NORTH-WEST UNIVERSITY, VAAL TRIANGLE

Language identification for proper name pronunciation

by

Oluwapelumi Giwa

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy

in the
Faculty of IT and Economic science
School of IT

April 2016

Declaration of authorship

I, OLUWAPELUMI GIWA, declare that this thesis, titled 'LANGUAGE IDENTIFICATION FOR PROPER NAME PRONUNCIATION' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this university.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this university or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly acknowledged.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

"You are always a student, never a master. You have to keep moving forward"

Conrad Hall

NORTH-WEST UNIVERSITY, VAAL TRIANGLE

Abstract

Faculty of IT and Economic science
School of IT

Doctor of Philosophy

by Oluwapelumi Giwa

The ability to predict the pronunciation of proper names is of importance to speech recognition applications that utilise names, such as directory enquiry systems. One of the factors that has been shown to improve the modelling of proper names, is the ability to identify the language of origin of a particular name. Proper names present specific challenges, which typically result in poor language identification (LID) accuracy: they are short, can be spelled in idiosyncratic ways and may have multiple language origins. In South Africa, the difficulty of identifying the language of origin of a name is exacerbated by two factors: co-existence of multiple languages and the scarcity of resources for model training.

In this thesis, we first investigate existing LID approaches applicable to words in isolation, specifically focusing on those techniques that have been identified to produce high accuracy when resources are limited. We assess the strengths and weaknesses of existing LID techniques when applied to generic words and highlight various factors that influence the performance accuracy with which the language of individual words can be classified.

A novel approach to LID of isolated words is then developed using an existing pronunciation modelling technique. Specifically, the LID task is recast as a pronunciation modelling task, and 'joint sequence models' are applied to obtain accurate single-word predictions. We evaluated the algorithm and found that the approach outperformed other conventional LID techniques in terms of identification accuracy, with low training data requirements. The results show that this new approach is able to reach identification accuracies greater than 97% on generic words.

Given that suitable corpora for South African names were not available prior to the study, we developed two corpora as part of this work: the 'Southern African corpus for multilingual name pronunciation' (Multipron corpus) contains names in four languages (Afrikaans, English, Sesotho and isiZulu) as produced by speakers of the particular languages; the 'South African directory enquiry' (SADE) corpus contains a wide variety of names produced in a directory enquiries system, produced by speakers of the same four languages as above. When applying

this technique to the above corpora, one finds that LID of proper names is a difficult task, but identification accuracy of over 80% was still obtained.

In practice, there are cases where words belong to more than one language of origin. This has not been studied extensively (for either generic words or proper names), even though it is of practical importance. We investigate the ability of the proposed technique to perform LID of multilingual words, specifically for under-resourced languages.

This thesis concludes by investigating the implications of LID of proper names for pronunciation prediction by analysing G2P accuracy of dictionaries developed using the auto-generated LID information, as well as the recognition accuracy of an automatic speech recognition system developed using these dictionaries. We define a new G2P performance metric – bilateral V-PA – which deals with variants in a way that is conceptually more consistent than existing performance metrics. We show that the new G2P accuracy measure correlates well with the ASR results observed. Based on an analysis of different approaches to dictionary creation, we provide guidelines for incorporating LID information during pronunciation modelling of proper names.

Keywords: *language identification, joint sequence models, support vector machines, grapheme-to-phoneme, naïve Bayes, SADE, Multipron, classification accuracy*

Acknowledgements

This research was performed in the Multilingual Speech Technologies (MuST) Research Group of the North-West University. It was guided by Prof. Marelie Davel, for the past four years as my PhD advisor, mother and an ideal mentor. I feel extremely privileged to have had the opportunity to work with her over the past four years.

Contents

List of Tables

List of Figures

Abbreviations

ASR	**A**utomatic **S**peech **R**ecognition
CMN	**C**epstral **M**ean **N**ormalisation
CVN	**C**epstral **V**ariance **N**ormalisation
DNN	**D**eep **N**eural **N**etwork
EM	**E**xpectation **M**aximisation
FMLLR	**F**eature-space **M**aximum **L**ikelihood **L**inear **R**egression
FN	**F**alse **N**egative
FP	**F**alse **P**ositive
G2P	**G**rapheme to **P**honeme
GMMs	**G**aussian **M**ixture **M**odels
HMMs	**H**idden **M**arkov **M**odels
JSMs	**J**oint **S**equence **M**odels
LID	**L**inear **D**iscriminant **A**nalysis
LID	**L**anguage **I**dentification
LM	**L**anguage **M**odel
MFCCs	**M**el-**F**requency **C**epstral **C**oefficients
MLE	**M**aximum **L**ikelihood **E**stimate
Multipron	South African Corpus for **Multi**lingual Name **Pron**unciation
MuST	**Mu**ltilingual **S**peech **T**echnologies
MVP	**M**atching **V**ariant **P**ercentage
NB	**N**aïve **B**ayes
NLP	**N**atural **L**anguage **P**rocessing
P2P	**P**honeme to **P**honeme
RBF	**R**adial **B**asis **F**unction
ROC	**R**eceiver **O**perator **C**haracteristic

SADE	**S**outh **A**frican **D**irectory **E**nquiry
S-PA	**S**ingle-best variant G2P **P**hone **A**ccuracy
S-WA	**S**ingle-best variant G2P **W**ord **A**ccuracy
SVM	**S**upport **V**ector **M**achine
T-LID	**T**ext-based **L**anguage **I**dentification
TN	**T**rue **N**egative
TP	**T**rue **P**ositive
V-PA	**V**ariant-based G2P **P**hone **A**ccuracy
V-WA	**V**ariant-based G2P **W**ord **A**ccuracy
WER	**W**ord **E**rror **R**ate

Dedicated to my loving parents, Pastor and Mrs. Giwa, for their prayers, support and encouragement.

Chapter 1

Introduction

For many speech processing applications, it is important to be able to predict the pronunciation of proper names correctly. Both speech recognition systems (such as directory assistance systems) and speech synthesis systems that utilise names (such as audio-book generators or screen readers) require accurate pronunciations of names in order to function. For example, given an Afrikaans name, such as 'Paul' mentioned in an English sentence, 'Paul' as */p @u l/* in Afrikaans will be pronounced completely differently from the English name 'Paul' as */p O: l/* (using SAMPA[1] notation).

In order to improve the performance accuracy of an automatic speech recognition (ASR) system, we require a better understanding of why the accurate pronunciation of proper names is difficult, and specifically, what role the source language of a name plays in this process. Our focus is on defining a Language Identification (LID) technique that is applicable to the source languages of proper names, and analysing the performance of such a technique within the South African context.

1.1 The pronunciation of proper names

This thesis focuses on proper names. Before continuing, we will be making a distinction between common words and proper names. For linguistic terms discussed in this chapter, the standard terminology described in [2] is used. Proper names are entity-specific and refer to the names of people, places or things [3]. Common words represent generic individual text strings that are not entity-specific. Common words sometimes contain substructures, which can be decomposed into parts such as prefix, suffix, unstressed root and stressed root. While proper names also contain such decompositions, units and rules used for pronunciation prediction may be different.

[1] The 'Speech Assessment Methods Phonetic Alphabet' is a standard computer-readable notation for phoneme descriptions. See [1].

Proper names are a large set of words, without a concise pronunciation pattern. Evidence shows that many personal names found in different languages originate from other languages [4], and their pronunciations are influenced by the rules of the original language. For example, over two decades ago, more than 1.5 million family names existed in America that were derived from dozens of languages [4].

As applicable in this chapter and the subsequent ones, we define what the 'source language' of a term is (either generic word or proper names). During the course of the study we realised that 'source language' is subjective. Specifically, with reference to this study, the 'source language' of a term is defined as the most likely language a term originated from - this means that the term was first used and typically follows the spelling system of that specific language [5]. For example, 'John' originated as an English name, even though it may be used in many other language communities. Other examples include 'Pieter' = Afrikaans; 'Rand' = English or Afrikaans; 'Zuma' = isiZulu. In particular, we noticed huge disagreement among language practitioners with regard to source languages of loan terms. To simplify the tasks, the following rules were proposed:

- If a word from language A is incidentally used in language B (for example, English digits in Sepedi), then they should only be tagged as language A (English in this case).
- If a word from language A has become incorporated into language B to the extent that it is now considered part of that language, it should be tagged as A and B.
- If a word from language A has changed its spelling when it was incorporated into language B, it should only be tagged as B.

Detailed examples:

- The English digit 'seven' used in an isiZulu sentence would be tagged as English, not isiZulu.
- The word 'Zulu', which has become a standard word in English, would be tagged as both isiZulu and English.
- The word 'Rand', which originated as an Afrikaans name but has been fully integrated into English, will be tagged as both English and Afrikaans.
- The word 'Zoeloe' which is the Afrikaans version of the word 'Zulu', would only be tagged as 'Afrikaans', not as English or isiZulu as well.
- The name 'Solomon', which originated from Hebrew, would only be tagged as 'English', even though it is used in Sesotho language communities.

- The name 'Phila' which originated as a Greek name but has been fully integrated into English and isiZulu, will be tagged as both English and isiZulu.
- The name 'Zuma' which is an isiZulu name, would only be tagged as 'isiZulu'.

How different speakers handle unknown or unfamiliar proper names can be attributed to various reasons, such as irregularity of word spelling or borrowed words from a different language, which could leads to different pronunciation variations. In such variations, speakers tend to replace unfamiliar phoneme(s) with those they believe are the closest match in their mother tongue [6–8]. Church [9] believes that humans adopt what he calls a pseudo-foreign accent, which speakers in the original tongue use to communicate a word in a foreign language (non-native language) by modifying the parameters of stress rules in a simple manner to produce foreign-sounding outputs.

Previous works affirmed that knowledge of the source language of proper names is important in determining the correct pronunciation of proper names and also increases accuracy in natural speech synthesis [9–11]. Kgampe and Davel [5] demonstrated this using a small set of respondents to show that the linguistic origin of proper names and the mother tongue of the respondent have a significant effect on the pronunciation of names. Similarly, Modipa and Davel [12] showed that prediction of loan words using letter-to-sound rules in the speaker's mother tongue provides suitable results for loan words, with proper names included.

A few factors that have been identified that make the pronunciation of proper names difficult in comparison to generic words include:

- Names can be of very diverse etymological origin and can be borrowed from another language without following the process of assimilation to the phonological pattern of the new language [13].
- There are a great many distinct name types [14];it is not possible to create a dictionary of all possible names.
- Pronunciation of proper names is idiosyncratic, meaning there may be several pronunciations [15].

Current pronunciation prediction techniques typically still rely on a combination of manual and automatic processing [16]. Data-driven methods, such as rule-based methods, can achieve a reasonable level of accuracy in predicting how proper names will be pronounced. These same data-driven techniques become less accurate for pronunciations of words that do not follow the standard pronunciation rules of the language, hence are poor predictors of personal names. It

is expected that better results may be obtained using a more sophisticated modelling approach that uses language of origin as parameter.

Words, phrases and proper names are often used across language boundaries in multilingual settings, especially for minority languages, where *code-switching* with a dominant language can become an intrinsic part of the language itself [17]. Systems such as call routing or voice-driven navigation systems process proper names and foreign words; these tend to have pronunciations that are difficult to predict [18]. Therefore, knowing the language of origin of such words can improve modelling accuracy [11, 19]. As these categories of words (proper names, foreign words) can be important content terms in an utterance [15], there is a need to handle them carefully. To model these categories of words properly through language-specific pronunciation, it becomes necessary to be able to identify the language of origin of a word in *isolation*, that is, a single word from one language may be embedded in a matrix sentence of the second language.

1.2 Language identification of proper names

LID techniques have been applied in different natural language applications, such as machine translation [20], speech synthesis or ASR [21], pronunciation prediction [9, 11], and information extraction applications.

The LID task can be divided into two classes: written (text-based) and spoken LID; that is, LID from text or speech. Text-based LID (T-LID) is a symbolic processing task [22]. This thesis only focuses on T-LID.

T-LID has been carried out using various methods ranging from simple statistical methods to complex pattern-recognition algorithms [23–33]. Approaches include decision trees, which use questions about contexts of words [33], Markov models [24, 34], the combination of linguistic and statistical methods [13, 25], n-gram support vector machine (SVM) classifiers [26, 27, 35], naïve Bayes classifiers [28, 29, 35, 36], neural networks [33], language-based rules [23], the normalized dot product [37] and k-nearest neighbour and relative entropy techniques [38].

Earlier studies on T-LID did not include many of the more modern statistical n-gram modelling techniques, such as n-gram discounting or model pruning that have been important for improving classification accuracy [32, 35]. Different smoothing methods applied in LID include Dunning's add-one smoothing [32], which originated from Laplace's rule of succession, shared back-off techniques [35] and Knesser-Ney interpolation [32].

Much of the research in the T-LID field has been performed on running text (see [36] for an overview), but several studies have focused on identifying the language of origin of short text samples (Section 2.4.2). To the best of our knowledge, limited previous work has focused on

identifying the language of words (proper names or common words) in isolation. Studies that do exist [22, 39, 40], are discussed in the next chapter.

LID of proper names is a challenging task. The biggest challenges to LID of proper names are the following:

- Names tend to be short. Many LID techniques only become highly accurate when applied to longer strings (as many as 15 characters or more) [36].
- Ambiguity in name origin. The same name or name component could have more than one language origin [4].
- Different parts in a name may stem from different origins, for example, names of American Chinese such as "Mary Wang" [41]

The work presented in this thesis focuses on LID for the pronunciation of proper names in isolation.

1.3 Approach

The main objective of this thesis is to determine the most appropriate approach to identify the language of origin of proper names automatically, in such a way that the results are useful for pronunciation prediction. We aim to examine existing statistical LID methods and identify a technique that is applicable to the specific task. In so doing, the study will seek the following:

- To develop a benchmark corpus for proper names in four selected South African languages, namely Afrikaans, isiZulu, Sesotho and English.
- To review existing LID techniques and investigate their performance when applied to a word in isolation. Given the relevance of data sparseness to this task, specific attention will be paid to different smoothing techniques.
- To develop an approach to automatic LID that is applicable to proper names and generalises well, given limited data.
- To evaluate the implications of these technique for the pronunciation prediction of proper names in an under-resourced environment.

1.4 Thesis overview

The thesis is structured as follows:

- In Chapter 2 we provide background information on existing LID techniques used for both generic words and proper names. In this chapter, the focus is on LID of short-text segments.
- In Chapter 3 a benchmark corpus for the analysis of proper name identification and pronunciation modelling is developed. We focus on the design, collection and analysis of the corpus, and highlight the importance of this corpus for further research on understanding multilingual and cross-lingual name pronunciation.
- In Chapter 4 we experiment with LID techniques used for identification of common words (generic words), specifically words in isolation, while describing in detail each technique employed for the work. We compare different classification techniques that have been reported to yield good performance on short text segments by applying them to individual words, and investigate the relationship between factors that affect identification accuracy.
- In Chapter 5 a newly proposed LID technique based on joint sequence models (JSMs) for the identification of isolated words is discussed. We focus on joint sequence models and demonstrate how the LID task can be recast as a pronunciation modelling task.
- In Chapter 6 we apply the new LID method to proper names and analyse performance. We also use this method to create an additional language-tagged corpus. While the corpus in Chapter 3 only included personal names, the new corpus includes various types of proper names that can be found in a directory enquiries application.
- Many proper names are multilingual. In Chapter 7 we investigate how the best-performing T-LID technique can be adapted to perform multilingual word classification.
- In Chapter 8 the implications of the LID results obtained for the pronunciation prediction of proper names are evaluated and the focus is placed on grapheme-to-phoneme (G2P) analysis and ASR recognition accuracy.
- In Chapter 9 the contribution of this work is summarised, and future work and applications are proposed.

1.5 Conclusion

In this brief introduction to the thesis, we discussed the rationale for focussing efforts on obtaining a better understanding of T-LID of proper names, both in developing techniques that can deal with this task, and understanding the implications for the recognition of proper names.

In the next chapter, we present relevant background in support of the work that follows.

Chapter 2

Literature review

2.1 Introduction

This chapter will examine the background information and ideas with regard to research in LID and other topics discussed in subsequent chapters:

- Section 2.2 discusses approaches to proper name pronunciation prediction.
- Section 2.3 provides an overview of T-LID, and discusses different T-LID techniques in relation to short and long text segments.
- Section 2.5 discusses current approaches to LID of proper names.
- Section 2.6 examines a few evaluation techniques that have been used in literature up to date for evaluating LID systems.
- Section 2.7 examines use cases where LID has been applied.

2.2 Proper name pronunciation prediction

Being able to determine the language of origin of proper names is important to any natural language processing (NLP) application. As discussed earlier in Section 1.1, there are a number of factors that make the pronunciation of proper names difficult. Current pronunciation prediction techniques typically still rely on a combination of manual and automatic processing [16]. Data-driven methods, such as G2P rule-based methods, can achieve a reasonable level of accuracy in predicting how proper names will be pronounced. These same data-driven techniques become less accurate for proper name pronunciations, whose orthographic form could be archaic or foreign or do not follow the standard pronunciation rules of the target language.

In order to address the complications associated with proper name pronunciation prediction, various authors [11, 19, 42] propose two lexical modelling approaches that include: (1) G2P conversion based on language-specific rules, (2) phoneme-to-phoneme (P2P) conversion. The language-specific G2P conversion approach makes use of the source language of the proper name in context before applying the language-specific G2P rules to predict its pronunciation. According to [11, 19], knowing the language of origin of proper names can improve their modelling accuracy. Llitjos and Black [11] used a decision tree for G2P conversion. To generate alternative pronunciations, they added multi-phones. In their work they used a classification and regression tree (CART) technique to train a decision tree for each letter to phone map. To predict words' language of origin, they adopt a tri-gram based language model with Laplace smoothing (to assign non-zero probability to out-of-vocabulary words). Language predictions obtained from the LID technique were fielded as a feature to the CART building process. They reported an improved accuracy of 17%. Yang *et al.* [42] approach G2P conversion of proper names using the G2P-P2P approach. Their approach can be subcategorised into three phases: (1) Firstly, the phonemic transcription generated by the G2P, together with the word's orthography, is passed to a language-specific P2P converter. (2) An alignment is performed between the word's initial phonemic transcription and the orthography in order to determine the graphemic context of the P2P converter. (3) Finally, the P2P converter generates alternative variants from learned rules.

In a related work, van den Heuvel *et al.* [43] approached proper name pronunciation prediction using the process of syllable generation. Their technique constitutes two approaches, namely a deductive and an inductive process. Results show that using the deductive approach (identifying syllables, prefixes and suffixes) yielded no significant performance increase when tested on first names. In contrast, they observed improvement in performance when tried on surnames (last names and toponyms).

Réveil *et al.* [19] carried out an important study on how the language of origin of the word affects the ASR performance. Their experiment used a language-specific G2P converter, mono- and multi-lingual acoustic model and language-specific P2P converter. They observed that when pronouncing foreign words, speakers used their own mother tongue language G2P rules rather than the language of origin's specific G2P rules. In order to test their observation, they used the speaker's mother tongue G2P rules to generate pronunciation variants of foreign words. They reported a decrease in performance accuracy of the ASR system and concluded that speakers use the G2P rules of the language of origin of foreign words during pronunciation.

In Chapter 8, we use language-specific G2P trained on generic words to produce pronunciation and also include transcription variants based on the LID output. The inclusion of pronunciation variants of proper names supports the work of van der Heuvel *et al.* [44] and thus the observation that ASR accuracy for proper names can be improved with pronunciation variants.

2.2.1 G2P conversion

An automatic G2P conversion engine uses existing G2P rules to predict the phonemic transcription of words, given their orthographic form. Different data-driven methods exist for G2P conversion, namely pronunciation by analogy (PbA) [45], default and refine (D&R) [46], JSMs [47], instance-based learning [48], decision trees [49], hidden markov models (HMMs) based on Bayesian techniques [50], and neural networks [51].

The subsections below provide a brief introduction to the main G2P conversion methods used for this work. For further details on JSMs, see [47].

2.2.1.1 Joint sequence models

JSMs were defined by Bisani and Ney [47]. Developed for G2P modelling, the technique is built on the concept of a 'graphone', an m-to-n alignment between small sections of graphemes and phonemes that form the basic units for probability modelling. Both the possible alignments and the graphones themselves are estimated through embedded maximization using a training dictionary. The probability of one unit occurring given the other(s) is similarly estimated using the same training data. To predict a phonemic transcription, the most likely graphone sequences are estimated, given the sequence of graphemes that form the orthography of the word.

The JSM technique is reviewed in more detail in Section 5.2.

2.2.1.2 Default&Refine algorithm

Default&Refine (D&R) is a rule-based learning algorithm that uses the language-specific information to construct the most general rule applicable to the language in context. This algorithm generalises well given limited data and good accuracy. D&R uses the reverse rule extraction order for rule ordering during rule extraction process, that is, the first rule extracted is considered last. These extracted rules then constitute the general rules necessary for dictionary generation. With multiple rules extracted, D&R sets a default rule and re-estimate the rule by performing a repeated process using unprocessed samples. For more information, see [46].

In Section 8.2.4, we use this algorithm to generate different pronunciation lexicons.

2.3 Text-based language identification

2.3.1 Text categorisation concept

Over the years, the classification problem has been widely studied among various communities namely information retrieval, data mining and database communities. General classification problems have many real world applications, such as medical imaging [52], optical character recognition in the field of computer vision [53], statistical NLP [54], document classification etc.

One way of grouping classification problems, is to consider 'Any-of' and 'One-of' problems separately [55]. Tasks grouped under 'Any-of' problems involve classification of classes where the object can belong to more than one class simultaneously, a single class or even none. Sometimes, literature refers to this kind of problem as multi-label classification. Classification problems referred to as a 'One-of' constitute classes that are mutually exclusive, where a record is a member of only one class.

A text classification task is regarded as one of the categories of text classification problems. The idea behind text classification can be illustrated as: Given a set of text training samples $D = x_1, x_2, x_3, ..., x_N$, and a set of sample labels $C = c_1, c_2, c_3, ..., c_N$, such that each text sample is associated with a class label, train a text classification model using the given training data that relate underlying features of each sample with its corresponding class label. Therefore, for a given list of unlabelled text instances, use the classification model to predict a class label for each test sample instance.

Text classification has a wide variety of applications, among others e-mail spam filtering [56–58], news categorisation in a hierarchical form [59], document sorting by subject categories [60] and categorisation of document by topics [61].

2.3.2 Language identification

Language classification can be framed as two separate tasks: language identification and language detection. In language detection, the input consists of two parts: text observation and a language claim. Given the input the goal is to validate the language claim, that is, accept or reject the claim. This is a binary classification problem where a threshold based decision logic is employed at the output of the system to accept or reject claims. Figure 2.1 shows a schematic representation of the two categories of language classification system.

LID is the act of predicting the source language in which a whole document or part of a document is written. In order to train an LID system, proper data transformation is required. The fundamental task of data transformation is known as 'text representation', which is a way of

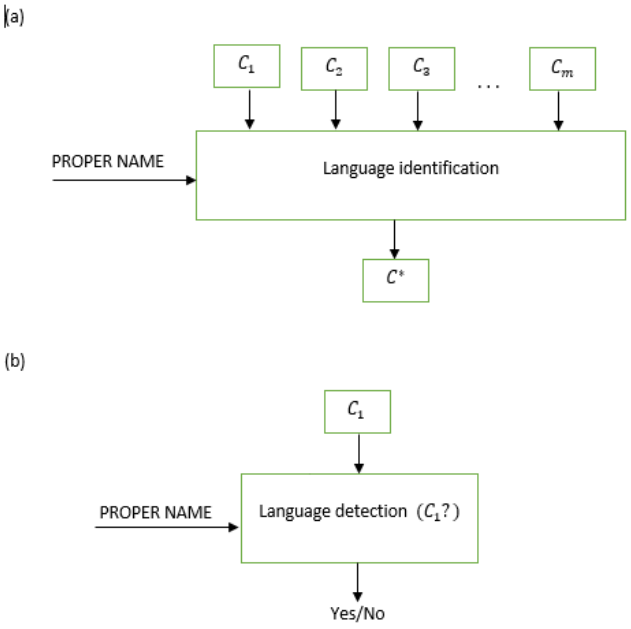

FIGURE 2.1: Block schemas comparing (a) language identification and (b) language detection.

transforming a raw set of data into something suitable the classifier can process. This task is sub-divided into two forms: (1) tokenisation, and (2) feature selection.

Tokenisation [62] as a form of text representation is the act of splitting a continuous stream of text characters into tokens or chunks for possible distinction among applicable languages. The extractable chunks are further categorised as 'character-oriented' or 'word-oriented' in order to create a classification model [62]. Most earlier works in information retrieval [63] and text categorisation [64] applied word-oriented models for text tokenisation. This model employs the term 'bag-of-words', where a document is represented as a distribution of words with their corresponding frequency, such that the arrangement of the word sequence is not important. One major drawback associated with word-oriented models is segmentation of words in languages that do not employ spaces as a delimiter, such as Chinese [65]. However, a few authors reported that this model produces good results when used to discriminate between closely related languages. For example, Tiedemann and Ljubešić [66] applied the bag-of-word model technique to discriminate between the Bosnian, Croatian and Serbian languages. In their work, they used word frequencies, especially those regarded as valid in the target languages, to discriminate among the three plucentric Serbo-Croatian languages. For a similar task, Zampieri [67] distinguished between continental and colonial varieties of languages in French, Spanish and Portuguese.

In recent works, one of the most commonly used models for LID is 'character-oriented'. This model is sometimes referred to as character n-gram models. This is the segmentation of a document into specific character sequences that are adjacent and overlap each other. The n parameter represents the length of the character sequence allowed to be extracted as a single

element in a text string. In this model, each adjacent and overlapping character sequence is counted separately as an individual token. For example, using character n-gram, the text string 'africa' can be represented as:

TABLE 2.1: Different n-gram tokens from word 'africa'.

1-gram	a, f, r, i, c, a
2-gram	af, fr, ri, ic, ca
3-gram	afr, fri, ric, ica
4-gram	afri, fric, rica
5-gram	afric, frica

For performance-related reasons, previous work never pointed to a clear or optimal value for n. Authors such as Grefenstette [68] and Suzuki et al. [69] set the value of n at 3. Takçi and Ekinci [70] and Takçı and Güngör [71] used a value of $n = 1$ and 2 while dismissing the two values as an insufficiently informative parameter value for LID. Other authors, such as [35, 36, 72], experimented with different discrete values of n in the range of 2 to 7, and reported mixed conclusions. Prager [72] reported the best outcome on 4-gram, while Botha and Barnard concluded that 3-gram on SVMs and 6-gram on naïve Bayes gave the best results. (Using larger n-grams with SVMs increased the computational complexity significantly.) Another variation on the above technique is the possibility of using a range of n values, where n-gram features are mixed together in a set. Cavnar and Trenkle [73] experimented with a combination of n values in the range of 1 to 4.

Studies showed that n values of 3 and 4 produce optimal results [35, 36, 72, 74–77]. For South African languages, an example of under-resourced languages, previous researcher observed that $n = 3$ or 4 is a good choice for words in isolation [35]. In a different task, Botha and Barnard [36] reported an optimal value on n equals 3 for SVMs. McNamee and Mayfield [78] found that $n = 4$ was preferable for European languages. According to Lui [62], n value of 3 or 4 is successful because those n parameters correspond to the average morpheme size in a language, thereby capturing language-specific features and characteristics such as prefixes and suffixes. However, there are exceptions to the underlying n values of 3 or 4. Brown [79] reported the highest performance on an n value of 6, with performance reduction for higher n values. His work supported earlier work on discriminating between similar languages, where word-oriented models successful. An n value of 6 generally equates to the average word-length of most languages.

One issue associated with a character-oriented model is data sparsity [80]. (A broad set of languages, especially those with a large variety of symbols, where a large proportion of those characters are present less often, falls in this category.) A benefit of character n-gram (looking at the overlapping and adjacent character sequences) is the provision of linguistically motivated features that may be language-specific. Also, character n-gram models are useful especially for languages without white space as word delimiters. Questions are raised as to whether an

extracted character sequence should span across word spaces for languages that use white space as delimiters. Examples of previous work, such as that of Grefenstette [68] and Brown [79], allow white space as part of a character (as in Table 2.1 shown above), thereby enforcing it with other character string, while Cavnar and Trenkle [73] exclude this extension in their model.

After proper text representation, which consists of a text distribution that spans the entire possible character sequence space, one is faced with a feature selection process. This process involves the exclusion of non-informative features, thereby extracting important informative features that transform features in lower-level to higher-level orthogonal dimensions [81]. The overall concept involves the conversion of character stream into frequency count in the character sequence space, and selects the best k features. Each character sequence, known as character n-gram, embodies the characteristics of each language that need to be learned from the data. In a practical sense, the generated features (character n-gram) are exponentially large and proves computationally complex. In order to reduce the dimensionality of the feature space, one needs to select a subset of sequences that are important based on their frequencies in order to discriminate correctly between languages. For example, Brown [79] used a frequency count sequence to reduce the feature space explicitly, thereby giving rise to a smaller model size and less computational cost. Brown's approach states that if the frequencies of a shorter and long sequence are equal, the shorter character sequence is excluded from the final training set. As noted in [82], excluding less relevant features might not necessarily improve the classification accuracy of a model, even though it is believed that uncommon features contribute less information compared to frequent features [83]. Therefore, in improving classification accuracy, the cumulative effect of features, such as the infrequent ones, can still help; for example, Peng et al. [83] use a statistical language model through a back-off estimator, which explicitly considers all character sequences by measuring their importance as a contribution to the final model.

Examples of known feature selection metrics applied to text categorisation problems include information gain [84, 85] (used, for example, in binary classification to reduce feature space in a naïve Bayes model and decision-tree method), mutual information and χ^2 statistics [86, 87] (used, for example, in a neural network approach to select input features), principal component analysis [86–88], document clustering techniques [89], inductive learning algorithm [90], bi-normal separation [85], the Gini index [91], distance to transition point [92], strong class information words [93].

2.4 Learning algorithms for short and long text

In the previous section, we examined how text can be represented in data and through various techniques to extract features from a sequence set. Similar to the diversity that was explored in feature selection, different algorithms were applied to LID tasks. Over the years, machine

learning algorithms such as SVM [27, 35, 36, 39, 70, 76], neural networks [33, 70, 80], decision trees [22], vector-space models [72, 94], naïve Bayes [22, 32, 35, 36, 68, 76] have proved to be successful techniques for LID tasks.

Most of the above-mentioned techniques, when applied to LID tasks, based their concept on Bayesian inference [62]. Given a document, D, and set of N languages, $L = (l_1, l_2, ..., l_N)$, Bayesian inference computes the posterior probability from two mandatory parameters, namely:

- likelihood estimates of the document, given a language model, $p(D \mid l_i)$, and
- prior probability estimates over the language set, $p(l_i)$.

Authors have also used the uniform prior approach [31, 68, 95, 96] for estimating prior probability. This approach assumes that all languages are equally likely to represent the source language in which a document is written; that is, assigning equal probability value across all languages in the language set. In order to estimate the likelihood, $p(D \mid l_i)$, different approaches have been applied, namely Markov processes [77, 95], naïve Bayes [22, 31, 32, 35, 36, 68, 76], compressive models [96, 97] and neural networks [33].

