et al., 2009) uses infinite state space with Dirichlet Process prior. The model structure is classical HMM consisting

only of transitions and emissions and containing no morphological features. Inference is done by using beam sampler introduced in (Gael et al., 2008) which enables parallelized implementation.

One close model for morphology stems from Bayesian word segmentation (Goldwater et al., 2009) where the task is to induce word borders from transcribed sentences. Our segmentation model is in principle the same as the unigram word segmentation model and the main difference is that we are using blocked sampler while (Goldwater et al., 2009) uses point-wise Gibbs sampler by drawing the presence or absence of the word border between every two characters.

In (Goldwater et al., 2006) the morphology is learned in the adaptor grammar framework (Johnson et al., 2006) by using a PYP adaptor. PYP adaptor caches the numbers of observed derivation trees and forces the distribution over all possible trees to take the shape of power law. In the PYP (and also DP) case the adaptor grammar can be interpreted as PYP (or DP) model with regular PCFG distribution as base measure.

The model proposed in (Goldwater et al., 2006) makes several assumptions that we do not: 1) segmentations have a fixed structure of stem and suffix; and 2) there is a fixed number of inflectional classes. Inference is performed with Gibbs sampler by sampling for each word its stem, suffix and inflectional class.

7 Conclusion

In this paper we presented a joint unsupervised model for learning POS tags and morphological segmentations with hierarchical Dirichlet Process model. Our model induces the number of POS clusters from data and does not contain any hand-tuned parameters. We tested the model on many languages and showed that by introcing a likelihood scaling heuristic it produces state-of-the-art POS induction results. We believe that the tagging results could further be improved by adding additional features concerning punctuation, capitalization etc. which are heavily used in the other state-of-the-art POS induction systems but these features were intentionally left out in the current model for enabling to test the concept of joint modelling of two dependent tasks.

We found some evidence that the tasks of POS induction and morphological segmentation are dependent by conducting semisupervised experiments where we gave the model gold standard tags and segmentations in turn and let it learn only segmentations or tags respectively and found that the results in fully unsupervised setting are better. Despite of that, the model failed to learn as good segmentations as the state-of-the-art morphological segmentation model Morfessor. One way to improve the segmentation results could be to use segment bigrams instead of unigrams.

The model can serve as a basis for several further extensions. For example, one possibility would be to expand it into multilingual setting in a fashion of (Naseem et al., 2009), or it could be extended to add the joint learning of morphological paradigms of the words given their tags and segmentations in a manner described by (Dreyer and Eisner, 2011).

Acknowledgments

We would like to thank the anonymous reviewers who helped to improve the quality of this paper. This research was supported by the Estonian Ministry of Education and Research target-financed research theme no. 0140007s12, and by European Social Funds Doctoral Studies and Internationalisation Programme DoRa.

References

D. Aldous. 1985. Exchangeability and related topics. In *École d'été de Probabilités de Saint-Flour, XIII—1983*, pages 1–198. Springer.

Taylor Berg-Kirkpatrick, Alexandre Bouchard-Côté, John DeNero, and Dan Klein. 2010. Painless unsupervised learning with features. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 582–590.

Phil Blunsom and Trevor Cohn. 2011. A hierarchical Pitman-Yor process HMM for unsupervised Part of Speech induction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, pages 865–874.

Christos Christodoulopoulos, Sharon Goldwater, and Mark Steedman. 2010. Two decades of unsupervised POS induction: How far have we come? In *Proceed-*

ings of the 2010 Conference on Empirical Methods in Natural Language Processing, pages 575–584.

Christos Christodoulopoulos, Sharo Goldwater, and Mark Steedman. 2011. A Bayesian mixture model for PoS induction using multiple features. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 638–647, Edinburgh, Scotland, UK.

Alexander Clark. 2003. Combining distributional and morphological information for Part of Speech induction. In *Proceedings of the Tenth Conference on European Chapter of the Association for Computational Linguistics - Volume 1*, pages 59–66.

Mathias Creutz and Krista Lagus. 2005. Inducing the morphological lexicon of a natural language from unannotated text. In *In Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning*, pages 106–113.

Markus Dreyer and Jason Eisner. 2011. Discovering morphological paradigms from plain text using a Dirichlet Process mixture model. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 616–627.

Toma Erjavec. 2010. MULTEXT-East version 4: Multilingual morphosyntactic specifications, lexicons and corpora. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation*.

Michael D. Escobar and Mike West. 1995. Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, 90(430).

Thomas S. Ferguson. 1973. A Bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1(2):209–230.

Jurgen Van Gael, Yunus Saatci, Yee Whye Teh, and Zoubin Ghahramani. 2008. Beam sampling for the infinite Hidden Markov Model. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1088–1095.

Jurgen Van Gael, Andreas Vlachos, and Zoubin Ghahramani. 2009. The infinite HMM for unsupervised PoS tagging. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, pages 678–687.

Sharon Goldwater and Tom Griffiths. 2007. A fully Bayesian approach to unsupervised Part-of-Speech tagging. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 744–751, Prague, Czech Republic.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. 2006. Interpolating between types and tokens by estimating power-law generators. In *Advances in Neural Information Processing Systems 18*, Cambridge, MA.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. 2009. A Bayesian framework for word segmentation: Exploring the effects of context. *Cognition*, 112:21–54.

Aria Haghighi and Dan Klein. 2006. Prototype-driven learning for sequence models. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 320–327, New York City, USA.

Mark Johnson, Thomas L. Griffiths, and Sharon Goldwater. 2006. Adaptor grammars: A framework for specifying compositional nonparametric Bayesian models. In *Advances in Neural Information Processing Systems 19*, pages 641–648.

Yoong Keok Lee, Aria Haghighi, and Regina Barzilay. 2010. Simple type-level unsupervised POS tagging. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 853–861.

Daichi Mochihashi, Takeshi Yamada, and Naonori Ueda. 2009. Bayesian unsupervised word segmentation with nested Pitman-Yor language modeling. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1*, pages 100–108.

Tahira Naseem, Benjamin Snyder, Jacob Eisenstein, and Regina Barzilay. 2009. Multilingual part-of-speech tagging: Two unsupervised approaches. *Journal of Artificial Intelligence Research*, 36:1–45.

Lawrence R. Rabiner. 1989. A tutorial on Hidden Markov Models and selected applications in speech recognition. In *Proceedings of the IEEE*, pages 257–286.

Andrew Rosenberg and Julia Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 410–420, Prague, Czech Republic.

Yee Whye Teh, Michel I. Jordan, Matthew J. Beal, and David M. Blei. 2006. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.

Yee Whye Teh. 2006. A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 985–992.

Appendix C

Publication II

Sirts, K. and Goldwater, S. (2013). Minimally-supervised morphological segmentation using adaptor grammars. *Transactions of the Association for Computational Linguistics*, 1(May):231–242.

Minimally-Supervised Morphological Segmentation using Adaptor Grammars

Kairit Sirts
Institute of Cybernetics
Tallinn University of Technology
sirts@phon.ioc.ee

Sharon Goldwater
School of Informatics
The University of Edinburgh
sgwater@inf.ed.ac.uk

Abstract

This paper explores the use of Adaptor Grammars, a nonparametric Bayesian modelling framework, for minimally supervised morphological segmentation. We compare three training methods: unsupervised training, semi-supervised training, and a novel model selection method. In the model selection method, we train unsupervised Adaptor Grammars using an over-articulated *metagrammar*, then use a small labelled data set to select which potential morph boundaries identified by the metagrammar should be returned in the final output. We evaluate on five languages and show that semi-supervised training provides a boost over unsupervised training, while the model selection method yields the best average results over all languages and is competitive with state-of-the-art semi-supervised systems. Moreover, this method provides the potential to tune performance according to different evaluation metrics or downstream tasks.

1 Introduction

Research into unsupervised learning of morphology has a long history, starting with the work of Harris (1951). While early research was mostly motivated by linguistic interests, more recent work in NLP often aims to reduce data sparsity in morphologically rich languages for tasks such as automatic speech recognition, statistical machine translation, or automatic text generation. For these applications, however, completely unsupervised systems may not be ideal if even a small amount of segmented training data is available. In this paper, we explore the use of Adaptor Grammars (Johnson et al., 2007) for minimally supervised morphological segmentation.

Adaptor Grammars (AGs) are a nonparametric Bayesian modelling framework that can learn latent tree structures over an input corpus of strings. For example, they can be used to define a morphological grammar where each word consists of zero or more prefixes, a stem, and zero or more suffixes; the actual forms of these morphs (and the segmentation of words into morphs) are learned from the data. In this general approach AGs are similar to many other unsupervised morphological segmentation systems, such as Linguistica (Goldsmith, 2001) and the Morfessor family (Creutz and Lagus, 2007). A major difference, however, is that the morphological grammar is specified as an input to the program, rather than hard-coded, which allows different grammars to be explored easily. For the task of segmenting utterances into words, for example, Johnson and colleagues have experimented with grammars encoding different kinds of sub-word and super-word structure (e.g., syllables and collocations), showing that the best grammars far outperform other systems on the same corpora (Johnson, 2008a; Johnson and Goldwater, 2009; Johnson and Demuth, 2010).

These word segmentation papers demonstrated both the power of the AG approach and the synergistic behavior that occurs when learning multiple levels of structure simultaneously. However, the best-performing grammars were selected using the same corpus that was used for final testing, and each paper dealt with only one language. The ideal unsupervised learner would use a single grammar tuned on

Transactions of the Association for Computational Linguistics, 1 (2013) 255–266. Action Editor: Kristina Toutanova.
Submitted 11/2012; Published 5/2013.

one or more development languages and still perform well on other languages where development data is unavailable. Indeed, this is the basic principle behind Linguistica and Morfessor. However, we know that different languages can have very different morphological properties, so using a single grammar for all languages may not be the best approach if there is a principled way to choose between grammars. Though AGs make it easy to try many different possible grammars, the process of proposing and testing plausible options can still be time-consuming.

In this paper, we propose a novel method for automatically selecting good morphological grammars for different languages (English, Finnish, Turkish, German, and Estonian) using a small amount of gold-segmented data (1000 word types). We use the AG framework to specify a very general binary-branching grammar of depth four with which we learn a parse tree of each word that contains several possible segmentation splits for the word. Then, we use the gold-segmented data to learn, for each language, which of the proposed splits from the original grammar should actually be used in order to best segment that language.

We evaluate our approach on both a small development set and the full Morpho Challenge test set for each language—up to three million word types. In doing so, we demonstrate that using the posterior grammar of an AG model to decode unseen data is a feasible way to scale these models to large data sets. We compare to several baselines which use the annotated data to different degrees: parameter tuning, grammar tuning, supervised training, or no use of annotated data. In addition to existing approaches—unsupervised and semi-supervised Morfessor, unsupervised Morsel (Lignos, 2010), and unsupervised AGs—we also show how to use the annotated data to train semi-supervised AGs (using the data to accumulate rule statistics rather than for grammar selection). The grammar selection method yields comparable results to the best of these other approaches.

To summarize, our contributions in this paper are: 1) scaling AGs to large data sets by using the posterior grammar to define an inductive model; 2) demonstrating how to train semi-supervised AG models, and showing that this improves morphological segmentation over unsupervised training; and 3) introducing a novel grammar selection method for AG models whose segmentation results are competitive with the best existing systems.

Before providing details of our methods and results, we first briefly review Adaptor Grammars. For a formal definition, see Johnson et al. (2007).

2 Adaptor Grammars

Adaptor Grammars are a framework for specifying nonparametric Bayesian models that can be used to learn latent tree structures from a corpus of strings. There are two components to an AG model: the *base distribution*, which is just a PCFG, and the *adaptor*, which "adapts" the probabilities assigned to individual subtrees under the PCFG model, such that the probability of a subtree under the complete model may be considerably higher than the product of the probabilities of the PCFG rules required to construct it. Although in principle the adaptor can be any function that maps one distribution onto another, Johnson et al. (2007) use a Pitman-Yor Process (PYP) (Pitman and Yor, 1997) as the adaptor because it acts as a caching model. Under a PYP AG model, the posterior probability of a particular subtree will be roughly proportional to the number of times that subtree occurs in the current analysis of the data (with the probability of unseen subtrees being computed under the base PCFG distribution).

