ATHABASCA UNIVERSITY

AUTOMATED SPOKEN LANGUAGE DETECTION

BY

RIPLEY PENNELL

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE IN INFORMATION SYSTEMS

FACULTY OF SCIENCE AND TECHNOLOGY

ATHABASCA, ALBERTA

JANUARY, 2022

Approval of Thesis

The undersigned certify that they have read the thesis entitled

AUTOMATED SPOKEN LANGUAGE DETECTION

Submitted by

Ripley Pennell

In partial fulfillment of the requirements for the degree of

Master of Science in Information Systems

The thesis examination committee certifies that the thesis
and the oral examination is approved

Supervisor:
Dr. Maiga Chang
Athabasca University

Committee Members:
Dr. Ali Dewan
Athabasca University

Dr. Kuo-Chen Li
Chung-Yuan Christian University

External Examiner:
Dr. Seng Yue Wong
University of Malaya

January 4, 2022

1 University Drive, Athabasca, AB, T9S 3A3 Canada
Toll-free (CAN/U.S.) 1.800.788.9041 (ext 6821)
fgs@athabascau.ca | fgs.athabascau.ca | athabascau.ca

Abstract

This research allows two individuals to speak their language with an application detecting what languages are being spoken, allowing automatic translation. Existing relevant Systematic Literature Reviews (SLRs) articulated the need for this research. An SLR with quantitative and qualitative analysis identified the best algorithm to use, the i-vector algorithm. To integrate it onto a mobile platform it had to be completely recreated, referencing Kaldi. A voice database was created using Mozilla Common Voice and four (4) models were trained using TensorFlow, each showing unique improvements. The final model is deployed in an Android application using Chaqoupy for environment translation. Evaluation produced an accuracy of 81% and a 95.7 on the System Usability Scale. Evaluation data was transformed for normality and analyzed using a one-way analysis of variance and a two independent samples t-test. This research can be applied to all languages and has no dependency on accents.

Keywords: Natural Language Processing, i-vector, Language Identification (LID), Automatic Speech Recognition (ASR), Kaldi, Mozilla Common Voice, TensorFlow, Chaqoupy

Table of Contents

List of Tables

List of Figures and Illustrations

Chapter 1. Introduction

Motivation and Goal

One of the greatest barriers in communication is the vast multitude of differing languages that humans use. Good communication must be clear and quick. While it is possible to bridge the language barrier through body language or use of images, these methods of communication lack those two factors. The translation of written communication is a popular alternative form of communication which does bring clarity but lacks speed. Verbal communication is both clear and quick, but translation is much more difficult as there is no reliable method to identify what language is being spoken. Further research into spoken language detection can help overcome one of the greatest barriers in clear and quick communication.

The military can benefit greatly from improving its ability to communicate by automatically detecting a spoken language. One of the core tenants of winning in warfare is command and control, which is not possible without proper communications [1]. As militaries often wage war in countries with differing native tongues, communications can break down when dealing with local civilians and foreign friendly forces. A current solution is the use of local translators that know all the languages spoken in the region, but they pose a significant security risk, do not scale well, and require resources to protect. Research into how to automatically detect a spoken language for translation can prove to be an excellent alternate to local translators.

Communication barriers have been significantly reduced due to technical, automated innovations. The internet currently allows 55% of the world's population to communicate with each other [2]. Although most of these users speak different languages, applications exist that can translate phrases automatically. Google Translate is a powerful application that can not only translate text,

but also translate entire web pages, images, and even audio [3]. While the tool can automatically detect the language of written communication, it cannot automatically detect the language of verbal communication despite the translation quality. This requires users to know the language they are trying to understand and manually select it, reducing the efficiencies of verbal communication. This modern communication issue can be resolved with further research into automatic language detection.

Research Purpose

The purpose of this research is to solve an existing gap in the technological employment of voice translation, the automatic detection of the spoken language. This solution will allow two individuals to speak their own language, and with the presence of a device running an app, will be able to understand each other. It does not focus on the translation itself, but rather on the automated detection of the spoken language to automatically set the required parameters for a seamless conversation. Not only is the automatic detection a matter of convenience, it may also be a matter of necessity for the conversation to take place, as the speakers may be unable to convey to each other what language they are speaking. The purpose of the research is accomplished by answering a specific research question and proving the inherit hypotheses.

The research question is: “Can machine learning algorithms be used to increase the effectiveness of spoken language detection?” This question is answered by a combination of two hypotheses, one focusing on the technical ability to classify the detected language accurately and the other focusing on the degree of usability in its employment. The technical hypothesis is the more important hypothesis as the usability hypothesis greatly depends on the classifications being accurate. The research question helps identify the two hypotheses that need to be proved in this research.

The technical hypothesis (H_T) to be proven is: "A machine learning algorithm can classify a language being spoken in real-world scenarios." As classifications with complex factors are very unlikely to be correct every time, focus is being put on whether it can be correct at all. If it is possible, but very unreliable, then this hypothesis will be proven but the usability hypothesis will fail. "A language being spoken in real-world scenarios" refers to an individual speaking into a device with the background noise of an environment. This hypothesis is the first hypothesis to be proven, then focus will be put on optimizing the algorithms for usability.

The usability hypothesis (H_U) to prove is: "The perceived usability toward the application with the proposed machine learning algorithm built-in is high." Users will be presented with the language they are speaking. This would then be used to increase the effectiveness of verbal translation applications as it means that the user would not have to manually select the languages being spoken. This is important because in most cases the user of the verbal translation application would not be able to identify the spoken language themselves. This hypothesis is the second hypothesis to be proven, which will ultimately answer the research question.

Thesis Structure

Chapter 1 introduces the research by explaining the research motivation and goal, the research purpose, and the thesis structure. The chapter begins with an explanation of the research motivation and goal by giving a brief introduction on verbal communications, current technological employments and uses, and current issues. The research purpose follows, defining the research questions and underlying hypotheses that must be proven. The chapter concludes with this outline structure, outlining each chapter. The first chapter serves to set up the reader to better understand the follow-on research.

Chapter 2 explains the systematic literature review (SLR), which selects and critically analyzes current literature to answer specific, formulated questions [4]. The chapter begins with an outline of existing relevant and similar literature review research on speech recognition and machine learning to form a base upon which to conduct the SLR. This is followed by the process taken to conduct the SLR, expressing keyword sets, search targets, the process of searching, summary of results, and inclusion and exclusion criteria with filtering graphics. The quantitative analysis and results with graphics are next which portray what approaches are most likely to succeed. The chapter concludes with a qualitative analysis to help explain the quantitative analysis, summarizing all that was extracted by the conduct of the SLR. The second chapter pulls together the collection of literature that was reviewed and enabled the follow-on research.

Chapter 3 is on the spoken language detection, outlining the design, algorithm, and test cases. The chapter begins with a description of the method used to automatically detect the spoken language. This leads into the specific machine learning algorithm that is used to classify what the spoken language is. The chapter concludes with the models that were trained and improved upon using different test cases, including ones attempting to mimic “real-world scenarios” as described in the technical hypothesis. The third chapter outlines how the technical hypothesis is proven.

Chapter 4 describes the prototype of the research and its architecture and workflow, the app and use cases, and evaluation plan. The chapter begins with the design of how the research is employed, documenting the intended architecture and workflow. The description of how this translated into the design of the app follows, noting use cases of its employment. The chapter concludes with the plan on how to evaluate the app to answer the research question. The fourth chapter specifies how both hypotheses are proven through the use of a prototype app.

Chapter 5 conveys the results of the research, assessing the accuracy, usability, and overall findings of the evaluations. The chapter begins with a report on the accuracy of the classification and usability of the application, proving the technical and usability hypotheses. Proceeding this is normality tests on the recorded data as well as their transformations. These transformations are required for the follow-on data analysis which observes how different independent variables impacted the usability. The chapter concludes with a documentation on the findings and discussions of the evaluation, noting key observations. The fifth chapter proves the two hypotheses and answers the research question.

Chapter 6 is the conclusion of the research, summarizing findings and contributions, stating limitations, and outlining future works. The chapter begins with a summary that outlines what was conducted, what contributions the research has, and what the main findings of the research are. It then states the limitations of the research, outlining the areas that the research can be improved in. The chapter concludes with theories of future works based on the findings, limitations, and recommendations of the research. The sixth and final chapter compiles and summarizes all the research that was done in this thesis.

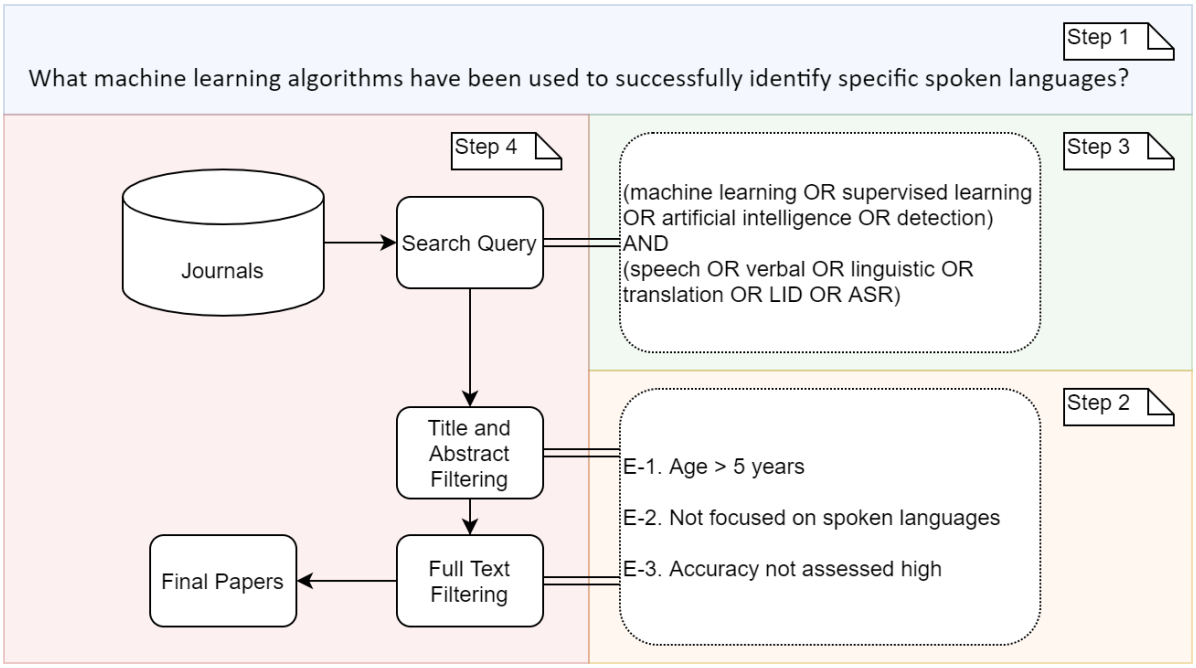
Chapter 2. Systematic Literature Review (SLR)

SLR Process

The SLR process for this research consists of four steps, outlined in Figure 1. An SLR is different from other forms of reviews as it involves a detailed plan and search strategy to eliminate biases. The first step functions as the base and goal of the SLR by formulating the review question that will be answered. The second step determines what will be analyzed in the SLR by defining the exclusion criteria with several factors. The third step outlines how publications will be collected for the SLR by developing the search strategy and locating the studies that will be used. The fourth and final step is the actual conduct of the SLR utilizing the previous steps by selecting the studies [5]. The SLR process is a rigorous review of current research that serves as a base for this research.

Figure 1

Study Selection Process Followed for the SLR



The first step in conducting the SLR is to formulate the review question [5], which is "What machine learning algorithms have been used to successfully identify specific spoken languages?" This review question is more focused on the technical hypothesis as the technical portion of the research can leverage existing research much more than the usability portion. By understanding what machine learning algorithms have been used to classify languages, the more accurate ones can be chosen for this research. By also understanding what languages these algorithms were used with, the ones used with more varied languages can be chosen for this research. This is important as this research focuses on the classification of multiple languages, so if an algorithm is accurate but only works on two languages, it will not be acceptable. Conversely, if an algorithm works with many languages but is largely inaccurate, it will also not be acceptable. This review question will determine the best algorithm to use in this research.

The second step in conducting the SLR is to define the exclusion criteria [5], which is based on the research's publication year (i.e., its age, older than five years), non-focus on spoken languages, and classification accuracy (less than 80%). The age of the literature being somewhat recent is important to leverage the most current technology available. The program regulations of Athabasca University's Master or Science in Information Systems, the program this research is being conducted under, has a time limit of five years to complete the degree due to the ever changing volatility of the subject matter. Fortunately, five years ago is 2015 which was a landmark year for machine learning [6] [7] [8]. For these reasons, a five-year limit will be enforced (E-1).

The machine learning algorithms being used on languages specifically is important to understand how they will perform with the dataset that this research will be processing. If a machine learning algorithm performed excellently at determining something red from something blue, it may not be ideal for this research. In this example, this is because the machine learning algorithm was only used to make a classification based on one factor, whereas the dataset in this research will have many factors. While this exclusion might seem redundant, it will be required as keywords are not a reliable way of filtering. By excluding literature that is focused on classifying things other than spoken languages, more relevant machine learning algorithms will be reviewed (E-2).

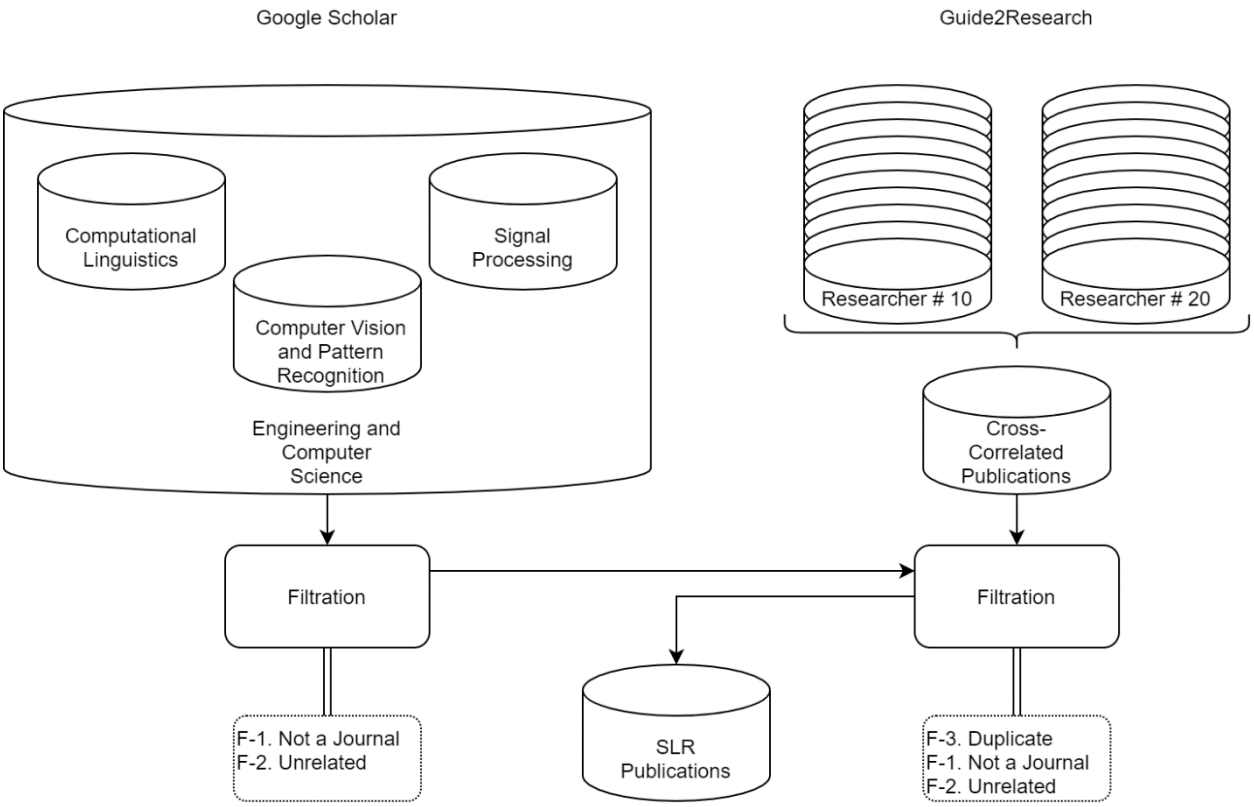
The accuracy of the classification being high is important to prove the technical hypothesis. It is however even more important that it be high for the usability hypothesis, as it will greatly negatively impact the user's perception of the research if the wrong classification is ever made. Given most users would only deal with a handful of languages and those languages would likely be contained in a small, common subset of all languages, the accuracy does not need to be

incredibly high to reduce the chances the wrong classification is given once. As algorithms will perform with very different accuracies based on the dataset being classified, a high, hard-coded value would miss highly accurate algorithms that were used on difficult datasets. Authors that wrote up the research are then relied upon to decide whether the algorithm accuracy is considered high depending on the dataset it was used for. While the authors would be biased towards reporting the algorithms as performing with a high accuracy, the accuracy will be examined during the SLR analysis to counter this bias. For the application of this research, machine learning algorithms that the author does not consider to be high will be excluded (E-3). These exclusion criteria will ensure a review of recent, relevant, and accurate machine learning algorithms.

The third step in conducting the SLR is to develop the search strategy and locate studies [5], which starts with methods to intelligently identify the literature sources as is shown in Figure 2. There are two methods used to select the literature sources: the first being well-cited, high-quality publications relating to the subject field; and the second being high-quality publications that well-cited researchers in the subject field published their work in. The use of two methods to locate studies reduces the chance of an inherit bias eliminating the consideration of relevant, useful publications. Figure 2 illustrates how the publications were collected and filtrated. By using only the most well-cited and high-quality publications in the subject field, the SLR is much more efficient.

Figure 2

Collection and Filtration of SLR Sources



To find well-cited, high-quality publications, Google Scholar's top publications along with filtration criteria was used. The top publications listed in Google Scholar was used to find well-cited publications, which lists publications in order of their h-index. This metric, the h-index, was created by Google Scholar and represents "the largest number h such that at least h articles in that publication were cited at least h times each" [9]. The publications within Google Scholar are also sorted by category, with the category most relevant to the subject field being "Engineering and Computer Science." The subcategories most relevant to the subject field are "Computational Linguistics," "Computer Vision and Pattern Recognition," and "Signal Processing." As Google Scholar lists 20 publications in each sub-category, this creates a list of 60 well-cited publications with 56 unique listings.

To determine which of these well-cited publications are high-quality, the filtration criteria need to be defined. Many of the publications are workshops and conferences, which have less detailed research due to length limitations and face less review and value scrutiny due to the fiscal incentive to have as large an audience as possible. The first filter (F-1) of the publication being a journal eliminates 27 publications, with 29 publications remaining. Since none of the sub-categories directly fits the subject field, there remains some journals that are not relevant to the research. The second filter (F-2) of the publication being relevant to the research eliminates 12 publications, with 17 publications remaining.

Table 1 contains the list of well-cited, high-quality publications relating to the subject field and the filtration used. Publications in red were filtered out with F-1 and publications in orange were filtered out with F-2, leaving the well-cited, high-quality publications in green. Through the use of Google Scholar's amalgamation of well-cited publications and some filter criteria, 17 publications were located to act as the basis for the SLR.

Table 1

Filtration of Well-Cited, High-Quality Publications Related to Subject Field

Publication	h5-index	Excluded by
IEEE/CVF Conference on Computer Vision and Pattern Recognition	299	F-1
IEEE/CVF International Conference on Computer Vision	176	F-1
European Conference on Computer Vision	144	F-1
Meeting of the Association for Computational Linguistics (ACL)	135	F-1
IEEE Transactions on Pattern Analysis and Machine Intelligence	131	
IEEE Transactions on Image Processing	113	F-2
Conference on Empirical Methods in Natural Language Processing (EMNLP)	112	F-1
IEEE Transactions on Wireless Communications	110	F-2
IEEE Transactions on Signal Processing	97	
Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (HLT-NAACL)	90	F-1
IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)	86	F-1
Pattern Recognition	85	

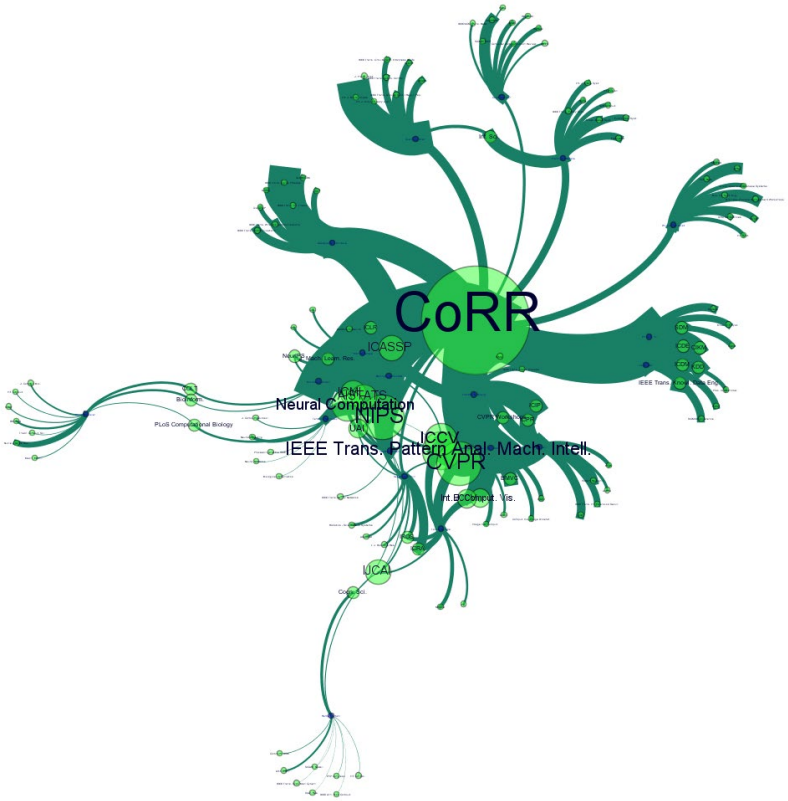
Publication	h5-index	Excluded by
Conference of the International Speech Communication Association (INTERSPEECH)	81	F-1
IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops	73	F-1
IEEE Transactions on Circuits and Systems for Video Technology	71	
International Journal of Computer Vision	70	F-2
Medical Image Analysis	67	F-2
IEEE Signal Processing Letters	66	
Signal Processing	62	
IEEE Signal Processing Magazine	60	
Pattern Recognition Letters	59	
British Machine Vision Conference (BMVC)	57	F-1
IEEE/ACM Transactions on Audio, Speech, and Language Processing	57	
IEEE Journal of Selected Topics in Signal Processing	57	
IEEE Wireless Communications Letters	55	F-2
Workshop on Applications of Computer Vision (WACV)	54	F-1
Transactions of the Association for Computational Linguistics	53	
IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)	51	F-1
International Workshop on Semantic Evaluation	50	F-1
Computer Vision and Image Understanding	50	F-2
International Conference on Computational Linguistics (COLING)	49	F-1
IEEE Vehicular Technology Conference, VTC	46	F-1
Conference of the European Chapter of the Association for Computational Linguistics (EACL)	45	F-1
Journal of Visual Communication and Image Representation	45	F-2
IEEE International Conference on Automatic Face & Gesture Recognition	41	F-1
SIAM Journal on Imaging Sciences	40	F-2
Digital Signal Processing	40	
Conference on Computational Natural Language Learning (CoNLL)	39	F-1
Applied and Computational Harmonic Analysis	39	
Signal Processing: Image Communication	39	F-2
International Conference on Language Resources and Evaluation (LREC)	38	F-1
International Conference on 3D Vision	37	F-1
IEEE Transactions on Computational Imaging	37	F-2
Image and Vision Computing	36	F-2
Workshop on Machine Translation	35	F-1
International Conference on Pattern Recognition	35	F-1
Computer Speech & Language	33	
Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)	29	F-1
Computational Linguistics	26	
IEEE Spoken Language Technology Workshop (SLT)	25	F-1
International Joint Conference on Natural Language Processing (IJCNLP)	24	F-1
Natural Language Engineering	23	
IEEE International Conference on Semantic Computing	23	F-1
Language Resources and Evaluation	22	
Biomedical Natural Language Processing	22	F-2
Workshop on Representation Learning for NLP	20	F-1
F-1. Not a Journal	F-2. Unrelated	SLR Publication

To find high-quality publications that well-cited researchers in the subject field published their work in, Guide2Research's listing of top researchers was used to cross-correlate publications along with filtration criteria. Fortunately, Guide2Research, which lists the top researchers based on Google Scholar's h-index in a certain field, has a category specifically for computer science in speech recognition. The top 20 researchers across all countries are used to cross-correlate publications that leaders in the subject field would look to publish their work in. The researchers are not limited to being in North America nor English-speaking countries only as this would prevent considering valuable research across the world. However, publications that are not available in English are not considered. As some of these researchers have published works in over 150 venues, only their top 10 venues will be considered.

Figure 3 illustrates the link analysis between the top 20 researchers and their publication venues. Similar to how Google Scholar focused on the top 20 publications per category, only the top 20 cross-correlated publications are considered. The first and second filters (F-1 and F-2) are also applied to this dataset along with a third filter (F-3) which eliminates publications already being considered by the first method of selecting publication sources.

Figure 3

Link Analysis of Well-Cited Researchers and Their Publication Venues



To better see the correlation between the correlated publications and the publications in Google Scholar, the third filter (F-3) will be conducted first. This helps understand if there is any discrepancy between top researchers and top publications. Given the amount of publications that were filtered this way, the much faster and seemingly less accurate Google Scholar method is proven to be a good method to gather publications that well-cited researchers in the subject field published their work in.

The third filter (F-3) of the publication already being considered through the Google Scholar method eliminates eight publications, with 12 publications remaining. As most research that is published in a journal is also published in a conference, and most research that is published in a

conference is not published in a journal, it makes sense that the most common publication avenues for leading researchers in the subject field would be conferences. The first filter (F-1) of the publication being a journal eliminates 9 publications, with 3 publications remaining. Since the subject field is more of a secondary or tertiary form of research specialization, it also makes sense that these well-cited researchers in the subject field would also commonly publish research in unrelated publications. The second filter (F-2) of the publication being relevant to the research eliminates one publications, with two publications remaining.

Table 2 contains the list of high-quality publications that well-cited researchers in the subject field published their work in. Publications in blue were filtered out with F-3, publications in red were filtered out with F-1, and publications in orange were filtered out with F-2, leaving the high-quality publications that well-cited researchers in the subject field published their work through in green. Through the use of Guide2Research's amalgamation of well-cited researchers in the subject field with some cross-correlation and filter criteria, 2 publications were added to act as the basis for the SLR, totalling 19 publications.