It remains a challenge to select the best LID algorithm irrespective of the document representation employed. Studies have attempted to compare techniques and arrived at contrasting conclusions. Vojtek and Bieliková [77] compared two LID techniques proposed by Dunning [95] and Teahan [96], based on the Markov process. Their experiments were conducted on a Multilingual Reuters Corpus with eight European languages and novels in Slavic languages. They reported close accuracy for both techniques employed. Baldwin and Lui [76] also compared three LID techniques - naïve Bayes, k-nearest neighbor (k-NN), and SVMs, on three data sets. They reported mixed conclusion based on each data set. On the 'EuroGOV' data set SVMs produces the near-perfect score value, on the 'TCL' data SVMs and the 1-NN model based on skew divergence yielded the best performance, while on 'Wikipedia' data (with large numbers of languages) the 1-NN model cosine-based performed best. Majliš [98] compared five different LID techniques on varied language sizes, and found that SVMs outperformed other techniques. Hakkinen and Tien [22] compared a decision tree and n-gram methods. They concluded that the n-gram based method performed better on longer text samples, while decision trees did better on short words such as proper names. They also emphasised that the decision tree method did well with learning lexical structure information. Mandl *et al.* [99] compared four algorithms (naïve Bayes, vector space models, word-based models, and the out-of-place metric), and reported that the naïve Bayes method gave the lowest error rate against other proposed methods. Similarly, Vatanen *et al.* [32] experimented with two classifiers and smoothing techniques in identifying short text segments. Their reports show that naïve Bayes classification outperformed a ranking

method on sample text length in the range of 5 to 21 characters. To increase identification accuracy they tested different smoothing techniques such as Katz smoothing, absolute discounting, and modified Kneser-Ney discounting. They observed the best result with absolute discounting.

2.4.1 Language identification techniques of long text segments

Differentiating between short and long text segment could invariably be term-subjective. A short text segment could indirectly mean a sentence, phrase or any standalone word (proper name, noun or generic word) in its shortest form in a particular language, characterised by the fact that the text length is very short. Examples of short text samples include mobile text messages that contain up to 160 characters, Operating Systems filenames (up to 255 in length), blog comments, news titles *etc.* In literature, authors categorise short text samples in various ways. Tromp and Pechenizkiy [100] relate short text samples to Twitter messages. Vatanen *et al.* [32] referred to character length in the range of 5 to 21 as short text.

In [65], LID of long text samples is regarded as a solved problem, in which approaches ranging from statistical to pattern recognition algorithms have been applied [23, 27, 28]. When classifying longer text segments, accuracy quickly approaches 100% given enough text; for example, Cavnar *et al.* [73] used rank difference to predict the distance between the most frequent n-gram in the language model and the text document. They extracted their evaluation set from Usenet newsgroup articles written in 14 different languages. They achieved an accuracy of 99.8% on text of 300 characters or more, while retaining the first 400 most common n-grams up to length 5. In a related work, Kruengkrai *et al.* [27] showed a similar result when classifying 17 languages with average length of 50 bytes, while ignoring character-encoding systems during processing (that is, irrespective of the number of characters, 50 bytes of data were used). They achieved an accuracy of 99.7% with an SVM classifier.

Apart from using character n-gram based methods for long text segments, other methods worth mentioning include linguistic models and a compression-based approach for LID. Johnson [101] experimented with stop words obtained from different languages to identify the language of origin of a given document, with longer segments (2 - 4 sentences), obtaining an accuracy approaching 100%. Grefenstette [68] experimented with short words and part-of-speech correlation to classify long text documents that contain sentences with a varied number of words, and reported an accuracy of 100% on sentences with more than 20 words. Giguet [30] proposed a cross-language tokenisation model based on grammatical words that exhibit characteristics relevant to a specific language to discriminate between languages of a given document, and reported an error rate of 0.01% for documents with more than 8 words. Lins and Gonçalves [102] used syntactically-derived closed grammatical classes to identify written words instead of words or letter sequence.

They carried out an experiment on 6 document classes and obtained an accuracy of 99% using well-formatted text data, while observing lower accuracy on HTML text document data.

2.4.2 Language identification techniques of short text segments

Recently, there has been a renewed interest in LID with the focus on short text segments [32, 35, 36, 100, 103–110]. In contrast to LID of long text documents with accuracy approaching 100%, classification of a short textual fragment such as proper names, generic words in isolation and very short sentences (fewer than approximately 15 characters) is a more complex task owing to the lack of contextual information. Different traditional T-LID methods have been applied to short text segments, which are domain-specific, while less effort has been directed at finding an effective technique for LID of short texts irrespective of any domain (microblog messages, queries directed at search engines). Earlier work by [76, 111] shows that not all LID techniques generalise across domains. Vatanen *et al.* [32] used the Cavnar ranking method and a naïve Bayes classifier to identify short text segments. They experimented with 281 languages using a fairly small training set, and for test samples in the range of 5-21 characters, they obtained accuracy of less than 90%. Similarly, Bhargava and Kondrak [39] used SVMs to classify proper names while training on a small data set of 900 names and testing on 100 names. They obtained their best identification rate of 84% using an SVMs with a radial basis function (RBF). Gottron and Lipka [112] compared different n-gram approaches for LID of short and query style text with an average length of 45.1 characters long. They reported a high accuracy value for the naïve Bayes (5-grams) technique over other methods, with 99.4% for short newswire text and 81.6% on single words.

Most recent work has been directed at especially microblog domains [100, 106–108]. Bergsma et al. [106] examined LID on Twitter messages specifically for under-resourced languages, and found that systems trained on out-of-domain data obtained from Wikipedia outperformed other off-the-shelf commercial and academic LID software (TextCat, GoogleCLD, Langid.py). They reported improved performance accuracy using compression-based language models of 97.0% (trained on Wikipedia), 97.4% using maximum entropy classifier (trained solely on Twitter data), 97.9% using compression-based language models of 97.0% (trained on both Wikipedia and Twitter). They also mentioned the factors that contribute to higher performance accuracy, such as training data, length of the tweet and previous information across multiple tweets. In a related work, Carter *et al.* [107] applied a character n-gram distance metric to Twitter messages. Their method incorporated domain-specific information drawn from metadata-related information such as a page link to the tweet message or author. They reported a performance accuracy increase of 3% if the model trained on microblog messages and a further increase in performance when the standard method is augmented with individual prior messages. Also, Tromp and Pechenizkiy [100] used a supervised LID technique based on a graph-based n-gram

structure to identify Twitter messages. Their result showed higher performance accuracy of over 90% for the proposed technique compared to the standard n-gram based approach that never obtained accuracy higher than 90%.

For a different task in the field of search engine domains, Ceylan and Kim [113] applied a decision tree technique based on linguistic features in order to classify search engine queries. Their technique showed an improvement in accuracy from 65.2% to 82.7% when compared with the Cavnar and Trenkle method [73].

As the text becomes shorter, so the task becomes more difficult. All the work discussed above focused more on LID at word level in a short text segment document, while little attention was directed at tagging isolated words without context. LID of isolated words (without context) has been carried out using approaches such as dictionaries, the character n-gram language model, JSMs, SVMs and conditional random fields [35, 104, 109, 114]. In our previous work [35], we compared two techniques (naïve Bayes and SVM) to identify the language of origin of words in isolation. The experiment in the current work incorporates discounting techniques in order to compensate for unseen tokens mostly associated with the general naïve Bayes technique. We found that SVMs (regarded as the state-of-the-art) technique for an LID task across domains outperforms any of the smoothing techniques. In a related work [104], JSMs (a pronunciation modelling technique) was compared with SVMs technique for T-LID of words in isolation. Experiments conducted on four South African languages (Afrikaans, English, Sesotho and iziZulu) reported competitive results. The JSM-based system obtained an F1-measure of 97.2% compared to a state-of-the-art SVM technique with an F1-measure of 95.2%. King and Abney [109] used a weakly supervised approach for identifying the languages of single words in a multilingual document. They experimented with different ranges of data sizes and reported that conditional random fields models trained with generalised expectation outperformed sequence classifiers.

Not all methods can be applied to words in isolation, with linguistic models (such as the stop words used by Johnson [101] or the closed grammatical classes used by Lins and Gonçalves [102]) not being applicable to this task. One technique that is not n-gram based that is worth mentioning, is the use of a data compression model for LID, as introduced by Hategan *et al.* [40]. They evaluated the performance of the algorithm on individual names, isolated words from 6 European languages, and reported an accuracy of above 80% on the two best results.

2.4.3 Factors that influence text-based LID accuracy

Research into LID has identified various key factors that could directly influence T-LID accuracy [36]. These include:

- Size of training data: Identification accuracy is directly affected by various training data sizes [28]. To reduce the risk of over-fitting and over-training the system, it is useful to evaluate methods based on how quickly the model converges, given different sizes of training corpus [38].
- Size of input text: The longer the size of the text used as input, the more reliably identification can be performed [36]. Dunning [95] showed that the performance of a naïve Bayes classifier could be increased from 92% to 99% if the input test length were increased from 20 to 500 characters.
- Effect of n-gram length: Increasing the length of the n-gram directly improves the identification accuracy, given a training corpus of sufficient size. This advantage comes at a cost: an exponential increase in time and memory complexity.
- Effect of classification method used: Some methods train faster compared to others on a lower n-gram, while some do better on a higher n-gram with better identification accuracy.
- Similarities of languages: Languages that fall in the same families of languages tend to be more difficult to distinguish, compared to those that fall outside such families[36].

2.5 Language identification of proper names

From isolated words to proper name identification, task complexity increases. LID of proper names is difficult owing to associated features present in names, such as ambiguity in name origin where the same name or name component could have more than one language of origin. Also, different parts in a name may stem from different language origins. Another notable problem is the shortness feature of proper names, since many LID techniques only become highly accurate when applied to longer string length. With these properties, over the years, little focus has been directed at LID of proper names. LID of proper names has been approached using language models [20], and SVMs [39].

Konstantopoulos [115] examined LID of proper names. He experimented with soccer players' names obtained from 13 languages. He reported an initial average F_1 score of 27% when tested on a general n-gram language model. With more discriminated training data based on short length, an average F_1 score of 50% was obtained on last names and 60% on first names. In related work, Li *et al* [20] used an n-gram language model to identify proper names in English, Chinese and Japanese. They reported an overall accuracy of 94.8% when classifying names among these three languages. (As these three languages are not closely related, the classification task becomes easier, explaining the high accuracy achieved.)

Bhargava and Kondrak [39] experimented with two data corpora, namely Transfermarkt corpus (containing European soccer players' names in 13 possible languages) and the 'Chinese-English-Japanese' (CEJ) corpus containing first and last names. The work used the SVM technique with n-gram counts as features. On the Transfermarkt corpus, they reported a best accuracy of 79.9% and 56.4% on full names and last names respectively. On CEJ corpus, they reported an accuracy of 97.6% across the three languages.

2.6 Evaluation techniques

In this section, we define the performance metrics used in subsequent chapters to evaluate LID accuracy. From a pattern recognition point of view, language identification and language detection utilise different evaluation techniques. For language identification, the typical metric used is the misclassification or error rate, while in language detection (a binary classifier is trained for each language), two types of errors are separately evaluated: false negatives and false positives. False negatives occur when a correct target language is wrongly rejected, and false positives when an erroneous target language is wrongly accepted. As there is a tradeoff between these two types of errors, a result at only one operating point (for one set of parameter choices) does not represent the system's performance adequately. For this reason, the system is compared at many operating points using the Receiver Operating Characteristics curve or the Detection error Trade-Off curve. (See Figure 7.2 in Section 7.5.3 as an example.).

In this work, we mainly analyse the language identification task and report on identification accuracy, which equates to 1 minus the error rate (1 - error rate), expressed as a percentage. However, we also use precision / recall to better analyse the interplay among languages during language identification, even if it is then only for a single threshold. It is only when we address multilingual language identification (in effect a language detection task), that we trade off precision and recall by adjusting a threshold.

Given a classifier and a set of names, each associated with one or more from a predefined set of class labels $\{C_1, ..., C_m\}$ there are four possible outcomes:

- True Positive (TP) - names that are correctly identified as belonging to a specific source language.
- True Negative (TN) - names that are correctly rejected as belonging to a specific source language.
- False Negative (FN) - names that are incorrectly rejected as belonging to a specific source language.

- False Positive (FP) - names that are incorrectly identified as belonging to a specific source language.

Table 2.2 represents a confusion matrix across all classes C

TABLE 2.2: Contingency or confusion matrix table across all class C.

Class C		Reference set	
		Positive	Negative
Test Outcomes	Positive	TP	FP
	Negative	FN	TN

Using the above described outcomes, there are two distinctive ways in which results can be summarised per class:

- Precision, (Pr), is the percentage of correctly labelled instances among all the labelled instances.
- Recall, (Re), is the percentage of the reference language tags that were identified correctly.

For the purpose of clarity, standard precision, recall, and F-measure can be represented as follows using Table 4.1:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.1}$$

$$\text{Recall} = \frac{TP}{TP + FN}. \tag{2.2}$$

The most often reported evaluation metric approach is classification accuracy [116]. Given an evaluation set of names, where each name has a corresponding correct class, C_i, from a predefined set of class labels, $\{C_1, ..., C_m\}$, classification accuracy can be defined as proportion of the correctly labelled names to the total number of names in the testing set. In contrast, the error rate is the proportion of wrongly labelled names of the total number of names in a set (1 - accuracy). This can be represented mathematically as:

$$\text{Classification accuracy} = \frac{TP + TN}{TP + FN + TN + FP} \tag{2.3}$$

$$\text{Error rate} = \frac{FP + FN}{TP + FN + TN + FP}. \tag{2.4}$$

In this work, precision and recall are often used to shed light on LID performance when analysed on a per-language basis, as well as when multilingual tags are allowed (that is, words are allowed

to belong to more than one class). A change in the allowed number of LID tags introduces a trade-off between precision and recall.

Another common performance metric that is reported in literature is called the F-score [117]. This metric, also known as the F1-measure, was initially introduced by van Rijsbergen [118] in information retrieval. The F1-measure is simply the harmonic mean of precision and recall, in which equal importance is placed on both precision and recall. This is represented mathematically as:

$$\text{F1-measure} = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}. \tag{2.5}$$

From here on, we refer to F1-measure as F-measure.

In order to obtain the overall performance of a system based on standard precision and recall, there are two closely related metrics, namely *Micro-Average* and *Macro-Average* [119]. *Macro-Average* is computed by averaging all precision and recall across all m classes, thereby assigning equal weight to each class. *Micro-Average* is computed globally by summing over all the contingency matrices across all m classes and computing precision and recall. Mathematically, we can represent this as:

Micro-Average score:

$$\text{Precision}^{\mu}_{micro} = \frac{\sum_{i=1}^{m} TP_i}{\sum_{i=1}^{m}(TP_i + FP_i)} \tag{2.6}$$

$$\text{Recall}^{\mu}_{micro} = \frac{\sum_{i=1}^{m} TP_i}{\sum_{i=1}^{m}(TP_i + FN_i)} \tag{2.7}$$

Macro-Average score:

$$\text{Precision}_{macro} = \frac{1}{m}\sum_{i=1}^{m}\frac{TP_i}{TP_i + FP_i} \tag{2.8}$$

$$\text{Recall}_{macro} = \frac{1}{m}\sum_{i=1}^{m}\frac{TP_i}{TP_i + FN_i} \tag{2.9}$$

2.6.1 Receiver operator characteristic

In order to visualise and analyse the quality of this technique across different thresholds, we use the receiver operating characteristic (ROC) curve. The ROC provides qualitative analysis, and

is mostly used to analyse the quality of a classifier. When comparing threshold-based techniques we also evaluate the ROC by plotting the true positive rate (TPR) (also known as 'recall', see eq. 2.2) against the true negative rate (TNR) rather than against the false positive rate, as also often done.

As before, the TPR is the ratio of correct language tags that were identified correctly. The TNR is the number of wrong language tags that were tagged as incorrect by the system, divided by the total number of incorrectly identified terms. In other words, TNR is the probability of the rejected language tags that are relevant. In a ROC curve, the further the curve from the diagonal to the upper right-hand side, the better the performance of the system. Data points located far from the centre of the origin on the lower-left hand in the curve depict poor performance.

We can mathematically represent TNR as:

$$\text{TNR} = \frac{TN}{TN + FP}. \tag{2.10}$$

This section addressed performance measures related to LID accuracy specifically. G2P and ASR accuracy measures are introduced in Chapter 8.

2.7 Applications of LID

Over the years, there have been different motivations for automatic LID across different fields of work. In this section, we briefly discuss some of the use cases that motivate automatic LID.

Firstly, using LID for translation is one of the areas cited as motivation [120]. This involves routing a document written in one or multiple languages for translation to the target language. This routing process could involve parsing the document to a transcriber or a machine translation system. An example of a mono-lingual document routing process can be seen in Google chrome web browser. For documents written in multiple languages that need translation, different approaches such as text segmentation [121] and word-level LID [109, 114] have been proposed.

Secondly, automatic LID can be used for linguistic purposes such as corpus development. Authors such as Scannel [122] used LID methods as a tool to create text corpora for under-resourced languages via the internet, while Lewis and Xia [123] used LID techniques as an approach to gather interlinear text from different online linguistic text documents to build an enriched multilingual repository of linguistically analysed data. Resnik [124] showed that LID can be used to minimise false positives and automatically produce a parallel corpus from the web.

Apart from the use case we are interested in (using LID to produce better pronunciation models) other applications like: LID can be used as one of the components to perform multilingual

sentiment analysis; for example, Tromp [125] used LID technique to capture the grammar of different languages with their specific characteristics. LID can be used as a language detector for a text-to-speech conversion engine. Thomas and Liron [126] used the LID system in a message inquiry system to select the appropriate text-to-speech engine for converting text messages to speech output.

2.8 Conclusion

In this chapter, we reviewed prior work on the pronunciation of proper names, as well as applicable LID techniques. The focus was placed on LID of short text segments as an area central to this thesis. LID of proper names as a subset of short text segments has received some attention in recent studies. By examining the available studies, we found that there is a large performance gap between the results achieved on long and short segments, and that the LID of proper names, as evaluated under realistic conditions, is still a challenging task. This is true of well-studied languages but even more relevant for under-resourced languages.

The focus of this thesis is on the construction of an approach to automatic LID that applies to proper names and generalises well, given limited data. The next chapter focuses on accomplishing this task through the effort of creating and preparing a language-labelled dataset.

Chapter 3

Development of benchmark corpus for proper names

3.1 Introduction

This chapter describes the design and development of a benchmark corpus containing proper names from four South African languages. The corpus, referred to from here onwards as the South African multilingual proper names (Multipron) corpus, was produced by speakers whose first language was one of four languages: Afrikaans, isiZulu, English and Sesotho. These languages constitute the main languages spoken in Gauteng and represent a substantial part of the South African linguistic landscape. To build a trusted corpus, we carefully selected a design strategy for pronunciation variability that was influenced by the size of the corpus, speaker and language diversity. Prompt lists and speakers were selected from a local university campus to obtain representative samples of the four languages[1].

Internationally, several long-term efforts aimed at the development of pronunciation models for proper names have been undertaken. A few prominent examples include:

- ONOMASTICA [128], a multi-year, international project that developed pronunciation models for a large number of European names. This project contains a total of 11 languages and approximately 1 million names per language were included.
- Autonomata Spoken Names Corpus [129] - This corpus design contains 3540 unique names of Dutch, French, English, Turkish and Moroccan origin. Proper names, namely street

[1]This work was performed in collaboration with Jan W.F. Thirion, Marelie H. Davel and Etienne Barnard and partially published in [127]. The author of this thesis specifically focussed on all name collection and tagging, audio data collection and analysis.'

names, person names (first names and last names) and city names, were included in the corpus.

Even in the well-studied corpora listed above, however, proper name pronunciation is not seen as a problem that has been solved [130]. Subsequent sections present the design, collection and analysis of the Multipron corpus.

3.2 Corpus design

The overall corpus design is based on the criteria and transcription protocols that were developed for the Autonomata Spoken Name Corpus [129]. We account for some of (a) the particular characteristics of the South African language landscape and (b) the salient factors that have emerged from research on names in multilingual environments.

The most important design choices, and the motivations for them include:

- Speaker and language diversity: The development of the corpus was based on four South African languages (Sesotho, isiZulu, Afrikaans, and English). These languages represent the largest language group in South Africa and commonly spoken in Gauteng, where the collection took place.
- Using name lists of previous and present students at North West University (a large residential university), while verifying that none of the names were inadvertently misspelt.
- Phonetic content: The goal was to understand cross-lingual pronunciation of South African names. Most proper names include first names and last names, where both names come from the same source origin. As reported in [131], phonetic content is of importance to ASR systems. Attention is directed at understanding different speaker variabilities in the corpus. In this work , the aim is to understand the impact inter-speaker and intra-speaker variation has on the ASR performance. Mixing first name and last name from different source origins exhibits both intra-speaker and inter-speaker variation. Intra-speaker variation refers to variations in the speech of the same speaker, while inter-speaker variation refers to variations between speakers. One interesting question is: do speakers change pronunciation when presented with names that come from different source origins? How often do speakers use the language of origin to articulate pronunciation of names when presented with names from a different source origin?

3.3 Target languages

South Africa has eleven official languages, which fall into two language families (Bantu and Germanic). Seven of the nine Bantu languages in this set also cluster into two subfamilies (Nguni and Sotho); languages within these subfamilies are to a greater or lesser extent similar [132]. Since both the speaker language and the language of origin of a spoken name influence the expected pronunciation, complete coverage of all pairs would require an excessive number of sub-corpora. It was, therefore, decided to limit attention to a subset of four languages. These languages, namely isiZulu, Sesotho, English and Afrikaans, are the most common languages in the Gauteng Province where the collection was done. Moreover, these four languages represent a reasonable fraction of the variability that occurs across the major South African languages. For example, isiZulu and Sesotho are, respectively, languages from the Nguni and Sotho subfamilies. English and Afrikaans are by some measures the two languages that are individually the least similar to any other official language in South Africa [132].

We have decided to weigh English more heavily than the other languages in the corpus for two reasons: (a) because it functions as an interlingua in South African commerce, especially in urban settings and (b) to compensate for the well-known dissonance between English orthography and pronunciation. Hence, more samples of English words are likely to be necessary if rules of comparable accuracy are to be derived, and the corpus contains twice as many English names (40%) as any of the other three languages (20% each). The speakers, however, are drawn in equal measure from all four language groups (more details below).

3.4 Selection of names and speakers

In ONOMASTICA, special effort was made to achieve a predetermined balance between frequent and rare words. However, for the languages that were selected, sufficiently diverse corpora are not available to aim for such a balance. Also, given the very limited knowledge that is available on the cross-lingual pronunciation of South African names, we decided that somewhat more prototypical cases would be most useful for the current corpus. It was decided to select names that are in current use: a name list of recent and present students at a large residential university was used as starting point, and first-language speakers of the four target languages were asked to select names that are typically associated with their language. In South Africa, full names are most commonly spoken in the format *Personal name - Family name*. In fact, this format is so common that personal names are known as "first names" and family names as "last names". Thus, separate lists were created for first names and last names in each of the four target languages. From these lists of selected names, a total of 600 first names and 600 last names were drawn in specific language ratios.

3.4.1 Final name-list selection and verification

The initial name-list obtained in Section 3.4 was further analysed. To ensure that the selection of names was valid across the four languages, we employed a crowd-sourcing technique on the university campus. This method involved asking students and staff members who were first-language speakers to tag names with their corresponding language of origin. We created a spreadsheet comprising a name-list and probable language of origin in the rows and columns respectively. Note that languages not present as 'possible language of origin' were added manually at the end of the columns. All volunteers were asked to include their initials under each corresponding language column. In order to ensure that volunteers did not assume or guess during the selection process, they were asked to tag names only when they were certain of the language of origin. This was necessary to allow volunteers to focus on names of which they had prior knowledge in the name-list.

Prior to final name-list annotation, two steps were taken to validate the crowd-sourced list. These steps helped to reduce the final list to a manageable size giving high precision on language of origin. Firstly, we compute a threshold value for each name in the word-list. This threshold value is the minimum number of voting count expected before a word and its associated tagged language could be considered. To obtain an optimised threshold value, a preliminary analysis was carried out on a few name-lists (tagged with their source language(s)) to obtain a suitable threshold value. Proper names with a threshold count lower than the optimised value were discarded in the final name-list.

Secondly, we asked language practitioners who were first-language speakers in any of the four target languages to validate each tagged word. Names in the list tagged with a wrong language of origin were removed.

3.4.2 Speaker selection

The rapid language changes that are occurring in South Africa imply substantial diversity of speakers, even when they nominally share the same first language. It is therefore not feasible to aim for a representative sample of any significant subset of South African society within the small corpus of the study. As a consequence, it was decided to employ students on a university campus for all the recordings. (Fortunately, such students are a highly relevant demographic for the types of practical services that may require multilingual name recognition.)

3.5 Combination of first and last names

A choice had to be made on how pairs of first and last names would be drawn from the languages in the trial, since it is not generally true that the personal and family name of a particular person will originate from the same language. As a very simple model of cross-language name combinations, and to match the bias towards English names, the name combinations were chosen according to the following frequencies:

- 25% of the names are English-English (EE);
- ZZ, SS and AA name pairs contribute 15% each (45% in total); and,
- the combinations EZ, ES, EA, ZE, SE and AE each constitutes 5% of the corpus.

EZ represents English-isiZulu combinations, which denotes English and isiZulu first name and last name respectively, ES represents English-Sesotho, EA represents English-Afrikaans, ZE represents isiZulu-English, SE represents Sesotho-English and AE represents Afrikaans-English. Typical names selected in each of these categories are shown in Table 3.1.

Languages	Example
EE	Catherine Wallace
ZZ	Nolwazi Zilwana
SS	Mosebo Moremedi
AA	Reinhard Viljoen
EZ	Gareth Mbuyisa
SE	Khothatso Wingard

TABLE 3.1: The names in the prompt list are combinations of English, isiZulu, Sesotho and Afrikaans first and last names.

3.6 Extracting prompt lists

A characteristic of the Autonomata corpus that has been very useful in practice is the way in which it separates both speakers and content. That is, the prompted content was separated into a number of different lists, and a distinct subgroup of speakers pronounced each list, thus making it possible to analyse both speaker differences and content-related differences. To obtain this same benefit, the names are split into three separate lists, each containing 200 full names (first name and last name). Each list was recorded by 16 different speakers (4 first-language speakers of each language, of whom 2 were female and 2 male), yielding the corpus speaker design shown in Table 3.2. Speakers did not repeat across lists, and 48 speakers were used in total.

Language	List A		List B		List C		Total
	M	F	M	F	M	F	
isiZulu	2	2	2	2	2	2	12
Sesotho	2	2	2	2	2	2	12
English	2	2	2	2	2	2	12
Afrikaans	2	2	2	2	2	2	12
Total	8	8	8	8	8	8	48

TABLE 3.2: The corpus consisted of three separate lists of 200 full names each; each list was recorded by four first-language speakers of each language, of whom two were female and two male.

3.7 Recording process

For the recording process, mobile telephones running the data-collection application Woefzela [133] were used. The application prompts speakers by displaying the utterance to be spoken (in the format first name last name) on the screen of the mobile telephone, along with controls for starting and stopping the recording of each prompt, as well as repeating unsuccessful prompts. It also monitors the signal-to-noise ratio and duration of each prompt, and requests the speaker to repeat prompts that were apparently not recorded successfully.

Each session started with the signing of a consent form by the speaker, followed by a training session in which speakers were taught to use the controls. During the main recording session, Woefzela kept track of the recording progress, displaying the number of completed and remaining prompts to the speaker. Recordings were done in a quiet office environment, but no special precautions were taken to ensure acoustic quality, since the goal with the corpus was to study the broad phonetic details of name pronunciation, rather than fine acoustic details. All recordings were stored in the default format provided by Woefzela, namely 16-bit single-channel files, sampled at 16 kHz and saved in Microsoft Wav format.

We also collect relevant meta-information from each speaker, including the following:

- Number of years of residence in the current dialect region.
- Primary language(s) at home while growing up.
- Current primary language(s) at home, if different.
- Additional languages that the respondent either understood or spoken.
- Education level (highest qualification).
- Age (in years).
- Gender.

Apart from the pronunciation of each name, no additional name-specific information was captured. This is different from the corpus of [134] where the speaker's familiarity with and knowledge of the language of origin of each name were also captured. It is interesting to consider whether a speaker will pronounce a name differently if it had previously been heard (and repeated from memory) and if the same name was read from a list. Both these scenarios occur in practice (for example, in a directory assistance system), and will require further analysis, to which the current corpus will not provide an exact answer. The overall corpus design is summarised in Table 3.3.

Names	Categories	Personal names: first names and surnames
	Languages	isiZulu, Afrikaans, Sesotho, English
	Distribution	Frequent names, contextually diverse
Respondents	Primary language(s)	isiZulu, Afrikaans, Sesotho and/or English
	Dialect	Gauteng
	Gender	Balanced between male and female
	Age	Adult speakers (>18)
Recordings	Type	Native and non-native pronunciations of personal names
	Style	Read prompts
	Quality	Low noise, natural environment
	Encoding	16 kHz, Microsoft wav files, 16 bit, mono
Prompts	List constitution	200 full names per list (200 first names and 200 last names)
	Language origin	25% EE, 15% AA, 15% SS, 15% ZZ 5% AE, 5% SE, 5% ZS, 5% EA, 5% ES, 5%EZ
Meta-data	Per name	Orthography, name type, name language.
	Per respondent	Primary languages, additional languages, gender, age, dialect grouping, educational level grouping.
	Per recorded name	Name, respondent, pronunciation language, phonemic transcription.
	Phonemic transcription	Phoneme string. Addition of stress, syllable information. Combination of manual and automated transcription and verification.

TABLE 3.3: Overall corpus design; to be read in conjunction with Table 3.2.

3.8 Analysis of spoken prompts

In order to give an indication of some of the language differences that occur in the corpus, Tables 3.4, 3.5 and 3.6 list the ten most common letter unigrams, bigrams and trigrams of the prompted names in each of the four languages. It is obvious that the two Germanic languages are somewhat similar, and that the two languages from the Bantu family also share commonalities (though apparently less so than English and Afrikaans).

To investigate this indication, we also computed cross-entropies between the n-grams that occur in the various languages. Table 3.8 shows those values calculated on the trigrams for all pairs of languages. It is clear that the languages have significantly different trigram entropies, with Sesotho having the lowest entropy and English the highest; the ordering of entropies agrees with the indication of the amount of regularity in names in each of the languages. The cross-entropies for different languages confirm the observations derived from the frequent n-grams above.

isiZulu	Sesotho	English	Afrikaans
a 0.120	o 0.131	e 0.120	e 0.144
i 0.106	a 0.127	a 0.096	a 0.109
n 0.097	e 0.120	n 0.091	n 0.092
e 0.084	m 0.079	r 0.089	r 0.088
l 0.081	t 0.073	l 0.067	i 0.076
o 0.058	l 0.068	i 0.061	l 0.061
m 0.056	s 0.065	o 0.060	t 0.052
u 0.053	h 0.057	s 0.054	s 0.047
s 0.053	i 0.052	t 0.041	o 0.042
h 0.050	k 0.045	h 0.037	h 0.034

TABLE 3.4: The ten most frequent letter unigrams in the corpus, for each language.

isiZulu	Sesotho	English	Afrikaans
an 0.037	mo 0.051	er 0.029	an 0.043
le 0.031	ts 0.039	on 0.024	er 0.034
si 0.030	le 0.032	an 0.023	ie 0.032
la 0.027	ma 0.029	ar 0.020	ri 0.028
ng 0.022	an 0.029	en 0.019	en 0.028
el 0.022	se 0.027	ne 0.016	ar 0.023
ma 0.019	lo 0.022	in 0.016	ma 0.022
nd 0.019	th 0.020	ll 0.015	el 0.021
il 0.018	di 0.020	ri 0.015	li 0.018
zi 0.017	ok 0.019	le 0.015	te 0.018

TABLE 3.5: The ten most frequent letter bigrams in the corpus, for each language.

isiZulu	Sesotho	English	Afrikaans
ile 0.016	ane 0.017	son 0.011	mar 0.015
ele 0.011	ets 0.012	ell 0.007	rie 0.010
and 0.011	tsh 0.011	ers 0.007	ari 0.010
ane 0.010	ots 0.011	ine 0.006	tte 0.009
ela 0.009	mok 0.011	har 0.005	ter 0.008
nga 0.009	ele 0.010	ton 0.005	ett 0.008
lan 0.009	tha 0.010	enn 0.005	lie 0.007
ani 0.009	mot 0.010	nne 0.005	ize 0.007
osi 0.007	tsi 0.010	lin 0.005	lan 0.007
ong 0.007	tse 0.009	ill 0.004	van 0.007

TABLE 3.6: The ten most frequent letter trigrams in the corpus, for each language.

	isiZulu	Sesotho	English	Afrikaans
isiZulu	-6.012	-7.401	-7.754	-7.782
Sesotho	-7.378	-5.944	-7.816	-7.913
English	-7.889	-7.954	-6.660	-7.474
Afrikaans	-7.885	-7.985	-7.373	-6.261

TABLE 3.7: Cross-entropies for all language pairs, as computed from letter trigram statistics.

