TALLINN UNIVERSITY OF TECHNOLOGY
Faculty of Information Technology

Martin Karu 152905IABM

# WEAKLY SUPERVISED TRAINING OF SPEAKER IDENTIFICATION MODELS

Master's thesis

Supervisor: Tanel Alumäe

Senior Researcher

Tallinn 2017

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Martin Karu 152905IABM

# KAUDSE JUHENDAMISEGA KÕNELEJATUVASTUSE MUDELITE TREENIMINE

Magistritöö

Juhendaja: Tanel Alumäe

Vanemteadur

Tallinn 2017

## Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Martin Karu

08.05.2017

# Abstract

The thesis studies deep neural networks and machine learning for training speaker identification models. This enables identifying speakers solely based on the characteristics of their voice. Usually, these models are trained on large amounts of discretely annotated audio data, but creating such datasets is time-consuming.

We propose a novel approach based on weakly supervised learning for training speaker identification models. Each audio recording in the training data is annotated only by a set of speakers. Alignments between speaker labels and speech segments are not provided. The audio is pre-processed with speaker diarization and i-vector extraction. The model is trained with backpropagation and label regularization as the cost function – a novelty in neural networks. The method compares the average prior probabilities of annotated data with the posterior probabilities output by the model. The process is validated for recall and precision with a random held-out set of audio files not used in the training process.

Our experiments on the dataset from Estonian Public Broadcasting archive prove that weakly supervised training is a highly accurate method for identifying those speakers who occurred several times in the training data.

This thesis is written in English, is 61 pages long and contains 8 chapters, 13 figures, 7 equations, and 10 tables.

## Annotatsioon

## KAUDSE JUHENDAMISEGA KÕNELEJATUVASTUSE MUDELITE TREENIMINE

Magistritöö eesmärgiks on uurida süvanärvivõrkude ja masinõppe kasutamist kõnelejatuvastuse mudelite treenimiseks. Tegemist on tehnoloogiaga, mis võimaldab tuvastada helifailis kõnelevaid inimesi hääle karakteristika alusel. Tavaliselt kasutatakse kõnelejamudelite treenimiseks suuri andmehulkasid, kus iga kõnesegmendi kohta on teada tegelik kõneleja. Taoliste andmete kogumine on aja- ja ressursimahukas.

Töös rakendatakse uudset lähenemist, kus kõnelejate tuvastamiseks kasutatakse kaudse juhendamise abil treenitud mudeleid. Iga salvestise kohta on teada ainult esinejate nimekiri. Seosed kõnelejate nimede ja kõnesegmentide vahel puuduvad. Enne kõnetuvastuse mudeli rakendamist töödeldakse helifaile. Süsteem jaotab esmalt helifaili lühikesteks osadeks, märgistab kõik kõne sisaldavad segmendid ning koondab jaotised vastavalt tuvastatud kõnelejate arvule kobaratesse. Seejärel kirjeldab süsteem helisegmentide põhjal kõnelejate hääle omadused 600-mõõtmelise i-vektorina.

Mudelit treenitakse tagasisidestuse kaudu ning kuluvõrrandina kasutatakse närvivõrkude vaatenurgast uudset lähenemist: tähistuste korrapärastamist (*label regularization*). Tegemist on meetodiga, mis võrdleb eelnevate ja tagumiste tõenäosuste aritmeetilisi keskmisi – erinevuste korral muudetakse mudeli kihtide ja sisendite kaalusid. Eelnevateks tõenäosusteks on saate andmetes märgitud inimesed ja tagumisteks mudeli poolt pakutud väärtused.

Tulemusi valideeritakse suvaliselt valitud raadiosaadete abil, mis ei esinenud treeningandmetes. Helifailidest luuakse kõnelejate i-vektorid. i-Vektoritele märgitakse esinejate nimed kahel moel: kõnelejatuvastuse mudeli kaudu ning käsitsi. Hinnatakse nii saagist – mitmele kõnelejale oskas süsteem nime lisada – kui ka täpsust ehk kõnelejate osakaal, kes on õigesti tuvastatud.

Katsetused, mis viidi läbi Eesti Rahvusringhäälingu „Päevakaja“ uudiste arhiivi salvestustega, kinnitavad, et kaudne kõnetuvastuse mudelite juhendamine on sobiv meetod varasemalt treeningandmetes esinenud inimeste tuvastamiseks.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 61 leheküljel, 8 peatükki, 13 joonist, 7 valemit ja 10 tabelit.

# List of abbreviations and terms

| | |
|---|---|
| AI | Artificial Intelligence |
| BIC | Bayesian Information Criterion |
| CLR | Conditional Logistic Regression |
| DNN | Deep Neural Network |
| ID | Unique Identifier |
| HMM | Hidden Markov Model |
| LIUM | Laboratoire d'Informatique de l'Université du Maine<br>Research Laboratory of the University of Maine |
| ReLU | Rectified Linear Units |
| TUT | Tallinn University of Technology |

# Table of Contents

## List of Figures

# List of Tables

# List of Equations

# 1 Introduction

This thesis investigates training speaker identification models in the context of weakly supervised machine learning. Our model is trained with thousands of recordings and accompanied metadata from the Estonian Public Broadcasting news archive. We use a deep neural network that is trained using label regularization as the cost function.

The chapter describes the approaches and methodology used in the thesis and the novelty of them. It outlines the basic differences between supervised and weakly supervised learning and the occasions of using either. We define the problem, set a hypothesis for solving it and set target goals to validate the constructed model.

## 1.1 Problem

The thesis investigates two unexplored areas of machine learning: firstly, using weakly supervised learning in the context of speaker identification, and secondly, implementing label regularization as the cost function in a neural network.

Up until now, most scientific research in the area of machine learning has been done with either supervised or unsupervised learning. The first methodology requires discretely labelled data for training, but obtaining it is time- and otherwise resource consuming. The unsupervised learning uses unlabelled data, but it usually involves even larger datasets for correctly training the model. Weakly supervised learning is more closely related to supervised learning with the availability of somewhat labelled data. The approach the thesis endeavours includes combining the data across thousands of audio recordings to create a model for correctly identifying the speakers.

The second problem encompasses the way a neural network model for speaker identification needs to be trained. Relations must be formed between the audio segment representations and the list of annotated speakers that correspond to them. To achieve this, we propose using a technique called label regularization. In typical label regularization applications, the systems are trained with both strongly labelled and unlabelled data, which can originate from different domains. In the scope of the thesis, we task label regularization to match the average prior probabilities of the metadata labels to the average posterior probabilities given by the speaker identification model.

## 1.2 Objective

We seek to create a weakly supervised neural network model, train it with audio recordings and metadata from over 6600 news recordings from the Estonian Public Broadcasting archives in order to correctly identify speakers in new recordings.

The results are validated in two aspects: recall – the proportion of audio segments that have been labelled – and precision – the proportion of labels that are correct. This is accomplished using a random held-out set of audio files that was not present in the training. The labels assigned by the model are compared against manually annotated speaker labels.

The goal is to achieve a result of 70% on recall and 90% precision. A confidence threshold is used to retain only the correct predictions and accomplish the set objectives.

## 1.3 Methodology

The main speaker identification model is implemented as a weakly supervised deep neural network – an artificial neural network, where deep learning is applied.

In training, the inputs include a fixed-dimensional representation of speaker audio segment clusters called i-vectors, and a set of annotated speakers for each recording. When using the model to identify speakers in new shows, the list of speakers is not used and instead a confidence threshold is introduced. This is used to eliminate predictions that are likely to be incorrect.

The model is trained using backpropagation. The cost function is implemented as label regularization. The method encourages average model predictions within each show to match label priors based on the annotated speakers in given show. This is in contrast to typical supervised learning, where backpropagation is implemented by comparing strong labels against the model's posterior probabilities. The method is easy to implement and scales well to large amounts of training data.

## 1.4 Outline

The first chapter describes the approaches and methodology used in the thesis and the novelty of them. It introduces the basic differences between supervised and weakly supervised learning and the occasions of using either. We define the problem, set a hypothesis for solving it and set target goals to validate the constructed model.

The second chapter gives a detailed overview of the software design patterns used in the thesis, e.g. DNNs, speaker diarization, i-vector extraction and label regularization. As well, the underlying mathematical logic is depicted in the amount that is necessary to understand the technical applications.

The third chapter analyses current solutions that are related to our problem in some aspects. None of the solutions deal with either of the two problems postulated in "1.1 Problem".

The fourth chapter looks at the composition and statistical aspects of the data used in the identification experiments. It is collected from Estonian Public Broadcasting radio news shows called "Päevakaja". The data consists of two parts: audio recordings and the metadata related to them.

The fifth chapter specifies the solution part of the thesis. It includes a detailed explanation of both the training and speaker identification processes. The data pre-processing and post-filtering is explained.

This sixth chapter validate whether the objectives of the thesis have been met. That is: the model and its training is sufficient to correctly label new show's speakers. A random held-out set of recordings is used and after processing it through speaker identification, it is compared against manually labelled data.

This seventh chapter analyses how to further improve the speaker identification system. It gives examples, where the model can be used right away and possibilities of implementing the findings of the thesis in similar research topics.

The eighth chapter draws conclusions on whether weakly labelled data and label regularization are suitable for machine learning. The objectives and hypotheses are revisited and analysed whether they were achieved as planned.

# 2 Background Theory

This chapter gives a detailed overview of the software design patterns used in the thesis, e.g. DNNs, speaker diarization, i-vector extraction and label regularization. As well, the underlying mathematical logic is depicted in the amount that is necessary to understand the technical applications.

## 2.1 Algorithms

### 2.1.1 Hidden Markov Models (HMM)

Hidden Markov Models (HMM) are used in almost all the current speech and speaker recognition algorithms, computer vision (i.e. image and video recognition), in data compression, and in other areas of AI.

A hidden Markov model is a tool for representing the probability distributions over sequences of observations (Ghahramani, 2001). The observation at time $t$ is defined as $Y_t$. The result is a discrete alphanumeric value or other object as long as its probability distribution over time can be defined.

HMMs are defined by two main properties. Firstly, the observation $Y_t$ done at a time $t$ upon a process $P$ produces a state $S_t$, which is hidden from the observer. Secondly, it assumes that the state of this hidden process satisfies the Markov property – given the value of $S_{t-1}$, the current state $S_t$ is independent of all the states prior to $t - 1$. This means that the state at any time contains all information necessary for predicting the future of the process. The definition of the HMM also assumes that any state is discrete – this is sometimes defined as the third characterizing property.

### 2.1.2 Universal Background Model (UBM)

A Universal Background Model (UBM) is a model used in biometric (e.g. speaker) recognition and verification systems. It comprises of general person-independent characteristics' definitions and compares them against person-specific feature characteristics' models (Reynolds, 2009).

### 2.1.3 Gaussian Mixture Model (GMM)

A mixture model in statistics is a probabilistic model that represents sub-populations within the whole population. It allows to define sub-populations based on fixed or known parameters with unknown or varying parameters. Gaussian mixture models assume all data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters.

Mixture models are used for making statistical assumptions about individual observations based on postulated sub-population attributes. This behaviour makes GMMs easy to be used in unsupervised or clustering procedures (Greenberg, 2014).

## 2.2 Speaker diarization

Speaker Diarization is the process of deciding who spoke when in an audio stream (Vijayasenan, 2009). The process involves determining the number of speakers and splitting the audio stream into acoustically homogeneous segments (Madikeri, 2012). The results contain the number of speakers and a set of audio segment clusters for each of them. These values are extracted in an unsupervised manner. Speaker diarization can be applied to several types of audio streams such as news recordings, conversational telephone speech and meeting recordings.

The process starts with feature extraction, which builds derived values from initial data. Next, the speech activity detection process is applied. The parts in which speech is present or absent are determined. After isolating speech, the system looks for speaker changes and cuts the audio into 1-2-second-long uniform segments. These are considered homogeneous because of their short length.

The segments are linked together per similarity measures. Agglomerative bottom-up clustering of acoustic segments is used. More precisely, Agglomerative Information Bottleneck bases the clustering upon information theoretic principles. Once the clusters have been found, their boundaries are refined using an ergodic Hidden Markov Model (HMM) with duration constraints.

In most state-of-the-art models Bayesian Information Criterion (BIC) is used as the complexity metric, but other similar metrics with equally viable stopping criterion are available. Given the unsupervised nature of the linking – the final number of clusters is unknown; the stopping criterion is often considered arbitrary to the outcome. Thus, selecting any of the often-used stopping criterion should achieve desired results.

