TALLINNA TEHNIKAÜLIKOOL

Infotehnoloogia teaduskond

Karl Kask 154911IAPB

NÄOTUVASTUSEL PÕHINEV RAAMATUKOGU LAENUTUSSÜSTEEM

Bakalaureusetöö

Juhendaja: Martin Verrev

MSc

Tallinn 2019

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Karl Kask

20.05.2019

Annotatsioon

Lõputöö eesmärgiks on edasi arendada näo- ning optilisel märgituvastusel põhinevat raamatukogu laenutus infosüsteem. Selle jaoks analüüsitakse ning leitakse sobivaim variant vastavalt ärinõuetele mitmete näotuvastuse ning OCR teenuste seast sh. *Microsoft Azure Cognitive Services*, *OpenCV*, *Google Cloud Vision API* ning *Face++ Cognitive Services*. Ärinõuetele vastavuse analüüs toimub ettevõtte Helmes AS raamatukogus, mida kasutatakse laborkeskonnana. Lõputöö tulemusena valmiv infosüsteemi hakkab kasutama nii Helmes AS raamatukogu kui ka tulevikus Uhtna raamatukogu.

Bakalaureusetöö käigus luuakse kolmekihilise infosüsteem, mis koosneb *Angular*-is loodud kasutajaliidesest, *Spring Boot* serverirakendusest ning *PostgreSQL* andmebaasist.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 31 leheküljel, 8 peatükki, 20 joonist, 4 tabelit.

Abstract

A Library Rental System Based on Facial Recognition

The goal of this thesis is to further develop a library rental system prototype based on facial recognition identifying the user and optical character recognition identifying the book being rented. In order to do this there is an analysis of the existing prototype of the developed system which this thesis is based on. In this analysis it will be determined how to improve and make the new system better. There is also an analysis of current facial and optical character recognition solutions including Microsoft Azure Cognitive Services, OpenCV, Google Cloud Vision API and Face++ with various experiments to determine the best solution according to the business requirements established in the analysis part of this thesis. These requirements will be validated at first with the experiments conducted to determine the best solutions the developed application is going to use but also with user testing in the office library of Helmes AS. The finished application developed in this thesis will be used both in the office library of Helmes AS and also in the library of Uhtna in the future.

The background and history of both facial recognition and optical character recognition is also discussed.

The system developed in this thesis will use Spring Boot 2 for the backend, Angular 7 for the frontend and PostgreSQL for the database.

The thesis is in Estonian and contains 31 pages of text, 8 chapters, 20 figures, 4 tables.

Lühendite ja mõistete sõnastik

OCR	Optilisel märgituvastus *(Optical Character Recognition)*
FERET	*Facial Recognition Technology*
FRVT	*Face Recognition Vendor Test*
FRGC	*Face Recognition Grand Challenge*
API	*Application Programming Interface*
HTML	Hüperteksti märgistuskeel *(HyperText Markup Language)*
CSS	Kaskaadlaadistik *(Cascading Style Sheets)*
JWT	JSON-il põhinev juurdepääsu märkide avatud standard *(JSON Web Token)*
FDDB	*Face Detection Data Set and Benchmark*
300-W	*300 Faces In-the-Wild Challenge*
LFW	*Labeled Faces in the Wild*
ID	*Identity Document*
REST	*Representational State Transfer*

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

Käesolevas töös uurin võimalusi kasutada näotuvastust isikutuvastuseks ning optilist märgituvastust markeeritud esemete, antud töö juhul raamatute, tuvastamiseks. Helmes AS Tallinna kontoris tulid hiljuti kasutusele biomeetrilistel andmetel põhinevad turvaseadmed uste avamiseks, seal hulgas sõrmejäljelugejad ning sügavuskaamerad näotuvastuseks. Sealt tuli ka idee ära kasutada näotuvastust Helmese kontoris muudel otstarvetel, näiteks tehnikaseadmete laenutamisel kui ka kontori raamatukogus. Esimene prototüüplahenduse lõin ma 2018 aasta suvel ning minu bakalaureusetöö põhieesmärgiks on prototüübi edasiarendamine ja seeläbi parendamine ning viimine täielikult eraldiseisvale kujule, ehk lahti siduda Helmese enda infrastruktuurist. Eraldiseisvat süsteemi peab saama kasutada integreerituna ka muudes asutustes.

Näotuvastus on esile kerkinud kui üks kõige aktiivsemaid arvuti nägemisvõimekust ära kasutavaid teadusuuringute valdkondi. Selle jaoks on välja töötatud mitmeid meetodeid, mis peavad lahendama mitmeid väljakutseid seoses näopildi võimaliku varieeruvusega. Näiteks näoilme, poos ning valgustus. See kõik toob endaga kaasa muutuva ning tihti väga robustse sisendi mis iganes süsteemile [1].

OCR ehk optiline märgituvastus on tehnoloogia, mille abil on võimalik arvutitel piltidel kujutatud teksti skännida, ning seda siis omakorda töödelda juba tekstidokumendis oleva tekstina. Tänapäeval on OCR lahendused juba üpris võimekad arvestades, et sisendi pildil võib tekst olla suure kalde all, ebakontrastne või ka näiteks eritooniline. Samuti on praegused OCR lahendused võimelised tuvastama täpselt teksti näiteks lõikude ja pealkirjadena. Selleks jagatakse tekst plokkideks ning nende plokkide kaupa selgitatakse välja teksti asetus [2].

Bakalaureusetöö teises ning kolmandas peatükis räägin vastavalt näotuvastuse ja optilise märgituvastuse ajaloost, meetoditest ning rakendustest. Töö neljandas peatükis seletan lahti olemasoleva prototüüplahenduse ning selle probleemid. Viiendas peatükis viin läbi eksperimendid eesmärgiga leida paremaid tehnoloogiaid näotuvastuseks ning raamatute tuvastamiseks, et neid hiljem uues arenduses kasutada. Kuuendas peatükis räägin uue

lahenduse eesmärkidest, nõuetest ning realisatsioonist, millega bakalaureusetöö eesmärgid saavutan. Seitsmendas peatükis räägin töö tulemustest ning viimases kaheksandas peatükis võtan kokku kogu töös saavutatu.

2 Näotuvastus

Näotuvastussüsteem on biomeetriline tarkvara, mis suudab kokku sobitada inimese digitaalse näopildi varasemalt treenitud piltide baasist teise sama inimese näopildiga ja seeläbi tuvastada näiteks isiku identiteedi või mis iganes muu vastavuse, millega inimese näopilti parasjagu seostati.

Sellised süsteemid leiavad tänapäeval tüüpiliselt kasutust turvasüsteemides juurdepääsu kontrolli näol, kus ta on küll ebatäpsem näiteks silma vikerkesta- või sõrmejäljelugejatest, kuid näotuvastuse eeliseks on kasutajale kontaktivaba ning mitteinvasiivne protsess. Veel kasutusalasid esineb inimeste ja arvutite vahelisel suhtlusel ning ka videojärelvalves, näiteks turvalisuse tagamisel avalikus ruumis [3]. Hiljuti on näotuvastussüsteemid kasutust leidnud ka kommertseesmärkidel, kus näiteks inimestele paremini suunatud reklaami tegemiseks jälgitakse nende liikumisi avalikus ruumis [4].

Joonis 1. Näotuvastus tarkvara kasutamine Pekingis Mengvii kontoris.

Hiinas on laialdaselt kasutusel näotuvastusega populatsiooni indekseerimine just korrakaitse eesmärgil [5] ning samuti kasutatakse näotuvastust veel näiteks sigade loendamiseks [6].