These results are all based on the name orthographies - it would, of course, be interesting to compute measures of similarity for the spoken forms as well. As an initial step in this direction, we have simply phonetised each of the words in the corpus, using the pronunciation rules that were derived for generic words in the appropriate language [135]. For the reasons discussed in Section 1.1, these rules are not expected to be accurate for name pronunciations, but they do at least give an indication of how the languages compare with one another. As shown in Table 3.8, these (cross-)entropies are quite similar to those computed for the letter n-grams, except for an apparently increased shift in the difference between the two Germanic languages in Table 3.8. However, that increase is somewhat misleading: the cross-entropy measure is based on a binary distinction (either two phonemes are the same or they are different), whereas phonemes can actually be more or less similar. Thus, the vowel shifts that are quite common in Germanic languages are exaggerated by this measure - for the Bantu languages, which use a smaller set of vowels, this phenomenon is less important.

	isiZulu	Sesotho	English	Afrikaans
isiZulu	-6.068	-7.545	-7.876	-7.777
Sesotho	-7.554	-6.170	-7.957	-7.894
English	-7.923	-7.965	-6.667	-7.736
Afrikaans	-7.824	-7.928	-7.776	-6.283

TABLE 3.8: Cross-entropies for all language pairs, as computed from triphone statistics.

3.9 Conclusion

We described the development of a multilingual corpus that is specifically aimed at containing samples of the diverse proper names that occur in South Africa. This corpus is freely distributed under an open content license[2]. This corpus was the first of its kind in South Africa. Particularly, this corpus captures the balance between own-language pronunciations and imitation of foreign-language pronunciation, as well as the different styles of imitation that occur, which are also of broader interest.

For this thesis, Chapter 8 uses the final name-list discussed in Section 3.4.1 to verify the implications of LID for proper name prediction. The corpus was also used by [127] and developed

[2] http://rma.nwu.ac.za/index.php/south-african-multilingual-proper-names-multipron-corpus.html

further, as discussed in Chapter 8. Recently, a new corpus (the SADE corpus) that also includes multilingual proper names was collected specifically for a directory enquiry system in South Africa [136]. This corpus is introduced in more detail in Section 6.3.

As a contribution for under-resourced languages, we demonstrated the usefulness of annotating name-lists using a crowd-sourcing method. Also, in relation to comparing regularities across widely different corpora (such as the Autonomata corpus, which was collected in Europe, and other corpora from different parts of the world), this corpus should show how to gain better understanding of the parameters that influence the various approaches to proper name pronunciation. It is likely that the corpus will be useful in other environments where several languages are spoken in the same community, or in closely interacting communities and will provide empirical language data for further research work.

Before evaluating how well LID techniques perform on the new Multipron corpus, we first experiment with generic words in isolation. This is the topic of the next chapter.

Chapter 4

Language identification of generic words

4.1 Introduction

In this chapter, previous work done on LID of general text is extended to classification of individual words in isolation. The aim is to compare different classification techniques that have been reported to have good performance on short text segments by applying them to individual words and by analysing the effect of smoothing. We adopt character n-gram models; such techniques have demonstrated good performance over a variety of applications. Furthermore in this context, we investigate the relationship between word length, n-gram length and performance accuracy (as the factors that influence LID accuracy) of each classification technique employed.

4.2 Naive Bayes classification

Naïve Bayes classification is based on applying Bayes' theorem [137] with naïve independence assumptions. In each class, the distribution of the words is model by a classifier using a probabilistic model with independence assumptions concerning different n-gram features about the distributions. Also, naïve Bayes models use the method of maximum likelihood for parameter estimation.

Naïve Bayes classification is commonly used with two variant models (the multinomial model and multivariate Bernoulli - known as the binary independence model), where the goal of both models are to compute the posterior probability of a language class, based on n-gram feature distribution in a word. These models employ the assumption referred to as 'bag of words' while discarding the feature position in the word. Other variants of naïve Bayes include: the Poison

model and the negative binary independence model [138]. Below is the major difference between multinomial and multivariate models:

- Multivariate Bernoulli model - This model represents features in a word. It uses the presence or absence of an n-gram token. This model assumes n-gram in the word to be binary, where a binary value indicates the presence or absence of an n-gram in a word.
- Multinomial model - This model uses n-gram frequencies as features in a word. It allows word in each class to be modelled as samples drawn from a multinomial n-gram distribution. Thus, the conditional probability of a word, given a particular language class is the product of each n-gram observed in the corresponding language class.

More information on the differences between the models mentioned above may be found in [139]. Previous works show that the multinomial model outperforms other naïve Bayes variants [138, 139]. Following the derivation and discussion in Section 4.2.1, we used mathematical representations from [83, 140, 141].

4.2.1 Multinomial naïve Bayes models

The techniques of this model represent a word as a set of n-grams with their associated frequencies (number of times an n-gram occurs in a word). As earlier stated, the objective is to compute the most probable class, k_i, given feature vector x, where $k_i \exists K$. K represents the set of all possible langauge class labels.

In the context of this work, let S be a set of training examples and let each sample be represented by n feature vectors, X $= x_1, x_2, ..., x_n$, with their class labels. Let there be m classes: $k_1, k_2, ..., k_m$. To predict, a sample X is selected to belong to class k_i, if and only if:

$$P(k_i \mid X) > P(k_j \mid X); for\ 1 \leq j \leq m;\ j \neq i \tag{4.1}$$

where $P(k_i \mid X)$ is the conditional probability of a class k_i given a sample X, and $P(k_j \mid X)$ is the conditional probability of a class k_j given a sample X. Thus, $P(k_i \mid X)$, is the conditional probability that a word belongs to class k_i given that we know the feature vector representing the word, X.

A simple application of Bayes' rule for text classification decomposes posterior probability into computation of prior probability $P(k_i)$ and likelihood $P(X \mid k_i)$, that is:

$$P(k_i \mid X) = \frac{P(X \mid k_i)P(k_i)}{P(X)} \tag{4.2}$$

where $P(X \mid k_i)$ represents the likelihood of a sample X belonging to class k_i, and class a priori probability $P(k_i)$ represents the class count relative frequency in the sample set. The intuition behind multiplying likelihood by the class a priori is to give high probability to more common outcomes, and low probability to less common outcomes.

To simplify the computation of $P(X \mid k_i)$ because a long list of parameters is required to describe the multi-dimensional distribution in Equation 4.2, naïve Bayes assumes statistical independence of features, often called *class conditional independence*. That is, for $i \neq j$, where x_i and x_j are conditionally independent given class label, k_i. Under this simplifying assumption, likelihood, $P(X \mid k_i)$, is calculated as:

$$P(X \mid k_i) = \prod_{j=1}^{N} P(x_j \mid k_i) \tag{4.3}$$

This simplifying assumption allows easy computation of the likelihood $P(X \mid k_i)$ with fewer parameters. This assumption simplifies Equation 4.2 computationally to:

$$P(k_i \mid X) = P(k_i) \times \frac{\prod_{j=1}^{N} P(x_j \mid k_i)}{P(X)} \tag{4.4}$$

where class k_i is selected such that $\prod_j P(x_j \mid k_i)P(k_i)$ is optimised, where $P(x_j|k_i)$ is then the likelihood of a specific n-gram feature being observed in a given class, and the word being classified consists of j n-grams.

To compute the class with the highest posterior probability from Equation 4.4 using the maximum a posteriori (MAP) classifier, we maximise the posterior $P(k_i \mid X)$:

$$K^* = \arg\max_{i \in M} \{P(k_i \mid X)\} \tag{4.5}$$

$$= \arg\max_{i \in M} \left\{ P(k_i) \times \frac{\prod_{j=1}^{N} P(x_j \mid k_i)}{P(X)} \right\} \tag{4.6}$$

$$= \arg\max_{i \in M} \left\{ P(k_i) \times \prod_{j=1}^{N} P(x_j \mid k_i) \right\} \tag{4.7}$$

where Equation 4.7 predicts the class with the highest probability using the naïve Bayes assumption as described above. Equation 4.6 to Equation 4.7 are possible because $P(X)$ stays the same for all classes. One can represent N feature vectors in Equation 4.6 as:

$$K^* \quad = \arg\max_{i \in m} \ \{P(k_i) \times P(x_1, x_2, .., x_n \mid k_i)\}\,. \tag{4.8}$$

To avoid floating point underflow when computing conditional probabilities, it is advisable to perform computations in logarithms of probabilities, which result in additions of probabilities. Equation 4.8 can be rewritten by taking logarithm:

$$log\ p(k_i \mid X) = [log\ p(k_i + \sum_{j=1}^{N} log\ p(x_j \mid k_i)] - log\ p(x). \tag{4.9}$$

Equations 4.10 and 4.11 show how to estimate the likelihood, $p(x_1, x_2, .., x_n \mid k_i)$, and the prior probability $p(k_i)$ used in determining posterior probability, $p(k_i \mid X)$ is:

$$P(k_i) = \frac{\text{no of } n\text{-grams feature from } k_i}{\sum_{j=1}^{m} \text{no of } n\text{-grams feature from } k_j} \tag{4.10}$$

$$P(x_j \mid k_i) = \frac{\text{count } (x_j, k_i)}{\sum_{x_j \in M} \text{count } (x_j, k_i)} \tag{4.11}$$

In Equation 4.11, likelihood is the number of times the n-gram feature, x_j, appears among all n-grams under class, k_i, in the training set. Note that in Equation 4.11 a zero probability value can occur when one encounters a new attribute that has not been observed in the training corpus, that is, $count(x_j, k_i)$ equals 0. In practice, a typical approach to avoid zero probability is using Laplace smoothing:

$$P(x_j \mid k_i) = \frac{\text{count } (x_j, k_i + a_j)}{\sum_{x_j \in M} \text{count}(x_j, k_i + a)} \tag{4.12}$$

where a_j represents an appropriate constant, usually 1, and a represents $\sum_j a_j$. There are some advanced smoothing techniques that have been used to improve naïve Bayes classifiers other than Laplace smoothing, producing effective text classifiers [142, 143]. The next section discusses three statistical n-gram language models, which we later use to augment naïve Bayes technique.

4.3 Statistical n-gram language modelling

Speech recognition systems are one of the applications where a language modelling (LM) technique is widely used. Previous works show that statistical language models are applied in various other natural language application areas, such as machine translation, text-to-speech systems and information retrieval [144, 145]. This section is based on derivation and discussion from [83, 146].

The goal of LM is to build models that can predict the probability distribution over word strings, thus, a model that can re-distribute probability density among tokens by putting low probability on rare or unseen tokens and high probability on tokens [147] occurring more often. The quality of a statistical language model can be measured based on the entropy or perplexity of the corpus. A good language model should yield a small perplexity value, that is, the lower the perplexity value, the better the model.

$$Perplexity = \sqrt[T]{1/P(x_1 \dots x_T)} \tag{4.13}$$

$$Entropy = log_2 \text{ Perplexity} \tag{4.14}$$

Due to data sparsity (where possible word sequences are not observed) in the training set, the most successful way of building a language model is to use n-gram model. In an n-gram model, the probability $P(x_1 x_2 ... x_N)$ of observing any word sequence can be depicted using the chain rule of probability as:

$$P(x_1 x_2 ... x_N) \; = \; \prod_{i=1}^{N} P(x_i \mid x_1 ... x_{i-1}). \tag{4.15}$$

In Equation 4.15, the n-gram model approximates the left part of the Equation by adopting the Markov n-gram independence assumption, which states that the probability of a word depends only on the previous words. Thus, the only words relevant to predict the conditional probability are dependent on the context history of the preceding $n-1$ words.

$$P(x_i \mid x_1 ... x_{i-1}) \; = \; P(x_i \mid x_{i-n+1} ... x_{i-1}) \tag{4.16}$$

Equation 4.17 shows the conditional probability of the preceding $n-1$ words as estimated from n-gram MLEs (frequency counts):

$$P(x_i \mid x_{1-n+1}...x_{i-1}) = \frac{count(x_{i-n+1}...x_i)}{count(x_{i-n+1}...x_{i-1})} \tag{4.17}$$

MLE can be used by naïve Bayes classifiers to estimate class probability. Smoothing probability can help to address poor probability estimates of MLE, which result in zero probability of missing n-gram sequence models.

Different advanced smoothing techniques have been proposed and applied to the T-LID task [147], such as Katz smoothing [148], Witten-Bell smoothing [149], absolute discounting [150], Kneser-Ney discounting [150] and Jelinek-Mercer [151] methods. We exclude modified Kneser-Ney (regarded as a state-of-the-art smoothing technique) from further experiments due to the small vocabulary size of this task, since modified Kneser-Ney assumes a larger vocabulary size [147]. Rather, this work focuses on three (Katz, Witten-Bell and absolute discounting) smoothing techniques.

4.3.1 Katz backoff with Good-Turing discounting

In the speech recognition domain, Katz smoothing is a widely used smoothing technique [147]. It uses Good-Turing discounting to calculate adjusted counts, $count^*$, which determine how much probability density goes to unseen n-grams.

The intuition behind Good-Turing discounting is to estimate the probability of items-we-saw c times with the MLE probability of items-we-saw $c+1$ times in the corpus. The concept is based on replacing items-we-saw c times (MLE counts) for frequency of frequency of items-we-saw c times (N_c) with a smoothed count, $count^*$. $count^*$ is estimated as a function of N_{c+1}:

$$count^* = (c+1)\frac{N_{c+1}}{N_c} \tag{4.18}$$

where c represents frequency of n-grams.

In order to estimate items-we-saw zero times, we use:

$$P^*_{GT} = \frac{N_1}{N} \tag{4.19}$$

where N_1 represents frequency of frequency of items-we-saw ones, N represents the total number of items-we-saw in the training corpus.

Back-off is used whenever higher-order n-grams are unavailable by backing off to lower-order n-grams. Katz smoothing can be represented as:

$$P^*(x_i \mid x^{i-1}_{i-N+1}) = \frac{count^*(x^i_{i-N+1})}{\sum_{x_i} count^*(x^i_{i-N+1})} \tag{4.20}$$

where $count(x)$ counts how many times x appears in the training set, x_i represents the position of the i^{th} character in the given context, N is the n-gram parameter, and $count^*$ is an adjusted count, with:

$$\begin{aligned} count^*(x_{i-N+1}^i) &= d_i count(x_{i-N+1}^i) \text{ if } count(x_{i-N+1}^i) > 0 \\ &= \alpha(x_{i-N+1}) P(x_i) \text{ otherwise} \end{aligned} \tag{4.21}$$

where α represents a back-off weight and d_i a discount ratio according to the Good-Turing estimate. This distributes leftover probability density to lower-order n-grams.

4.3.2 Witten-Bell discounting + interpolation

Witten-Bell discounting defines models recursively in terms of linear interpolation between the n^{th} and $(n-1)^{th}$ order maximum likelihood models. The discounted probability density is evenly distributed in the training set among previously unseen words with the same history. This can be represented as:

$$\begin{aligned} &P_{WB}(x_i | x_{i-N+1}^{i-1}) = \\ &\lambda_{x_{i-N+1}^{i-1}} \, P_{MLE}(x_i \mid x_{i-N+1}^{i-1}) \\ &+ 1 - \lambda_{x_{i-N+1}^{i-1}} \, P_{WB}(x_i \mid x_{i-N+2}^{i-1}) \end{aligned} \tag{4.22}$$

where x_i represents the position of the i^{th} character in the given context, $\lambda_{x_{i-N+1}^{i-1}}$ is the discounted probability density and $1 - \lambda_{x_{i-N+1}^{i-1}}$ is the probability mass that needs to be distributed evenly to previously unseen types.

4.3.3 Absolute discounting + interpolation

Absolute discounting uses a fixed discounting parameter, D, to reduce probability mass of seen types by subtracting a fixed value. Absolute discounting interpolates higher and lower-order n-grams by using information from lower-order n-gram models. For each seen type one subtracts any fixed value between 0 and 1 from the higher-order n-gram. The estimated leftover probability mass is assigned to lower-order n-grams.

$$\begin{aligned} P_{abs}(x_i \mid x_{i-N+1}^{i-1}) = &\frac{max(count(x_{i-N+1}^i) \; - \; D, 0)}{\sum_{x_i} \; count(x_{i-N+1}^i)} \\ &+ (1 - \lambda_{x_{i-N+1}^{i-1}}) P_{abs}(x_i \mid x_{i-N+2}^{i-1}) \end{aligned} \tag{4.23}$$

where x_i represents the position of the i^{th} character in the given context, D is the discount weight, and λ is a normalising constant (that is, probability mass that has been discounted).

In this section, we discussed techniques used to estimate the probability of seeing a specific n-gram, that is, a way to model the n-grams distribution. The next section discusses how to augment a standard naïve Bayes classifier with statistical n-gram language models to improve identification accuracy.

4.4 Applying n-gram smoothing techniques to language identification

In general, language models such as text classifiers attempt to distinguish documents by identifying different attributes. These attributes may include n-grams, word length, dictionary terms and semantic meanings. To apply n-gram smoothing techniques to LID, a language model is first created per language from training data. Then, an input text string is passed to each language model to predict its language origin. The best-fitting model across all languages that gives the lowest perplexity value is selected, as explained earlier in Section 4.3. Note that each input text string is disintegrated into n-gram attributes. This section's mathematical representations are based on the derivation and discussion in [83, 146].

One can use the naïve Bayes model to explain the n-gram language model concept of text classification. In naïve Bayes, one categorises a word accordingly:

$$k = \arg\max_{j \in m} P(k_j \mid w) \tag{4.24}$$

Using Bayes' rule, Equation 4.24 can be rewritten as:

$$k = \arg\max_{j \in m} P(k_j) P(w \mid k_j) \tag{4.25}$$

$$k = \arg\max_{j \in m} P(k_j) \prod_{i=1}^{N} P(x_i \mid x_1 ... x_{i-1}, k_j) \tag{4.26}$$

$$k = \arg\max_{j \in m} P(k_j) \prod_{i=1}^{N} P_{k_j}(x_i \mid x_1 ... x_{i-1}) \tag{4.27}$$

where, $P(w \mid k_j)$ is the likelihood of a word w given class k_j. This likelihood can be estimated by using the n-gram language model. One correlation between naïve Bayes and the language model is that likelihood is related to entropy or perplexity. In the current tasks, $P_{k_j}(x_i \mid x_1 ... x_{i-1})$ can be estimated using any of the smoothing techniques discussed above in Section 4.3.

The advantages of using a naïve Bayes language modelling-based approach over standard naïve Bayes classifiers include:

- For larger n (modelling of longer context), and in the presence of a sparse data set, n-gram language models allow for experimenting with more advance discounting techniques to improve identification accuracy.
- Markov dependence between adjacent attributes is considered in an n-gram language modelling based approach, which enhances performance over the standard naïve Bayes approach that considers attributes to be independent of one another, given the class.
- Finally, the n-gram language model incorporates all possible attributes (n-gram) and handles the problem of over-fitting associated with feature explosion.

4.5 SVM classification

SVMs are arguably one of the most successful classification techniques in machine learning [152]. This classifier was first introduced for separable data sets by Vapnik [153]. The SVM classifier is widely used because of its high-performance accuracy, flexibility with various data sources and ability to model high-dimensional data [154]. SVMs, which are a family of the *kernel methods* [155, 156], can produce non-linear decision boundaries out of a linear classifier by mapping data from the original feature space to some higher-dimensional feature space.

To obtain good performance accuracy, SVMs require understanding of how they work. This section gives a brief introduction to what SVMs are and the parameter choices that must be taken into consideration when training SVMs. A more extensive and detailed explanation can be found in [157]. This section and its subsections use discussions and derivations from [157, 158].

4.5.1 Linear classifiers, separable and inseparable data

SVMs are an example of binary classifiers. From here on, we use the term *linearly separable* to denote data that allows linear decision boundary separation between the binary samples.

Prior to explaining the concept of large-margin classification for separable data, one must define what linear classifiers are.

A dot product between two vectors is a fundamental concept used to define a linear classifier, sometimes referred to as an *scalar product* or *inner product*. A dot product can be represented mathematically as:

$$w^T x = \sum_i w_i x_i. \tag{4.28}$$

Mathematical representation of a linear classifier based on a linear *discriminant function* is:

$$f(x) = w^T x + b = 0 \tag{4.29}$$

where x represents a vector with components, w represents a *weight vector*, b represents the bias. In this scenario $b = 0$ represents the hyperplane $w^T x = 0$. In Equation 4.29, the sign of the discriminant function, $f(x)$, allows the hyperplane to divide the data into two linearly separated data.

For a dataset to be linearly separable, an SVMs constructs a linear hyperplane, which separates two classes while maximising the distance from the hyperplane to the class samples. Considering the class samples, the distance from the closest data point to the hyperplane is referred to as the *margin*.

Using geometric considerations, the hyperplane margin defined by w with respect to a dataset D can be represented as:

$$m_D(w) = \frac{1}{2}\hat{w}^T(x_+ - x_-) \tag{4.30}$$

where $\hat{w}$ is a unit vector in the direction of w, and data point x_+ and x_- are the closest points to the hyperplane among the positive and negative samples.

Taking any data point on the plane that satisfies the Equation of the plane:

$$\begin{aligned} w^T x' + b &= 0 \\ w^T x'' + b &= 0. \end{aligned} \tag{4.31}$$

The differences between the two data points can be represented as:

$$w^T(x' - x'') = 0 \tag{4.32}$$

where vector w is orthogonal to $(x' - x'')$ as a vector. Therefore, vector w is orthogonal to every vector on the hyperplane, hence, it is orthogonal to the hyperplane.

To calculate the distance between data point x and the plane, take any generic data point, x_i on the hyperplane and project $x - x_i$ on vector w. This can be represented as:

$$\text{distance} = \left|\hat{w}^T(x - x_i)\right|. \tag{4.33}$$

where $\hat{w}$ is:

$$\hat{w} = \frac{w}{\|w\|}. \tag{4.34}$$

The absolute value is used because of the unknown direction of the weight vector w.

Replacing $\hat{w}$ in Equation 4.33 with Equation 4.34, gives:

$$\text{distance} = \frac{1}{\|w\|}\left|w^T x - w^T x_i\right| \tag{4.35}$$

Replacing missing values in Equation 4.35 with the initial assumption of bias, b, then:

$$\text{distance} = \frac{1}{\|w\|}\left|w^T x + b - w^T x_i - b\right| \tag{4.36}$$

where $w^T x_i - b$ is the value of the Equation of the plane for a point on the plane.

Moreover, to find a hyperplane that maximises the margin to all data points by solving for:

$$\begin{aligned} &\max \frac{1}{\|w\|} \\ &\text{subject to: } k_i(w^T x_i + b) \geq 1, \forall i, i = 1, \text{.....}, \text{S} \end{aligned} \tag{4.37}$$

where w represents a weight vector normal to the hyperplane, x_i is the data sample, k_i is the class label (+1 or -1), and S represents the number of training samples.

Equation 4.36 gives:

$$\text{distance} = \frac{1}{\|w\|}. \tag{4.38}$$

Maximising the margin helps minimise probabilistic error on unseen data, hence minimising $\|w\|$. Maximising the margin in Equation 4.38 is equivalent to minimising $\|w\|^2$. This results in a constrained optimisation problem:

$$\begin{aligned} &\text{mininmise } \frac{1}{2}\|w\|^2 \\ &\text{subject to: } k_i(w^T x_i + b) \geq 1, \forall i, i = 1, \text{.....}, \text{S}. \end{aligned} \tag{4.39}$$

Since it is assumed that data are linearly separable, the constraints in Equation 4.39 ensure that examples are correctly classified based on the maximum margin classifier. In reality, most often, data are not linearly separable. Knowing that, a larger margin allows classifiers to misclassify data points. To allow variable errors, we introduce *slack variables*, ξ_i, that permit margin errors in the range $1 \geq \xi_i \geq 0$.

To penalise misclassification and margin errors, we augment Equation 4.39 with the term $C \sum_i \xi_i$. This optimsation problem is represented as:

$$\begin{aligned} &\text{mininmise } \frac{1}{2}\|w\|^2 \ + C\sum_i \xi_i \\ &\text{subject to: } k_i(w^T x_i + b) \geq 1 - \xi_i, \xi_i \geq 0 \end{aligned} \tag{4.40}$$

From Equation 4.40, it is evident that constant $C > 0$ controls the importance of margin maximisation, thereby penalising class misclassification in the training data. Parameter C is refered to as the *cost parameter*. Larger C values result in large weight on misclassifications. Equation 4.40 is known as the *soft-margin SVMs*, which was first introduced by Cortes and Vapnik [159].

Using the Lagrange multipliers method, one can reformulate Equation 4.40 to obtain *dual* formulation to form the Lagrange function, in which the constraints are multiplied by Lagrange multipliers α_i and subtracted from the original objective function [156, 159, 160]:

$$\begin{aligned} &\text{mininmise}_a \sum_{i=1}^{S} \alpha_i - \frac{1}{2}\sum_{i=1}^{S}\sum_{j=1}^{S} k_i k_j \alpha_i \alpha_j x_i^T x_j \\ &\text{subject to: } \sum_{i=1}^{S} k_i \alpha_i = 0, \quad 0 \leq \alpha_i \leq C \end{aligned} \tag{4.41}$$

Weight vector w in terms of the input samples is:

$$w = \sum_{i=1}^{S} k_i \alpha_i x_i \tag{4.42}$$

4.5.2 Non-linear SVM

Up to this point, the focus has been on linear SVMs. This section gives a brief overview of non-linear SVMs. The nonlinear boundary arises from the use of a $kernel function$, to map input vectors into high-dimensional feature space, F. In the search space, F, we sought an

optimal hyperplane that could translate to a linear decision boundary from a non-linear decision boundary. Kernel function is defined as:

$$kernel(x, x') = \phi(x)^T \phi(x') \tag{4.43}$$

where $kernel(x, x')$ is the *kernel function* and ϕ is the mapping. Equation 4.42 above shows that a kernel can be computed with only the associated inner product without calculating the mapping ϕ.

In the subsequent experiment, we experiment with RBF kernels as a non-linear classifier. RBF kernels allow SVMs to produce non-linear decision boundaries [161, 162]. A RBF kernel can be represented as:

$$kernel(x_i, x_j)_{RBF} = \exp^{-\gamma \|x_i - x_j\|^2} \tag{4.44}$$

where γ for the binary SVM is expressed as $\frac{1}{2\sigma^2}$. For a more detailed explanation of non-linear SVMs and their parameters, interested readers are referred to [157].

It turns out that RBF kernels depend on C and γ for good performance. The effect of different values of C was described in Section 4.5.1. As for the γ parameter, larger values increase the flexibility of the decision boundaries, thus increasing its curvature. This also leads to overfitting of the training data. With small values, the decision boundary is close to linear.

4.5.3 Multiclass classification of SVMs

As discussed earlier in Sections 4.5.1 and 4.5.2, the focus has been on two class labels. SVMs can be used for multiclass classification. There are different techniques that help classify multiple classes. These techniques vary from popularity to complexity. The two most common techniques used for multiclass classification are *one-versus-all* and *one-versus-one*.

Given m classes, the *one-versus-all* technique trains a model by constructing a hyperplane per class. For prediction, all class models are combined for multiclass classification, in which the class with the largest margin is selected as the classification label. Mathematically, this can be represented as:

$$\begin{aligned} \hat{k} &= \arg\max_{j=1,\ldots m} g^j(x) \\ \text{where } g^j(x) &= \sum_{i=1}^{S} k_i \alpha_i^j kernel(x, x_i) + b \end{aligned} \tag{4.45}$$

where S represents the number of samples, m represents the number of classes, b represents bias, and x represents the data components.

One-versus-all exhibits linear complexity because only m hyperplanes are needed for m classes.

For the *one-versus-one* technique, classification is computed by constructing two hyperplanes for two selected classes. This is carried out across model pairs for m classes on the training set. This technique exhibits polynomial complexity, where $(m-1) \times \frac{m}{2}$ hyperplanes are needed for m classes.

4.5.4 Normalisation

Data normalisation or scaling is a sensitive part of SVMs training and testing [163]. Normalisation is a way of reducing the weight of frequent n-gram counts by preventing larger n-gram counts from dominating smaller n-gram counts. Various benefits are associated with normalising data, which include speeding up training and avoiding computational complexity with numerical values.

Normalisation can be performed either on the data (input features) or on the level of the kernel employed (normalising in feature space) [163]. Features are continuous, that is, they have different ranges of values, which can be measured using different scales. It is good practice to scale all features to a standard range. In a sparse data set, scaling features to a standard range is not recommended, because each feature will contain a different normalisation constant, thus, destroying sparsity [158].

Normalisation at the level of the kernel can be accomplished using *cosine kernel*. This technique normalises the kernel function $k(x, x')$ to:

$$kernel_{cosine}(x, x') = \frac{kernel(x, x')}{\sqrt{kernel(x, x)(x', x')}}. \tag{4.46}$$

For further details on which SVMs libraries were used, the size of n-gram models and how we normalised the data, see Section 4.6.5.

4.6 Experimental design

We evaluated the identification accuracy of different classification techniques discussed above. The aim was to identify the language origin of a single word at a time. Experiments were carried out on generic text corpora from four South African languages, and the results were presented

in terms of LID accuracy across all languages. Using empirical measurements, we analysed the following aspects:

- For standard NB classifiers: the difference in identification accuracy when using word types or tokens at different training corpus sizes and the implications of different smoothing techniques for different n-grams at different training corpus sizes.
- For SVMs: the effect of kernel choice.
- For both classifiers: the interplay between n-gram length, word length and classification accuracy.

4.6.1 Data

Generic text in three South African languages (Afrikaans, Sesotho, and isiZulu) was obtained from a pre-release of the NCHLT text corpora [133]. These corpora were collected from "gov.za", which is a South African government domain. This domain is devoted to topics relevant to South Africa, which include news events across South Africa or beyond, speeches, and statements from various cabinet meetings in the Republic of South Africa. All text is encoded in UTF-8 format to accommodate special characters found in Afrikaans and Sesotho. The South African English data was obtained from a broadcast news corpus [164]. We performed text normalisation to remove punctuation marks, numbers, formulae, dashes and brackets.

TABLE 4.1: Original data that contain all words. The number of unique words, total number of characters and average word length per language are shown.