An AG model can be defined by specifying the CFG rules (the support for the base distribution) and indicating which non-terminals are "adapted", i.e., can serve as the root of a cached subtree. Given this definition and an input corpus of strings, Markov chain Monte Carlo samplers can be used to infer the posterior distribution over trees (and all hyperparameters of the model, including PCFG probabilities in the base distribution and PYP hyperparameters). Any frequently recurring substring (e.g., a common prefix) will tend to be parsed consistently, as this permits the model to treat the subtree spanning that string as a cached subtree, assigning it higher probability than under the PCFG distribution.

Adaptor Grammars have been applied to a wide variety of tasks, including segmenting utterances into words (Johnson, 2008a; Johnson and Goldwater, 2009; Johnson and Demuth, 2010), classifying documents according to perspective (Hardisty et al., 2010), machine transliteration of names (Huang et

al., 2011), native language identification (Wong et al., 2012), and named entity clustering (Elsner et al., 2009). There have also been AG experiments with morphological segmentation, but more as a proof of concept than an attempt to achieve state-of-the-art results (Johnson et al., 2007; Johnson, 2008b).

3 Using AGs for Learning Morphology

Originally, the AG framework was designed for unsupervised learning. This section first describes how AGs can be used for unsupervised morphological segmentation, and then introduces two ways to use a small labelled data set to improve performance: semi-supervised learning and grammar selection.

3.1 Unsupervised Adaptor Grammars

We define three AG models to use as unsupervised baselines in our segmentation experiments. The first of these is very simple:

$$\begin{aligned} \text{Word} &\rightarrow \underline{\text{Morph}}^{+} \\ \underline{\text{Morph}} &\rightarrow \text{Char}^{+} \end{aligned} \tag{1}$$

The underline notation indicates an adapted nonterminal, and $^{+}$ abbreviates a set of recursive rules, e.g., Word $\rightarrow \underline{\text{Morph}}^{+}$ is short for

$$\begin{aligned} \text{Word} &\rightarrow \text{Morphs} \\ \text{Morphs} &\rightarrow \underline{\text{Morph}}\ \text{Morphs} \\ \text{Morphs} &\rightarrow \underline{\text{Morph}} \end{aligned}$$

Grammar 1 (**MorphSeq**) is just a unigram model over morphs: the Morph symbol is adapted, so the probability of each Morph will be roughly proportional to its (inferred) frequency in the corpus. The grammar specifies no further structural relationships between morphs or inside of morphs (other than a geometric distribution on their length in characters).

Experiments with AGs for unsupervised word segmentation suggest that adding further latent structure can help with learning. Here, we add another layer of structure below the morphs,[1] calling the resulting grammar **SubMorphs**:

$$\begin{aligned} \text{Word} &\rightarrow \underline{\text{Morph}}^{+} \\ \underline{\text{Morph}} &\rightarrow \underline{\text{SubMorph}}^{+} \\ \underline{\text{SubMorph}} &\rightarrow \text{Char}^{+} \end{aligned} \tag{2}$$

For capturing the rules of morphotactics, a grammar with linguistically motivated non-terminals can be created. There are many plausible options and the best-performing grammar may be somewhat language-dependent. Rather than experimenting extensively, we designed our third grammar to replicate as closely as possible the grammar that is implicitly implemented in the Morfessor system. This **Compounding** grammar distinguishes between prefixes, stems and suffixes, allows compounding, defines the order in which the morphs can occur and also allows the morphs to have inner latent structure:

$$\begin{aligned} \text{Word} &\rightarrow \underline{\text{Compound}}^{+} \\ \underline{\text{Compound}} &\rightarrow \underline{\text{Prefix}}^{*}\ \underline{\text{Stem}}\ \underline{\text{Suffix}}^{*} \\ \underline{\text{Prefix}} &\rightarrow \underline{\text{SubMorph}}^{+} \\ \underline{\text{Stem}} &\rightarrow \underline{\text{SubMorph}}^{+} \\ \underline{\text{Suffix}} &\rightarrow \underline{\text{SubMorph}}^{+} \\ \underline{\text{SubMorph}} &\rightarrow \text{Char}^{+} \end{aligned} \tag{3}$$

3.2 Semi-Supervised Adaptor Grammars

The first new use of AGs we introduce is the *semi-supervised* AG, where we use the labelled data to extract counts of the different rules and subtrees used in the gold-standard analyses. We then run the MCMC sampler as usual over both the unlabelled and labelled data, treating the counts from the labelled data as fixed.

We assume that the labelled data provides a consistent bracketing (no two spans in the bracketing can partially overlap) and the labels of the spans must be compatible with the grammar. However, the bracketing may not specify all levels of structure in the grammar. In our case, we have morpheme bracketings but not, e.g., submorphs. Thus, using the SubMorphs grammar in semi-supervised mode will constrain the sampler so that Morph spans in the labelled data will remain fixed, while the SubMorphs inside those Morphs will be resampled.

[1] Because the nonterminal labels are arbitrary, this grammar can also be interpreted as adding another layer on top of morphs, allowing the model to learn morph *collocations* that encode dependencies between morphs (which themselves have no substructure). However preliminary experiments showed that the morph-submorph interpretation scored better than the collocation-morph interpretation, hence we chose the corresponding nonterminal names.

The main change made to the AG inference process[2] for implementing the semi-supervised AG was to prune out from the sampling distribution any non-terminals that are inconsistent with the spans/labels in the given labelling.

3.3 AG Select

Both the unsupervised and semi-supervised methods described above assume the definition of a grammar that adequately captures the phenomena being modelled. Although the AG framework makes it easy to experiment with different grammars, these experiments can be time-consuming and require some good guesses as to what a plausible grammar might be. These problems can be overcome by automating the grammar development process to systematically evaluate different grammars and find the best one.

We propose a minimally supervised model selection method AG Select that uses the AG framework to automatically identify the best grammar for different languages and data sets. We first define a very general binary-branching CFG grammar for AG training that we call the *metagrammar*. The metagrammar learns a parse tree for each word where each branch contains a different structure in the word. The granularity of these structures is determined by the depth of this tree. For example, Grammar 4 generates binary trees of depth two and can learn segmentations of up to four segments.

$$
\begin{array}{ll}
\text{Word} \rightarrow \underline{\text{M1}} & \\
\text{Word} \rightarrow \underline{\text{M1}}\ \underline{\text{M2}} & \underline{\text{M11}} \rightarrow \text{Chars}^{+} \\
\underline{\text{M1}} \rightarrow \underline{\text{M11}} & \underline{\text{M12}} \rightarrow \text{Chars}^{+} \\
\underline{\text{M1}} \rightarrow \underline{\text{M11}}\ \underline{\text{M12}} & \underline{\text{M21}} \rightarrow \text{Chars}^{+} \\
\underline{\text{M2}} \rightarrow \underline{\text{M21}} & \underline{\text{M22}} \rightarrow \text{Chars}^{+} \\
\underline{\text{M2}} \rightarrow \underline{\text{M21}}\ \underline{\text{M22}} &
\end{array}
\tag{4}
$$

Next we introduce the notion of a *morphological template*, which is an ordered sequence of non-terminals whose concatenated yields constitute the word and which are used to parse out a specific segmentation of the word. For example, using Grammar 4 the parse tree of the word *saltiness* is shown in Figure 1. There are four possible templates with four different segmentations: **M1 M2** (*salt_iness*), **M11 M12 M2** (*sal_t_iness*), **M1 M21 M22** (*salt_i_ness*), and **M11 M12 M21 M22** (*sal_t_i_ness*).

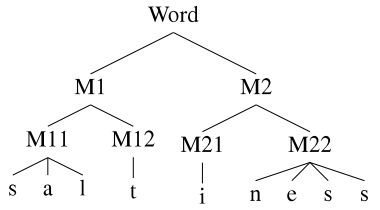

Figure 1: The parse tree generated by the metagrammar of depth 2 for the word *saltiness*.

The morphological template consisting only of non-terminals from the lowest cached level of the parse tree is expected to have high recall, whereas the template containing the non-terminals just below the Word is expected to have high precision. Our goal is to find the optimal template by using a small labelled data set. The grammar selection process iterates over the set of all templates. For each template, the segmentations of the words in the labelled data set are parsed out and the value of the desired evaluation metric is computed. The template that obtained the highest score is then chosen.

For each language we use a single template to segment all the words in that language. However, even using (say) a four-morph template such as **M11 M12 M21 M22**, some words may contain fewer morphs because the metagrammar permits either unary or binary branching rules, so some parses may not contain **M12** or **M2** (and thus **M21 M22**) spans. Thus, we can represent segmentations of different lengths (from 1 to 2^n, where n is the depth of the metagrammar) with a single template.[3]

For our experiments we use a metagrammar of depth four. This grammar allows words to consist of up to 16 segments, which we felt would be enough for any word in the training data. Also, iterating over all the templates of a grammar with bigger depth would not be feasible as the number of different templates increases very rapidly.[4]

[2]We started with Mark Johnson's PYAG implementation, http://web.science.mq.edu.au/~mjohnson/code/py-cfg.tgz, which we also used for our unsupervised and grammar selection experiments.

[3]We also experimented with selecting different templates for words of different length but observed no improvements over the single template approach.

[4]The number of templates of each depth can be expressed recursively as $N_i = (N_{i-1} + 1)^2$, where N_{i-1} is the number of

3.4 Inductive Learning

Previous work on AGs has used relatively small data sets and run the sampler on the entire input corpus (some or all of which is also used for evaluation)—a *transductive* learning scenario. However, our larger data sets contain millions of word types, where sampling over the whole set is not feasible. For example, 1000 training iterations on 50k word types took about a week on one 2.67 GHz CPU. To solve this problem, we need an *inductive* learner that can be trained on a small set of data and then used to segment a different larger set.

To create such a learner, we run the sampler on up to 50k word types, and then extract the *posterior grammar* as a PCFG.[5] This grammar contains all the initial CFG rules, plus rules to generate each of the cached subtrees inferred by the sampler. The sampler counts of all rules are normalized to obtain a PCFG, and we can then use a standard CKY parser to decode the remaining data using this PCFG.

4 Experiments

4.1 Data

We test on languages with a range of morphological complexity: English, Finnish, Turkish, German, and Estonian. For each language we use two small sets of gold-annotated data—a *labelled set* for semi-supervised training or model selection and a *dev set* for development results—and one larger gold-annotated dataset for final tests. We also have a large *unlabelled* training set for each language. Table 1 gives statistics.

The data sets for English, Finnish, Turkish and German are from the Morpho Challenge 2010 competition[6] (MC2010). We use the MC2010 training set of 1000 annotated word types as our labelled data, and for our dev sets we collate together the development data from all years of the MC competition. Final evaluation is done on the official MC2010 test sets, which are not public, so we rely on the MC organizers to perform the evaluation. The words in each test set are an unknown subset of the words in the unlabelled corpus, so to evaluate we segmented the entire unlabelled corpus and sent the results to the MC team, who then computed scores on the test words.

	Unlab.	Lab.	Dev	Test
English	0.9M	1000	1212	16K
Finnish	2.9M	1000	1494	225K
Turkish	0.6M	1000	1531	64K
German	2.3M	1000	785	62K
Estonian	2.1M	1000	1500	74K

Table 1: Number of word types in our data sets.

The Estonian wordlist is gathered from the newspaper texts of a mixed corpus of Estonian.[7] Gold standard segmentations of some of these words are available from the Estonian morphologically disambiguated corpus;[8] we used these for the test set, with small subsets selected randomly for the labelled and dev sets.

For semi-supervised tests of the AG Compounding grammar we annotated the morphemes in the English, Finnish and Estonian labelled sets as prefixes, stems or suffixes. We could not do so for Turkish because none of the authors knows Turkish.