Current speaker diarization systems require too much time or computing power to be used in regular settings. Research is being done on faster-than-real-time diarization systems with low computational complexity. These can be used in many cases, such as mapping meeting notes to the speakers during the meeting. The processing power required is satisfied with a common desktop computer or a high-end laptop.

## 2.3 i-Vectors

i-Vector approach is used in state-of-the-art speaker recognition systems. The process includes multiple steps. G. Greenberg describes it in detail in the paper “Speaker Recognition i-Vector Machine Learning Challenge” – see Figure 1.

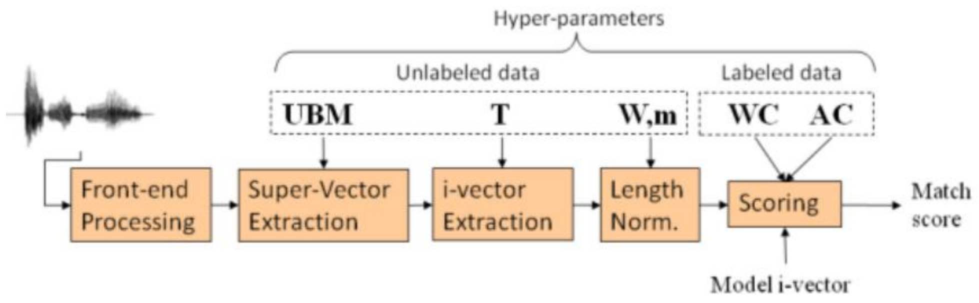

Figure 1 Simplified Block Diagram of i-Vector Extraction and Scoring (Greenberg, 2014)

Speech activity detection is run on an audio segment such as a radio show or a copy of a telephone call. The audio is processed and locations of speech in it are located. The acoustic features that convey information about the speaker are extracted. This creates a sequence of feature vectors – typically mel-frequency cepstra (short-term power spectrum of a sound) at 100 feature vectors per second (Greenberg, 2014).

In speaker verification systems, a Universal Background Model (UBM) is used to identify the distribution of feature vector sequences. In this case, it is implemented as a Gaussian mixture model (GMM). The model is trained with speech samples from a large set of speakers in order to represent general speech characteristics. The sequence of parameters' distribution is represented as a relative to the UBM.

The parameters of the speaker's characteristics' are transformed. Firstly, using a total variability matrix (T) to a 600-dimension vector. Secondly, by whitening the vector – a global mean (m) is subtracted from the values, and then scaled by the inverse square root of a global covariance matrix (W) – see Figure 1. Lastly, the vector is then normalized to unit length. These transformations create a more suitable vector for the GMM to process.

Usually, as the last step, a score between the model and a sample i-vector is computed. Most likely, a cosine distance, but other similar methods can be used. This is employed in supervised learning manner – the vectors are sent directly into the DNN and backpropagated uniquely.

The segments after the initial front-end processing are known as the system's hyper-parameters. Before using the model, it must be trained. The model represents statistical variance and distribution of general features. Thus, we can train it using unlabelled data.

The resulting i-vectors are input directly to the DNN – both in supervised and weakly supervised learning.

## 2.4 Deep Neural Network (DNN)

Deep Neural Networks (DNN) extend Artificial Neural Networks (ANN) to find complex non-linear relationships by incorporating Deep Learning (DL). Before this, most machine learning techniques exploited shallow-structured architecture ANNs. These typically contained one or two layers of non-linear feature transformations (Li, Dong, 2014). DNNs extend the traditional classification models by having a larger capacity for learning and finding relations. This is managed through multiple hidden layers between the input and output layers (Szegedy, 2013). Added layers enable composing features from lower layers, which allows processing complex data. Compared to similarly configured shallow ANNs, the complex models use fewer units and are better optimized. (Bengio, 2009)

DNNs and deep learning combine the research areas of neural networks, artificial intelligence, graphical modelling, optimization, pattern recognition, and signal processing. Deep learning methods are used at an increasing level to exploit complex, compositional nonlinear functions. Compared to ANNs the main reasons behind the ability to learn distributed and hierarchical feature representations are:

1) Hardware implementations that drastically increase chip processing abilities. These are put to practice in general-purpose graphical processing units (GPGPU);
2) The ability to use larger – both labelled and unlabelled – datasets for training;
3) The recent advances in machine learning and signal/information processing research (Li, 2014).

An observation can be represented in multiple forms: i.e. an audio file can be represented by its length, waveform, or the transcribed text. Similarly, an image can be described by the collection of pixels, a colormap or other means. The purpose of Deep Learning is to use multiple levels of representation and abstraction to make sense of data (Deep Learning Tutorials, 5th of May 2017). The most suitable method for representing data and the layers of abstractions that produce the required results can be chosen by trial and error. It is possible to combine different methods, models and data representations.

Feed-forward DNNs are mostly trained with a backpropagation algorithm. The goal is to minimize the error rate. The network is initialized with randomly chosen weights. A cost function is applied: the model's output values are compared to the expected values, a penalty is calculated and the model's weight features are altered. The backpropagation algorithm can be selected based on the type of data used (Rojas, 1996). In 1986, Rumelhart et al. showed with experiments that backpropagation could create complex internal representations of the incoming data in the hidden layers of neural networks (Schmidhuber, 2014).

The model used in the thesis is implemented as a type of artificial neural models: a feedforward network. The input layers accept the data feed data to the strictly ordered hidden layers. The layers in the model are connected consecutively in a defined order and no cycles are created. Lastly, the processed data reaches the output: Softmax layer. This defines posterior probabilities for speakers.

### 2.4.1 Softmax Layer

The Softmax function is used as the ultimate layer in the system. The Softmax function insures that the outputs are positive and their sum is one. Because we are dealing with probabilities, it is interpreted as 100% and allows the DNN to interpret the output values more linearly. (Collobert, 2008).

### 2.4.2 Dense Layers

The neural networks' layers consist of different number of units. In feed-forward networks, each node can send the data to some or all the units in the next layer. If there are few connections, the layer is called sparse. If every unit is connected to every unit in the next layer, the layer is called dense.

### 2.4.3 ReLU Activation Function

The dense layers output is defined by the set of inputs. This is otherwise known as an activation function. The thesis uses rectified linear units (ReLU) method for activation, mostly because of two aspects. Firstly, it is able to create more generalized internal relations, and secondly, ReLU does not use division or exponential calculations. Thus, it is faster than other similar functions.

### 2.4.4 Dropout Layers

Deep neural networks can accomplish complex goals and are powerful machine learning systems. One of the greatest problems they face is overfitting. This means that a model optimizes its algorithms to minimize the error – for exactly the data that is used for training it. Afterward, when it is used for classifying new values, the error rate is higher than expected.

Dropout is a technique that modifies the DNN and during training randomly removes units or their connections with nodes in the following layer. The randomness of the dormant units creates abstract weights that satisfy a broader set of input data (Srivastava, 2014).

## 2.5 Label Regularization – Weakly Supervised Learning

Supervised machine learning methods require manually labelled training data. The demand for strongly labelled data limits the applicability of machine learning: either a small dataset can be utilized or a lot of time and resources must be invested. Therefore, applying machine learning in the context of weakly labelled data has drawn considerable amount of attention in recent years.

In fully supervised learning, each set of inputs is uniquely mapped to an expected (correct) output. Weakly supervised learning combines a set of inputs (or a list of input sets) with a list of possible values. The third option would be to use unlabelled data, which can cluster the i-vectors, but not label them at all. For example, if an i-vector is processed by a DNN: with strongly labelled, the cost function can simply compare the expected outcome with the label provided and adjust weights accordingly. Weakly labelled approach can validate that the outcome is in the list of possible labels. Unlabelled data would be clustered by the model and to start adding labels, it should be combined with a smaller list of supervised or semi-supervised data. The initial size of the dataset for unlabelled approach is exceedingly larger than with the other two types.

If the DNN were tasked with computer vision and the dataset were a collection of images. Fully supervised systems could be used to identify singular items on them, i.e. a photo of a baseball, a photo of a basketball etc. Similarly, self-taught systems with unlabelled data can be used to cluster the images, most likely containing a single item in them. Weakly supervised systems would instead be able to identify images with multiple objects based on statistical occurrences. For example, the images for training can be "hat and gloves", "hat and shoes", "gloves, shoes and scarf", "hat and belt" and so forth. As visible, the object "hat" exists in three cases with other varying items. Not once is the object defined as a unique label. Subsequently, when using the system to identify images, object "hat" should be correctly labelled. The probability for correctly classifying inputs – using weakly labelled data – is described in further chapters. (Stanford UFLDL Tutorial, 2017)

In our task, weak labels originate from the human-provided metadata for each show. The metadata lists the identities of all the speakers appearing in the show. However, precise timing information (i.e. who speaks when) is not given. This allows us to cast our task to a weakly labelled learning problem: using speaker diarization, we can find the likely partitioning of speech segments based on the speakers; then, we calculate an i-vector for each such partition. Ideally, the number of i-vectors is equal to the number of speaker identities provided in the metadata, and we know that each i-vector corresponds to exactly one of the speaker identities, but of course, we do not know the exact mapping from i-vectors to speaker identities. Our task is to train a model that learns the true mapping, given many such weak mappings.

We propose to solve our weakly supervised learning problem using a technique called label regularization. Label regularization is a special case of a more general method called expectation regularization (Druck et al., 2007), originally proposed for semi-supervised learning. In label regularization, the discriminative machine learning model is trained using a certain amount of strongly labelled data, and a certain amount of unlabelled data, possibly coming from another domain. However, the prior probabilities of the labels of unlabelled data are given, and the task of label regularization is to match the average posterior probabilities of the model on unsupervised data with the given prior probabilities. This is implemented using an additional term for the training objective function that characterizes the difference between the given prior probability and model's average posterior probability over training data (Equation 1):

$$D(\tilde{p}||p_\theta)$$

Equation 1 Difference Between Prior and Posterior Probabilities

The proposed distance metric is the Kullback–Leibler divergence – Equation 2 (Kullback, Leibler 1951):

$$D(\tilde{p}||p_\theta) = \sum_y \tilde{p} \log \frac{\tilde{p}}{\hat{p}_\theta}$$

Equation 2 Distance of Prior and Posterior Probabilities as KL-divergence

In our case, we not only know the prior probabilities of the labels (speaker identities) of our whole training set. We we also know the prior probabilities over groups of examples, corresponding to the i-vectors of each individual show. The prior probabilities corresponding to the speakers that appear in the show is equal to $1/N_s$ where $N_s$ is the number of distinct speakers in show $s$ (since the speaker identity should be assigned to one and only one i-vector of the show). It is equal to zero for all other speakers (Equation 4). That is, our objective function $l$ is a sum of the KL-divergences of the known prior and predicted posterior probabilities over all shows in our training set (Equation 3):

$$l = \sum_{y} D(\tilde{p}_s || p_{\theta_s})$$

Equation 3 The Sum of KL-divergences

Where

$$\tilde{p}_s(speaker) = \begin{cases} \frac{1}{N_s}\ if\ speaker\ occurs\ in\ show\ s \\ 0\ otherwise \end{cases}$$

Equation 4 Prior Probabilities for Speakers

In practice, this training should only work if the speakers occur in multiple shows, and no speakers occur always in the same shows together. If the same two or more speakers are present in the same shows, the identification model gives an equal probability for their labels. As explained in “5.1.2 Using the Model for Labelling Speakers“, the results are refined using a confidence threshold. Therefore, for two similar speakers, if the threshold is set above 50%, both are invalidated.

# 3 Previous Implementations for Stated Problem

The problem of using weak supervision with speaker identification, as described in this thesis, has not been investigated in any earlier research to the best of the Author's knowledge.

The chapter describes a few previous works that are related to our problem in some aspects. None of the papers cover the problems stated in the thesis.

## 3.1 The 2015 Sheffield System for Longitudinal Diarisation of Broadcast Media

Broadcast media audio stream diarization is a particularly difficult task, due to the large number of speakers and additional background noises. Longitudinal diarization is a type of speaker diarization executed across a collection of connected audio streams. Using information gathered from previous shows within a series improves the performance of the model and the outcome, but it requires ways of matching speakers across consecutive files.

Longitudinal Speaker Diarisation System by the University of Sheffield (Milner, 2015) was created for participation in the 2015 Multi–Genre Broadcast (MGB) challenge and it used data from BBC archives. The system constructed consists of three main stages:

1) Speech Activity Detection using a DNN;
2) Audio segmentation and clustering by speakers with DNN-based models;
3) Speaker linking across shows.