	Total characters	Unique words	Average word length
Training Set	A - 1 840 494 E - 1 840 547 S - 1 840 697 Z - 1 840 570	A - 15 727 E - 15 765 S - 15 680 Z - 15 799	A - 8.64 E - 7.21 S - 9.80 Z - 10.38
Total	7 362 308	62 971	
Development Set	A - 221 040 E - 221 023 S - 221 087 Z - 220 933	A - 2 101 E - 2 060 S - 2 073 Z - 2 114	A - 8.28 E - 7.86 S - 8.78 Z - 9.87
Total	884 083	8 348	
Test Set	A - 221 140 E - 221 125 S - 221 007 Z - 221 041	A - 2 110 E - 2 080 S - 2 120 Z - 2 111	A - 7.28 E - 7.86 S - 8.18 Z - 8.80
Total	884 313	8 421	

4.6.1.1 Data partitioning

The data is randomly partitioned into a training set (80%), development set (10%) and test set (10%) based on the number of characters in running text. Table 4.1 contains the original data set, while Table 4.2 contains the newly partitioned set after removing overlapping words from the test and development set: that is, only unique words are retained. For the purpose of this experiment, Tables 4.1 and 4.2 will be referred to as the "all" and "unique" data sets, respectively. In Tables 4.1 and 4.2, "*A*" represents Afrikaans, "*E*" English, "*S*" Sesotho, and "*Z*" isiZulu. The rationale behind the experiment is that the T-LID (on running text) results are typically obtained by retaining multiple copies of the same token; word-based LID results are typically obtained from dictionaries, where only unique tokens are retained automatically.

In order to investigate the relationship between identification accuracy and training set sizes; we conducted a series of experiments, which involves creating different training subsets of different sizes from the "all" data set, namely: 250KB, 500KB, 1M, and 1.8M. To avoid bias of having the same words that run across both the test set and train set; we constructed different data subsets by removing the identical words from the data sets. Finally, the unique words extracted for training are 75KB, 113KB, 162KB, and 204KB in size, which are equivalent to the respective subset sizes, that is, 250KB, 500KB, 1M, and 1.8M. Table 4.2 shows the largest unique data set (240KB) together with the development and test data set. We used the *test* and *development* set from Table 4.2 across all experiments.

TABLE 4.2: Repartitioned data set after removing repeated words. The number of unique words, total number of characters and average word length per language are shown.

	Total characters	Unique words	Average word length
Training Set	A - 204 456 E - 204 751 S - 204 764 Z - 204 925	A - 15 727 E - 15 765 S - 15 680 Z - 15 799	A - 8.00 E - 7.99 S - 9.06 Z - 10.97
Total	818 896	62 971	
Development	A - 9 911 E - 9 931 S - 9 940 Z - 9 933	A - 740 E - 717 S - 738 Z - 710	A - 8.39 E - 7.85 S - 8.49 Z - 9.99
Total	39 715	2 905	
Test Set	A - 9 901 E - 9 961 S - 9 910 Z - 9 935	A - 744 E - 723 S - 745 Z - 707	A - 7.30 E - 7.78 S - 8.30 Z - 8.05
Total	39 707	2 919	

4.6.2 General discussion of experiment

In order to identify the language origin of individual words, we generated text of different n-gram character lengths from the corpus used. All generic words include word boundary markers (indicated by #). A preliminary experiment (not shown here) using naïve Bayes classifier shows that using word boundary markers improved classification accuracy for the current task. For each word all n-grams are extracted, for example, the word #BREAD# would be represented by the tri-grams #BR, BRE, REA, EAD, AD#.

During naïve Bayes training, an n-gram model is trained, to estimate the probabilities of the individual n-grams based on their counts in the training corpus. For example, when training a tri-gram model, a vector of tri-gram values is created in a T-dimensional space, where T is the number of possible tri-grams, and each tri-gram in the vector is associated with a specific likelihood estimate. For SVM training, a similar T-dimensional vector is constructed, but now each example is represented by the number of times a specific n-gram occurs in that training sample. These T-dimensional training samples are then used to train an SVM model.

Each test is characterised by the following parameters: language, training data used (size, unique/all), adopted n-gram model and technique employed. For example, given a text string from a particular language, extract its n-gram features. The n-gram features are passed as a parameter to a designated n-gram model. This n-gram model constitutes specific n-gram features, unique words or all words as the data set, classification technique employed, SVMs or naïve Bayes, on which the system was trained.

4.6.3 Evaluation metrics

To evaluate the performance achieved by the different n-gram models, we evaluate their accuracies based on classification accuracy (as describe in 2.6).

4.6.4 Analysis and results

This section focuses on evaluating the accuracy achieved by the different n-gram models on the test set. We examined how each n-gram model performed with different word lengths and LID classifiers. The influence of different training data set sizes on accuracy was also investigated. Specifically for this experiment, n-gram models ranging from 1 to 7 was used.

4.6.4.1 Naïve Bayes baseline back-off

The baseline model is based on a back-off technique that estimates word probability for tokens with n-gram length greater than or equal to the word length. Using standard naïve Bayes, the probability of an n-gram with length greater than or equal to the word length becomes zero. Some form of back-off is required to be able to produce results over all test types regardless of word length or n-gram parameter (prior to investigating more sophisticated smoothing techniques).

The baseline strategy adopts a technique that backs off to an n-gram size of the word length (W), minus a fixed value (x), where x is a small value. To obtain a suitable estimate for the fixed value, x, we performed 5-fold cross-validation on the training data and evaluated the effect of subtracting a fixed value, x, by setting x in the range of $W/2$ to W, where W represents total character counts per word without word boundaries.

For the baseline model, we found $W/2$ to be the most suitable parameter for x. This back-off technique is used in the baseline model to enable us to compare the results with more advanced techniques in later experiments.

4.6.4.2 Effects of using unique words over all available words

In the first experiment, we examine the influence of using unique words (types) against all words (tokens). Types that represent using a dictionary as input compared to tokens where running text are used as input. A reduction in computational complexity and training time is one of the benefits of using types over tokens. We therefore examine the influence of using only types against tokens in training, using the data set from Tables 4.1 and 4.2, partitioned into different sized subsets. For proper comparison, only one test data set is used across all different subsets.

Figure 4.1 shows the classification accuracies obtained between "unique" and "all", as defined above. It also shows the difference in identification accuracy using different n-gram models across various training data sizes. Relative difference can be referred to as the change in value, which is estimated by subtracting values of "all" from "unique". This baseline result is based on the naïve Bayes baseline back-off described in Section 4.6.4.1. Most notable, smaller n-gram models benefit most from the use of types, while higher-order models perform better under all tokens, with less than 1% percentage gain over lower-order models. These accuracies can be considered to be corpus-dependent. For further experiments, types are used over tokens (Table 4.2), as this results in reduced training time and computational complexity.

Table 4.3 shows the overall identification accuracy of the naïve Bayes baseline back-off method across various data sizes. With smaller data sets, higher-order n-gram models show poor performance, with the 7-gram model producing the worst performance. With more training data an

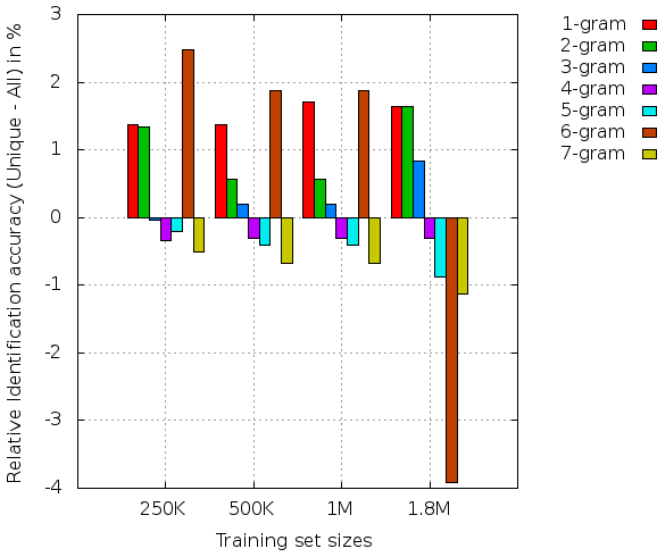

FIGURE 4.1: Difference in LID accuracy when comparing the baseline n-gram models trained using only unique tokens with one trained on all tokens. Results are provided at different training set sizes.

improvement in accuracy across all models is observed; however, the improvement on lower-order n-gram models is larger with the 4-gram model benefitting most.

Note that classification accuracy increased for all models with more training data. This means that we were unable to achieve asymptotic performance with the data sets available, which implies that higher accuracies are possible with increased training data.

TABLE 4.3: Classification accuracy of baseline naïve Bayes systems trained on unique types at different training sizes and evaluated on test set.

	1-gram	2-gram	3-gram	4-gram	5-gram	6-gram	7-gram
250K	58.715	75.176	78.488	78.488	77.384	76.614	75.878
500K	58.884	73.771	78.555	78.554	77.852	77.016	76.447
1M	58.983	75.845	79.793	80.462	79.893	79.258	79.090
1.8M	58.983	76.179	80.261	80.763	80.027	79.425	79.190

4.6.4.3 Smoothing analysis

Tables 4.4, 4.5, and 4.6 show classification accuracies of three smoothing techniques using different n-gram models and training data sets. The question is how we handle unseen features in the test data. Smoothing methods help to deal better with data sparseness by borrowing probability mass from higher-order n-grams and redistributing it among lower-order n-grams.

The discount parameter value (D) for absolute discounting is estimated by applying 5-fold cross-validation on the training set. To avoid overfitting and reduction in likelihood of the held-out

data, a discount value is selected that keeps the likelihood of the held-out data stable across the validation set.

As expected, with more training data, performance increases with n-gram length. Moreover, we observed a reduction in classification accuracy from the 6-gram model upwards. This is probably due to both the average word length of the data set and increased data sparsity of higher order models.

In conclusion, for the current data set, the 4-gram model seems to be the preferred model across all three modelling techniques employed. The best accuracy (of 87.72%) was obtained using Witten-Bell discounting. This outperformed both absolute discounting and Katz smoothing, even though the difference in accuracies were relatively small. In general (for higher-order models) an 8% accuracy increase was observed when comparing Witten-Bell discounting to the same n-gram model without smoothing.

TABLE 4.4: Classification accuracy using Witten-Bell smoothing at different n-gram lengths, evaluated on test set.

	1-gram	2-gram	3-gram	4-gram	5-gram	6-gram	7-gram
250K	72.767	83.239	85.212	85.781	85.279	84.811	84.778
500K	72.968	82.971	84.978	85.781	85.480	84.978	84.711
1M	72.700	83.673	86.417	86.250	86.785	86.283	86.183
1.8M	73.202	84.008	86.584	87.722	87.387	87.554	87.287

TABLE 4.5: Classification accuracy using Katz smoothing at different n-gram lengths, evaluated on test set.

	1-gram	2-gram	3-gram	4-gram	5-gram	6-gram	7-gram
250K	72.533	82.937	84.343	85.212	84.778	84.744	84.744
500K	72.968	83.272	85.012	85.614	85.882	85.547	85.547
1M	64.604	82.168	85.079	84.778	84.911	85.212	85.212
1.8M	73.202	84.108	86.751	87.487	86.986	86.818	86.818

TABLE 4.6: Classification accuracy using absolute discounting (d = 0.24) at different n-gram lengths. Accuracy evaluated on test set while d was calculated by applying 5-fold cross-validation on training set.

	1-gram	2-gram	3-gram	4-gram	5-gram	6-gram	7-gram
250K	72.800	83.406	85.112	85.815	84.844	84.410	84.041
500K	72.633	83.205	84.978	85.313	85.346	84.543	84.343
1M	72.432	83.607	86.517	86.183	86.417	85.982	85.647
1.8M	72.968	84.008	86.484	87.354	86.685	86.618	85.714

4.6.5 Support Vector Machines

As discussed in Section 4.5, SVMs use the concept of a hyper-plane constructed in a multidimensional space to minimise error function. To recapitulate what was discussed earlier in the section above on SVMs for classification, four decisions are vital to proper classification: (1) data preparation, which concerns selection of the training data and deciding how many classes should be included; (2) converting samples to feature vectors, where each sample in a training set should contain a class label, feature index and feature value; (3) data normalisation or scaling, in order to prevent over-fitting (which results in a zero mean and unit variance); and (4) proper choice of SVMs kernels for good classification accuracy. Determining the parameter of choice also involves setting various hyper-parameters associated with a specific kernel. Predicting ideal hyper-parameter values can be achieved using a grid search on the development set.

This experiment employed two SVM libraries, namely LibSVM [163] and LIBLINEAR [165], for classification. These two libraries use different methods for multi-class classification. LIBSVM uses the one-against-one method for classification, where one SVM is constructed for each pair of classes. Classification is performed using a voting strategy. LIBLINEAR uses one-against-rest, where a classifier is trained for each class, prior to voting.

We carried out the experiment with two widely used kernels, namely RBF and a linear kernel. The SVM training set was generated by combining n-grams across languages to create the SVM training set. Each selected model has a feature dimension equal to the number of n-gram combinations. After creating the feature vectors, we scaled each attribute in the range of [0, 1] in the training set. The same scaled value used in building the model was used to scale the test and development set. 5-fold cross-validation was performed on the training set to obtain optimal kernel parameters. The n-gram model length was limited to 5-gram because of the longer training time and extensive resource usage associated with higher-order n-gram models.

Tables 4.7 and 4.8 show classification accuracies obtained using the RBF kernel and linear kernel (respectively) and different n-gram models across various training data sets. All models improved with more training data, and the RBF kernel produced higher accuracies than the linear kernel.

TABLE 4.7: LID accuracy using a linear kernel at different n-gram lengths, evaluated on the test set.

	2-gram	3-gram	4-gram	5-gram
250K	83.506	85.012	84.744	77.651
500K	83.473	85.547	84.778	83.473
1M	84.610	86.718	86.317	84.778
1.8M	84.744	87.454	86.986	83.674

TABLE 4.8: LID accuracy using an RBF kernel at different n-gram lengths, evaluated on the test set.

	2-gram	3-gram	4-gram	5-gram
250K	85.012	86.685	85.012	81.632
500K	85.982	86.283	85.881	82.335
1M	86.952	87.086	87.621	84.744
1.8M	87.789	88.123	88.157	84.376

TABLE 4.9: A comparison of LID accuracy for different classifiers investigated.

	NB (n=4)	WB (n=4)	Katz (n=4)	ABS (n=4)	Linear (n=3)	RBF (n=3)
250K	78.488	85.781	85.212	85.815	85.012	86.685
500K	78.555	85.781	85.614	85.313	85.547	86.283
1M	80.462	86.250	84.778	86.183	86.718	87.086
1.8M	80.763	87.722	87.488	87.354	87.454	88.123

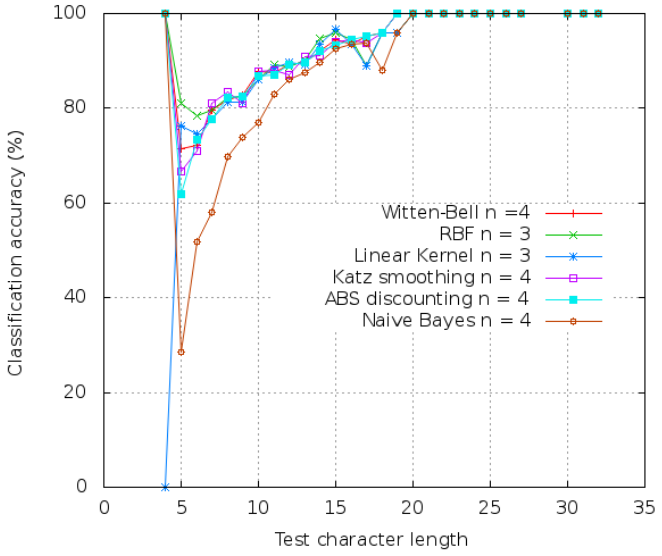

FIGURE 4.2: LID accuracy of words of different lengths, when training with 1.8M data set, evaluated on test set. Note that word boundary markers are included in the calculation of word length.

4.6.5.1 Effect of corpus size

The size of the training data set is one of the factors that influences LID accuracy of running text. For the task at hand, it had to be determined what effect different training data sizes have on accuracy, and how different classifiers and smoothing techniques perform as training data sizes increase.

In Table 4.9, results are shown for different LID techniques and training set sizes. On the smallest data set, NB with smoothing techniques outperformed the SVM with a linear kernel. For all training set sizes investigated, an SVM with an RBF kernel performed best. NB classification

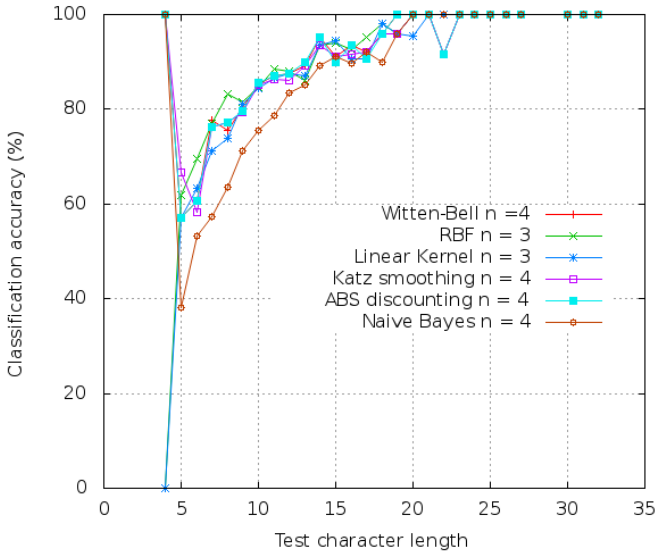

FIGURE 4.3: LID accuracy of words of different lengths, when training with 250KB dat set, evaluated on test set. Note that word boundary markers are included in the calculation of word length

with absolute discounting and Witten-Bell smoothing, as well as SVM classification with a linear kernel, shared similar performance across a range of training data sizes.

4.6.5.2 Effect of word length

In this section, the aim was to analyse another factor that influences classification accuracy. We were interested to see how classifiers and smoothing techniques performed at different word lengths. Using the largest and smallest training data sets, figures 4.2 and 4.3 show the classification accuracy achieved when evaluating words with different character lengths.

The fluctuation in classification accuracy for words shorter than 5 characters could be due to the fact that these words occur fairly infrequently in the test data set. The SVM RBF classifier obtained the highest performance for short character lengths (5 characters or fewer), while naïve Bayes classification achieved the worst performance for the same scenario. With longer words (20 characters or more), the naïve Bayes classifier achieves a classification accuracy of 100%. This result confirms previous results on using Naïve Bayes for long sentences. Also, with 12 or more characters, the difference in classification using naïve Bayes with and without smoothing becomes insignificant.

4.7 Conclusion

In this chapter, we experimented with two of the most promising classification methods when performing LID of generic words in isolation: both are n-gram based; one applies standard naïve Bayes classification and the other utilises a SVM. Moreover, the difference between using unique types and all tokens when training a naïve Bayes classifier was investigated, and a computational win was found when using unique types, which was not offset by any loss in LID accuracy.

As the baseline classifier, a naïve Bayes back-off method was used for words with character length greater than or equal to the n-gram model parameter. This method produced good results across all n-gram lengths. The naïve Bayes classifier achieved an identification accuracy of 80.76% using a 4-gram model when tested on unique types.

Furthermore, to reduce the total number of unseen tokens as the n-gram model increased, we experimented with Katz smoothing, absolute discounting and Witten-Bell smoothing. Among the three smoothing techniques, Witten-Bell smoothing achieved the best performance, but only with a small margin (less than 1%). Smoothing improved average classification accuracy for higher order n-grams by a percentage of approximately 8% (from 79.89% to 87.55%).

The highest classification accuracy of 88.12% was obtained using an SVM with an RBF kernel and an n-gram length of $n = 3$. This classifier clearly outperformed the naïve Bayes classifier without smoothing, with NB classification accuracy improving considerably when smoothing is used.

In addition, the experiments showed that better identification can still be achieved with more training data. Achieving an asymptotic performance depends not only on the available data set; it also depends on the word length, with longer words achieving very high accuracies on the 1.8M data set. For the four-language task studied, the accuracy of the SVMs (88.16%, obtained with a RBF) was higher than that of the naïve Bayes classifier (87.62%, obtained using Witten-Bell smoothing), but the latter result was associated with a significantly lower computational cost. The computational cost increases as the training set size increases, based on the trade-off between accuracy and computational cost required for a specific application.

The tools and baseline results used here will be referred back to later in this thesis. In the next chapter (Chapter 5), it is determined whether it is possible to obtain better results using a novel algorithm. In Chapter 6, these techniques are applied to proper names, specifically.

Chapter 5

Joint Sequence Models for T-LID

5.1 Introduction

In Chapter 4 we examined two commonly used classifiers that were applied to LID of generic words in isolation. The analysis showed that good performance accuracy could be achieved on words in isolation, given sufficient training data. As it is generally believed that more training data improve performance accuracy, we are interested in whether it is possible to obtain better results using JSMs on the same amount of data as before.

JSMs were developed as a grapheme-to-phoneme (G2P) conversion technique – aimed at predicting the pronunciation of a word from its orthographic form. Over the past decade, JSMs have become one of the most popular G2P algorithms. This leads to an interesting question: is it possible to map the T-LID task to the pronunciation task and obtain good performance accuracy? All the algorithms discussed in Chapter 4 use n-grams as features; these are the same features we would now like to use with JSMs.

In this chapter, the applicability of JSMs to the T-LID task is investigated. Performance is analysed when comparing JSMs with the better-known SVM classifiers (used as the baseline classifier in the experiments in this chapter). Prior to experimental analysis, we used a Google Translate-based technique for spell-checking and language verification of the dataset. For more information on how to perform text preprocessing for under-resourced languages using Google Translate, see Appendix A.

5.2 Joint sequence models

Following descriptions and formulae in [47], we review JSMs as described below. Although, different mathematical notations have been implored for clarity purposes where necessary thereby

tailoring our definitions to resemble a traditional T-LID technique. JSMs [47] are based on the concept of 'gralanguages'. Each gralanguage consists of a sequence of letter linked to a sequence of languages modelled as a single unit. Gralanguages are sometimes referred to as joint multigrams [166]. A gralanguage with only one letter and one language is referred to as *singular gralanguage*. The groupings of gralanguages into segments is referred to as *co-segmentation*.

Mathematically, gralanguages can be described as:

$$q = (x, k) \in Q \subseteq X^* \times K^* \tag{5.1}$$

where q represents the gralanguage, x represents a letter string, k represents a language string, Q represents the set of gralanguages, X represents the set of letters, and K represents the set of languages.

Before we describe the approach to use JSMs for LID (Section 5.3), the JSM learning model (Section 5.2.1) and the main parameter choices that influence model performance (Section 5.2.2) are reviewed. For more detail on JSMs, see [47].

5.2.1 Conceptual approach

During training, JSMs must deduce alignment on the level of letter-to-language. For example, given N word samples with their corresponding language of origin, but without co-segmentation, how do JSM align on the level of letter and language, given the JSMs? The probability of a co-segmentation is computed as:

$$p(x, k, \zeta) = p(q) \tag{5.2}$$

where ζ represents the segmentation into joint units. This segmentation represents a hidden variable. JSMs use the expectation maximisation (EM) algorithm to estimate the maximum likelihood of the training model. Generally, EM algorithms are used for obtaining ML or MAP estimates when some of the data is not observed. In the present case, the segmentation ζ is the hidden variable that is not observed.

The EM algorithm is used to improve ML estimates of the training data in an iterative fashion. The joint probability, $p(g, \varphi)$, for the most probable joint-segmentation for a particular sequence of letters can be modelled by summing over all relevant co-segmentations:

$$p(x, k) = \sum_{q \in S(x,k)} p(q_1, ..., q_G) \tag{5.3}$$

where k represents the language sequence, x represents the letter sequence, q represents the gralanguage sequence, G is the length of the gralanguage sequence and S represents the set of

all joint segmentations. The joint probability distributions of eq. 5.3 can be approximated to a probability distribution:

$$p(q_1^G) \approx \prod_{i=1}^{G+1} p(q_i \mid q_{i-1}, ..., q_{i-M-1}) \tag{5.4}$$

where G is the length of the gralanguage sequence and M denotes the order of the multi-gram model employed.

The probability distribution of eq. 5.4 can be re-estimated for the updated parameters ζ using EM algorithm for the higher-order of the multi-gram model $M > 1$:

$$p(q;\zeta) = \prod_{j=1}^{|q|} p(q_j \mid h_j;\zeta) \tag{5.5}$$

$$e(q,h;\zeta) = \sum_{i=1}^{N} \sum_{q \in S(x_i,k_i)} p(q \mid x_i,k_i;\zeta) n_{q,h}(q) \tag{5.6}$$

$$e(q,h;\zeta) = \sum_{i=1}^{N} \sum_{q \in S(x_i,k_i)} \frac{p(q;\zeta)}{\sum_{q \in S(x_i,k_i)} p(q';\zeta)} n_{q,h}(q) \tag{5.7}$$

$$p(q \mid h;\zeta') = \frac{e(q,h;\zeta)}{\sum_{q'} e(q',h;\zeta)} \tag{5.8}$$

where h represents the sequence of preceding joint units $h_j = (q_{j-M+1}, ..., q_{j-1})$, $n_{q,h}(q)$ represents the number of times the order of the multi-gram model gralanguage (M-gram) $q_{j-M+1}, ..., q_j$ occurs in the sequence of q, $e(q;\zeta)$ represents the *evidence* for q, which is the expected number of times gralanguage q occurs in the training sample under the updated parameters ζ. Evidence values are estimated on the training set after extracting the held-out data samples. For the first order of the multigram mode (M = 1), symbol h can be discarded for parameter re-estimation in eq. 5.8.

For EM iterative procedures, JSMs initialise model parameters by assigning an identical initial probability to all multi-grams satisfying a gralanguage length constraint that is manually set. This uniform initial probability distribution is the inverse of all allowable gralanguages:

$$p_0(q) = \left\{ \sum_{l=0}^{L} \sum_{r=0}^{L} |X|^l |K|^r \right\}^{-1} \tag{5.9}$$

where l and r represent letter and language token respectively.

JSMs compute the posterior probability of a translation:

$$p(k \mid x) = \frac{\sum_{q \in S(x,k)} p(q)}{p(x)}. \tag{5.10}$$

5.2.2 Parameter definition

The following main algorithmic choices influence performance:

- *Model initialisation* can be performed in two main ways:
  - Using a flat probability distribution, where all multigrams are initialised to an identical initial probability (see eq. 5.9). One issue relating to uniform initial probability distribution is that all joint multi-grams construed from the training samples are assigned a probability mass, whereas in reality not all these gralanguages contribute to the final model. This in turn increases the gralanguage inventory for the correct model. JSMs address this by using discounted evidence trimming, in which joint multi-grams with evidence values lower than the discounted parameter are removed from the model. The discounted evidence is distributed over unseen tokens. Evidence in eq. 5.8 can be re-estimated as:

    $$\hat{e}(q, h; \zeta) = \max e(q, h; \zeta) - d_M, 0 \tag{5.11}$$

    where M represents the order of distribution, $d_M > 0$ represents the discounted parameter.
  - Initialising based on counts, where initialisation is based on the number of occurrences of a gralanguage.

    $$c(q) := \sum_{i=1}^{N} \sum_{l_1=1}^{|x_i|} \sum_{l_2=l_1}^{|x_i|} \sum_{r_1=1}^{|k_i|} \sum_{r_2=r_1}^{|k_i|} \times \beta((x_{l_1} \smile ... \smile x_{l_2}, k_{r_1} \smile ... \smile \varphi_{r_2}) = q) \tag{5.12}$$

  Both these methods are subjected to an important parameter: the gralanguage length constraint (L), which must be manually set. (See below.)
- Higher *m-gram model orders*, M, produce better accuracy and gradually reach an asymptotic level (as M is increased).
- *Gralanguage length*: Gralanguage length (l_{min}, l_{max}, r_{min}, r_{max}) can be restricted during training. These values indicate the minimum and maximum number of letters (l for left-hand side) and languages (r for right-hand side) respectively allowed per gralanguage unit. These parameters have a significant effect on the size of the gralanguage inventory

produced. [47] reported higher accuracy using singular gralanguages with high model order size as opposed to when allowing unconstrained gralanguage generation with higher-order models. For more details, see [47].

- *Estimating discount*: One generally known problem of ML estimates is that they tend to over-fit the training data of the predicted model. The standard JSM implementation uses modified Kneser-Ney for discounting the EM algorithm to handle unseen data better and prevent over-fitting. Discount parameters are estimated on held-out data, across all model orders, one model order at a time.

$$p_M(q \mid h) = \frac{\max e(q,h) - d_M, 0}{\sum_{q'} e(q',h)} + \lambda(h) p_{M-1}(q \mid \bar{h}) \tag{5.13}$$

where $p_{M-1}(q \mid \bar{h})$ represents the generalised, lower-order of the multi-gram distribution conditioned on the preceding history $\bar{h_i} = (q_{i-M+2}, ..., q_{i-1})$. $\lambda(h)$ makes the overall distribution sum to one. After parameter estimation, a fold-back strategy can be used to return held-out data to the training set.

5.3 Using JSMS for LID

In this section, the approach to using JSMs for LID is explained.

5.3.1 Dictionary setup

The T-LID task is recast as a 'pronunciation learning' task, with each phone being replaced by a language identifier, repeated for the length of the word. For example, the English word 'queen' or Sesotho word 'dumela' will be represented in the lexicon as:

```
#queen#     E E E E E E E
#dumela#    S S S S S S S S
```

where E represents English, S represents Sesotho and # represents a word boundary marker (as found useful for naïve Bayes classification [35]). In effect, we are therefore modelling letter chunks rather than gralanguage chunks.

5.3.2 Classifying text

Given an input word, the task is to find the most likely source language (as defined in Section 1.1). Once a string of 'phones' (in the present case, language identifiers) has been predicted, one of two simple voting schemes is used to select the final language of origin:

1. Majority voting: This technique uses a simple concept, in which the language identifier observed most often is selected. It can also be referred to as selecting the language identifier with the highest number of counts. Mathematically:

$$\begin{aligned} C(X) &= \text{mode}\,\{h_1(X), h_2(X), ..., h_M(X)\} \\ C(X) &= \text{mode}_{l=1}^{M}\ h_l(X). \end{aligned} \tag{5.14}$$

   where M represents a list of languages, X represents the input word string, C represents the source language for the word X, and $h_l(X)$ represents the language identifier for the particular word X.

   There are scenarios where more than one language identifier give equal highest counts (voting is tied). In such a case, we regard the source language as 'Unknown'.

   To illustrate the above Equations in a simple task, given a prediction "Anglican - E E S S Z Z E E ", majority voting yields "E" (English) as the source language because of its frequency count. Similarly "Anglican - E E E A A A A E ", will be considered to be 'Unknown', since the strategy is indecisive owing to a voting tie.

2. Log probability voting: For a specific language identifier, the likelihood is multiplied (for that particular language), and the language with the highest likelihood is selected. In practice, the log-likelihoods across gralanguage sequences are summed in order to reduce computational complexity. This technique can be carried out in four steps:

   - Compute each gralanguage log-likelihood.
   - With gralanguage log-likelihood estimated, categorise each language identifier with its associated likelihood.
   - For each category, sum over all matching gralanguage sequences.
   - Select the category (language identifier) with the highest likelihood estimate.

   Based on further analysis, a third approach will be introduced in Section 6.2.4.

5.4 Experimental set-up

The applicability of JSMs to T-LID in the context of a 4-language task is analysed. An SVM baseline is obtained against which to evaluate the JSMs results. The effect of using different

JSMs parameters is analysed; specific attention is paid to the interplay between training corpus size, word length and classification accuracy.