4.2 Evaluation

We evaluate our results with two measures: segment border F1-score (SBF1) and EMMA (Spiegler and Monson, 2010). SBF1 is one of the simplest and most popular evaluation metrics for morphological segmentations. It computes F1-score from the precision and recall of ambiguous segment boundaries—i.e., word edges are not counted. It is easy and quick to compute but has the drawback that it gives no credit for one-morpheme words that have been segmented correctly (i.e., are assigned no segment borders). Also it can only be used on systems and gold standards where the output is just a segmentation of the surface string (e.g., *availabil+ity*) rather than a morpheme analysis (e.g., *available+ity*). For this reason we cannot report SBF1 on our German data, which annotations contain only analyses.

EMMA is a newer measure that addresses both

templates in the grammar of depth one less and $N_0 = 0$.

[5] This can be seen as a form of Structure Compilation (Liang et al., 2008), where the solution found by a more costly model is used to define a less costly model. However in Liang et al.'s case both models were already inductive.

[6] http://research.ics.aalto.fi/events/morphochallenge2010/datasets.shtml

[7] http://www.cl.ut.ee/korpused/segakorpus/epl

[8] http://www.cl.ut.ee/korpused/morfkorpus/

of these issues—correctly segmented one-morpheme words are reflected in the score, and it can evaluate both concatenative and non-concatenative morphology. EMMA works by finding the best one-to-one mapping between the hypothesized and true segments. The induced segments are then replaced with their mappings and based on that, F1-score on matching segments is calculated. Using EMMA we can evaluate the induced segmentations of German words against gold standard analyses. EMMA has a freely available implementation,[9] but is slow to compute because it uses Integer Linear Programming.

For our dev results, we computed both scores using the entire dev set, but for the large test sets, the evaluation is done on batches of 1000 word types selected randomly from the test set. This procedure is repeated 10 times and the average is reported, just as in the MC2010 competition (Kohonen et al., 2010a).

4.3 Baseline Models

We compare our AG models to several other morphology learning systems. We were able to obtain implementations of two of the best unsupervised systems from MC2010, Morfessor (Creutz and Lagus, 2007) and Morsel (Lignos, 2010), and we use these for comparisons on both the dev and test sets. We also report test results from MC2010 for the only semi-supervised system in the competition, semi-supervised Morfessor (Kohonen et al., 2010a; Kohonen et al., 2010b). No dev results are reported on this system since we were unable to obtain an implementation. This section briefly reviews the systems.

4.3.1 Morfessor Categories-MAP

Morfessor Categories-MAP (Morfessor) is a state-of-the-art unsupervised morphology learning system. Its implementation is freely available[10] so it is widely used both as a preprocessing step in tasks requiring morphological segmentations, and as a baseline for evaluating morphology learning systems.

Morfessor uses the Minimum Description Length (MDL) principle to choose the optimal segment lexicon and the corpus segmentation. Each morph in the segment lexicon is labelled as a stem, prefix, suffix or non-morph. The morphotactic rules are encoded as an HMM, which specifies the allowed morph sequences with respect to the labels (e.g., a suffix cannot directly follow a prefix).

The morphs in the segment lexicon can have a hierachical structure, containing submorphs which themselves can consist of submorphs etc. We hypothesize that this hierarchical structure is one of the key reasons why Morfessor has been so successful, as the experiments also in this paper with different grammars show that the ability to learn latent structures is crucial for learning good segmentations.

One essential difference between Morfessor and the proposed AG Select is that while we use the labelled data to choose which levels of the hierarchy are to be used as morphs, Morfessor makes this decision based on the labels of the segments, choosing the most fine-grained morph sequence that does not contain the non-morph label.

Morfessor includes a free parameter, perplexity threshold, which we found can affect the SBF1 score considerably (7 points or more). The best value for this parameter depends on the size of the training set, characteristics of the language being learned, and also the evaluation metric being used, as in some cases the best SBF1 and EMMA scores are obtained with completely different values.

Thus, we tuned the value of the perplexity threshold on the labelled set for each language and evaluation metric for different unlabelled training set sizes.

4.3.2 Semi-Supervised Morfessor

Recently, the Morfessor system has been adapted to allow semi-supervised training. Four versions of the system were evaluated in MC2010, using different degrees of supervision. Results reported here are from the Morfessor S+W system, which performed best of those that use the same kind of labelled data as we do.[11] This system uses the Morfessor Baseline model (not Cat-MAP), which incorporates a lexicon prior and data likelihood term. The semi-supervised version maintains separate likelihoods for the labelled and unlabelled data, and uses the development set to tune two parameters that weight these terms with respect to each other and the prior.

[9] http://www.cs.bris.ac.uk/Research/MachineLearning/Morphology/resources.jsp#eval

[10] http://www.cis.hut.fi/projects/morpho/morfessorcatmapdownloadform.shtml

[11] Morfessor S+W+L performs better, but uses training data with morpheme analyses rather than surface segmentations.

4.3.3 Morsel

Morsel (Lignos, 2010) is an unsupervised morphology learning system introduced in MC2010; we obtained the implementation from the author. Morsel learns morphological analyses rather than segmentations, so it can be evaluated only using EMMA. There are two options provided for running Morsel: aggressive and conservative. We used the development set to choose the best in each experimental case.

The MC data sets contain gold standard morphological analyses (as well as segmentations) so we could compute Morsel's EMMA scores using the analyses. However, we found that Morsel obtains higher EMMA scores when evaluated against gold standard segmentations and thus we used this option in all the experiments. (EMMA scores for other systems were also computed using the segmentations.)

4.4 Method

The experiments were conducted in two parts. First, we evaluated different aspects of the AG models and compared to all baseline models using the dev set data. Then we evaluated the most competitive models on the final test data.

For the development experiments, we compiled unlabelled training sets with sizes ranging from 10k to 50k word types (using the most frequent word types in each case). For the AG results, we report the average of five different runs made on the same training set. We let the sampler run for 1000 iterations. No annealing was used as it did not seem to help. The table label resampling option was turned on and the hyperparameter values were inferred.

We trained all AG and baseline models on each of these training sets. For AG Select, the words from the labelled data set were added to the training set to allow for template selection.[12] To compute results in transductive mode, the words from the dev set were also added to the training data. In inductive mode, the dev set was instead parsed with the CKY parser.

Preliminary experiments showed that the performance of unsupervised AG and AG Select improved with larger training sets, though the effect is small (see Figure 2 for results of AG Select in transductive mode; the trend in inductive mode is similar). Based on these and similar results with other baseline systems, all results reported later for unsupervised models (AG and baseline) and AG Select were obtained using training sets of 50k words.

In contrast to the above models, the semi-supervised AG does not always improve with more unlabelled data (see Figure 2) and in the limit, it will match the performance of the same grammar in the unsupervised setting. Other semi-supervised approaches often solve this problem by weighting the labelled data more heavily than the unlabelled data when estimating model parameters—effectively, assuming that each labelled item has actually been observed more than once. However, duplicating the labelled data does not make sense in the AG framework, because duplicate items will in most cases just be cached at the root (Word) node, providing no additional counts of Morphs (which are where the useful information is). It might be possible to come up with a different way to weight the labelled data more heavily when larger unlabelled sets are used, however for this paper we instead kept the labelled data the same and tuned the amount of unlabelled data. We used the dev set to choose the amount of unlabelled data (in the range from 10k to 50k types); results for semi-supervised AG are reported using the optimal amount of unlabelled data for each experiment.

For test set experiments with semi-supervised AG, we evaluated each language using whichever grammar performed best on that language's dev set. For test set experiments with AG Select, we chose the templates with a two-pass procedure. First, we trained 5 samplers on the 50k training set with labelled set added, and used the labelled data to choose the best template for each inferred grammar. Then, we decoded the dev set using each of the 5 grammar/template pairs and based on these results, chose the best of these pairs to decode the test set.

4.5 Results

We present the dev set results in Table 2(a) for transductive and in Table 2(b) for inductive learning. In each table, unsupervised models are shown in the upper section and the semi-supervised models and AG Select below. Morsel appears only in Table 2(a) since it only works transductively. Semi-supervised grammars cannot be trained on German, since we

[12] We also experimented with smaller sets of labelled data. In most cases, the template selected based on only 300 word types was the same than the one selected with 1000 word types.

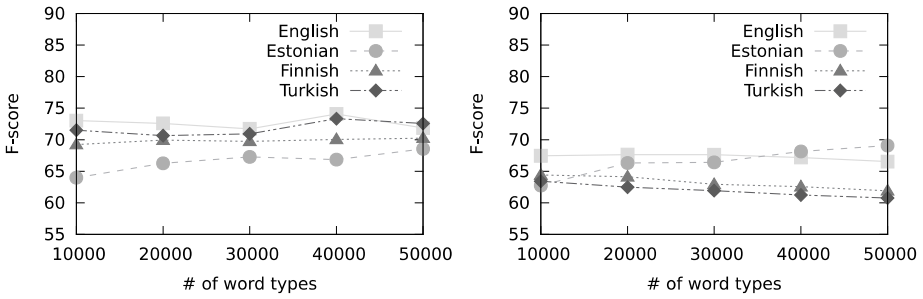

Figure 2: Effect of training data size on dev set SBF1 for AG Select (left) and semi-supervised SubMorphs grammar (right) in transductive mode.

only have gold standard analyses, not segmentations.

The SubMorphs grammar performs the best of the unsupervised AG models, with the Compounding grammar being only slightly worse. We also tried the Compounding grammar without the sub-morph structures but the results were even worse than those of MorphSeq. This shows that the latent structures are important for learning good segmentations.

In all cases, the semi-supervised AGs perform better (ofen much better) than the corresponding unsupervised grammars. Even though their average scores are not as high as AG Select's, they give the best dev set results in many cases. This shows that although for semi-supervised AG the grammar must be chosen manually, with a suitable choice of the grammar and only a small set of labelled data it can improve considerably over unsupervised AG.

In transductive mode, the AG Select performs the best in several cases. In both transductive and inductive mode, the results of AG Select are close to the best results obtained and are consistently good across all languages—it achieves the best average scores of all models, suggesting that the model selection method is robust to different types of morphology and annotation schemes.

Table 3 presents the test set results. We include scores for unsupervised Morfessor in both transductive and inductive mode, where transductive mode trains on the entire unlabelled corpus and inductive mode trains on the 50k subset. The semi-supervised Morfessor scores are taken from the MC results page[13] after verifying that the evaluation methodology and labelled data used is the same as ours.[14]

There is a good deal of variation between development and test results, indicating that the dev sets may not be a representative sample. The most notable differences are in Turkish, where all models perform far worse on the test than dev set. However, AG Select performs slightly better on the test set for the other languages. Thus its average SBF1 score actually improves on the test set and is not much worse than semi-supervised Morfessor. While its average performance drops somewhat on test set EMMA, it is still as good as any other model on that measure. Again, these results support the idea that AG Select is robust to variations in language and data set.

We also note the surprisingly good performance of Morfessor in transductive mode on Estonian; this could possibly be due to the larger amount of training data used for the test set results, but it is not clear why this would improve performance so much on Estonian and not on the other languages.

5 Discussion

To give a sense of what the AG Select model is learning, we provide some examples of both correctly and incorrectly induced segmentations in Table 4. These examples suggest that for example in English, M1 is used to model the stem, M21 is for the suffix or the second stem in the compound word, and the rest of the elements in the template are for the remaining suffixes (if any).

Table 5 presents examples of some of the most frequently used metagrammar rules and cached rules

[13] http://research.ics.aalto.fi/events/morphochallenge/

[14] Sami Virpioja, personal communication.