Table 2

Filtration of High-Quality Publications of Well-Cited Researchers in Subject Field

Publication	Weighted Degree	Excluded by
Computing Research Repository (CoRR)	1765	F-1
IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)	380	F-3
Neural Information Processing Systems (NIPS)	375	F-1
IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)	235	F-3
IEEE Transactions on Pattern Analysis and Machine Intelligence	221	F-3
IEEE International Conference on Computer Vision (ICCV)	162	F-3
International Journal of Bifurcation and Chaos	159	
IEEE Transactions on Signal Processing	155	F-3
Knowledge Discovery and Data Mining (KDD)	151	F-1
International Conference on Machine Learning (ICML)	145	F-1

Publication	Weighted Degree	Excluded by
International Conference on Pattern Recognition (ICPR)	136	F-3
IEEE Transactions on Knowledge and Data Engineering	136	
IEEE International Conference on Data Mining (ICDM)	126	F-1
European Conference on Computer Vision (ECCV)	117	F-3
IEEE International Conference on Data Engineering (ICDE)	115	F-1
IEEE International Conference on Image Processing (ICIP)	108	F-1
SIAM International Conference on Data Mining (SDM)	104	F-1
British Machine Vision Conference (BMVC)	103	F-3
International Conference on Information and Knowledge Management (CIKM)	102	F-1
Neural Computation	95	F-2

F-3. Duplicate	F-1. Not a Journal	F-2. Unrelated	SLR Publication

With the literature sources identified, all that remains to complete the search strategy is to identify the keywords, which consists of both general and specific keywords combining two different areas of research. The two concepts that are being jointly searched for are classifications and language. The two concepts are joined by an “and” statement as they are only useful if combined, and keywords among both concepts are joined by an “or” statement as there are many ways to describe those concepts.

The general keywords are words that describe the subject field in an obvious way, terms that anyone unfamiliar with the subject field would logically think of. For general keywords on classifications, “machine learning,” “supervised learning,” “artificial intelligence,” and “detection” were identified. For general keywords on language, “speech,” “verbal,” “linguistic,” and “translation” were identified.

The specific keywords are words that better describe the subject field but would only be known to those with experience within the field. For specific keywords on classifications, no specific keywords were identified. For specific keywords on language, “LID,” short for Language Identification, and “ASR”, short for automatic speech recognition, were identified. The

identification of the general and specific keywords completes the search strategy, the third step of the SLR.

The fourth step (see Figure 1) in conducting the SLR is to select studies [5], which used the previously identified sources, keywords, and exclusion criteria. Access was confirmed for the 19 publications that were identified for the SLR. Queries were made in each repository using their custom advanced search options, achieving the “or” and “and” links criteria of the identified keywords. For databases that had a limit on the number of Boolean connectors, “or” keywords were dropped and a separate search was done to cross-correlate differences and logically assemble the intended data set. This returned X papers which then needed to be filtered down using the identified exclusion criteria. After completing the filtering based on the titles and abstract, the number of papers was reduced by 9,599, leaving 63 papers to be fully read for more filtering. Further filtering reduced the number of papers by 54, leaving nine papers for analysis. Table 3 contains the list of nine (9) papers that were selected for the SLR.

Table 3

Selected Studies of the SLR

Paper Title	Author	Journal	Reference
Advanced Data Exploitation in Speech Analysis: An overview	Zhang et al.	IEEE Signal Processing Magazine	[10]
Speech Processing for Digital Home Assistants: Combining Signal Processing With Deep-Learning Techniques	Haeb-Umbach et al.	IEEE Signal Processing Magazine	[11]
Efficient estimation and model generalization for the totalvariability model	Travadi and Narayanan	Computer Speech & Language	[12]
Supervised i-vector modeling for language and accent recognition	Ramoji and Ganapathy	Computer Speech & Language	[13]
Residual convolutional neural network with attentive feature pooling for end-to-end language identification from short-duration speech	Monteiro et al.	Computer Speech & Language	[14]
On the use of deep feedforward neural networks for automatic language identification	Lopez-Moreno et al.	Computer Speech & Language	[15]
Parametric representation of excitation source information for language identification	Nandi et al.	Computer Speech & Language	[16]

Paper Title	Author	Journal	Reference
Implicit processing of LP residual for language identification	Nandi et al.	Computer Speech & Language	[17]
Regularization of neural network model with distance metric learning for i-vector based spoken language identification	Lu et al.	Computer Speech & Language	[18]

The languages and datasets for the papers selected for the SLR are displayed in Table 4. The first two (2) papers focused on algorithms used to identify languages but presented no experimental results, meaning no specific languages or databases were given. While the other seven (7) papers did provide the datasets the algorithms were trained with, none of them are publicly available. Most of the papers used datasets created by the United States Government which require approval processes to access. Others used databases that can be purchased for a large cost, with the data being delivered on a purchased hard drive. In total, 86 unique languages were considered by the SLR papers. While the SLR papers did not provide accessible datasets, they did show that their algorithms work on a very wide array of languages.

Table 4

Languages and Datasets Captured in the SLR

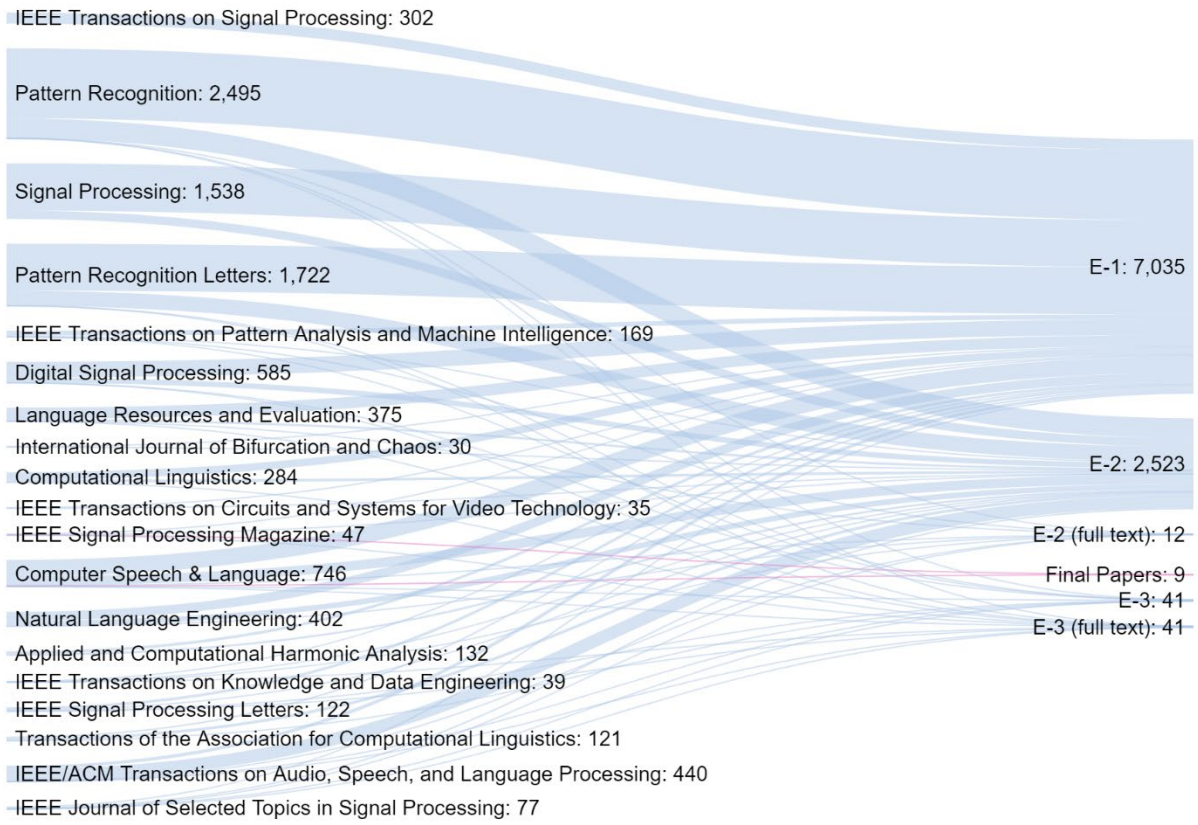
Reference	Languages	Dataset
[10]	No specific language given	No specific dataset given
[11]	No specific language given	No specific dataset given
[12]	Arabic, Dari, Farsi, Pashto, and Urdu	Defense Advanced Research Projects Agency (DARPA) Robust Automatic Transcription of Speech (RATS) database [19]
[13]	Arabic – Egyptian Arabic, Iraqi Arabic, Levantine Arabic, and Maghrebi Arabic Chinese – Mandarin and Min Nan English – British English and General American English Slavic – Polish and Russian Iberian – Caribbean Spanish, European Spanish, Latin American Spanish, and Brazilian Portuguese	National Institute of Standards and Technology (NIST) Language Recognition Evaluation (LRE) 2017 [20]
[14]	Cantonese, Mandarin, Indonesian, Japanese, Russian, Korean, Vietnamese, Kazakh, Tibetan, and Uyghur	Oriental Language Recognition (OLR) 2018 Challenge [21]

Reference	Languages	Dataset
[15]	Amharic, Bosnian, Cantonese, Creole (Haitian), Croatian, Dari, English (American), English (Indian), Farsi, French, Georgian, Hausa, Hindi, Korean, Mandarin, Pashto, Portuguese, Russian, Spanish, Turkish, Ukrainian, Urdu, and Vietnamese	NIST LRE 2009 [20]
[16]	Arunachali, Assamese, Bengali, Bho-jpuri, Chhattisgarhi, Dogri, Gojri, Gujrati, Hindi, Indian English, Kannada, Kashmiri, Konkani, Manipuri, Mizo,Malayalam, Marathi, Nagamese, Nepali, Oriya, Punjabi,Rajasthani, Sanskrit, Sindhi, Tamil, Telugu, Urdu, English, Farsi, French, German, Japanese, Korean, Mandarin Chinese, Spanish, Vietnamese, Arabic Iraqi, Russian, Arabic Levantine, Slovak, Arabic Maghrebi, Hindi, Spanish, Arabic MSA, Lao, Bengali, Thai, Czech, Panjabi, Turkish, Dari, Pashto, Ukrainian, and Polish	Indian Institute of Technology Kharagpur-Multi Lingual Indian Language Speech Corpus (IITKGP-MLILSC) [22], Oregon Graduate Institute Multi-Language Telephone-based Speech (OGI-MLTS) [23], and NIST LRE 2011 [20]
[17]	Arunachali, Assamese, Bengali, Bho-jpuri, Chhattisgarhi, Dogri, Gojri, Gujrati, Hindi, Indian English, Kannada, Kashmiri, Konkani, Manipuri, Mizo,Malayalam, Marathi, Nagamese, Nepali, Oriya, Punjabi,Rajasthani, Sanskrit, Sindhi, Tamil, Telugu, Urdu, English, Farsi, French, German, Japanese, Korean, Mandarin Chinese, Spanish, and Vietnamese	IITKGP-MLILSC [22] and OGI-MLTS [23]
[18]	Arabic – Egyptian Arabic, Iraqi Arabic, Levantine Arabic, Maghrebi Arabic, and Modern Standard Chinese – Cantonese, Mandarin, Min Nan, and Wu English – British English, General American English, and Indian English French – West African French and Haitian Creole Slavic – Polish and Russian Iberian – Caribbean Spanish, European Spanish, Latin American Spanish, and Brazilian Portuguese	NIST LRE 2015 [20]

A breakdown on the papers collected from each publication as well as the filtration reduction is in Figure 4. This visualization illustrates how the final papers came from only a select few journals. It also shows how some journals did not have any publications that even made it through the first exclusion criteria of age.

Figure 4

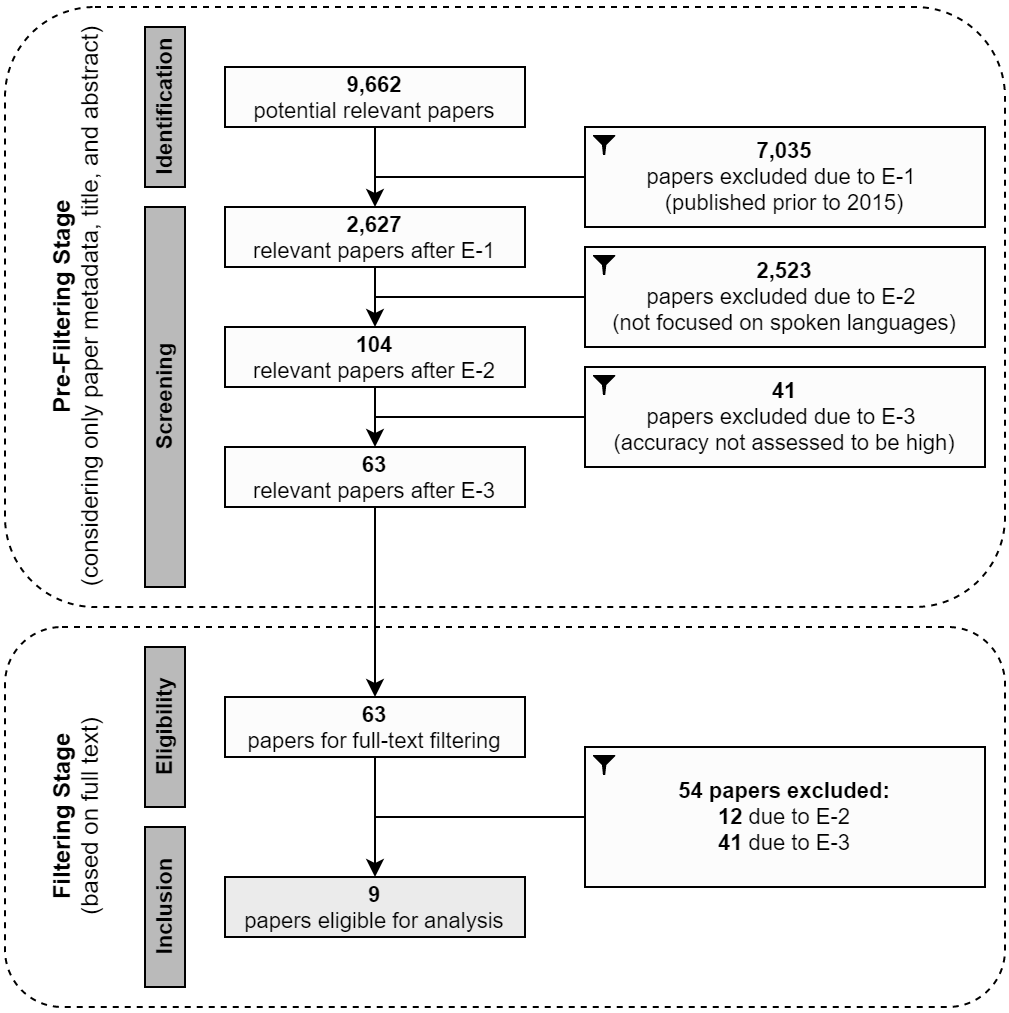
Breakdown of Papers From Publications and Filtration



The exclusion process and results of the SLR are in Figure 5. This displays how each exclusion criteria impacted the large initial number of potential relevant papers in both filtering stages. The SLR returned nine papers for analysis to answer the review question, “What machine learning algorithms have been used to successfully identify specific spoken languages?”

Figure 5

Process and Results of the SLR Execution



The set of initial papers before filtration was so large because the keywords caught nine (9) different concepts, four (4) of which were beneficial to fully read but were ultimately filtered out for being out of the research scope.

- The most beneficial group of papers that were read fully but were filtered out of the SLR were on separating speech from noise, or Voice Activity Detection (VAD) [24] [25]. As the technical hypothesis requires classification be done in "real-world scenarios," understanding current methodologies to separate speech from noise is beneficial. Many of

these papers also listed resources of large audio files of varying languages in real-world scenarios [26] [27] .

- The second beneficial group of papers was on identifying speakers based on audio [28]. Understanding how to differentiate who is talking is useful for preventing an individual from having their language classified twice or someone speaking in the background have their voice used for the classification.
- The third beneficial group of papers was on detecting emotions or diseases through audio [29] [30]. These papers focused on classifying audio into groups of different diseases similar to classifying audio into groups of different languages, which would have been good to consider if no papers on language identification existed.
- The fourth beneficial group of papers was on translation of spoken language techniques, or Spoken Language Understanding (SLU) using machine learning [31] [32]. These papers highlighted the value of this research as they always required the input language to be identified.

Most search criteria will have the phenomenon of returning unavoidable less relevant papers due to a variety of foreseen and unforeseen reasons, such as these five concepts in this SLR.

- The first less relevant group of papers was on the translation of data in general. This group was the largest group of papers in the initial set, as the word “translation,” in the context of manipulating data, is very often used when papers discuss machine learning.
- The second less relevant group of papers was on improving speech performance [33]. While these papers detail using machine learning on audio samples, the goal is very different and aims to manipulate the data rather than classify it.

- The third less relevant group of papers was on keyword detection. Although many of the papers in this subset did use audio samples, and although it might often work, proper language identification should not rely on an individual saying a keyword.
- The fourth less relevant group of papers was on correcting speech for an individual learning how to speak a certain language. While this does focus on classifying audio into good or bad categories, the classification is centered around a starting vector of a correct way to pronounce a word, whereas classifying audio into different languages has no starting vector.
- The fifth less relevant group of papers were papers that stated how machine learning is used to implement things like ASR in the introduction section. The paper would then go on to talk about an entirely different application of machine learning, but because it mentioned ASR just once as a potential use case as background information, it was among the search results.

The mix of unintended relevant and less relevant groups of papers discovered by the SLR helped acquire some solutions to relevant potential future problems, as well as understand the impact of the keyword selection.

Quantitative Analysis

From executing the SLR, the review question can be answered quantitatively by identifying the most common algorithm used to successfully identify a spoken language, which is i-vector. The SLR returned 12 different algorithms with a large mix of accuracies depending on its application. Figure 6 displays the tally of each algorithm.

Figure 6

Count of Identified Algorithms Through the SLR

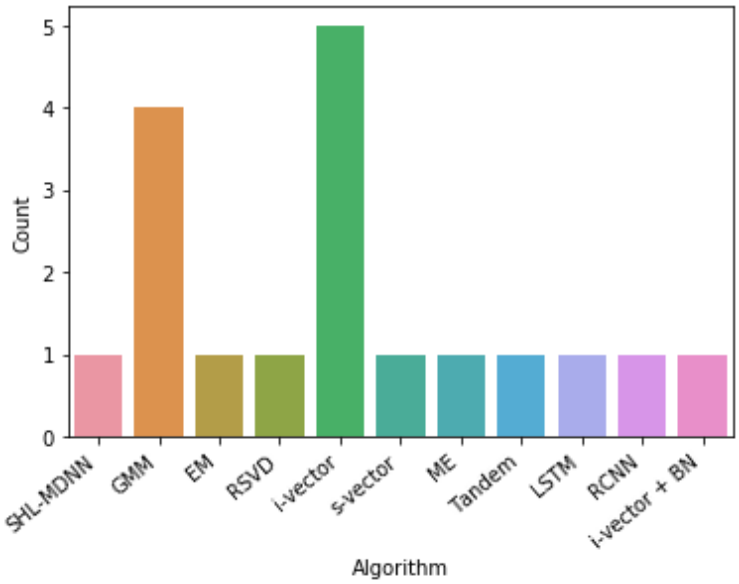


Table 5 contains a breakdown of the quantitative analysis, detailing which papers of the SLR contained which algorithm. Some algorithms appeared in multiple papers which allows for comparison. The comparisons of algorithms using the same data is extremely helpful as their accuracies can vary widely based on the data and application.

Table 5

Algorithms and Their Accuracies

Paper	Algorithm									
	SHL-MDNN	GMM	RSVD	i-vector	s-vector	ME	Tandem	LSTM	RCNN	i-vector + BN
Zhang et al. [10]	Unk*									
Haeb-Umbach et al. [11]		Unk*								
Travadi and Narayanan [12]		82.2	83.6							
Ramoji and Ganapathy [13]				85.7	85.4					
Monteiro et al. [14]				94.58			95.2	95.26	97.24	
Lopez-Moreno et al. [15]				99.79						99.82
Nandi et al. [16]		66**		68**						
Nandi et al. [17]		63.7**				65**				
Lu et al. [18]				94.9						
* Application of algorithm was deemed sufficient; accuracy was not given but is high										
** Accuracy is on classifying different Indian languages that are incredibly similar, would perform much better on different languages										

- Shared-Hidden-Layer Multilingual DNN (SHL-MDNN) was seen once and is an algorithm that has languages share layers before using independent layers, allowing jointly optimized training sets. An accuracy was not stated but deemed to be of high-performance [10].

- Gaussian Mixture Models (GMM) was seen four times and is an algorithm that captures language specific excitation source information, maximized using the Expectation Maximization (EM) algorithm. In the first case, no accuracy was given but it was deemed to be successful in its implementation which was in running recognition systems for each language in parallel and selecting the one returning the highest score [11]. In the next three cases, GMM was used as a benchmark as an established LID algorithm, which all performed slightly worse than the comparing algorithms [12] [16] [17].

- Randomized Singular Value Decomposition (RSVD) was seen once and is an algorithm that reduces the computational complexity of parameter estimation through the use of randomized algorithms. Through a test-case, it performed slightly better than GMM [12].

- i-vector was seen five times and is an algorithm which uses an unsupervised learning paradigm to convert variable length speech utterances into a fixed dimensional feature vector. In the first three cases, it was used as a benchmark as an established LID algorithm, performing slightly worse than the comparing algorithms [13] [14] [15]. In the fourth case, i-vector performed slightly better than the established GMM with EM algorithm, proving it is the superior benchmark of the two benchmark algorithms [16]. In the fifth case, the i-vector algorithm was implemented with a pair-wise distance metric learning regularization to improve its performance [18].

- S-vector was seen once and is an algorithm that is simply a supervised version of the unsupervised i-vector algorithm. Through a test-case, it performed slightly better than i-vector in some scenarios but had a lower highest accuracy [13].

- Maximum Entropy (ME) was seen once and is an algorithm that uses evidence based discrete modeling to automatically learn the conditional probabilities. Through a difficult test-case, it performed slightly better than GMM [17].

- Tandem was seen once and is an algorithm that transforms data using dimension reduction, then uses cluster analysis. Through a test-case, it performed slightly better than i-vectors [14].

- Long Short-Term Memory (LSTM) was seen once and is a type of Recurrent Neural Network (RNN) that also identifies long-term dependencies. Through a test-case, it performed slightly better than tandem and i-vectors [14].
- Residual Convolutional Neural Networks (RCNN) was seen once and focuses on understanding the contextual segments of input data. Through a test-case, it performed better than LSTM, tandem, and i-vectors [14].
- i-vector + bottleneck (BN) was seen once and is and algorithm than combines i-vector with BN, a Deep Neural Network (DNN) where inputs are bottleneck features. Through a test-case, the combination of the two algorithms outperformed either individually.

The algorithms discovered through the SLR helped understand that i-vectors is the most popular and best algorithm to detect a spoken language. The i-vector algorithm was used the most and is clearly recognized as a benchmark for more experimental and proof of concept algorithms. It also sometimes even performed better than the algorithm the paper was publishing. It was also shown it could be combined or melded with other concepts to improve its performance, showing that it is customizable and adaptable. The quantitative analysis done in the SLR shows that i-vectors is the best algorithm to select for this research.

Existing Relevant SLRs

An existing relevant SLR on sign language recognition has a similar goal of collecting all current research on language recognition, but focusing on the movement of hands rather than the sound of speech [34]. This SLR also has similar challenges as sign language varies based on regions just like spoken languages. There are seven research questions that all require quantitative analysis, either being answered with a pie chart or a number. The structure of the SLR is broken

into the 12 languages it considers, giving a description of collected papers recognizing that specific language before presenting pie charts and numbers to answer each research question. This would have made the SLR very easy to divide between researchers and is very easy to consume. Only one keyword is used to search for papers and the papers are collected from the four large sources of IEEE, ACM, Elsevier, and Springer. The only exclusion criteria given is the age of the paper, though the papers are filtered further by unstated criteria. No recommendations or application of results are given at the end of the SLR. There is also no mentioning of how to recognize sign languages apart from each other is given, demonstrating that there is an expectation in language detection that the language is already known. This inherent expectation that languages are manually selected for recognitions is a phenomenon that expands even to sign language, demonstrating the problem of automatic language detection.

A second existing relevant SLR on Automatic Speech Recognition (ASR) shares the same goal of collecting all current research on spoken language recognition, but also focuses on concepts other than recognizing the actual spoken languages [35]. Interestingly, no research questions were given in this SLR. The structure of the SLR first describes the models within the architecture of speech recognition systems before briefly discussing some papers on ASR in different languages, finishing with papers that consider interesting advancements in ASR. These advancements are using deep learning for ASR, recognizing emotions using ASR, using robust methods to generate ASR systems, and developing high performance ASR. A very brief qualitative analysis is given to identify the contributions of some of the papers discussed. Neither the keywords to select the papers is given, nor is the sources of the paper, nor exclusion criteria, nor a description of the SLR process. It also does not have any recommendations of how this information could be applied. The section on ASR in different languages avoids the language

recognition problem by using separately designed algorithms. This again shows the inherent expectation that the language is given before being used.

As these were the only SLRs related to detecting languages, there is a justification of the need of this research and the SLR. Both existing SLRs incorporates various languages, but neither has a solution on how to detect which language was being communicated. The papers also demonstrated the need to properly describe the SLR process as they leave the readers with many questions on what was actually conducted and why. The research question must be given and it must be useful so that the results can be applied. Exclusion criteria must be stated and well defined. The search strategy, detailing the databases used and why, as well as the keywords must be given. Finally, a description of how the search was done with graphics on visualizing the filtration must be given. With the papers, both quantitative and qualitative analysis should be done to give a full understanding of what the papers of the SLR have to offer. The SLR should also have a clear application for why it is being conducted. These two SLRs justified the need for this SLR and research and also demonstrated what SLRs need to be effective.

Qualitative Analysis

The qualitative analysis is done using a weighted decision matrix [36]. This matrix allows the different qualitative factors of the algorithms to be compared to each other. By first outlining the qualities that are most important and applying a weight to them based on the requirements of the best algorithm, biases can be removed. Using a constructed scale to evaluate each quality allows each algorithm to be scored against each other. By multiplying that scale based on the weight given to each quality, algorithms that score higher with qualities that are more important will have a higher total score. The total scores can then be compared, with the highest score belonging to the algorithm that has the highest score in the highest valued qualities.

From executing the SLR, the review question can be further informed qualitatively by considering the following qualities of the algorithms: accuracy, establishment in the field, ability to handle low resource languages. These qualities are broken into the three categories of low, medium, and high. Accuracy is an important component to the algorithm for the same reason it was set as a filtration criteria. Being an established algorithm in the field, vice being a proof-of-concept, means that there are more resources and tools available to make efficient use of it. It is also important that algorithms used for detecting languages can overcome the challenge of handling low-resource languages, languages where there is difficulty in finding suitable data to train the models, such as Urdu [37]. These qualities can help to better contextualize the quantitative results.

Table 4 contains a breakdown of the qualitative analysis, detailing the important factors of the best algorithms from the SLR to select. A weighted scoring was used, giving more points for more important factors.

Table 6

Qualitative Results of the SLR

Algorithm	Accuracy (1x Weight)	Establishment in Field (3x Weight)	Handle Low Resource Languages (2x Weight)	Total
SHL-MDNN	2 (2)	2 (6)	3 (6)	14
GMM	2 (2)	3 (9)	3 (6)	17
RSVD	3 (3)	1 (3)	1 (2)	8
i-vector	2 (2)	3 (9)	3 (6)	17
s-vector	1 (1)	1 (3)	2 (4)	8
ME	1 (1)	1 (3)	3 (6)	10
Tandem	2 (2)	2 (6)	2 (4)	12
LTSM	3 (3)	2 (6)	2 (4)	13
RCNN	3 (3)	1 (3)	2 (4)	10
i-vector + BN	3 (3)	2 (6)	2 (4)	13
Low		Medium	High	

- For accuracy, since a filtration was already set for the algorithms to be high, the relative difference between the scores are small, and a low score still represents an absolute high accuracy. This makes accuracy be the least important factor. The accuracy score is not simply setting cut-offs for the given accuracies given in the papers, but a blend of understanding how the algorithms did compared to others on the same data sets, and how hard the data sets were to classify. For example, ME has the second lowest reported classification in the SLR but was tasked to classify between five closely related languages spoken in India. However, while it did slightly outperform GMM, the data set that GMM was tasked with was 27 closely related languages spoken in India.
- For establishment in the field, the difference between score is high as some algorithms are established benchmarks that have been used and optimized for decades, while others are proof-of-concepts that would have no tools. This makes establishment in the field be the most important factor. Algorithms that were used as benchmarks and created long ago scored higher, and those that were repeatedly used as benchmarks scored even higher.