2.1 Näotuvastuse ajalugu

Näotuvastusega tegelevad tehnoloogiad said alguse 1960-ndate keskel, kui Woody Bledsoe koostöös Helen Chan Wolf-i ja Charles Bisson-iga alustasid kasutama arvuteid näopiltide ära tundmiseks. Bledsoe algne mudel koosnes näo orientiiride nagu silmade, suu, nina asukohtade käsitsi sisestamisest programmi, mille arvuti omakorda matemaatiliste valemite abil pööras, et võtta arvesse variatsioone poosis. Peale Bledsoe-d jätkati tema tööga Stanfordi uurimisinstituudis, ning eksperimentides, kus kasutati üle 2000 näoga andmebaase suutis arvuti järjekindlalt edestada inimesi samades äratundmisülesannetes [3]. Sellest ajast alates on sarnaste süsteemide arendamisega edasi tegeletud, ning arenduste propageerimiseks ning innustamiseks on püstitatud rahalise preemiaga ülesandeid nagu näiteks Ameerika Ühendriikide valitsuse poolt rahastatud 1993. aastal alustanud „Facial Recognition Technology" (FERET), 2000. aastal alustanud Ameerika Ühendriikide riikliku standardite ja tehnoloogia instituudi „Face Recognition Vendor Test" (FRVT), sama asutuse 2004. aastal alustanud „Face Recognition Grand Challenge" (FRGC), mille raames selgitati välja, et 2006. aastaks oli masinõppel põhinev isikute näotuvastuse täpsus suurenenud 2002 aastaga võrreldes 10 ning 1995 aastaga võrreldes 100 korda [7]. Sarnaste programmidega on jätkatud ning on välja töötatud järjestikused väljakutsed nagu Iris Challenge Evaluation (ICE), Multiple Biometric Grand Challenge (MBGC) ja Multiple Biometric Evaluation (MBE), et edasi toetada biomeetriliste tarkvaraprojektide arengut, mis suudaksid täpseid tulemusi anda ka mitteideaalsete sisenditega olgu selleks näiteks madala kvaliteediga pildid [8].

3 Optiline märgituvastus

Optiline märgituvastus on pildil kujutatava käsitsi kirjutatud, trükitud või prinditud teksti mehaaniline või elektrooniline konversioon arvuti jaoks arusaadavale kodeeritud kujule [9]. OCR tehnoloogia on laialdaselt kasutusel igasuguste dokumentide digitaalsel kujul jäädvustamiseks ning töötlemiseks, olgu selleks kas arved, pangateated, lepingud või muud sellised dokumendid. Kui varajased OCR lahendused töötasid tähthaaval ühe kindla šriftiga, siis modernsed arenenud süsteemid suudavad konverteerida täpselt pea kõiki kirjatüüpe ning ka käsitsi kirjutatud tekste. Lisaks suudavad tänapäeva lahendused üle kanda ka teksti vormistust ning jaotust näiteks pealkirjadeks ja lõikudeks.

3.1 Optilise märgituvastuse ajalugu

Erinevalt näotuvastusest algas OCR tehnoloogiate areng pea 100 aastat varem – 1870ndatel. Kõige esimesed lahendused oli mõeldud pimedate inimeste abistamiseks lugemisel. 20. sajandi keskel leiutatakse esimesed tööstuslikud seadmed, millega oli võimalik tõlgendada näiteks morsetähestikku.

Joonis 2. Optacon.

Natuke hiljem, 20. sajandi kolmandal veerandil ilmusid turule esimesed personaalsed kaasaskantavad OCR seadmed, näiteks Optacon [10], mis konverteeris tavateksti punktkirja, võimaldades pimedatel inimestel lugeda igasuguseid trükitud materjale.

Joonis 3. 1968 aastal väljastatud OCR-A font.

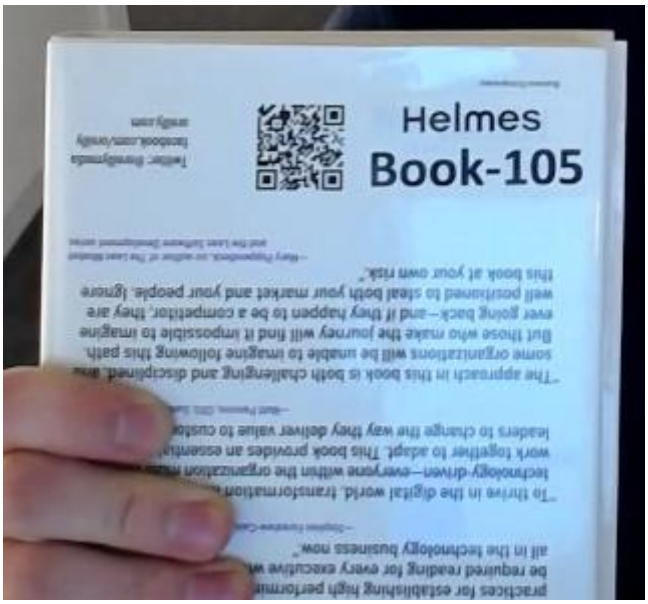
Samuti hakati looma spetsiaalseid kirjatüüpe, et hõlbsustada OCR lahenduste tööd näiteks OCR-A. Alates aastast 1974 hakati OCR skännereid laialdaselt kasutama hinnasiltide ning passide lugemisel. Samuti loodi mitmed tekstituvastuse arendamisega seotud firmad nagu Caere Corporation, ABBYY ja Kurzweil Computer Products Inc. Viimane nendest tuleb välja esimese lahendusega, mis suudab mitmeid erinevad šrifte konverteerida. 2000ndates on kättesaadavad tasuta kasutavad OCR lahendused Adobe Acrobat, WebOCR või Google Drive [11].

4 Olemasoleva lahenduse kirjeldus

2018 aasta suve algul alustasin laenutussüsteemi arendamisega Helmes AS kontoris olemasoleva raamatukogu jaoks. Põhieesmärgiks seadsin luua süsteemi, mis muudab töötajate jaoks raamatute laenutusprotsessi automaatseks ja mugavaks. Eesmärgiks oli luua süsteem 2-nädalaga mis oli jaotatud kahekuisele ajaperioodile. Varem toimus laenutamise protsess käsitsi e-kirjade vahendusel. Tulevikus pidi süsteem võimaldama ka muude esemete, näiteks väikeelektroonika laenutamist.

4.1 Tehnoloogiad

Prototüübi lõin *Spring Boot* serverirakendusena, mille kasutajaliidese realiseerisin *Javascript*-i ja *HTML/CSS*-i kooslusena. Andmebaasina kasutasin prototüübi jaoks MariaDB lahendust. Füüsiliste seadmete poole pealt seati üles puutetundlik ekraan seinal ning veebikaamera selle ekraani kohal. Samuti sai märgistatud kõik süsteemis ringlevad raamatud vastava raamatu identifitseeriva kleepsuga.



Joonis 4. Raamat Helmes ID kleepsuga.

Arhitektuuriliselt oli prototüüplahendus tugevalt seotud Helmese sisemise infrastruktuuriga. Tarkvara kasutas välise info jaoks Helmes-e sisemisi API-sid, seal hulgas eraldi API-isid raamatute, kasutajate piltide ning kasutajate andmete jaoks.

Näotuvastuse jaoks hakkas prototüüp kasutaja *Microsoft Azure Cognitive Services* teenuste seast *Face API* teenust, ning raamatute tuvastamiseks samast lahenduste paketist *Computer Vision API OCR* teenust.

4.1 Funktsionaalsus

Loodud rakenduse põhifunktsionaalsus pidi võimaldama kasutajal võtta raamaturiiulist raamatu, liikuma soovitud raamatuga ekraani juurde ning vajutama ekraanil laenutamise nuppu, misjärel kuvab ekraan veebikaamera jäädvustatava video ekraanile, mille järgi kasutaja saab enda näo ning raamatu ID kleepsu kaamera vaatevälja sättida. Kasutaja olles vastavad kohendused teinud vajutab ekraanil tuvastamise nupule. Kasutajast ning raamatust tehakse pilt, mis saadetakse tarkvara poolt kõige pealt näotuvastuseks *Microsoft Face API*-ile ning seejärel *Microsoft Computer Vision API*-ile raamatu ID kindlaks tegemiseks. Kui mõlemad operatsioonid on edukad, kuvatakse tulemused ekraanile ning kasutaja saab otsustada kas tulemused on õiged ning kinnitada laenutuse või siis tegevuse katkestada ja soovi korral uuesti proovida. Raamatute tagastamise protsess on tarkvara mõistes identne välja arvatud, et kasutaja enda edukas tuvastus pole nõutud.

4.2 Probleemid

Prototüüplahenduse suurimaks probleemiks oli kasutajate kogemuste põhjal raamatu tuvastamisega. Tihti ei suutnud *Computer Vision API OCR* teenus piisavalt efektiivselt tuvastada raamatu ID kleepsult teksti ning raamatu pidi asetama kaamerale üsna lähedale, et oleks mõistlikult hea võimalus raamatu tuvastamise õnnestumiseks. Samuti jätsin arendusprotsessi kiirustades kasutamata mitmed võimalused optimeerida tegevuste kiirust, näiteks toimuvad päringud näotuvastuse ning OCR API-dele sünkroonselt, kuigi selleks mõjuvat põhjust ei ole.