(a) Transductive mode	Border F1-score					EMMA					
	Eng	**Est**	**Fin**	**Tur**	**Avg**	**Eng**	**Est**	**Fin**	**Tur**	**Ger**	**Avg**
AG MorphSeq	61.5	54.0	56.9	59.5	58.0	74.7	74.1	63.7	53.5	59.4	65.1
AG SubMorphs	66.2	66.9	60.5	59.5	63.3	79.1	83.4	66.8	53.4	57.4	68.0
AG Compounding	63.0	64.8	60.9	60.9	62.4	75.4	81.6	65.5	53.7	62.2	67.7
Morfessor	69.5	55.7	65.0	69.3	64.9	**81.3**	75.3	67.8	62.2	**62.7**	69.9
Morsel	-	-	-	-	-	76.8	74.4	66.1	50.1	55.9	64.7
AG ssv MorphSeq	64.4	57.3	63.0	68.9	63.4	74.4	75.9	65.6	59.6	-	-
AG ssv SubMorphs	67.6	**69.1**	64.4	63.4	66.1	79.5	**84.4**	69.2	56.1	-	-
AG ssv Compounding	70.0	67.5	**71.8**	-	-	79.5	82.8	**74.0**	-	-	-
AG Select	**71.9**	68.5	70.2	**72.6**	**70.8**	77.5	81.8	73.2	**63.0**	62.4	**71.6**

(b) Inductive mode	Border F1-score					EMMA					
	Eng	**Est**	**Fin**	**Tur**	**Avg**	**Eng**	**Est**	**Fin**	**Tur**	**Ger**	**Avg**
AG MorphSeq	57.6	54.0	55.4	59.8	56.7	72.0	73.8	62.6	53.7	58.9	64.2
AG SubMorphs	66.1	67.5	61.6	59.8	63.7	78.6	**83.7**	67.4	53.4	56.0	67.8
AG Compounding	62.0	64.8	57.4	61.1	61.3	73.5	81.1	61.9	53.2	61.0	66.2
Morfessor	68.9	51.1	63.5	68.2	62.9	**80.9**	72.0	68.1	60.6	60.8	68.5
AG ssv MorphSeq	64.6	56.9	63.1	**70.3**	63.8	72.7	73.3	65.9	**61.2**	-	-
AG ssv SubMorphs	70.1	**69.7**	66.3	67.9	68.4	80.4	**83.7**	70.5	59.0	-	-
AG ssv Compounding	**70.5**	67.2	**70.0**	-	-	77.3	81.9	70.5	-	-	-
AG Select	69.8	68.8	67.5	70.1	**69.1**	77.3	81.9	**71.1**	59.7	**62.6**	**70.5**

Table 2: Dev set results for all models in (a) transductive and (b) inductive mode. Unsupervised AG models and baselines are shown in the top part of each table; semi-supervised AG models and grammar selection method are below.

	Border F1-score						EMMA						
	Eng	**Est**	**Fin**	**Tur**	**Avg**	**-Est**	**Eng**	**Est**	**Fin**	**Tur**	**Ger**	**Avg**	**-Est/Ger**
Morf. trans	67.3	**73.9**	61.2	57.1	64.9	61.9	78.4	78.8	61.8	49.8	**65.2**	66.8	63.3
Morf. ind	65.7	57.7	60.8	60.1	61.1	62.2	76.5	70.5	59.6	47.0	64.1	63.5	61.0
Morsel	-	-	-	-	-	-	**81.9**	77.2	63.3	47.8	59.0	65.8	**64.3**
Morf. ssv	**77.8**	-	**71.7**	**68.9**	-	**72.8**	80.6	-	62.1	**49.9**	-	-	64.2
AG ssv best	70.3*	68.6†	64.9*	58.2*	65.5	64.5	75.9*	80.3†	61.3*	46.1*	-	-	61.1
AG Select	74.4	71.7	70.0	61.4	**69.4**	68.6	81.3	**81.0**	**64.0**	47.5	63.8	**67.5**	**64.3**

Table 3: Test set results for unsupervised baselines Morfessor CatMAP (in transductive and inductive mode) and Morsel; semi-supervised Morfessor; and AG semi-supervised ($\star$ marks the Compounding grammar, $\dagger$ denotes SubMorphs grammar, and $*$ is the MorphSeq grammar) and grammar selection methods. Results are shown for each language, averaged over all languages (when possible: Avg), and averaged over just the languages where scores are available for all systems (-Est, -Est/Ger).

for English, together with their relative frequencies. It shows that at the Word level the binary rule is selected over three times more frequently than the unary rule. Also, most of the more frequently used grammar rules expand the first branch (rooted in M1) into more finegrained structures. The second branch (M2) is mostly modelled with the unary rule.

Among the frequently cached rules we see the common English prefixes and suffixes. One of the most frequent cached rule stores the single letter *e* at the end of a word, which often causes oversegmentation of words ending in *e* (as seen in the incorrect examples in Table 4). This problem is common in unsupervised morphological segmentation of English (Goldwater et al., 2006; Goldsmith, 2001).

We also took a look at the most frequent cached rules learned by the semi-supervised AG with the SubMorphs grammar, and observed that Morphs

Correct Segmentations		Incorrect Segmentations	
Word	**Segmentation**	**Induced**	**Correct**
treatable	[tr.ea.t]$_{M1}$ [a.b.le]$_{M21}$	disagree_s	dis_agree_s
disciplined	[dis.cip.l.i.n]$_{M1}$ [e.d]$_{M21}$	reduc_e	reduce
monogamous	[mon.o.g.a.m]$_{M1}$ [o.u.s]$_{M21}$	revalu_e	re_value
streakers	[st.r.e.a.k]$_{M1}$ [e.r]$_{M21}$ [s]$_{M2211}$	derid_e	deride
tollgate	[t.o.l.l.]$_{M1}$ [g.a.t.e]$_{M21}$ [s]$_{M2211}$	accompani_ed	ac_compani_ed
foxhunting	[f.o.x]$_{M1}$ [h.u.n.t]$_{M21}$ [ing]$_{M2211}$	war_y	wary
muscovites	[m.u.sc.o.v]$_{M1}$ [i.t.e]$_{M21}$ [s]$_{M2211}$	indescrib_able	in_describ_able
standardizes	[st.a.n.d.a.rd]$_{M1}$ [i.z.e]$_{M21}$ [s]$_{M2211}$	orat_es	orate_s
slavers'	[sl.a.v]$_{M1}$ [e.r]$_{M21}$ [s]$_{M2211}$ [']$_{M2212}$	alger_ian_s	algeri_an_s
earthiness'	[e.ar.th]$_{M1}$ [i]$_{M2111}$ [ness]$_{M2211}$ [']$_{M2212}$	disput_e_s	dispute_s
instinkt	[in.st.in.kt]$_{M1}$	meister_likkust	meisterlikkus_t
rebis	[re.b.i]$_{M1}$ [s]$_{M2}$	min_a	mina
toitsid	[to.it]$_{M1}$ [s.id]$_{M2}$	teiste	teis_te
armuavaldus	[a.rm.u]$_{M11}$ [ava.ld.u.s]$_{M12}$	kuritegu_de_sse	kuri_tegu_desse
määgivale	[mää.g.i]$_{M11}$ [v.a]$_{M12}$ [l.e]$_{M2}$	liharoa_ga	liha_roa_ga
keskuskoulussa	[kesk.us]$_{M11}$ [koul.u]$_{M12}$ [s.sa]$_{M2}$	polte_tti_in	polte_tt_i_in
peruslähteille	[per.u.s]$_{M11}$ [l.ä.ht.e]$_{M12}$ [i]$_{M211}$ [ll.e]$_{M212}$	kulttuuri_se_lt_a_kin	kulttuurise_lta_kin
perunakaupoista	[per.u.n.a]$_{M11}$ [k.au.p.o]$_{M12}$ [i]$_{M211}$ [st.a]$_{M212}$	tuote_palki_ntoja	tuote_palkinto_j_a
yöpaikkaani	[yö]$_{M11}$ [p.ai.kk.a]$_{M12}$ [a]$_{M21}$ [n.i]$_{M22}$	veli_puo_lt_a	veli_puol_ta
nimettäköön	[ni.m.e]$_{M11}$ [tt.ä]$_{M12}$ [k.ö]$_{M21}$ [ö.n]$_{M22}$	ota_ttava	otatta_va

Table 4: Examples of segmented words in English (top), Estonian (middle) and Finnish (bottom). Correctly segmented words are in the left part of the table. The identified segments are in brackets indexed by the respective template nonterminal; dots separate the metagrammar generated parse tree leaves. Examples of incorrectly segmented words together with the correct segmentation are on the right.

Freq (%)	Rule	Freq (%)	Cached Rule
9.9	Word → M1 M2	1.2	(M2 (M21 (M211 (M2111 s))))
5.7	M1 → M11 M12	0.9	(M2 (M21 (M211 (M2111 e)) (M212 (M2121 d))))
3.1	Word → M1	0.7	(M2 (M21 (M211 (M2111 i)) (M212 (M2121 n g))))
2.5	M11 → M111	0.6	(M2 (M21 (M211 (M2111 e)))
1.8	M2 → M21	0.4	(M2 (M21 (M211 (M2111 '))) (M22 (M221 (M2211 s))))
1.4	M12→ M121 M122	0.3	(M1112 a)
1.4	M111 → M1111 M1112	0.3	(M2 (M21 (M211 (M2111 y))))
0.9	M12 → M121	0.3	(M2 (M21 (M211 (M2111 e))) (M212 (M2121 r)))
0.9	M11 → M111 M112	0.2	(M2 (M21 (M211 (M2111 a))))

Table 5: Examples from English most frequently used metagrammar rules and cached rules together with their relative occurrence frequencies (in percentages).

tended to contain only a single SubMorph. This helps to explain why the SubMorphs grammar in semi-supervised AG improved less over the unsupervised AG as compared to the MorphSeq grammar—the rules with only a single SubMorph under the Morph are essentially the same as they would be in the MorphSeq grammar.

Finally, we examined the consistency of the templates chosen for each of the 5 samplers during model selection for the test set (Section 4.4). We found that there was some variability in the templates, but in most experiments the same template was chosen for the majority of the samplers (see Table 6). While this majority template is not always the optimal one on the dev set, we observed that it does produce consistently good results. It is possible that using the majority template, rather than the optimal template for the dev set, would actually produce better results

	Majority template
English	M1 M21 M2211 M2212 M222
Finnish	M11 M12 M211 M212 M22
Turkish	M11 M12 M211 M212 M22
German	M11 M121 M122 M21 M221 M222
Estonian	M11 M12 M2

Table 6: Majority templates for each language. Note that the Estonian gold standard contains less fine-grained segmentations than some of the other languages.

on the test set, especially if (as appears to be the case here, and may often be the case in real applications) the dev and test sets are from somewhat different distributions.

It must be noted that both AG Select and semi-supervised AG are computationally more demanding than the comparison systems. Since we do inference over tree structures, the complexity is cubic in the input word length, while most segmentation systems are quadratic or linear. Even compared to the unsupervised AG, AG Select is more expensive, because of the larger grammar and number of cached symbols. Nevertheless, our systems can feasibly be run on the large Morpho Challenge datasets.

Other recent unsupervised systems have reported state-of-the art results by incorporating additional information from surrounding words (Lee et al., 2011), multilingual alignments (Snyder and Barzilay, 2008), or overlapping context features in a log-linear model (Poon et al., 2009), but they have only been run on Semitic languages and English (and in the latter case, a very small corpus). Since they explicitly enumerate and sample from all possible segmentations of each word (often with some heuristic constraints), they could have trouble with the much longer words of the agglutinative languages tested here. In any case the results are not directly comparable to ours.

6 Conclusion

In this paper we have introduced three new methods for Adaptor Grammars and demonstrated their usefulness for minimally supervised morphological segmentation. First, we showed that AG models can be scaled to large data sets by using the posterior grammar for defining an inductive model, that on average results in the same accuracy as compared to full transductive training.

Second, we implemented semi-supervised AG inference, which uses labelled data to constrain the sampler, and showed that in all cases it performs much better than the unsupervised AG on the same grammar. Semi-supervised AG could benefit from labelled data reweighting techniques frequently used in semi-supervised learning, and studying the proper ways of doing so within the AG framework would be a potential topic for future research.

Our final contribution is the AG Select method, where the initial model is trained using a very general grammar that oversegments the data, and the labelled data is used to select which granularity of segments to use. Unlike other morphological segmentation models, this method can adapt its grammar to languages with different structures, rather than having to use the same grammar for every language. Indeed, we found that AG Select performs well across a range of languages and also seems to be less sensitive to differences between data sets (here, dev vs. test). In addition, it can be trained on either morphological analyses or segmentations. Although we tuned all results to optimize the SBF1 metric, in principle the same method could be used to optimize other measures, including extrinsic measures on downstream applications such as machine translation or information retrieval. In future we hope to show that this method can be used to improve performance on such applications, and also to explore its use for related segmentation tasks such as stemming or syllabification. Also, the method itself could potentially be improved by designing a classifier to determinine the best template for each word based on a set of features, rather than using a single template for all words in the language.