- For being able to handle low resource languages, the relative differences are high but the value of the concept is low due to this research relying on translation services that will not work on low resource languages, though over time they might. This makes the ability to handle low resources languages a semi-important factor. Algorithms that can share layers between similar languages score higher as they can be trained without needing more of the correct language. Test cases in the SLR that used smaller data sets, yet scored higher in accuracy, also score higher.

The qualitative factors of the algorithms discovered through the SLR helped strengthen the confidence that that the i-vectors algorithm is the best algorithm to detect a spoken language. The i-vector algorithm was shown to have an adequate level of accuracy needed for this research. It was also shown to be extremely well established in the field. Finally, it was shown to be able to handle low resource languages extremely well. Although the i-vector algorithm was tied with GMM as scoring the highest, i-vector is slightly more established and has specifically outperformed GMM on the same dataset.

Both the quantitative and qualitative analysis show the i-vector algorithm is the best algorithm for this research. Quantitatively, the i-vector algorithm was used the most and had a reliably high accuracy. It is a popular algorithm because of its age, first created in 2006 [39] with significant optimizations with joint factor analysis in 2011 [40] and hybridized with competing theories in 2014 [41]. The algorithm also performs with high accuracy as it is able to represent acoustic variations of speech utterances of varying durations as a fixed-length feature vector [18]. Qualitatively, the i-vector algorithm showed an adequate level of accuracy, high establishment in the field, and a high capability to handle low-resource languages. Results from LID competitions run every few years by NIST showed how the i-vector algorithm was reliably accurate compared

to many other algorithms using the same dataset [13] [15] [16]. This algorithm is also defined as "state-of-the-art" [14] [15] [18] by the papers in the SLR. Finally, the i-vector algorithm was able to retain a high accuracy even with languages with only an hour or less of speech data [13] [14] [15] [16]. The research findings of the SLR have clearly identified the i-vector algorithm to be the best algorithm for the thesis.

Chapter 3. Spoken Language Detection

Method

The steps required to automatically detect a spoken language consisted of finding a database of spoken languages, using the i-vector algorithm to extract features from each data point, training a neural network on the features, and finally using the generated model to detect a spoken language. The database, Mozilla Common Voice[1], was discovered through the papers that were read as part of the SLR. The i-vector algorithm, implemented with inspiration from Kaldi [42], that was proven through the SLR to be the best algorithm for this use case was first used with the Kaldi platform before being converted into a more manual method within the environment. The neural network, Tensorflow[2], was used to train the model as well as evaluate what language is being spoken. Figure 7 shows the high-level process.

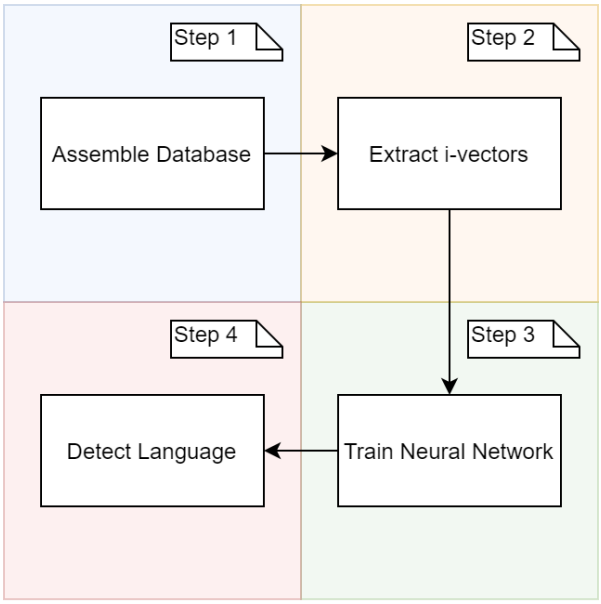
---

[1] Mozilla, "Common Voice." https://commonvoice.mozilla.org/ (accessed Dec. 18, 2020).

[2] TensorFlow, "TensorFlow Core | Machine Learning for Beginners and Experts," Google. https://www.tensorflow.org/overview (accessed Dec. 18, 2020).

Figure 7

Method to Automatically Detect Spoken Language



The dataset used (see Step 1 in Figure 7) in this research was discovered from papers read while conducting the SLR. Databases containing thousands of hours of speech from a variety of speakers were used for the algorithms that were considered as part of the SLR. While some of these databases were either not available to the public or had a significant cost, some were completely available to the public with no cost. Databases with only one speaker recording themselves were not used regardless of how many hours of speech were recorded due to the importance of training the algorithm in this research is to not rely on any characteristics a single speaker has for the entire language classification. This is especially difficult for low-resource languages, though the i-vector algorithm was identified also to work well even with smaller datasets as the qualitative analysis in the SLR found. Due to being unable to account for every type of accents due to the very large combinations of languages needed, the decision was made to not account for any specific accents and instead rely on the algorithm to identify the specifics

of speaking a language, regardless of the accent. The SLR provided a large list of publicly available databases of many speakers speaking many languages.

The selected database, Mozilla Common Voice, is the perfect database for this research. At the time the database was assembled, 9,300 hours of speech data was available, 7,400 of which were validated, for 60 different languages. This incredibly large database is also available to the public with no associated cost. The database also has a large amount of speakers for each language, with English for example having 66,151 unique speakers[3]. These speakers have many different accents to eliminate any pronunciation bias from determining the language, with English for example having over 17 accents. The database also consists of low-resource languages, even having the previously identified low-resource language of Urdu[4] [37].

The assembled database contains seven (7) languages due to feasibility of testing in a Canadian military environment as well as training time. The languages are English, French, Russian, Chinese (China), Arabic, Persian, and German. English and French are obvious choices for testing purposes as almost all Canadians can speak either or both of Canada's official languages. Linguists working on a military base would be trained in the language of Canada's adversaries, meaning there will be linguists who speak Russian, Chinese (China), Arabic, and Persian. Finally, German was selected as a language to represent a European language, which is also a popular secondary language to learn. Table 7 contains the statistics for each language in the database.

---

[3] Mozilla, "Datasets," *Mozilla*, Dec. 11, 2020. https://commonvoice.mozilla.org/en/datasets (accessed Dec. 18, 2020).

[4] Mozilla, "Languages," *Mozilla.* https://commonvoice.mozilla.org/en/languages (accessed Dec. 18, 2020)

Table 7

Statistics of Each Language Selected in the Assembled Database in This Research

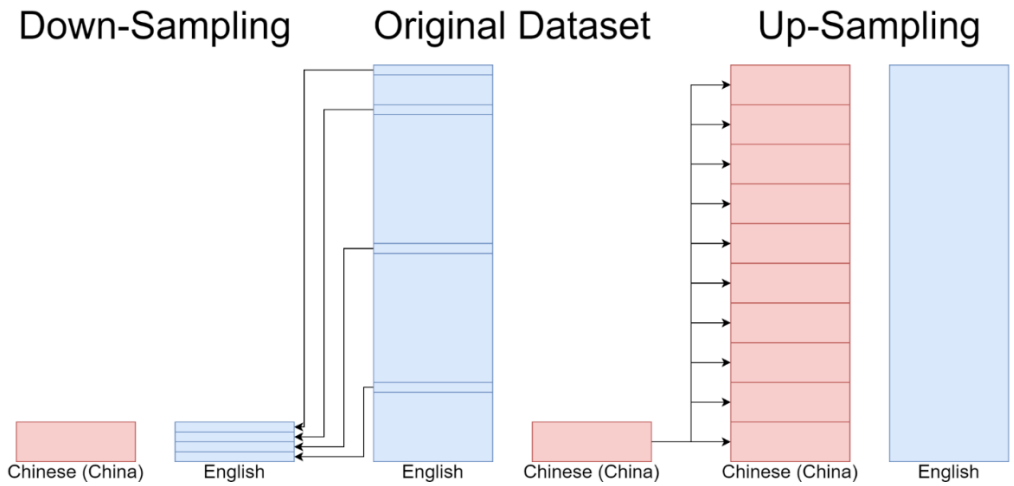
Language	Size (GB)	Validated Hours	Number of voices	Number of clips
English	56	1,688	66,151	1,226,615
French	18	621	12,950	459,109
Russian	3	130	1,410	74,370
Chinese (China)	2	56	3,501	36,472
Arabic	2	45	659	39,953
Persian	8	284	3,654	253,592
German	22	777	12,655	565,087
Total	111	3,601	100,980	2,655,198

As this research was fortunate to have such a large dataset, down-sampling is used to account for the imbalances in data availability. The simplest solution for data imbalance is in resampling strategies, as is shown in Figure 8. The largest imbalance of data is between Chinese (China) and English, having 36,472 and 1,226,615 clips, respectively. As having 36,472 was observed to be an acceptable amount of data based on testing and training time, each language could be down-sampled to that many clips. Had this not been enough, more advanced techniques than up-sampling could be employed to make up for this fundamental problem of dealing with imbalanced data in data science[5]. By employing down-sampling, 36,472 clips from each language were assembled to create the database for this research.

[5] TensorFlow, "Classification on imbalanced data | TensorFlow Core," *Google*. https://www.tensorflow.org/tutorials/structured_data/imbalanced_data (accessed Dec. 21, 2020).

Figure 8

Data Resampling Strategies



To ensure the database has the correct file formats, FFmpeg[6] was used to convert the mp3 files into wav files. Kaldi's implementation of i-vector extraction requires data to be in the wav format[7], which is a costly format for storage. The dataset provided by Mozilla uses two methods to reduce the amount of storage required, by compressing the language sets into tar files, and more importantly, storing the voice clips as mp3s. As the original data is an mp3, a file format that is compressed, converting it back to a wav file does not add any new information, instead just holding it in a larger container. FFmpeg does an excellent job at quickly converting mp3 files to wav files, however each file converted is around 10 times larger. This means the database of 111 GB is now 1.11 TB if all files are to be converted.

As down-sampling is being used, the first approach to handle this storage issue was to only convert the clips that were identified to be used for training and testing, which brings the storage

---

[6] FFmpeg, "FFmpeg," *telepoint*. https://ffmpeg.org/ (accessed Dec. 22, 2020).

[7] Kaldi, "Kaldi: Data preparation," *doxygen*. https://kaldi-asr.org/doc/data_prep.html (accessed Dec. 22, 2020).

requirements of the converted clips to be 2 GB per language multiplied by seven (7) languages multiplied by 10 to be 140 GB, which is close to the original database storage requirements. This approach was optimized further when it was decided to implement Kaldi's i-vector extraction into the python environment, which allowed file conversions to only be done through FFmpeg when used. While this did lower the storage requirement down to 14 GB, it has the cost of increasing the processing time as files will be converted multiple times instead of just once. This processing time cost, when compared to the processing time of training the neural network, was deemed to not be significant. FFmpeg is able to efficiently convert the mp3 files in the dataset into wav files for i-vector feature extractions.

With the database assembled, a method to extract i-vectors (i.e., Step 2 in Figure 7) was required which was first implemented with Kaldi [42]. While other tools exist for i-vector feature extraction, such as bob[8], Kaldi has extremely in-depth documentation and is a very popular tool for signal processing. It also hosts a GitHub with many different examples and applications[9]. As it is hosted on GitHub, it is by nature an open-source toolkit which allows one to inspect and tweak the code to run as efficiently as possible. While Kaldi is written primarily in C++, it is wrapped with Python and bash scripts to make the required calls. Due to its documentation and popularity, as well as it being open-source with many examples and flexible in its programming languages, Kaldi was the selected method to extract i-vectors from the assembled database.

---

[8] Idiap Research Institute, "Bob — bob 8.0.0 documentation," *Idiap Research Institute*. https://www.idiap.ch/software/bob/docs/bob/docs/stable/index.html (accessed Jan. 12, 2021).

[9] kaldi-asr, "kaldi," *GitHub*, Jan. 13, 2021. https://github.com/kaldi-asr/kaldi (accessed Jan. 12, 2021).

The biggest limitation of Kaldi that needs to be accounted for is that it is only supported on Linux. This is an issue as the environment so far has been on Windows. While there is documentation for how to implement Kaldi in Windows, it is a very inferior version[10]. As the process for this research is broken down into the steps shown in Figure 7, the solution for this issue was to simply pipe the database into a virtual machine running Linux, extract the i-vector features, then pipe the features back to the Windows environment for training. For detecting the language, the recorded voice clip would again have to be piped into the virtual machine to have the i-vector feature extracted before having the features be used in the classification model. By sending data between virtual machines, Kaldi's limitation of only being available on Linux could be averted.

While implementing a virtual machine to use Kaldi was functional, it was deemed to not be an acceptable solution when considering the research holistically. Piping data between the operating systems allowed the system to function, and some scripting allowed the process to be automated by utilizing a shared folder that both operating systems had access to. However, the end result of this research is to be deployed on an application running on a phone. Implementing this design of using two different operating systems is not a feasible option. Although the training could be done using this method to generate the model that would be deployed on the phone, the phone would still require Linux to extract the i-vectors of what is being spoken for the classification. Due to needing the algorithm to run in a single environment, the working algorithm using Kaldi was deemed unfit for this research.

---

[10] kaldi-asr, "kaldi/windows/INSTALL.md," *GitHub*, Apr. 08, 2020. https://github.com/kaldi-asr/kaldi (accessed Jan. 12, 2021).

A potential solution to continue to use Kaldi for i-vector relied on online access. The Kaldi toolkit has an implementation that extracts i-vector features online[11]. While this component still requires a Linux operating system, it could be wrapped in python so that it could be called in any environment. There are however three large issues with this solution. The first is that one of the intended applications of this research is for military communications, which should be robust. The requirement for the phone to be connected to the internet is deemed to not be acceptable, as internet is not always available in combat scenarios. The second issue is another dependency issue, which relies on Kaldi to continue their online services indefinitely. The third and final issue is that this implementation creates a privacy issue, as now voice clips are being sent online to a server outside of the research's control. The dependency and privacy issues of implementing Kaldi online deemed it an unfit solution for i-vector extraction.

Although the toolkit Kaldi could not be used itself in this research, it could be implemented into the working environment. As the toolkit is open-source, all of the code to extract the i-vector features is available. By looking through the C++ code, an implementation could be rewritten in any other language. Kaldi's i-vector extraction method was re-written in Python, leveraging some powerful python packages to take care of some of the more difficult steps. Creating an algorithm that is an implementation of Kaldi into a python environment was the solution to extract i-vector features.

With the i-vector feature extraction method completed, a model can be trained on them using TensorFlow (i.e., Step 3 in Figure 7). This machine learning framework is an open-source library

[11] "Kaldi: online2/online-ivector-feature.h File Reference," *doxygen*. https://kaldi-asr.org/doc/online-ivector-feature_8h.html#details (accessed Jan. 12, 2021).

created by the Google Brain team for internal Google use [43], with plenty of documentation and examples available[12]. It is available exclusively on Python and has many of the packages required to conduct machine learning training and classification. The rest of the environment for this research is based on the implementation of TensorFlow.

The environment for the algorithm part of this research that utilizes TensorFlow is built using two systems, Visual Studio Code and Anaconda. Visual Studio Code[13] is a highly customizable Integrated Development Environment (IDE) that has many useful extensions. A key extension used in this research was its Bitbucket extension[14], which allows the work to continually be synchronized at an online repository. While this IDE can be used to program in many different languages, Python is used as it is the language used for interacting with TensorFlow. To set up the Python environment inside of the IDE, Anaconda was used. Anaconda[15] is a data science platform that not only provides popular data science packages, but aids greatly in the configuration of them. This is very beneficial as TensorFlow was incredibly hard to install and optimally configure manually. By utilizing the IDE Visual Studio Code and the data science platform Anaconda, TensorFlow was easy to set up and use.

---

[12] TensorFlow, "Guide | TensorFlow Core," *Google*. https://www.tensorflow.org/guide (accessed Jan. 13, 2021).

[13] Microsoft, "Visual Studio Code - Code Editing. Redefined," *Microsoft*. https://code.visualstudio.com/ (accessed Jan. 13, 2021).

[14] Visual Studio Marketplace, "Visual Studio Bitbucket Extension," *Microsoft*. https://marketplace.visualstudio.com/items?itemName=MistyK.VisualStudioBitbucketExtension (accessed Jan. 13, 2021).

[15] Anaconda, "Anaconda | The World's Most Popular Data Science Platform," *Anaconda*. https://www.anaconda.com/ (accessed Jan. 13, 2021).

One of the key steps to effectively using TensorFlow was to ensure it was using the graphics card instead of the processor that the IDE would normally use. While using the Central Processing Unit (CPU) is possible with TensorFlow, training is much faster when using the dedicated Graphics Processing Unit (GPU). The computer that conducted the training in this research used an Operating System of Windows 10 and had 8GB of RAM, a 1TB hard drive, and most importantly an i5-3750 3.40 GHz CPU and a GTX 970 GPU. Switching TensorFlow training from this CPU to this GPU decreased the training time by over 80%. It is much faster as TensorFlow can use Compute Unified Device Architecture (CUDA), which is a parallel computing platform developed by NVIDIA[16]. The GTX 970 was specifically purchased for this research, as it has a CUDA computing capability of 5.2[17] which is just over the minimum requirement, as well as having 1,664 CUDA cores[18]. After installing all the required software and manually setting environment path variables, provided in Table 8, TensorFlow is able to automatically use the GPU instead of the CPU. By utilizing the GPU instead of the CPU, TensorFlow can train models over five times (5x) faster.

---

[16] NVIDIA, "CUDA Zone," *NVIDIA Developer*, Jul. 18, 2017. https://developer.nvidia.com/cuda-zone (accessed Jan. 13, 2021).

[17] NVIDIA, "CUDA GPUs," *NVIDIA Developer*, 2021. https://developer.nvidia.com/cuda-gpus (accessed Jan. 13, 2021).

[18] A. Kuznetsov, "Nvidia GPUs sorted by CUDA cores," *GitHub*. https://gist.github.com/cavinsmith/ed92fee35d44ef91e09eaa8775e3284e (accessed Feb. 14, 2021).

Table 8

Required Software and Environment Variables to Have TensorFlow Utilize the GPU

Required Software	Environment Variable
NVIDIA GPU Drivers	N/A
CUDA Toolkit	SET PATH=[installed path of]\NVIDIA GPU Computing Toolkit\CUDA\v11.0\bin;%PATH% SET PATH=C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.0\include;%PATH%
CUPTI	SET PATH=[installed path of]\NVIDIA GPU Computing Toolkit\CUDA\v11.0\extras\CUPTI\lib64;%PATH%
cuDNN SDK	SET PATH=[installed path of]\cuda\bin;%PATH%
TensorRT	N/A
tensorflow-gpu	N/A

With TensorFlow functioning correctly, the model architecture could be created for training. The model used in this research is called a Convolutional Neural Network (CNN) and its kernel layers are detailed in Figure 9. A CNN is typically used for training on images, as the stated in the documentation for the first layer of the model, Conv2D[19]. The reason an image processing layer is used is because the output of the audio features are spectrograms which are an image. The convolution layer summarizes the features present in the image.

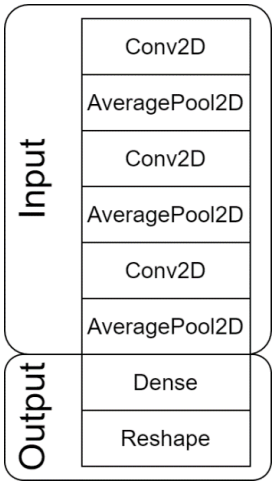
The three (3) convolution layers are interchanged with three (3) pooling layers, specifically AveragePooling2D[20]. Pooling is an essential part of the model to help prevent model overfitting. By down-sampling the image between each layer, the features learned in the convolutional layers will be more focused on general features instead of specific oddities that might exist [44]. This pooling layer in particular reduces the dimensions of the image by half, using an average value for each patch of the map.

[19] TensorFlow, "tf.keras.layers.Conv2D | TensorFlow Core v2.4.0," *Google*. https://www.tensorflow.org/api_docs/python/tf/keras/layers/Conv2D (accessed Jan. 13, 2021).

These six (6) layers are followed with a layer to create an output of the model, the dense layer[21]. This final layer computes the dot product between the input and the kernel, giving it a single value. One final layer is used to ensure this aligns with the expected classifications, the reshape layer[22]. By shaping the final layer of the output to consider the different languages that are classified, the model can compare the final dense layer output to give the language spoken. TensorFlow was used to generate a CNN model with three convolution layers, three pooling layers, a dense layer, and a reshape layer.

Figure 9

Model Kernel Layers



The method in which TensorFlow trains a model is in batches and epochs. The batch size denotes how many data training samples are observed before updating the weights [45]. Training

---

[20] TensorFlow, "tf.keras.layers.AveragePooling2D | TensorFlow Core v2.4.0," *Google*. https://www.tensorflow.org/api_docs/python/tf/keras/layers/AveragePooling2D (accessed Jan. 13, 2021).

[21] TensorFlow, "tf.keras.layers.Dense | TensorFlow Core v2.4.0," *Google*. https://www.tensorflow.org/api_docs/python/tf/keras/layers/Dense (accessed Jan. 13, 2021).

[22] TensorFlow, "tf.keras.layers.Reshape | TensorFlow Core v2.4.0," *Google*. https://www.tensorflow.org/api_docs/python/tf/keras/layers/Reshape (accessed Jan. 13, 2021).

samples per batch has been set to 128. This strikes a balance between speed and accuracy, while also statistically ensuring that no batch is missing one of the seven (7) languages being considered. Each batch is also a random selection of the training data, so that the model is statistically never trained on a single language.

An epoch represents a single pass through all training data that is contained in those batches. For this dataset, 1,595 batches of 128 are used for training data. This means that one epoch considers 204,160 training data samples. After completing each batch, the current accuracy of the model can be given. To do this, the data needs to be split into training and testing data. A split is done during the batch allocation so that 80% of the data is used for training, and 20% of the data is used for testing. This allows the model to learn from misclassifications as well as the ability to generate a plot showing how the model becomes more accurate as it continues to be trained.

With the model architecture completed, it can now be trained and used to detect the spoken language (see Step 4 in Figure 7). After training, the model and its assets can be saved and loaded with ease. Voice clips are recorded and placed in a directory to be read and classified. Once the voice clip is read, it has its i-vector features extracted using the same method the database uses. The features are then given to the model, and a classification is given. By using a similar method to how the voice clips in the database were formatted for training, new voice clips can be classified using the trained model to deliver the classification of the language being spoken.

Algorithm

The algorithm to extract i-vector features from voice clips was implemented from Kaldi [42]. Many of the examples hosted by Kaldi, referred to as recipes, contain their own implementation

of an i-vector extractor, such as extract_ivectors.sh[23] in the recipe wsj, which aimed to detect words being spoken from sentences being read from the Wall Street Journal. These extractors typically contain significantly more functions and lines of code than required, as they are derived from the extremely large Kaldi file ivector-extract.cc[24]. By using the files created by Kaldi as well as the recipes they have provided, an implementation of an i-vector feature extractor is possible.

To load and manipulate audio files, the package librosa is used [46]. This tool can use the previously mentioned FFmpeg to load any type of audio file. Audio clips are loaded in with a sampling rate of 16,000 Hz. This rate is double for what is considered adequate for human speech, is the optimal sampling rate for capturing human language [47], and is the default sampling rate in Kaldi examples. The number of frames for the samples is set to 25 ms and the frame step is set to 10 ms as recommended by Kaldi[25]. At the sample rate of 16,000 Hz, this creates 400 samples. The frame step is however 160 samples, meaning that after sample 160, the next 400 samples being to overlap. This is illustrated in Figure 10. Audio files can be loaded and split into samples using the package librosa along with FFmpeg.

---

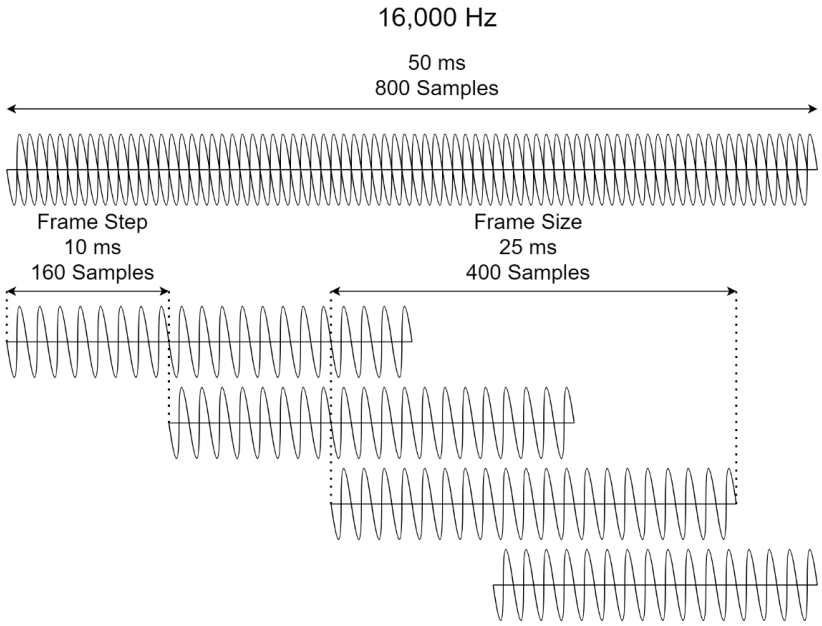
[23] Kaldi, "kaldi/egs/wsj/s5/steps/nnet/ivector/extract_ivectors.sh," *GitHub*. https://github.com/kaldi-asr/kaldi (accessed Jan. 14, 2021).

[24] Kaldi, "Kaldi: ivectorbin/ivector-extract.cc File Reference," *doxygen*. https://kaldi-asr.org/doc/ivector-extract_8cc.html (accessed Jan. 14, 2021).

[25] Kaldi, "Kaldi: Feature extraction," *doxygen*. https://kaldi-asr.org/doc/feat.html (accessed Jan. 14, 2021).

Figure 10

Frame Layout of Audio Samples



The librosa package can also be used as a feature extractor by generating Mel Frequency Cepstral Coefficients (MFCC). Although MFCC was developed over 30 years ago [48], it is still one of the best performing methods to shape sounds. This is because it represents the audio in a way that humans understand it, rather than simply in terms of pure frequency. As psychophysical studies have proven that humans perceive sound a way that does not follow a linear scale, each tone with a subjective pitch is measured on a scale called the Mel Scale [49]. This subjective pitch attempts to mimic the human cochlea which vibrates at different spots depending on the frequency, using the calculation of the power spectrum of each frame. Frequencies can be converted into the Mel Scale with the following formula: $m = 2595\log_{10}(1 + f/100)$ [49]. Using the layout of samples librosa loads, 40 MFCCs are created per audio file for a low frequency sample rate as recommended by Kaldi. Each MFCC considers 1,001 utterances, which has been shown to be ideal in testing by other researchers [50] [51]. By understanding how to change

audio into data that represents how humans hear audio, librosa can generate MFCCs to start the process of training a model.