Näotuvastuse poolelt töötas süsteem umbkaudu 70% juhtudel, kus töötaja üles laetud pildil on olemas selge näokujutis, millelt puuduvad igasugused nägu katvad esemed nagu näiteks päikeseprillid. Prototüüp töötas ka juhtudel kus pildil oli lisaks näole inimese kogu keha. Kuid töötajad kellel piisavalt sobiv pilt puudus ei olnud mugavat võimalust endast otstarbekam näopilt süsteemi sisestada ning seeläbi edukalt süsteemi kasutada.

5 Tehnoloogiate valik ja võrdlus

Olemasoleva süsteemi korral jäi enne arendamisega alustamist kiire arenduse huvides analüüsimata, millised välised tehnoloogiad oleksid kõige sobivamad ning efektiivsemad just antud süsteemi jaoks. Võimalikult hea uue arenduse huvides teen seda nüüd. Valin välja 3 näotuvastuse tehnoloogiat ning 3 potentsiaalset lahendust raamatute tuvastamiseks ning võrdlen nende võimekusi eksperimentidega, mis esindavad võimalikke probleemseid sisendeid, mis süsteemi kasutamisel võivad ette tulla.

5.1 Eksperimendid näotuvastustehnoloogiate valikuks

Näotuvastuse lahendite võrdlusesse valisin juba olemasolevas süsteemis kasutusel oleva *Microsoft*-i lahenduse. Sellele lisaks väga populaarse *OpenCV*, mis on lokaalselt töötav lahendus võimekusega sooritada näotuvastust ka reaalajas videovooga. Kolmandaks valikuks on Hiina *Face++* teenus. Tehnoloogiad valisin välja arvestades nende populaarsust arendajate seas ning *Java* serverirakendusega integreerimise lihtsust.

5.1.1 Microsoft Face API

Microsoft pakub oma pildianalüüsi lahenduste seas väga võimekat näotuvastuse lahendust, mis sai kasutusele võetud ka antud projekti prototüübis. API võimaldab mitme näo tuvastust pildilt, inimese kahe näopildi järgi verifitseerimist, sarnaste portreede otsingut, nägude grupeerimist ning ka nägude identifitseerimist [12].

Tasuta paketis on päringute arvuks 20 päringut minutis, kuni 30 000 päringut kuus. Tasuline pakett võimaldab 20 päringut sekundis ning päringute hind varieerub vastavalt kasutusele alates 0,338 kuni 0,844 eurot tuhande päringu kohta. Mida suurem on tehtud päringute arv, seda madalam on hind [13].

5.1.2 OpenCV

OpenCV on vaba lähtekoodiga arvuti nägemisvõimekust kasutav tarkvara, mis sisaldab endas üle 2500 optimeeritud vanemat ning tänapäevast pildianalüüsi ja masinõppe algoritmi. Sellel on maailmas üle 47 000 kasutajaga kommuun ning seda kasutatakse ulatuslikult nii ettevõtetes nagu *Google, Yahoo, Microsoft, Intel* kui ka teadusuuringutes ja riiklikes asutustes [14].

Antud töö alternatiivide seast on *OpenCV* ainus vabavarana saadaval, seega täiesti tasuta ning lokaalselt töötav lahendus.

5.1.3 Face++

Face++ Cognitive Services on süvaõppel põhinevat piltide analüüsi pakkuv platvorm. Teenus on Hiinas tehisintellekti arendamisega tegeleva idufirma *Megvii* üks märkimisväärsemaid tooteid. Alates aastast 2012 arendatud süsteem on arendajate seas saavutanud suure populaarsuse ning on võitnud mitmeid maailmaklassi võistlusi näotuvastuse valdkonnas, nende seas *FDDB*, *300-W* and *LFW* [15].

Platvormi eelisteks on lihtne kasutatavus ning kättesaadavus arendajatele. Tasuta paketis on võimalik luua üks tasuta API võti ning päringute hulk on piiratud kolmele sekundis, mis on antud projekti jaoks piisav. Tasulise paketi eest on võimalik tasuda kasutatud mahtude põhiselt, 13.05.2019 on päringu hinnaks 0,0001 USA dollarit [16].

5.1.4 Eksperimentide läbiviimine

Väliste näotuvastuslahenduste analüüsimiseks sooritasin 8 eksperimenti. Eksperimendid püstitasin vastavalt reaalses süsteemis potentsiaalsetele problemaatilistele sisenditele. Kõik testitavad lahendused treeniti identselt *Yale*-i nägude andmebaasiga [17], kuhu lisasin enda portreepildi aastast 2014, mis toob ekperimentides välja ka olukorra, kus kasutaja treenitakse mitmeid aastaid vana pildiga. Eksperimentide jooksul mõõtsin nii protsessiks kuluvat aeg kui ka teenuse enesekindlust väljastatud tulemusega. Erinevalt kahest teisest alternatiivist ei tagasta *OpenCV* enesekindlust protsentides vaid hoopis tuvastatava näo ning treenitud näo tuvastamiseks kasutatavate orientiirpunktide vahemaana. See tähendab, et väiksem väärtus tähendab suuremat enesekindlust. See väärtus ei ole otseselt võrreldav teiste lahendustega, seega antud töö jaoks selgitasin välja peaaegu identsete piltide võrdluses, et aktsepteeritav enesekindluse väärtus *OpenCV* puhul on vähem kui 55. *Microsoft*-i ning *Face++* teenustele seadsin aktsepteeritavaks väärtuseks 60%. Aja limiidiks määrasin kõigile tehnoloogialetele ärilistes nõuetes seatud 3 sekundit. Piltide taustad on neutraalset tooni, et vältida sisendis liigset müra.

Esimene eksperiment testis lahenduse võimekust tuvastada kasutajat treenimisfotoga sarnases poosis ja valguses tehtud portreepildiga. Kõige edukamalt esines *Face++* lahendus, mis tagastas õige tulemuse 1,651 sekundiga ning enesekindlusega 87.502%.

Microsoft-i *Face API* tulemus oli pea sekund aeglasem ajaga 2,757 ning enesekindlusega 59,098%. *OpenCV* saavutas ajaga 1,168 sekundit enesekindluse 89,05, mis on küll kõige kiirem aeg, kuid enesekindlus on üle akstsepteeritava väärtuse piiri.

Joonis 5. Esimese eksperimendi portreed. Vasakul treenitud portree ja paremal eksperimendi portree.

Teine eksperiment sarnanes esimesele, kuid pildil kannab testitav kasutaja läbipaistva raamiga lugemisprille. *Face++* esines sarnaselt esimesele eksperimendile kõige paremini saavutades enesekindluse 87,327% ajaga 2,577 sekundit. *Microsoft Face API* enesekindlus langes võrreldes esimese eksperimendiga 54,782 protsendile, mis saavutati ajaga 1,472 sekundit ning *OpenCV* tulemus paranes, kuid ei saavutanud aktsepteeritavat väärtust enesekindlusega 80,022 ajaga 1.415 sekundit.

Joonis 6. Teise eksperimendi portreed. Vasakul treenitud portree ja paremal eksperimendi portree.

Kolmandas eksperimendis kannab testitaval portreel kasutaja päikeseprille. Antud sisendiga saab edukalt hakkama vaid *Face++* teenus, mis saavutab kõrge enesekindluse 81,436% ajaga 1,651 sekundit. Nii *Microsoft Face API* kui ka *OpenCV* ebaõnnestuvad tuvastamisel vastavalt aegadega 1,185 sekundit ja 1,004 sekundit.

Joonis 7. Kolmanda eksperimendi portreed. Vasakul treenitud portree ja paremal eksperimendi portree.

Neljandas testis suurendasin distantsi kasutaja ning kaamera vahel. Jätkuvalt oli kõige võimekam *Face++*, mis saavutas 1,264 sekundiga 88,425 protsendilise enesekindluse. *Microsoft Face API* võimekus paranes sooritades 1,344 sekundiga 63,876 protsendise

enesekindluse ning *OpenCV* sai õige tulemuse 1,671 sekundiga, kuid enesekindlus suurenes arvuni 115,899, mis on märkimisväärselt üle aktsepteeritava piiri.

Joonis 8. Neljanda eksperimendi portreed. Vasakul treenitud portree ja paremal eksperimendi portree.

Viienda eksperimendi korral on pildil kasutaja nägu osaliselt kaetud raamatuga. Kuna test sarnaneb päikeseprillide testiga, siis tulemused on ka vastavalt sarnased. *Face++* jätkab sarnaselt edukalt saavutades 82,066 protsendilise enesekindluse 3,037 sekundiga, mis on veidi üle aktsepteeritava 3 sekundi limiidi. *Microsoft*-i *Face API* ning *OpenCV* mõlemad ebaõnnestuvad vastavalt aegadega 1,686 sekundit ja 0,89 sekundit.