Acknowledgments

This work was supported by the Tiger University program of Estonian Information Technology Foundation for the first author. We thank Constantine Lignos for releasing his Morsel code to us, Sami Virpioja for evaluating test set results, and Federico Sangati for providing useful scripts.

References

Mathias Creutz and Krista Lagus. 2007. Unsupervised models for morpheme segmentation and morphology

learning. *ACM Transactions of Speech and Language Processing*, 4(1):1–34, February.

Micha Elsner, Eugene Charniak, and Mark Johnson. 2009. Structured generative models for unsupervised named-entity clustering. In *Proceedings of NAACL*, pages 164–172. Association for Computational Linguistics.

John Goldsmith. 2001. Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, 27(2):153–198, June.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. 2006. Interpolating between types and tokens by estimating power-law generators. In *Advances in Neural Information Processing Systems 18*, pages 459–466, Cambridge, MA. MIT Press.

Eric A. Hardisty, Jordan Boyd-Graber, and Philip Resnik. 2010. Modeling perspective using adaptor grammars. In *Proceedings of EMNLP*, pages 284–292. Association for Computational Linguistics.

Zellig Harris. 1951. *Structural Linguistics*. University of Chicago Press.

Yun Huang, Min Zhang, and Chew Lim Tan. 2011. Nonparametric bayesian machine transliteration with synchronous adaptor grammars. In *Proceedings of ACL: short papers - Volume 2*, pages 534–539. Association for Computational Linguistics.

Mark Johnson and Katherine Demuth. 2010. Unsupervised phonemic chinese word segmentation using adaptor grammars. In *Proceedings of COLING*, pages 528–536. Association for Computational Linguistics.

Mark Johnson and Sharon Goldwater. 2009. Improving nonparameteric bayesian inference: experiments on unsupervised word segmentation with adaptor grammars. In *Proceedings of NAACL*, pages 317–325. Association for Computational Linguistics.

Mark Johnson, Thomas L. Griffiths, and Sharon Goldwater. 2007. Adaptor grammars: A framework for specifying compositional nonparametric bayesian models. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 641–648. MIT Press, Cambridge, MA.

Mark Johnson. 2008a. Unsupervised word segmentation for sesotho using adaptor grammars. In *Proceedings of ACL Special Interest Group on Computational Morphology and Phonology*, pages 20–27. Association for Computational Linguistics.

Mark Johnson. 2008b. Using adaptor grammars to identify synergies in the unsupervised acquisition of linguistic structure. In *Proceedings of ACL*, pages 398–406. Association for Computational Linguistics.

Oskar Kohonen, Sami Virpioja, and Krista Lagus. 2010a. Semi-supervised learning of concatenative morphology. In *Proceedings of ACL Special Interest Group on Computational Morphology and Phonology*, pages 78–86. Association for Computational Linguistics.

Oskar Kohonen, Sami Virpioja, Laura Leppänen, and Krista Lagus. 2010b. Semi-supervised extensions to morfessor baseline. In *Proceedings of the Morpho Challenge 2010 Workshop*, pages 30–34. Aalto University School of Science and Technology.

Yoong Keok Lee, Aria Haghighi, and Regina Barzilay. 2011. Modeling syntactic context improves morphological segmentation. In *Proceedings of CoNLL*, pages 1–9. Association for Computational Linguistics.

Percy Liang, Hal Daumé, III, and Dan Klein. 2008. Structure compilation: trading structure for features. In *Proceedings of ICML*, pages 592–599. Association for Computing Machinery.

Constantine Lignos. 2010. Learning from Unseen Data. In Mikko Kurimo, Sami Virpioja, and Ville T. Turunen, editors, *Proceedings of the Morpho Challenge 2010 Workshop*, pages 35–38. Aalto University School of Science and Technology.

Jim Pitman and Marc Yor. 1997. The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *Annals of Probability*, 25(2):855–900.

Hoifung Poon, Colin Cherry, and Kristina Toutanova. 2009. Unsupervised morphological segmentation with log-linear models. In *Proceedings of NAACL*, pages 209–217. Association for Computational Linguistics.

Benjamin Snyder and Regina Barzilay. 2008. Unsupervised multilingual learning for morphological segmentation. In *Proceedings of ACL*, pages 737–745. Association for Computational Linguistics.

Sebastian Spiegler and Christian Monson. 2010. Emma: A novel evaluation metric for morphological analysis. In *Proceedings of COLING*, pages 1029–1037. Association for Computational Linguistics.

Sze-Meng Jojo Wong, Mark Dras, and Mark Johnson. 2012. Exploring adaptor grammars for native language identification. In *Proceedings of EMNLP*, pages 699–709. Association for Computational Linguistics.

Appendix D

Publication III

Sirts, K., Eisenstein, J., Elsner, M., Goldwater, S. (2014). POS induction with distributional and morphological information using a distance-dependent Chinese restaurant process. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 265–271.

POS induction with distributional and morphological information using a distance-dependent Chinese restaurant process

Kairit Sirts
Institute of Cybernetics at
Tallinn University of Technology
sirts@ioc.ee

Jacob Eisenstein
School of Interactive Computing
Georgia Institute of Technology
jacobe@gatech.edu

Micha Elsner
Department of Linguistics
The Ohio State University
melsner0@gmail.com

Sharon Goldwater
ILCC, School of Informatics
University of Edinburgh
sgwater@inf.ed.ac.uk

Abstract

We present a new approach to inducing the syntactic categories of words, combining their distributional and morphological properties in a joint nonparametric Bayesian model based on the distance-dependent Chinese Restaurant Process. The prior distribution over word clusterings uses a log-linear model of morphological similarity; the likelihood function is the probability of generating vector word embeddings. The weights of the morphology model are learned jointly while inducing part-of-speech clusters, encouraging them to cohere with the distributional features. The resulting algorithm outperforms competitive alternatives on English POS induction.

1 Introduction

The morphosyntactic function of words is reflected in two ways: their distributional properties, and their morphological structure. Each information source has its own advantages and disadvantages. Distributional similarity varies smoothly with syntactic function, so that words with similar syntactic functions should have similar distributional properties. In contrast, there can be multiple paradigms for a single morphological inflection (such as past tense in English). But accurate computation of distributional similarity requires large amounts of data, which may not be available for rare words; morphological rules can be applied to any word regardless of how often it appears.

These observations suggest that a general approach to the induction of syntactic categories should leverage both distributional and morphological features (Clark, 2003; Christodoulopoulos et al., 2010). But these features are difficult to combine because of their disparate representations. Distributional information is typically represented in numerical vectors, and recent work has demonstrated the utility of continuous vector representations, or "embeddings" (Mikolov et al., 2013; Luong et al., 2013; Kim and de Marneffe, 2013; Turian et al., 2010). In contrast, morphology is often represented in terms of sparse, discrete features (such as morphemes), or via pairwise measures such as string edit distance. Moreover, the mapping between a surface form and morphology is complex and nonlinear, so that simple metrics such as edit distance will only weakly approximate morphological similarity.

In this paper we present a new approach for inducing part-of-speech (POS) classes, combining morphological and distributional information in a non-parametric Bayesian generative model based on the *distance-dependent Chinese restaurant process* (ddCRP; Blei and Frazier, 2011). In the ddCRP, each data point (word type) selects another point to "follow"; this chain of following links corresponds to a partition of the data points into clusters. The probability of word w_1 following w_2 depends on two factors: 1) the *distributional* similarity between all words in the proposed partition containing w_1 and w_2, which is encoded using a Gaussian likelihood function over the word embeddings; and 2) the *morphological* similarity between w_1 and w_2, which acts as a prior distribution on the induced clustering. We use a log-linear model to capture suffix similarities between words, and learn the feature weights by iterating between sampling and weight learning.

We apply our model to the English section of the the Multext-East corpus (Erjavec, 2004) in order to evaluate both against the coarse-grained and

Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers), pages 265–271,
Baltimore, Maryland, USA, June 23-25 2014.

fine-grained tags, where the fine-grained tags encode detailed morphological classes. We find that our model effectively combines morphological features with distributional similarity, outperforming comparable alternative approaches.

2 Related work

Unsupervised POS tagging has a long history in NLP. This paper focuses on the POS induction problem (i.e., no tag dictionary is available), and here we limit our discussion to very recent systems. A review and comparison of older systems is provided by Christodoulopoulos et al. (2010), who found that imposing a one-tag-per-word-type constraint to reduce model flexibility tended to improve system performance; like other recent systems, we impose that constraint here. Recent work also shows that the combination of morphological and distributional information yields the best results, especially cross-linguistically (Clark, 2003; Berg-Kirkpatrick et al., 2010). Since then, most systems have incorporated morphology in some way, whether as an initial step to obtain prototypes for clusters (Abend et al., 2010), or as features in a generative model (Lee et al., 2010; Christodoulopoulos et al., 2011; Sirts and Alumäe, 2012), or a representation-learning algorithm (Yatbaz et al., 2012). Several of these systems use a small fixed set of orthographic and/or suffix features, sometimes obtained from an unsupervised morphological segmentation system (Abend et al., 2010; Lee et al., 2010; Christodoulopoulos et al., 2011; Yatbaz et al., 2012). Blunsom and Cohn's (2011) model learns an n-gram character model over the words in each cluster; we learn a log-linear model, which can incorporate arbitrary features. Berg-Kirkpatrick et al. (2010) also include a log-linear model of morphology in POS induction, but they use morphology in the likelihood term of a parametric sequence model, thereby encouraging all elements that share a tag to have the same morphological features. In contrast, we use *pairwise morphological similarity* as a prior in a non-parametric clustering model. This means that the membership of a word in a cluster requires only morphological similarity to some other element in the cluster, not to the cluster centroid; which may be more appropriate for languages with multiple morphological paradigms. Another difference is that our non-parametric formulation makes it unnecessary to know the number of tags in advance.

3 Distance-dependent CRP

The ddCRP (Blei and Frazier, 2011) is an extension of the CRP; like the CRP, it defines a distribution over partitions ("table assignments") of data points ("customers"). Whereas in the regular CRP each customer chooses a table with probability proportional to the number of customers already sitting there, in the ddCRP each customer chooses another *customer* to follow, and sits at the same table with that customer. By identifying the connected components in this graph, the ddCRP equivalently defines a prior over clusterings.

If c_i is the index of the customer followed by customer i, then the ddCRP prior can be written

$$P(c_i = j) \propto \begin{cases} f(d_{ij}) & \text{if } i \neq j \\ \alpha & \text{if } i = j, \end{cases} \quad (1)$$

where d_{ij} is the distance between customers i and j and f is a decay function. A ddCRP is *sequential* if customers can only follow previous customers, i.e., $d_{ij} = \infty$ when $i > j$ and $f(\infty) = 0$. In this case, if $d_{ij} = 1$ for all $i < j$ then the ddCRP reduces to the CRP.

Separating the distance and decay function makes sense for "natural" distances (e.g., the number of words between word i and j in a document, or the time between two events), but they can also be collapsed into a single similarity function. We wish to assign higher similarities to pairs of words that share meaningful suffixes. Because we do not know which suffixes are meaningful *a priori*, we use a maximum entropy model whose features include all suffixes up to length three that are shared by at least one pair of words. Our prior is then:

$$P(c_i = j|\mathbf{w}, \alpha) \propto \begin{cases} e^{\mathbf{w}^\top \mathbf{g}(i,j)} & \text{if } i \neq j \\ \alpha & \text{if } i = j, \end{cases} \quad (2)$$

where $g_s(i, j)$ is 1 if suffix s is shared by ith and jth words, and 0 otherwise.