The system was validated on a set of 19 shows from five different TV series. The result provided a Diarization Error Rate (DER) of around 50%.

The paper by the University of Sheffield is similar to our thesis until the Speaker Identification part. The main emphasis on the paper was upon improving the speaker DER. In our case, an existing diarization implementation is used on each single audio stream, i-vectors are extracted and the linking is handled in order to connect labels to the speakers' clusters.

## 3.2 Semi-Supervised Learning for Transductive Speaker Annotation

The problem of annotating speech with speaker information is imposed and validated using transductive learning. The paper postulates that this approach is optimal in many cases, archived recordings as a prime example (Täckström, 2008).

The thesis proposes using a simple semi-supervised algorithm. It proves that when using a very small number of labelled segments for training, the transductive model outperforms the inductive fully supervised Support Vector Machine algorithm. When increasing the sample size, the models' performance is on par.

Täckström's paper is similar to ours as it proves a low sample of labelled training data is able to give the correct answer. It differs because the training data is still supervised meaning each segment is discretely linked to a specific speaker.

## 3.3 Automatic Named Identification of Speakers Using Diarization and ASR Systems

Automatic named identification considers extracting the name of the speaker from their own speech. The audio stream is processed using speaker diarization, creating linked utterances belonging to the speakers present in the recording. Speech recognition is ran on the utterances to have a searchable text corpus. As the final step, the algorithm searches the text for speaker names – if present, the speech segments are linked to the name found. The algorithm relies on two possibilities: the speakers introduce themselves or the host does it for them (Jousse, 2009).

Speaker Diarization System was experimented on French broadcast news recordings from the ESTER 1 evaluation campaign. All the data was provided automatically – diarization, speech-to-text and even the names of the speakers. The result was a two-fold increase in speaker identification than without using the automatic named identification model.

The paper focuses on naming a single recording's diarized audio segments and does not extend the information – either the speaker models or the names – to neighbouring audio streams.

## 3.4 Unsupervised Speaker Identification Using Overlaid Texts in TV Broadcast

Similarly, to "Automatic Named Identification of Speakers Using Diarization and ASR Systems" this paper proposes their approach on finding the name of the speaker automatically from text related to the audio stream. In this case, instead of the speaker name being vocally presented, the information is obtained from imagery. The video is processed and any overlaid text is indexed, i.e. the presenters' and guests' names and titles are displayed at the beginning of their appearance (Poignant, 2012).

The audio streams are processed using speaker diarization and the data is linked to the indexed information. The model considered the co-occurrence of two or more people – the names of partners or otherwise connected people are often displayed simultaneously.

The best un-supervised system reached a weighted harmonic mean of 70.2% and 81.7% if the presenters of the show were left out. As a comparison, a supervised speaker identification system with 535 trained speaker models was used providing results of 57.5% and 45.7% respectively.

The paper describes using unsupervised training for identifying speakers with external methods for gathering speaker name data. This method focuses on identifying parties in a single recording thus it would be possible to even further improve the outcome if it were combined with the approach proposed in this thesis.

# 4 Data

This chapter describes the data used in the identification experiments. It is collected from Estonian Public Broadcasting radio news shows called “Päevakaja”. The data consists of two parts: audio recordings and the metadata related to them. The subsections give an overview of the statistical aspects of both i-vectors and speaker lists. Processing the data is described in “5. Solution”.

## 4.1 Audio Files

Audio recordings are an integral part of speaker identification – the process starts from them. The shows consist mostly of a single person speaking at any time, making data pre-processing simpler. There is no need to separate audio waveforms before speaker diarization.

Shows are usually in Estonian, but they often contain foreign news pieces. Thankfully, the process is impervious to it anyway – speech is represented by i-vectors and there is no indication of words spoken or languages used.

## 4.2 Metadata

Information is collected to label the speakers after processing the audio form. The knowledge that we gather consists of two parts: an identifier for the show (e.g. URL or an arbitrary sequence number) and the names of speakers in it.

Names of the speakers are expressed in a uniform way and represented as “[Last name] [First name]” – also present in the validation data shown in “Appendix 1 – Validation”. During training, a list of distinct values is maintained. By eliminating variability in the metadata, relations between different shows are simpler to define. The name is attached as a label to the person’s audio segments’ cluster – the grouping receives the speaker’s name. Afterwards, when using the DNN to identify speakers, the label of the most similar cluster is set as the most probable name for the new speaker’s recording.

Speakers can be either presenters or guests of the show discussing a relevant topic or reporters and people connected to the subject at hand. The largest clusters, as demonstrated with upcoming statistics, are created by the people working in Estonian Public Broadcasting corporation and producing the shows.

## 4.3 Statistics

### 4.3.1 Total Number of Shows Used

The number of shows since the first airing is slightly over 6800. In the last 5 years, on average, one show is added daily.

The training data was extracted in the end of November 2016. At that, the number of shows aired was 6604. The metadata stated that 4366 of them contain more than two speakers. Thus, 33.8% of the shows cannot be used for training because they contain too little information.

The i-vectors extraction is more reliable with only 50 shows with fewer than two speakers.

### 4.3.2 Dates of Broadcasts – Years and Amounts

“Päevakaja” show has been on air since 1944, but the number of shows recorded annually varies.

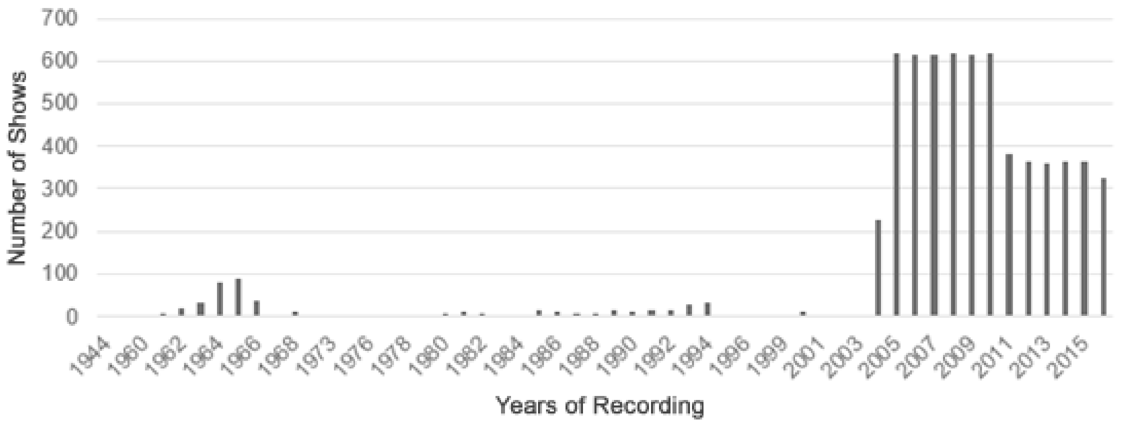

Figure 2 Number of Shows Annually

Figure 2 displays the years the show has been aired. The values represent the total number of shows per year. The earliest were recorded in 1944 and a slight rise in the number set in the 1960s. Shows since 2004 make up 92% of all data.

The period from 2005 to 2010 had close to two recordings per day, since 2011 the Estonian Public Broadcasting has aired approximately one show per day. The slight decline at the end of 2016 is due to the training data being extracted on the 23rd of November – about 40 shows from 2016 were not used.

### 4.3.3 Show Length

The shows deliver daily news reports and commentary from people involved. Most of the shows range from 10 to 24 minutes in length. A small percentage of recordings are less than 5 minutes or more than 25 minutes long.

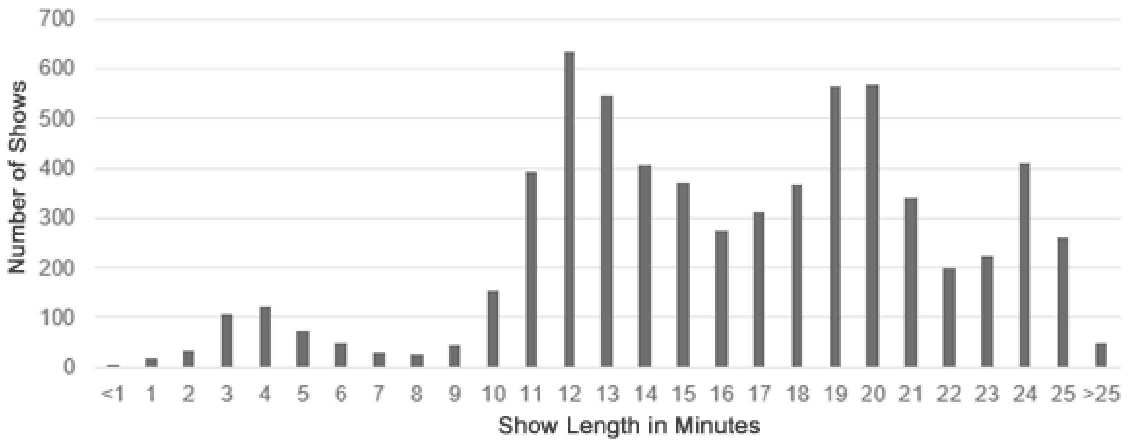

Figure 3 Length of Shows (in Minutes) with the Number of Occurrences

The Figure 3 displays that the length of the 6604 shows has fluctuated. The x-axis contains audio stream lengths rounded down to the nearest minute value. The values correspond to the number of shows with given length.

The average show length is 16 minutes and 47 seconds.

### 4.3.4 Average Show Length across Recording Years

It is possible to graph the distribution of show length across the years.

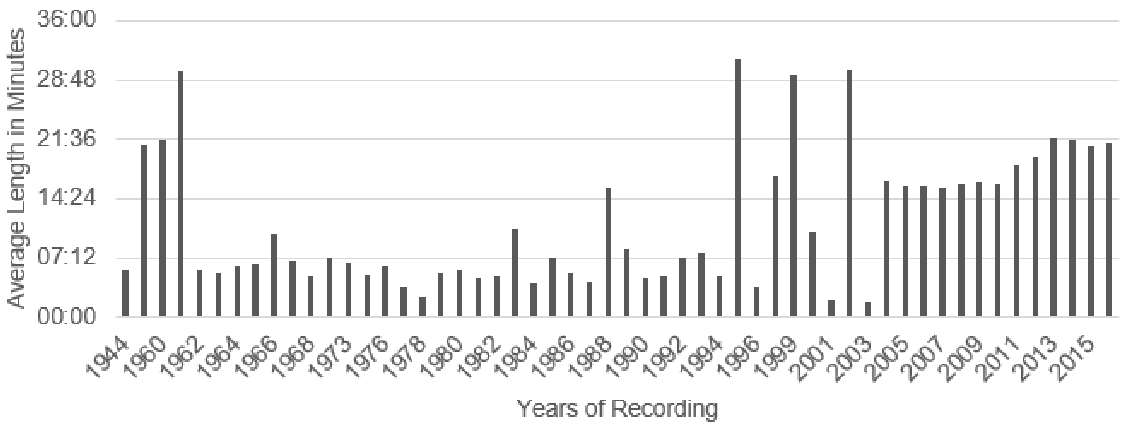

Figure 4 Average Show Length Annually

Instead of displaying the total number of shows with any given length, Figure 4 contains recording years left-to-right and the average show lengths as values.

Until the mid-2000s, the majority of shows were under 5 minutes. Since 2005, the average has gone up to 17 minutes and 39 seconds. The highest arithmetic mean can be found in 1995. An average show length of 31 minutes and 19 seconds, but in fact, the year only contained a single show. Similarly, in 2002 and 1999, there are high averages, but only two and three shows recorded respectively.

### 4.3.5 Speaker and Vector Count in Shows

One of the most important aspects using a DNN is the amount of data available for training it.