Joonis 9. Viienda eksperimendi portreed. Vasakul treenitud portree ja paremal eksperimendi portree.

Järgmised kaks eksperimenti katsetavad tuvastuse võimekust erinevates poosides. Esiteks kuuendas testis on kasutaja nägu pööratud umbes 45 kraadi küljele. *Face++* jätkab edukat trendi saavutades 79,437% enesekindluse ajaga 1.918 sekundit. *Microsoft*-i tulemus on veidi kehvem seni saavutatud keskmisest tulemusega 51,832% , kui ajaks mõõtsin 1,582 sekundit. *OpenCV* tuvastus ebaõnnestus ajaga 0,91 sekundit.

Joonis 10. Kuuenda eksperimendi portreed. Vasakul treenitud portree ja paremal eksperimendi portree.

Seitsmenda testi pildil on kasutaja nägu suunatud 45 kraadi üles. Tulemused on üpris sarnased eelmise eksperimendiga, kus *Face++* saavutab peaaegu identse enesekindluse 80,747% ajaga 1,614 sekundit. *Microsoft Face API* tulemus langeb märkimisväärselt alla aktsepteeritud 60% piiri 42,703 protsendini ajaga 1,468 sekundit ning *OpenCV* taas ebaõnnestub, sel korral ajaga 1,226 sekundit.

Joonis 11. Seitsmenda eksperimendi portreed. Vasakul treenitud portree ja paremal eksperimendi portree.

Viimases eksperimendis katsetatakse teenuste võimekust ebaideaalsetes valgustingimustes, ehk portree tegemisel asetasin madala võimsusega valguse näo vasakule poolele, mis tekitab suure varju näopildi paremale küljele. Kõik lahendused esinevad sarnaselt tavalise näopildi eksperimendile. *Face++* saavutab enesekindluse 86,865% ajaga 3,196 sekundit. Tuleb märkida, et aeg jääb üle aktsepteeritud 3 sekundi piiri. *Microsoft Face API* saavutab sarnaselt esimesele testile enesekindluse 56,386% ajaga 2,117 sekundit ning *OpenCV* peegeldab samuti esimest testi tulemusega 82,418 ajaga 1,242 sekundit.

Joonis 12. Kaheksanda eksperimendi portreed. Vasakul treenitud portree ja paremal eksperimendi portree.

5.1.5 Kokkuvõte

Üleüldiselt kõige paremini antud eksperimentides esineb *Face++* API langedes napilt alla 80 protsendise enesekindluse ainult ühes testis ning koguni kuues testis jäi aeg alla lubatud piiri. Aja piiri ületas lahendus kahel korral, kuid ka siis vaid 0,2 sekundiga kõige rohkem. Tuvastamine ei ebaõnnestunud mitte ühelgi korral. *Microsoft Face API* tulemused olid võrreldes *Face++* lahendusega kesised. Enesekindlus jäi läbivalt alla 60 protsendi saavutades kõrgpunkti 63,876 protsendiga. Ajaliselt jäid kõik kaheksa ekperimenti alla lubatud 3 sekundi piiri, kuid peab mainima, et kahel korral tuvastamine ebaõnnestus. Kõige kehvemini esines *OpenCV*, mis näitas küll ilma võrgu päringuteta kõige kiiremaid aegu, kuid ebaõnnestus tuvastamisega pooltes eksperimentides. Testides, kus kasutaja tuvastamine õnnestus ei saavutatud mitte ühelgi korral aktsepteeritavat enesekindluse tulemust, milleks oli väiksem kui 55.

	Face++	OpenCV	Microsoft Face API
1. Tavaolukord	1,651 s. 86,502%	1,168 s. 89.05	2,747 s. 59,098%
2. Tavaolukord lugemisprillidega	2,577 s. 87,327%	1,415 s. 80	1,472 s. 54,782%
3. Tavaolukord päikeseprillidega	1,651 s. 81,436%	1,004 Ebaõnnestumine	1,185 s. Ebaõnnestumine
4. Kauge	1,264 s. 88,425%	1,671 s. 115,899	1,344 s. 63,876%
5. Kaetud	3,037 s. 82,066%	0,89 s. Ebaõnnestumine	1,686 s. Ebaõnnestumine
6. Näo poos kõrvale	1,918 s. 79,437%	0,91 s. Ebaõnnestumine	1,582 s. 51,832%
7. Näo poos üles	1, 614 s. 80,747%	1,226 s. Ebaõnnestumine	1,468 s. 42,703%
8. Vähene valgus	3,198 s. 86,865%	1,242 s. 82,418	2,117 s. 56,386%

Tabel 1. Näotuvastuse eksperimentide tulemused

Tulemuste tabelis on märgitud rohelisega ajalimiidi sisse mahtunud ning aktsepteeritava enesekindluse väärtusega tulemused. Kollasega on märgitud tulemused mis tuvastasid pildilt küll õige näo, kuid ei suutnud kas rahuldada 3 sekundilist ajalist nõuet või tulemuse

enesekindlus nõuet. Punasega on märgitud tulemused mis ei suutnud tuvastada pildi pealt õiget nägu või mitte ühtegi nägu.

5.1.6 Kasutajatestimine *Face++* tehnoloogia valideerimiseks

Loodud rakenduse lõplikuks valideerimiseks viisin läbi 5 isikuga kasutajatestimise Helmes AS kontoris. Esiteks treeniti näotuvastuseks kõikide kasutajate Helmese sisesüsteemi profiilipiltidega. Seejärel proovisid kasutajad kas süsteem tunneb neid ära. Protsessi käigus mõõtis rakendus kulunud aega ning salvestas selle koos näotuvastuse tulemusega. Igat kasutajat testisin kahe erineva pildiga. Tulemused on esimese treenimise kohta head. Ükski tuvastamine ei ületanud ärilistes nõuetes määratud 3 sekundilist aja piiri ning pooled tuvastamised toimusid juba enesekindlusega mis ületas nõutud 90% piiri, kusjuures vaid üks tuvastus jäi 84,027 protsendiga alla 85% piiri. Tulemuste tabelis on märgitud rohelisega ärinõuetele vastavad tulemused ning kollasega on märgitud tulemused, mis jäävad enesekindlusega alla nõutud 90% piiri.

	Foto 1	Foto 2
Kasutaja 1	2,123 s. 89,609%	1,433 s. 92,01%
Kasutaja 2	1,769 84,027%	1,398 s. 86,706%
Kasutaja 3	1,522 s. 92,012%	1,524 s. 90,764%
Kasutaja 4	1,62 s. 88,107%	2,056 s. 89,629%
Kasutaja 5	1,372 s. 90,511%	1,475 s. 90,57%

Tabel 2. Kasutajatestimise tulemused peale 1. treenimist.

Peale esimest kasutajatestimise vooru treenisin näotuvastuse mudelit uuesti juba süsteemi poolt varasematel nädalatel salvestatud samade kasutajate piltidega ning korraldasin samad testid uuesti, et välja selgitada kuidas süsteemi võimekus muutub. Peale teist treenimist paranesid absoluutselt kõikide kasutajate näotuvastuste tulemused ning nüüd rahuldasid tuvastamised seatud 90 protsendilise enesekindluse nõuet. Taoline treenimine hakkab süsteemi kasutades toimuma automaatselt iga laenutamisega. Seega võib järeldada, et ka uuendatud süsteemi poolt on tagatud üle 90%-ine näotuvastuse enesekindlus. Tulemuste tabeli legend on sama esimese vooru tulemuste tabeliga.

	Foto 1	Foto 2
Kasutaja 1	2,157 s. 95,064%	1,946 s. 96,678%
Kasutaja 2	1,672 s. 93,78%	1,875 s. 93,326%
Kasutaja 3	1,84 s. 93,956%	1,95 s. 94,431%
Kasutaja 4	1,669 s. 91,964%	1,908 s. 90,806%
Kasutaja 5	1,635 s. 94,471%	1,616 s. 94,896%

Tabel 3. Kasutajatestimise tulemused peale 2. treenimist.

5.2 Eksperimendid raamatutuvastustehnoloogiate valikuks

Raamatute tuvastamiseks hakkan kasutama OCR lahendust, mis hakkab lugema raamatute tagumisele kaanele kleebitud ID koodi. Valikusse võtsin olemasoleva süsteemi *Microsoft*-i *OCR* teenuse, *Microsoft*-i teise tekstituvastuse teenuse nimega *recognizeText* ning *Google Cloud Vision API*. Sarnaselt näotuvastustehnoloogiatele valisin teenused välja arvestades populaarsust ning *Java* integratsiooni lihtsust.