With each audio file converted into MFCCs, the i-vector feature extraction can be combined with TensorFlow to start training the model. In addition to CUDA GPUs being significantly quicker at training models as previously discussed, CUDA GPUs can also extract i-vectors. In 2020, NVIDIA implemented a feature in its CUDA platform to extract i-vectors from MFCCs, based specifically on Kaldi [52]. This also allows i-vectors to be extracted quicker, as by using the GPUs, many extractions can be done in parallel. Batches of audio files can then be converted into MFCCs and be sent to TensorFlow for training, which will use CUDA to not only train faster, but also extract i-vectors before training.

Test Cases and Improvements

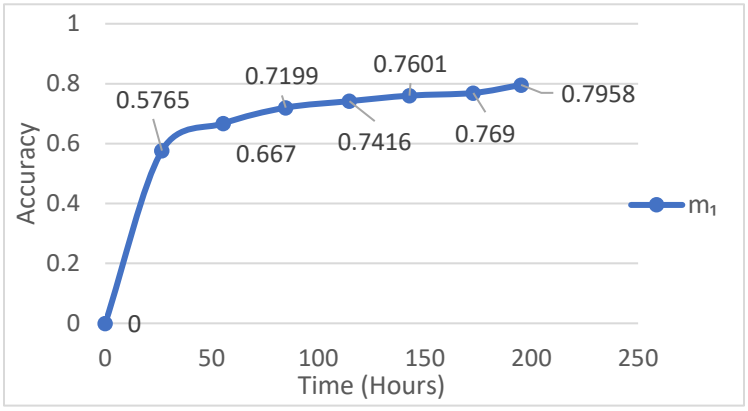
Utilizing the design and algorithms previously mentioned, a model (m_1) was trained to detect a spoken language. The model was trained with seven (7) epochs, with each epoch taking on average 27 hours to complete. After completing the final epoch, the model reported an accuracy of 80% using the randomly selected testing data from the training database. To contextualize this accuracy, a random classification between seven (7) languages would have an accuracy of 14%. This shows that the model has been able to learn the differences between languages and can classify them.

Figure 11 shows the training results of the model (m_1). Epochs are represented by each marker, with the seventh epoch achieving a 80% accuracy over around 200 hours of training time. A curve first occurs after the first epoch, which continues until the second last epoch. Interestingly, the slope increases on the last epoch, showing a sudden increase in the accuracy momentum. As

the model continues to increase by a fair amount after each epoch, there is little risk that over-fitting is being done and more training epochs can be done to increase the accuracy even more.

Figure 11

Model (m_1) Accuracy Over Time, at Each Epoch



The cost of using the incredibly powerful TensorFlow and CUDA platforms is volatility. Although the model (m_1) should have only required around 200 hours, or just under eight (8) days to complete, in practice it took over a month. This is because the model (m_1) had frozen, crashed, or become corrupted numerous times. Fortunately, TensorFlow has a checkpoint functionality[26], which allowed the model (m_1) to continue training after an interrupt. The checkpoints were however not reliable, often requiring the model to restart at the last epoch that was completed. As the training utilized nearly 100% of the GPU, interacting with the machine was very risky. Using a program such as Google Chrome, which can use the GPU for some resource intensive webpages, was observed to create a busy loop as the model would not move onto another batch, yet continue to use 100% of the GPU indefinitely.

[26] TensorFlow, "Training checkpoints | TensorFlow Core," *Google*. https://www.tensorflow.org/guide/checkpoint (accessed Jan. 16, 2021).

Training would also often be interrupted with an error stating “ValueError: frames must be specified for non-seekable files.” This would imply that a file is corrupted. By having the model print what file was being read while training, these files that caused errors were seemingly random and were valid files. This was proven by it being possible for a full epoch to be trained, so this error message was not accurately stating the issue. A solution was to use a try and catch method around the batch, which would work for all cases except the last batch. Unfortunately, this error often occurred on the last batch of the epoch and would also corrupt the TensorFlow checkpoint, causing around 27 hours of training to be wasted. While using TensorFlow and CUDA for big datasets, the process needs to be continually monitored to reduce the amount of wasted time.

Although the model (m_1) reports an accuracy of 80%, this value does not denote the true accuracy of the model and requires a separate testing database. There is an inherent bias to using the same database to produce training and testing data. The importance of the given accuracy is that is shows that the model is progressing. A much better measurement of the accuracy needs data that has no relation to the data that was used for training purposes. Although an accuracy of 80% is well above randomness, a separate testing database is needed.

A separate testing database was created to denote the true accuracy of the model (m_1), which is shown in Appendix A. For each language, 10 voice clips were created, with English and French clips having 15 more clips so that the database has 100 clips. Different languages had a different number of users in each clip, such as German having most of the voice clips from one individual, and Arabic having all unique speakers. The complexity of what is said differs as well, Chinese (China) had more complex sentences, while Persian had very basic questions consisting of a small number of words. Some of the voice clips also have background noise, speaking mistakes,

and displays of different emotions. The testing database created independently from the training database will give a truer accuracy of the model's ability to detect a spoken language.

The accuracy of the model (m_1) on the testing database is 38% (see Table 9). While this number is much lower than the 80% accuracy seen with the testing data from the training database, the 38% accuracy gives a better idea of how it will classify the more realistic audio it will be classifying. While this value is low, it remains higher than the 14% accuracy that random classifications would give, which indicates the model did at least learn something about the differences between languages.

Table 9

Accuracy and Time-Spent Recognizing a Spoken Language With Model (m_1)

Language	Accuracy	Average Time-Spent
English	28%	0.44s
French	24%	0.44s
Russian	40%	0.44s
Chinese (China)	100%	0.44s
Arabic	60%	0.44s
Persian	40%	0.44s
German	10%	0.44s
Overall	**38%**	**44s**

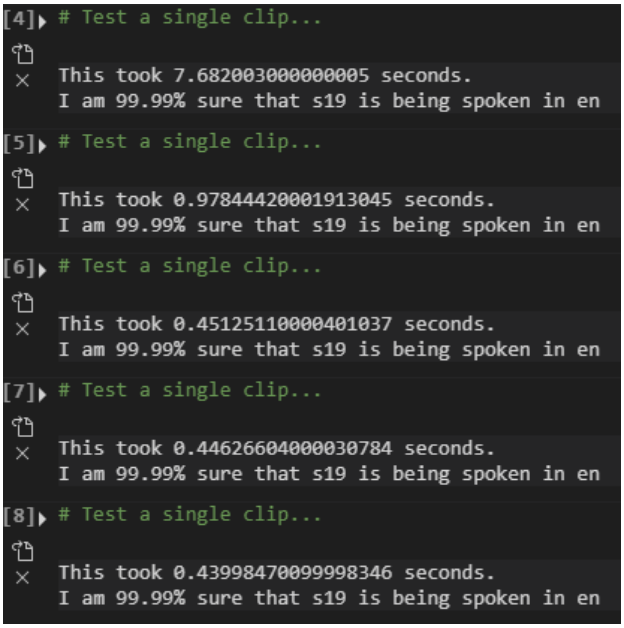
It is also interesting to note that the model (m_1) had an accuracy of 100% when classifying Chinese (China). While this could be attributed to the voice clips being longer and more complex, the accuracy was not high for English and French which were also longer and complex according to the results Table 9 lists. There must therefore be a significant difference between Chinese (China), and the other languages. The accuracy of the model (m_1) using testing data from a separate database than the training database displays that the true accuracy is much lower.

The average time spent contained in Table 9 is based on the voice clip being tested 100 times after warming up the model. The times are all very similar as, although the voice clips will

contain different data for each language, the data is captured in the exact same container for processing. Unlike a language such as C, Python sacrifices code optimization for greater user functionality. This causes the first few classifications to take significantly more time as Python begins to optimize systems calls and memory management for repeated functions. Figure 12 shows how the time to classify a voice clip shrinks as the same code snippet is repeatedly called. To account for this behaviour, the times captured in Table 9 were averaged after warming up the model with five (5) classifications.

Figure 12

Python Calls for Model Classification Hasten After Repeated Calls



The accuracy of the model (m_1) must be improved by observing the three characteristics of a good model, amount of data, amount of training data, and quality of data. Having a good amount of data means that the training will consider many permutations of the thing it is trying to classify, reducing the chance a new classification has no elements it has seen in training. The amount of training time is important for the model to confirm assumptions being made. Training for too long has the possible unwanted side-effect of overfitting, which means it will only

classify objects that are exactly what it was trained on. Finally, the quality of data is important as if during training it is learning aspects of a wrongly classified object, it will not recognize the real object in future classifications. By looking at the amount of data, the quality of data, and the amount of training time for the current model (m_1), it can be improved and a higher accuracy can be achieved.

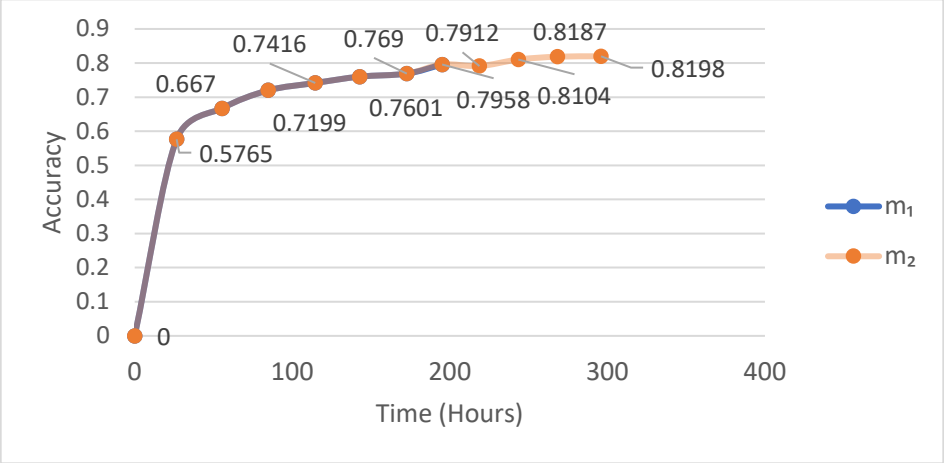
For the amount of data considered in the model (m_1), little can be improved. There are already hundreds of hours of audio for each language, to the point where data storage is already an issue. One solution to increase the amount of data is to up-scale the database instead of down-scaling, however this would lead to unwanted biases. This would also increase the time for a single epoch to increase tenfold, taking over two (2) months to complete, which is not acceptable. The number of data is the one characteristic that can be left alone.

For the training time, there are simpler and more complex solutions. The simpler solution is to simply train the model (m_1) for more epochs. As the accuracies continued to rise at a steady state, it can be inferred that running a few more epochs would increase the accuracy. While this is an easy solution, the issue lies in how volatile the system is. As each epoch takes significantly more time to successfully complete due to the volatility of the system, the model cannot simply be trained by 30 more epochs. A more complex solution can avoid this issue though. By pre-processing the data before the training, the training epochs can become significantly shorter. The current implementation of i-vector feature extraction however relies on data being processed while being trained. Coming up with a solution to account for this can be investigated if it can be proven that more training time would indeed increase the accuracy. The accuracy should increase if more training epochs are permitted, which is only feasible if data is pre-processed, a complex issue given the current implementation of i-vector feature extraction.

To prove that more training epochs would increase the accuracy, as well as warrant the need to do data pre-processing, a second model (m_2) was created and tested, with the results shown in Appendix A and training output illustrated in Figure 13. This model (m_2) was trained for 11 epochs instead of just the seven (7) epochs permitted for the first model (m_1). The accuracy using the training dataset increased steadily as predicted, with a slope that indicates even more training epochs would result in a higher accuracy. The accuracies are similar for the first seven (7) epochs because the same random seed is used for the down-sampling and which data is used as training or testing. After around 300 hours of training time, the model (m_2) was able to achieve an accuracy of 82%, meaning 100 hours of training time more than the first model (m_1) achieved an accuracy four (4) points higher.

Figure 13

Models (m_1 and m_2) Accuracy Over Time, at Each Epoch



To better understand the accuracy, Table 10 lists the results showing that the model (m_2) achieved an accuracy of 40% on the testing database, which is shown in Appendix A. This is two (2) points higher than the first model (m_1). The second model (m_2) achieving an accuracy of 82% during training and 40% during testing indicates that it is worthwhile to investigate how to conduct data pre-processing so that many more epochs can be completed.

Table 10

Accuracy and Time-Spent Recognizing a Spoken Language With Model (m_2)

Language	Accuracy	Compared to m_1	Average Time-Spent
English	40%	↑ (from 28%)	0.45s
French	40%	↑ (from 24%)	0.45s
Russian	50%	↑ (from 40%)	0.45s
Chinese (China)	70%	↑ (from 100%)	0.45s
Arabic	40%	↓ (from 60%)	0.45s
Persian	50%	↑ (from 40%)	0.45s
German	0%	↓ (from 10%)	0.45s
Overall	**40%**	**↑ (from 38%)**	**45s**

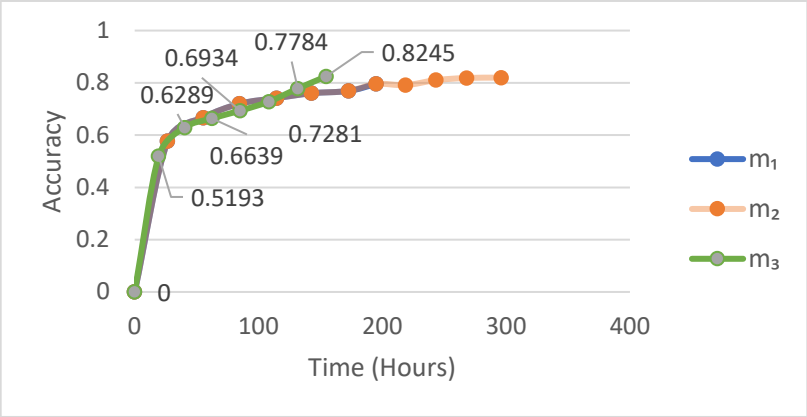
For the quality of data, there is much that can be improved. The dataset uses a crowd-sourced method of validating clips. Random internet users can listen to clips and compare it to what is supposed to be being said, and either upvote it or downvote it. Clips are considered to be validated if the upvotes are simply higher than the downvotes. Many of the clips only have a single upvote and no downvote, meaning a lot of trust is being put into a single random internet user. By manually listening to a few hundred clips, many were found to be considered validated but were simply static, music without lyrics, or simply nothing. It is an almost impossible task for one individual to go through the 2,655,198 voice clips in the training database and validate them all. One simple solution is to increase the threshold of the already employed validation method, increasing the amount of upvotes that are needed for each downvote. Due to the sheer amount of data however, some anomalies sprinkled in the dataset may not actually impact the training. Looking further, more complex solutions to filtering the data can be done, such as limiting the lengths of clips, if it is proven that data filtration will improve the accuracy. The accuracy should increase if the data is better filtered for quality, which can invoke more complex filtration criteria than relying on random internet users.

To prove that more data filtration would increase the accuracy, as well as warrant the need to do more complex data transformations, a third model (m_3) was created and tested, with the results

shown in Appendix A and training output illustrated in Figure 14. This model (m_3) was trained with voice clips that had at least four (4) times as many upvotes as downvotes by the random internet users. This resulted in the down-sampled training database having 212,247 voice clips, with the Chinese (China) clips again setting the down-sampled threshold. This is 17% less than the 255,304 clips used in the training database for the first two models (m_1 and m_2), meaning epochs were completed relatively faster. The model (m_3) was trained for seven (7) epochs just as the first model (m_1) for easier comparison. While the accuracies during training were less than the first model (m_1), the model (m_3) maintained a stronger slope, achieving an accuracy of 82%. This may just be one more point than the first model (m_1), however the testing database accuracy is even more important than usual here as the first model may have been scoring a higher accuracy due to thinking a language was simply static noise, as it was trained on.

Figure 14

Models (m_1, m_2, and m_3) Accuracy Over Time, at Each Epoch



On the testing database, Table 11 lists the results showing that the model (m_3) achieved an accuracy of 46%, six (6) points above the first model (m_1). The third model (m_3) achieving an accuracy of 82% during training and 46% during testing indicates that it is worthwhile to investigate how to better filter the data so that the training database only has quality data.

Table 11

Accuracy and Time-Spent Recognizing a Spoken Language With Model (m_3)

Language	Accuracy	Compared to m_2	Average Time-Spent
English	64%	↑ (from 40%)	0.43s
French	44%	↑ (from 40%)	0.43s
Russian	0%	↓ (from 50%)	0.43s
Chinese (China)	90%	↑ (from 70%)	0.43s
Arabic	20%	↓ (from 40%)	0.43s
Persian	50%	= (from 50%)	0.43s
German	30%	↑ (from 0%)	0.43s
Overall	**46%**	↑ (from 40%)	**43s**

Now that is has been established that the accuracy will improve with more quality data and more training time, a fourth model (m_4) can be created exploring both concepts. The first step is to look at how the quality of the data can be improved. This can be done by limiting the amount of data in the testing database based on certain characteristics, as well as performing some data augmentation to standardize the voice clips. The second step is to look at how data can be pre-processed so that training epochs are much smaller. If features can be extracted before the training occurs, the training will be significantly faster. The fourth model (m_4) combines the strengths of the previous model improvements.

To increase the quality of the data, the training database can be used to create a separate filtered training database. The first filter which has already been applied in training the third model (m_3) was to increase the threshold for a validated clip based on upvotes from random internet users. The second filter is to ignore any voice clips that have no audio data in them. This is done by loading the voice clip as an array through librosa and verifying if unique values exist in the array. The third filter is to only use voice clips that lasted more than seven (7) seconds. Many of the voice clips are a single word or number, which is not ideal for training as it applies too much emphasis on a small number of sounds compared to sentences being read. Once voice clips pass these three filters, they are copied to a new location. This created a new filtered training database

of 29,820 clips. While Chinese (China) has the lowest number of total voice clips, Arabic had the lowest number of voice clips that met these filters, down-sampling all languages to 4,260 clips each.

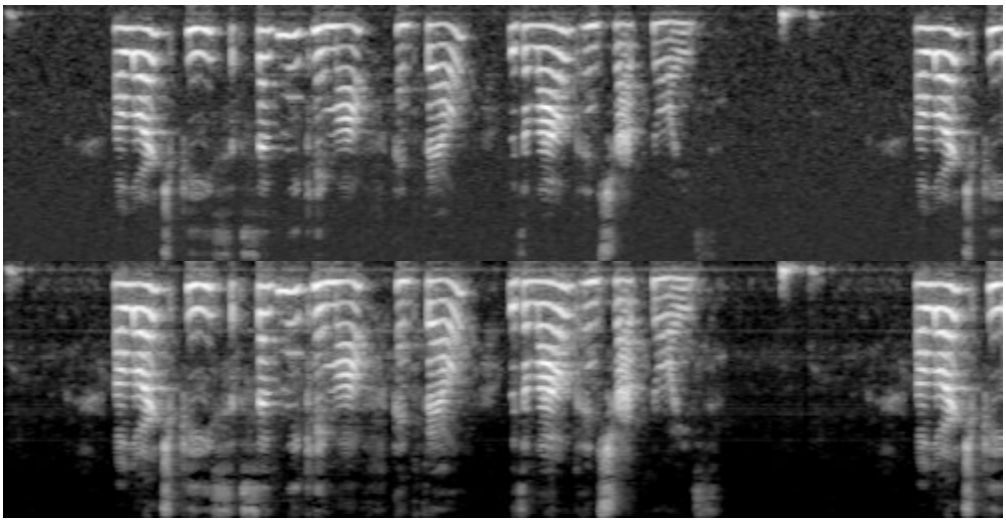
The quality of the data can be further improved through data augmentation. Almost all the voice clips in the database were likely recorded sitting in front of a computer based on the nature of how the voice clips were collected. As the model needs to be able to account for background noise to correctly classify voice captured in real world scenarios, noise can be added to the clips to train the model to focus better on a spoken voice. As librosa ultimately loads voice clips into an array, an array of the same size with a Gaussian distribution can simply be added to simulate background noise. This database of filtered voice clips with noise added can be included with the existing filtered database, creating a database of 59,640 voice clips. This will allow the model to be trained on both clips with background noise and clips without, achieving a better understanding of how to carve out a voice from noise. By augmenting the filtered database with a filtered database with added noise, the quality of the data can help generate a more accurate model when considering real world data.

Pre-processing can be used to reduce the amount of time needed to aptly train the model. Instead of doing computations while training the model, features can be extracted and stored for later use. The features can be stored as a spectrogram as an image. Figure 15 shows a spectrogram of a pre-processed voice clip with and without added noise. To standardize the images, each voice clip was also looped so that it is exactly 10 seconds long, which can be observed occurring at the very right of Figure 15. The model can then be trained by only reading in images, improving the speed of a training epoch greatly. By pre-processing the feature extraction into a spectrogram

image, the model can be trained quickly enough that even other, time consuming optimizations can occur.

Figure 15

Spectrogram of Voice Clip With (Top) and Without (Bottom) Added Noise



As the model can now be trained in a day, some more computational costing improvements can be done to the model. The six (6) layer kernel can be greatly expanded on, allowing for the training to consider far more aspects of the given features. Inceptionv3 [53] is a significantly larger CNN with 48 layers, optimized to classify image recognition models. Although this takes significantly more time to train, the pre-processing of features more than accounts for the time loss. By using the inceptionv3 CNN, the model can greatly benefit from a more complex kernel.

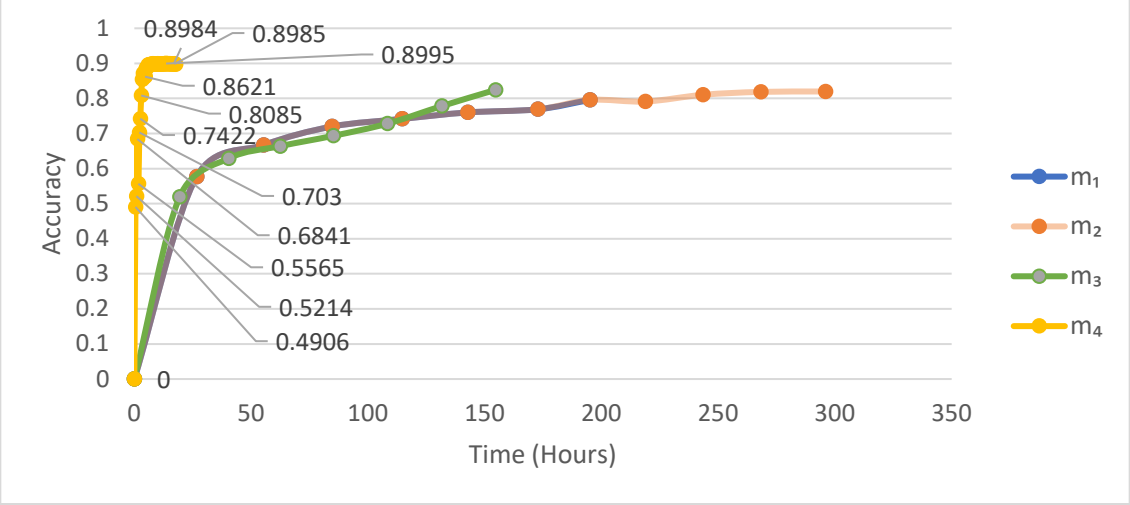
Now that the model can be fully trained, a risk that must now be mitigated is overfitting. Two approaches are used to avoid this, early stopping and learning rate decay. Early stopping uses the validation data to determine when the accuracy stops improving and forces the model to stop. The learning rate decay determines how much the model can impact its parameters at each training epoch, preventing the model from quickly reaching suboptimal conclusions. Combining

early stopping and learning rate decay ensures that the fully trained model stops once it reaches optimal conclusions.

A fourth model (m_4) was generated and trained with all of the improvements with an accuracy of 90%. Figure 16 shows its accuracy and training time compared to the previous models. Not only was the model trained significantly quicker, but it also achieved a much higher accuracy. Although the accuracy of 90% is only 10 points higher than the 80% accuracy of the first model (m_1), each point is much more difficult to obtain. It is therefore better to observe the difference in accuracy from the error percentage, meaning this model (m_4) had a 10% error rate compared to a 20% error rate, a 50% improvement. The accuracy of the fourth model (m_4) is significantly higher thanks to the improvements explored.

Figure 16

Models (m_1, m_2, m_3, and m_4) Accuracy Over Time, at Each Epoch

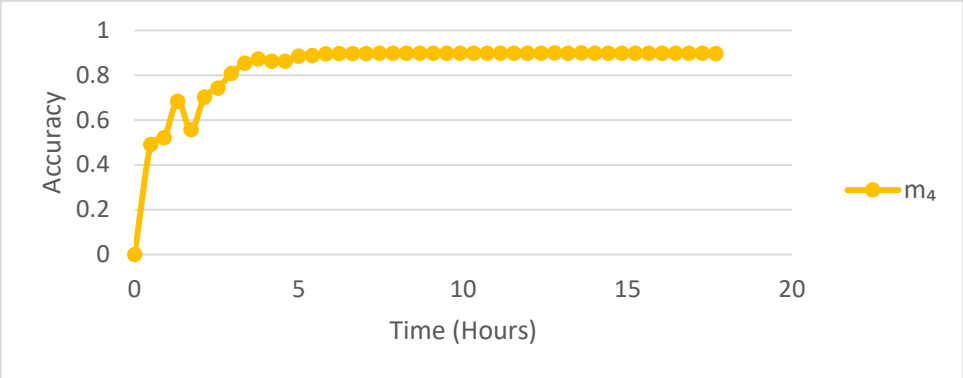


The training behavior of the fourth model (m_4) displays how the early stopping and learning decay rate helped influence its high accuracy. Figure 17 shows just the training timeline of the fourth model (m_4). The accuracy does not universally improve, showing how the model was beginning to come to an incorrect conclusion. The learning rate decay prevented this impacting

the learning too much and it was able to consider other conclusions that improved the accuracy. After 10 epochs of no improvements, the early stopping prevented the model from going through the 100 epochs, and instead end at 43 epochs. The model (m_4) does an excellent job at displaying the effectiveness of early stopping and learning rate decay.

Figure 17

Model (m_4) Accuracy Over Time, at Each Epoch



The fourth model (m_4) also performed better on the real-world data with the testing database, Table 12 lists the results showing that model (m_4) achieved an accuracy of 60%. Appendix A lists all the accuracy test results for the four trained models. Now that the accuracy has passed the 50% threshold, this model (i.e., model m_4) can listen to an individual speaking and have a better conclusion of the language being spoken by creating and classifying multiple voice clips. Although it is still 40 points from a perfect 100%, it has still achieved a high accuracy given that the testing data is purposely difficult to comprehend. This model (m_4) has a high enough accuracy to be deployed onto an application to correctly classify a spoken language.

Table 12

Accuracy and Time-Spent Recognizing a Spoken Language With Model (m_4)

Language	Accuracy	Compared to m_3	Average Time-Spent
English	68%	↑ (from 64%)	0.66s
French	60%	↑ (from 44%)	0.66s
Russian	40%	↑ (from 0%)	0.66s
Chinese (China)	80%	↓ (from 90%)	0.66s
Arabic	50%	↑ (from 20%)	0.66s
Persian	50%	= (from 50%)	0.66s
German	50%	↑ (from 30%)	0.66s
Overall	**60%**	↑ (from 46%)	**66s**

The changes in the time required to make a classification per voice clip is logical when the differences between the models are understood. The first model (m_1) set the base-line time of 44 seconds. As the second model (m_2) was simply given more training time, it was a larger model and took a slightly higher time of 45 seconds. The third model (m_3) filtered the training data further and therefore dealt with a lower number of voice clips, resulting in a smaller model and a slightly lower time of 43 seconds. The fourth model (m_4) has a seemingly abnormally high time of 66 seconds but can be attributed to having to create and store files outside of the interpreter. After a few calls as a warm-up, the time spent to classify a voice clip is very reasonable with all four (4) models.