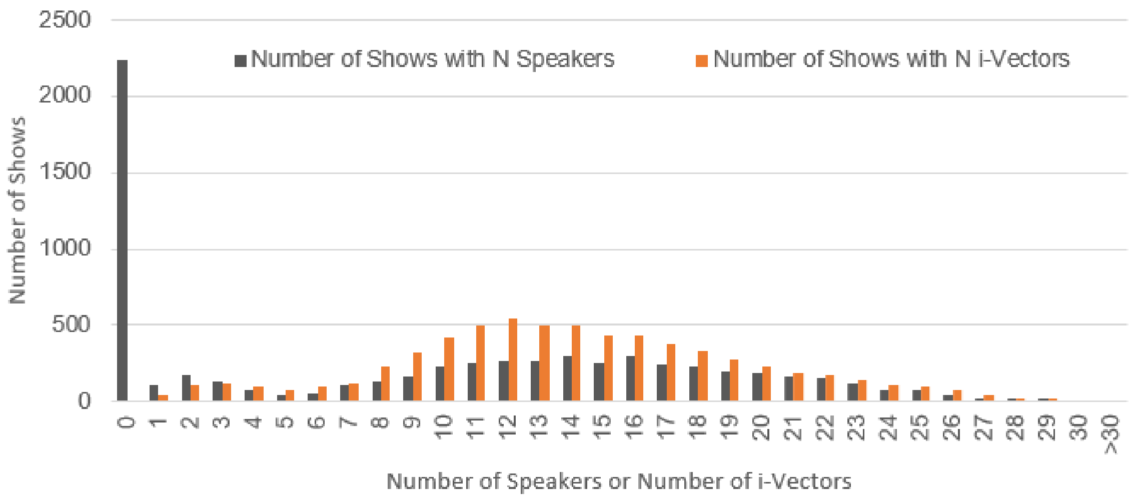

Figure 5 Number of Shows with N Speakers Overlaid with Number of Shows with N i-Vector

Figure 5 represents two values: firstly, the number of speakers annotated in the show's metadata (displayed in dark grey) and, secondly, the number of i-vectors extracted from the recording's audio stream (displayed in orange hue). The x-axis is the number of speakers and/or i-vectors; the y-axis - the number of shows with given parameters.

There is a noticeable spike at the beginning of Figure 5. The number of shows with zero speakers is 2237 – a total of 33.8% of all shows. Additionally, 1.6% of shows have exactly one speaker annotated in the metadata.

The i-vector values are more equally distributed. The number of shows with no i-vectors is eight – forming only 0.1% of the dataset – and only one i-vector in 42 shows (0.6%).

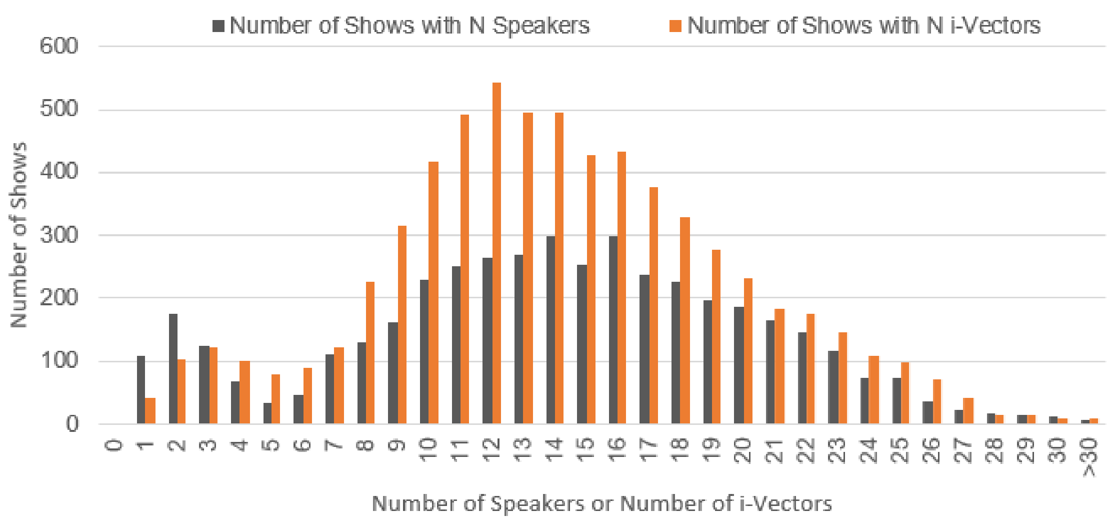

Figure 6 Number of Shows with N Speakers Overlaid with Number of Shows with N i-Vectors (without 0-values)

After removing the shows with zero speakers or i-vectors, the distribution is more distinguishable in Figure 6.

First, the number of shows with 6 to 27 i-vectors always exceeds the number of shows with 6 to 27 speakers. This means that even though metadata contained a great deal of shows with zero speakers, the actual audio recordings involve presenters and multiple guests.

Second, the i-vector and the speakers' distributions are similar when the 0-values are removed. The arithmetic mean over 6596 shows is 14.1 i-vectors per show and 14.0 speakers per show over 4367 values. With the zero-values included, the i-vector average does not shift – still 14.1 i-vectors per show. The speakers per show average lowers to 9.2.

### 4.3.6 Derived Metrics for Speakers and i-Vectors

Per the metadata:

- On average, each show has 14 speakers;
- 110 shows contain a single speaker.

- There is 185 shows with 25 or more speakers. Out of these 17 recordings, contain 30 or more speakers.

- The greatest number of guests was present on the 31st of December in 2004. It was the New Year's Eve show with 45 speakers. The second most visited recording is from The New Year's Eve in 2002 with 35 guests.

### 4.3.7 Average Number of Speakers and i-Vectors per Show Annually

Recordings originate from a long period. This allows us to graph the average number of speakers in a show per year and the average number of i-vectors per show per year. Figure 7 overlays these values.

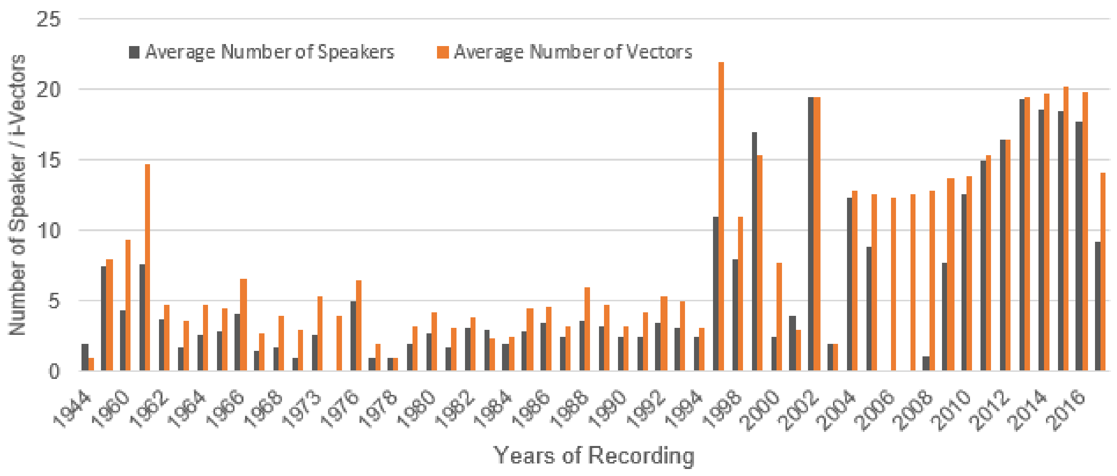

Figure 7 Average Speaker and i-Vector Count per Show by Year

In the beginning of the 1960s, start of 2000s and shows since June 2009 have been annotated diligently. The Figure 7 has the years since the first recording up until the end of 2016. The values display the number of either i-vectors or speakers on average in a show for every year.

In the period from 1965 to 1994, the show was mostly on hold. This is displayed on the Figure 7 - the period that has continuously the least number of both speakers and i-vectors.

Most shows have 8 to 20 i-vectors. The amount of i-vectors is almost always higher or on par with the speakers' value. The only exceptions are years 1944, 1983, and 2001.

This result is expected. The number of i-vectors in a show compared to speakers (by metadata) is greater because of two main reasons: firstly, metadata is unreliable, and, secondly, one person's voice can be counted as two (or in some cases even 3) separate ones.

In the years from 2006 to 2008, we can see that the average number of i-vectors remains steadily around 12. However, the average number of speakers per show drops. This means that the shows were continuously recorded in the years in question, but metadata annotation was forsaken.

Clustering one person as multiple happens if there are different background conditions. The diarization diverges enough to count one person's voice as multiple ones if for example:

1) The presenter starts talking when the Estonian Public Broadcasting show's jingle is still being played;
2) The presenter talks normally into the microphone;
3) The presenter has recorded a news segment beforehand, e.g. outside the (noise cancelling) studio with wind whistle or passing traffic noises.

The model must take into account these possibilities and, if necessary, within one show label multiple clusters with the same person's name.

### 4.3.8 Number of Occurrences for Speakers

Most people are invited to only one show. So much so, that Figure 8, which is used to display the number of occurrences for speakers, must display the values in log-scale. The x-axis displays the unique number of times people have been to the show. The y-axis graphs the number of people that have been annotated as many times in the metadata.

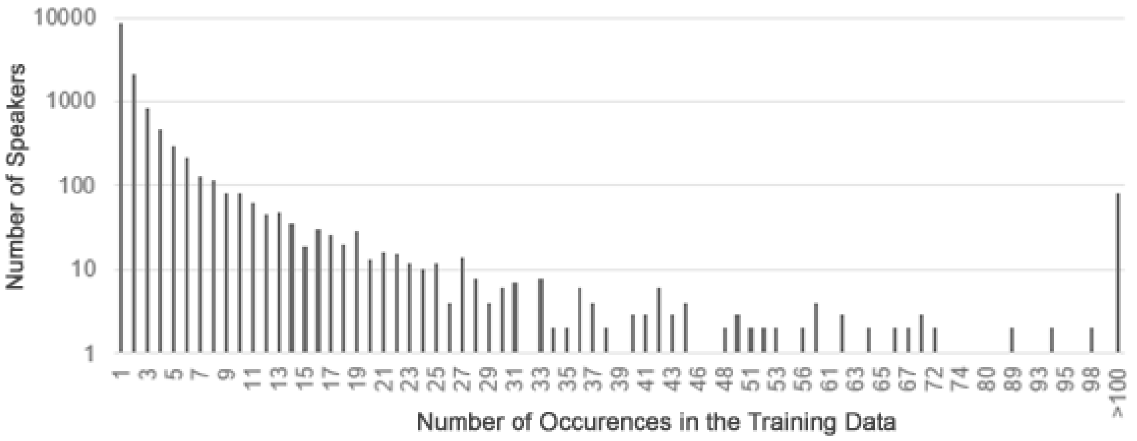

Figure 8 Total Number of Occurrences for Distinct Speakers in Training Metadata (in log-scale)

The total number of unique speakers in the training data is 13770. Out of all, 63.8% are described in only one show and additionally 15.5% in two shows.

### 4.3.9 Most Frequent Speakers

Figure 8 shows that there are 80 people with more than 100 occurrences. The people with the most recurrences are displayed in Table 1 Most Annotated Speakers.

| Order | Person | Number of Occurrences | Occupation/Relation |
|---|---|---|---|
| 1 | Toom Uku | 1346 | Presenter |
| 2 | Mälberg Mall | 1230 | Presenter |
| 3 | Eentalu Riina | 1179 | Presenter |
| 4 | Vare Kai | 1040 | Presenter |
| 5 | Karjatse Tõnu | 946 | Presenter |
| 6 | Otsmaa Margitta | 940 | Presenter |
| 7 | Kiisler Indrek | 906 | Presenter |
| 8 | Kelmsaar Vallo | 669 | Presenter |

| Order | Person | Number of Occurrences | Occupation/Relation |
|---|---|---|---|
| 9 | Salme Janek | 618 | Presenter |
| 10 | Gaškov Ago | 508 | Reporter for the show |
| 11 | Ansip Andrus | 498 | Former Prime Minister of Estonia<br><br>Vice President of the European Commission |
| 12 | Ojakivi Mirko | 483 | Reporter for the show |
| 13 | Kenk Olev | 461 | Reporter for the show |
| 14 | Lass Liisu | 437 | Presenter |

Table 1 Most Annotated Speakers

# 5 Speaker Identification

The chapter specifies the solution part of the thesis. It includes a detailed explanation of both the training and speaker identification processes; the data processing beforehand and post-filtering for retaining only confidently identified speaker labels.

## 5.1 Process

The imperative part of the thesis lies in the process of identifying speakers. Upcoming chapters describe how weakly supervised learning is used and the data is processed.

### 5.1.1 Training the Model

Before using the DNN for identifying speakers, it must be trained. In our case, the speaker identification model processes weakly labelled data and backpropagates using label regularization. The process, as shown in Figure 9, is also further explained as an algorithm (Table 2 Algorithm for Training the DNN).