5.2.1 Microsoft Computer Vision API

Microsoft-il on teksti tuvastamiseks kaks erinevat teenust – *OCR* ja *recognizeText*. Esimene on kergekaalulisem ning väiksema võimekusega, kuid kiirem. Teenusega on mahutatud tuvastamise protsess ühe päringu sisse, ehk tulemused tagastatakse küsiva päringu vastusena. *OCR* teenuse hind jääb alates 1.265 kuni 0.549 eurot 1000 päringu kohta [17].

Kulukam, kuid võimekam *recognizeText* teenus suudab teksti tuvastada palju suurema võimekusega. Uuem teenus suudab teksti ära tunda suuremate arvu šriftide korral ka ebaideaalsetes tingimustes, milleks võivad olla näiteks pildifaili ebakvaliteetsus või teksti osaline kinnikaetus. *recognizeText* teenuse tööks kulub aga rohkem aega ning ei tööta ühe päringu piires. Olles saatnud küsiva päringu peab süsteem pidevalt saatma uusi päringuid kuni API tagastab tulemustega vastuse. Teenuse hinnaks on alates 2.109 eurot 1000 päringu kohta [18].

5.2.2 Google Cloud Vision API

Antud töös analüüsin raamatute tuvastamiseks ka *Google* teenust nimega *Vision API*, milles kasutatakse eel-treenitud masinõppe mudeleid, et pildianalüüsi käigus avastada objekte, nägusid, trükitud ning käega kirjutatud teksti ja koostada selle käigus piltidele sügavalt iseloomustavad metaandmed. *Google* pakub oma pilveteenuste seas ka teist masinõppel põhinevat arvuti nägemisoskust kasutavat teenust - *AutoML Vision*, millega on võimalik treenida oma enda isiklike eripäradega piltide analüüsi mudelid. Teenus võimaldab ka läbi graafilise kasutajaliidese isiklike mudelite täpsust, viiteaega ning suurust optimeerida ja neid mudeleid kasutamiseks eksportida [19]. Käesolevas töös on aga *Vision API* teenus piisava võimekusega seega spetsiifilise mudeli treenimine pole vajalik.

Kui kuni 1000 päringuni on *Google* tekstituvastus teenus tasuta, siis enamate päringute korral on teenuse maksumus 1,5 USA dollarit. Väga suurte päringute arvuga, milleks on rohkem kui 5 miljonit langeb hind 0,6 USA dollarini kuus [18].

5.2.3 Eksperimentide läbiviimine

OCR lahenduste analüüsi kaasati *Microsoft*-i mõlemad *OCR* ja *recognizeText* teenused ning nende kõrvale ka *Google Cloud Vision API* tekstituvastus teenus. Lahenduste testimiseks korraldasin 7 eksperimenti, millega üritasin ära katta paljusid probleemseid sisendeid ja piirjuhtusid, mis võivad reaalselt süsteemi kasutamisel tekkida. Piltide taustad on neutraalsed, et vältida eksperimentide sisendites liigset müra. Tulemuste analüüsis vaatlen, kas teenus suutis edukalt lugeda pildilt raamatu ID kleepsu. Antud juhul valiti raamatu ID-ks „Book-94”, mis peab olema ka eeldatav õnnestumise tulemus. Ajalimiidiks püstitasin sarnaselt näotuvastusega äriliste nõuetele vastavalt 3 sekundit.

Esimeses testis asetasin raamatu pildile normaalsele kaugusele kaamerast, milleks on umbes 1 meeter. Kõige edukam testis oli *Microsoft*-i *OCR* teenus, tuvastades eeldatava tulemuse „Book-94-” ajaga 0,703 sekundit. *Google Cloud Vision API* tulemus oli identne, kuid aega kulus veidi rohkem, 1,051 sekundit. Õnnestunud tulemused saavutas ka *Microsoft*-i teine *recognizeText* teenus, mis suutis tuvastada teksti raamatu ID kleepsul ning lisaks ka palju muud teksti raamatu kaanelt. *Microsoft*-i *recognizeText* teenusel kulus tuvastuseks aega 2,982 sekundit, mis mahub napilt nõutud aja piiridesse.

Joonis 13. Esimese tekstituvastuse eksperimendi pilt.

Teises testis pöörasin raamatut 90 kraadi küljele ning pildi tegin samalt kauguselt. *Microsoft*-i *OCR* teenus saavutab veidi pikema ajaga, 1,082 sekundit, identselt esimese ekperimendiga tuvastamise tulemuse „Book-94-". *Google Cloud Vision API* 2,21 sekundiga saavutatud tuvastus sisaldab rohkem teksti kui esimesel testil, mille seas on olemas ka vajalik ID kirje. *Microsoft*-i *recognizeText* API suutis samuti ID tuvastuse edukalt sooritada, kuid aega selleks kulus lubatust rohkem, 5,284 sekundit.

Joonis 14. Teise tekstituvastuse eksperimendi pilt.

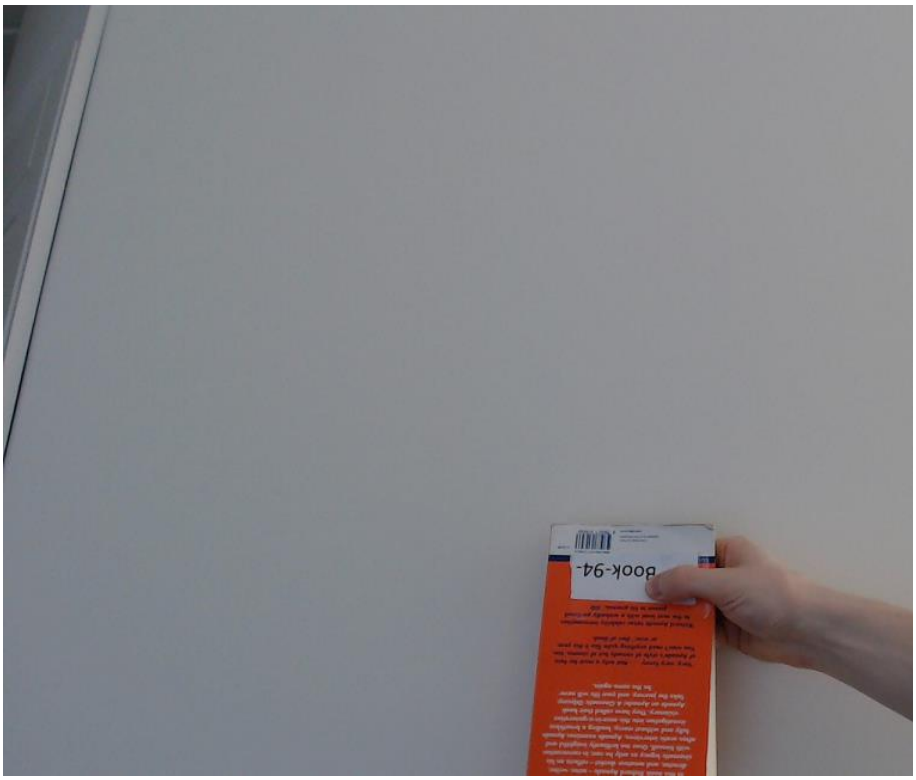
Kolmandas testis pöörasin raamatu veel 90 kraadi, et raamat oleks täiesti tagurpidi samal kaugusel mis esimesel kahel testil. Nii *Microsoft*-i *OCR* teenus kui ka *Google Cloud Vision API* tuvastasid identselt ainult vajaliku teksti „Book-94-" vastavalt aegadega 0,77 ja 1,604 sekundit. *Microsoft*-i *recognizeText* teenus tagastas teksti rohkem, mille hulgas oli olemas vajalik ID tekst. Aega läks API-l 7,407 sekundit, mis ületab lubatud normi kahekordselt.

Joonis 15. Kolmanda tekstituvastuse eksperimendi pilt.

Neljandas eksperimendis viisin raamatu kaamerast umbes kolme meetri kaugusele, et testida teenuste võimekust juhul kui raamatu ID tekst on väiksem ning halvemini nähtaval. Testis esines edukalt vaid *Google* lahendus tuvastades minimaalse vajaliku teksti „Book-94". Aega läks API-l lubatust rohkem 4,306 sekundit. Mõlemad *Microsoft*-i lahendused ebaõnnestusid. Kui *recognizeText* teenus suutis tuvastada pildilt teksti „$6-3008" 2,768 sekundiga, siis *OCR* teenus ei leidnud pildilt üldse teksti ajaga 2,068 sekundit.