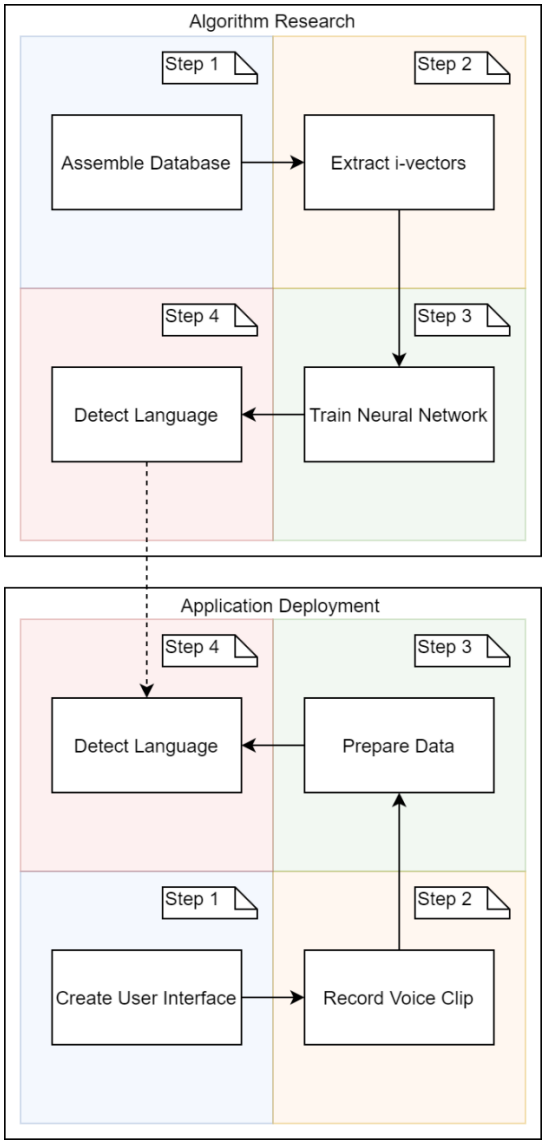
Chapter 4. Prototype

Architecture and Workflow

To utilize this research, a deployable application must be developed that can take advantage of the developed model, shown in Figure 18. This figure includes Figure 7 to show how the research completed so far rolls into the application deployment. The output of the research thus far is a model that, given voice clip data, can classify the spoken language. Before the model can be used however, other steps must be taken. First, the User Interface (UI) must be developed so that a user can interact with the model. Second, the user must be able to record a voice clip for the model. Third, the backend of the application must be able to take the voice clip and manipulate it so that it is an acceptable format for the model to ingest. Finally, the model can detect the language and present it to the user.

Figure 18

Model Research and Deployment



Before the application can be deployed, a platform to develop the application needs to be chosen. Android Studio is the main IDE that is used for creating and deploying android applications. This IDE is heavily supported with documentation but uses either Java or Kotlin languages. As the research thus far used Python and the IDE Visual Studio Code, it would be ideal if the platform could also be developed with the same IDE and language. While Google products such as TensorFlow fully utilize python, android application simply cannot natively be programmed in

Python. This problem of Python being unsupported for application development by Google has led others to create platforms and tools to bridge the gap.

Kivy[27] is an open-source platform that allows applications to be developed with Visual Studio Code in Python and packaged into an application. This packaging not only creates Android applications, but also iOS application. Applications developed with Kivy can also avoid having to be packaged and put onto an official store by using the existing Kivy Launcher application[28]. This allows Python scripts to be exported as a Kivy file and tested on a mobile device. This could be very beneficial if the application could not be put onto the application store for reasons such as application size or ethics. Applications can be tested even quicker however as they can be launched directly with a pop-up window. As most application development platforms require the heavy cost of running an emulator of a phone to test the applications, this is very beneficial for rapid development. Kivy appeared to be a viable solution for deploying an application using Python and Visual Studio Code.

Unfortunately, Kivy was found to not be a viable solution after reaching the third step of Figure 18, Prepare Data. A UI was rapidly developed, shown in Figure 19. Developing a UI in Kivy, although simple, lacks the robust UI manipulation of a dedicated application IDE. It however quickly creates a functioning UI, which is really all that is required to deploy this research. Creating the audio functionality was also quick and simple, making use of Kivy's own API for

---

[27] Kivy, "Kivy," *Kivy*. https://kivy.org/ (accessed Feb. 16, 2021).

[28] M. Virbel, "Kivy Launcher," *Google Play*. https://play.google.com/store/apps/details?id=org.kivy.pygame (accessed Feb. 16, 2021)

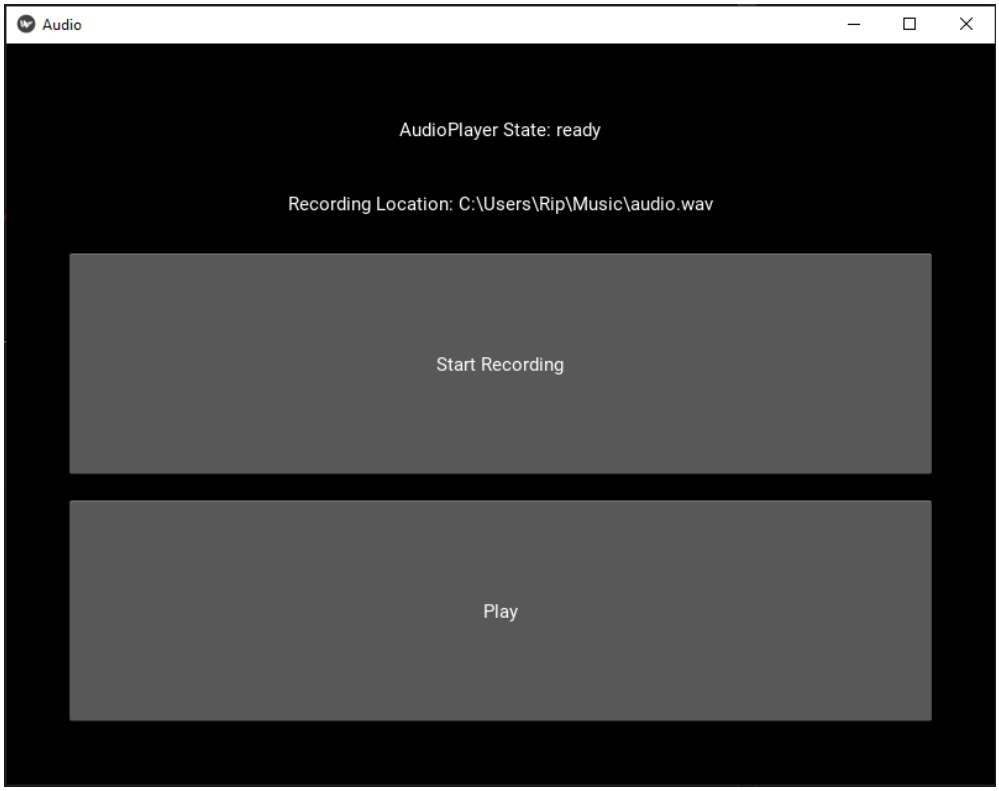
handling audio[29]. Since Kivy's API is used, this allows for the application to be deployed onto either Android or iOS, as it will deal with the unique system calls that handle audio. When starting to interact with the voice clip, Kivy's major weakness was brought to light. While Kivy can use certain libraries, they have to be specifically supported and very few are. TensorFlow is not one of the supported libraries. Due to this reason, development with the Kivy platform had to be abandoned for another solution to run Python scripts on Android.

---

[29] Kivy, "Audio," *Kivy*. https://kivy.org/doc/stable/api-kivy.core.audio.html (accessed Feb. 16, 2021)

Figure 19

Kivy Application to Record and Play Audio



A few other solutions for running Python scripts on Android were tested such as Beeware[30], QPython[31], and PyQt[32], but the ultimate solution was Chaquopy[33]. This tool allows the Android Studio IDE to make calls to a Python script, downloading the libraries it requires to function. While this unfortunately means that the Visual Studio IDE cannot be used, at least the current Python solution could be called with Java in an IDE specifically designed for developing and deploying Android applications. With Chaquopy being able to handle calls to a Python script and

---

[30] R. Keith-Magee, "BeeWare," *BeeWare*. https://beeware.org/ (Accessed Feb. 16, 2021)

[31] QPython, "QPython – Python on Android," *QPython*. https://www.qpython.com/ (Accessed Feb. 16, 2021)

[32] The QT Company, "What is PyQt?," *Riverbank Computing*. https://www.riverbankcomputing.com/software/pyqt/intro (Accessed Feb. 16, 2021)

[33] Chaquo, "Chaquopy," *Chaquo Ltd*. https://chaquo.com/chaquopy/ (Accessed Feb. 16, 2021)

its required libraries, the development for this application could be moved to the Android Studio IDE.

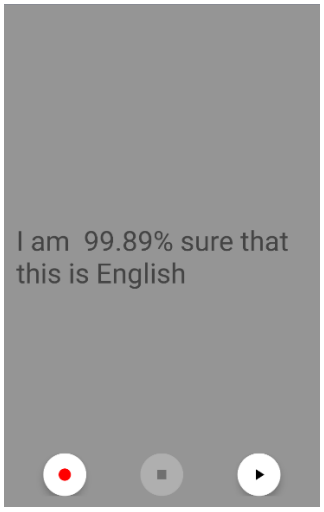
Application Design

Using Android Studio as an IDE and Chaquopy to handle data manipulation and calls to the model, a well-designed application was created to house this research. Although creating the UI in Android Studio took some time, it was able to look quite professional. Recording and storing voice clips were simple to accomplish, making use of Android API in Java. The voice clips were able to be manipulated in Python through Chaquopy. Finally, the classification was able to be made in Python through Chaquopy, creating an optimal deployment of this research.

The UI for this application was designed to be minimalistic and intuitive. Figure 20 shows the UI of the developed application. Icons were used in place of buttons as the images can invoke more of an understanding on what the buttons do than if they were buttons with text. The IDE was able to make these icons more aesthetic by taking care of spacing them out equally no matter the screen size, changing their colors, giving them a shadow effect, and most importantly, allowing them to fade. Depending on the state of the application, buttons are either enabled or disabled, and fading the buttons allows the users to understand what buttons can be pressed at any given state. The IDE does an excellent job at creating an aesthetically pleasing UI the promotes functionality.

Figure 20

User Interface of the Application Deployment



Recording and storing the voice clips at the push of the buttons was accomplished using MediaRecorder[34], a native Android API. When a recording is started, the MediaRecorder is invoked, specifying the format and location to save the voice clip. To do this however, permission must be asked to access the microphone. In primary implementations of the application, permission would also be asked for reading and writing to external storage, though this was avoided by later saving the voice clip to an internal directory of the application. The format specified is an m4a file, encoded with Advanced Audio Coding (AAC), the highest quality audio format available[35]. The file is then stored at a specific location so that it could be fed into the model for classification, as well as be played back if the user desired.

---

[34] Android Developers, "MediaRecorder," *Google Developers*. https://developer.android.com/reference/android/media/MediaRecorder (Accessed Feb. 18, 2021)

[35] Android Developers, "Supported media formats," *Google Developers*. https://developer.android.com/guide/topics/media/media-formats (Accessed Feb. 18, 2021)

Preparing the data for the model was completed in Python through Chaquopy, though this highlighted an issue with how Chaquopy installs the packages the Python scripts require. One of the libraries needed to prepare the data is librosa, which loads the voice clip so that it can be manipulated. This library however can only load in Waveform Audio File Format (WAVE) files. This was previously solved by using FFmpeg which was installed alongside librosa, however the installation of librosa that Chaquopy does skips the FFmpeg installation. As MediaRecorder is unable to output the WAVE files that librosa requires, this was a problem.

The first solution to load the voice clips for manipulated was to convert the python code into the native Android Java code. A Java package called jlibrosa[36] was used to make the same librosa calls in Java. While this was able to manipulate the voice clips, this unfortunately did not manipulate the data the exact same way librosa did. Comparing outputs from librosa and jlibrosa showed similar data, but they were not exact. As giving similar but not exact data to the model would significantly alter its accuracy, this was not a viable solution.

The second solution was to focus on how to convert the m4a file to a WAVE file. As a WAVE file is not an encoding, but a container, a WAVE file can be manually made. This is done by creating a new file, filling out the WAVE header information[37], followed by the voice clip raw data. Fortunately, a Java package called mobile-ffmpeg[38] exists, which can use FFmpeg on a mobile device to do just that. By parsing in a string to emulate how the call to FFmpeg would be,

---

[36] Subtitle-Synchronizer, "jlibrosa," *GitHub*. https://github.com/Subtitle-Synchronizer/jlibrosa (Accessed Feb. 18, 2021)

[37] FileFormat, "What is a WAV file?," *Aspose Pty Ltd.* https://docs.fileformat.com/audio/wav/ (Accessed Feb. 18, 2021)

[38] tanersener, "mobile-ffmpeg," *GitHub*. https://github.com/tanersener/mobile-ffmpeg (Accessed Feb. 18, 2021)

the saved voice clip could be converted to a WAVE file. This allowed librosa to load and prepare the data for the model.

With the data correctly prepared, a classification can be made and presented to the user, though after taking a long time. As observed earlier, the first classification that is made takes significantly longer than subsequent classifications. Further time was added by Chaquopy as well as it must invoke the python instance and along with some other set up. The time for the first classification on lesser hardware was taking upwards of two (2) minutes, which was far too long for the user. Two solutions were implemented to address this inexcusable wait.

The first solution to handling the timing delays was converting the TensorFlow model output from the previous research into a TensorFlow Lite model. This conversion results in size reduction, latency reduction, and an increased accelerator compatibility. The original 257MB TensorFlow model was able to be converted into an 85MB TensorFlow Lite model. Loading this lite model onto the android device significantly reduced the amount of time to load in the model and make calls to it. The cost of this optimized performance boost comes with accuracy. With the Inceptionv3 CNN, accuracy is reported to drop from 78% to 77.5%[39]. Although maintaining the accuracy of the model is extremely important in the deployment of it, the accuracy loss is small enough to be considered acceptable.

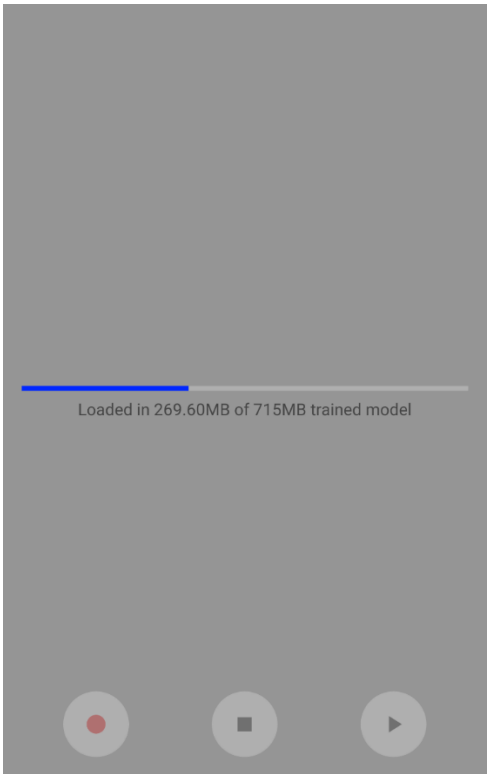
The second solution to handling the timing delays was to alter the UI to occupy the user while a classification is made immediately upon starting the application. The UI addition is shown in Figure 21. The loading bar steadily fills at a predetermined rate while the model is loaded in and

[39] TensorFlow, "Model optimization," *Google*. https://www.tensorflow.org/lite/performance/model_optimization (accessed Feb. 18, 2021).

a classification is made on a pre-loaded voice clip. The size of the model is said to be much larger than it really is so that the user can better appreciate the loading time. Once a classification is made, the loading bar is instantly filled and the icons become active so that the user can interact with the models with no delays.

Figure 21

User Interface to Account for Model Delay Time



With the application fully developed and ready to deploy to the Google Play Store, the issue of the size of the application was discovered. As Chaquopy installs all of the packages and their dependencies directly into the application storage, the application is quite large. The TensorFlow package alone is 80MB and the model requires even more storage. Although the model was able to be converted into a lite model, the storage limits of an Android Application Package (APK) to be on the Google Play Store is 100MB. This limit can be increased by instead using an Android App Bundle (AAB), which is a way for the Google Play Store to dynamically generate APKs

based on the device downloading the application[40]. The storage limit for an AAB is 150MB[41]. The current 228MB AAB can still be deployed onto mobile devices manually or through privately hosting it but deploying to the Google Play Store would allow the research to reach a much greater audience. The 228MB AAB can be analyzed to show that the AAB would be 149MB without the model, which would allow the deployment onto the Google Play Store.

The solution to the size constraint of the Google Play Store was to employ the Play Asset Delivery[42] function of the Google Play Store. This service is typically used for games, allowing assets to be patched in as required. The Play Asset Delivery has three (3) modes to deploy further assets, install-time, fast-follow, and on-demand[43]. Using install-time allows for the model to be packaged with the application as it is installed, providing the needed functionality. Unfortunately, this treats the model as an asset file, which means that it is compressed and not at a static location as needed.

To attain a static reference to the model, the model needs to be extracted from being an asset. When the application first starts, the AssetManager[44] is used to copy all assets to the internal application storage. This results in the application size growing on the device, as the model now

---

[40] Android Developers, "About Android App Bundles," *Google Developers*. https://developer.android.com/guide/app-bundle (Accessed Feb. 18, 2021)

[41] Android Developers, "About Android App Bundles," *Google Developers*. https://developer.android.com/topic/performance/reduce-apk-size (Accessed Feb. 18, 2021)

[42] Android Developers, "Play Asset Delivery," *Google Developers*. https://developer.android.com/guide/app-bundle/asset-delivery (Accessed Feb. 18, 2021)

[43] Android Developers, "Integrate asset delivery (Java)," *Google Developers*. https://developer.android.com/guide/playcore/asset-delivery/integrate-java (Accessed Feb. 18, 2021)

[44] Android Developers, "AssetManager," *Google Developers*. https://developer.android.com/reference/android/content/res/AssetManager (Accessed Feb. 18, 2021)

exists as an asset as well as fully uncompressed in internal storage. For this reason, the UI change to add the loading bar in Figure 21 also overstates the real size of the model. The application uses 722MB of internal storage due to also uncompressing the Chaquopy packages, so rather than the user wondering why it takes so much room, an easy, understandable, and ultimately false reason is given to them. The loading time for the loading bar is also doubled when it has to ask for microphone permissions, to account for the time to copy the model out of the AssetManager. This results in the fully functional application with a 149MB AAB that can be deployed onto the Google Play Store[45].

Evaluation Plan

To evaluate this research, the developed application must be used in conjunction with certain criteria to adequately measure the accuracy and usability. First (E-1), the test must be conducted in a real-world scenario, which was previously defined as an individual speaking into a device with the background noise of an environment. Second (E-2), many willing individuals must be found who can speak one or more of the seven languages considered in the application. Third (E-3), the evaluation must be quick and anonymous to avoid some of the biases that could be present. Fourth (E-4) and finally, the results of the evaluation must capture all necessary information and be easy to consolidate to create some meaningful findings on the accuracy and usability of the application.

Meeting the first criterium (E-1) requires that the test be conducted outdoors. By conducting the test outdoors, the noise will be much more random and uncontrollable, which is what is needed

[45] R. Pennell, "Automated Spoken Language Detector," *Google Play*. https://play.google.com/store/apps/details?id=rip.thesis.automatedspokenlanguagedetector (accessed Jan. 6, 2022)

to aptly simulate a real-world scenario. At any given point, a somewhat unique background noise will be present for each individual test, consisting of different noises coming from sources such as automobiles, birds, the wind, passerby conversations, and machinery on a building. Meeting with individuals outside is also much safer to conduct during the coronavirus pandemic. The pandemic also means all users will be wearing a mask, which will add yet another randomness factor to truly test the accuracy.

While beyond the scope of the evaluation criteria, other steps will be taken for the evaluation due to the coronavirus pandemic. As the evaluation will be conducted using a single phone, a thick protective screen will be used on top of the phone, so that hand sanitizer can be liberally and excessively. This will prevent damage to the device, while also relieving users of any sanitary dangers. This however may have an impact on usability, as the extra screen and hand sanitizer may have an impact on the users' ability to control the touch screen. When setting up the testing area, it will be done to remain six (6) feet away from any main walk paths, which could also have an impact on the background noise of passerby conversations. A face shield will also be worn, to protect all users who agree to participate.

Meeting the second criterium (E-2) requires that the test be conducted on a military base. This is the perfect location to find users to evaluate this research, based on three reasons. First, there are many members of the military who speak multiple languages. Second, military users are the targeted audience for this research. Third, it will be easy to track down users who speak multiple languages by using internal military databases of the base. Every member of a military base that speaks one of the needed secondary languages will be asked to partake in the evaluation of this research.

Meeting the third criterium (E-3) requires that biases be understood and accounted for. The first identified bias (B-1) is the professional relationship. As the tests will be conducted on a military base, participants who are a lower rank may be inclined to rate the application higher for fear of offending a military officer. This will partially mitigated by not wearing a uniform while preparing the tests, however the rank might still be known by being seen previously. A further mitigation will be done by giving the participants as much anonymity as possible. While one method to do this would be to have a book that the participant could flip through and put their information on a random page, the coronavirus pandemic restricts having items multiple participants must touch. Instead, a box with a slit, similar to an amnesty box, will be used for participants to drop papers into. The papers will be pre-filled with the required information and only ask for bullets to be checked, so that the unique handwriting of a participant does not identify them. The participants will also be told that the lock that is put on the amnesty box will not be opened until 100 participants have put their information in, so that they are not concerned with being the only person to speak a certain language that day.

The second identified bias (B-2) is the improvisation ability of the participant. While the participants will have self-identified as being able to speak a different language, sticking a device in front of them and telling them to say something random will certainly be off-putting to those without much improvisation experience. To mitigate this, some questions will be printed out on the table, designed such that the responses are likely unique to prevent classifications on the exact same phrase. They will also try to prevent the participant from using proper nouns, as they will be the same for all languages. These questions will allow the participant to instantly be able to create their own phrase in their tested language.

The third identified bias (B-3) is the urge of the participant to assist the algorithm. As the model was trained on a mix of native language and secondary language speakers, some resilience is already built into the model. Since the participants will understand that the application is trying to guess the correct language, they may put extra emphasis into the accent to attempt to help the algorithm classify correctly. Doing this prevents the evaluation data from being real-world data. To mitigate this, participants will be told to not force an accent and to speak as naturally as possible and that the model has been trained with many accents. Participants will also be asked to self-identify how much of an accent they have, to see how the model adapts. This will appease to the participants' natural urge to assist the algorithm.

The fourth identified bias (B-4) is the time availability of the participant. As almost all participants will be conducting this test during working hours, many participants will not be able to spend too much time away from their desk. The participant might take any opportunity they see to quickly end the test, so that they can continue with their workday. The mitigation for this is to design the test to take the minimal amount of time possible and ensuring all paths that can be taken during the test are short. Since the same phone is being used, no time is wasted having to download the very large application and loading in the very large model. The participant will instantly have the app primed for their use and will only get one classification per language they can speak. They can of course experiment with the application after if they wish, and relevant findings from this unstructured evaluation will be recorded with their permission. The evaluation will take no more than a minute, and the pre-filled questionnaire with bullets to fill out will also take no longer than a minute. Unfortunately, this means that participants will not be writing down the phrase that they used for the classification. However, this would not be possible

regardless in order to preserve anonymity. This allows the test to be very quick with no apparent shortcuts that the participant might gravitate to.

Meeting the fourth criterium (E-4) requires that the pre-filled questionnaire be designed to account for meaningful findings for the research as well as be easy to amalgamate. A unique identifier will be given to each data point at the point of amalgamation, as prior to the collection it is not needed. Content that must be in the questionnaire revolve around the research question and the two hypotheses of the research. To capture the accuracy of the research, the language spoken, the degree of a self-identified accent, the language identified, and the level of confidence will be recorded. This will allow a comparison to be done between tests with real-world data, and the test data that was used to track the different models' performances. To capture the usability of the research, a System Usability Scale (SUS) will be used. The SUS is a technology independent method to measure usability that has become an industry standard since its release in 1986 [54]. It consists of a 10-item questionnaire with five (5) response options, ranging from strongly disagree to strongly agree. There is then a science to calculating a final score and how that score is to be interpreted. The results of the tests in combination with the SUS will allow for an evaluation to aptly amalgamate the accuracy and usability of the research.

Accounting for the four (4) criteria, a step-by-step evaluation plan can be created.

1. Set up the testing environment.
    a. Stand up a folding desk, outside of the only entrance to the base compound. Ensure it is six (6) feet away from all main pathways and entrances.
    b. Place on the desk 10 black and 10 blue disposable pens in case a participant does not have a pen in their uniform.

c. Place on the desk 125 copies of a pre-filled questionnaire, which is shown in Appendix B.
d. Place on desk 10 questions that participants can opt to use.
e. Prime the application for execution on a phone with a thick layered protective screen.
f. Place on desk a large hand sanitizer dispenser.
g. Place on desk a box of wipes for use on the phone in between participants.
h. Place a Bristol board sign against table with the question, “Speak a second language?”

2. Gather participants.
    a. Ask passersby if they speak one of the languages which data is still required for.
    b. Meet with a linguist for some of the needed languages and get a contact list of potential participants.
    c. Meet with the base admin staff to check who has self identified as speaking on of the needed languages.
    d. Continue to search for participants until 100 tests are completed, with the appropriate ratios of data to achieve a similar dataset to the testing database.
3. Running the evaluation.
    a. Briefly explain to the participant the military intent of the research.
    b. Wipe the phone with sanitary products.
    c. Hand the phone to the participant, without any instructions other than use the application.

d. Give the participant the questionnaire and ask them to fill it out and place in the amnesty box. Move away from the table so that the questionnaire cannot be seen.
e. Respond to any questions they might have from this point.

This evaluation plan was submitted to and approved by the Athabasca University Research Ethics Board, with the approval shown in Appendix C. Upon receipt of the ethics approval, the evaluation plan was enacted exactly as outlined above.

Chapter 5. Results

Accuracy and System Usability

Real-world evaluation of the research, conducted as per the evaluation plan, generated highly accurate results, which are shown in Appendix D. The accuracy is even higher than the accuracy of the testing database, with the evaluation accuracy being 81% compared to the testing accuracy being 60%. This high accuracy of 81% approaches the accuracy the model achieved during training, 90%. As 14% accuracy would be accuracy of a completely random classifier for seven (7) languages, an accuracy of 81% concretely proves the technical hypothesis (H_T) of this research.

The precision, recall, and F-score of each language is shown in Table 13. The precision value helps answer the question of how many of those identified as a certain language were actually correct? In the case of Persian, which achieved a perfect precision score of 1, anytime the application deemed someone was speaking Persian, it was correct. Chinese (China), which had the lowest precision score of 0.64, was the most mistaken classification of the algorithm. It makes logical sense that the language with the highest accuracy also has the lowest precision, as it means that the algorithm is most likely to identify a language as Chinese (China) if it is going to be incorrect.