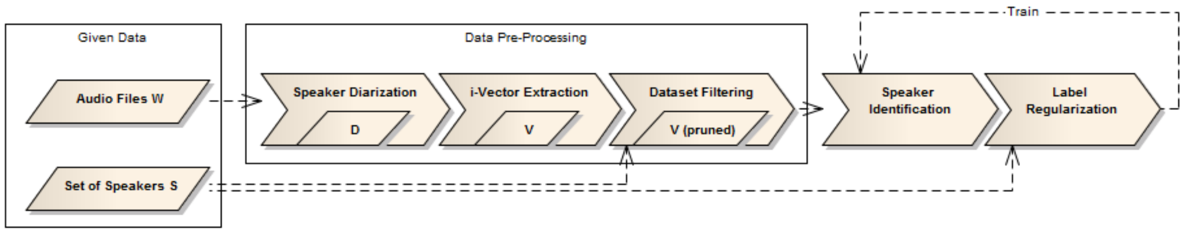

Figure 9 Training Process Diagram

#### 5.1.1.1 Algorithm for Training the DNN

**Given Data**

1) Audio files for shows $W = w_i$, where $i = 1 \dots N$.
2) Set of speakers for each show $S = s_{il_i}$,

   where $l_i$ is the number of speaker identities found in show $i$.

**Algorithm**

| **Action** | **Result** |
|---|---|
| 1. Process audio streams using speaker diarization | Audio cluster segments for each show $D = v_{ik_i}$, where $k_i$ is the number of speaker clusters found in show $i$. |
| 2. Extract i-vectors | i-Vectors for each show $V = v_{ik_i}$. |
| 3. Filter the set of i-vectors | Pruned set of i-vectors for each show $V$. |
| 4. Create a Speaker Identification DNN with $d$ inputs and $c$ outputs<br>$d$ – Dimensionalitity of the i-vectors (600)<br>$c = Count(N)$ - Number of unique speakers over all the shows. | |
| 5. Train for $E$ epochs.<br><br>$for\ j = 1 \dots N$:<br>    $update\ DNN\ using\ label - regularization$<br>        $l = D(\tilde{p}_j \| \| p_{\theta j})$<br>Where:<br>$\tilde{p}_j(speaker) = \begin{cases} \frac{1}{N_j} & if\ speaker\ occurs\ in\ show\ j \\ 0 & otherwise \end{cases}$<br>Equation 5 Prior Probabilities for Speakers<br>$p_\theta$ – output of the DNN using its current parameters θ. | |

Table 2 Algorithm for Training the DNN

### 5.1.2 Using the Model for Labelling Speakers

The trained model is used for identifying speakers in new shows. The process – displayed on Figure 10 and described as an algorithm in Table 3 Algorithm for Labelling Speakers in New Shows - differs from training the system. The given data does not contain a speakers' information. Therefore, instead of filtering the speaker list before the speaker identification process, it is done post. Most importantly: the i-vectors extracted from the recording are classified – if probable labels are found.

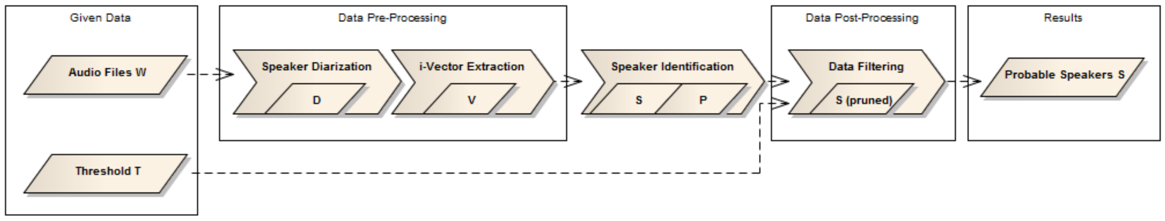

Figure 10 Labelling Process Diagram

#### 5.1.2.1 Algorithm for Labelling Speakers in New Shows

| **Given Data** | |
|---|---|
| 1) Audio files for shows $W = w_i$, where $i = 1 \dots N$.<br>2) Threshold T – confidence score required to classify the i-vector. | |
| **Algorithm** | |
| **Action** | **Result** |
| 1. Process audio streams using speaker diarization | Audio cluster segments for each show $D = v_{ik_i}$, where $k_i$ is the number of speaker clusters found in show $i$. |
| 2. Extract i-vectors | i-Vectors for each show $V = v_{ik_i}$. |
| 3. Classify $V$ using the trained model $DNN_\theta$ | 1) $S = s_{ik_i}$ – name of the speaker corresponding to the model output.<br>2) $P = p_{ik_i}$ – posterior probability of the speaker to be the owner of the cluster. |
| 4. Retain predictions having a posterior probability greater than the threshold $T$. | Pruned set of probable speakers for each show $S$. |
| 5. Output a list of confidently identified speakers $S$ | Set of probable speakers for each show $S$. |

Table 3 Algorithm for Labelling Speakers in New Shows

### 5.1.3 Explanation

The core of the process lies in the Speaker Identification DNN. This solves the main problem of the thesis. To properly employ it and achieve the goal set, the model must first be trained. Afterwards, it can be used for attaching labels to audio files.

When training, it is important to teach the model relations between the radio shows and the speakers in them. These constitute as the main input data. The most efficient way to train the DNN is to use i-vectors instead of audio recordings or diarized audio segments. This means that the input data must be transformed and pre-processed. The DNN is then trained one show at a time. It matches the $N$ i-vectors with the $M$ speakers. Over several training rounds – called epochs – statistical relations and clusters are formed.

After training, the DNN should be able to label speakers’ speech segments. Therefore, we can use the same model for labelling new shows.

When using the model for labelling new recordings, the given data does not contain a list of speakers. Instead of this, we introduce a confidence threshold. If the system suggests a label with a corresponding probability that is less than the threshold, we regard it as a weak guess. If it exceeds the threshold, the system is confident enough that the speaker is correctly identified. Labelling new shows also creates an output a list of speakers with their corresponding audio segments. The suitable threshold values and training epoch amounts, where the model output is usually precise, can be found empirically.

## 5.2 Data Pre-processing

### 5.2.1 i-Vectors from Audio Recordings

As displayed on Figure 9 and Figure 10, the audio files are pre-processed in the same way. The dataset comprises of 6604 audio recordings.

First, speaker diarization is applied. Initially, the audio is in a single recording. After the process, there are N audio segments, which are grouped into M clusters.

Second, i-vectors are extracted. Audio segment clusters are put in and a text representation is created, see Figure 11.

-0.0977829, 4.147483, 1.895904, -1.869378, 0.6388721, 3.437083,-1.016486, …

Figure 11 i-Vector Example

In its full length, the i-vector contains 600 values for characterizing the speaker’s voice.

### 5.2.2 Speaker Diarization Within Each Show

Speaker diarization is performed using the LIUM SpkDiarization (Meignier, Merlin, 2010) toolkit. Input audio data is first segmented into shorter sentence like chunks. Segments are then classified as speech or non-speech using a Gaussian mixture model built from our about 240 hours’ worth of training data, which originates from broadcast speech and lecture recordings. Segments containing speech are clustered, each cluster corresponding ideally to one unique speaker in the recording. BIC clustering followed by CLR-like clustering is applied (Barras et al., 2006).

### 5.2.3 i-Vector Extraction

The diarized data is further processed and a fixed-length 600 dimensional i-vector is extracted for each speaker. The process implements a speaker recognition system based on the Kaldi toolkit (Povey et al., 2011).

### 5.2.4 Pruning the Dataset

The training data was gathered on the 23rd of November 2016 and at that time, there were 6604 valid "Päevakaja" shows. Ten recordings were manually removed because they contained only the show's jingle.

The dataset is pruned based on the values of the two lists: i-vectors in shows and speakers in shows. The former is created with pre-processing methods – see "5.2.1 i-Vectors from Audio Recordings". The latter is the metadata.

### 5.2.5 Removing Shows with not Enough Speakers

For training the speaker identification, only shows with two or more speakers were used. 2347 shows were ruled out per the Figure 5 in chapter "4.3.4. Speaker and Vector Count in Shows". Thus, 4257 shows could be used for training the model.

### 5.2.6 Removing Shows with not Enough i-Vectors

The model is trained with shows with at least two i-vectors. As Figure 5 demonstrates, eight shows container zero i-vectors and 42 shows contained one. Two of the shows matched with the ones defined in "5.2.2.1 Removing Shows with not Enough Speakers". Therefore, 48 shows were ruled out from the training data.

After pruning the dataset, 4209 shows will be used for training the model.

### 5.2.7 List of Distinct Speakers

The system defines a list of possible speakers.

| List of Speakers |
|---|
| Speaker 1 |
| Speaker 2 |
| Speaker 3 |
| Speaker 4 |
| … |
| Speaker 13770 |
| Unknown |

Table 4 Ordered List of Distinct Speakers

The list contains distinct speaker values (Table 4). An extra "unknown" value is added because speaker names and shows with too few relations are removed. See "5.2.2 Pruning the Dataset". The values are used for applying labels to new users.

## 5.3 DNN architecture

The DNN – as shown on Figure 12 – consists of an input layer, output layer and several hidden deep learning layers in between.

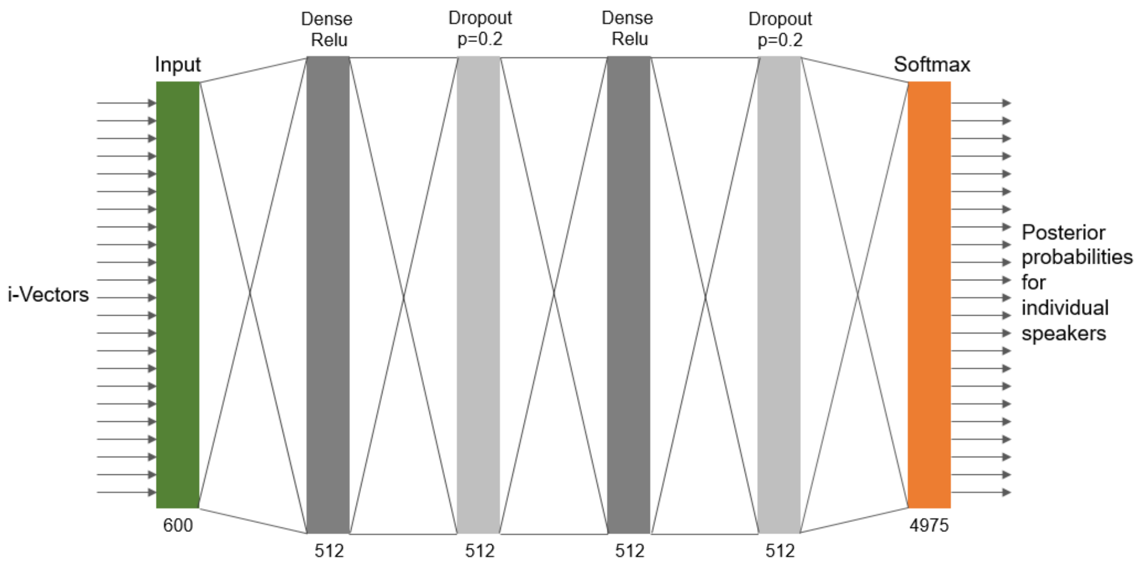

Figure 12 Representation of the Deep Neural Network's Architecture

The architecture and different layers are described more precisely in the upcoming subsections.

### 5.3.1 The Input Layer

The input layer accepts i-vectors with a fixed 600-value length. These characteristics define the speaker’s voice.

### 5.3.2 Hidden Layers

The model consists of four hidden layers. These create the inner connections to process complex data. Two of the layers are dropout layers and two are dense layers implementing ReLU algorithm.

### 5.3.3 Dense Layers with ReLU Activation Function

The internal relations are formed using dense ReLU layers. Each node passes its information to every node in the next layer. The use of fully connected layers helps form complex representations of the input data.

### 5.3.4 Dropout Layers

Machine learning can accomplish complex tasks with neural networks. However, when using dense layers, overfitting may occur. This can be minimized by using dropout layers.

### 5.3.5 Output Probability Vector

The Softmax function is the final layer. This insures that the outputs are positive percentage values. The output is a list of speakers with a corresponding probability. It represents how likely they are the person depicted by the input i-vector.

## 5.4 Training the Model

The most efficient way to train the model is with i-vectors. This is a uniform text representation of the speakers’ audio segments. Each i-vector contains exactly 600 numeric values that characterize the speaker’s voice. Therefore, the model needs exactly 600 inputs.

The DNN is then trained one show at a time. It is taught to correlate the N i-vectors to the M speakers of each show to another. The model is trained over 20 epochs – the number was found empirically. The model was used to label new shows and 20 training rounds was enough to satisfy the objectives of the thesis. During the training epochs, statistical relations are formed. Based on this, speakers from new shows can be identified.