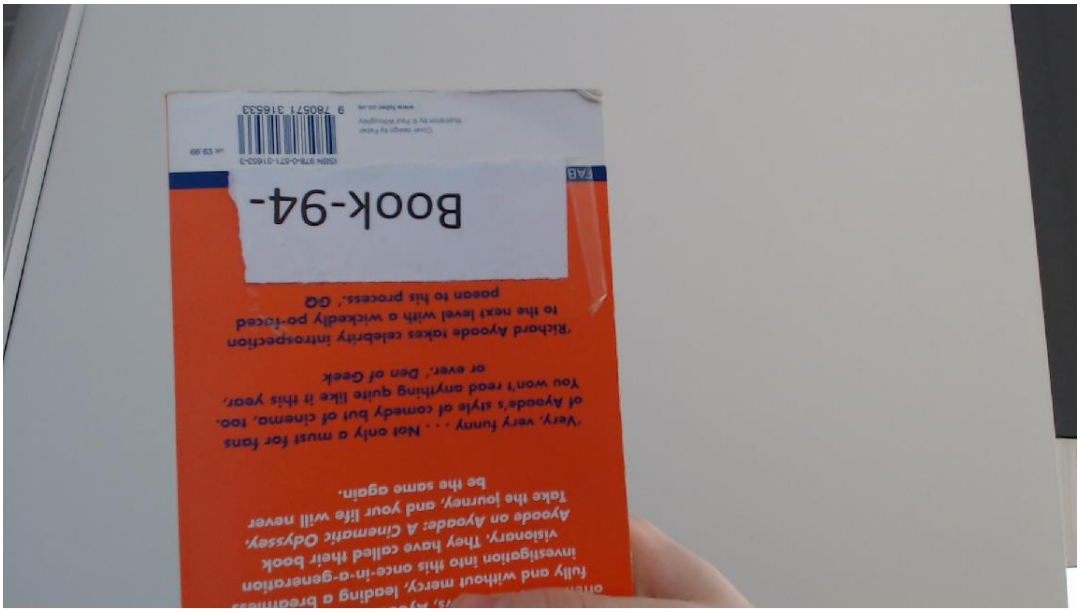
Joonis 16. Neljanda tekstituvastuse eksperimendi pilt.

Viienda eksperimendi eesmärgiks oli välja selgitada, kuidas teenused saavad hakkama ID teksti osalise kaetusega, ehk API peab olemasoleva infoga „ennustama", mis sümboliga võiks tegemist olla. Kõige edukam antud testis oli *Google* lahendus, mis suutis piisava ID teksti tuvastada 1,591 sekundiga. *Microsoft*-i *recognizeText* teenus suutis ajalimiiti ületades 5,284 sekundiga tuvastada teksti, milles sisaldus piisab ID kirje ning *OCR* teenus tuvastas ebaõnnestunult teksti „dook-94-" 0,691 sekundi jooksul.



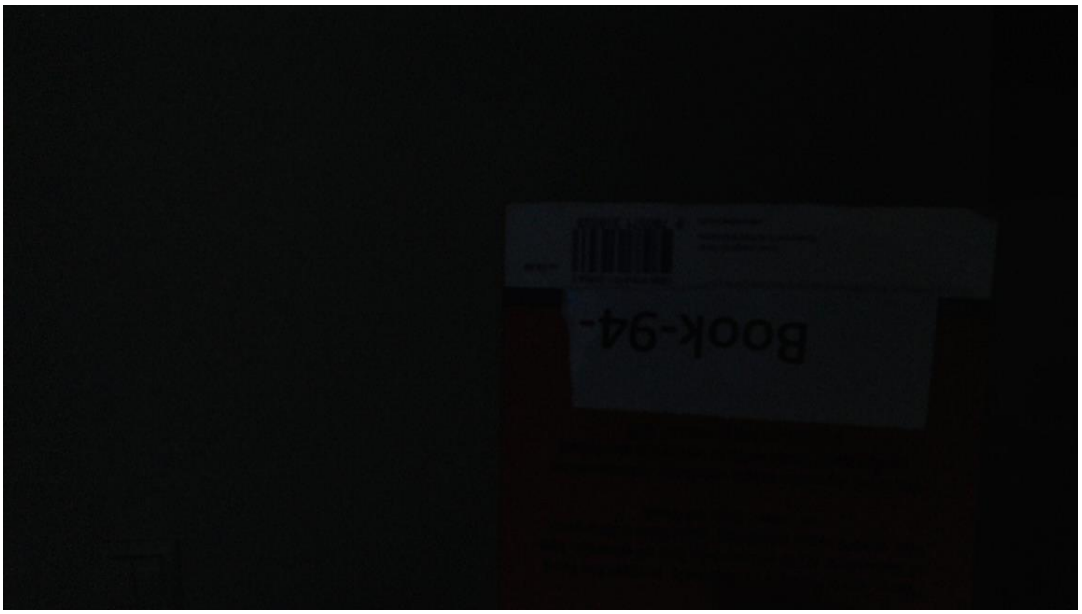
Joonis 17. Viienda tekstituvastuse eksperimendi pilt.

Kuuenda testi mõte oli tuua raamat kaamerale väga lähedale ning katsetada kuidas saavad teenused ajaliselt hakkama, kui kergesti tuvastatavat teksti on palju. Testis olid edukad *Google* ja *Microsoft*-i *OCR* teenus tuvastades muu seas õige ID kirje vastavalt 2,681 ja 1,479 sekundiga. *Microsoft*-i *recognizeText* tuvastas ID teksti edukalt, kuid teenus ületas 3,594 sekundiga lubatud aja limiidi.



Joonis 18. Kuuenda eksperimendi pilt.

Viimase eksperimendi käigus testisin lahenduste võimekust ekstreemselt madala valgustuse korral. Ainukesena sai ülesandega edukalt hakkama *Google* lahendus. Tuvastades minimaalselt vaja mineva teksti „Book-94” 4,879 sekundiga, mis ületab lubatud ajalist piiri. Microsofti *OCR* teenus ei tuvasta pildilt 2,21 sekundi jooksul mitte midagi ning teine *recognizeText* teenus tuvastab 4,312 sekundiga teksti „-16-yoog”, mis ei ole korrektne.



Joonis 19. Seitsmenda tekstituvastuse eksperimendi pilt.

5.2.4 Kokkuvõte

Kokkuvõtvalt esines üle seitsme testi kõige edukamalt *Google Cloud Vision API*, mis sai edukalt hakkama kõigis eksperimentides ületades ajalimiidi kahel juhul, kuid mitte rohkem kui 1,9 sekundit. Microsoft-i *recognizeText* teenus saavutas eduka tulemuse viies testis, ületades ajalimiidi neljal korral, kusjuures kõige kauem kulus vasteks 7,407 sekundit. *Microsoft*-i kiirem *OCR* teenus oli edukas vaid neljas eksperimendis kuid kõik edukad sooritused olid ajalimiidi sees.

	Google Cloud Vision API	Microsoft Computer Vision API recognizeText	Microsoft Computer Vision API OCR
1. Tavaolukord	1,051 s. OK	2,982 s. OK	0,703 s. OK
2. Küljel	2,21 s. OK	5,938 s. OK	1,082 s. OK
3. Tagurpidi	1,604 s. OK	7,407 s. OK	0,77 s. OK
4. Kaugel	4,306 s. OK	2,768 s. Ei ole OK	2,068 s. Ei ole OK
5. ID kaetud	1,591 s. OK	5,284 s. OK	0,691 s. Ei ole OK
6. Lähedal	2,681 s. OK	3,594 s. OK	1,479 s. OK
7. Vähe valgust	4,879 s. OK	4,312 s. Ei ole OK	2,21 s. Ei ole OK

Tabel 4. Tekstituvastuse eksperimentide tulemused.

Tulemuste tabelis on rohelisega märgitud õnnestunud ID kirje tuvastamised 3 sekundilise ajalimiidi piires. Kollasega on märgitud edukad tuvastamised millele kulus aega enam kui 3 sekundit ning punasega on märgitud tulemused mille korral korrektse ID kirje tuvastamine ebaõnnestus.

6 Realisatsioon

6.1 Eesmärgid

Antud lõputöö jaoks hakkasin olemasolevat prototüüpi edasi arendama. Uue arenduse põhieesmärkideks on lahendada võimalikult palju prototüübi probleeme ja see lahti siduda kõikidest Helmese sisesüsteemidest ning viia kogu lahendus täielikult eraldiseisvale kujule, mis võimaldaks selle kasutusele võtmist ka teistes raamatukogudes nagu näiteks Uhtna raamatukogu.