Table 13

Precision, Recall, and F-Score of Each Language

Language	Precision	Recall	F-score	Support
English	0.81	0.84	0.82	25
French	0.88	0.84	0.86	25
Arabic	0.89	0.8	0.84	10
Russian	0.75	0.6	0.67	10
Persian	1	0.8	0.89	10
Chinese (China)	0.64	0.9	0.75	10
German	0.73	0.8	0.76	10
Total Accuracy	0.81			100
Macro avg	0.81	0.8	0.8	100
Weights avg	0.82	0.81	0.81	100

The recall value helps answer the question of how many times was a language correctly identified? Chinese (China) had the highest recall score of 0.9, meaning it is very likely that if someone is speaking Chinese (China), it will be correctly identified. Russian had the lowest recall score of 0.6, which is still well above the recall of a random selector, 0.14. These two languages were also the most and least accurate in the testing database.

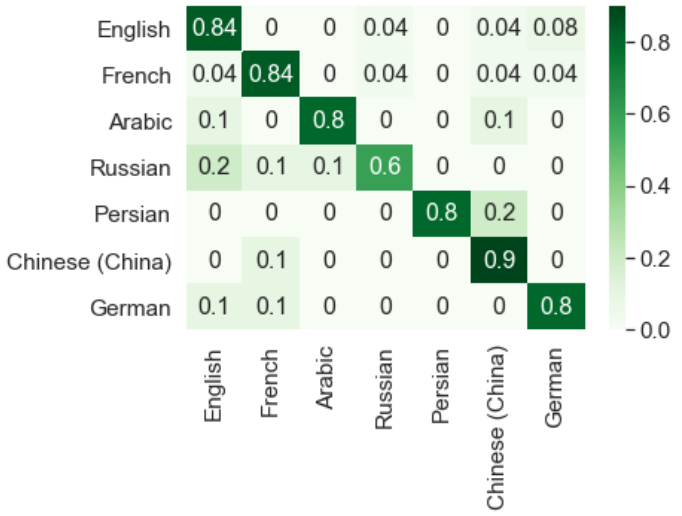
The F-score is the average between precision and recall, helping to answer the question on which language is the application performing the best with? Although Persian did not achieve the highest recall score, its perfect precision score causes it to have the highest F-score of 0.89. Russian has the lowest F-score of 0.67. Although Chinese (China) had the highest recall score, it is the second last in F-score due to its low score in precision. If only the recall or accuracy were considered, it would be deemed that the algorithm handles Chinese (China) the best, but in actuality, by looking at the F-score, it is one of the worst.

The confusion matrix of the evaluation results is shown in Figure 22. This matrix shows the two most common confusions, both with a score of 0.2 which is at least double of every other confusion. Russian speakers were mistakenly classified as English, and Persian speakers were mistakenly classified as Chinese (China). As these values are just over the accuracy of random,

0.14, there appears to be slight similarities between the languages. The rest of the confusions were less than 0.14, indicating the mistakes were a random guess.

Figure 22

Precision, Recall, and F-Score of Each Language



Real-world evaluation of the research, conducted as per the evaluation plan, generated high user ratings, which are shown in Appendix D. The overall SUS score was 95.7. This is well above the average SUS score of 68 and is close to a score of 100 which represents the best imaginable design [55]. While a bias might be assumed, biases were already identified and mitigated as part of the evaluation plan. The high score is likely due to a combination of the algorithm having a high accuracy and the application being very simple and intuitive. As the SUS score of 95.7 is much higher than the average SUS score of 68, this concretely proves the usability hypothesis (H_U) of this research.

The SUS score of the application by language is shown in Table 14. Both Persian and Chinese (China) speakers rated the application the highest, with Arabic speakers rating it the lowest, though still very high. Arabic speakers also shared the highest range (90-100) of scores, despite having the lowest average score. Comparing this table with Table 13, the accuracy of each

language, it is clear that the correlation between high accuracy and a high SUS score is not absolute, as Russian had the lowest accuracy, but Arabic had the lowest SUS score.

Table 14

SUS Scores Description for Each Language

Language	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean		Min	Max
					Lower Bound	Upper Bound		
English	25	96	5.352	1.070	93.79	98.21	80	100
French	25	96	5.774	1.155	93.62	98.38	80	100
Arabic	10	93.5	3.545	1.121	94.21	99.29	90	100
Russian	10	95.75	4.888	1.546	90.00	97.00	88	100
Persian	10	97	5.898	1.865	91.53	99.97	80	100
Chinese (China)	10	97	3.073	0.972	94.80	99.20	90	100
German	10	94	5.028	1.590	90.40	97.60	88	100
All	**100**	95.7	5.076	0.508	94.69	96.71	80	100

The SUS score and accuracy not having an absolute correlation is another indicator that the application is highly successful. Despite being given a wrong classification, users still found that the usability of the application was very high, which means the application is meaningful and useful in general. Another interesting phenomenon was that the SUS question on inconsistency sometimes had a perfect score despite showing the wrong classification. This is certainly because of how users understand the current maturity of current translation capabilities, allowing them to have the patience with their incorrect results as long as the application performs the way it does.

The SUS score of the application by degree of accent is shown in Table 15. Users with no accent rated the application the highest, with users with strong regional accents rating it the lowest, though still very high. There were very few users which identified as speaking with a strong regional accent, with the majority either having no accent or a strong accent. Users with a strong regional accent had the most diverse ratings, though this is likely due to the very small sample size of four (4).

Table 15

SUS Scores Description for Each Degree of Accent

Degree of Accent	N	Mean	SD	Std. Error	95% Confidence Interval for Mean		Min	Max
					Lower Bound	Upper Bound		
I have no accent	41	96.28	5.066	0.791	94.68	97.88	80	100
I have a bit of an accent	18	94.86	5.718	1.348	92.02	97.70	80	100
I have a strong accent	37	95.74	4.709	0.774	94.17	97.31	80	100
I have a strong regional accent	4	93.13	6.250	3.125	83.18	103.07	85	100
All	**100**	95.7	5.076	0.508	94.69	96.71	80	100

The SUS score of the application by correctness is shown in Table 16. Users given a correct result rated the application higher than those shown an incorrect result. The bounds for a 95% confidence interval for the mean do not overlap between the two groups, with a gap existing between 92.87 and 96.41. The minimum SUS score of a user shown a correct result, 88, is almost the same as the average SUS score of a user shown an incorrect result, 89.61. This suggests that there is an impact on the user's usability depending on whether a correct result is shown.

Table 16

SUS Scores Description for Whether Result was Correct

Correct	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean		Minimum	Maximum
					Lower Bound	Upper Bound		
No	19	89.61	6.784	1.556	86.34	92.87	80	100
Yes	81	97.13	3.262	0.362	96.41	97.85	88	100
All	**100**	95.7	5.076	0.508	94.69	96.71	80	100

The SUS scores for each SUS question is shown in Table 17. Question seven (7) and eight (8) are tied for being answered the best. These two questions scored well certainly due to the simplicity and intuitiveness of the application. Question one (1) was answered the worst, though still very well. This question focuses on how applicable this application would be to their lives, which is perhaps the most archaic question of the 10 as it touches on the daily lives and needs of the users.

Table 17

SUS Score of Each Question

SUS Question	Average Score	Difference From Perfect Score
1. I think that I would like to use this system frequently.	4.54	0.46
2. I found the system unnecessarily complex.	1.18	0.18
3. I thought the system was easy to use.	4.95	0.05
4. I think that I would need the support of a technical person to be able to use this system.	1.06	0.06
5. I found the various functions in this system were well integrated.	4.93	0.07
6. I thought there was too much inconsistency in this system.	1.35	0.35
7. I would imagine that most people would learn to use this system very quickly.	4.96	0.04
8. I found the system very cumbersome to use.	1.04	0.04
9. I felt very confident using the system.	4.59	0.41
10. I needed to learn a lot of things before I could get going with this system.	1.06	0.06
All	N/A	0.17

Normality Tests and Transformations of Collected Data

To conduct statistical analysis on the SUS scores of each language, the data first needs to be tested for normality, as the statistical analysis assumes the data is normally distributed[46]. To conduct a normality test, the SUS score of each language was analyzed to determine the skewness and kurtosis of each dataset. Skewness is a measurement that helps define how asymmetric the distribution is while kurtosis measures the peakedness of a distribution. A z-score can be generated for both by dividing the value by its standard error. If the absolute z-score for either of these are above the absolute value of 1.96, then the normality test has failed and statistical analysis cannot be conducted on the dataset [56].

The results of the normality test are in Table 18. Over half of the languages failed the normality test. Every language has a negative skewness value, which indicates a negative skew, meaning

[46] UCLA, "WHAT IS THE DIFFERENCE BETWEEN CATEGORICAL, ORDINAL AND INTERVAL VARIABLES?," *Statistical Consulting Group*. https://stats.idre.ucla.edu/other/mult-pkg/whatstat/what-is-the-difference-between-categorical-ordinal-and-interval-variables (Accessed Aug. 18, 2021)

the tail on the left is larger than on the right. This is unsurprising as many perfect scores were given and it is not possible to give an above perfect score, or a score above the "ceiling". Statistical analysis can still be done on this dataset despite failing the normality test as the data can fortunately be transformed into normality.

Table 18

Normality Test of Raw SUS Scores for Each Language

Language	Skewness z-value	Below |1.96|?	Kurtosis z-value	Below |1.96|?
English	-3.98	No	3.67	No
French	-3.77	No	2.50	No
Arabic	-1.17	Yes	-0.28	Yes
Russian	-0.38	No	-1.36	Yes
Persian	-3.61	No	5.20	No
Chinese (China)	-1.93	Yes	1.66	Yes
German	-0.49	Yes	-1.14	Yes
All	-6.21	No	3.71	No

Before transforming the data, more normality tests should be done first so that all of the data can be transformed at once. Table 19 shows the remaining normality tests on other datasets that should be analyzed for usability. The impact of a person's accent on SUS scores and the impact of a person being presented with a correct classification on SUS scores are important to understand. These datasets also failed the normality tests and require transformation.

Table 19

Normality Test of Raw SUS Scores for Each Language

Accent	Skewness z-value	Below |1.96|?	Kurtosis z-value	Below |1.96|?
I have no accent	-4.49	No	3.25	No
I have a bit of an accent	-3.32	No	2.37	No
I have a strong accent	-3.69	No	2.89	No
I have a strong regional accent	-0.55	Yes	0.35	Yes
Correctness	**Skewness z-value**	**Below |1.96|?**	**Kurtosis z-value**	**Below |1.96|?**
Incorrect	0.03	Yes	-1.21	Yes
Correct	-4.90	No	2.63	No

There are multiple methods to transform data into normality[47]. The selected method is typically used for positive skewness but altered to work with negative skewness since all normality test failures in the dataset are negatively skewed. A base 10 logarithm of each value works for a positive skewness. Taking the max value in the data set, adding one (1) so that no final value is ever zero (0), and subtracting each variable and doing a base 10 logarithm on the result works for a negative skewness. In this case, the max SUS score which was achieved multiple times is 100. 101 subtracted by the variable with a base 10 logarithm of the result provides a transformed dataset that should test as normally distributed.

Normality tests on all useability datasets after dataset transformation is shown in Table 20. For almost all datasets, the transformation was a success. The only dataset that was not normalized was SUS scores of users shown an incorrect classification, which did have the lowest skewness z-score of all datasets before the transformation. Its unique behaviour is indicative that being shown an incorrect classification has a large impact on the SUS score. With the datasets all

[47] IBM SPSS Statistics, "Transforming Variable to Normality for Parametric Statistics," *IBM*. https://www.ibm.com/support/pages/transforming-variable-normality-parametric-statistics (Accessed Aug. 18, 2021)

transformed, applicable data analysis can be done to better understand the impact each independent factor has on the SUS score.

Table 20

Normality Test of Transformed SUS Scores for Each Language

Language	**Skewness z-value**	**Below |1.96|?**	**Kurtosis z-value**	**Below |1.96|?**
English	0.54	Yes	-1.37	Yes
French	0.97	Yes	-1.25	Yes
Arabic	-0.08	Yes	-1.42	Yes
Russian	-1.24	Yes	0.42	Yes
Persian	0.26	Yes	-0.15	Yes
Chinese (China)	-0.38	Yes	-0.74	Yes
German	-1.09	Yes	-0.42	Yes
Accent	**Skewness z-value**	**Below |1.96|?**	**Kurtosis z-value**	**Below |1.96|?**
I have no accent	0.97	Yes	-1.76	Yes
I have a bit of an accent	-0.12	Yes	0.04	Yes
I have a strong accent	-0.24	Yes	-1.69	Yes
I have a strong regional accent	-1.29	Yes	0.82	Yes
Correctness	**Skewness z-value**	**Below |1.96|?**	**Kurtosis z-value**	**Below |1.96|?**
Incorrect	-2.51	No	1.08	Yes
Correct	0.24	Yes	-2.63	No
All	0.18	Yes	-2.59	No

Data Analysis

A one-way analysis of variance (ANOVA) test[48] was conducted on the language dataset to determine the impact a spoken language had on the SUS score. This test requires the independent variable to be categorical, with an interval dependent variable. The type of language is categorical and the SUS scores is interval.

Results of the one-way ANOVA test on languages and transformed SUS score is shown in Table 21. The Games-Howell post hoc test was used as the sample sizes between English and French are different from the rest. The one-way ANOVA revealed that there was not a statistically

significant difference in the transformed SUS scores between at least two groups (F(6, 93) = [0.70] , p = 0.653). The Games-Howell post hoc test for multiple comparisons found that there was no statistically significant difference in transformed SUS scores between any of the languages.

---

[48] UCLA, "One-way ANOVA," *Statistical Consulting Group*. https://stats.idre.ucla.edu/spss/whatstat/what-statistical-analysis-should-i-usestatistical-analyses-using-spss/#1anova (Accessed Aug. 18, 2021)

Table 21

One-Way ANOVA Test on Languages and Transformed SUS Scores

(I) Language	(J) Language	Mean Difference (I-J)	Standard Error	Significance	95% Confidence Interval	
					Lower Bound	Upper Bound
Arabic	Chinese	-0.0159	0.17707	1.000	-0.6025	0.5707
	English	-0.0192	0.16152	1.000	-0.5545	0.5161
	French	0.0070	0.16324	1.000	-0.5320	0.5460
	German	-0.2350	0.19062	0.872	-0.8649	0.3949
	Persian	-0.0674	0.19082	1.000	-0.6979	0.5632
	Russian	-0.3006	0.17733	0.628	-0.8880	0.2868
Chinese (China)	Arabic	0.0159	0.17707	1.000	-0.5707	0.6025
	English	-0.0033	0.14634	1.000	-0.4804	0.4738
	French	0.0229	0.14823	1.000	-0.4588	0.5045
	German	-0.2191	0.17793	0.873	-0.8087	0.3705
	Persian	-0.0515	0.17814	1.000	-0.6419	0.5389
	Russian	-0.2847	0.16361	0.600	-0.8253	0.2559
English	Arabic	0.0192	0.16152	1.000	-0.5161	0.5545
	Chinese	0.0033	0.14634	1.000	-0.4738	0.4804
	French	0.0262	0.12926	1.000	-0.3715	0.4239
	German	-0.2158	0.16247	0.830	-0.7547	0.3232
	Persian	-0.0482	0.16270	1.000	-0.5880	0.4917
	Russian	-0.2814	0.14665	0.491	-0.7597	0.1969
French	Arabic	-0.0070	0.16324	1.000	-0.5460	0.5320
	Chinese	-0.0229	0.14823	1.000	-0.5045	0.4588
	English	-0.0262	0.12926	1.000	-0.4239	0.3715
	German	-0.2420	0.16418	0.756	-0.7845	0.3006
	Persian	-0.0744	0.16441	0.999	-0.6178	0.4691
	Russian	-0.3076	0.14854	0.404	-0.7904	0.1752
German	Arabic	0.2350	0.19062	0.872	-0.3949	0.8649
	Chinese	0.2191	0.17793	0.873	-0.3705	0.8087
	English	0.2158	0.16247	0.830	-0.3232	0.7547
	French	0.2420	0.16418	0.756	-0.3006	0.7845
	Persian	0.1676	0.19162	0.972	-0.4656	0.8008
	Russian	-0.0656	0.17819	1.000	-0.6560	0.5248
Persian	Arabic	0.0674	0.19082	1.000	-0.5632	0.6979
	Chinese	0.0515	0.17814	1.000	-0.5389	0.6419
	English	0.0482	0.16270	1.000	-0.4917	0.5880
	French	0.0744	0.16441	0.999	-0.4691	0.6178
	German	-0.1676	0.19162	0.972	-0.8008	0.4656
	Russian	-0.2332	0.17840	0.840	-0.8244	0.3579
Russian	Arabic	0.3006	0.17733	0.628	-0.2868	0.8880
	Chinese	0.2847	0.16361	0.600	-0.2559	0.8253
	English	0.2814	0.14665	0.491	-0.1969	0.7597
	French	0.3076	0.14854	0.404	-0.1752	0.7904
	German	0.0656	0.17819	1.000	-0.5248	0.6560
	Persian	0.2332	0.17840	0.840	-0.3579	0.8244

A one-way ANOVA test[49] was conducted on the accent dataset to determine the impact an accent had on the SUS score. This test requires the independent variable to be categorical, with an interval dependent variable. The type of accent is slightly ordinal, but should be considered categorical due to the non-linearity of each rating and the SUS scores is interval.

A one-way ANOVA was performed to compare the effect of accents on transformed SUS scores with the results shown in Table 22. The Games-Howell post hoc test was used as the sample sizes between different accents were not controlled and vary widely. The one-way ANOVA revealed that there was not a statistically significant difference in the transformed SUS scores between at least two groups ($F(3, 96) = [1.126]$, $p = 0.343$). The Games-Howell post hoc test for multiple comparisons found that there was no statistically significant difference in transformed SUS scores between any of the accents.

---

[49] UCLA, "One-way ANOVA," *Statistical Consulting Group*. https://stats.idre.ucla.edu/spss/whatstat/what-statistical-analysis-should-i-usestatistical-analyses-using-spss/#1anova (Accessed Aug. 18, 2021)

Table 22

One-Way ANOVA Test on Accents and Transformed SUS Scores

Accent	Accent Comparison	Mean Difference (I-J)	Standard Error	Sig.	95% Confidence Interval	
					Lower Bound	Upper Bound
I have no accent	I have a bit of an accent	-0.18594	0.11412	0.375	-0.4926	0.1207
	I have a strong accent	-0.09015	0.09891	0.799	-0.3500	0.1697
	I have a strong regional accent	-0.27947	0.26747	0.738	-1.4584	0.8994
I have a bit of an accent	I have no accent	0.18594	0.11412	0.375	-0.1207	0.4926
	I have a strong accent	0.09579	0.11409	0.835	-0.2110	0.4026
	I have a strong regional accent	-0.09354	0.27345	0.984	-1.2418	1.0547
I have a strong accent	I have no accent	0.09015	0.09891	0.799	-0.1697	0.3500
	I have a bit of an accent	-0.09579	0.11409	0.835	-0.4026	0.2110
	I have a strong regional accent	-0.18933	0.26746	0.889	-1.3683	0.9897
I have a strong regional accent	I have no accent	0.27947	0.26747	0.738	-0.8994	1.4584
	I have a bit of an accent	0.09354	0.27345	0.984	-1.0547	1.2418
	I have a strong accent	0.18933	0.26746	0.889	-0.9897	1.3683

A two independent samples t-test[50] was conducted on the correctness dataset to determine the impact a correct classification had on the SUS score. This test requires the independent variable to be two independent groups, with an interval dependent variable. The classification being correct or incorrect are two independent groups and the SUS scores is interval.

Results of the two independent samples t-test on correctness and transformed SUS score is shown in Table 23. The Levene's test[51] has a p-value of 0.509 which is above 0.05 and therefore the null hypothesis of the variances being equal is accepted and assumed. There was a significant difference in transformed SUS scores for showing the incorrect (M=0.931, SD=0.093) and

[50] UCLA, "Two independent samples t-test," *Statistical Consulting Group*. https://stats.idre.ucla.edu/spss/whatstat/what-statistical-analysis-should-i-usestatistical-analyses-using-spss/#2ittest (Accessed Aug. 18, 2021)

[51] NIST, "Levene Test for Equality of Variances," *Engineering Statistics Handbook.* https://www.itl.nist.gov/div898/handbook/eda/section3/eda35a.htm (Accessed Aug. 30, 2021)

correct (M= 0.432, SD=0.042) language; t(98)=5.068, p=0.000. This is consistent with the concluded observations of the raw data concerning SUS scores and correctness in Table 15.

Table 23

Two Independent Samples t-test of Correctness and Transformed SUS Scores

Levene's Test for Equality of Variances			t-test for Equality of Means						
Equality of Variances	F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval	
								Lower	Upper
Assumed	0.440	0.509	5.068	98	0.000	0.499	0.095	0.304	0.694
Not Assumed			4.866	25.928	0.000	0.499	0.103	0.288	0.710

Findings and Discussion

Overall, the application and the research produced outstanding results. A total accuracy of 81% is significantly higher than 14%, the accuracy of a random classification of seven (7) languages. This concretely proves the technical hypothesis (H_T) of this research. A total SUS score of 95.7 is significantly higher than 68, the average SUS score according to industry standards [55]. This concretely proves the usability hypothesis (H_U) of this research. Both hypotheses of this research are proven and therefore, the answer is yes to the research question, "Can machine learning algorithms be used to increase the effectiveness of spoken language detection?"

From the evaluation, three beneficial theories are apparent for this research. First, there appears to be no dependency on which language is being spoken. This demonstrates this research can be expanded to handle many languages. Second, there appears to be little dependency on the accent of a speaker. This demonstrates that the algorithm has a deep understanding of what makes a certain language that language. Third, the concept of this research is meaningful and useful to the military and the greater public. This demonstrates that this research has the potential to be used with current technologies to improve both military operations and the public's daily lives.

In terms of both accuracy and usability, all languages that this research considered were successful. Although Russian had the lowest recall of 0.6, this is still well above the recall of a random selector, 0.14. It is not too far from the average accuracy, which is 81%. It's precision and F-score are also quite high, at 0.75 and 0.67 respectively. Conversely, Arabic had the lowest SUS score of 93.5, which is much higher than the average SUS score of 68 according to industry standards [55]. This is also not too far from the average SUS score of all languages, 95.7.

All languages being successful proves the potential of this research. The research was designed in such a way that the specific languages themselves were not important. The database used contains voice clips from both high-resourced and low-resourced languages, meaning that other languages can be trained for the application and likely have a high accuracy and high usability. The cost to incorporate more languages is simply the training time. There is the possibility that the average accuracy of all languages will decrease, as the amount of false positives will increase. It is however almost certain that the accuracy of the individual languages will always be well above the accuracy of a random selection, which will also decrease as more languages are added.

The accent of the speaker did not appear to have an impact on accuracy and usability. The evaluation consisted of 41 users who reported having no accent and 37 users who reported having a strong accent for the spoken language. The average accuracy of the users with no accent was 80.49%, while the average accuracy of the users with a strong accent was 83.78%. Conversely, the average SUS score of users with no accent was 96.28, while the average SUS score of the users with a strong accent was 95.74. Both users who self-reported as having no accent and having a strong accent had excellent accuracy and usability with the application.

The performance of the application's accuracy with strong accents proves a great potential for this research. The application actually performed better with users with a strong accent (83.7%)

than users with no accent (80.49%). This is impressive given that the database used was mostly native speakers, that is speakers with no accent. This indicates that the algorithm used to determine the language has a deep understanding of what makes a language a language, and not just relying on a user sounding like someone from the training database. An unsupervised machine learning algorithm would have likely relied on similar sounds that native speakers commonly say, which is why the custom i-vector algorithm was so important to the success of this research.

The relationship between the SUS scores and the accuracy is complex. The two independent samples t-test on correctness and transformed SUS score indicated that there was a significant difference in the way users rated the application depending on whether they were shown the correct language or not. While this alone would demonstrate that the relationship is correlational and likely causational, there are instances where this is shown to be false. The users with strong accents had a higher overall accuracy (83.7%), but a lower overall SUS score (95.74) than users with no accent (80.49% and 96.28) without significant difference (refer to Table 22). Looking at individual tests, users with an accuracy of zero (0), since they only used the application once, still rated the application exceptionally high. It is logical, and was observed, that users shown the correct language would rate the application higher, yet observing the data specifically and comparing certain datasets, this was not always the case.

The complex relationship between the SUS scores and the accuracy proves the potential usefulness of this application. As stated above, users that identified themselves as being able to speak a language without an accent, despite having a lower accuracy, rated the application higher. This is almost certainly due to these users understanding how useful this research can be. Many of these users with no accents were required for their military profession to know these

languages and so even if they were shown the incorrect language, they understood the application's usefulness to be very high. This indicates that there is great potential for this application to be incorporated with other language applications to produce compelling solutions and break-down current language barriers.

Chapter 6. Conclusion

Summary

This research provided a solution to one of the greatest barriers in communication by automatically detecting the language being spoken. The military will greatly benefit from this research as working in regions with a multitude of languages is a continuous problem. Current solutions to verbal translation require the manual selection of languages being spoken, which is not always a possibility. This research is important to the military and any individual who will be in areas where they cannot speak every local language.

The purpose of this research, to solve a gap in current verbal translation through the automatic detection of a spoken language, was accomplished. A positive answer was given to the research question: “Can machine learning algorithms be used to increase the effectiveness of spoken language detection?” Proof was validated for the technical hypothesis: “A machine learning algorithm can classify a language being spoken in real-world scenarios.” Proof was also validated for the usability hypothesis: “The perceived usability toward the application with the proposed machine learning algorithm built-in is high.” By proving the two hypotheses, the research question could be answered, accomplishing the purpose of the research.

Existing relevant SLRs were examined to justify the need to conduct one specifically for this research. Of the two that were found, neither were able to identify acceptable methods or algorithms to use in this research. The first SLR was a well outlined SLR focused on sign language recognition but was missing some key information on how it was conducted. The second SLR was a poorly outlined SLR focused on ASR that did not have a clear application use. Both SLRs however demonstrated that there is an inherent assumption when dealing with

different languages that the language will be known or manually selected, justifying the need of this research. The SLRs also exhibited the need for SLRs to be highly detailed so that readers can understand why and how it was conducted. The existing SLRs aided in understanding what is needed to create an effective SLR and why one needed to be conducted.

The SLR completed for this research consisted of four highly detailed steps. The first step formulated the review question, "What machine learning algorithms have been used to successfully identify specific spoken languages?" The second step defined the three exclusion criterium of the research's publication year (i.e., its age, older than five years), non-focus on spoken languages, and classification accuracy (less than 80%). The third step was a highly detailed search strategy and location of studies, using two methods to intelligently identify the best literature sources for this research, defining their filter criterium of not being a journal, being unrelated to the research, and a duplicate source, and identifying the keywords to search for. The fourth step was to select the studies using the defined parameters outlined in the previous steps. The SLR considered 57 publication sources and narrowed these sources down to 19. From these sources, 9,662 papers were considered and narrowed down to nine (9). The SLR did an excellent job of systematically finding relevant works for analysis to answer the review question.

Quantitative analysis conducted with the papers outputted from the SLR identified the i-vector algorithm to be the best algorithm for this research. It was used the most and is clearly recognized as a benchmark for the application of detecting a spoken language. The accuracy of the algorithm is also high, sometimes even higher than the algorithm it was being a benchmark for. The customization available for the algorithm was also an important factor. Of the 11

algorithms identified in the SLR, the i-vector algorithm was the best algorithm for this research based on the quantitative analysis.