## 5.5 Backpropagation Through Label Regularization

The neural network is not fully supervised – the model is not trained with exact vector and speaker name pairs. Instead, the data is weakly labelled. Therefore, backpropagation is done one show at a time and implemented using label regularization. Label regularization is a special case of expectation regularization (Druck et al., 2007). The usual label regularization approach uses both strongly labelled data and unlabelled data for training.

Within the thesis, the prior probabilities of the labels (speaker identities) over groups of examples (shows) are known. Firstly, speakers who are annotated in the show's metadata have a prior probability of $\frac{1}{N_s}$, where $N_s$ represents the number of speakers in the show. Therefore, the posterior probability should also be optimized for this value (Equation 6):

$$\tilde{p}_s(speaker) = \frac{1}{N_s}$$

Equation 6 Average Posterior Probability

Secondly, for speakers, who are not present in the annotated metadata, the model must be optimized to output the probability of zero (Equation 7):

$$\tilde{p}_s(speaker) = 0$$

Equation 7 Average Posterior Probability for Speakers Not in the Show

The distance is calculated as a sum of the KL-divergences of the known prior and predicted posterior probabilities over all shows in our training set. This is because we are not looking at a single speaker's probabilities, but the whole list as one.

## 5.6 Technical implementation

The programme was written in Python. The language was chosen because of the multitude of available libraries and the ease of use in prototyping. Methods for manipulating data and implementing machine learning algorithms is are readily available. It was possible to concentrate on the theoretical side of the problems first. After that, it was possible to write the code with existing libraries and validate the proposed approaches.

The complete code used for training is available in “Appendix 2 – Python Code for Training the Model”. The modified code that was used to identify speakers in new shows is brought out in “Appendix 3 – Python Code for Applying the Model”.

### 5.6.1 Keras

The speaker identification DNN was implemented with Keras – a Deep Learning library for Theano and TensorFlow. Keras has implemented the low-level functionality of neural networks. The library is accessible via a Python API. The objective of Keras is to allow high-level manipulation of neural networks (Keras.io, 4th of April 2017).

All the DNN’s layers – defined in “5.3 DNN Architecture” – are immediately available for use. Nonetheless, it is possible to change the characteristics of the machine learning algorithms completely. This means that the outcome of the defined models is fully in the hands of their creator.

# 6 Validation

This chapter assures that the model is accurately constructed and trained; the labels attached by it are in fact correct. A random held-out set of recordings is extracted, processed through speaker identification and compared against manually labelled data. The upcoming sections validate whether the objectives of the thesis have been met.

## 6.1 Measuring the Precision

The model is validated for recall and precision using a random set of audio files not used for training it, labelling each speaker vector manually and comparing the results. The goal is 70% recall – how many of the speakers in the random set were also in the training data – and 90% precision – the number of recalled speakers' vectors recognized correctly by the model.

## 6.2 Process of Validation

For validation, five shows from the Estonian Public Broadcasting's "Päevakaja" series were selected at random from a later period of time – training data was up until November 2016, test data was from April 2017. The audio was processed similarly to as described in "5.1.2 Using the Model for Labelling Speakers in a New Show". Before submitting the i-vectors to the Speaker Identification DNN, they were manually labelled to compare the DNN's outcome to the correct values.

The complete results are brought out in "Appendix 1 – Validation".

The "Päevakaja" shows selected were:

- https://arhiiv.err.ee/vaata/paevakaja-nr-20444
- https://arhiiv.err.ee/vaata/paevakaja-nr-20445
- https://arhiiv.err.ee/vaata/paevakaja-nr-20446
- https://arhiiv.err.ee/vaata/paevakaja-nr-20447

- https://arhiiv.err.ee/vaata/paevakaja-nr-20448

The total number of speakers described in the metadata was 80. Juhan Kilumets, who occurs often in the show as a sports reporter, was not annotated. However, the speaker identification process was able to classify his audio segments correctly.

## 6.3 Probability for Correct Speaker Identification

The correlation between the number of occurrences in the training data and the precision.

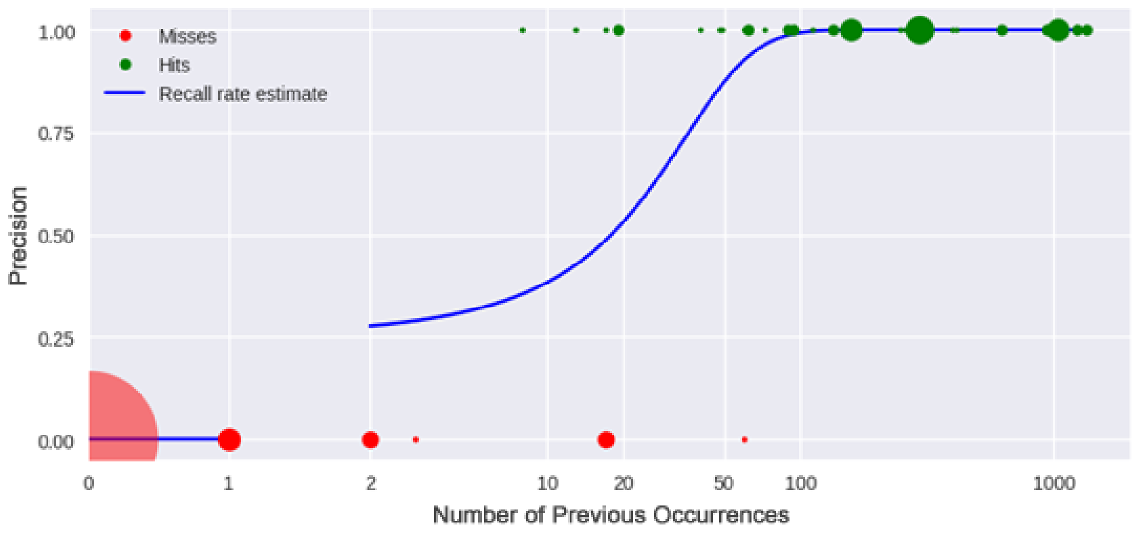

Figure 13 Number of Speaker Occurrences in Training Data

Figure 13 characterizes the relationship between the number of speaker occurrences in training data and the probability of re-identifying the speaker in a new show. Each red dot corresponds to one or more missed identifications in the validation data. Each green point corresponds to one or more identified speakers. The sizes of the dots correspond to the number of such cases. The blue line is the actual recall probability, estimated using logistic regression over the validation data. Since we prune the training data so that speakers with less than two occurrences are eliminated, such speakers naturally receive a zero recall probability.

Logistic regression shows that re-identification becomes more likely than not when the number of training occurrences for the speaker is around 17, although the minimal number of training occurrences that resulted in successful re-identification in the validation data was only eight. On the other hand, the maximal number of occurrences that still did not result in confident re-identification was 60. Therefore, the tipping point, where identification will be probable cannot be absolutely defined with such little validation data.

In most cases, the system does not have problems identifying speakers with over 10 previous occurrences. The validation data contains only two instances of this. Eerik-Niiles Kross has 17 recordings but is not recognized in any cases. It may be that the previous audio segments were of telephone calls or the voice characteristics were not similar enough to the previous i-vectors. Erle Loonurm, who occurred in 60 previous recordings, had two different clusters in a single show. The second one was identified correctly. The first had the show's jingle playing in the background and was not recognized.

The least likely "person" to be identified was the show's jingle. This is a miscalculation on behalf the speaker diarization model used, i.e. the chime should not be characterized as voice altogether. The isolated chime was represented as six clusters in the five shows. Other speakers, who were not identified by the DNN, either had no previous occurrences or were not even annotated in the show's data. For example, speakers from an un-annotated crowd asked questions from the current Prime Minister of Estonia.

### 6.3.1 Recall and Precision Metrics

The success is calculated based on the number of speaker clusters created by speaker diarization. The model at times creates multiple clusters for a single person and no clusters for some. Still, the diarization method works correctly almost all the time and its success rate is not the topic of the thesis.

The goal set is 70% on recall. Including all the vector clusters – the show's chimes and speakers not annotated – it was 59%. That means 35 out of 88 vectors did not receive any label at all. The goal was not accomplished. Because the objective of the thesis is to create relations based on previous occurrences, the model should not be invalidated because of unknown speakers. Therefore, it is reasonable to remove the six show's jingles, three undocumented speakers and the 23 audio segment clusters belonging to people with zero occurrences in the training data. When the model's recall is validated with data from previously occurring speakers, the objectives are achieved. The system attaches 53 labels the 56 relevant audio segment clusters – resulting a value of 94.6%.

Precision is the metric that verifies that the label set to a cluster is indeed correct and worth being added. The goal is set at 90%, which indicates that if the system applies a label, then 9 out of 10 times, it will be valid. Out of the 53 vector clusters recalled, 52 were correct, i.e. 98%, thus more than accomplishing the goal. This metric is even more important than recall because it represents that the problem stated in the thesis is correctly solved.

## 6.4 Speakers Not Recognized in the Speaker Diarization

Speaker Diarization created 88 clusters total, but six of them were the show's chime, three were un-annotated speakers and many times a single speaker had multiple clusters. Because of this, out of the 80 speakers, 13 did not receive a cluster, where the speaker was the owner of the most segments.

| **Person** | **Number of Prior Occurrences** |
|---|---|
| Nauris Klava | 1 |
| Marine Le Pen | 1 |
| Toomas Kiho | 11 |
| Tiia Korv | 0 |
| Karli Lambot | 6 |
| Paul Ryan | 1 |
| Sebastien Baheux | 0 |
| Hilja Karakatš | 0 |
| Philippe Martin | 0 |
| Valérie Nataf | 0 |
| Katrin Rehemaa | 40 |

| **Person** | **Number of Prior Occurrences** |
|---|---|
| Urmas Sule | 42 |
| Arthur Muller | 0 |

Table 5 Speakers not Recognized

The list (Table 5) gives insight, why speaker diarization could not even create a cluster for some speakers. Seven of these people are foreign politicians or public representatives, meaning they were most likely to be broadcasted as a short recording, sometimes taken from a press release with a lot of background noise. In some cases, because of the brevity of the recordings, the audio segments are combined with another person’s cluster. For example, Paul Ryan’s press release was added to Juhan Kilumets’s cluster.

Overall, diarizing and not diarizing speakers, is not the main effort in the thesis, but a process for shaping the data available to be used in the Speaker Identification DNN.

# 7 Usage and Future Work

This chapter analyses, how to further improve the system. The real-world usages of the system are illustrated along with ways of implementing the findings of the thesis in similar research topics.

## 7.1 Implementing the Speaker Identification Model AS-IS

It is possible to use the speaker identification system in its current form. The Institute of Cybernetics in Tallinn University of Technology has a hub for displaying latest “Päevakaja” news recordings (Kõnesalvestuste Brauser, 2017). The institute has been working many years on speech-to-text and speaker diarization with the exact show used in the thesis. At the moment, the speech segments are annotated as “Speaker *N*”. We can incorporate the trained speaker identification model with the already working functionality. This will further improve the value and quality of the information in the developed system.

## 7.2 Possible Usage

Speaker identification is a useful tool in many cases.

The simplest way to extend the list of applications of this thesis is to find a similar recordings’ archive and apply weakly supervised learning to them. The objective would be to properly annotate the recordings.

Similarly, it is possible to use Estonian Parliament’s verbatim records (Verbatim Records, Parliament of Estonia, 23.12.2017). These recordings contain both audio-visual data and the list of speakers. If the model has been trained with politicians’ voice characteristics, it can also be applied to identify speech from public forums. This is most relevant during election periods.

Label regularization can be used in combination with strictly labelled and unlabelled data. This means that the writing of verbatim records of courts or shorthand reports in business meetings could be implemented using the speaker identification model. To train the model, either a short example of each participants' voice would be required or weakly labelled data or recordings from previous meetings would suffice.

The results can be used in many forms.

1) Together with speech recognition, an indexed and searchable text corpus is created.
2) It is possible to re-listen the participants' sentences - speaker diarization assembles a list of each speaker's audio segments.
3) The aggregated length of each speakers' audio segments represents the amount of time they had on the show. This is important in political debates – each candidate should have equal time to convey their message and views.

## 7.3 Continuity of the "Päevakaja" Data

In recent years, the data has been annotated diligently. This is shown on Figure 2 in "4.3.4.2. Average Number of Speakers and i-Vectors per Show per Year".