6.2 Nõuded

Uue arenduse nõueteks on nii kasutaja kui ka raamatu tuvastusprotsessi maksimaalseks ajaliseks pikkuseks 3 sekundit, see on minimaalne aeg millega minu kogemustel kasutatavad tehnoloogiad järjepidevalt tuvastamise ülesannetega hakkama saavad.

Kogu laenutusprotsess peab võtma aegu kuni 10 sekundit. See aeg algab siis, kui kasutaja vajutab kasutajaliideses nuppu „Borrow“ või „Return“. 4 sekundit ajast antakse kasutajale enda näo ning raamatu kohendamiseks ning veendumiseks, et edukaks tuvastamiseks on pildil kõik vajalik olemas. 3 sekundit sellest ajast võib kuluda kasutaja ning raamatu tuvastamisele. Ülejäänud 3 sekundit on mõeldud ajaks, mille jooksul kasutaja loeb tuvastuse tulemusi ning vajutab kasutajaliideses vastavat nuppu tegevuse kinnitamiseks või katkestamiseks.

Raamatu laenutaja äratundmisel peab süsteemi enesekindlus olema suurem kui 90%. See on piisavalt suur väärtus, et veenduda isiku identiteedis ning vältida valetuvastusi.

Lahenduse administratiivse komponentidega peab olema võimalik registreerida uusi kasutajaid, ning nende näopildid süsteemi registreerida samades tingimustes sama kaameraga, millega toimub laenutusprotsess. See tagab võimalikult täpse algse treenimise kasutaja tuvastamiseks kasutatavale mudelile. Samuti peab olema võimalik läbi süsteemi administraatoril lisada ja kustutada raamatute andmeid. Nende andmete hulka kuuluvad näiteks raamatu pealkiri, autor, kaanepilt, lühikirjeldus, tuvastamiseks kasutatav ID. Andmeid salvestatakse raamatute konkreetsete eksemplaride kaupa.

Süsteem peab töötama täiesti eraldiseisva süsteemina välja arvatud näotuvastamiseks ja raamatute tuvastamiseks mõeldud teenused, kuid samas lubama mugavat integratsiooni väliste rakendustega vastavalt asutuse vajadustele, mis süsteemi kasutada soovib.

6.3 Tehnoloogiad

Eesmärkide saavutamiseks ning äriliste nõuete rahuldamiseks alustasin kõigepealt kogu koodibaasi ümberkirjutamisega uuematele ning arenduse mõttes mugavamatele tehnoloogiatele. Serverirakenduses varem kasutatud *Spring Framework 4*-ja asendasin *Spring Boot 2*-ga, mis eelkõige lihtsustab rakenduse avaldamist. Puhtal *Javascript*-il põhineva kasutajaliidese asemele võtsin kasutusele palju võimekama Angular 7-e ning andmebaasi tasemel vahetasin *MariaDB* populaarsema *PostgreSQL*-i vastu.

Koodi täieliku ümberkirjutamise eesmärgiks oli ka üleüldine koodi enda parendamine puhta koodi põhimõtetel. Samuti hakkasin kasutama *Docker*-it, mis võimaldab mugavamat süsteemi avaldamist identse rakenduse seadistusega sõltumata avaldamiseks kasutatava serveri operatsioonisüsteemist. Kasutajate autentimiseks, mis prototüübist täielikult puudus hakkasin kasutama *JWT*-il põhinevat lahendust.

6.4 Serverirakendus

Süsteemi edasi arendamise teiseks sammuks sai eelkõige serverirakenduste funktsionaalsuse optimeerimine. Mille käigus kasutasin ära rohkem võimalusi tegevuste asünkroniseeriseks. Selle põhiliseks näiteks on suhtlus komponentidega, mis tegelevad kasutaja ning raamatu tuvastamisega. Uues arenduses toimuvad need tegevused üksteisest sõltumata üheaegselt erinevatel lõimetel. Samuti sai viidud kõik andmete kogumiseks kasutatav andmebaasi suhtlus asünkroonsele kujule, et see ei aeglustaks ebavajalikult kasutajakogemust. Sellised meetmed aitavad saavutada ette seatud 3 sekundilist tuvastusperioodi.

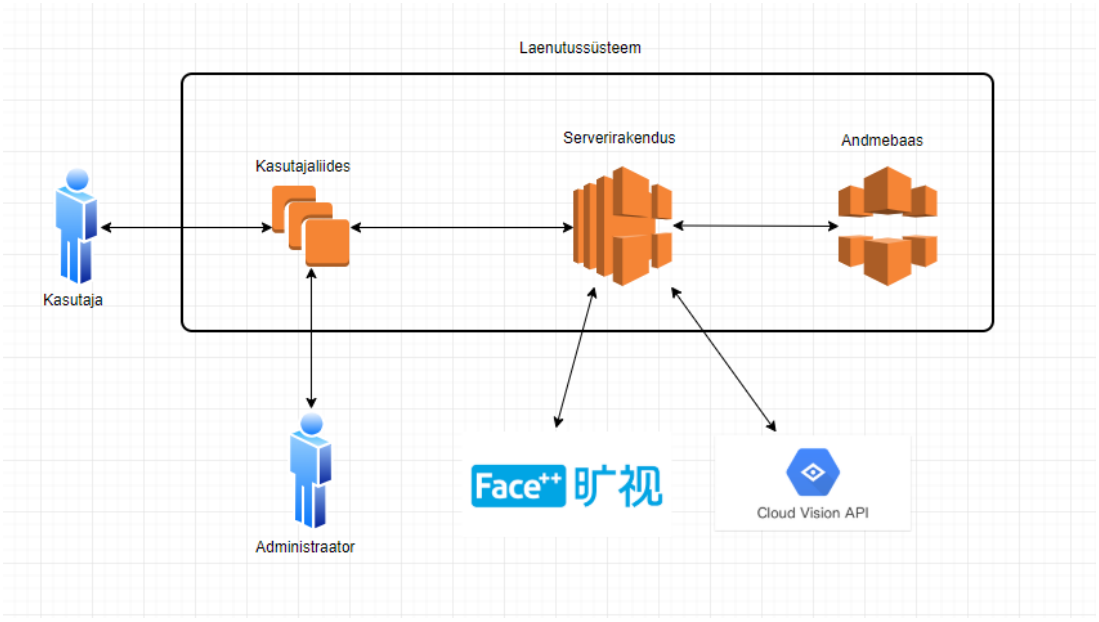
Ärilistes nõuetes seatud 90 protsendilise süsteemi näotuvastuse enesekindluse saavutamiseks lõin süsteemi uues arenduses loogika, kus iga õnnestunud laenutamise järel kasutatakse laenutamise käigus tehtud uut pilti antud kasutaja näotuvastuse mudeli täiustamiseks. Sellise loogikaga toimub järjepidev kasutajate mudelite treenimine, ning süsteemi näotuvastus muutub iga laenutusega paremaks. See tähendab samuti, et kui

kasutaja näo tunnused muutuvad, on siiski näotuvastamise mudel selle jaoks treenitud. Lisaks sellele, kui prototüüplahenduses oli kasutajate treenimine võimalik vaid kasutajate enda üles laetud profiilipiltidega, mis paljudel juhtudel ei olnud piisavalt sobivad näotuvastuse jaoks treenimiseks, siis uues lahenduses on kasutajal võimalik ise ennast treenida just selles keskkonnas, mida ka hiljem raamatute laenutamiseks kasutatakse. See aitab saavutada 90 protsendilist eesmärki märkimisväärselt kiiremini.

6.5 Kasutajaliides

Väljanägemiselt jäi kasutajaliidese pool üldiselt samaks, kuid täielikult sai ümber kirjutatud programmaatiline pool. Kõik vaated jaotati *Angular*-i komponentideks, lõin parema sessiooni halduse ning kasutajate autentimise ning mugavalt hallatava autoriseerimise. Kogu veebirakendus töötab ühel lehekülje mõttel, milles navigeerimiseks kasutatakse komponentide ehk vaadete vahetamist. Arenduses kasutasin ära ka võimalusi modaalide kasutamiseks, näiteks tuvastustulemuste kasutajaliideses kuvamisel. Kõik autoriseerimise reeglid realiseerisin valge nimekirja põhimõtetel. Suhtlus serverirakendusega toimub läbi REST päringute. Kõik päringud on loogiliselt jaotatud kasutajaliidese funktsionaalsel poolel jaotatud loogiliselt teenusteks, millel on serverirakendusel vastavalt kontrollerid.