5.4.1 Data sets

The specific T-LID task involves four South African languages (English, Afrikaans, isiZulu and Sesotho). Training data was obtained from the *NCHLT-inlang* dictionaries [167] as discussed earlier in Section 4.6.1. The correctness of the word-list was verified using automated spell checkers, G-Translate (cf. Appendix A.2.1 for more details) and language practitioners, although the analysis showed that the published lists still contained errors. A second round of (higher precision, lower recall) spell-checking was performed using language practitioners and first-language speakers for the three languages for which this was possible, namely Afrikaans, isiZulu and English. This resulted in an 'original' (from *NCHLT-inlang*) and 'spell-checked' (further refined here) list. In addition, a third set ('no_bilingual') was created, containing no known bilingual words (all words that occurred across training corpora were removed). Table 5.1 shows the final data statistics for each data set partition.

TABLE 5.1: Final 12K data statistics for each data set partition as obtained from *NCHLT-inlang* dictionaries.

	Original		Spell checked		No bilingual	
Language	Avg. word character	Character count	Avg. word character	Character count	Avg. word character	Character count
Afrikaans	11.5	137 488	11.5	137 661	11.5	137 591
English	9.0	107 560	9.0	107 611	9.0	107 545
Sesotho	9.3	110 950	9.3	110 997	9.3	110 970
isiZulu	10.4	124 380	10.4	124 440	10.4	124 360

5.4.2 Partitioning of the data set

During experiments, training set sizes were limited artificially to investigate the effect of training corpus size: the required number of training samples were selected randomly per language. Training corpora were constructed by combining an equal number of words per language. All results were obtained through 4-fold cross-validation. Where parameter optimisation was required, a small (10%) development set was used and folded back after training. Specified training set sizes included the development set, and indicated the number of training samples *per language*. For each data set (Section 5.4.1), test sets were kept constant across all experiments.

5.4.3 Evaluation metrics

Standard precision, recall and the combined F-measure were used to evaluate performance per language (Section 2.6 for more details). To estimate a confidence interval, the standard error of the different measures across n folds ($\sigma/\sqrt{(n)}$)) was calculated, with σ the standard deviation of the specific measure considered.

5.5 Experiments and results

5.5.1 SVM baseline

In order to obtain a baseline result, we used the SVM implementation that produced the best T-LID results in [35]. Each training sample was represented by an N-dimensional vector, with N the number of unique n-grams observed. Each feature constitutes an n-gram index and n-gram value, where the value indicates the number of times that the specific n-gram was observed in the training sample, and the index represented the n-gram axis in N-dimensional space; the language identifier per sample acted as class label. As discussed earlier in Section 4.5.4, training data was scaled in the range of [0,1] to avoid over-fitting and prevent features with a high range from overriding those with smaller values. An RBF kernel (as was shown to outperform a linear kernel) was used and n-gram lengths of 3 provided optimal results. While these n-grams seem small, this is a result of the test words, many of which are themselves quite short.

In Section 4.5, it was stated that an RBF kernel requires that two hyper-parameters be optimised: the soft margin parameter (C) and kernel width ($gamma$). As motivated in [157], $gamma$ was set to the reciprocal of the sample dimensionality (with sample dimensionality ranging from approximately 4,500 to 6,500 in this case) and a grid search for C (in the range 10^{-4} to 10^{4}) performed, using a development set. Results are shown in Table 5.2; all results are 4-fold cross-validated. As expected, results on the spell-checked data are better than on the original set, and removing bilingual words makes the task a little easier. Also, larger training sets improve accuracy, with additional gains expected for data sets larger than 12K.

Comparing Tables 5.2 and 4.9, we show results that depict two experiments that investigate different classifiers for T-LID task. Specifically, both SVM experiments (from Tables 5.2 and 4.9) use the same implementation that produce the best result. However, differences lie in the dataset sizes and evaluation metric employed. It is, of course, difficult to make a direct comparison across datasets. We observe that accuracy increases given enough training data as discussed in Section 4.6.5.1 for both experiments.

TABLE 5.2: Precision (P), recall (R) and F-measure (F) achieved with SVM baseline for different training data sets. The confidence interval is based on estimated standard error. 'or', 'sp' and 'nb' indicate original, spell-checked and no_bilingual data set respectively.

Size	P_{or}	P_{sp}	P_{nb}	R_{or}	R_{sp}	R_{nb}	F_{or}	F_{sp}	F_{nb}
2K	90.86± 0.002	91.13± 0.005	90.53± 0.005	90.84± 0.002	91.08± 0.005	90.50± 0.005	90.85	91.10	90.51
4K	92.94± 0.002	93.04± 0.001	93.13± 0.001	92.94± 0.002	93.06± 0.001	93.14± 0.001	92.94	93.05	93.14
6K	93.48± 0.001	93.60± 0.002	93.78± 0.002	93.51± 0.001	93.63± 0.002	93.84± 0.002	93.50	93.62	93.81
8K	94.21± 0.001	94.09± 0.002	94.33± 0.002	94.21± 0.001	94.10± 0.002	94.34± 0.002	94.21	94.10	94.33
10K	94.59± 0.001	94.82± 0.001	94.64± 0.001	94.60± 0.001	94.82± 0.001	94.65± 0.001	94.59	94.82	94.65
12K	94.74± 0.002	95.05± 0.001	95.15± 0.001	94.73± 0.002	95.05± 0.001	95.16± 0.001	94.73	95.05	95.16

5.5.2 Initial JSM implementation

The training data is constructed and JSM models trained as described in Section 5.3. In order to build a model, a few model parameters are considered such as model initialisation with counts, unconstrained contexts and full EM is used during training. Discount parameters are optimized using a 10% hold-out set, and folded back later. Model order is increased until asymptotic performance is obtained (for all experiments at a model order of $M = 8$).

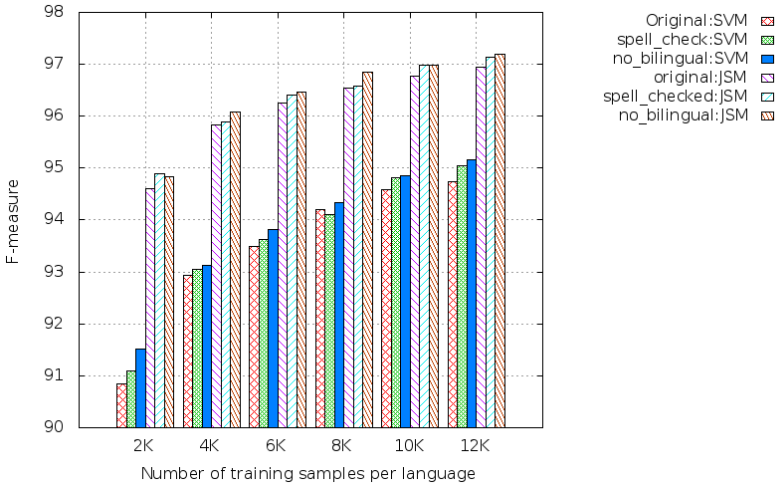

FIGURE 5.1: Comparing identification accuracy of (context-unconstrained) JSM and SVM across different data sizes, when trained and evaluated on three different data sets: original, spell_checked and no_bilingual.

Fig. 5.1 displays the results obtained for the same data sets analysed in Section 5.5.1, when classified using the initial JSM implementation. Similar trends are observed as with SVM classification: performance increases without yet reaching an asymptote; the 'original' task is the most

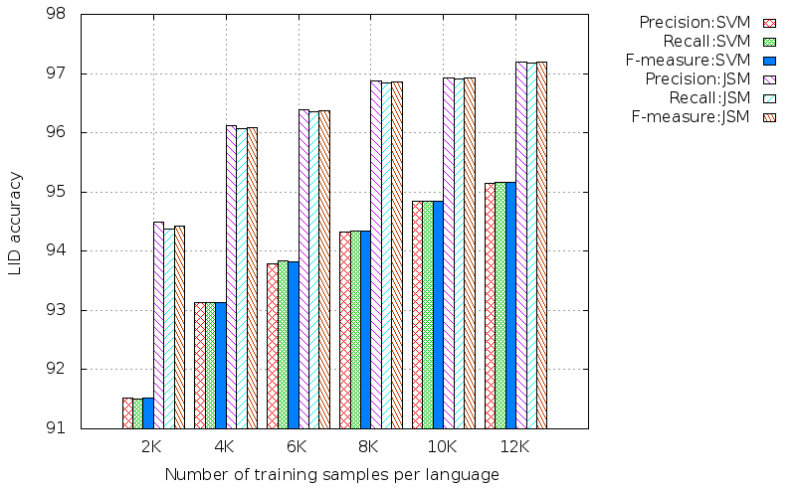

FIGURE 5.2: Precision, recall and F-measure of (context-unconstrained) JSMs and SVMs systems for different data sizes when trained and evaluated on the *no_bilingual* data set.

difficult, the 'no_bilingual' one the easiest. However, for all data sizes, the JSM implementation outperforms the SVM implementation: at 2K, an absolute increase of 3.32% is observed; at 12K, an absolute increase of 2.03%. This translates to a relative error reduction of approximately 40% over the different data sizes evaluated. Fig. 5.2 shows precision, recall and F-measure separately for the JSM system, achieved on the *no_bilingual* data set: the only data set used from this point onwards. As the test sets are balanced across languages, little additional information is provided by considering precision and recall separately and only the F-measure is considered from here onwards.

5.5.3 Effect of gralanguage length constraints

In [47], Bisani and Ney found that allowing gralanguages to have arbitrary length produced good G2P performance at low-order models, but poorer performance at higher-order models. Their best results were obtained when restricting letters (not gralanguages) to be singular (0 or 1) and allowing model order to increase unrestricted until asymptotic performance was reached. We therefore consider whether any benefits can be obtained from restricting gralanguage length.

Given the way the T-LID task has been framed (see Section 5.3), restricting either component of the gralanguage has the same effect as restricting both. That is, when restricting gralanguage length to y, we are in effect restricting both the left- and right-hand side to y.The (l_{min}, l_{max}, r_{min}, r_{max}) notation introduced in Section 5.3 is used and results are evaluated when restricting gralanguage length to 1 or 2, respectively.

For the task at hand, we observe that when varying l_{max} and r_{max}, results are comparable irrespective of whether l_{min} and r_{min} are set to 0 or 1. (This is not expected from pronunciation modelling tasks where a null letter or phoneme is required, but is specific to the way the LID

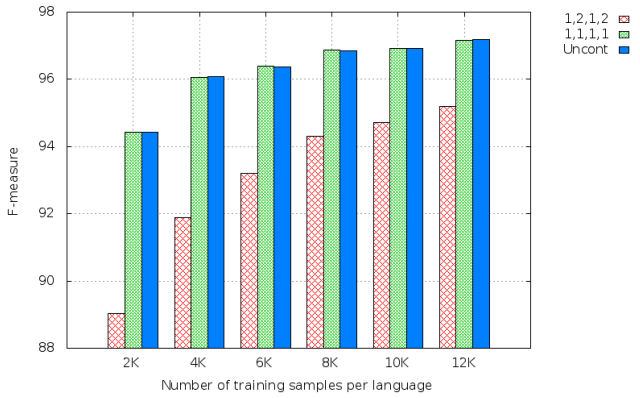

FIGURE 5.3: LID accuracy with different context constraints at different training data sizes.

task has been framed.) As using a minimum length value of 0 trains much slower than using a value of 1, 1 is used from here onwards. Note that the difference between gralanguage length 0 and 1 parameters is that gralanguage length 0 allows a language null while gralanguage length 1 does not.

Fig. 5.3 shows LID accuracy (using F-measure) across different data sizes for different gralanguage lengths. As can be seen, there is almost no difference between singular gralanguages (1, 1, 1, 1) and the 'unconstrained' system. In fact, analysis of the gralanguage inventory produced shows that unconstrained gralanguage formation automatically produces singular gralanguages for this task. Restricting gralanguages to be of length 2 decreases accuracy. (Even though the algorithm is not prevented from forming singular gralanguages when restricting the length of chunks to 2, sub-optimal chunk formation occurs early in the training process.) In the remainder of the analysis, unconstrained gralanguage formation is allowed.

5.5.4 Log probability voting

For most words, tie resolution is not required. We are interested in understanding the extent to which log probability voting is required at all. For the smallest (2K) and largest (12K) training sets, we evaluate the number of errors per word length that do involve ambiguous labels (labels that can possibly be resolved using either majority or log probability voting). It is only a minority of errors (12.8% of errors at 2K, 4.2% of errors at 12K) that involve ambiguity: all other errors are tagged with a single (incorrect) language label. The number of errors that do contain ambiguous labels when training with the 12K set is shown in Fig.5.4. When using log probability voting, we observe a slight increase in performance for almost all test sample lengths (with only one decrease in performance, observed at length 7). The same trend is observed at 2K: while not seen at every word length individually, a small improvement in accuracy is observed overall.

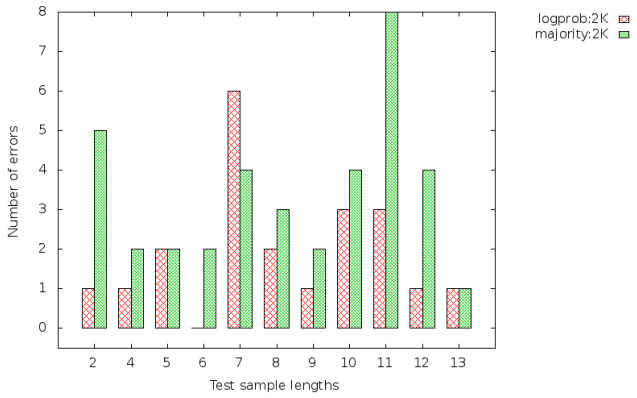

FIGURE 5.4: Analysis of errors using two different tie resolution strategies on largest training data set.

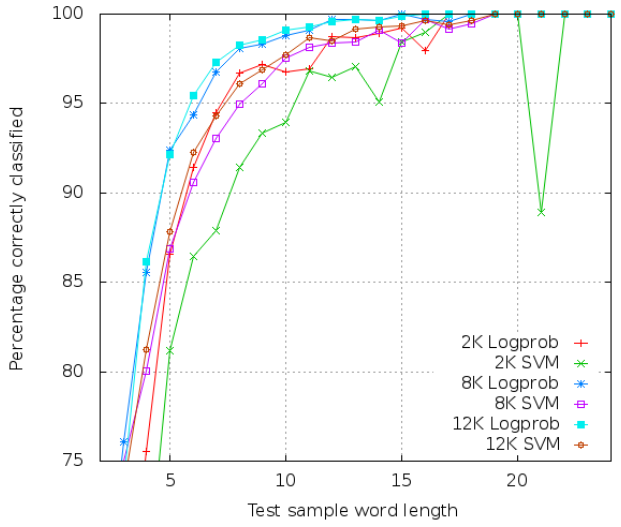

FIGURE 5.5: Comparative classification accuracy of words of different lengths for SVM baseline and JSM (with log probability voting) at 2K, 8K and 12K training set sizes.

5.5.5 Effect of training corpus size and word length

In this section, the new method (unconstrained JSMs with log probability voting) is compared with the baseline SVM classifier, given different training set sizes and test word lengths. Results are shown in Fig. 5.5. As expected, we observe that performance increases with both longer word lengths and additional training data. At longer word lengths (> 23 characters) both techniques achieve perfect classification accuracy. However, when considering shorter words, there is a clear advantage to using the JSM-based technique, across all training set sizes. It is interesting to note that the errors made by the two techniques show limited overlap: if required, additional wins may be possible by combining the two approaches.

We also observe that attaining asymptotic accuracy on different word lengths depends on available training samples: test sample length on the smallest data set obtained at 18 character length

in comparison to 15 character length on the largest data size. Generally, the SVMs technique suffers considerably from limited training data.

5.6 Conclusion

This chapter summarises JSMs as a novel algorithm for LID of individual words in isolation. For pronunciation modelling, accurate word-based T-LID is often required. We find that by recasting the T-LID task as a pronunciation modelling task, we can apply JSMs with limited modification to the training process, and obtain competitive accuracies in this way, especially on short test words. Two strategies are evaluated for tie resolution, and we find small gains in using the conditional log probabilities, rather than a simple majority voting scheme. An additional advantage of the new technique is that it is fast and simple to train, and does not require the extensive optimisation that is typically part of the SVMs training process.

For the specific task studied (a 4-language classification task) we find that the proposed JSM-based technique reduces the SVM error with approximately 40% across a range of training data sizes evaluated. For the largest training set (48K words, 12K per language) an accuracy of 97.2% is obtained, which compares well with the best SVM classifier evaluated (at 95.2%).

Finally, the JSMs technique will be utilised in the subsequent chapters, particularly on LID of proper names and multilingual words in isolation.

Chapter 6

Language identification of proper names using JSM

6.1 Introduction

In this chapter we explore LID of proper names using JSMs. Earlier results were obtained on generic words; we now shift our focus to proper names, specifically. It is investigated how JSMs generalise given a small training sample to predict the language of origin of a name.

We first determine how well the JSM technique performs for proper names using the Multipron corpus (Section 6.2) before we apply this technique to tag an initial version of a corpus prior to human verification (Section 6.3).

6.2 Multipron analysis

6.2.1 Data

We experiment with two different corpora: NCHLT-inlang and Multipron. The NCHLT-inlang dictionaries were developed in parallel with the NCHLT speech corpus [168]. For this work, only the word lists from the dictionaries, consisting of 15 000 unique words per language, were used. These were estimated to be frequent words based on available corpus counts, and LID accuracy was verified using automated spell checkers and language practitioners [167]. The same edited lists as used in Section 5.4.1, were used, where a second round of higher precision, lower recall spell-checking was performed for three of the languages (Afrikaans, isiZulu and English). These lists do not contain multilingual words. Specifically, we use a 40K-word subset of the NCHLT

data for compartive purposes. Table 7.5 shows the language distribution and word statistics of the 40K-word subset.

Multipron, as described earlier in Chapter 3, contains a combination of both firstname and lastname as a single entity separated by a space. This work uses the Multipron dictionary described in [127] and extracts its word list. We separate the names based on a space delimiter, which resulted in firstname and lastname as a single word.

TABLE 6.1: NCHLT 40k subset: language distribution and word statistics.

Language	Word count	Average word length	Character count
Afrikaans	10K	10.5	104 531
English	10K	7.9	79 768
Sesotho	10K	8.3	82 537
isiZulu	10K	9.4	93 617

6.2.2 Data partitioning

The multipron data set contains only mono-lingual words. Considering word-list count per language, we notice that the Afrikaans language has the lowest number of words. Therefore, a subset per language that matches the total number of words available in Afrikaans is selected resulting in 1008 words across all four languages (252 words per language). Table 6.2 shows the data distribution and statistics of the original and selected subset from Multipron corpus.

Because of the limited size of the extracted set, we use 10-fold cross-validation, dividing the sample set at random into 10 equal sizes. It was ensured that the word-list from each language was proportionally distributed among each of the 10 parts. This protocol is repeated 10 times, with each round consisting of different test samples. Per fold, the model is trained on 90% of the data samples with the other 10% set aside as testing data. The overall accuracy is computed by estimating the average classification accuracy across all k folds. For NCHLT 40K, which uses an already existing model, testing data samples of each part of the K-fold are applied against the once-off trained model.

TABLE 6.2: Multipron corpus: language distribution and word statistics of the original and selected subset.

	Word Count		Average word length		Character count	
Language	Original	Subset	Original	Subset	Original	Subset
Afrikaans	252	252	6.9	6.9	1 743	1 743
English	522	252	6.2	6.2	3 232	1 563
Sesotho	264	252	7.7	7.4	2 032	1 875
isiZulu	254	252	7.2	7.2	1 820	1 814

6.2.3 Evaluation metrics

In this work, we employ macro-average classification accuracy as described in Section 2.6 to evaluate the performance of the different models. Also, we estimate a confidence interval (as described in Section 5.4.3) by calculating the standard error of the different measures across n folds.

6.2.4 Forced pronunciation voting strategy

In Section 5.5.4, we proposed a voting strategy for computing source language of a word when the final LID output of a single word prediction contains different Language IDs, we refer to this as 'logprob sum'. In this section, we propose another approach where the final LID string is forced to give a single-language prediction. This refers to the prediction of a single string, where this string consists of repetitions of a single (unique) LID.

This approach helps resolve ambiguity observed in 'log probability' voting by making sure the final LID string is forced to be monolingual internally. That is, a combination of language tags is not allowed. We refer to this technique as 'forced pron'.

6.2.5 Baseline

This experiment uses an SVM implementation with its parameters as discussed in Section 5.5.1 to obtain a baseline result. Using the same 10-fold cross-validated set discussed above, we extract 10% of the total training data for parameter estimation. The RBF hyper-parameters, namely the soft margin parameter (C) and kernel width ($gamma$), were estimated with $gamma$ set to the reciprocal of the sample dimensionality and a grid search used for C. Across all folds, we set the n-gram length as 3, and scale training data in the range of [0,1].

TABLE 6.3: LID results for the Multipron test set, using SVMs as the baseline techniques.

Data set	Accuracy
Multipron	76.60 ± 1.70
NCHLT 40K	74.80 ± 1.12

Results show that training on the matched data (multipron) give better performance accuracy as compared to using an unmatched data for training.

6.2.6 JSM results

Training of JSMs use similar training parameters as described in Section 5.5.2. Results are obtained for both NCHLT-inlang and Multipron models. Specifically, the following model parameters are used: model initialisation with counts, unconstrained contexts, full EM, discounting (optimised used a 10% hold-out set, and folded back post training), and a model-order of $M = 8$. For this experiment, both 'logprob sum' and 'forced pron' voting techniques are considered for estimating word LID across sets. Table 6.4 displays the results obtained on 10-fold cross-validation of the same data sets analysed in Section 6.2.2.

TABLE 6.4: LID results for the Multipron test set, using different training data sets and JSM-based techniques.

Data set	Technique	Accuracy
NCHLT 40K	logprob sum	75.80 ± 1.07
NCHLT 40K	forced pron	76.00 ± 0.99
Multipron	logprob sum	76.10 ± 1.34
Multipron	forced pron	78.60 ± 1.61

In this experiment, for either of the corpora, we observe that 'forced pron' outperforms the 'logprob' technique, with a small but consistent percentage. Also, we observe that the in-domain (Multipron) model outperforms the out-of-domain model trained on a far larger dataset. We obtain a relative error reduction of approximately 3% over the out-of-domain model, despite the fact that the out-of-domain model is based on a dataset that is 44 times bigger than the in-domain set.

In order to measure the accuracy with which the two models represent the underlined data set, standard error is used. A sample average of 78.60% deviating from the actual mean of the set by 1.61% represents a high standard error for a model trained on an in-domain set. This significant deviation could be associated with the size of the training set. With the 'forced pron' technique, the improvement from SVM results (Section 6.2.5) can be seen.

6.3 Using JSMs for corpus development

After successfully applying JSMs to proper names in Section 6.2 above, the proposed technique is applied to word LID tagging during corpus creation.

In this section, a newly developed corpus for South Africa proper names is introduced. A summary of the design, collection and analysis of a South African directory enquires (SADE)[1]

[1]This corpus was developed in collaboration with Jan W.F. Thirion, Charl van Heerden and Marelie H. Davel and partially published in [136]. The author's main contribution to the corpus involved language tagging all the words present in the corpus, and all aspects related to language identification'

corpus is presented. Further details on data collection, annotation and corpus verification are discussed in more depth in [136]. The focus is on word LID, which aims to associate each word obtained during prompt selection with its most probable language of origin. We experiment with three different techniques (see Section 6.3.2) to produce preliminary language tags for all words. To evaluate the efficiency of the three tagging techniques, we compare performance of the tagging strategies in Section 6.4.1.

6.3.1 Corpus development process

In this section, various steps involved in building the SADE corpus are summarised. The SADE corpus was developed specifically for the directory enquiry application domain of South Africa. The corpus contains audio samples from multilingual speakers producing different proper names and reflects a range of scenarios a directory enquiries system could encounter. This corpus is the first of its kind in South Africa in such an application domain (directory enquiry domain). The corpus further supports multilingual pronunciation modelling research with different meta-information.

6.3.1.1 Prompt data

Data prompts were selected from publicly available and frequently occurring names from a combination of various Internet queries, as well as personal names from a local tertiary institution's (North-West University) academic registers and volunteers. Lists of prompt data were constructed into 80 unique prompt sheets for a speaker size of 40 individuals contributing 400 utterances each. Each speaker was mandated to take part in two sessions. Prompt sheets were generated to contain 37% unique prompts and 63% overlapping ones.

6.3.1.2 Collection platform

Audio recording was done over a telephone channel, recorded in 8kHz sampling rate, 16-bit linear PCM format. The corpus collection platform consisted of an Asterisk interactive voice response system using an ISDN telephony card. In addition, this process incorporated an off-line quality control mechanism, which aligned every recording transcription with the prompt transcription using PDP to estimate a score value. In turns, the score values obtained were used to verify the accuracy of a recorded text.

6.3.1.3 Speaker selection

Speech data was collected from native speakers from Gauteng, North-West and Western Cape, who were comfortable with speaking and reading the language. This phase aimed at recording 10 speakers per language. The speakers were chosen based on a balanced mix of genders and only adults in the age range of 18 to 60 were included.

6.3.1.4 Data collection protocol

This protocol contains information where all the pre-recording, recording and post-recording requirements are defined in order to ensure that high quality data can be collected. A few of the processes contained in the protocol include:

- A prompt sheet emailed to all the participants prior to recording to do a rehearsal and identify words that were alien to them and difficult to pronounce.
- All participants were required to sign a consent form
- As discussed earlier in Section 6.3.1.1, the prompt sheet forwarded to each participant contained session information (two sessions in all), such as the session ID, sheet number, speaker number and a modulo-10 checksum. Session IDs allow identification of invalid session IDs entered during a session, while the modulo-10 checksum prevents errors by the participant during the data entry phase.

6.3.1.5 Data annotation process

These processes describe the data annotation tags that are expected to be useful during pronunciation modelling. The annotation tags were classified as meta-information, which included:

- Utterance verification: This is to identify whether the collected audio matches the prompt provided to the respondent.
- Pronunciation modelling: Phonemic pronunciation of the word using SAMPA representation is generated and verified.
- Intended language: Also referred to as the 'pronunciation language', this is the specific language for which the pronunciation conventions most closely match the pronunciation produced.
- Word LID: This aims to associate each word with it most probable language of origin.

The names contained in the corpus are not only person names, but also include the names of songs, restaurants and places, for example. These often consist of phrases (such as '*The Hillside Tavern*'). The resulting set of words is therefore a mixture of names and some generic words in isolation. Note that individual words are extracted explicitly from phrases found in the corpus. For further information on items 1 to 3 of the data annotation process, refer to [136]. The next section focuses on word LID, which aims to identify the original language community in which a word was first used, as well as languages into which it has been incorporated.

6.3.2 Word language identification

Identifying the 'source language' of a word is a surprisingly difficult tag to define unambiguously. In most multilingual societies, code-switching (whereby words from other languages are embedded in the primary or 'matrix' language) is a frequent phenomenon [169]. Some words then become incorporated in the primary language over time, often also changing their spelling and/or pronunciation to fit the conventions of the primary language better. This linguistic incorporation process – from pure code-switching to loaned words to full integration – is a gradual one, with no strict boundaries between events. In this chapter we refer to the 'source language' of a word (based on the earlier definition in Section 1.1 with examples) as the most likely language from which a word originated - this means that the word was first used and typically follows the spelling system of that specific language. When tagging word language, we do not consider frequently occurring code-switched words as being part of the primary language; however words that are fully integrated into the language are included.

Specifying the correct source language of words is a matter of opinion especially for loaned words, with individual linguists differing in opinion. For the current task, we tended towards including more rather than fewer languages where there was uncertainty.

In this work, we experiment with three different techniques to obtain preliminary language tags for all words. More details are given below. The initial language tags produced from the different tagging techniques are compared with each other. Words for which language tags disagree are flagged for manual verification.

6.3.2.1 T-LID using existing word lists

In this work, the first approach to the T-LID technique is based on existing lists of known words in South African languages. The lists of words are extracted from the original prompts and evaluated against the words in existing word lists (the NCHLT text corpora, as well as public word lists obtained from the internet - assumed to be less accurate) and existing dictionaries (specifically, the Lwazi and NCHLT dictionaries [168]). As the dictionary-based word lists tend

to be more accurate, words are first evaluated against the more accurate word lists. Only if the word is not found in these, it is then further evaluated against the less accurate word lists.

6.3.2.2 T-LID using JSMs

The second approach uses JSMs [47] to perform the T-LID task, as discussed earlier in Section 5.2. The T-LID task is recast as a G2P task, and the standard JSM training process is applied. This experiment adds word boundary markers to the training and testing data. In addition, log-probability voting is employed to select the final source language of a word.

Training data is obtained from the NCHLT-inlang [168] dictionaries, providing 15 000 unique words per language. Data is preprocessed by removing words with fewer than two characters, converting all characters to lower-case and removing any loan words that are easily identifiable based on any foreign characters included in the word. In a final round of preprocessing, any multilingual words (the same word included in more than one language list) are removed. Table 6.6 lists the number of words per language in the data corpus after preprocessing and multilingual word removal.

From this data set shown in Table 6.5, a balanced corpus is selected as training data shown in Table 6.6, while randomly extracting 150 words per language as development set and folded back after training. At 1 650 words, this is sufficient for discount estimation, based on the analysis in [47]. Note that specified training set sizes include the development set and indicate the number of training samples *per language* as well.

Language	Word counts	Average word length
Afrikaans	14 413	11.81
English	13 662	7.61
isiNdebele	10 249	11.74
Sepedi	14 285	9.25
Sesotho	9 896	9.87
Siswati	13 289	11.79
Setswana	10 472	10.21
Tshivenda	12 916	9.33
isiXhosa	12 814	10.76
isiZulu	8 690	11.15
Xitsonga	13 028	9.63

TABLE 6.5: Word counts per language in the NCHLT corpus after preprocessing and multilingual word removal.

This implementation of JSMs produces a single T-LID prediction for each word. In the next chapter, a variant of the JSM technique that produces multiple language candidates is reviewed.

Language	Word counts	Average word length
Afrikaans	8 690	11.59
English	8 690	7.78
isiNdebele	8 690	11.40
Sepedi	8 690	9.35
Sesotho	8 690	9.80
Siswati	8 690	11.80
Setswana	8 690	10.51
Tshivenda	8 690	9.12
isiXhosa	8 690	10.76
isiZulu	8 690	11.00
Xitsonga	8 690	9.52

TABLE 6.6: Training data extracted from the NCHLT dictionaries, after processing.

6.3.2.3 T-LID using web information

In the third approach, we experiment with two different techniques using web information, referred to here as 'Google Detect' and 'Google Translate'. Both techniques employ the Google translation systems to identify language identity for those languages that have sufficient web presence.

Google Detect[2] uses the built-in functionality provided by the Google Translate API [3] to guess the LID of any input string provided by the user. For the current task Google detect is used to produce a single T-LID prediction for each word.

The 'Google Translate' technique first guesses the LID of a word and then uses one or more of a set of 'proxy languages' to determine whether the word is translatable. As described earlier in Appendix A, proxy languages are typically world languages such as English, French or Spanish: any language count can be selected. If the number of times the word changed when translated to proxy languages exceeds a threshold, this is considered a good indication that the word is indeed in the original language. While a low count of translated word or untranslated word does not confirm that such a word does not exist in the original language, success provides a strong indication that it does.

In summary, these two web-based techniques, although benefitting from enough data, are only applicable to languages that have a sufficient web presence.