It is possible and necessary to further train the model – new speakers may arise: be it reporters and presenters or politicians and public figures. For example, the topic of French presidential elections arose during the period after collecting the training data (the end of November) and before selecting the validation data (the end of April). Thus, the validating process did not recognize and identify Emmanuel Macron, because there were no previous occurrences for him. If the process is continuously used, the results will improve as well.

## 7.4 Future Work

The objectives of the thesis were achieved. Nonetheless, the process can be further improved.

One of the biggest problems arose with validation: the show's jingle was not annotated in the shows' metadata but was present in the audio segments. To improve the identification process, the show's jingle could be added as a "guest" to each show. Thus, it can be identified as well.

If the process is implemented in a continuously updated environment such as "Kõnetuvastuse Brauser", the speaker identification process should often be run on older recordings as well. A show exists, after which it is highly probable that the speaker is correctly labelled – let this be defined as the $N^{th}$ show. The thesis found empirically that this might be the 8th or 17th show the person has occurred in. Therefore, there also exist $N - 1$ shows, where the person was identified as "Unknown" or even incorrectly. After the $N^{th}$ show, the speaker identification model is sufficiently trained to identify the person. Thus, we are able to correctly label the speaker in the previous $N - 1$ shows that were first used for training the model.

# 8 Summary

The chapter analyses whether the defined problems were solved and the set objectives were achieved. The thesis is constructed around a speaker identification DNN, which was trained with label regularization over thousands of recordings with weakly labelled data. The problems, which it tried to solve, were creating a weakly supervised model for speaker identification and using label regularization as a cost function.

The weakly supervised DNN was used in two phases: firstly, it was trained with 6604 audio recordings and the metadata related to them, and secondly, it was used to identify speakers in new radio shows to validate it. For both the training part and using the model, audio recordings were pre-processed with speaker diarization and i-vector extraction. The uniform data was input to the DNN one show at a time.

The DNN was trained with backpropagation using a set of annotated speakers for each show. The cost function was implemented as label regularization, which encourages average model predictions within each show to match label priors based on the annotated speakers in given show. If these values did not match, the model's weights were adjusted accordingly. A consolidated representation of the speakers' audio segments was formed.

When identifying new shows, the DNN yielded a list of speakers with their respective posterior probabilities. Only the speakers with a confidence score above a certain threshold were retained.

The objectives were verified and validated using a randomly selected held-out of news recordings. The basis for a perfect score was not the number of distinct speakers identified, but the number of correctly labelled i-vectors. These values often differ, because the speaker diarization may create multiple classifications for some people and no clusters for others.

The goals were 70% on recall and 90% on precision. By the strictest measurements, the objective for recall was not met. Out of 82 i-vectors, 29 were not labelled – that corresponds a recall rate of 64.6%. Out of the unlabelled values, three belonged to speakers that were undocumented in the shows' metadata and another 23 represented people with zero occurrences in the training data. After isolating speakers for whom no training data exists, the objectives are achieved. 53 labels out of 56 relevant audio segment clusters are attached – resulting a value of 94.6%. The precision metric was achieved successfully: 98% of all labels were classified correctly. Only one label out of all 53 identifications, was a false positive.

The thesis proves that it is possible and recommendable to use weakly supervised training to identify speakers from audio recordings. The combination of using the weakly labelled data with label regularization is enough to create valid models for identifying speakers. This is validated with the recall and precision metrics, which conclude that the model can label audio segments confidently and correctly.

# 9 Bibliography

- IEEE Transactions on Audio, Speech, and Language Processing: Multistage speaker diarization of broadcast news. (2006). / C. Barras, Xuan Zhu, S. Meignier, and J. L. Gauvain.
- Bengio Y. (2009). Learning Deep Architectures for AI. Montreal, Canada.
- Collobert R. and Weston J. (2008). A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. Princeton, USA.
- Deep Learning Tutorials. (2nd of May 2017). Deep Learning [*Online*] http://deeplearning.net/tutorial/
- Druck G. (2011). Generalized Expectation Criteria For Lightly Supervised Learning. Massachusetts, USA.
- Ferras M. and Bourlard H. (2012). Speaker diarization and linking of large corpora. In *Proceedings of the IEEE Workshop on Spoken Language Technology*. Martigny, Switzerland.
- Ghahramani Z. (2001). An Introduction to Hidden Markov Models and Bayesian Networks. London, England.
- Speaker Recognition i-Vector Machine Learning Challenge. (2014). / G. S. Greenberg, D. Bansé, G. R. Doddington., D. Garcia-Romero, J. J. Godfrey, T. Kinnunen, A. F. Martin, A. McCree, M. Przybocki, M. A. Reynolds. Gaithersburg, USA.
- Institude of Cybernetics (12th of March 2017). Tallinn University of Technology [*Online*] http://bark.phon.ioc.ee/tsab/p/index
- Automatic Named Identification Of Speakers Using Diarization And Asr Systems. (2009). V. Jousse, S. Petit-Renaud, S. Meignier, Y. Estève, C. Jacquin. Nantes, France.
- Keras.io. (4th of April 2017). keras.io [Online] https://keras.io/
- Khoury, E., El Shafey L., Ferras M., and Marcel S. (2014). Hierarchical speaker clustering methods for the nist i-vector challenge In *Odyssey: The Speaker and Language Recognition Workshop*. Martigny, Switzerland.
- Kullback S., Leibler R. A. (1951) On Information and Sufficiency In *The Annals of Mathematical Statistics*. Washington D.C., USA.
- Madikeri S., Bourlard H. (2012). Kl-Hmm Based Speaker Diarization System For Meetings. Martigny, Switzerland.
- Meignier S. and Merlin T. (2010). LIUM SpkDiarization: an open source toolkit for diarization In *CMU SPUD Workshop*. Dallas, USA.
- The 2015 Sheffield System For Longitudinal Diarisation Of Broadcast Media. (2015). / R. Milner, O. Saz, S. Deena, M. Doulaty, R. W. M. Ng, T. Hain. Sheffield, UK.
- Unsupervised Speaker Identification using Overlaid Texts in TV Broadcast (2012). / J. Poignant, H. Bredin, V. B. Le, L. Besacier, C. Barras, G. Quénot. Grenoble, France.
- IEEE ASRU Workshop: The Kaldi speech recognition toolkit. (2011). / D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely. Microsoft Research, USA.
- Päevakaja uudised. (23rd of November 2016). Estonian Public Broadcasting [*Online*] https://arhiiv.err.ee/seeria/paevakaja/0/69

- Reynolds D. (2009). Universal Background Models∗ In *Encyclopedia of Biometrics*. Massachusetts, USA.
- Rojas R. (1996). The Backpropagation Algorithm. Berlin, Germany.
- Schmidhuber J. (2014). Deep Learning in Neural Networks: An Overview. Manno-Lugano, Switzerland.
- Self Taught Learning. (15th of March 2017). Standford University [*Online*] http://ufldl.stanford.edu/tutorial/selftaughtlearning/SelfTaughtLearning/
- Dropout: A Simple Way to Prevent Neural Networks from Overfitting. (2014). / N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov. Toronto, Canada.
- Szegedy C., Toshev A., Erhan D. (2013). Deep Neural Networks for Object Detection. San Francisco, USA.
- Täckström O. (2008) Semi-Supervised Learning for Transductive Speaker Annotation. Uppsala, Sweden.
- Verbatim records - Parliament of Estonia. (23rd of December 2016). riigikogu.ee [Online] http://stenogrammid.riigikogu.ee/en/
- Vijayasenan D. (2009). An Information Theoretic Approach to Speaker Diarization of Meeting Data. Martigny, Switzerland.

# 10 Appendix 1 – Validation

## 10.1 “Päevakaja Nr 20444”

Aired on: 20th of April 2017.

17 annotated speakers: **Salme Janek**, Sester Sven, Toom Uku, Kaasik Ülo, Ossinovski Jevgeni, Aug Lembi, Vare Kai, Mälberg Mall, Terras Riho, Lepik Indrek, Muller Arthur, Tralla Johannes, Lõhmus Asko, Leoma Rain, Saluorg Jane, Grabbi-Kaiv Silve, Vedru Johannes.

| **Gender** | **Manually Labeled** | **Prior Occurences** | **Speaker Identification Output** | **Recall?** | **Correct?** |
|---|---|---|---|---|---|
| F | *Show's chime* | *0* | - | No | - |
| F | Grabbi-Kaiv Silve | 112 | Grabbi-Kaiv Silve | Yes | Yes |
| F | Saluorg Jane | 49 | Saluorg Jane | Yes | Yes |
| F | Vare Kai | 1040 | Vare Kai | Yes | Yes |
| F | Aug Lembi | 3 | - | No | - |
| M | Salme Janek | 618 | Salme Janek | Yes | Yes |
| M | Mälberg Mall | 1230 | Mälberg Mall | Yes | Yes |
| M | Salme Janek | 618 | Salme Janek | Yes | Yes |
| M | Terras Riho | 48 | Riho Terras | Yes | Yes |

| Gender | Manually Labeled | Prior Occurences | Speaker Identification Output | Recall? | Correct? |
|---|---|---|---|---|---|
| M | Lepik Indrek | 158 | Lepik Indrek | Yes | Yes |
| M | Tralla Johannes | 295 | Tralla Johannes | Yes | Yes |
| M | Leoma Rain | 0 | - | No | - |
| M | Lõhmus Asko | 0 | - | No | - |
| M | Vedru Johannes | 17 | Vedru Johannes | Yes | Yes |
| M | Toom Uku | 1346 | Toom Uku | Yes | Yes |
| M | Sester Sven | 89 | Sester Sven | Yes | Yes |
| M | Ossinovski Jevgeni | 94 | Ossinovski Jevgeni | Yes | Yes |
| M | Sester Sven | 89 | Sester Sven | Yes | Yes |
| M | Kaasik Ülo | 19 | Kaasik Ülo | Yes | Yes |

Table 6 Validation Results for Show “Päevakaja Nr 20444”

## 10.2 “Päevakaja Nr 20445”

Aired on: 20th of April 2017.

24 annotated speakers: **Vare Kai**, Rehemaa Katrin, Sule Urmas, Ossinovski Jevgeni, Lepik Indrek, Mälberg Mall, Nataf Valérie, Martin Philippe, Tralla Johannes, Baheux Sebastien, Ehand Epp, Peterson Indrek, Tali Margus, Hindre Madis, Karakatš Hilja, Hinto

Luule, Urman Peeter, Kilusk Jaan, Nutov Mirjam, Faust Maria, Erm Anne, Karjatse Tõnu, Sild Kertu, Kaasik Ragnar.

| **Gender** | **Manually Labeled** | **Prior Occurences** | **Speaker Identification Output** | **Recall?** | **Correct?** |
|---|---|---|---|---|---|
| F | Vare Kai | 1040 | Vare Kai | Yes | Yes |
| F | Vare Kai | 1040 | Vare Kai | Yes | Yes |
| F | Vare Kai | 1040 | Vare Kai | Yes | Yes |
| F | Nutov Mirjam | 149 | Nutov Mirjam | Yes | Yes |
| F | Hinto Luule | 0 | - | No | - |
| F | Erm Anne | 8 | Erm Anne | Yes | Yes |
| F | Ehand Epp | 1 | - | No | - |
| F | Sild Kertu | 62 | Sild Kertu | Yes | Yes |
| M | *Show's chime* | *0* | - | No | - |
| M | Ehand Epp, muu | 1 | - | No | - |
| M | Ehand Epp | 1 | - | No | - |
| M | Kilusk Jaan | 0 | - | No | - |
| M | Hindre Madis | 135 | Hindre Madis | Yes | Yes |
| M | Tali Margus | 0 | - | No | - |
| M | Urman Peeter | 0 | - | No | - |
| M | Karjatse Tõnu | 946 | Karjatse Tõnu | Yes | Yes |

| Gender | Manually Labeled | Prior Occurences | Speaker Identification Output | Recall? | Correct? |
|---|---|---|---|---|---|
| M | Maria Faust (Tõnu Karjatse interview) | 0 | - | No | - |
| M | Mälberg Mall | 1230 | Mälberg Mall | Yes | Yes |
| M | Lepik Indrek | 158 | Lepik Indrek | Yes | Yes |
| M | Kaasik Ragnar | 133 | Kaasik Ragnar | Yes | Yes |
| M | Rehemaa Katrin | 40 | Rehemaa Katrin | Yes | Yes |
| M | Ossinovski Jevgeni | 94 | Ossinovski Jevgeni | Yes | Yes |
| M | Tralla Johannes | 295 | Tralla Johannes | Yes | Yes |

Table 7 Validation Results for Show "Päevakaja Nr 20445"

## 10.3 "Päevakaja Nr 20446"

Aired on: 20th of April 2017.