6.6 Arhitektuur



Joonis 20. Uue arenduse arhitektuur.

Kui Helmes AS kontoris jääb süsteem arhitektuuriliselt sarnaseks prototüübiga, siis antud töö eesmärgiks on see lahti siduda Helmes AS infrastruktuurist ning võimaldada süsteemi kasutust teistes organisatsioonides ning asutustes. Selle saavutamiseks asendasin kõik moodulid, mida prototüüp kasutas näiteks raamatute ja kasutajate info talletamiseks, *PostgreSQL* andmebaasi tabelitega. Samuti realiseerisin vaated kasutajaliidesesse raamatute ja süsteemi kasutajate haldamiseks.

7 Tulemused

Eksperimentidest tulenevalt on selgelt antud töö jaoks kõige otstarbekamad valikud näotuvastuseks kasutada *Face++* teenust ning raamatute tuvastuseks *Google Cloud Vision API*-t. Mõlemad teenused lubavad tavatingimustes ning ka keerulisemate sisenditega mahtuda ajalises kui ka täpsuse mõõtmes lubatud ärilistesse piiridesse. Samuti selgitasin välja, et *Face++* näotuvastuse enesekindluse on võimalik tõsta üle 90% vaid ühe treeninguga, mille sisendiks kasutatakse isiku näopilti, mis on tehtud tuvastamisega sarnases keskkonnas. Teised võrdluses osalevad tehnoloogiad olid märkimisväärselt kas ajalimiidi või tuvastamise efektiivsuse suhtes kehvemad kui süsteemi jaoks välja valitud lahendused.

Koos uute väliste tuvastustehnoloogiatega ning kasutajaliideses ning serverirakenduses tehtud paranduste ja optimeerimistega on uue süsteemiga võimalik täita arenduse alguses püstitatud nõuded. Süsteem suudab normaaltingimustel, milleks on mõistlik tavaline toavalgustus vähemalt 200 lx-i ning kasutaja ning raamatud asukoht umbes 1 meetri kaugusel kaamera vaateväljas, järjepidevalt teostada näotuvastuse ning raamatu tuvastamise kiiremini kui 3 sekundit. Samuti said täidetud nõuded süsteemi administreerimisliidesele. Nii kasutajate kui ka raamatute haldust on võimalik läbi kasutajaliidese teostada. Lisaks on süsteem täielikult lahti seotud Helmes AS sisesüsteemidest ning ettevõttesisesed komponendid ja teenused on asendatud andmebaasitabelitega.

8 Kokkuvõte

Käesoleva töö eesmärgiks oli uurida mitmeid lahendusi olemasoleva raamatukogu laenutussüsteemi prototüüplahenduse edasi arendamiseks ning vastavad arendused teostada.

Töö käigus võrdlesin 8 eksperimendiga kolme näotuvastuse tehnoloogiat ning 7 eksperimendiga kolme optilise märgituvastuse tehnoloogiat. Lisaks viisin läbi kõige edukama näotuvastustehnoloogia valideerimise kasutajatestimisega. Kõige edukamalt esinevad tehnoloogiad integreerisin uude täielikult ümberkirjutatud koodibaasiga laenutussüsteemi. Koodibaasi ümberkirjutamise käigus võtsin nii kasutajaliideses, serverirakenduses kui ka andmebaasi tasemel kasutusele uuemad ning võimekamad tehnoloogiad, ning optimeerisin funktsionaalsust ning struktureerisin koodi ennast paremini kui seda sai tehtud antud töös arendatud süsteemi eelkäija arendamisel.

Minu bakalaureusetöö ühe edasiarendusena oleks võimalik uurida täiendavaid näotuvastuse ja OCR süsteeme, kuna tegemist on väga kiiresti areneva valdkonnaga. Samuti oleks potentsiaalne süsteemi täiustamine tehnoloogiate kombineerimine näiteks ühe tehnoloogiaga pildi lõikamine nii andmemahult kui ka tuvastamise mõttes optimaalsemale kujule ning ainult selle edastamine tuvastamiseks. Lisaks on võimalik edasiarendus reaalne implementatsioon mõne muu asutuse raamatukogus.

Kasutatud kirjandus

[1] M. A. Muqeet ja R. S. Holambe, „A collaborative representation face classification on separable adaptive directional wavelet transform based completed local binary pattern features,“ 04 08 2018. [Võrgumaterjal]. Saadaval: https://www.sciencedirect.com/science/article/pii/S2215098617311783. [Kasutatud 23 04 2019].

[2] Nicomsoft, „Optical Character Recognition (OCR) – How it works,“ 05 02 2012. [Võrgumaterjal]. Saadaval: https://www.nicomsoft.com/optical-character-recognition-ocr-how-it-works/. [Kasutatud 23 04 2019].

[3] Wikipedia, „Facial recognition system,“ 2019. [Võrgumaterjal]. Available: https://en.wikipedia.org/wiki/Facial_recognition_system. [Kasutatud 04 05 2019].

[4] C. Reports, „Facial Recognition: Who's Tracking You in Public?,“ 30 12 2015. [Võrgumaterjal]. Saadaval: https://www.consumerreports.org/privacy/facial-recognition-who-is-tracking-you-in-public1. [Kasutatud 02 05 2019].

[5] E. Barrett, „In China, Facial Recognition Tech Is Watching You,“ 28 10 2018. [Võrgumaterjal]. Saadaval: http://fortune.com/2018/10/28/in-china-facial-recognition-tech-is-watching-you/. [Kasutatud 02 05 2019].

[6] S. Wee ja E. Chen, „China’s Tech Firms Are Mapping Pig Faces,“ 24 02 2019. [Võrgumaterjal]. Saadaval: https://www.nytimes.com/2019/02/24/business/china-pig-technology-facial-recognition.html. [Kasutatud 15 05 2019].

[7] M. W. Pontin, „Better Face-Recognition Software,“ 30 05 2007. [Võrgumaterjal]. Saadaval: https://www.technologyreview.com/s/407976/better-face-recognition-software. [Kasutatud 04 05 2019].

[8] Wikipedia, „Multiple Biometric Grand Challenge,“ 2019. [Võrgumaterjal]. Saadaval: https://en.wikipedia.org/wiki/Multiple_Biometric_Grand_Challenge. [Kasutatud 04 05 2019].

[9] Wikipedia, „Optical character recognition,“ 2019. [Võrgumaterjal]. Saadaval: https://en.wikipedia.org/wiki/Optical_character_recognition. [Kasutatud 05 05 2019].

[10] Wikipedia, „Optacon,“ 2019. [Võrgumaterjal]. Saadaval: https://en.wikipedia.org/wiki/Optacon. [Kasutatud 05 05 2019].

[11] Wikipedia, „Timeline of optical character recognition,“ 2019. [Võrgumaterjal]. Saadaval: https://en.wikipedia.org/wiki/Timeline_of_optical_character_recognition. [Kasutatud 05 05 2019].

[12] M. Azure, „What is the Azure Face API?,“ 20 02 2019. [Võrgumaterjal]. Saadaval: https://docs.microsoft.com/en-us/azure/cognitive-services/face/overview. [Kasutatud 13 05 2019].

[13] M. Azure, „Cognitive Services pricing - Face API,“ 2018. [Võrgumaterjal]. Saadaval: https://azure.microsoft.com/en-us/pricing/details/cognitive-services/face-api. [Kasutatud 13 05 2019].

[14] OpenCV, „About,“ 2019. [Võrgumaterjal]. Saadaval: https://opencv.org/about. [Kasutatud 13 05 2019].

[15] Face++, „What is Face++ Cognitive Services?,“ 2019. [Võrgumaterjal]. Saadaval: https://www.faceplusplus.com/about-us. [Kasutatud 13 05 2019].

[16] Face++, „Pricing and Plans,“ 2019. [Võrgumaterjal]. Saadaval: https://www.faceplusplus.com/pricing. [Kasutatud 13 05 2019].

[17] UCSD, „Yale Face Database,“ [Võrgumaterjal]. Saadaval: http://vision.ucsd.edu/content/yale-face-database. [Kasutatud 13 05 2019].

[18] M. Azure, „Cognitive Services Pricing - Computer Vision API,“ 2019. [Võrgumaterjal]. Saadaval: https://azure.microsoft.com/en-us/pricing/details/cognitive-services/computer-vision. [Kasutatud 13 05 2019].

[19] G. Cloud, „Vision API,“ [Võrgumaterjal]. Saadaval: https://cloud.google.com/vision. [Kasutatud 13 05 2019].