[2] \https://cloud.google.com/translate/v2/using_rest

[3] Google Translate API was developed by Google as a proprietary application based on statistical machine translation

6.3.3 Manual review and validation

The steps listed below summarise ways in which final SADE word lists with their corresponding language tags were generated. During the course of the project, language identities were manually reviewed and corrected based on the following reasons:

- Initially, all words not found in existing word lists were classified using the JSMs and forwarded for a preliminary manual review. This was performed by a single language practitioner who was able to identify obvious mistakes but who was not an expert in all languages occurring in the corpus.
- In the next stage, results obtained from the above techniques (word-list, JSMs and web information) were compared: any words that had not been previously verified, or for which at least two of the proposed techniques disagreed in-terms of T-LID tag, were flagged for further manual confirmation or correction. The word list was, therefore, partitioned into a 'Phase 1 tagged' and a 'Phase 2 tagged' list. The Phase 1 list (words where at least two of the above techniques agreed on language tags) was incorporated as is into the corpus, while the Phase 2 list was sent for manual review to various practitioners (where language experts represented all 11 South African languages).

The final corpus was tagged using the combination of Phase 1 and Phase 2 results.

6.3.3.1 Manual validation

Once the corpus had been completed, 600 words were sent for manual validation in order to determine the accuracy of the entire process. Three hundred (300) words were selected from each of the Phase 1 and Phase 2 tagged lists and forwarded to volunteers for review. Again, the focus was on identifying the accuracy of tags containing South African languages.

Good accuracies were observed, as shown in Table 6.7. Standard definitions of precision, recall and F1-measure are used to estimate final LID accuracy based on manual verification, as discussed in Section 2.6. Of the 600 words, results were calculated for 582 words only, as verifiers were not able to tag 18 of the test words. These words contained idiosyncratic spellings (for example, the word 'kream'), making LID difficult, even for human verifiers.

Tagged list	Words in test set	Precision	Recall	F-measure
Phase 1	300 (291)	99.69	95.54	97.57
Phase 2	300 (291)	99.50	99.50	99.50

TABLE 6.7: Final LID accuracy estimate based on manual validation.

6.4 Corpus analysis

6.4.1 LID technique comparison

In this section the performance of the different LID tagging techniques is compared in order to evaluate the efficiency of the overall LID tagging strategy. The results are evaluated using two different scenarios, particularly excluding web-based techniques in one of the scenarios where languages with a sufficient web presence are manageable. The two scenarios include:

- Comparing all techniques on only three languages - those languages with sufficient web presence.
- Comparing all but the web-based techniques on the 11 SA languages.

The term '3- and 11-partition-set' will be used to identify the two above-mentioned scenarios independently. Standard precision and recall are used, as well as the F-measure, to report accuracy for the above two scenarios 2.6.

Table 6.8 shows the language distribution in the reference set across words per tagged list for each partition set. Each reference set is randomly re–partitioned to contain a balanced word-list across languages.

In phase 1, using the reference set for 11 languages, as shown in Table 6.8, a very low count for the isiNdebele and Siswati languages is observed. In the subsequent experimental analysis (under 11-partition), the two languages are excluded to enable accurate comparison.

6.4.2 Three-language evaluation

Web-based automated techniques support three South African languages, namely Afrikaans, English and isiZulu (at the time of performing this experiment). These three languages represent the target languages on which all the automated techniques will be compared. Prior to computing final results, all reference tags were preprocessed using the automated technique prediction sets by removing language tag(s) of words that did not exist in the target set; that is, corresponding language tags per word could contain only languages in the target set.

SA Languages	3 languages		11 languages	
	$phase1_3$	$phase2_3$	$phase1_{11}$	$phase2_{11}$
Afrikaans	1245	555	1245	555
English	6439	802	6439	802
Isizulu	121	357	121	357
Sesotho	-	-	69	272
isiNdebele	-	-	9	148
Tsonga	-	-	27	108
Siswati	-	-	4	113
Venda	-	-	36	76
Sepedi	-	-	49	249
Setswana	-	-	69	534
Xhosa	-	-	26	306

TABLE 6.8: Language distribution across words in the reference set per tag list before repartitioning. Reference sets are based on 3 and 11 SA languages.

After preprocessing, across automated technique prediction sets, cases are observed where words contain empty language tags. In such a case, it is assumed that the particular technique could not predict the language tag of such a word. Hence, avoiding bias in the techniques after preprocessing, two sets of results in one table are shown.

In each table, precision, recall and the F-measure are shown by comparing hypothesised tags with reference tags across the different techniques. Since some of the hypothesised tags may be in languages outside the set of three currently being considered, two separate cases are analysed: (1) all words present in the reference set are used, irrespective of whether the hypothesised tags include any of the relevant languages. We refer to this case as 'all'; (2) only words with relevant language tags in the hypothesised sets are retained during analysis (in both the hypothesis and reference sets). This is referred to as the 'found' case. As expected, both sets of comparison produce the same precision but different recall values. Tables 6.9 and 6.10 show two sets of results obtained across all automated techniques.

P_{all}, R_{all}, and $F1_{all}$ represent precision, recall and F-measure values, respectively, computed on only tagged words returned by a specific automated technique (as shown in the 'Techniques' column) against all tagged words contained in the reference set (case 'all'). R_{found} and $F1_{found}$ represent recall and F-measure values respectively, estimated on only tagged word lists contained in a specific automated technique against identical word lists in the reference set (case 'found'). The columns giving 'Words in test set' represent the total number of unique words present in the reference set. 'Words with lang tag' represent the total count of words with language tag per automated technique.

In Tables 6.9 and 6.10, results obtained from the two partitioned sets demonstrate performance variabilities of each of the automated techniques. In Table 6.9 (containing mainly common names), the non-web-based techniques, namely JSMs and word-list, show higher performance

Technique	Words in test set	Words with lang tag	P_{all}	R_{all}	R_{found}	$F1_{all}$	$F1_{found}$
JSM	359	313	89.457	74.271	86.420	81.160	87.912
Word-list	359	263	69.444	66.313	88.968	67.842	78.003
Google-Detect	359	300	78.333	62.334	74.841	69.424	76.547
Google-Translate	359	326	65.525	76.127	84.412	70.430	73.779

TABLE 6.9: Comparing automated techniques tested on three South African languages using data sets from 'Phase 1' tag list as reference labels. Words with fewer than two characters length are excluded

Technique	Words in test set	Words with lang tag	P_{all}	R_{all}	R_{found}	$F1_{all}$	$F1_{found}$
JSM	993	533	69.04	33.06	61.23	44.72	64.90
Word-list	993	793	56.37	62.44	76.12	59.25	64.77
Google-Detect	993	733	83.22	54.81	75.12	66.09	78.97
Google-Translate	993	568	71.14	40.97	69.62	52.00	70.37

TABLE 6.10: Comparing automated techniques tried on three SA languages using data set from 'Phase 2' tag list as reference labels. Words with fewer than two characters length are excluded

accuracy than the two web-based techniques. The JSMs technique compared to the word-list technique, proved to be decisive on the phase 1 tagged lists, even though the resulting model used for testing was trained on a small set of generic words.

As we transit from common to more complex words (phase 2 tagged list) in Table 6.10, it is observed that the web-based technique outperformed other non-web-based techniques in terms of accuracy. This improvement in performance could be associated with the number of available training samples used for model building, and probably the diversity of those words. The JSMs technique obtained a very low precision of 69.04% when tested against all words in the reference set (while penalising the technique on words with empty language tags). The same technique shows an improvement in performance when tested on only word lists where language tags are present with an F1-measure of 64.90%.

6.4.3 Eleven-language evaluation

The automated techniques were tested on 11 South African languages. These languages represent the target set on which non-web-based techniques will be tested. This evaluation excludes both web-based techniques because of the unavailability of the other eight South African languages, and employs the same preprocessing steps and analysis as discussed above in Section 6.4.2.

As before, the performance accuracy of the non-web-based techniques were reported on two separate reference sets where (1) evaluation is performed on all existing word lists contained in each automated technique against the same word-list contained in the reference set ('all');

Technique	Words in test set	Words with lang tag	P_{all}	R_{all}	R_{found}	$F1_{all}$	$F1_{found}$
JSM	233	226	93.36	87.55	90.17	90.36	91.74
Word-list	233	209	52.76	87.14	96.77	65.73	68.29
JSM	80	80	88.75	85.54	85.54	87.12	87.12
Word-list	80	80	53.15	91.57	91.57	67.26	67.26

TABLE 6.11: Comparison of automated techniques tested on 11 South African languages using data set from 'Phase 1' tagged list as reference set. Words of fewer than two characters and language tags such as isiNdebele and Siswati were removed from the reference and method sets.

Technique	Words in test set	Words with lang tag	P_{all}	R_{all}	R_{found}	$F1_{all}$	$F1_{found}$
JSM	644	644	62.73	37.55	37.55	46.98	46.98
Word-list	644	365	53.61	37.27	57.78	43.97	55.62
JSM	258	258	52.17	53.63	53.63	52.89	52.89
Word-list	258	258	63.95	30.73	30.73	41.51	41.51

TABLE 6.12: Comparison of automated techniques tested on 11 South African languages using data set from 'Phase 2' tagged list as reference set. Words fewer than two characters and language tags such as isiNdebele and Siswati were removed from the reference and method sets.

(2) evaluation is performed on word lists with language tags while making sure tag counts are balanced across 11 languages ('found').

In Table 6.11, results show that the JSMs technique outperforms the 'word-list' technique on less complex words with a huge difference in accuracy. On the contrary, in Table 6.12, a different pattern of results is obtained where classification accuracies of the two non-web-based techniques are very close. This drop in performance when evaluating on the phase 2 word-list could be associated with poor word language structure or loan words.

Furthermore, when evaluating results on the second reference set in Tables 6.11 and 6.12, the JSMs technique gives better identification performance over the 'word-list' technique. That is, while JSM technique gives a balanced value between recall and precision, the 'word-list' technique shows high precision with low recall.

6.4.4 Conclusion

In this section, we applied the new JSM technique to proper names, and found that the performance gain over an SVM-based technique was retained. The new JSM technique was combined with a number of others – two web-based and a word-list based technique – to develop a larger corpus of language-tagged words.

The performance of the different automated LID techniques using the final language tags was discussed and analysed. The analysis shows that combining different LID techniques gives good classification accuracy for LID tagging. For LID of complex names (notably loan words), both web-based techniques outperformed the non-web-based techniques, presumably because the web-based techniques have encountered many of those terms in the appropriate language context to predict the correct result given the vast amount of training data available. For LID of common words, non-web-based techniques performed equally well and could rival web-based methods.

Chapter 7

Language identification of multilingual proper names

7.1 Introduction

In Chapter 5 a method of using JSMs for LID was described and evaluated. Initially, the technique was only applied to generic words, not proper names. In Chapter 6, JSMs were applied with two other commercial classifiers to LID of proper names, and the comparative performance that can be obtained when applying the three techniques was analysed.

In all cases, only a single language tag was hypothesized per word. In reality, there are numerous cases where documents or terms (either proper names or generic words) belong to more than one language of origin. Multilingual LID is not a well-studied task, with existing T-LID techniques focusing either on identifying the single language within a portion of running text, or to a lesser extent, the language identity of isolated words. (See Chapter 2.)

In this chapter, we investigate how to use JSMs for multilingual classification of proper names. In practical terms, the question then is: given a proper name, can we accurately predict which source languages are relevant, where such names may be either mono- or multilingual? Specific consideration is given to four South African languages, using the SADE corpus.

7.2 Joint Sequence Models for multilingual T-LID

In this section, it is first described how to compute the posterior probability of a translation using the JSM algorithm and how it is relevant to the work, before extending it to the multilingual case in Sections 7.3 and 7.5.

7.2.1 Training and transcription phase

During training, as noted in Section 5.2.2, parameter choices, such as gralanguage length, discounting, m-gram model order and how models are initialised, influence the performance accuracy of the JSM model. For this work, 1-1 gralanguage alignment was used during training, which means that the minimum and maximum number of letters to LID mappings allowed per gralanguage length was 1. As models typically saturate before the 8^{th} order, an 8^{th} order model was used. Discounting was allowed during training in order to handle unseen tokens and avoid overfitting. All held-out data used for parameter estimation was folded back to the training set; see Section 5.3 for more details on data representation prior to JSMs training.

In standard JSMs, a forward algorithm is used to compute the joint probability, $p(x, k)$, of co-segmentation between a letter sequence x and language sequence k. The probability of a source sequence x is in principle determined by summing all the matching gralanguage sequences across the sequence path given in Equation 5.10. This value can also be approximated by simply taking the maximum value to produce Equation 7.1:

$$p(k \mid x) = \frac{max_{q \in S(x,k)} p(q)}{p(x)} \tag{7.1}$$

where $S(x, k)$is the set of all co-segmentations of x, and $p(q)$ represents the probability distribution over the sequences of gralanguages (all probabilities as estimated during training).

JSMs allow the ability to generate different pronunciation variants. That is, for each word, a number of pronunciation variants may be produced; the path posterior probability given the word is used to select the winning candidate among the variants (using Equation 7.1). As in the T-LID case, two of the best voting strategies was used, where; (1) a variant consists of an LID string; ambiguity is resolved either by selecting the language identifier with the highest frequency counts (within the variant) or by summing the language-specific log-probabilities internally to the word; (2) each variant is forced to be monolingual internally. That is, a combination of language tags is not allowed.

7.3 Approach

We use the JSM-based technique developed in earlier chapters, and extend it to multilingual classification of words.

Two main issues must be addressed:

1. *The data to train the JSM models.* Is it better to use matching data (names) even if this data set is very small, or better to use a significantly larger set, even if it is unmatched (generic words)?

2. *Options for extending the technique for multilingual data.* JSMs can generate variant outcomes, as well as the likelihood and posterior probability given the word, per outcome. How well do these values predict true multilingual words?

Empirical results are obtained by experimenting with different data sets and threshold values. The SADE multilingual name corpus [136] and the NCHLT *in-lang* dictionaries [167] are selected for experimentation. These data sources are described in more detail in Section 7.4. Different classification options are discussed and experimented with in Section 7.5. We first propose an improvement to the monolingual classification technique, before experimenting with different approaches to multilingual classification.

As performance measures the standard definitions of precision, recall and F1-measure are used, as discussed in Section 2.6. When comparing threshold-based techniques, as used in this work, the ROC is evaluated by plotting the TPR (eq. 2.2) against the TNR (eq. 2.10) rather than against the false positive rate, as also often done. The reason behind this decision is that we are interested in how to maximise the performance of the classifier in order to depict a relationship of few wrongly rejected against few wrongly identified rather than depicting the relative trade-off which is the inverse relationship between benefit (TPR) and cost (FPR).

7.4 Data

Two data sets are used in this analysis: the SADE corpus [136] to obtain tagged samples of multilingual names, and the NCHLT dictionaries [167] to obtain samples of generic monolingual words in matching languages. LID performance is analysed for four South African languages, namely Afrikaans, English, isiZulu and Sesotho.

As initially discussed in Section 6.3.1, the SADE corpus was developed to improve directory enquiry applications in South Africa. The corpus contains audio samples from multilingual speakers producing different proper names, and reflects a range of scenarios directory enquiries systems could encounter. SADE data prompts were developed using publicly available names from a combination of internet queries, personal names from a local tertiary institution (North-West University) and volunteers. Each name was annotated with a number of tags, including the most probable source language. Only the word lists, and the LID tags of the $v1.1.$ corpus in the current analysis were used.

The NCHLT dictionaries were developed in parallel with the NCHLT speech corpus [168]. For this work, only the word lists from the dictionaries, consisting of 15 000 unique words per language, were used. These were estimated to be frequent words based on available corpus counts, and LID accuracy was verified using automated spell checkers and language practitioners [167]. The same edited lists as used in Section 5.4.1 were used, where a second round of higher precision, lower recall spell-checking was performed for three of the languages (Afrikaans, isiZulu and English). These lists do not contain multilingual words.

7.5 Analysis and results

We first analyse the distribution of multilingual words in the SADE corpus and create a suitable train and test set (Section 7.5.1) before applying JSMs to the monolingual test case (Section 7.5.2). The LID technique is then extended to cater for multilingual words in Section 7.5.3.

7.5.1 Data analysis

While the SADE corpus contains words in a large variety of languages, only the subset consisting of words tagged with the four target languages (Afrikaans, English, Sesotho and isiZulu) was used. Per language, the number of unique words, average word length and total character count are displayed in Table 7.1. The actual data set is almost exclusively bilingual; Afrikaans and English definitely dominate. This data set presents the best available newly developed corpus on multilingual names in South Africa.

TABLE 7.1: SADE corpus: language distribution and word statistics.

Language	Word count	Average word length	Character count
Afrikaans	1 050	6.9	7 308
English	6 634	7.4	48 733
Sesotho	465	7.8	3 612
isiZulu	458	8.1	3 689

While the majority of the words are monolingual, a significant percentage of them (9.3%) were identified as multilingual. The distribution between mono- and multi-lingual words per language in Table 7.1 is shown in Table 7.2. As before, the number of unique words, average word length and total character count are displayed.

Because of the small word count of the isiZulu language, we randomly selected a monolingual subset, per language, that equals 401 unique words. For verification purposes, language practitioners were asked to review the newly extracted mono-lingual word-list with its corresponding LIDs within the specific target language together with the multilingual word-list shown in Table 7.2. The results obtained from language practitioners were analysed and observed few changes

TABLE 7.2: SADE corpus: mono- and multi-lingual distribution and word statistics.

	Word Count		Average word length		Character count	
Language	Mono	Multi	Mono	Multi	Mono	Multi
Afrikaans	449	601	8.5	5.8	3 808	3 500
English	5 980	654	7.5	5.7	4 974	3 759
Sesotho	411	54	8.1	4.9	3 345	267
isiZulu	401	57	8.4	5.3	3 387	302

across both words and LIDs. In order to create a balanced training set, a subset of 321 per language was randomly selected while retaining the remaining words as testing data without restricting their range (both mono- and multi-lingual subset).

Most of the multilingual words in the corpus are bilingual, with a very small set of three-lingual and a single four-lingual word ('sale'). Examples of three-lingual words include 'tutu', 'tone' and 'pole'. The exact distribution of words is shown in Fig. 7.1. Note that the figure is restricted to 800 words (on the y-axis), even though the first bar exceeds this number (with a value of 7241 words). Table 7.4 shows language identities of bilingual words in the SADE test set. For example, a word count of 314 represents the total number of words that exist in Afrikaans and English only, 15 represents the total word count that exists both in Sesotho and isiZulu.

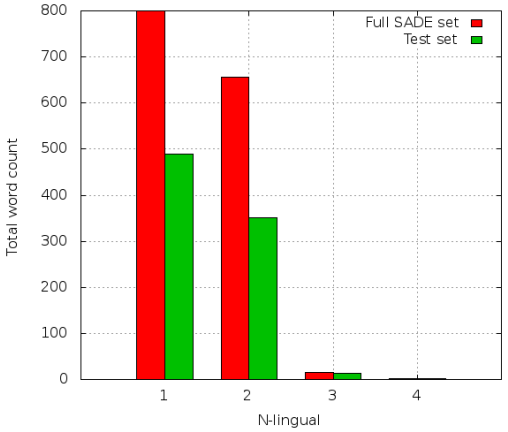

FIGURE 7.1: Number of one- two- three- and four-lingual words in the SADE full and test data sets.

The greatest possibility for confusion exists between English and Afrikaans words, as can be seen from Table 7.4, which lists the number of bilingual words in the SADE test set.

TABLE 7.3: Number of mono- and multilingual words in the SADE train and test partitions.

	Training set		Test set	
Language	Mono	Multi	Mono	Multi
Afrikaans	321	-	155	329
English	321	-	137	348
Sesotho	321	-	98	34
isiZulu	321	-	99	34

TABLE 7.4: Language identities of bilingual words in the SADE test set

Languages		Word count
Afrikaans	English	314
Sesotho	isiZulu	15
English	isiZulu	13
English	Sesotho	7
Afrikaans	Sesotho	2

7.5.2 LID of monolingual names

For the generic model, we use a 40K-word subset of the NCHLT data. The objective is to make use of an already existing model based on previous work done in [104]. For comparative purposes, the same statistics as shown for the SADE data in Tables 7.1 and 7.2 are displayed for the NCHLT data in Table 7.5. This provides two different training data sets: a small (1 284) SADE set, and a large (40K) NCHLT set.

TABLE 7.5: NCHLT 40K subset: language distribution and word statistics.

Language	Word count	Average word length	Character count
Afrikaans	10K	10.5	104 531
English	10K	7.9	79 768
Sesotho	10K	8.3	82 537
isiZulu	10K	9.4	93 617

We first obtain results using the identical technique as described in Section 5.5.4. Specifically, the language-specific log-probabilities are summed internally to the word. This technique (referred to as 'logprob sum') produced the most accurate results in [104]. An extension to this technique is also reported, as discussed in Section 6.2.4, where each variant is forced to be monolingual internally. That is, a five-letter word will only be tagged as 'EEEEE' or 'ZZZZZ': a combination such as 'EEZZZ' is not allowed. In the four-language task, this means each word would produce at most four variants, each with an associated posterior probability (see Equation 7.1). These posteriors are then used to select the winning candidate, and any mixed variants are simply discarded. This technique is referred to as 'forced pron'.

TABLE 7.6: LID results for the SADE monolingual test set, using different training data sets and JSM-based techniques.

Data set	Technique	Accuracy
NCHLT 40K	logprob sum	77.96
NCHLT 40K	forced pron	78.16
SADE	logprob sum	79.39
SADE	forced pron	81.02

In Table 7.6 we report on results. Interestingly, the much smaller SADE training data clearly fits the test data better and produces more accurate results. The new 'forced pron' technique also shows a small but consistent improvement, across all measures.

7.5.3 LID of multilingual names

Using the SADE models and the 'forced pron' LID technique, two approaches are evaluated for determining when a word may be truly multilingual:

- We define an absolute threshold based on the posterior probability: any variants with a posterior probability higher than the threshold are accepted as additional source languages. This technique is later referred to as 'absolute posterior'.
- We define a relative threshold based on the log likelihood: any variants with a relative likelihood within the range of the best variant are accepted as additional source languages. This technique is later referred to as 'relative likelihood'.

In both cases, the best performing variant is automatically selected: thresholds are only used to determine whether more than one source language may potentially apply.

The ROC curves for both these methods are displayed in Fig. 7.2. (As the graph approaches the top right corner, the more accurate the technique.) The full test set from Table 7.3 is used. As expected, the two techniques provide very similar results, with optimal F_1 scores of 79.99% and 79.85% obtained, by 'absolute posterior' and 'relative likelihood', respectively.

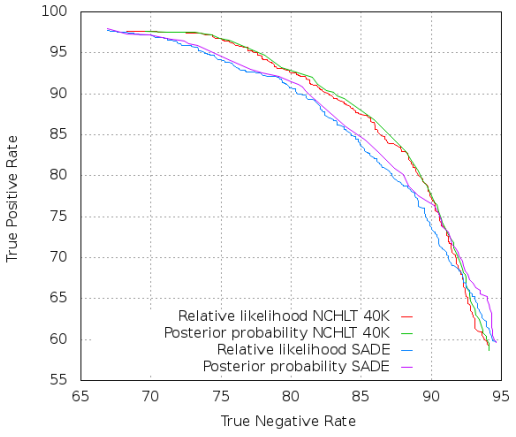

FIGURE 7.2: ROC curves for the SADE combined test set comparing the 'absolute posterior' and 'relative likelihood' approaches.

From the ROC curve in Fig. 7.2 it is clear that the techniques perform as expected, but the question is how well they perform. In order to answer this question, we consider a simple baseline whereby we select first the single best variant ('top-1'), and then the two best variants

('top-2'). In the first case all words are therefore treated as if they are monolingual, and all additional source languages will automatically be 'false rejects'. In the second case, a large number of superfluous source languages will be hypothesised, and many 'false accepts' accepted. The results from these four methods are compared in Table 7.7.

TABLE 7.7: Comparison of different multilingual classification approaches using the SADE combined test set.

Approach	Model	Recall	Precision	F-measure
top-1	NCHLT 40K	58.67	84.78	69.35
top-1	SADE	59.56	86.07	70.40
top-2	NCHLT 40K	91.89	66.51	77.17
top-2	SADE	91.98	66.84	77.42
relative likelihood	SADE	84.60	75.60	79.85
absolute posterior	SADE	84.80	75.70	79.99
relative likelihood	NCHLT 40K	86.00	77.60	81.58
absolute posterior	NCHLT 40K	86.20	77.80	81.79

From Table 7.7 it is clear that this is a difficult task: the first baseline result ('top-1') produces an F-measure of only 69.35% and 70.40% using the NCHLT 40K and SADE models, respectively. The second baseline approach ('top-2') shows improvement over the 'top-1' technique, where a recall value of 92% (obtained on both NCHLT 40K and SADE models) means a large number of the correct tags were correctly identified. However, this high recall value comes at a cost: specifically a precision that falls to 67%. As more tags are accommodated based on the number of variants, there is a high likelihood of also identifying the wrong tags. In contrast to this, the 'top-1' approach achieves a better precision value, but with lower recall values.

The baseline results are improved when we leverage the trade-off between the 'false rejects' and 'false accepts' related to recall and precision respectively. For both approaches, an optimum point across different threshold values where TNR equals TPR is selected, and the recall and precision values are computed. The optimum threshold value where TNR equals TPR for 'relative likelihood' is -3.17, while the 'absolute posterior' is 0.026. Converting the optimum threshold value of 'relative likelihood' to a probability form produces a value of 0.042, which is (as expected) close to the optimal value returned by the 'absolute posterior' approach.

Interestingly, we observe that using the NCHLT 40K model on 'absolute posterior' produces the best performance for multilingual classification of proper names. However, this contradicts the initial observation with regard to monolingual classification, where the SADE model produced the best performance. This shows that we cannot judge the performance of a multilingual classifier by considering only its monolingual classification accuracy.

7.6 Conclusion

This work focused on the multilingual classification of proper names, a task that has not been well studied to date. Although the work is targeted at four South African languages, the techniques used are language-independent.

First, we proposed an improvement on the monolingual classification of words using JSMs, building on earlier work described in Section 5.5.2 By forcing JSMs to produce output strings that are associated with a single language, we obtain more trustworthy posteriors to analyse. This approach was compared to the best available to date ('logprob sum'), and an improvement was found in the F-measure from 79.39% to 81.02%, training and testing on the same set of proper names.

In order to classify proper names as multilingual, we experiment with two baseline methods ('top-1' and 'top-2') where both approaches produced a tradeoff between recall and precision. To strike a balance between the two metric values, two new techniques ('relative likelihood' and 'absolute posterior') were proposed. While the difference between the two new methods is statistically insignificant, both outperform the baseline, with 'absolute posterior' using the NCHLT models producing an F-measure of 81.79%.

Finally, we observe that the identification performance on monolingual proper names does not necessarily translate to similar performance for multilingual classification. For the LID of monolingual names, models trained on SADE outperformed models trained on NCHLT 40K with an F-measure of 81.02% (compared to 78.16%). For LID of multilingual names, the NCHLT 40k models produced better results than SADE. (81.79% compared to 79.99%).

In conclusion, it has been shown that even though LID of multilingual proper names is a challenging task, an adapted version of JSMs provides good classificaton accuracy.

Chapter 8

Implications for proper name recognition

8.1 Introduction

In Chapters 6 and 7, it was shown how to apply JSMs to LID of proper names. It was also shown in Section 7.5.2 how JSMs generalise well from a small training set, which is important for under-resourced languages. In Section 7.5.3, methods to apply JSMs to multilingual classification of proper names were investigated.

In this chapter the aim is to investigate the implication of using the newly developed LID technique for proper name recognition. Specifically, we want to understand the implications of LID results on the performance of an ASR system. In Section 8.2 the implication of LID accuracy is first analysed from a G2P perspective. If LID tags are used when auto-generating pronunciation dictionary, how well do these dictionaries match manually developed dictionaries? The same dictionaries are then used to train and evaluate ASR systems in Section 8.3.

8.2 G2P analysis

During the G2P analysis, the word list, reference pronunciation dictionary and word-based language tags associated with each corpus are used. Audio information is only utilised during the ASR analysis.

Two corpora introduced previously, Multipron (Section 3.2) and SADE (Section 6.3.1), are used. For each corpus we generate four dictionaries automatically, using G2P prediction. The fifth dictionary was manually generated during corpus development and is referred to as the reference

dictionary. It represents the best available dictionary that exists in each corpus. The method used to generate such dictionaries will be discussed in Section 8.2.1. For automatic language generation, four use cases are considered. In each case, the most probable pronunciation is produced using language-specific G2P. The only difference among the dictionaries relates to which language is selected when producing the G2P pronunciation. The following options are experimented with:

- Language-specific G2P-generated dictionary based on each word's true source language.
- Language-specific G2P-generated dictionary based on LID 'single-language' prediction.
- Language-specific G2P-generated dictionary based on LID 'multiple-language' prediction.
- Language-specific G2P-generated dictionary where we assume that each word originates from all four target language sets (Afrikaans, English, Sesotho and isiZulu).

For both corpora, the four language-specific G2P-predicted dictionaries are compared with the reference dictionary.

8.2.1 Reference dictionaries

In earlier work (Sections 6.2 and 6.3.1), only the word lists and language tags of the two corpora were used. Here the pronunciation dictionaries are used for the first time. We first explain how the corpus dictionaires were created in Section 8.2.2, before discussing how the corpus dictionaries were adapted for use in this set of experiments.

8.2.2 Initial corpus dictionaries

The initial corpus dictionaries for the SADE and Multipron corpora were obtained in different ways. For the Multipron analysis, this work uses the pronunciation dictionary described in [127]. The original Multipron corpus described in Section 3.2 constitutes a combination of *name-surname* as a single word with their corresponding pronunciations, which were manually transcribed per audio clip. Further work on the corpus in [127] created a new pronunciation dictionary where name pairs are split into a single entity according to the language of origin of the speaker and name.

In order to split name pairs, the authors used dynamic programming (DP) for G2P alignment on the original Multipron dictionary[170]. For the DP process, authors used a name's orthography (name pairs joined with an '=' symbols) and its transcription as reference and observation

respectively. The DP alignment was based on the Needleman-Wunsch algorithm that uses log-likelihood probabilities for scoring. After splitting the entire aligned sequence by mapping the '=' symbol to a space delimiter in the transcription, additional manual checks were carried out, such as word boundary effects (where same grapheme/phoneme found at the *firstname* and *lastname* boundary were corrected), all /l/ phonemes initially separated from first names were restored.

The SADE pronunciation dictionary, on the other hand, was obtained semi-automatically by combining manual verification and correction with G2P prediction. Initial name pronunciations were obtained using G2P rules extracted from already existing resources. In order to verify incorrect pronunciation, a phone-based dynamic programming score (PDP) [171] was used. These scores are based on speech recogniser output, and are calculated using either of two techniques, namely a flat scoring matrix (assuming equal cost for all phone substitutions) and a data-specific matrix (using variable weight across phone substitutions). PDP output was also used to generate cross-lingual pronunciation variants. A final round of verification was carried out on audio/transcription to flag wrong transcription and mark for manual correction [127].

8.2.3 Phone mapping

The goal of creating a phone mapping is to use common phonemes from all available corpora to obtain a final phone set that either has the same symbol or is as closely related as possible. Different conventions are used in different corpora, and these must be reconciled prior to analysis. Scenarios where corpora share the same symbol are retained. However, there are cases where it is necessary to use a different mapping strategy because of the absence of certain phonemes.