Qualitative analysis conducted with the papers outputted from the SLR continued to identify the i-vector algorithm is the best algorithm for this research. The algorithm was shown to have an acceptable level of accuracy when compared to the others. It however was shown to be extremely well established in the field. Additionally, it was able to handle low resource languages extremely well. While the analysis had its score tie with the GMM algorithm, it outperformed GMM in a paper that used the two with the same dataset. Of the 11 algorithms identified in the SLR, the i-vector algorithm was confirmed to be the best algorithm for this research based on the qualitative analysis.

The method of using an i-vector algorithm was implemented successfully to detect a spoken language, broken down into four (4) steps. First a database of spoken languages was formed using Mozilla Common Voice[52], down-sampled and manipulated by FFmpeg[53] and librosa [46]. Second, an i-vector extraction method was created using a rewritten implementation of Kaldi [42]. Third, a neural network, specifically CNN models[54], were trained using TensorFlow[55]. Fourth and finally, the finished model allowed for new voice recordings to have their language classified with excellent accuracy.

---

[52] Mozilla, "Common Voice." https://commonvoice.mozilla.org/ (accessed Dec. 18, 2020).

[53] FFmpeg, "FFmpeg," *telepoint*. https://ffmpeg.org/ (accessed Dec. 22, 2020).

[54] TensorFlow, "tf.keras.layers.Conv2D | TensorFlow Core v2.4.0," *Google*. https://www.tensorflow.org/api_docs/python/tf/keras/layers/Conv2D (accessed Jan. 13, 2021).

[55] TensorFlow, "TensorFlow Core | Machine Learning for Beginners and Experts," Google. https://www.tensorflow.org/overview (accessed Dec. 18, 2020).

The algorithm, the second step of the classification method, was successfully implemented in a Python environment as a compatible implementation of Kaldi [42]. This had to be done as TensorFlow is only available in Python and porting extracted i-vectors was not an available option [43]. Many of the open source Kaldi files that extracted i-vectors written in C++[56] were rewritten into Python, using librosa to generate MFCCs [48] and an NVIDIA CUDA GPU to extract then extract the i-vectors [52].

Four (4) different models were trained and evaluated, with the fourth and best model benefiting from the observed strengths of the others. The first model (m_1) achieved an accuracy of 38% on the testing database. Increasing the training time (m_2) improved this accuracy to 40%. Increasing the quality of the data (m_3) further improved this accuracy to 46%. To accomplish both at the same time for the final model (m_4), implementations were done to further improve the training time and quality of the data. Stricter validation and adding noise improved the quality of the data. Pre-processing the voice clips into spectrogram images greatly improved the training time, which also allowed the model to use Inceptionv3, a CNN with many more layers. This final model (m_4) achieved an accuracy of 60% on the testing database.

The architecture of the prototype allowed for the i-vector algorithm to successfully be deployed onto a phone, broken down into four (4) steps. First, the UI was developed so that the finished model could be interacted with. Second, the capability to record a user's voice and store it appropriately was created. Third, the voice clip was manipulated to an acceptable format for the

---

[56] Kaldi, "kaldi/egs/wsj/s5/steps/nnet/ivector/extract_ivectors.sh," *GitHub*. https://github.com/kaldi-asr/kaldi (accessed Jan. 14, 2021).

model to ingest. Fourth and finally, the finished model makes a classification based on the user's voice clip.

The application was designed using Chaqoupy[57] to allow the architecture of the prototype to function on a mobile device. Android Studio cannot process the Python commands needed to interact with the TensorFlow model, however Chaqoupy is able to load in Python libraries to an application to be called with Java commands. As the Python packages required have many dependencies, there were storage challenges to create an app within the 100MB Google Play Store limit. This was overcome by using a TensorFlow Lite model to reduce the size of the model, as well as using the Play Asset Delivery[58] function of the Google Play Store to side-load the model during the install.

Evaluation criteria was created to determine the accuracy and usability of this research fairly and objectively, broken down into four (4) principles. First, the test was conducted in a real-world scenario. Second, evaluated users had an even spread amongst the seven (7) languages considered in the prototype. Third, all biases that can impact the results were identified and mitigated. Fourth and finally, the results captured all necessary information to create meaningful findings on the accuracy and usability of the research. A step-by-step evaluation plan was created, with many steps considering the coronavirus pandemic. The evaluation plan was then granted a Certification of Ethics Approval by the University of Athabasca for conduct.

---

[57] Chaquo, "Chaquopy," *Chaquo Ltd*. https://chaquo.com/chaquopy/ (Accessed Feb. 16, 2021)

[58] Android Developers, "Play Asset Delivery," *Google Developers*. https://developer.android.com/guide/app-bundle/asset-delivery (Accessed Feb. 18, 2021)

The evaluation of the research provided results with high accuracy and high usability. The total accuracy was 81%. This is higher than the accuracy for the testing database (60%) and significantly higher than a random classification (14%). The users rated the application with an average SUS score of 95.7. This is significantly higher than the industry standard average SUS score of 68 and this score approaches the best possible design [55]. These SUS scores were then broken into three (3) datasets to determine what impacted the scoring, the independent variables being language, accent, and correctness.

To conduct data analysis on the evaluation results, normality tests and transformations were conducted. Almost all datasets were kurtotic and heavily skewed negatively, which was expected due to the scores being close to the maximum values. As statistical analysis requires the data to be normal, transformation was conducted using logarithms. This produced three (3) datasets that were able to pass the normality test.

Data analysis was conducted on the evaluation results using two types of tests, ANOVA tests[59] and a two independent samples t-test[60]. The ANOVA tests were conducted on the language and accent datasets with a Games-Howell post hoc test and found no statistically significant difference between what languages the users spoke or their self-identified accent. The two independent samples t-test however did find a significant statistical difference between whether a user was shown a correct or incorrect classification of the language they spoke.

---

[59] UCLA, "One-way ANOVA," *Statistical Consulting Group*. https://stats.idre.ucla.edu/spss/whatstat/what-statistical-analysis-should-i-usestatistical-analyses-using-spss/#1anova (Accessed Aug. 18, 2021)

[60] UCLA, "Two independent samples t-test," *Statistical Consulting Group*. https://stats.idre.ucla.edu/spss/whatstat/what-statistical-analysis-should-i-usestatistical-analyses-using-spss/#2ittest (Accessed Aug. 18, 2021)

Apart from the excellent accuracy and usability scores, three (3) concepts were proven as a result of the evaluation of this research. First, there appears to be no dependency on which language is being spoken, allowing the research to scale to many more languages. Second, there appears to be no dependency on the accent of the user, meaning the model is truly understanding what makes a language a language. Third and finally, the research is meaningful and useful to the military and greater public as users shown an incorrect classification still rated the application high, indicating the idea is exceptionally useful even if it did not function correctly for them.

Limitations

While the goal of this research is to ultimately provide a high quality solution to the problem of language translation, it will never be able to replace human translators working under ideal circumstances. The scope of this research only aims to solve the problem of using human translators with a high cost in situations where perfect accuracy is not required, namely local human translators in combat zones. Human translators working at companies and government agencies will continue to be able to translate with a higher accuracy than a machine due to the illogical and innate emotional portion of speaking a language. Idioms cause phrases such as the French phrase “c’est mon rayon,” which literally translates to “that’s my ray,” to actually translate to “that’s my cup of tea,” neither of which clearly indicate the meaning of something interesting someone. This research will not negatively impact human translators working in ideal circumstances and provides a solution to human translators in unideal circumstances.

While the current application implementation of this research is able to be hosted on the Google Play Store, replicating and improving the application will certainly lead to size challenges. Although the size challenges were overcome by using the Play Asset Delivery to side-load the model, the way Chaqoupy functions requires that all python libraries and its dependencies are

stored in the initial application file. For this research, it was extremely fortunate that the application package with all of the compressed libraries and dependencies was 149 MB, as the Google Play Store can only hold 150 MB. Adding any additional packages, or additions to the UI, will certainly break the 150 MB limit. The creator of Chaqoupy was contacted about this behaviour and although he is aware of the problem, no solution currently exists.

Almost every tool used in this research is a stable, open-source project, freely available to the public, except for Chaqoupy. Each time the application is run, a call is made to a server to validate the Chaqoupy license being used. If the application does not have a license, it will automatically close itself in five (5) minutes. While paid licenses do exist, the creator of the project generously will give out free licenses if they are contacted and shown that the overarching project is open-source. For this research, a free license was acquired after speaking with the creator and sharing some solutions to overcome size constraints using the Play Asset Delivery for the wider open-source community.

Although the accuracy of this research is high, it almost certainly suffers somewhat from having the i-vector extraction custom made instead of using Kaldi. As Kaldi has had significant development since its inception in 2009[61], it will almost certainly perform better than a reimplementation in another environment. There are many optimizations that can be used to further increase the accuracy of the i-vector extraction, all available open-source on a continually updated GitHub page owned by Kaldi[62]. The solutions to utilize Kaldi in the development

---

[61] "History of the Kaldi project," *doxygen*. http://kaldi-asr.org/doc/history.html (accessed Sep. 12, 2021).

[62] kaldi-asr, "kaldi," *GitHub*, Jan. 13, 2021. https://github.com/kaldi-asr/kaldi (accessed Jan. 12, 2021).

environment, using a Windows-based Kaldi[63] or cloud-based Kaldi[64], are currently in their infancy, but once matured, will likely allow this research's accuracy to improve.

Current virtual machine standards also harms the capability to use both TensorFlow and Kaldi together due to GPU resources. In this research , the GPU, specifically NVIDIA CUDA [52] was essential to both i-vector extraction and model training in TensorFlow. Although there have been some advancements to allow virtual machines to utilize GPUs, it is not possible for a virtual machine to utilize NVIDIA CUDA. This is because the GPU is seen as a virtual hardware rather than the real hardware[65]. Therefore, utilizing this research in an application hosted on a mobile phone that is not running Linux presents a compatibility challenge if Kaldi is going to be used for the i-vector extraction. This challenge may be solved in the future.

Future Works

The current implementation of model training allows the model to easily be expanded. More languages from the same database can be added to the existing model as each language was treated as its own modular entity. As the training database is simply folders of voice clips of a certain language, other databases can also be extracted. Once a new folder with a new language is put into the correct directory, the language simply needs to be added to the array of considered languages and the algorithm will handle its addition.

---

[63] kaldi-asr, "kaldi/windows/INSTALL.md," *GitHub*, Apr. 08, 2020. https://github.com/kaldi-asr/kaldi (accessed Jan. 12, 2021).

[64] "Kaldi: online2/online-ivector-feature.h File Reference," *doxygen*. https://kaldi-asr.org/doc/online-ivector-feature_8h.html#details (accessed Jan. 12, 2021).

[65] vmware, "Frequently Asked Questions about VMware Fusion," *VMware Inc*, Dec. 2, 2008. https://communities.vmware.com/t5/VMware-Fusion-Documents/Frequently-Asked-Questions-about-VMware-Fusion/ta-p/2779216 (accessed Sep. 12, 2021)

The datasets for each language can also be expanded. The current limitation on the datasets is only due to the amount of time required for the model to be trained. This limitation can be removed by utilizing cloud computing. TensorFlow, which is owned by Google, can easily be implemented to use Google Cloud resources[66] at a reasonable fiscal cost, allowing an incredibly large model to be trained in a reasonable amount of time. Utilizing more data per language can also avoid the use of down-sampling and instead using more advanced techniques in data science to handle imbalanced datasets than simply up-sampling[67]. The currently used Mozilla Common Voice database has a significant amount of data that is not being used and another database may not even be required.

Further augmentation of the data within the database can be done to increase the robustness of the classifications. Although the current database has different genders speaking, the pitch can be altered to emulate new speakers saying the same voice lines. Voice clips can also be interlaced with noises that would be expected to be heard in the background of use cases, such as car motors. Interlacing voice clips on top of each other can also create new clips that simulate users talking over each other.

Introducing a constant, rolling window of voice sampling and classification will not only allow the application to effortlessly identify the language of the current speaker, but also increase the applications accuracy. The current implementation of the application was designed for evaluation purposes, but in practice the algorithm will need to be consistently listening and making

---

[66] TensorFlow, "Training Keras models with TensorFlow Cloud | TensorFlow Core," *Google*. https://www.tensorflow.org/guide/keras/training_keras_models_on_cloud (Accessed Feb. 8, 2021)

[67] TensorFlow, "Classification on imbalanced data | TensorFlow Core," *Google*. https://www.tensorflow.org/tutorials/structured_data/imbalanced_data (accessed Dec. 21, 2020).

classifications. This will increase the accuracy as the algorithm can use the average classification of a certain speaker. Since the accuracy is significantly above the accuracy of a random classification (14%), the average classification will almost certainly be correct.

With an average language classification, the application will require ASR to identify when a certain user is speaking. One of the existing relevant SLRs considered during this research's SLR was specifically on ASR. Many of the papers read for the SLR were also specifically about ASR and how to identify when a certain user is speaking. Utilizing existing ASR algorithms, once a classification is made using a rolling window to identify what language a specific user is speaking, the application can then stop making a language classification and just remember what language the users are speaking.

To utilize the research in a useful way, it will then need to be overlayed into an application that can make use of knowing what language users are speaking. Google Translate requires users to manually select what language they are speaking before listening and automatically translating the spoken dialog into the specified languages. Integrating this research with Google Translate will allow users to skip this manual selection and simply begin speaking in their languages. The application will identify the language being spoken and set the appropriate fields, allowing Google Translate to start translation. This will result in users being able to open up the application and begin speaking their own language knowing that it will be translated into the other users' languages all automatically.

Once this research has been integrated with a translation service, the next step to improve classifications would be to account for accents. While the number of accents is very large, computed as the number of languages in the world squared when accounting for no accents and not accounting for regional accents, it is a finite number. Although finding datasets would be

difficult, the application can introduce a feedback loop to capture any new accents. By training the model further through “field-use,” the robustness of the application can continue to grow for all users.

References

[1] Department of National Defence, "CFJP01 - Canadian Military Doctrine," Joint Doctrine Branch, Ottawa, 2009.

[2] Internet World Stats, "World Internet Users Statistics and 2020 World Population Stats," *Miniwatts Marketing Group*, Jul. 20, 2020. https://www.internetworldstats.com/stats.htm (accessed Jul. 27, 2020).

[3] Google, "Google Translate," *Chrome Web Store*, Mar. 17, 2020. https://chrome.google.com/webstore/detail/google-translate/aapbdbdomjkkjkaonfhkkikfgjllcleb?hl=en-GB (accessed Jul. 27, 2020).

[4] K. McNally, "Library Resource Guides: Literature Review: Systematic literature review," *Charles Sturt University Library*, Jul. 16, 2020. https://libguides.csu.edu.au/c.php?g=476545&p=3997202 (accessed Jul. 27, 2020).

[5] L. S. Uman, "Systematic Reviews and Meta-Analyses," *J. Can. Acad. Child Adolesc. Psychiatry*, vol. 20, no. 1, pp. 57–59, Feb. 2011, Accessed: Jul. 27, 2020. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3024725/

[6] J. Clark, "Why 2015 Was a Breakthrough Year in Artificial Intelligence," *Bloomberg.com*, Dec. 08, 2015. Accessed: Nov. 05, 2020. [Online]. Available: https://www.bloomberg.com/news/articles/2015-12-08/why-2015-was-a-breakthrough-year-in-artificial-intelligence

[7] "The Machine Learning Times of Year 2015 - A Powerful Growth Story," *Analytics Vidhya*, Nov. 24, 2015. https://www.analyticsvidhya.com/blog/2015/11/infographic-rise-machine-learning-year-2015/ (accessed Nov. 05, 2020).

[8] A. Kurenkov, "A Brief History of Neural Nets and Deep Learning," *Skynet Today*, Sep. 27, 2020. https://www.skynettoday.com/overviews/neural-net-history (accessed Nov. 05, 2020).

[9] Google, "Google Scholar Metrics Help," *Google Scholar*, Jun. 2020. https://scholar.google.com/intl/en/scholar/metrics.html#metrics (accessed Aug. 12, 2020).

[10] Z. Zhang, N. Cummins, and B. Schuller, "Advanced Data Exploitation in Speech Analysis: An overview," *IEEE Signal Process. Mag.*, vol. 34, no. 4, pp. 107–129, Jul. 2017, doi: 10.1109/MSP.2017.2699358.

[11] R. Haeb-Umbach *et al.*, "Speech Processing for Digital Home Assistants: Combining Signal Processing With Deep-Learning Techniques," *IEEE Signal Process. Mag.*, vol. 36, no. 6, pp. 111–124, Nov. 2019, doi: 10.1109/MSP.2019.2918706.

[12] R. Travadi and S. Narayanan, "Efficient estimation and model generalization for the totalvariability model," *Comput. Speech Lang.*, vol. 53, pp. 43–64, Jan. 2019, doi: 10.1016/j.csl.2018.07.003.

[13] S. Ramoji and S. Ganapathy, "Supervised I-vector modeling for language and accent recognition," *Comput. Speech Lang.*, vol. 60, p. 101030, Mar. 2020, doi: 10.1016/j.csl.2019.101030.

[14] J. Monteiro, J. Alam, and T. H. Falk, "Residual convolutional neural network with attentive feature pooling for end-to-end language identification from short-duration speech," *Comput. Speech Lang.*, vol. 58, pp. 364–376, Nov. 2019, doi: 10.1016/j.csl.2019.05.006.

[15] I. Lopez-Moreno, J. Gonzalez-Dominguez, D. Martinez, O. Plchot, J. Gonzalez-Rodriguez, and P. J. Moreno, "On the use of deep feedforward neural networks for automatic language identification," *Comput. Speech Lang.*, vol. 40, pp. 46–59, Nov. 2016, doi: 10.1016/j.csl.2016.03.001.

[16] D. Nandi, D. Pati, and K. S. Rao, "Parametric representation of excitation source information for language identification," *Comput. Speech Lang.*, vol. 41, pp. 88–115, Jan. 2017, doi: 10.1016/j.csl.2016.05.001.

[17] D. Nandi, D. Pati, and K. S. Rao, "Implicit processing of LP residual for language identification," *Comput. Speech Lang.*, vol. 41, pp. 68–87, Jan. 2017, doi: 10.1016/j.csl.2016.06.002.

[18] X. Lu, P. Shen, Y. Tsao, and H. Kawai, "Regularization of neural network model with distance metric learning for i-vector based spoken language identification," *Comput. Speech Lang.*, vol. 44, pp. 48–60, Jul. 2017, doi: 10.1016/j.csl.2017.01.006.

[19] K. Walker, X. Ma, D. Graff, S. Strassel, S. Sessa, and K. Jones, "RATS Speech Activity Detection," *Linguistic Data Consortium*, Feb. 16, 2015. https://catalog.ldc.upenn.edu/LDC2015S02 (accessed Nov. 04, 2020).

[20] National Institute of Standards and Technology, "Language Recognition," *U.S. Department of Commerce*, Jan. 24, 2011. https://www.nist.gov/itl/iad/mig/language-recognition (accessed Nov. 04, 2020).

[21] D. Wang, Z. Tang, and Q. Chen, "OLR Challenge 2018," *Center for Speech and Language Technology*, Apr. 04, 2019. http://cslt.riit.tsinghua.edu.cn/mediawiki/index.php/OLR_Challenge_2018 (accessed Nov. 04, 2020).

[22] S. Maity, A. K. Vuppala, K. S. Rao, and D. Nandi, "IITKGP-MLILSC speech database for language identification," in *2012 National Conference on Communications (NCC)*, Feb. 2012, pp. 1–5. doi: 10.1109/NCC.2012.6176831.

[23] R. Cole and Y. Muthusamy, "OGI Multilanguage Corpus," *Linguistic Data Consortium*, 1994. https://catalog.ldc.upenn.edu/LDC94S17 (accessed Nov. 04, 2020).

[24] T. Drugman, Y. Stylianou, Y. Kida, and M. Akamine, "Voice Activity Detection: Merging Source and Filter-based Information," *IEEE Signal Process. Lett.*, vol. 23, no. 2, pp. 252–256, Feb. 2016, doi: 10.1109/LSP.2015.2495219.

[25] J. Kim and M. Hahn, "Voice Activity Detection Using an Adaptive Context Attention Model," *IEEE Signal Process. Lett.*, vol. 25, no. 8, pp. 1181–1185, Aug. 2018, doi: 10.1109/LSP.2018.2811740.

[26] S. Seshadri and O. Räsänen, "SylNet: An Adaptable End-to-End Syllable Count Estimator for Speech," *IEEE Signal Process. Lett.*, vol. 26, no. 9, pp. 1359–1363, Sep. 2019, doi: 10.1109/LSP.2019.2929415.

[27] T. Aguiar de Lima and M. Da Costa-Abreu, “A survey on automatic speech recognition systems for Portuguese language and its variations,” *Comput. Speech Lang.*, vol. 62, p. 101055, Jul. 2020, doi: 10.1016/j.csl.2019.101055.

[28] E. Lleida and L. J. Rodriguez-Fuentes, “Speaker and language recognition and characterization: Introduction to the CSL special issue,” *Comput. Speech Lang.*, vol. 49, pp. 107–120, May 2018, doi: 10.1016/j.csl.2017.12.001.

[29] K. Wu, D. Zhang, G. Lu, and Z. Guo, “Joint learning for voice based disease detection,” *Pattern Recognit.*, vol. 87, pp. 130–139, Mar. 2019, doi: 10.1016/j.patcog.2018.09.013.

[30] J. S. Almeida *et al.*, “Detecting Parkinson’s disease with sustained phonation and speech signals using machine learning techniques,” *Pattern Recognit. Lett.*, vol. 125, pp. 55–62, Jul. 2019, doi: 10.1016/j.patrec.2019.04.005.

[31] M. Calvo, L.-F. Hurtado, F. Garcia, E. Sanchis, and E. Segarra, “Multilingual Spoken Language Understanding using graphs and multiple translations,” *Comput. Speech Lang.*, vol. 38, pp. 86–103, Jul. 2016, doi: 10.1016/j.csl.2016.01.002.

[32] A. O. Bayer and G. Riccardi, “Semantic language models with deep neural networks,” *Comput. Speech Lang.*, vol. 40, pp. 1–22, Nov. 2016, doi: 10.1016/j.csl.2016.04.001.

[33] J. M. Martin-Doñas, A. M. Gomez, J. A. Gonzalez, and A. M. Peinado, “A Deep Learning Loss Function Based on the Perceptual Evaluation of the Speech Quality,” *IEEE Signal Process. Lett.*, vol. 25, no. 11, pp. 1680–1684, Nov. 2018, doi: 10.1109/LSP.2018.2871419.

[34] A. Wadhawan and P. Kumar, "Sign Language Recognition Systems: A Decade Systematic Literature Review," *Arch. Comput. Methods Eng.*, pp. 1–29, Dec. 2019, doi: 10.1007/s11831-019-09384-2.

[35] Y. Kumar and N. Singh, "A Comprehensive View of Automatic Speech Recognition System - A Systematic Literature Review," in *2019 International Conference on Automation, Computational and Technology Management (ICACTM)*, Apr. 2019, pp. 168–173. doi: 10.1109/ICACTM.2019.8776714.

[36] F. Salustri, "Weighted Decision Matrix," *Ryerson University*, Jul. 25, 2020. https://deseng.ryerson.ca/dokuwiki/design:weighted_decision_matrix (accessed Nov. 05, 2020).

[37] W. Khan *et al.*, "Urdu part of speech tagging using conditional random fields," *Lang. Resour. Eval.*, vol. 53, no. 3, pp. 331–362, Sep. 2019, doi: 10.1007/s10579-018-9439-6.

[39] P. Kenny, "Joint factor analysis of speaker and session variability: Theory and algorithms," Jan. 2006.

[40] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-End Factor Analysis for Speaker Verification," *IEEE Trans. Audio Speech Lang. Process.*, vol. 19, no. 4, pp. 788–798, May 2011, doi: 10.1109/TASL.2010.2064307.

[41] M. Li and S. Narayanan, "Simplified supervised i-vector modeling with application to robust and efficient language identification and speaker verification," *Comput. Speech Lang.*, vol. 28, no. 4, pp. 940–958, Jul. 2014, doi: 10.1016/j.csl.2014.02.004.

[42] D. Povey *et al.*, "The Kaldi Speech Recognition Toolkit," *IEEE Signal Process. Soc.*, p. 4, Dec. 2011.

[43] "Google Just Open Sourced the Artificial Intelligence Engine at the Heart of Its Online Empire," *Wired*. Accessed: Jan. 27, 2021. [Online]. Available: https://www.wired.com/2015/11/google-open-sources-its-artificial-intelligence-engine/

[44] J. Brownlee, "A Gentle Introduction to Pooling Layers for Convolutional Neural Networks," *Machine Learning Mastery*, Apr. 21, 2019. https://machinelearningmastery.com/pooling-layers-for-convolutional-neural-networks/ (accessed Jan. 13, 2021).

[45] J. Brownlee, "How to Control the Stability of Training Neural Networks With the Batch Size," *Machine Learning Mastery*, Jan. 20, 2019. https://machinelearningmastery.com/how-to-control-the-speed-and-stability-of-training-neural-networks-with-gradient-descent-batch-size/ (accessed Jan. 16, 2021).

[46] B. McFee *et al.*, "librosa: Audio and Music Signal Analysis in Python," Jan. 2015, pp. 18–24. doi: 10.25080/Majora-7b98e3ed-003.

[47] Amrita Vishwa Vidyapeetham Virtual Lab, "Sampling Frequency and Bit Resolution for Speech Signal Processing (Theory)," *ICT*. https://vlab.amrita.edu/?sub=3&brch=164&sim=474&cnt=1 (accessed Jan. 14, 2021).

[48] K. S. Rao and M. K. E, *Speech Recognition Using Articulatory and Excitation Source Features*. Springer, 2017.

[49] M. Hossan, S. Memon, and M. Gregory, "A novel approach for MFCC feature extraction," Jan. 2011, pp. 1–5. doi: 10.1109/ICSPCS.2010.5709752.

[50] S. Seyedin, S. M. Ahadi, and S. Gazor, "New Features Using Robust MVDR Spectrum of Filtered Autocorrelation Sequence for Robust Speech Recognition," *The Scientific World Journal*, Dec. 31, 2013. https://www.hindawi.com/journals/tswj/2013/634160/ (accessed Jan. 15, 2021).

[51] H. M. Naing, R. Hidayat, R. Hartanto, and Y. Miyanaga, "Discrete Wavelet Denoising into MFCC for Noise Suppressive in Automatic Speech Recognition System," *Int. J. Intell. Eng. Syst.*, vol. 13, pp. 74–82, Apr. 2020, doi: 10.22266/ijies2020.0430.08.

[52] L. Barnes and J. Luitjens, "Extracting Features from Multiple Audio Channels with Kaldi," *NVIDIA Developer Blog*, Aug. 20, 2020. https://developer.nvidia.com/blog/extracting-features-from-multiple-audio-channels-with-kaldi/ (accessed Jan. 15, 2021).

[53] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," Dec. 2015, Accessed: Jan. 27, 2021. [Online]. Available: https://arxiv.org/abs/1512.00567v3

[54] J. Sauro, "Measuring Usability with the System Usability Scale (SUS)." https://measuringu.com/sus/ (accessed Mar. 20, 2021).

[55] "The System Usability Scale & How it's Used in UX | Adobe XD Ideas," *Ideas*. https://xd.adobe.com/ideas/process/user-testing/sus-system-usability-scale-ux/ (accessed Aug. 26, 2021).