We can create an infinite mixture model by combining the ddCRP prior with a likelihood function defining the probability of the data given the cluster assignments. Since we are using continuous-valued vectors (word embeddings) to represent the distributional characteristics of words, we use a multivariate Gaussian likelihood. We will marginalize over the mean μ and covariance Σ of each cluster, which in turn are drawn from Gaussian and inverse-Wishart (IW) priors respectively:

$$\Sigma \sim IW(\nu_0, \Lambda_0) \qquad \mu \sim \mathcal{N}(\mu_0, {}^{\Sigma}/_{\kappa_0}) \quad (3)$$

The full model is then:

$$P(\mathbf{X},\mathbf{c},\boldsymbol{\mu},\boldsymbol{\Sigma}|\Theta,\mathbf{w},\alpha) \quad (4)$$
$$= \prod_{k=1}^{K} P(\Sigma_k|\Theta)p(\mu_k|\Sigma_k,\Theta)$$
$$\times \prod_{i=1}^{n} (P(c_i|\mathbf{w},\alpha)P(\mathbf{x}_i|\mu_{z_i},\Sigma_{z_i})),$$

where Θ are the hyperparameters for $(\boldsymbol{\mu},\boldsymbol{\Sigma})$ and z_i is the (implicit) cluster assignment of the ith word $\mathbf{x}_i$. With a CRP prior, this model would be an infinite Gaussian mixture model (IGMM; Rasmussen, 2000), and we will use the IGMM as a baseline.

4 Inference

The Gibbs sampler for the ddCRP integrates over the Gaussian parameters, sampling only follower variables. At each step, the follower link c_i for a single customer i is sampled, which can implicitly shift the entire block of n customers $\text{fol}(i)$ who follow i into a new cluster. Since we marginalize over the cluster parameters, computing $P(c_i = j)$ requires computing the likelihood $P(\text{fol}(i), \mathbf{X}_j|\Theta)$, where $\mathbf{X}_j$ are the k customers already clustered with j. However, if we do *not* merge $\text{fol}(i)$ with $\mathbf{X}_j$, then we have $P(\mathbf{X}_j|\Theta)$ in the overall joint probability. Therefore, we can decompose $P(\text{fol}(i), \mathbf{X}_j|\Theta) = P(\text{fol}(i)|\mathbf{X}_j,\Theta)P(\mathbf{X}_j|\Theta)$ and need only compute the change in likelihood due to merging in $\text{fol}(i)$:[1]:

$$P(\text{fol}(i)|\mathbf{X}_j,\Theta) = \pi^{-nd/2}\frac{\kappa_k^{d/2}|\Lambda_k|^{\nu_k/2}}{\kappa_{n+k}^{d/2}|\Lambda_{n+k}|^{\nu_{n+k}/2}}$$
$$\times \prod_{i=1}^{d}\frac{\Gamma\left(\frac{\nu_{n+k}+1-i}{2}\right)}{\Gamma\left(\frac{\nu_k+1-i}{2}\right)}, \quad (5)$$

where the hyperparameters are updated as $\kappa_n = \kappa_0 + n$, $\nu_n = \nu_0 + n$, and

$$\mu_n = \frac{\kappa_0\mu_0 + \bar{x}}{\kappa_0 + n} \quad (6)$$
$$\Lambda_n = \Lambda_0 + Q + \kappa_0\mu_0\mu_0{}^T - \kappa_n\mu_n\mu_n^T, \quad (7)$$

where $Q = \sum_{i=1}^{n}\mathbf{x}_i\mathbf{x}_i^T$.

Combining this likelihood term with the prior, the probability of customer i following j is

$$P(c_i = j|\mathbf{X},\Theta,\mathbf{w},\alpha)$$
$$\propto P(\text{fol}(i)|\mathbf{X}_j,\Theta)P(c_i = j|\mathbf{w},\alpha). \quad (8)$$

Our non-sequential ddCRP introduces cycles into the follower structure, which are handled in the sampler as described by Socher et al. (2011). Also, the block of customers being moved around can potentially be very large, which makes it easy for the likelihood term to swamp the prior. In practice we found that introducing an additional parameter a (used to exponentiate the prior) improved results—although we report results without this exponent as well. This technique was also used by Titov and Klementiev (2012) and Elsner et al. (2012).

Inference also includes optimizing the feature weights for the log-linear model in the ddCRP prior (Titov and Klementiev, 2012). We interleave L-BFGS optimization within sampling, as in Monte Carlo Expectation-Maximization (Wei and Tanner, 1990). We do not apply the exponentiation parameter a when training the weights because this procedure affects the follower structure only, and we do not have to worry about the magnitude of the likelihood. Before the first iteration we initialize the follower structure: for each word, we choose randomly a word to follow from amongst those with the longest shared suffix of up to 3 characters. The number of clusters starts around 750, but decreases substantially after the first sampling iteration.

5 Experiments

Data For our experiments we used the English word embeddings from the Polyglot project (Al-Rfou' et al., 2013)[2], which provides embeddings trained on Wikipedia texts for 100,000 of the most frequent words in many languages.

We evaluate on the English part of the Multext-East (MTE) corpus (Erjavec, 2004), which provides both coarse-grained and fine-grained POS labels for the text of Orwell's "1984". Coarse labels consist of 11 main word classes, while the fine-grained tags (104 for English) are sequences of detailed morphological attributes. Some of these attributes are not well-attested in English (e.g. gender) and some are mostly distinguishable via semantic analysis (e.g. 1st and 2nd person verbs). Many tags are assigned only to one or a few words. Scores for the fine-grained tags will be lower for these reasons, but we argue below that they are still informative.

Since Wikipedia and MTE are from different domains their lexicons do not fully overlap; we

[1] http://www.stats.ox.ac.uk/~teh/research/notes/GaussianInverseWishart.pdf

[2] https://sites.google.com/site/rmyeid/projects/polyglot

Wikipedia tokens	1843M
Multext-East tokens	118K
Multext-East types	9193
Multext-East & Wiki types	7540

Table 1: Statistics for the English Polyglot word embeddings and English part of MTE: number of Wikipedia tokens used to train the embeddings, number of tokens/types in MTE, and number of types shared by both datasets.

take the intersection of these two sets for training and evaluation. Table 1 shows corpus statistics.

Evaluation With a few exceptions (Biemann, 2006; Van Gael et al., 2009), POS induction systems normally require the user to specify the number of desired clusters, and the systems are evaluated with that number set to the number of tags in the gold standard. For corpora such as MTE with both fine-grained and coarse-grained tages, previous evaluations have scored against the coarse-grained tags. Though coarse-grained tags have their place (Petrov et al., 2012), in many cases the distributional and morphological distinctions between words are more closely aligned with the fine-grained tagsets, which typically distinguish between verb tenses, noun number and gender, and adjectival scale (comparative, superlative, etc.), so we feel that the evaluation against fine-grained tagset is more relevant here. For better comparison with previous work, we also evaluate against the coarse-grained tags; however, these numbers are not strictly comparable to other scores reported on MTE because we are only able to train and evaluate on the subset of words that also have Polyglot embeddings. To provide some measure of the difficulty of the task, we report baseline scores using K-means clustering, which is relatively strong baseline in this task (Christodoulopoulos et al., 2011).

There are several measures commonly used for unsupervised POS induction. We report greedy one-to-one mapping accuracy (1-1) (Haghighi and Klein, 2006) and the information-theoretic score V-measure (V-m), which also varies from 0 to 100% (Rosenberg and Hirschberg, 2007). In previous work it has been common to also report many-to-one (m-1) mapping but this measure is particularly sensitive to the number of induced clusters (more clusters yield higher scores), which is variable for our models. V-m can be somewhat sensitive to the number of clusters (Reichart and Rappoport, 2009) but much less so than m-1 (Christodoulopoulos et al., 2010). With different number of induced and gold standard clusters the 1-1 measure suffers because some induced clusters cannot be mapped to gold clusters or vice versa. However, almost half the gold standard clusters in MTE contain just a few words and we do not expect our model to be able to learn them anyway, so the 1-1 measure is still useful for telling us how well the model learns the bigger and more distinguishable classes.

In unsupervised POS induction it is standard to report accuracy on tokens even when the model itself works on types. Here we report also type-based measures because these can reveal differences in model behavior even when token-based measures are similar.

Experimental setup For baselines we use K-means and the IGMM, which both only learn from the word embeddings. The CRP prior in the IGMM has one hyperparameter (the concentration parameter α); we report results for $\alpha = 5$ and 20. Both the IGMM and ddCRP have four hyperparameters controlling the prior over the Gaussian cluster parameters: Λ_0, μ_0, ν_0 and κ_0. We set the prior scale matrix Λ_0 by using the average covariance from a K-means run with $K = 200$. When setting the average covariance as the expected value of the IW distribution the suitable scale matrix can be computed as $\Lambda_0 = E[X](\nu_0 - d - 1)$, where ν_0 is the prior degrees of freedom (which we set to $d + 10$) and d is the data dimensionality (64 for the Polyglot embeddings). We set the prior mean μ_0 equal to the sample mean of the data and κ_0 to 0.01.

We experiment with three different priors for the ddCRP model. All our ddCRP models are non-sequential (Socher et al., 2011), allowing cycles to be formed. The simplest model, *ddCRP uniform*, uses a uniform prior that sets the distance between any two words equal to one.[3] The second model, *ddCRP learned*, uses the log-linear prior with weights learned between each two Gibbs iterations as explained in section 4. The final model, *ddCRP exp*, adds the prior exponentiation. The α parameter for the ddCRP is set to 1 in all experiments. For *ddCRP exp*, we report results with the exponent a set to 5.

Results and discussion Table 2 presents all results. Each number is an average of 5 experiments

[3] In the sequential case this model would be equivalent to the IGMM (Blei and Frazier, 2011). Due to the nonsequentiality this equivalence does not hold, but we do expect to see similar results to the IGMM.

		Fine types		Fine tokens		Coarse tokens	
Model	**K**	Model	K-means	Model	K-means	Model	K-means
K-means	104 or 11	16.1 / 47.3	-	39.2 / 62.0	-	44.4 / 45.5	-
IGMM, $\alpha = 5$	55.6	41.0 / 45.9	23.1 / 49.5	48.0 / 64.8	37.2 / 61.0	48.3 / 58.3	40.8 / 55.0
IGMM, $\alpha = 20$	121.2	35.0 / 47.1	14.7 / 46.9	50.6 / 67.8	44.7 / 65.5	48.7 / 60.0	48.3 / 57.9
ddCRP uniform	80.4	50.5 / 52.9	18.6 / 48.2	52.4 / 68.7	35.1 / 60.3	**52.1 / 62.2**	40.3 / 54.2
ddCRP learned	89.6	50.1 / 55.1	17.6 / 48.0	51.1 / **69.7**	39.0 / 63.2	48.9 / 62.0	41.1 / 55.1
ddCRP exp, $a = 5$	47.2	**64.0 / 60.3**	25.0 / 50.3	**55.1** / 66.4	33.0 / 59.1	47.8 / 55.1	36.9 / 53.1

Table 2: Results of baseline and ddCRP models evaluated on word types and tokens using fine-grained tags, and on tokens using coarse-grained tags. For each model we present the number of induced clusters K (or fixed K for K-means) and 1-1 / V-m scores. The second column under each evaluation setting gives the scores for K-means with K equal to the number of clusters induced by the model in that row.

with different random initializations. For each evaluation setting we provide two sets of scores—first are the 1-1 and V-m scores for the given model, second are the comparable scores for K-means run with the same number of clusters as induced by the non-parametric model.

These results show that all non-parametric models perform better than K-means, which is a strong baseline in this task (Christodoulopoulos et al., 2011). The poor performace of K-means can be explained by the fact that it tends to find clusters of relatively equal size, although the POS clusters are rarely of similar size. The common noun singular class is by far the largest in English, containing roughly a quarter of the word types. Non-parametric models are able to produce cluster of different sizes when the evidence indicates so, and this is clearly the case here.

From the token-based evaluation it is hard to say which IGMM hyperparameter value is better even though the number of clusters induced differs by a factor of 2. The type-base evaluation, however, clearly prefers the smaller value with fewer clusters. Similar effects can be seen when comparing IGMM and ddCRP uniform. We expected these two models perform on the same level, and their token-based scores are similar, but on the type-based evaluation the ddCRP is clearly superior. The difference could be due to the non-sequentiality, or becuase the samplers are different—IGMM enabling resampling only one item at a time, ddCRP performing blocked sampling.