For proper comparative analysis, the reference and the hypothesised dictionaries should share a similar phone set. The phonemes contained in the hypothesised dictionaries (see Section 8.2.4) capture distinctions that are linguistically important (NCHLT-based), compared to the phone set used to obtain the two reference dictionaries that only capture distinctions humans can easily make during cross-lingual transcriptions. For example, for languages in which duration is considered important, a differentiation is made between long and short vowels (*/u:/* and */u/*, */i:/* and */i/*) and speakers tend to produce distinct samples that are either lengthened or not. On the other hand, for languages that do not put emphasis on duration, speakers may arbitrarily produce */i:/* or */i/* or a duration in between – this is then very difficult to transcribe consistently. In cross-lingual transcriptions, such language-speciic transitions are therefore sometimes omitted [134].

Therefore, there is a need to obtain a common phone set that reconciles differences across the three corpora. Two phone sets are defined:

- Ensure that the same conventions are applied to phones thereby making the sets consistent but retaining all phone distinctions, referred to as 'detailed'.
- Apply a merging approach where not all phone distinctions are utilised, referred to as 'combined'. The 'detailed' phone set will therefore produce more conservative results than the 'combined' set.

The final phone maps used based on the above two approaches are included in Appendix B.

8.2.4 Generating G2P dictionaries

Figure 8.1 shows the different steps necessary to obtain a phonemic transcription for each G2P-based dictionary described earlier. Using the original language-tagged word list, word lists are automatically extracted without their corresponding language of origin. In order to generate tags for the newly extracted list, three cases are considered:

- JSM-based single-language word list: where the JSM technique is forced to classify words as monolingual; see Section 7.5.2.
- JSM-based multi-language word list: where words are classified as multilingual using the JSM threshold approach; see Section 7.5.3.
- All-four-language word list: we assume that all words originate from all four target languages by assigning each word to all the target languages.

One question to address is selecting the training data on which the LID system should be trained. To avoid bias towards any of the corpora employed, especially during later ASR experiments, a previous model (NCHLT 40K) trained on an *NCHLT-inlang* data set was used to predict the language of origin of proper names (see Section 7.5). These data use a 10 000 word subset of the original 15 000 word lists. This list contains only unique words that are monolingual. In order to train the JSM model, model parameters used include model initialisation with counts, unconstrained contexts, full EM, discounting (optimised used a 10% hold-out set, and folded back post training), model-order of $M = 8$. To select the final word LID across sets, we use 'forced pron' voting techniques.

In order to obtain phonemic transcriptions and generate a dictionary for the different word lists, G2P conversion is performed on each list using Default & Refine [46] as a rule-based algorithm. The algorithm is trained on language-specific generic text to extract G2P rules, which are then language-dependent. Each word translation is based on language-specific G2P rules.

The same *NCHLT-inlang* data introduced in Section 5.4.1 is used to train the G2P models. This time the pronunciations are used (not only the word lists), and training uses all 15 000 available words per language. Using these models, we generate four dictionaries:

- True-word LID dictionary: G2P-based dictionary obtained from the manually tagged word list.
- Single-language LID dictionary: G2P-based dictionary obtained from the 'JSM-based single-language word list.'
- Multi-language LID dictionary: G2P-based dictionary obtained from 'JSM-based multi-language word list.'
- All-four-languages dictionary: G2P-based dictionary obtained from the 'All-four-language word list.'

As discussed in Section 8.2.3, the resulting dictionaries can only be used when mapped to a reconciled phone set. The same mappings as introduced in Section 8.2.3 are applied here.

8.2.5 Evaluation data

This particular task involves four target languages, namely Afrikaans, English, Sesotho and isiZulu. Multipron, as described earlier in Section 8.2.1, is based on four target languages. SADE, on the other hand, encompasses all 11 South African languages, from which a subset of four languages is selected. Table 8.1 shows the language distribution and word statistics for both corpora.

TABLE 8.1: SADE vs Multipron corpus: language distribution and word statistics.

	Word count		Average word length		Character count	
Language	SADE	Multipron	SADE	Multipron	SADE	Multipron
Afrikaans	1 050	252	6.9	6.9	7 308	1 743
English	6 634	522	7.4	6.2	48 733	3 232
Sesotho	465	264	7.8	7.4	3 612	1 965
isiZulu	458	254	8.1	7.2	3 689	1 839

In the SADE corpus, words with two or less characters, spelled-out or words outside the four target languages were removed from the reference and hypothesised dictionaries for proper comparative analysis. (These are typically dealt with through a separate process.)

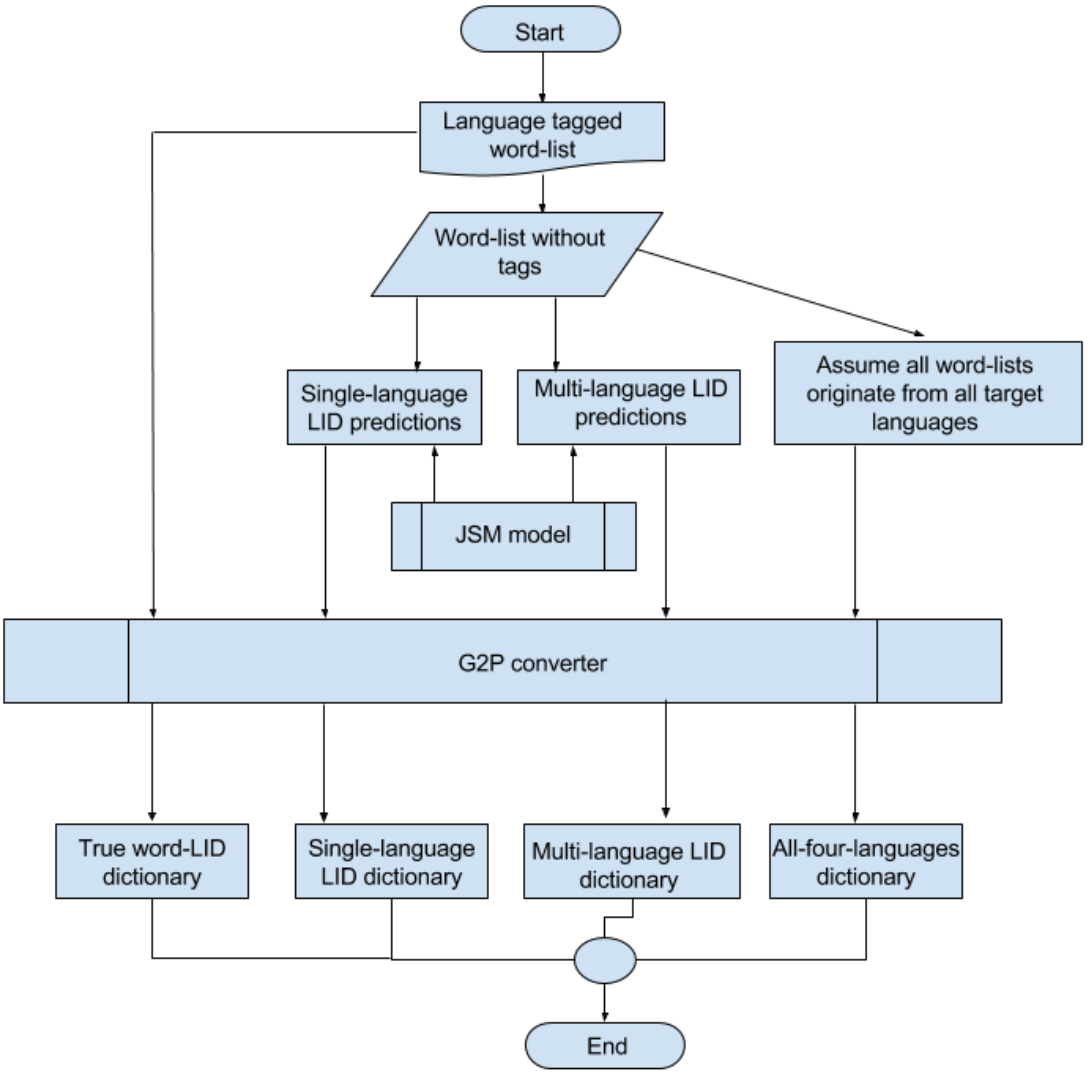

FIGURE 8.1: Process to generate the G2P-based dictionaries.

8.2.6 G2P variant-based accuracy measure

Given the pronunciation variability in the dictionaries, how do we evaluate G2P accuracy? There is no straightforward rule on how to analyse G2P accuracy for dictionaries with pronunciation variants. In this analysis, we use five metrics defined in [172], namely variant-based G2P phone accuracy (V-PA), variant-based G2P word accuracy (V-WA), matching variant percentage (MVP), single-best variant G2P phone accuracy (S-PA) and single-best variant G2P word accuracy (S-WA).

Variant-based G2P accuracy is computed per word, by evaluating each reference pronunciation against all hypothesised pronunciations (for that specific word) to obtain the best-matching pronunciation, as well as the accuracy score for that reference-hypothesis pair. These scores are then averaged across all pronunciation variants of the specific word, occurring in the reference

dictionary. A best-variant score of 1 for any given word means that there is a hypothesised pronunciation that matches the reference pronunciation, for every single pronunciation variant occurring in the reference dictionary.

V-PA and V-WA only differ on whether the accuracy score (that is averaged across all reference pronunciation variants) is a phone accuracy score (V-PA) or word accuracy score (V-WA). Both the overall V-WA and V-PA are then obtained by averaging these values over all the words in the dictionaries.

Since additional hypotheses are not penalised, variant-based G2P scores tend to increase as the number of variants in a hypothesised dictionary increases. For this reason it is useful to calculate the MVP to determine the extent to which variants are being over-generated. The MVP is calculated as the average number of variants in the hypothesised dictionary as a percentage of the average number of variants in the reference dictionary.

S-PA and S-WA are computed by obtaining only the single best accuracy per word (from the best-matching reference and hypothesis pair), and then averaging over all words in the dictionary. Similarly, S-WA is a word-based accuracy score and S-PA a phone-based accuracy score.

Given the above performance metrics, phone accuracy using two approaches: standard phone accuracy and aligned phoneme accuracy. Standard phoneme accuracy is the default approach, unless otherwise stated.

These two approaches differ as follows: If I represents insertions, C represents correct phone pairs, S represents substitutions, D represents deletions, and N represents the number of phones in the reference word, then

$$\mathrm{N} = C + S + D \tag{8.1}$$

and using equation 8.1, we can calculate:

$$\text{standard phone accuracy} = \frac{C - I}{N} \tag{8.2}$$

$$\text{aligned phone accuracy} = \frac{C}{N + I} \tag{8.3}$$

Standard phone accuracy (eq. 8.2) is typically used, but aligned phone accuracy (eq. 8.3) has the benefit that matching the hypothesis against the reference, or the reference against the hypothesis produces the same result. This is not always the case for standard phone accuracy.

8.2.7 LID results

We first evaluate the LID accuracy of the two JSM-based LID dictionaries (single-language and multi-language) and 'all-four' techniques, where we use the words' true language of origin as a

reference to estimate the LID accuracy. As discussed in Section 2.6, we report on the precision, recall and F-measure for the analysis. Table 8.2 shows the results of the two JSM-based and 'all-four' LID predictions using precision, recall and F-measure.

TABLE 8.2: LID Precision, Recall and F-measure using different LID approaches.

Dataset	*Dict*	Precision (%)	Recall (%)	F-measure (%)
SADE	All-four	27.11	100.00	44.66
	Multi	84.87	92.80	88.66
	Single	88.86	81.94	85.26
Multipron	All-four	26.20	100.00	41.52
	Multi	72.63	88.48	79.78
	Single	78.64	75.04	76.80

As expected, from the previous study in Section 7.5.3, multi-LID outperforms single LID across both data sets. Also, we expect very low precision on 'all-four' approach where we assume all word lists originate from all target languages.

8.2.8 G2P results

For G2P analysis and evaluation, we measure the different dictionary approaches with each corpus reference dictionary (as is found in the Multipron and SADE corpus). As mentioned earlier, these experiments use manually transcribed (also capturing pronunciation mistakes made by the speakers) and semi-automatic reference dictionaries, where one is more accommodative than the other. The Multipron reference dictionary contains more pronunciation variants per word as opposed to the SADE reference dictionary.

Per corpus, we report G2P accuracy using V-PA, V-WA, S-PA and S-WA where:

- we compare the four hypothesised dictionaries against the reference dictionary using the two different phone sets, and
- we analyse pronunciations of words for which the LID differs.

In order to understand the effect on pronunciations for scenarios where LID differs, we also consider:

- analysing wrongly and correctly classified words from both JSM-based-G2P-based dictionaries using the 'combined' phone set, and
- analysing confusability among languages for JSM-based single-language LID dictionary using the 'combined' phone set.

Tables 8.3 and 8.4 show V-PA, V-WA, S-PA and S-WA achieved using the 'detailed' and 'combined' phone sets, respectively, and using different dictionary approaches. Across all performance metrics, accuracy obtained using the 'detailed' phone set is lower as opposed to 'combined' because of the higher penalty incurred with the stricter phone mapping strategy employed. We observe that across the two phone sets, accuracy obtained using 'all-four' outperform other G2P-based dictionaries while 'multi' LID dictionary outperformed 'true-word' LID dictionary with 'single' LID dictionary performing the worst (of the four G2P-based dictionaries).

Given the performance measure used, the higher the number of pronunciation variants the better the G2P accuracy. This explains why the 'all-four' approach seems the best dictionary approach, since we over-generate pronunciation variants across all four target languages. These results are not expected to mirror ASR accuracy, as it is known that more variants tend to introduce higher confusability in ASR systems. Columns Ref_{avg} and Hyp_{avg} represent the average number of reference and hypothesised pronunciations per word. For the 'Multipron' corpus, the number of pronunciation variants per word is higher than in the SADE corpus due to the fact that each audio clip produced by a speaker was manually transcribed.

TABLE 8.3: V-PA, V-WA, S-PA and S-WA achieved with 'detailed' phone set for different dictionary approaches on two data sets.

Dataset	*Dict*	$V-PA$	$V-WA$	$S-PA$	$S-WA$	$Ref_{.avg}$	$Hyp_{.avg}$
Multipron	All-four	76.20%	19.93%	94.27%	71.13%	5.52	3.53
	Multi	70.55%	15.49%	90.64%	59.60%	5.52	1.35
	True word	68.44%	13.28%	89.15%	53.41%	5.52	1.05
	Single	66.95%	13.13%	88.03%	52.55%	5.52	1.00
SADE	All-	83.80%	39.74%	84.79%	42.03%	1.12	3.53
	Multi	80.49%	36.86%	81.73%	39.02%	1.12	1.32
	True word	79.89%	35.81%	81.06%	37.62%	1.12	1.08
	Single	76.97%	33.12%	78.53%	35.10%	1.12	1.00

TABLE 8.4: V-PA, V-WA, S-PA and S-WA achieved with 'combined' phone set for different dictionary approaches on two data sets.

Dataset	*Dict*	$V-PA$	$V-WA$	$S-PA$	$S-WA$	Ref_{avg}	Hyp_{avg}
Multipron	All-four	78.94%	25.19%	96.41%	81.73%	5.21	3.4
	Multi	73.77%	20.20%	93.26%	70.98%	5.21	1.33
	True word	72.07%	18.06%	92.12%	66.56%	5.21	1.05
	Single	70.54%	17.51%	90.86%	64.55%	5.21	1.00
SADE	All-four	89.59%	59.96%	90.53%	62.73%	1.12	3.39
	Multi	86.94%	57.26%	88.19%	60.20%	1.12	1.31
	True word	86.79%	56.53%	87.97%	59.12%	1.12	1.08
	Single	83.42%	52.91%	85.08%	56.04%	1.12	1.00

Comparing Tables 8.3 and 8.4, we observe that trends remain similar using either phone set. We use the 'combined' phone set from here onwards.

In order to understand the two JSM-based dictionaries better, we analyse the G2P accuracy of wrongly and correctly LID-tagged words of the dictionaries. Per corpus, we compare the G2P accuracy of 'single' and 'multi' LID dictionaries against the reference dictionary based on whether the LID tag was correct or wrong. Table 8.5 shows the results obtained when we analyse wrongly and correctly classified words using the 'combined' phone set for two of the JSM-based dictionaries.

TABLE 8.5: Multipron and SADE corpus: G2P accuracy when analysing correctly and wrongly LID-tagged words for the two JSM-based dictionaries. V-PA, V-WA, S-PA and S-WA achieved with 'combined' phone set.

Dataset	*Dict*	*Pattern*	$V-PA$	$V-WA$	$S-PA$	$S-WA$	Ref_{avg}	Hyp_{avg}
Multipron	Single	LID correct	73.45%	19.40%	93.24%	71.16%	5.13	1.00
	Single	LID wrong	60.21%	10.10%	82.23%	39.09%	5.61	1.00
	Multi	LID correct	72.82%	18.79%	92.67%	69.09%	5.14	1.02
	Multi	LID wrong	63.51%	11.97%	85.65%	45.51%	5.9	1.15
SADE	Single	LID correct	86.68%	57.78%	88.17%	60.88%	1.11	1.00
	Single	LID wrong	64.48%	25.13%	69.19%	32.72%	1.26	1.00
	Multi	LID correct	86.72%	57.26%	87.99%	59.97%	1.11	1.04
	Multi	LID wrong	69.03%	30.77%	73.68%	39.18%	1.29	1.16

A Comparison of 'single' and 'multi' LID dictionaries for correctly LID-tagged words show that the 'single' LID dictionary obtains the best performance across almost all the metrics for both corpora. With words where language tags are wrongly identified, better accuracy is obtained on multi-LID dictionaries in both corpora. Also, as expected, G2P accuracy is significantly higher when words are correctly LID-tagged than when errors occur. For further understanding on why the 'single' LID dictionary performs better on correctly LID-tagged and proves worse on wrongly LID-tagged, we extract a confusion matrix (as shown in Table 8.6) of the languages employed and predict their G2P accuracies using V-PA, as shown in Table 8.7.

TABLE 8.6: Multipron and SADE corpus: Confusion matrix among languages for single-language LID dictionary. Result shows only the word frequencies.

		Observed languages			
	Ref. languages	Afrikaans	English	Sesotho	isiZulu
Multipron	Afrikaans	105	93	15	5
	English	68	404	21	3
	Sesotho	0	7	250	5
	isiZulu	4	9	38	203
SADE	Afrikaans	368	63	14	0
	English	448	5499	203	47
	Sesotho	3	7	386	11
	isiZulu	1	8	53	331

In Table 8.6, we observe that there is no confusion between Sesotho and Afrikaans in the Multipron corpus, and between Afrikaans and isiZulu in the SADE corpus. One anomaly is the

high misclassification rate between English or Afrikaans (as reference) and Sesotho (as the observation). Tracing some of these errors we found that the Sesotho training data, even after verification by Sesotho language practitioners, was still not as clean as anticipated. We will review this in future work.

TABLE 8.7: Multipron and SADE corpus: G2P accuracies for single LID dictionary that match the confusion matrix in Table 8.6. Result shows V-PA using the 'combined' phone sets. Accuracies estimated from very low sample counts marked in italics.

		Observed languages			
	Ref. languages	Afrikaans(%)	English(%)	Sesotho(%)	isiZulu(%)
Multipron	Afrikaans	**85.29**	57.08	68.43	*71.00*
	English	54.27	**69.44**	50.08	*63.43*
	Sesotho	-	*48.19*	**77.13**	*85.29*
	isiZulu	*56.98*	*37.71*	77.66	**81.59**
SADE	Afrikaans	**93.85**	55.17	*66.55*	-
	English	53.17	**86.42**	54.93	62.10
	Sesotho	*65.00*	*55.56*	**96.32**	*96.27*
	isiZulu	*50.00*	*55.88*	88.65	**95.48**

In Table 8.7, we list the G2P accuracy for the same confusable pairs as in Table 8.6. Note that all confusable pairs did not have sufficient samples to produce informative results – results considered not to be informative are italicized.

For all informative pairs, the best G2P accuracies obtained occur diagonally (bolded in Table 8.7). G2P accuracies for words confusable between the two Germanic languages (Afrikaans and English) are closely related in both corpora. For the Bantu languages, we observe that using either one to predict the pronunciation of the other has less effect on G2P accuracy. A significant drop in performance is observed if any of these languages is used to predict pronunciation for the Germanic words. On the other hand, for the Germanic langauges, a large drop in performance is observed if the Afrikaans G2P converter is used to predict the pronunciation for a word from English, and vice versa.

8.3 ASR analysis

8.3.1 Data

For this experiment, we use the entire SADE corpus [136]. The SADE corpus contains 13 hours 56 minutes and 9 seconds of speech from 40 speakers, with 500 prompts per speaker. The data corpus is balanced across gender with 20 male and female speakers respectively. We partition the set into 35% and 65% of the entire corpus representing testing and training set respectively. No development set was used, and parameters set using the training data directly.

8.3.2 Pronunciation dictionary

In Section 8.2.4, we discussed how the pronunciation dictionaries for the G2P analysis were generated. In this section, we use the same dictionaries for ASR analysis. As pointed out in that same Section 8.2.4, spelled-out words, words with two characters or less and words identified as belonging to languages not studied were removed from the dictionaries. In this experiment, for each dictionary approach (see Section 8.2.4), in order to obtain a complete pronunciation lexicon necessary to train an ASR system and avoid out-of-vocabulary (OOV) tokens, we:

- identify all words not included in the previous analysis, and extract missing words with their corresponding pronunciations (either single or multiple) from the original SADE transcribed dictionary, referred to as 'supplemental pronunciation dictionary', and
- combine each dictionary being analysed (the case-specific dictionaries from Section 8.2.4) to obtain a complete pronunciation dictionary.

Note that the supplemental pronunciation dictionary retains variants (single or multiple) as contained in the original SADE dictionary. All phones were mapped to the 'combined' phone set using the mapping approach discussed in Section 8.2.3

8.3.3 Kaldi-Based training

We employ a standard Kaldi-based system using a recipe similar to the Babel recipes in Kaldi [173] to build the acoustic model. We build a context-dependent crossword HMM-based phone recogniser with triphone models and Gaussian mixture models (GMMs) as the statistical model. Each model has three emitting states. For speaker-specific transforms, we use feature-space maximum likelihood linear regression (FMLLR) and maximum likelihood linear transform (MLLT) per speaker. Cepstral Mean Normalisation (CMN), as well as Cepstral Variance Normalisation (CVN), are used to perform speaker-specific normalisation. The GMM-HMM acoustic model is trained on features that are obtained when seven frames are spliced together of 13-dimensional Mel-frequency cepstral coefficients (MFCCCs) each. The feature dimensionality is later reduced using linear discriminant analysis (LDA) to 40. To partially compensate for the implicit assumption of feature independence, a single semi-tied co-variance transforms applied on the features obtained using LDA. In order to initialise deep neural network (DNN) training, we use alignment obtained from the GMM-HMM model as the input, where the network has 3 hidden layers.

8.3.4 Evaluation

The test vocabulary and language model (LM) used has a significant effect on recognition accuracy. In order to minimise the effect of the LM, we utilise a flat LM knowing that recognition

accuracy will be poor. We therefore also use a trained n-gram-based LM to verify that our system is sufficiently accurate (that is, to evaluate the overall system development process). Finally, we create a variant-tagged system in order to be able to reconcile homophones, as explained in more detail below.

Recognition accuracy was evaluated in terms of the word error rate (WER) metric which aligns a recognized word string against the correct word string and computes the number of substitutions (S), deletions (D), Insertions (I) and the number of words in the correct sentence (N). WER can be computed as the sum of deletions, substitutions and insertions divided by the total number of words in the reference set.

Word recognition was performed on three sets of experiments, which include: (1) using a flat language model (all words are considered equally likely at all times) with no lexicon tags, (2) using a flat language model (LM) with lexicon tags, (3) using a trained LM with no lexicon tag. For the trained LM, we use 4-gram modified Kneser-Ney technique where the minimum n-gram order is set to 1. To determine the optimal LM weight, we perform a 2-fold cross-validation on the testing data. To understand how each of the dictionaries influences the performance of each system, we measure word error rate.

One of the most common reasons for ASR errors observed here was acoustic confusability due to homophones. Examples include spelled-out words, numerical words, dates, etc. While this ambiguity is typically resolved by the language model, a system developed with a flat language model may be unfairly penalised. To report the performance of the system, we therefore consider two cases where:

- All homophones are retained without performing any preprocessing task. See Table 8.8, example 1.
- Each word in the hypothesised string is remapped to its reference counterpart if the pronunciation of the reference word and observed word matches. See Table 8.8, example 2. For words with multiple pronunciations, we label the original lexicon with word tags to simplify this task. We refer to this dictionary as 'lexicon with tags'. These tags are used by the Kaldi system during the decoding process, which then identifies the pronunciation variant that was used.

Remapping homophones, in Table 8.9, we reduced the WER by only approximately 1% on average across the target dictionaries. Trends observed before or after homophones preprocessing (realigning homophones) are the same. Homophones therefore affected the results less than we initially anticipated.

As expected, we achieve the best result on the reference dictionary. ASR performance decreases as soon as predicted dictionaries are used. The order, from best to worst performing technique

TABLE 8.8: Examples of homophone remapping in a sentence. Ex. 1 represents the original decoded string, while Ex. 2 represents the preprocessed sentence after homophone remapping.

Ex. 1	
REF	l-3 communications hldgs.
HYP	l-3 communications holdings
Ex. 2	
REF	l-3 communications hldgs.
HYP	l-3 communications hldgs.

now being: manual, true-word, single, multi and all-four. While the differences between 'true-word', 'single' and 'multi' are small but consistent, there is a much larger performance difference between the reference dictionary and the above three, as well as between these three and the 'all-four' dictionary.

TABLE 8.9: WER of the variant-tagged system for different dictionary approaches prior and post reconciling homophones.

Dict.	WER(prior to preprocessing)	WER (after preprocessing)
Manual	57.8	56.4
True-word	64.1	62.2
Single	65.0	63.1
Multi	65.8	64.1
All-four	68.9	66.9

TABLE 8.10: Average number of pronunciation variants, as well as the WER of the variant-tagged system using flat, and WER of a the system without variant-tagged using flat and trained LMs.

		WER		
Dict.	Average variants	Flat LM (no tags)	Trained LM (no tags)	Flat LM (with tags)
Manual	1.12	57.30	14.60	57.80
True-word	1.08	62.70	16.40	64.10
Single	1.00	63.90	18.20	65.00
Multi	1.31	64.90	19.90	65.80
All-four	3.39	67.80	20.20	68.90

In Table 8.10, we observe that the same trends are observed across dictionary approaches (manual, true-word, single, multi, and all-four) regardless of the language model used (either flat or trained) with a flat LM with lexicon tags proving the worst. Since the lattices that are extracted during decoding do differ based on whether two phone string are variants of a single word, or competing words, the loss of accuracy from 'no tags' to 'with tags' is understandable.

We also observe that more variants tend to result in poorer ASR performance. While this observation does not hold for the 'true-word' dictionary, we presume that the good performance could be associated with the fact that the dictionary benefits more from prior information relating to the true source languages of words before G2P conversion. Moreover, the drop in performance of the recognition between 'true-word' and 'manual' dictionaries correspond to errors made by

the G2P converter, while the difference in WER between the two LID-based dictionaries and 'true-word' can be associated with the LID prediction error.

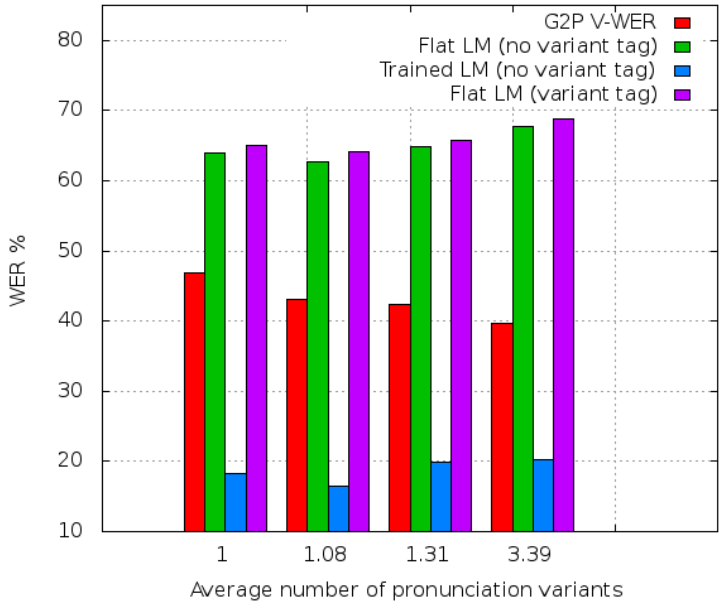

FIGURE 8.2: G2P and ASR accuracy obtained on different pronunciation variants.

In Figure 8.2, we plot ASR WER with the variant-based G2P phone error rate (V-PER), where V-PER equates to 1 - V-PA. We show that, using this metric, G2P accuracy cannot be used to predict ASR performance, with the 'all-four' dictionary obtaining best G2P accuracy while performing worst when analysing ASR performance. For the best ASR results, if the true source language of words are unknown, the 'single-LID' dictionary is the best option to use, if obtaining manual transcriptions of an entire vocabulary is unrealistic.

8.4 Unilateral versus Bilateral G2P analysis

Up to now, calculating G2P accuracy of variant-based dictionaries used a 'unilateral' approach where pronunciation variants in the hypothesised lexicon are evaluated against all the variants in the reference dictionary to estimate the best matching variants. This approach does not penalise techniques that over-generate pronunciation variants. In order to obtain a better indication of the balance between accuracy and number of variants, we define a new bilateral metric, that takes all the variants in both the reference and hypothesised lexicons into account.

Specifically, the new approach first pairs up all variants in the reference dictionary and all variants in the hypothesised dictionary, one-by-one. Per word, if there are more reference variants than hypothesised variants, each of the hypothesised variants will first be mapped to its best matching reference variant, and then hypothesised variants will be 're-used' to form pairs with all matching reference variants; and vice versa.

See Table 8.11 for a demonstration of the difference between unilateral and bilateral scoring.

TABLE 8.11: Example: comparing unilateral and bilateral V-PA for hypothetical words 'one' and 'two'.

Word	Ref prons	Hyp prons	Uni pairs	Uni V-PA	Multi pairs	Multi V-PA
one	w a n	w O n w a n O n e	w a n → w a n	100%	w a n → w a n w a n → w O n w a n → O n e	56%
two	t u: t u	t @	t u: → t @ t u → t @	50%	t u: → t @ t u → t @	50%

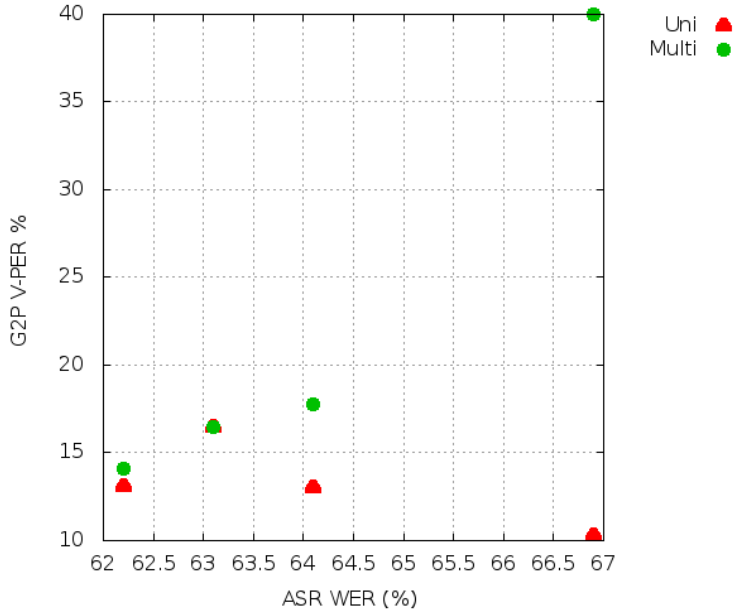

FIGURE 8.3: Comparison between unilateral and bilateral V-PA against ASR WER of the flat LM without variant-tagged lexicon.