[56] H.-Y. Kim, “Statistical notes for clinical researchers: assessing normal distribution (2) using skewness and kurtosis,” *Restor. Dent. Endod.*, vol. 38, no. 1, pp. 52–54, Feb. 2013, doi: 10.5395/rde.2013.38.1.52.

Appendix A: Model Results

Sentence	Language	Sentence in English	Speaker	Speaker's Native Language	m_1	m_2	m_3	m_4
s_1	en	In addition to this book there are a few things you will need to play Pathfinder. These supplies can be found at your local hobby shop.	u_1	en	de	de	de	de
s_2	en	Pathfinder is played in sessions, during which players gather in person or online for a few hours to play the game. A complete Pathfinder story can be as short as a single session, commonly referred to as a "one-shot," or it can stretch on for multiple sessions, forming a campaign that might last for months or even years.	u_1	en	de	de	en	en
s_3	en	Before creating your first character or adventure, you should understand a number of basic concepts used in the game.	u_1	en	de	zh-CN	zh-CN	zh-CN
s_4	en	During the game, players describe the actions their characters take and roll dice, using their characters' abilities. The GM resolves the outcome of these actions.	u_1	en	de	de	en	en
s_5	en	Whether you are the GM or a player, participating in a tabletop roleplaying game includes a social contract: everyone has gathered together to have fun telling a story.	u_1	en	de	zh-CN	en	en
s_6	en	Level is one of the most important statistics of the game, as it conveys the approximate power and capabilities of every individual creature.	u_1	en	de	de	en	en
s_7	en	An ability score that's above the average increases your chance of success at tasks related to the ability score, while those below the average decrease your chance.	u_1	en	de	de	en	en

Sentence	Language	Sentence in English	Speaker	Speaker's Native Language	m_1	m_2	m_3	m_4
s_8	en	The GM determines the premise and background of most adventures, although character histories and personalities certainly play a part. Once a game session begins, the players take turns describing what their characters attempt to do, while the GM determines the outcome, with the table working together toward a specific goal.	u_1	en	de	en	en	en
s_9	en	Every feat has a type to denote where its explanation can be found (for example, elf feats can be found in the elf ancestry) and its theme (wizard feats, for example, grant abilities that deal with spells).	u_1	en	de	en	en	en
s_{10}	en	Characters and their choices create the story of Pathfinder, but how they interact with each other and the world around them is governed by rules.	u_1	en	de	fa	fr	de
s_{11}	en	Throughout this mode of play, the GM asks the players what their characters are doing as they explore. This is important in case a conflict arises.	u_2	fr	de	en	en	en
s_{12}	en	Free actions, such as dropping an object, don't count toward the three actions you can take on your turn. Finally each character can use up to one reaction during a round.	u_3	fr	zh-CN	en	en	en
s_{13}	en	Attacking another creature is one of the most common actions in combat, and is done by using the Strike action. This requires an attack roll—a kind of check made against the Armor Class (AC) of the creature you're attacking.	u_3	fr	de	de	en	en
s_{14}	en	Strikes can be made using weapons, spells, or even parts of a creature's body, like a fist, claw, or tail. You add a modifier to this roll based on your proficiency rank with the type of attack you're using, your ability scores, and any other bonuses or penalties based on the situation.	u_3	fr	de	ru	ru	en

Sentence	Language	Sentence in English	Speaker	Speaker's Native Language	m_1	m_2	m_3	m_4
s_{15}	en	The target's AC is calculated using their proficiency rank in the armor they're wearing and their Dexterity modifier. An attack deals damage if it hits, and rolling a critical success results in the attack dealing double damage!	u_3	fr	de	en	en	en
s_{16}	en	I would like my steak medium rare.	u_4	en	ar	en	en	en
s_{17}	en	Does this dish have gluten?	u_5	en	en	en	zh-CN	ru
s_{18}	en	People that have experienced so-called 'lucid dreams' often describe them as being 'more real than reality'. They also describe reality after waking up from a 'lucid dream' to be like a 'whimsical dream'.	u_6	en	en	en	en	en
s_{19}	en	We had a power outage, so I had to reset the clock on my VCR. However, I was off by an hour, so the program I wanted to record wasn't recorded.	u_7	en	en	en	ru	ru
s_{20}	en	I wanted to visit Tom next Monday, but he said he was going to be busy, so I'm planning to visit him the Monday after next.	u_7	en	de	de	de	en
s_{21}	en	Yonder is the gymnasium. Down there to the right is the stables for the riding horse, but that building is more than a quarter mile from here.	u_8	en	en	de	de	fr
s_{22}	en	The screening of donated blood for the presence of various illnesses is expensive, but is also necessary to keep the blood supply as safe as possible.	u_9	en	en	de	en	de
s_{23}	en	When we watch a movie, play a video game, or read a book, we become emotionally attached to certain characters and gradually become like them.	u_{10}	en	en	fr	en	en
s_{24}	en	For the sake of completeness, let us mention that the ring R, considered as a module over itself, has submodules of arbitrarily large finite length.	u_{11}	en	en	en	en	en
s_{25}	en	I can't promise that you'll like these books but I think it would be a good idea to at least look them over.	u_{12}	en	ru	de	de	de

Sentence	Language	Sentence in English	Speaker	Speaker's Native Language	m_1	m_2	m_3	m_4
s_{26}	fr	When combat begins, all players make an initiative roll.	u_1	en	de	de	fr	de
s_{27}	fr	At the start of combat, the players that are unaware of the enemy's presences are automatically surprised (provided they are spotted by the opposing side of course).	u_1	en	de	de	fr	fr
s_{28}	fr	Most perception checks are done in response to an observable stimulus.	u_1	en	de	de	de	fr
s_{29}	fr	The perception skill can be used in a number of ways. Most often, this is a check against the opponent's stealth check.	u_1	en	de	de	fr	fr
s_{30}	fr	The perception skill also allows you to notice certain details in the character's environment.	u_1	en	de	de	de	de
s_{31}	fr	Start by determining the characteristics of your character. These six values represent the core qualities of your character, and there are a lot of things that depend on them.	u_2	fr	de	de	de	fr
s_{32}	fr	Then choose your character's race and indicate any modifiers that apply to your characteristics as well as any other racial traits. You can choose from seven basic races, and your GM may have more to add to the list.	u_2	fr	de	de	de	fr
s_{33}	fr	A character's class represents their profession (magician or warrior, for example). If this is a new character, it starts at level 1 in the chosen class.	u_2	fr	de	de	fr	fr
s_{34}	fr	Determine the number of skill ranks your character has, based on their class and Intelligence modifier (as well as any other bonuses, such as humans). Then divide these ranks between the skills while remembering that each of them cannot receive a number of ranks higher than your level.	u_2	fr	de	de	de	fr
s_{35}	fr	A new character enters play with a certain amount of gold depending on his class, which he can spend on equipment and materials (such as a chain mail or a leather backpack).	u_2	fr	de	de	en	en

Sentence	Language	Sentence in English	Speaker	Speaker's Native Language	m_1	m_2	m_3	m_4
s_{36}	fr	Start by determining the characteristics of your character. These six values represent the core qualities of your character, and there are a lot of things that depend on them.	u_3	fr	de	fr	de	fr
s_{37}	fr	Then choose your character's race and indicate any modifiers that apply to your characteristics as well as any other racial traits. You can choose from seven basic races, and your GM may have more to add to the list.	u_3	fr	de	fr	fr	fr
s_{38}	fr	A character's class represents their profession (magician or warrior, for example). If this is a new character, it starts at level 1 in the chosen class.	u_3	fr	fr	de	de	ru
s_{39}	fr	Determine the number of skill ranks your character has, based on their class and Intelligence modifier (as well as any other bonuses, such as humans). Then divide these ranks between the skills while remembering that each of them cannot receive a number of ranks higher than your level	u_3	fr	de	fr	fr	fr
s_{40}	fr	He is very much interested in Japanese history. We are surprised at his vast knowledge of the subject.	u_{13}	fr	de	de	de	de
s_{41}	fr	After reflecting on my life up to now, I decided that I needed to change my goals.	u_{14}	fr	ru	fr	ar	zh-CN
s_{42}	fr	I think it says something about human nature that the only form of life we have created so far is purely destructive.	u_{15}	fr	de	fr	fr	fr
s_{43}	fr	I have tried for hours to remember where I put my keys, but it has completely escaped me.	u_{16}	fr	fr	fr	fr	fr
s_{44}	fr	When you meet someone for the first time, be careful about how close you stand to that person.	u_{17}	fr	de	fr	ar	fr
s_{45}	fr	On a rainy morning he left his house early so as to be in time for school.	u_{18}	fr	ar	fr	fr	fr
s_{46}	fr	In a fit of anger he said everything he wanted to say and went home.	u_{19}	fr	fr	de	en	en

Sentence	Language	Sentence in English	Speaker	Speaker's Native Language	m_1	m_2	m_3	m_4
s_{47}	fr	The fireplace, lacking firewood, flames already starting to lose their vigour.	u_{20}	fr	fr	ru	ru	fr
s_{48}	fr	In hard times like this, no ordinary effort can get our company out of the red.	u_{21}	fr	fr	fr	fr	ar
s_{49}	fr	It is illegal for bicycles to pass on the right of cars.	u_{22}	fr	fr	ar	fa	fa
s_{50}	fr	The one resource more precious than any other was land.	u_{23}	fr	ru	fr	fr	ru
s_{51}	ar	Where do I pay?	u_{24}	ar	fr	de	de	ar
s_{52}	ar	Could I have this delivered?	u_{25}	ar	ar	fa	fa	ar
s_{53}	ar	Could I have a receipt please?	u_{26}	ar	fr	ar	fa	zh-CN
s_{54}	ar	Do you have this in a larger size?	u_{27}	ar	ar	ar	fr	ar
s_{55}	ar	Will I be charged a fee?	u_{28}	ar	fa	ar	ar	ar
s_{56}	ar	What is the confirmation number?	u_{29}	ar	ar	en	fr	ar
s_{57}	ar	I forgot my password.	u_{30}	ar	ar	de	en	en
s_{58}	ar	The price is incorrect.	u_{31}	ar	ar	ar	ar	ar
s_{59}	ar	Could I have a glass of tap water?	u_{32}	ar	de	en	en	en
s_{60}	ar	Could you please bring me a spoon?	u_{33}	ar	ar	en	fa	fa
s_{61}	de	Please take me to the hospital.	u_{34}	de	en	fa	en	de
s_{62}	de	What time do they open?	u_{34}	de	fr	zh-CN	de	de
s_{63}	de	Can I make an appointment?	u_{34}	de	fa	en	en	fr
s_{64}	de	My back hurts.	u_{34}	de	fr	ar	fr	de
s_{65}	de	Do you come here often?	u_{35}	de	en	zh-CN	zh-CN	de
s_{66}	de	What would you like to have?	u_{34}	de	ar	zh-CN	de	fr
s_{67}	de	I'm glad I came here tonight.	u_{36}	de	en	fa	fa	de
s_{68}	de	Where are you from?	u_{34}	de	de	en	de	en
s_{69}	de	What's your name?	u_{37}	de	fa	en	en	fr
s_{70}	de	What do you do?	u_{34}	de	en	fa	fa	zh-CN
s_{71}	fa	We have the place to ourselves.	u_{38}	fa	de	ru	de	zh-CN
s_{72}	fa	Who am I to say?	u_{39}	fa	ru	fa	fr	fa
s_{73}	fa	It's about time!	u_{40}	fa	ru	fa	fa	zh-CN
s_{74}	fa	Thank you very much	u_{41}	fa	fa	zh-CN	de	zh-CN
s_{75}	fa	Is there a good time to chat?	u_{42}	fa	fa	fa	fa	fa
s_{76}	fa	Where are we meeting?	u_{43}	fa	fa	fa	fa	fa
s_{77}	fa	Is there sales tax?	u_{44}	fa	fa	en	en	de

Sentence	Language	Sentence in English	Speaker	Speaker's Native Language	m_1	m_2	m_3	m_4
s_{78}	fa	I need some help please.	u_{45}	fa	en	ar	ar	fa
s_{79}	fa	May I speak to the store manager please?	u_{46}	fa	ar	en	fa	en
s_{80}	fa	I'm just browsing, thanks.	u_{47}	fa	zh-CN	fa	fa	fa
s_{81}	ru	Turn left at the stop sign.	u_{48}	ru	ru	ru	fr	ru
s_{82}	ru	Right around the corner from here.	u_{49}	ru	en	en	en	fr
s_{83}	ru	Go under the bridge.	u_{50}	ru	fa	ar	ar	ru
s_{84}	ru	Where does this train go?	u_{51}	ru	ar	ar	fr	ar
s_{85}	ru	Is this seat taken?	u_{52}	ru	ru	ru	fr	en
s_{86}	ru	What gate do we need to go to?	u_{53}	ru	ru	ru	ar	ar
s_{87}	ru	Is the ticket oneway or roundtrip?	u_{54}	ru	ru	ru	ar	ru
s_{88}	ru	Is there a parking garage near here?	u_{55}	ru	fr	fr	fr	fr
s_{89}	ru	How many stops are left?	u_{55}	ru	fa	fa	en	ru
s_{90}	ru	I am allergic to nuts.	u_{56}	ru	fa	de	en	en
s_{91}	zh-CN	No, I brought her here to have her teeth straightened. Look at her teeth, they're going in all directions. They look terrible.	u_{57}	zh-CN	zh-CN	ar	en	ru
s_{92}	zh-CN	Whoa! Amazingly, France used its newly-designed biological weapon. There are large-scale breakouts of the disease in the capital city. We're requesting that the UN send medical assistance!	u_{58}	zh-CN	zh-CN	zh-CN	zh-CN	zh-CN
s_{93}	zh-CN	President Trump has said he was pulling out of the agreement because it would not prevent what he termed the world's leading state sponsor of terror from getting a nuclear weapon.	u_{59}	zh-CN	zh-CN	zh-CN	zh-CN	zh-CN
s_{94}	zh-CN	If there is a market demand, we need to satisfy the demand. If there is no market demand, we then need to create the demand.	u_{60}	zh-CN	zh-CN	zh-CN	zh-CN	zh-CN
s_{95}	zh-CN	First, thank you very much for giving me this chance for an interview. My major was marketing. Ever since graduating, I've been in sales.	u_{61}	zh-CN	zh-CN	fa	zh-CN	zh-CN
s_{96}	zh-CN	It depends on the situation. I think they have some kind of tool which you can insert through the peephole and unlock the door.	u_{62}	zh-CN	zh-CN	zh-CN	zh-CN	zh-CN

Sentence	Language	Sentence in English	Speaker	Speaker's Native Language	m_1	m_2	m_3	m_4
s_{97}	zh-CN	Some countries are like this, so people become lazier and lazier. The country's economy also becomes worse and worse. I couldn't imagine that kind of life.	u_{63}	zh-CN	zh-CN	zh-CN	zh-CN	zh-CN
s_{98}	zh-CN	Good thing he's just our manager. Imagine being his wife or kids... they're the really unlucky ones!	u_{64}	zh-CN	zh-CN	fr	zh-CN	ar
s_{99}	zh-CN	Speaking of crying all day, it's me that's going to wallowing around the place now. It's hard to imagine that I have to wait a whole year to watch the next season!	u_{65}	zh-CN	zh-CN	zh-CN	zh-CN	zh-CN
s_{100}	zh-CN	It's good that you're so calm. I can't at all imagine what else has been stolen.	u_{66}	zh-CN	zh-CN	zh-CN	zh-CN	zh-CN
				Total Accuracy:	38%	40 %	46 %	60 %
				Total Time:	44s	45s	43s	66s

Appendix B: Evaluation Questionnaire

Steps have been taken to ensure your responses remain anonymous. This form will be put into a locked box which will only be opened once 100 participants complete evaluations. The Commanding Officer has the key to this box.

Accuracy – Select one per column

Language Spoken	Degree of Accent	Language Identified	Confidence
◯ English	◯ I have no accent	◯ English	◯ < 30%
◯ French	◯ I have a bit of an accent	◯ French	◯ 40% - 50%
◯ Arabic	◯ I have a strong accent	◯ Arabic	◯ 50% - 60%
◯ Russian	◯ I have a strong regional accent	◯ Russian	◯ 60% - 70%
◯ Persian		◯ Persian	◯ 70% - 80%
◯ Chinese (China)		◯ Chinese (China)	◯ 80% - 90%
◯ German		◯ German	◯ 90% - 100%

Usability – Select one per row

System Usability Scale Question	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
	1	2	3	4	5
1. I think that I would like to use this system frequently.	◯	◯	◯	◯	◯
2. I found the system unnecessarily complex.	◯	◯	◯	◯	◯
3. I thought the system was easy to use.	◯	◯	◯	◯	◯
4. I think that I would need the support of a technical person to be able to use this system.	◯	◯	◯	◯	◯
5. I found the various functions in this system were well integrated.	◯	◯	◯	◯	◯
6. I thought there was too much inconsistency in this system.	◯	◯	◯	◯	◯
7. I would imagine that most people would learn to use this system very quickly.	◯	◯	◯	◯	◯
8. I found the system very cumbersome to use.	◯	◯	◯	◯	◯
9. I felt very confident using the system.	◯	◯	◯	◯	◯
10. I needed to learn a lot of things before I could get going with this system.	◯	◯	◯	◯	◯

Appendix C: Certification of Ethical Approval

Athabasca University
RESEARCH CENTRE

CERTIFICATION OF ETHICAL APPROVAL

The Athabasca University Research Ethics Board (REB) has reviewed and approved the research project noted below. The REB is constituted and operates in accordance with the current version of the Tri-Council Policy Statement: Ethical Conduct for Research Involving Humans (TCPS2) and Athabasca University Policy and Procedures.

Ethics File No.: **24324**

Principal Investigator:
Mr. Ripley Pennell, Graduate Student
Faculty of Science & Technology\Master of Science in Information Systems (MScIS)

Supervisor:
Dr. Maiga Chang (Supervisor)

Project Title:
Evaluation of Thesis: Automated Spoken Language Detection

Effective Date: May 27, 2021 **Expiry Date**: May 26, 2022

Restrictions:

Any modification or amendment to the approved research must be submitted to the AUREB for approval.

Ethical approval is valid *for a period of one year*. An annual request for renewal must be submitted and approved by the above expiry date if a project is ongoing beyond one year.

A Project Completion (Final) Report must be submitted when the research is complete *(i.e. all participant contact and data collection is concluded, no follow-up with participants is anticipated and findings have been made available/provided to participants (if applicable))* or the research is terminated.

Approved by: **Date: May 27, 2021**

Jon Dron, Chair
School of Computing & Information Systems, Departmental Ethics Review Committee

---

Athabasca University Research Ethics Board
University Research Services, Research Centre
1 University Drive, Athabasca AB Canada T9S 3A3
E-mail rebsec@athabascau.ca
Telephone: 780.213.2033

Appendix D: Evaluation Results

Accuracy				System Usability Scale									
Language Spoken	**Degree of Accent**	**Language Identified**	**Confidence**	**1. I think that I would like to use this system frequently.**	**2. I found the system unnecessarily complex.**	**3. I thought the system was easy to use.**	**4. I think that I would need the support of a technical person to be able to use this system.**	**5. I found the various functions in this system were well integrated.**	**6. I thought there was too much inconsistency in this system.**	**7. I would imagine that most people would learn to use this system very quickly.**	**8. I found the system very cumbersome to use.**	**9. I felt very confident using the system.**	**10. I needed to learn a lot of things before I could get going with this system.**
English	I have no accent	German	50% - 60%	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Agree	Neutral	Strongly Agree	Strongly Disagree	Disagree	Strongly Disagree
Russian	I have no accent	English	80% - 90%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Agree	Strongly Agree	Strongly Disagree	Neutral	Strongly Disagree
Persian	I have a bit of an accent	Persian	80% - 90%	Neutral	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
English	I have no accent	English	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Disagree	Strongly Agree	Strongly Disagree
French	I have a bit of an accent	French	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
French	I have a bit of an accent	French	80% - 90%	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Persian	I have a strong accent	Chinese (China)	70% - 80%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Neutral	Agree	Strongly Agree	Strongly Disagree	Disagree	Strongly Disagree
German	I have a strong accent	German	90% - 100%	Strongly Agree	Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Chinese (China)	I have no accent	Chinese (China)	80% - 90%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Arabic	I have a bit of an accent	Arabic	80% - 90%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
French	I have a strong accent	French	90% - 100%	Agree	Strongly Disagree	Strongly Agree	Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Arabic	I have a strong accent	Arabic	70% - 80%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
French	I have no accent	German	60% - 70%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Disagree
Chinese (China)	I have a strong accent	Chinese (China)	70% - 80%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Persian	I have a strong accent	Persian	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
German	I have a bit of an accent	German	90% - 100%	Disagree	Neutral	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Persian	I have a strong accent	Persian	70% - 80%	Neutral	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Persian	I have a bit of an accent	Chinese (China)	50% - 60%	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
English	I have no accent	English	70% - 80%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
English	I have no accent	English	80% - 90%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
German	I have a strong accent	German	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
German	I have a bit of an accent	German	80% - 90%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Disagree	Agree	Strongly Disagree
Russian	I have no accent	Russian	80% - 90%	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
French	I have no accent	French	80% - 90%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
French	I have a strong accent	French	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Disagree	Strongly Agree	Strongly Disagree
Chinese (China)	I have no accent	Chinese (China)	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
English	I have no accent	Russian	80% - 90%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
English	I have no accent	English	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Disagree	Strongly Agree	Strongly Disagree	Agree	Strongly Disagree
Chinese (China)	I have no accent	Chinese (China)	70% - 80%	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
English	I have no accent	English	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Russian	I have no accent	French	60% - 70%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Neutral	Strongly Agree	Strongly Disagree	Neutral	Strongly Disagree
German	I have a strong accent	German	80% - 90%	Neutral	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Chinese (China)	I have a strong accent	Chinese (China)	80% - 90%	Agree	Neutral	Strongly Agree	Strongly Disagree	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Russian	I have no accent	Arabic	70% - 80%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Neutral	Strongly Agree	Strongly Disagree	Disagree	Strongly Disagree
German	I have a strong accent	German	50% - 60%	Disagree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
English	I have a strong regional accent	English	70% - 80%	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Agree	Strongly Disagree	Strongly Agree	Disagree
French	I have a strong accent	French	90% - 100%	Strongly Agree	Disagree	Strongly Agree	Strongly Disagree	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Chinese (China)	I have a strong accent	French	50% - 60%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Disagree	Strongly Agree	Strongly Disagree	Agree	Strongly Disagree
English	I have no accent	English	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Chinese (China)	I have a strong accent	Chinese (China)	60% - 70%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
French	I have a strong regional accent	French	70% - 80%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Persian	I have a strong accent	Persian	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
French	I have no accent	French	80% - 90%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
French	I have a strong accent	French	70% - 80%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Chinese (China)	I have a strong accent	Chinese (China)	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
English	I have a bit of an accent	English	90% - 100%	Neutral	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Arabic	I have a strong accent	Arabic	60% - 70%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Arabic	I have a strong accent	English	70% - 80%	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Disagree	Strongly Disagree
Persian	I have a strong accent	Persian	80% - 90%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
English	I have no accent	English	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Arabic	I have a strong accent	Arabic	90% - 100%	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
English	I have no accent	English	90% - 100%	Disagree	Strongly Disagree	Strongly Agree	Strongly Disagree	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Chinese (China)	I have no accent	Chinese (China)	80% - 90%	Strongly Agree	Neutral	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
English	I have no accent	English	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
English	I have no accent	English	90% - 100%	Strongly Agree	Strongly Disagree	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
French	I have no accent	French	60% - 70%	Strongly Agree	Strongly Disagree	Agree	Strongly Disagree	Strongly Agree	Disagree	Strongly Agree	Strongly Disagree	Agree	Strongly Disagree
English	I have a bit of an accent	English	90% - 100%	Strongly Agree	Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Arabic	I have a strong accent	Arabic	90% - 100%	Neutral	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
German	I have a strong accent	French	80% - 90%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Neutral	Strongly Agree	Strongly Disagree	Neutral	Disagree
French	I have no accent	French	70% - 80%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
French	I have no accent	English	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Agree	Strongly Agree	Strongly Disagree	Strongly Disagree	Strongly Disagree
Arabic	I have no accent	Arabic	60% - 70%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
English	I have no accent	English	90% - 100%	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Persian	I have a bit of an accent	Persian	90% - 100%	Strongly Agree	Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
French	I have no accent	French	90% - 100%	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Russian	I have a strong accent	Russian	60% - 70%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Disagree	Strongly Agree	Strongly Disagree
Arabic	I have a strong accent	Arabic	80% - 90%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
French	I have no accent	French	90% - 100%	Agree	Neutral	Strongly Agree	Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
French	I have a strong regional accent	Chinese (China)	50% - 60%	Neutral	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Neutral	Strongly Agree	Strongly Disagree	Neutral	Strongly Disagree
French	I have a strong accent	French	90% - 100%	Strongly Agree	Strongly Disagree	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
German	I have a strong accent	German	70% - 80%	Disagree	Strongly Disagree	Agree	Strongly Disagree	Strongly Agree	Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
French	I have no accent	French	70% - 80%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
English	I have no accent	English	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
English	I have no accent	English	80% - 90%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Persian	I have no accent	Persian	90% - 100%	Strongly Agree	Strongly Disagree	Agree	Strongly Disagree	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Arabic	I have a strong accent	Chinese (China)	60% - 70%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Neutral	Strongly Agree	Strongly Disagree	Agree	Strongly Disagree
Russian	I have a bit of an accent	Russian	90% - 100%	Strongly Agree	Neutral	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
German	I have a bit of an accent	English	60% - 70%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
English	I have a strong accent	English	80% - 90%	Neutral	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Disagree
French	I have no accent	French	80% - 90%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Disagree
Russian	I have a strong accent	Russian	60% - 70%	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Chinese (China)	I have a strong accent	Chinese (China)	80% - 90%	Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
French	I have a strong accent	French	80% - 90%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
French	I have a bit of an accent	French	90% - 100%	Strongly Agree	Neutral	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
French	I have a bit of an accent	French	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
French	I have a strong accent	French	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Persian	I have a bit of an accent	Persian	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Agree	Strongly Disagree
French	I have no accent	French	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
English	I have a bit of an accent	English	80% - 90%	Strongly Agree	Strongly Disagree	Strongly Agree	Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
English	I have a bit of an accent	German	60% - 70%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Agree	Strongly Agree	Strongly Disagree	Strongly Disagree	Strongly Disagree
French	I have a bit of an accent	Russian	50% - 60%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Agree	Strongly Agree	Strongly Disagree	Disagree	Strongly Disagree
German	I have a strong accent	German	60% - 70%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Disagree
Russian	I have a strong accent	Russian	80% - 90%	Disagree	Neutral	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
English	I have a strong regional accent	English	90% - 100%	Neutral	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
English	I have no accent	English	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Russian	I have a strong accent	English	50% - 60%	Strongly Agree	Strongly Disagree	Strongly Agree	Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Agree	Strongly Disagree
English	I have no accent	Chinese (China)	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Agree	Strongly Disagree	Neutral	Strongly Disagree
English	I have no accent	English	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Russian	I have no accent	Russian	70% - 80%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree
Arabic	I have no accent	Arabic	90% - 100%	Strongly Agree	Strongly Disagree	Strongly Agree	Strongly Disagree	Strongly Agree	Disagree	Strongly Agree	Strongly Disagree	Agree	Strongly Disagree
Total Accuracy			81%	**Total System Usability Scale Score**						95.7			