Further we can see that the ddCRP uniform and learned perform roughly the same. Although the prior in those models is different they work mainly using the the likelihood. The ddCRP with learned prior does produce nice follower structures within each cluster but the prior is in general too weak compared to the likelihood to influence the clustering decisions. Exponentiating the prior reduces the number of induced clusters and improves results, as it can change the cluster assignment for some words where the likelihood strongly prefers one cluster but the prior clearly indicates another.

The last column shows the token-based evaluation against the coarse-grained tagset. This is the most common evaluation framework used previously in the literature. Although our scores are not directly comparable with the previous results, our V-m scores are similar to the best published 60.5 (Christodoulopoulos et al., 2010) and 66.7 (Sirts and Alumäe, 2012).

In preliminary experiments, we found that directly applying the best-performing English model to other languages is not effective. Different languages may require different parametrizations of the model. Further study is also needed to verify that word embeddings effectively capture syntax across languages, and to determine the amount of unlabeled text necessary to learn good embeddings.

6 Conclusion

This paper demonstrates that morphology and distributional features can be combined in a flexible, joint probabilistic model, using the distance-dependent Chinese Restaurant Process. A key advantage of this framework is the ability to include arbitrary features in the prior distribution. Future work may exploit this advantage more thoroughly: for example, by using features that incorporate prior knowledge of the language's morphological structure. Another important goal is the evaluation of this method on languages beyond English.

Acknowledgments: KS was supported by the Tiger University program of the Estonian Information Technology Foundation for Education. JE was supported by a visiting fellowship from the Scottish Informatics & Computer Science Alliance. We thank the reviewers for their helpful feedback.

References

Omri Abend, Roi Reichart, and Ari Rappoport. 2010. Improved unsupervised pos induction through prototype discovery. In *Proceedings of the 48th Annual Meeting of the Association of Computational Linguistics*, pages 1298–1307.

Rami Al-Rfou', Bryan Perozzi, and Steven Skiena. 2013. Polyglot: Distributed word representations for multilingual nlp. In *Proceedings of the Thirteenth Annual Conference on Natural Language Learning*, pages 183–192, Sofia, Bulgaria. Association for Computational Linguistics.

Taylor Berg-Kirkpatrick, Alexandre B. Côté, John DeNero, and Dan Klein. 2010. Painless unsupervised learning with features. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 582–590.

Chris Biemann. 2006. Unsupervised part-of-speech tagging employing efficient graph clustering. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 7–12.

David M Blei and Peter I Frazier. 2011. Distance dependent chinese restaurant processes. *Journal of Machine Learning Research*, 12:2461–2488.

Phil Blunsom and Trevor Cohn. 2011. A hierarchical pitman-yor process hmm for unsupervised part of speech induction. In *Proceedings of the 49th Annual Meeting of the Association of Computational Linguistics*, pages 865–874.

Christos Christodoulopoulos, Sharon Goldwater, and Mark Steedman. 2010. Two decades of unsupervised POS induction: How far have we come? In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Christos Christodoulopoulos, Sharon Goldwater, and Mark Steedman. 2011. A Bayesian mixture model for part-of-speech induction using multiple features. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Alexander Clark. 2003. Combining distributional and morphological information for part of speech induction. In *Proceedings of the European chapter of the ACL*.

Micha Elsner, Sharon Goldwater, and Jacob Eisenstein. 2012. Bootstrapping a unified model of lexical and phonetic acquisition. In *Proceedings of the 50th Annual Meeting of the Association of Computational Linguistics*.

Tomaž Erjavec. 2004. MULTEXT-East version 3: Multilingual morphosyntactic specifications, lexicons and corpora. In *LREC*.

A. Haghighi and D. Klein. 2006. Prototype-driven learning for sequence models. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*.

Joo-Kyung Kim and Marie-Catherine de Marneffe. 2013. Deriving adjectival scales from continuous space word representations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Yoong Keok Lee, Aria Haghighi, and Regina Barzilay. 2010. Simple type-level unsupervised pos tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 853–861.

Minh-Thang Luong, Richard Socher, and Christopher D Manning. 2013. Better word representations with recursive neural networks for morphology. In *Proceedings of the Thirteenth Annual Conference on Natural Language Learning*.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 746–751.

Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A universal part-of-speech tagset. In *Proceedings of LREC*, May.

Carl Rasmussen. 2000. The infinite Gaussian mixture model. In *Advances in Neural Information Processing Systems 12*, Cambridge, MA. MIT Press.

Roi Reichart and Ari Rappoport. 2009. The nvi clustering evaluation measure. In *Proceedings of the Ninth Annual Conference on Natural Language Learning*, pages 165–173.

A. Rosenberg and J. Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 410–42.

Kairit Sirts and Tanel Alumäe. 2012. A hierarchical Dirichlet process model for joint part-of-speech and morphology induction. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 407–416.

Richard Socher, Andrew L Maas, and Christopher D Manning. 2011. Spectral chinese restaurant processes: Nonparametric clustering based on similarities. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, pages 698–706.

Ivan Titov and Alexandre Klementiev. 2012. A bayesian approach to unsupervised semantic role induction. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*.

Joseph Turian, Lev-Arie Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394, Uppsala, Sweden, July. Association for Computational Linguistics.

Jurgen Van Gael, Andreas Vlachos, and Zoubin Ghahramani. 2009. The infinite HMM for unsupervised PoS tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 678–687, Singapore.

Greg CG Wei and Martin A Tanner. 1990. A monte carlo implementation of the em algorithm and the poor man's data augmentation algorithms. *Journal of the American Statistical Association*, 85(411):699–704.

Mehmet Ali Yatbaz, Enis Sert, and Deniz Yuret. 2012. Learning syntactic categories using paradigmatic representations of word context. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 940–951.

Curriculum Vitae

Contact data

Name: Kairit Sirts
Email: kairit.sirts@gmail.com

Education

2008 – 2015	Tallinn University of Technology (TUT), Phd studies in computer science
2011 – 2012	Research collaborator at University of Edinburgh, ILCC (Nov 2011 – March 2012)
2010 – 2010	Visiting student at Aalto University Speech Group at Department of Information and CS (March 2010)
2003 – 2008	TUT, MSc studies in computer science
1998 – 2003	TUT, BSc studies in computer science

Teaching experience

Spring 2014	Course in machine learning; main instructor
Spring 2013	Seminar in machine learning; main instructor
2010 – 2014	Various basic informatics courses for undergraduate students in social sciences and economics

Positions held

2010 – ...	Institute of Cybernetics at TUT; engineer
2002 – 2007	Hansabank; project manager, system analyst

Summer schools

MLSS 2014	The Machine Learning Summer School, Max Planck Institute for Intelligent Systems, Tübingen, Germany
ISSPR 2011	International Summer School on Pattern Recognition, Plymouth, UK

ESSCaSS	Estonian Summer School on Computer and Systems Science, 2011 – 2014
ESSLLI 2010	European Summer School of Logic, Language and Information, Copenhagen, Denmark

Other academic activities

reviewer	EMNLP 2012, Interspeech 2013 (sub-reviewer), ACL SRW 2014, Baltic HLT 2014 (sub-reviewer), NAACL 2015, NAACL SRW 2015
mentor	TUT first year CS bachelor students 2013, 2014

Scientific work

[1] Kairit Sirts, Jacob Eisenstein, Micha Elsner and Sharon Goldwater. POS induction with distributional and morphological information using a distance-dependent Chinese restaurant process. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 265–271, 2014.

[2] Kairit Sirts and Sharon Goldwater. Minimally supervised morphological segmentation with Adaptor Grammars. *Transactions of the Association for Computational Linguistics*, 1:255–266, 2013.

[3] Kairit Sirts. Noisy-channel spelling correction models for Estonian learner language corpus lemmatisation. In *Human Language Technologies - The Baltic Perspective: Proceedings of the Fifth International Conference Baltic HLT 2012*, pages 213–220, 2012.

[4] Kairit Sirts and Tanel Alumäe. A hierarchical Dirichlet process model for joint part-of-speech and morphology induction. In *Proceedings of the NAACL HLT 2012: The 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 407–416, 2012.

[5] Kairit Sirts. Heterogeneous statistical language model. *Acta Universitatis Latviensis*, 757:85–93, 2010.

[6] Kairit Sirts and Leo Võhandu. Cutting the text corpora: applications with syllables and sub-languages. *Estonian Papers in Applied Linguistics*, 5: 251–266, 2009.

[7] Leo Võhandu, Eik Aab and Kairit Sirts. A preliminary structural view of the Estonian syllable system. *Estonian Papers in Applied Linguistics*, 4:263–269, 2008.

[8] Tanel Alumäe, Leo Võhandu, Bert Viikmäe and Kairit Sirts. Estonian speech-driven pc-interface for disabled persons. In *Proceedings of the Second Baltic Conference on Human Language Technologies*, pages 359–364, 2005.

Elulookirjeldus

Kontaktandmed

Nimi: Kairit Sirts
Email: kairit.sirts@gmail.com

Hariduskäik

2008 – 2015	Tallinna Tehnikaülikool (TTÜ), doktoriõpingud informaatikas
2011 – 2012	külalistudeng Edinburghi ülikoolis, ILCC (nov 2011 – märts 2012)
2010 – 2010	külalistudeng Aalto ülikooli kõnetehnoloogia grupis (märts 2010)
2003 – 2008	TTÜ, magistriõpingud informaatikas
1998 – 2003	TTÜ, bakalaureuseõpingud informaatikas

Õpetamiskogemus

kevad 2014	Masinõppe kursus; põhiõppejõud
kevad 2013	Masinõppe seminar; põhiõppejõud
2010 – 2014	Erinevad informaatika aluskursused sotsiaal- ja majanduserialade bakalaureusetudengitele

Töökogemus

2010 – ...	TTÜ Küberneetika Instituut; insener
2002 – 2007	Hansapank; projektijuht, süsteemianalüütik

Suvekoolid

MLSS 2014	The Machine Learning Summer School, Max Planck Institute for Intelligent Systems, Tübingen, Saksamaa
ISSPR 2011	International Summer School on Pattern Recognition, Plymouth, Suurbritannia

ESSCaSS	Estonian Summer School on Computer and Systems Science, 2011 – 2014
ESSLLI 2010	European Summer School of Logic, Language and Information, Kopenhagen, Taani

Muud akadeemilised tegevused

retsensent	EMNLP 2012, Interspeech 2013 (kaasretsensent), ACL SRW 2014, Baltic HLT 2014 (kaasretsensent), NAACL 2015, NAACL SRW 2015
mentor	TTÜ esimese aasta informaatika tudengitele 2013, 2014

Teadustegevus

[1] Kairit Sirts, Jacob Eisenstein, Micha Elsner ja Sharon Goldwater. POS induction with distributional and morphological information using a distance-dependent Chinese restaurant process. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, lk 265–271, 2014.

[2] Kairit Sirts ja Sharon Goldwater. Minimally supervised morphological segmentation with Adaptor Grammars. *Transactions of the Association for Computational Linguistics*, 1:255–266, 2013.

[3] Kairit Sirts. Noisy-channel spelling correction models for Estonian learner language corpus lemmatisation. In *Human Language Technologies - The Baltic Perspective: Proceedings of the Fifth International Conference Baltic HLT 2012*, lk 213–220, 2012.

[4] Kairit Sirts ja Tanel Alumäe. A hierarchical Dirichlet process model for joint part-of-speech and morphology induction. In *Proceedings of the NAACL HLT 2012: The 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, lk 407–416, 2012.

[5] Kairit Sirts. Heterogeneous statistical language model. *Acta Universitatis Latviensis*, 757:85–93, 2010.

[6] Kairit Sirts ja Leo Võhandu. Korpuste tükeldamine: rakendusi silpide ning allkeeltega. *Eesti Rakenduslingvistika Ühingu aastaraamat*, 5: 251–266, 2009.