13 annotated speakers: **Otsmaa Margitta**, Ryan Paul, Kiho Toomas, Lambot Karli, Leas Reene, Ratas Jüri, Birk Siim, Korv Tiia, Transtok Eduard, Kundla Rene, Pervik Aino, Merilain Merike, Kilumets Juhan.

| **Gender** | **Manually Labeled** | **Prior Occurences** | **Speaker Identification Output** | **Recall?** | **Correct?** |
|---|---|---|---|---|---|
| F | Otsmaa Margitta | 940 | Otsmaa Margitta | Yes | Yes |
| F | Transtok Eduard | 0 | - | No | - |
| F | Pervik Aino | 2 | - | No | - |
| F | *Random child, not annotated (Ratas Jüri)* | *0* | - | No | - |
| F | Merilain Merike | 259 | Merilain Merike | Yes | Yes |
| F | Leas Reene | 302 | Leas Reene | Yes | Yes |
| M | *Show's chime* | *0* | - | No | - |
| M | Kundla Rene | 13 | Kundla Rene | Yes | Yes |
| M | Kilumets Juhan | 309 | Kilumets Juhan | Yes | Yes |
| M | Random person, not annotated (Ratas Jüri) | 0 | - | No | - |
| M | Ratas Jüri | 72 | Ratas Jüri | Yes | Yes |
| M | Birk Siim | 0 | - | No | - |

Table 8 Validation Results for Show “Päevakaja Nr 20446”

Tallinn 2017

### 10.4 “Päevakaja Nr 20447”

Aired on: 20th of April 2017.

9 annotated speakers: **Loonurm Erle**, Tralla Johannes, Ehand Epp, Talvik Artur, Herkel Andres, Hindre Madis, Miil Tõnu, Vilgats Ester, Sild Kertu.

| **Gender** | **Manually Labeled** | **Prior Occurences** | **Speaker Identification Output** | **Recall?** | **Correct?** |
|---|---|---|---|---|---|
| F | Loonurm Erle (with jingle) | 60 | - | No | - |
| F | Loonurm Erle | 60 | Loonurm Erle | Yes | Yes |
| F | Vilgats Ester | 410 | Vilgats Ester | Yes | Yes |
| F | Sild Kertu | 62 | Sild Kertu | Yes | Yes |
| M | *Show's chime* | *0* | - | No | - |
| M | Miil Tõnu | 2 | - | No | - |
| M | Ehand Epp | 1 | - | No | - |
| M | Tralla Johannes | 295 | Tralla Johannes | Yes | Yes |
| M | Talvik Artur | 17 | - | No | - |
| M | Hindre Madis | 135 | Hindre Madis | Yes | Yes |

| **Gender** | **Manually Labeled** | **Prior Occurences** | **Speaker Identification Output** | **Recall?** | **Correct?** |
|---|---|---|---|---|---|
| M | Miil Tõnu (with jingle) | 2 | - | No | - |
| M | Herkel Andres | 95 | Herkel Andres | Yes | Yes |

Table 9 Validation Results for Show “Päevakaja Nr 20447”

## 10.5 “Päevakaja Nr 20448”

Aired on: 20th of April 2017.

17 annotated speakers: **Toom Uku**, Klava Nauris, Kaunissaare Kristjan, Simson Kadri, Otsmaa Margitta, Macron Emmanuel, Le Pen Marine, Tralla Johannes, Sobak Kristi, Kadai Martin, Joller Karmin, Karjatse Tõnu, Kross Eerik-Niiles, Mets Lembi, Tiko Teet, Saluveer Aarne, Jõemaa Ülle.

| **Gender** | **Manually Labeled** | **Prior Occurences** | **Speaker Identification Output** | **Recall?** | **Correct?** |
|---|---|---|---|---|---|
| F | *Show's chime* | *0* | - | No | - |
| F | *Random person, Sigre* | *0* | Josing Marje | Yes | No |
| F | Joller Karmin | 0 | - | No | - |
| F | Mets Lembi | 0 | - | No | - |
| F | Jõemaa Ülle | 246 | Jõemaa Ülle | Yes | Yes |
| F | Otsmaa Margitta | 940 | Otsmaa Margitta | Yes | Yes |
| F | Simson Kadri | 158 | Simson Kadri | Yes | Yes |
| F | Simson Kadri | 158 | Simson Kadri | Yes | Yes |
| F | Sobak Kristi | 398 | Sobak Kristi | Yes | Yes |
| M | *Show's chime* | *0* | - | No | - |
| M | Toom Uku | 1346 | Toom Uku | Yes | Yes |

| **Gender** | **Manually Labeled** | **Prior Occurences** | **Speaker Identification Output** | **Recall?** | **Correct?** |
|---|---|---|---|---|---|
| M | Karjatse Tõnu | 946 | Karjatse Tõnu | Yes | Yes |
| M | Kadai Martin | 0 | - | No | - |
| M | Kross Eerik-Niiles | 17 | - | No | - |
| M | Kross Eerik-Niiles | 17 | - | No | - |
| M | Tiko Teet | 0 | - | No | - |
| M | Saluveer Aarne | 19 | Saluveer Aarne | Yes | Yes |
| M | Kilumets Juhan | 309 | Kilumets Juhan | Yes | Yes |
| M | Kaunissaare Kristjan | 0 | - | No | - |
| M | Tralla Johannes | 295 | Tralla Johannes | Yes | Yes |
| M | Macron Emmanuel | 0 | - | No | - |
| M | Tralla Johannes | 295 | Tralla Johannes | Yes | Yes |

Table 10 Validation Results for Show “Päevakaja Nr 20448”

# 11 Appendix 2 – Python Code for Training the Model

```
#! /usr/bin/env python3.5

import argparse
import csv
import pandas
import random
import numpy
from keras import backend as K
from keras.models import Sequential
from keras.layers import Dense, Dropout

# Label regularization loss, according to Keras API
# Actually, our y_true is 1D, containing prior probabilities for
# our labels. But Keras API wants it to be a 2D array of shape
# (batch_size, num_classes)
# So, we expand it to 2D when calling train_on_batch (see below) and
# just take a mean
# in this function
def label_reg_loss(y_true, y_pred):
  # KL-div
  y_true = K.clip(y_true, K.epsilon(), 1)
  y_pred = K.clip(y_pred, K.epsilon(), 1)

  y_true_mean = K.mean(y_true, axis=0)
  y_pred_mean = K.mean(y_pred, axis=0)
  return K.sum(y_true_mean * K.log(y_true_mean / y_pred_mean), axis=-1)

if __name__ == '__main__':
    parser = argparse.ArgumentParser(description="Train a DNN")
    parser.add_argument("--save-model", default=None)
    parser.add_argument("--min-spk-occ", default=5, type=int,
      help="Keep speaker names that occur at least in that many shows")
    parser.add_argument("--num-epochs", type=int, default=20,
      help="Number of epochs to train")
    parser.add_argument("spk_file",
  help="File speaker data (IDs and i-vectors) in CSV format")
parser.add_argument("meta_file", help="Metadata file CSV format")

args = parser.parse_args()

metadata_df = pandas.read_csv(args.meta_file, sep=";", encoding='utf-8-sig')
speaker_df = pandas.read_csv(args.spk_file, sep=",", header=None)

# Dictionary that maps show ID to a set of names who appear in it
show2names = {}
# Reverse to above: speaker name -> set of show IDs
name2shows = {}

for index, row in metadata_df.iterrows():
    names_val = row['esinejad']
    if not pandas.isnull(names_val):
        names = set([s.strip() for s in names_val.split(",")])
        if len(names) > 0:
            show2names[row['id']] = names
            for name in names:
                name2shows.setdefault(name, set()).add(row['id'])




# keep names that occur at least args.min_spk_occ times across all shows
pruned_name2shows = \
  {name: shows for name, shows in name2shows.items() \
   if len(shows) >= args.min_spk_occ}
print("%s speakers left after pruning" % len(pruned_name2shows))

# pruned_name_list is a list of all names left after pruning, plus <unk>
# name_ids is a dict that maps names to their indexes in pruned_name_list
pruned_name_list = []
pruned_name_list = ["<unk>"]
pruned_name_list.extend(sorted(pruned_name2shows.keys()))
name_ids = {}
for name in pruned_name_list:
    name_ids[name] = len(name_ids)

# keep only the ivectors that are from a show that has name data
```

```
valid_speaker_df = \
  speaker_df[speaker_df[0].isin(show2names.keys())].reset_index(drop=True)

# name_ids_in_shows is a dict that maps show IDs to sets that contain
# all name IDs in that show, with a special ID for <unk> for
# pruned-out speakers
name_ids_in_shows = {}
for show, names in show2names.items():
    name_ids_in_show = set()
    for name in names:
        if name in name_ids:
            name_ids_in_show.add(name_ids[name])
        else:
            name_ids_in_show.add(name_ids["<unk>"])
    name_ids_in_shows[show] = name_ids_in_show

ivecs = valid_speaker_df.ix[:,3:].as_matrix()

# Create a DNN
model = Sequential()
model.add(Dense(512, activation='relu', input_shape=(ivecs.shape[1],)))
model.add(Dropout(0.2))
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(len(name_ids), activation='softmax'))
model.compile(optimizer='sgd', loss=label_reg_loss)

print("Model summary")
print(model.summary())

for epoch in range(args.num_epochs):
    # Train the DNN, show-by-show
    for show, names in random.sample(show2names.items(), k=len(show2names)):
        ivecs_for_show = \
          valid_speaker_df[valid_speaker_df[0] == show].ix[:,3:].as_matrix()
        # Label proportions: uniform over the names in that show
        label_props_for_show = numpy.zeros((len(name_ids)))
        label_props_for_show[list(name_ids_in_shows[show])] = \
          1.0 / len(name_ids_in_shows[show])
        # Expand label proportions, because Keras needs labels
        # to be of the same length as the minibatch
        # We will later un-expand it in the cost function
        label_props_expanded = numpy.repeat(
          label_props_for_show.reshape(1,-1),
          len(ivecs_for_show), axis=0)
        model.train_on_batch(ivecs_for_show, label_props_expanded)
    print("Finished epoch %d" % epoch)

print("Finished training")

if args.save_model:
  model.save(args.save_model)
  print("Saved model to %s" % args.save_model)
  with open("%s.names" % args.save_model, "wt", encoding='utf-8') as f:
    for name in pruned_name_list:
      print(name, file=f)
```

Tallinn 2017

# 12 Appendix 3 – Python Code for Applying the Model

```python
#! /usr/bin/env python3.5

import argparse
import csv
import pandas
import numpy
from keras.models import load_model
from train_dnn import label_reg_loss

def get_speaker_str(speaker_df, row_id):
    return "%s__%s__%s" %(speaker_df.ix[row_id, 0], \
                          speaker_df.ix[row_id, 1],
                          speaker_df.ix[row_id, 2])

if __name__ == '__main__':
    parser = argparse.ArgumentParser(description="Apply the model")
    parser.add_argument("--confidence-threshold", type=float, default=0.7,
      help="Posterior probability threshold for confident predictions")
    parser.add_argument("model_file", help="Previously traine model")
    parser.add_argument("dev_spk_file",
      help="File with dev speaker data (IDs and i-vectors) in CSV format")

    args = parser.parse_args()

    model = load_model(args.model_file,
                       custom_objects={"label_reg_loss" : label_reg_loss})

    # Load the name table, needed for mapping output IDs of the model
    # to real speaker names
    pruned_name_list = []
    name_ids = {}
    for l in open("%s.names" % args.model_file, "rt", encoding='utf-8'):
      name = l.strip()
      pruned_name_list.append(name)
      name_ids[name] = len(name_ids)
```

```
dev_speaker_df = pandas.read_csv(args.dev_spk_file, sep=",", header=None)
dev_ivecs = dev_speaker_df.ix[:,3:].as_matrix()
dev_predicted_targets = model.predict_on_batch(dev_ivecs)
dev_predicted_speakers = (dev_predicted_targets).argmax(axis=1)
dev_confident_predictions = \
  dev_predicted_targets[numpy.arange(len(dev_predicted_targets)), \
                        dev_predicted_speakers] > args.confidence_threshold

for i in numpy.where(dev_confident_predictions)[0]:
  if dev_predicted_speakers[i] != name_ids["<unk>"]:
      print(u"%s --> %s" % \
        (get_speaker_str(dev_speaker_df, i), \
         pruned_name_list[dev_predicted_speakers[i]].encode("utf-8")))
```

Tallinn 2017