[7] Leo Võhandu, Eik Aab ja Kairit Sirts. Eesti silbisüsteemi struktuurist. *Eesti Rakenduslingvistika Ühingu aastaraamat*, 4:263–269, 2008.

[8] Tanel Alumäe, Leo Võhandu, Bert Viikmäe ja Kairit Sirts. Estonian speech-driven pc-interface for disabled persons. In *Proceedings of the Second Baltic Conference on Human Language Technologies*, lk 359–364, 2005.

DISSERTATIONS DEFENDED AT TALLINN UNIVERSITY OF TECHNOLOGY ON *INFORMATICS AND SYSTEM ENGINEERING*

1. **Lea Elmik**. Informational Modelling of a Communication Office. 1992.

2. **Kalle Tammemäe**. Control Intensive Digital System Synthesis. 1997.

3. **Eerik Lossmann**. Complex Signal Classification Algorithms, Based on the Third-Order Statistical Models. 1999.

4. **Kaido Kikkas**. Using the Internet in Rehabilitation of People with Mobility Impairments – Case Studies and Views from Estonia. 1999.

5. **Nazmun Nahar**. Global Electronic Commerce Process: Business-to-Business. 1999.

6. **Jevgeni Riipulk**. Microwave Radiometry for Medical Applications. 2000.

7. **Alar Kuusik**. Compact Smart Home Systems: Design and Verification of Cost Effective Hardware Solutions. 2001.

8. **Jaan Raik**. Hierarchical Test Generation for Digital Circuits Represented by Decision Diagrams. 2001.

9. **Andri Riid**. Transparent Fuzzy Systems: Model and Control. 2002.

10. **Marina Brik**. Investigation and Development of Test Generation Methods for Control Part of Digital Systems. 2002.

11. **Raul Land**. Synchronous Approximation and Processing of Sampled Data Signals. 2002.

12. **Ants Ronk**. An Extended Block-Adaptive Fourier Analyser for Analysis and Reproduction of Periodic Components of Band-Limited Discrete-Time Signals. 2002.

13. **Toivo Paavle**. System Level Modeling of the Phase Locked Loops: Behavioral Analysis and Parameterization. 2003.

14. **Irina Astrova**. On Integration of Object-Oriented Applications with Relational Databases. 2003.

15. **Kuldar Taveter**. A Multi-Perspective Methodology for Agent-Oriented Business Modelling and Simulation. 2004.

16. **Taivo Kangilaski**. Eesti Energia käiduhaldussüsteem. 2004.

17. **Artur Jutman**. Selected Issues of Modeling, Verification and Testing of Digital Systems. 2004.

18. **Ander Tenno**. Simulation and Estimation of Electro-Chemical Processes in Maintenance-Free Batteries with Fixed Electrolyte. 2004.

19. **Oleg Korolkov**. Formation of Diffusion Welded Al Contacts to Semiconductor Silicon. 2004.

20. **Risto Vaarandi**. Tools and Techniques for Event Log Analysis. 2005.

21. **Marko Koort**. Transmitter Power Control in Wireless Communication Systems. 2005.

22. **Raul Savimaa**. Modelling Emergent Behaviour of Organizations. Time-Aware, UML and Agent Based Approach. 2005.

23. **Raido Kurel**. Investigation of Electrical Characteristics of SiC Based Complementary JBS Structures. 2005.

24. **Rainer Taniloo**. Ökonoomsete negatiivse diferentsiaaltakistusega astmete ja elementide disainimine ja optimeerimine. 2005.

25. **Pauli Lallo**. Adaptive Secure Data Transmission Method for OSI Level I. 2005.

26. **Deniss Kumlander**. Some Practical Algorithms to Solve the Maximum Clique Problem. 2005.

27. **Tarmo Veskioja**. Stable Marriage Problem and College Admission. 2005.

28. **Elena Fomina**. Low Power Finite State Machine Synthesis. 2005.

29. **Eero Ivask**. Digital Test in WEB-Based Environment 2006.

30. **Виктор Войтович**. Разработка технологий выращивания из жидкой фазы эпитаксиальных структур арсенида галлия с высоковольтным p-n переходом и изготовления диодов на их основе. 2006.

31. **Tanel Alumäe**. Methods for Estonian Large Vocabulary Speech Recognition. 2006.

32. **Erki Eessaar**. Relational and Object-Relational Database Management Systems as Platforms for Managing Softwareengineering Artefacts. 2006.

33. **Rauno Gordon**. Modelling of Cardiac Dynamics and Intracardiac Bio-impedance. 2007.

34. **Madis Listak**. A Task-Oriented Design of a Biologically Inspired Underwater Robot. 2007.

35. **Elmet Orasson**. Hybrid Built-in Self-Test. Methods and Tools for Analysis and Optimization of BIST. 2007.

36. **Eduard Petlenkov**. Neural Networks Based Identification and Control of Nonlinear Systems: ANARX Model Based Approach. 2007.

37. **Toomas Kirt**. Concept Formation in Exploratory Data Analysis: Case Studies of Linguistic and Banking Data. 2007.

38. **Juhan-Peep Ernits**. Two State Space Reduction Techniques for Explicit State Model Checking. 2007.

39. **Innar Liiv**. Pattern Discovery Using Seriation and Matrix Reordering: A Unified View, Extensions and an Application to Inventory Management. 2008.

40. **Andrei Pokatilov**. Development of National Standard for Voltage Unit Based on Solid-State References. 2008.

41. **Karin Lindroos**. Mapping Social Structures by Formal Non-Linear Information Processing Methods: Case Studies of Estonian Islands Environments. 2008.

42. **Maksim Jenihhin**. Simulation-Based Hardware Verification with High-Level Decision Diagrams. 2008.

43. **Ando Saabas**. Logics for Low-Level Code and Proof-Preserving Program Transformations. 2008.

44. **Ilja Tšahhirov**. Security Protocols Analysis in the Computational Model – Dependency Flow Graphs-Based Approach. 2008.

45. **Toomas Ruuben**. Wideband Digital Beamforming in Sonar Systems. 2009.

46. **Sergei Devadze**. Fault Simulation of Digital Systems. 2009.

47. **Andrei Krivošei**. Model Based Method for Adaptive Decomposition of the Thoracic Bio-Impedance Variations into Cardiac and Respiratory Components. 2009.

48. **Vineeth Govind**. DfT-Based External Test and Diagnosis of Mesh-like Networks on Chips. 2009.

49. **Andres Kull**. Model-Based Testing of Reactive Systems. 2009.

50. **Ants Torim**. Formal Concepts in the Theory of Monotone Systems. 2009.

51. **Erika Matsak**. Discovering Logical Constructs from Estonian Children Language. 2009.

52. **Paul Annus**. Multichannel Bioimpedance Spectroscopy: Instrumentation Methods and Design Principles. 2009.

53. **Maris Tõnso**. Computer Algebra Tools for Modelling, Analysis and Synthesis for Nonlinear Control Systems. 2010.

54. **Aivo Jürgenson**. Efficient Semantics of Parallel and Serial Models of Attack Trees. 2010.

55. **Erkki Joasoon**. The Tactile Feedback Device for Multi-Touch User Interfaces. 2010.

56. **Jürgo-Sören Preden**. Enhancing Situation – Awareness Cognition and Reasoning of Ad-Hoc Network Agents. 2010.

57. **Pavel Grigorenko**. Higher-Order Attribute Semantics of Flat Languages. 2010.

58. **Anna Rannaste**. Hierarcical Test Pattern Generation and Untestability Identification Techniques for Synchronous Sequential Circuits. 2010.

59. **Sergei Strik**. Battery Charging and Full-Featured Battery Charger Integrated Circuit for Portable Applications. 2011.

60. **Rain Ottis**. A Systematic Approach to Offensive Volunteer Cyber Militia. 2011.

61. **Natalja Sleptšuk**. Investigation of the Intermediate Layer in the Metal-Silicon Carbide Contact Obtained by Diffusion Welding. 2011.

62. **Martin Jaanus**. The Interactive Learning Environment for Mobile Laboratories. 2011.

63. **Argo Kasemaa**. Analog Front End Components for Bio-Impedance Measurement: Current Source Design and Implementation. 2011.

64. **Kenneth Geers**. Strategic Cyber Security: Evaluating Nation-State Cyber Attack Mitigation Strategies. 2011.

65. **Riina Maigre**. Composition of Web Services on Large Service Models. 2011.

66. **Helena Kruus**. Optimization of Built-in Self-Test in Digital Systems. 2011.

67. **Gunnar Piho**. Archetypes Based Techniques for Development of Domains, Requirements and Sofware. 2011.

68. **Juri Gavšin**. Intrinsic Robot Safety Through Reversibility of Actions. 2011.

69. **Dmitri Mihhailov**. Hardware Implementation of Recursive Sorting Algorithms Using Tree-like Structures and HFSM Models. 2012.

70. **Anton Tšertov**. System Modeling for Processor-Centric Test Automation. 2012.

71. **Sergei Kostin**. Self-Diagnosis in Digital Systems. 2012.

72. **Mihkel Tagel**. System-Level Design of Timing-Sensitive Network-on-Chip Based Dependable Systems. 2012.

73. **Juri Belikov**. Polynomial Methods for Nonlinear Control Systems. 2012.

74. **Kristina Vassiljeva**. Restricted Connectivity Neural Networks based Identification for Control. 2012.

75. **Tarmo Robal**. Towards Adaptive Web – Analysing and Recommending Web Users` Behaviour. 2012.

76. **Anton Karputkin**. Formal Verification and Error Correction on High-Level Decision Diagrams. 2012.

77. **Vadim Kimlaychuk**. Simulations in Multi-Agent Communication System. 2012.

78. **Taavi Viilukas**. Constraints Solving Based Hierarchical Test Generation for Synchronous Sequential Circuits. 2012.

79. **Marko Kääramees**. A Symbolic Approach to Model-based Online Testing. 2012.

80. **Enar Reilent**. Whiteboard Architecture for the Multi-agent Sensor Systems. 2012.

81. **Jaan Ojarand**. Wideband Excitation Signals for Fast Impedance Spectroscopy of Biological Objects. 2012.

82. **Igor Aleksejev**. FPGA-based Embedded Virtual Instrumentation. 2013.

83. **Juri Mihhailov**. Accurate Flexible Current Measurement Method and its Realization in Power and Battery Management Integrated Circuits for Portable Applications. 2013.

84. **Tõnis Saar**. The Piezo-Electric Impedance Spectroscopy: Solutions and Applications. 2013.

85. **Ermo Täks**. An Automated Legal Content Capture and Visualisation Method. 2013.

86. **Uljana Reinsalu**. Fault Simulation and Code Coverage Analysis of RTL Designs Using High-Level Decision Diagrams. 2013.

87. **Anton Tšepurov**. Hardware Modeling for Design Verification and Debug. 2013.

88. **Ivo Müürsepp**. Robust Detectors for Cognitive Radio. 2013.

89. **Jaas Ježov**. Pressure sensitive lateral line for underwater robot. 2013.

90. **Vadim Kaparin**. Transformation of Nonlinear State Equations into Observer Form. 2013.

92. **Reeno Reeder**. Development and Optimisation of Modelling Methods and Algorithms for Terahertz Range Radiation Sources Based on Quantum Well Heterostructures. 2014.

93. **Ants Koel**. GaAs and SiC Semiconductor Materials Based Power Structures: Static and Dynamic Behavior Analysis. 2014.

94. **Jaan Übi**. Methods for Coopetition and Retention Analysis: An Application to University Management. 2014.

95. **Innokenti Sobolev**. Hyperspectral Data Processing and Interpretation in Remote Sensing Based on Laser-Induced Fluorescence Method. 2014.

96. **Jana Toompuu**. Investigation of the Specific Deep Levels in *p*-, *i*- and *n*-Regions of GaAs p^+-*pin*-n^+ Structures. 2014.

97. **Taavi Salumäe**. Flow-Sensitive Robotic Fish: From Concept to Experiments. 2015.

98. **Yar Muhammad**. A Parametric Framework for Modelling of Bioelectrical Signals. 2015.

99. **Ago Mõlder**. Image Processing Solutions for Precise Road Profile Measurement Systems. 2015.