While more complex to evaluate (the new metrics requires an additional alignment per variant pair) the bilateral approach is conceptually a more valid indication of the role that variants play. When we now re-evaluate phone accuracy (see Table 8.12) the G2P results follow the ASR trends, as initially shown in Section 8.3.4, much more closely. In fact, the order of performance – true-word, single, multi and all-four – is a clear match, with the actual results also well correlated, as shown in Fig. 8.3. Note that the green dots (marking the bilateral G2P V-PER) now correlate very closely with ASR WER.

In Table 8.12, we observe that 'all-four' lexicon is affected most by the change in metric: producing a high accuracy with unilateral scoring, this is much lower using multlateral scoring. Similarly, the G2P accuracy of the 'multi' and 'true-word' dictionaries is also decreased. As expected, performance of the 'single' dictionary remains the same during unilateral or bilateral scoring stays the same across the two employed concepts (uni- and multi-lateral).

TABLE 8.12: Comparison of uni- and multi-lateral concepts using 'combined' phone set for different dictionary approaches on two data sets.

Dataset	*Dict*	$V-PA$	$V-WA$	$S-PA$	$S-WA$	Ref_{avg}	Hyp_{avg}
Multi	All-four	60.03	20.49	90.73	63.04	5.21	3.40
	Multi	82.20	51.44	88.31	60.50	5.21	1.33
	Single	83.52	53.17	85.11	56.30	5.21	1.00
	True word	85.89	55.37	88.06	59.40	5.21	1.05
Uni	All-four	89.59	59.96	90.53	62.73	1.12	3.39
	Multi	86.94	57.26	88.19	60.20	1.12	1.31
	Single	83.42	52.91	85.08	56.04	1.12	1.00
	True word	86.79	56.53	87.97	59.12	1.12	1.08

8.5 Aligned phone accuracy

In Section 8.2.6, we discussed two approaches (standard and aligned) used to estimate phone accuracy. In this section, we analyse the accuracy of aligned phone accuracy approach with results obtained in Section 8.4.

TABLE 8.13: G2P accuracy obtained using aligned phoneme accuracy, as well as the comparison between uni- and multi-lateral concepts using 'combined' phone set for different dictionary approaches on two data sets.

Dataset	*Dict*	$V-PA$	$V-WA$	$S-PA$	$S-WA$	Ref_{avg}	Hyp_{avg}
Multi	All-four	65.33	20.49	91.39	63.04	5.21	3.40
	Multi	83.99	51.44	89.29	60.50	5.21	1.33
	Single	85.11	53.17	86.53	56.30	5.21	1.00
	True word	87.14	55.37	89.05	59.40	5.21	1.05
Uni	All-four	90.56	60.27	91.37	63.04	1.12	3.39
	Multi	88.20	57.57	89.28	60.50	1.12	1.31
	Single	85.11	53.17	86.53	56.30	1.12	1.00
	True word	88.02	56.82	89.05	59.40	1.12	1.08

Table 8.13 shows the G2P accuracy obtained using aligned phoneme accuracy over uni- and multi-lateral concepts. Results show an approximate 1% difference but a similar trend as discussed in Section 8.4 above.

8.6 Conclusion

This chapter focused on the implications of producing language-based pronunciation variants, where different LID techniques are used to predict the most probable source language(s) of a word. Both G2P and ASR performance were analysed and compared.

To understand the implications of creating LID-based dictionaries, we considered four dictionaries that were generated using a combination of LID and G2P prediction. The same language-specific G2P predictors were used for all dictionaries, but different LID options were evaluated: (1) when the true source language is known ('true word'), (2) when a single source language is predicted ('single'), (3) when multiple source language are predicted ('multi'), and (4) when it is simply assumed all words may be from all relevant source languages ('all four'). These dictionaries were evaluated against a reference dictionary (developed with each of the corpora and manually corrected/verified) to measure G2P accuracy. ASR performance was evaluated by developing a full-blown ASR system and evaluating performance with both a flat and trained LM.

During G2P analysis, it became clear that the way in which variants are dealt with during accuracy calculation, has a large effect on measured performance. Using existing G2P accuracy measures, variants in the hypothesised dictionary are not penalised sufficiently, and the 'all four' dictionary produced the best results. Similarly, 'multi' performed better than 'single'. As expected, 'true word' performed the best, also providing an indication of the importance of correct LID. The latter was also observed when evaluating G2P accuracy for words that were either correctly or incorrectly classified, with correctly classified words obtaining significantly higher G2P accuracies.

We defined a new G2P performance metric – bilateral V-PA – which deals with variants in way that is conceptually more sound: all variants in the reference and hypothesis lexicons are first compared to create matching pairs (with either some of the reference or some of the hypothesized variants repeated per word, where variants of unequal number are to be matched). This technique automatically penalises a dictionary for over- or under-generating variants.

Using this new metric, if we order the dictionaries according to G2P performance (best to worse) we obtain an ordering of: 'true word', 'single', 'multi' and 'all four'. This ordering then matches the actual ASR performance observed, with best ASR results observed with the reference dictionary. Similarly, both G2P performance and ASR performance for 'true word', 'single' and 'multi' are fairly close, with 'all four' performing much more poorly.

Finally, we conclude that:

- Accurate LID is important for system performance when manually developed dictionaries are not available.
- The proposed LID techniques can be used to create dictionaries that produce ASR results that are comparable with that of a dictionary developed based on known source languages, even though none of these auto-generated dictionaries achieve the accuracies obtained with a manually developed dictionary.

- When creating an initial dictionary to be evaluated manually, the 'multi' LID dictionary approach is recommended.
- When building an ASR system without any manual verification, the 'single' LID dictionary is considered a better choice.
- When comparing G2P accuracies of variant-based dictionaries, the newly defined bilateral V-PA provides a better indication of expected ASR performance than standard V-PA.

In future work, it will be interesting to understand if multiple pronunciations can become beneficial if an ASR-directed threshold is selected or can bilateral V-PA be tweaked for further performance. In the next chapter, we review specific issues addressed in this thesis and other findings.

Chapter 9

Conclusion

9.1 Introduction

The theme of this work is centralised on language identification of proper names, within the context of better pronunciation modelling for proper name recognition. The focus is specifically on South African languages, which fits in the broader domain of under-resourced languages. Language identification of proper names is a difficult task; this is true of well-studied languages and even more so of under-resourced languages.

Specific issues addressed in this thesis are detailed in this chapter. In Section 9.2, an overview is given of the main outcomes of this study. In Section 9.3, we outline the most significant contributions of this thesis. Section 9.4 discusses future work.

9.2 Summary of thesis

Firstly, we were faced with the problem that there was no relevant corpus for South African languages, which could be used to better understand the factors that influence proper name pronunciation, as discussed in Section 1.1. To deal with this problem, two multilingual corpora that are specifically aimed at the diverse proper names that occur in South Africa were developed. The process to create and analyse these corpora was published in [170] and [136].

The two resources marked the first of their kind in South Africa. Initially, a 'Southern African corpus for multilingual name pronunciation' was developed (disussed in Chapter 3). In this corpus, we paired first and last names from four target languages in order to understand the variations in the pronunciation of proper names. As part of a contribution during the corpus development, a way was described to annotate a name-list using crowd-sourcing. Names were

carefully curated, and pronunciations manually annotated based on the actual audio produced by each speaker.

Further down the line, a 'South African directory enquiry' corpus was developed (discussed in Chapter 6). This corpus was designed specifically for the domain enquiry system in South Africa, where constituted names are not only people's names, but also include the names of songs, restaurants and places, for example, phrases such as '*The Hillside Tavern*'. The resulting set of words is, therefore, a mixture of names and some generic words. Specifically for under-resourced languages, ways were described to annotate name-lists using different LID techniques. While this is a larger audio corpus (more applicable to ASR analysis), its dictionary only contains canonical pronunciations, developed during a process that combined G2P prediction and manual verification of only a subset of words.

In Chapter 4, the task of LID of individual words (generic words in isolation) was examined. Two well-known classification methods were selected and applied to LID of isolated words. Some of these results were published in [35]. A comparison was made of different classification techniques that have been reported to have good performance on short text segments by applying them to individual words and by analysing the effect of smoothing. Character n-gram models were adopted; such techniques have demonstrated good performance over a variety of applications. The relationship between word length, n-gram length and performance accuracy (as a few factors that influence LID accuracy) of each classification technique employed also investigated. Prior to comparing different classification techniques, we investigated the difference between using unique words against all existing words (allowing word repetitions) when training a naïve Bayes classifier and found a computational win when using unique words, which was not offset by any loss in LID accuracy. To improve the baseline performance by reducing the total number of unseen tokens on higher-order n-gram models, we experimented with Katz smoothing, absolute discounting and Witten-Bell smoothing. Smoothing improved the average classification accuracy for higher order n-grams by a percentage of approximately 8%. The highest classification accuracy of 88.12% was obtained by using an SVM with an RBF kernel and an n-gram length of $n = 3$. (These results were not yet asymptotic, better identification can be achieved with more training data.)

In Chapter 5, a novel approach to T-LID of words in isolation was introduced. A subset of this work, describing how to apply JSMs to the LID of a generic word in isolation was published in [104]. The applicability of JSMs to the T-LID task was investigated. The best classifier analysed earlier (SVM) was compared with the novel technique for T-LID (JSMs) and the comparative performance that can be obtained when applying JSMs, rather than the SVM classifiers was analysed.

Prior to doing any comparative analysis, a method for language verification, applicable to text-based preprocessing tasks was proposed. This technique was used to clean up the data and investigate the significance of different proxy languages based on their web information. In

conclusion, using a few number of proxy languages with significant web presence produce a good performance when doing language verification of under-resourced languages. However, a higher number of proxy languages with limited web presence result in better performance for language verification of resourced languages.

Using JSMs, it was shown how the T-LID task can be recast as a 'pronunciation learning' task, with limited modification of the training process. During the training phase, different parameter choices that influence performance accuracy were discussed. In order to select the final language of origin of words, we proposed three voting schemes, namely force-pronunciation voting, log-probability voting and majority voting. For the specific task studied (a four-language classification task), it was found that the JSM-based technique reduced the SVM error by approximately 40% (relative) across a range of training data sizes evaluated. For the largest training set (48K words, 12K per language) an accuracy of 97.2% was obtained, which compared well with the best SVM classifier evaluated (at 95.2%).

Up to now, the focus had been on T-LID of generic words in isolation. In Chapter 6, experiments were undertaken with JSMs for T-LID of proper names. Initially, it was investigated how JSMs generalise given limited training sample sets to predict the language of origin of names and the technique derived in this study was applied to language tagging of a newly developed corpus (SADE). We observed that the JSMs model trained on an in-domain dataset outperformed an out-domain model trained on a far larger dataset. In addition, a relative error reduction of approximately 3% over the out-domain model was obtained, despite the fact that the out-domain model is based on a dataset that is 44 times bigger than the in-domain set.

For language tagging of the SADE word list, the efficiency of the overall LID tagging strategies was evaluated by comparing the performances of the different LID tagging techniques employed. The results were evaluated using two different scenarios; firstly, excluding web-based techniques when tested on 11 South African languages, and secondly, testing on only three South African languages while accommodating other web-based techniques. Analysis showed that for LID of complex names (notably loan words), web-based techniques outperformed the non-web-based techniques developed in this study, presumably because the web-based techniques have encountered many of those terms in the appropriate language context to predict the correct result given the vast amount of training data available. For LID of common words, non-web-based techniques perform equally well and could rival the web-based methods.

One interesting area that has to date received less research focus, is the LID of multilingual words. In Chapter 7, it was investigated how to use JSMs for multilingual classification of proper names. It was described how to use the posterior probability of a translation as a threshold value for LID of multilingual words using JSMs. In order to deal with this task, we proposed an improvement to the monolingual classification voting techniques developed earlier by forcing JSMs to produce output strings that are associated with a single language, in order

to obtain more trustworthy posteriors to analyse. This approach was compared to the best available voting technique ('logprob sum'), and an improvement in F-measure from 79.39% to 81.02% was observed, training and testing on the same set of proper names. To classify proper names as multilingual, an experiment was carried out with two baseline methods ('top-1' and 'top-2') where both approaches produced a trade-off between recall and precision. To strike a balance between the two metric values, we proposed two new techniques ('relative likelihood' and 'absolute posterior'). While the difference between the two new methods is statistically insignificant, both outperform the baseline with 'absolute posterior' using the NCHLT models producing an F-measure of 81.79%.

In Chapter 8, the implications of accurate language identification on name pronunciation are evaluated and the first results for previously unstudied South African languages are produced. In order to understand the implications, we considered four dictionaries that were generated using a combination of LID and G2P prediction. Using LID to predict source languages of name, we evaluated different LID options: (1) when the true source language is known ('true word'), (2) when a single source language is predicted ('single'), (3) when multiple source language are predicted ('multi'), and (4) when it is simply assumed all words may be from all relevant source languages ('all four'). In order to predict pronunciations, language-specific G2P predictors were used. These dictionaries were evaluated against a reference dictionary (developed with each of the corpora and manually corrected/verified) to measure G2P accuracy.

Experimenting using existing G2P performance metrics, we observed that the 'all-four' dictionary outperforms other G2P-based dictionaries while the 'multi' LID dictionary outperformed the 'true-word' LID dictionary, with the 'single' LID dictionary performing the worst (of the 4 G2P-based dictionaries). Also, we observe that the higher the number of pronunciation variants the better the G2P accuracy. This explains why the 'all-four' dictionary approach produced the best result since we over-generate pronunciation variants.

With regard to ASR using the four dictionaries, we perform word recognition on three sets of systems (flat and trained n-gram-based LM with no variant-tagged dictionary, and flat LM with variant-tagged dictionary). We use a lexicon with variant-tags to handle acoustic confusability due to homophones in an ASR system. Results show that the manual system (developed using a manually created pronunciation lexicon) produced the best results over other dictionary approaches.

During G2P analysis, it became clear that the way in which variants are dealt with during accuracy calculation, has a large effect on measured performance. Based on the existing G2P accuracy measures where variants in the hypothesised dictionary are not penalised sufficiently, we defined a new G2P performance metric – bilateral V-PA – which deals with variants in way that is conceptually more sound: all variants in the reference and hypothesis lexicons are first compared to create matching pairs (with either some of the reference or some of the hypothesized

variants repeated per word, where variants of unequal number are to be matched). This technique automatically penalises a dictionary for over- or under-generating variants.

Using this new metric, if we order the dictionaries according to G2P performance (best to worst) we obtain an ordering of: 'true word', 'single', 'multi' and 'all four'. This ordering then matches the actual ASR performance observed, with best ASR results observed with the reference dictionary. Similarly, both G2P performance and ASR performance for 'true word', 'single' and 'multi' are fairly close, with 'all four' performing much more poorly.

9.3 Contribution

In this section, the most significant contributions of this thesis are outlined:

- A novel approach to T-LID for both generic words and proper names are developed. The proposed method is well suited to classifying words in isolation, and generalises well with a limited amount of data compared to other conventional classifiers evaluated. This is important because of the limited available data for under-resourced languages. Given trade-off between accuracy and computational cost associated with most conventional classifiers, this method balances the two factors well. The new algorithm re-contextualises a pronunciation modelling algorithm (JSMs) to the T-LID task, and provides good classification accuracy. The performance of the algorithm was analysed and factors that influence its performance, identified.

- The study contributed to the development and curation of two specialist corpora for South African languages, useful for both pronunciation modelling tasks and automatic recognition system analysis. Also, the two newly developed corpora will be useful for additional analysis in other environments where several languages are spoken in the same community, or in closely interacting communities and also provide empirical language data for further research work. During corpus development, different techniques were experimented with. Amongst others, we proposed a crowd-sourcing technique as a way to annotate a name-list for the under-resourced dataset and use different automatic LID techniques for language tagging of a new corpus.

- A new task was specified and analysed: the multilingual LID of isolated words has received little interest to date. As far as we are aware, this specific task has not yet been studied. It was observed that the LID of multilingual proper names is a challenging task. We proposed using threshold values based on variable posterior probability estimates among LID tags to classify multilingual proper names and report on results.

- A method was proposed to correctly measure G2P accuracy of variant-based dictionaries. This technique is beneficial, providing there is no straightforward approach to analyse G2P accuracy for dictionaries with pronunciation variants. This method compares the pronunciation variants from the reference and hypothesised lexicons and automatically penalises a dictionary for over- or under-generating variants. By being able to penalise wrongly predicted pronunciation variants, this provides a way to mirror ASR performance more closely, and better judge an outcome prior to training and testing on the vocabulary.
- A comparison was made of different conventional T-LID techniques for short text segments, along with the factors that affect T-LID of words in isolation, such as text word-length and training set sizes.
- The implications of accurate LID for pronunciation modelling of proper names – as typically used in an ASR system – were carefully investigated. Results were obtained on four South African languages (not previously studied in this context) and guidelines suggested for using LID during pronunciation dictionary development.

9.4 Future work

The work discussed in this thesis has a wide range of potential extensions to improve on performance accuracy. Moreover, an adaptation of the current work to additional real-world scenarios can be performed.

- These techniques can be applied to other languages. This technique would also allow analysis of implications when using larger data sets. While the focus of this thesis was specifically on understanding under-resourced conditions, it would be interesting to consider the implications in better-resourced domains. An important proposition will be to train the model on more languages. Work such as [94] attempt to expand the language set on which a model is trained on to thousands. [62] reported that such a system suffers low accuracy level due to high misclassification rate of many languages not present in the testing set. A successful system could help identify closely-related languages based on language dialect variety.
- Multilingual LID is a more recent task and further optimisation will be investigated. As discussed earlier in Section 2.4.2, the closest related task addresses LID for multilingual documents (where a single document can belong to more than one language class). Further investigation is required to ascertain if any of the approaches applied to LID of multilingual LID of documents can be used for multilingual LID of isolated words.

- Bilateral variant-based G2P accuracy measures were defined in this work, but not analysed in further detail and on additional corpora. It would be interesting to determine whether the high correlation achieved between G2P and ASR accuracy generalise to other domains.
- Different T-LID techniques are analysed in this work, it would be interesting to see how the combination of these systems yields a better T-LID performance which subsequently helps in better pronunciation lexicon development. An initial application to South African languages will be prioritised.
- In this work specifically, JSM model based LID technique is used to assist pronunciation lexicon development. The decision was based on the fact that JSM model based LID technique outperforms other traditional T-LID techniques, where SVM approach performance is not too far. Further analysis is required to understand how much T-LID performance difference has real influence on the final speech recognition system development - will small T-LID accuracy difference like 1% or 2% make a significant impact on the ASR system.
- The work discussed in this thesis took a traditional approach where T-LID is carried out first and then pronunciation is inferred. A further research work would be taken to find out if possible to take a more integrated approach where LID and pronunciation inference are jointly solved.

9.5 Concluding remarks

In this thesis, we explored language identification for proper name recognition, specifically for under-resourced languages in South Africa. It was shown that language identification of proper names is a challenging task – more so than language identification of generic words – and that the quality of language identification has a direct influence on the quality of pronunciation prediction, and subsequently, recognition performance.

During the course of this work, various techniques, tools and resources were developed to improve and better understand the automated language identification of proper names. It is hoped that these will provide more insight into the task and open additional avenues for further study.

Bibliography

[1] D. Gibbon, R. Moore, and R. Winski, *Handbook of standards and resources for spoken language systems*, vol. 3. Germany: Mouton de Gruyter, 1997.

[2] E. Shohamy and D. Gorter, *Linguistic landscape: Expanding the scenery.* New York and London: Routledge, 2008.

[3] W. Van Langendonck, *Theory and typology of proper names.* Berlin: Mouton de Gruyter, 2007.

[4] M. Spiegel, M. Macchi, and K. Gollhardt, "Synthesis of names by a demisyllable-based speech synthesizer (SPOKESMAN)," in *Proc. EUROSPEECH-1989*, pp. 1117–1120, 1989.

[5] M. Kgampe and M. Davel, "Consistency of cross-lingual pronunciation of South African personal names," in *Proc. of the 21st Annual Symposium of the Pattern Recognition Association of South Africa (PRASA 2010)*, (Stellenbosch, South Africa), pp. 123–127, Nov. 2010.

[6] M. Benzeghiba, R. De Mori, O. Deroo, S. Dupont, T. Erbes, D. Jouvet, L. Fissore, P. Laface, A. Mertins, and C. Ris, "Automatic speech recognition and speech variability: A review," *Speech Communication*, vol. 49, no. 10–11, pp. 763–786, 2007.

[7] F. Stouten and J.-P. Martens, "Dealing with cross-lingual aspects in spoken name recognition," in *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding ASRU*, pp. 419–424, 2007.

[8] R. Eklund and A. Lindström, "Xenophones: An investigation of phone set expansion in Swedish and implications for speech recognition and speech synthesis," *Speech Communication*, vol. 35, no. 1, pp. 81–102, 2001.

[9] K. Church, "Stress assignment in letter to sound rules for speech synthesis," in *Proc. of the 23rd Annual Meeting of the Association for Computational Linguistics ACL*, pp. 246–253, 1985.

[10] T. Vitale, "An algorithm for high accuracy name pronunciation by parametric speech synthesizer," *Computational Linguistics*, vol. 17, no. 3, pp. 257–276, 1991.

[11] A. Font Llitjos and A. Black, "Knowledge of language origin improves pronunciation accuracy of proper names.," in *Proc. of the 2nd Annual Conference of the International Speech Communication Association (INTERSPEECH)*, (Aalborg, Denmark), pp. 1919–1922, 2001.

[12] T. Modipa and M. Davel, "Pronunciation modelling of foreign words for Sepedi ASR," in *Proc. of the 21st Annual Symposium of the Pattern Recognition Association of South Africa (PRASA)*, pp. 185–189, 2010.

[13] A. F. Llitjós, "Improving pronunciation accuracy of proper names with language origin classes," in *Proc. of the Summer School in Logic, Languge, and Information (ESSLLI) Student Session)*, p. 53, 2001.

[14] M. Adda-Decker and L. Lamel, "Multilingual dictionaries," in *Multilingual Speech Processing* (T. Schultz and K. Kirchhoff, eds.), pp. 123–166, Berlington, MA, USA: Academic Press, 2006.

[15] M. F. Spiegel, "Pronouncing surnames automatically," in *Proc. of the American Voice I/O Society (AVIOS) Conference*, pp. 109–132, 1985.

[16] M. F. Spiegel, "Proper name pronunciations for speech technology applications," *International Journal of Speech Technology*, vol. 6, no. 4, pp. 419–427, 2003.

[17] T. I. Modipa, M. H. Davel, and F. de Wet, "Implications of Sepedi/English code switching for ASR systems," in *Proc. Annual Symposium of the Pattern Recognition Association of South Africa (PRASA)*, (Johannesburg, South Africa), pp. 64–69, 2013.

[18] B. Réveil, J.-P. Martens, and H. van den Heuvel, "Improving proper name recognition by adding automatically learned pronunciation variants to the lexicon," in *Proc. Language Resources and Evaluation Conference (LREC)*, (Valletta, Malta), pp. 2149–2154, 2010.

[19] B. Réveil, J.-P. Martens, and B. Dhoore, "How speaker tongue and name source language affect the automatic recognition of spoken names," in *Proc. of the 10th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, (Brighton, UK), pp. 2995—-2998, 2009.

[20] H. Li, K. C. Sim, J.-S. Kuo, and M. Dong, "Semantic transliteration of personal names," in *Proc. Association for Computational Linguistics ACL*, pp. 120–127, 2007.

[21] W. Basson and M. H. Davel, "Category-based phoneme-to-grapheme transliteration," in *Proc. the 14th Annual Conference of the International Speech Communication Association (INTERSPEECH 2013)*, pp. 1956–1960, 2013.

[22] J. Hakkinen and J. Tian, "N-gram and decision tree based language identification for written words," in *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding ASRU'01*, pp. 335–338, 2001.

[23] Y. Chen, J. You, M. Chu, Y. Zhao, and J. Wang, "Identifying language origin of person names with n-grams of different units," in *Proc. International Conference on Acoustics, Speech and Signal Processing ICASSP*, pp. 729–732, 2006.

[24] A. Binas, "Markovian Time Series Models for Language Identification: Project Report," tech. rep., 2005.

[25] C. Souter, G. Churcher, J. Hayes, J. Hughes, and S. Johnson, "Natural language identification using corpus-based models," *Hermes Journal of Linguistics*, vol. 13, pp. 183–204, 1994.

[26] L.-F. Zhai, M.-h. Siu, X. Yang, and H. Gish, "Discriminatively trained language models using support vector machines for language identification," in *Proc. Speaker and Language Recognition Workshop*, pp. 1–6, 2006.

[27] C. Kruengkrai, P. Srichaivattana, V. Sornlertlamvanich, and H. Isahara, "Language identification based on string kernels," in *Proc. of the 5th International Symposium on Communications and Information Technologies ISCIT*, pp. 926–929, 2005.

[28] M. Padró and L. Padró, "Comparing methods for language identification," *Procesamiento del Lenguaje Natural*, vol. 33, pp. 155–162, September 2004.

[29] S. MacNamara, P. Cunningham, and J. Byrne, "Neural networks for language identification: A comparative study," *Information Processing and Management*, vol. 34, pp. 395–403, 1998.

[30] E. Giguet and E. D. L. Paix, "Categorization according to Language: A step toward combining linguistic knowledge and statistic learning," in *Proc. of the 4th International Workshop of Parsing Technologies (IWPT-1995)*, (Prague - Karlovy Vary, Czech Republic), 1995.

[31] D. Elworthy, "Language identification with confidence limits," in *Proc. Sixth Annual Workshop on Very Large Corpora (at ACL-98)*, pp. 94–101, 1998.

[32] T. Vatanen, J. J. Väyrynen, and S. Virpioja, "Language identification of short text segments with N-gram models.," in *Proc. of the 7th International Conference on Language Resources and Evaluation (LREC 2010)*, (Valetta, Malta), pp. 3423–3430, 2010.

[33] J. Tian and J. Suontausta, "Scalable neural network based language identification from written text," in *Proc. of the International Conference on Acoustics, Speech and Signal Processing ICASSP*, pp. 45–48, 2003.

[34] A. Xafopoulos, C. Kotropoulos, G. Almpanidis, and I. Pitas, "Language identification in web documents using discrete HMMs," *Pattern Recognition*, vol. 37, no. 3, pp. 583–594, 2004.

[35] O. Giwa and M. H. Davel, "N-gram based language identification of individual words," in *Proc. Annual Symposium of the Pattern Recognition Association of South Africa (PRASA)*, (Johannesburg, South Africa), pp. 15–21, 2013.

[36] G. R. Botha and E. Barnard, "Factors that affect the accuracy of text-based language identification," *Computer Speech and Language*, vol. 26, no. 5, pp. 307–320, 2012.

[37] M. Damashek *et al.*, "Gauging similarity with n-grams: Language-independent categorization of text," *Science*, vol. 267, no. 5199, pp. 843–848, 1995.

[38] P. Sibun and J. C. Reynar, "Language identification: Examining the issues," in *Proc. of the 5th Annual Symposium on Document Analysis and Information Retrieval SDAIR*, (Las Vegas, USA), pp. 125–135, 1996.

[39] A. Bhargava and G. Kondrak, "Language identification of names with SVMs," in *Proc. North American Chapter of the Association of Computational Linguistics: Human Language Technologies (NAACL-HLT)*, (Los Angeles, CA), pp. 693–696, 2010.

[40] A. Hategan, B. Barliga, and I. Tabus, "Language identification of individual words in a multilingual automatic speech recognition system," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, (Taipei, Taiwan), pp. 4357–4360, 2009.

[41] F. Yu, X. Feiyu, and U. Hans, "Determining the origin and structure of person names," in *Proc. of the 7th International Conference on Language Resources and Evaluation (LREC'10)*, (Valletta, Malta), pp. 3417–3422, 2010.

[42] Q. Yang, J.-P. Martens, N. Konings, and H. v. d. Heuvel, "Development of a phoneme-to-phoneme (p2p) converter to improve the grapheme-to-phoneme (g2p) conversion of names," in *Proc. of the 3rd International Conference on Language Resources and Evaluation LREC06*, (Genova, Italy), pp. 287–292, 2006.

[43] H. van den Heuvel, J.-P. Martens, and N. Konings, "G2P conversion of names: What can we do (better)?," in *Proc. of the 8th Annual Conference of the International Speech Communication Association (INTERSPEECH 2007)*, pp. 1181–1184, 2007.

[44] H. van den Heuvel, B. Réveil, and J.-P. Martens, "Pronunciation-based ASR for names," in *Proc. the 10th Annual Conference of the International Speech Communication Association (INTERSPEECH 2009)*, pp. 2991–2994, 2009.

[45] M. J. Dedina and H. C. Nusbaum, "PRONOUNCE: A program for pronunciation by analogy," *Computer Speech and Language*, vol. 5, no. 1, pp. 55–64, 1991.

[46] M. Davel and E. Barnard, "Pronunciation prediction with default & refine," *Computer Speech and Language*, vol. 22, no. 4, pp. 374–393, 2008.

[47] M. Bisani and H. Ney, "Joint-sequence models for grapheme-to-phoneme conversion," *Speech Communication*, vol. 50, no. 5, pp. 434–451, 2008.

[48] W. Daelemans, A. Van Den Bosch, and J. Zavrel, "Forgetting exceptions is harmful in language learning," *Machine Learning*, vol. 34, no. 1-3, pp. 11–41, 1999.

[49] O. Andersen, R. Kuhn, A. Lazaridès, P. Dalsgaard, J. Haas, and E. Noth, "Comparison of two tree-structured approaches for grapheme-to-phoneme conversion," in *Proc. of the 4th International Conference on Spoken Language Processing (ICSLP 96)*, (Philadelphia, USA), pp. 1700–1703, 1996.

[50] P. Taylor, "Hidden Markov models for grapheme to phoneme conversion.," in *Proc. of the 9th European Conference on Speech Communication and Technology - INTERSPEECH*, pp. 1973–1976, 2005.

[51] T. J. Sejnowski and C. R. Rosenberg, "Parallel networks that learn to pronounce English text," *Complex Systems*, vol. 1, no. 1, pp. 145–168, 1987.

[52] P. Smitha, L. Shaji, and M. G. Mini, "A review of medical image classification techniques," in *Proc. International Conference on VLSI, Communication and Instrumentation ICVCI*, pp. 34–38, 2011.

[53] C.-L. Liu and H. Fujisawa, "Classification and learning methods for character recognition: Advances and remaining problems," in *Machine Learning in Document Analysis and Recognition* (S. Marinai and H. Fujisawa, eds.), pp. 139–161, Springer, 2008.

[54] B. Medlock, "Investigating classification for natural language processing tasks," Tech. Rep. Technical Report No. UCAM_CLTR721, University of Cambridge, 2008.

[55] C. D. Manning, P. Raghavan, H. Schütze, *et al.*, *Introduction to information retrieval*. Cambridge: Cambridge University Press, 1st ed., 2008.

[56] W. W. Cohen, "Learning rules that classify e-mail," in *Proc. AAAI Spring Symposium on Machine Learning in Information Access*, p. 25, 1996.

[57] D. D. Lewis and K. A. Knowles, "Threading electronic mail: A preliminary study," *Information Processing and Management*, vol. 33, no. 2, pp. 209–217, 1997.

[58] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz, "A Bayesian approach to filtering junk e-mail," tech. rep., 2005.

[59] K. Lang, "Newsweeder: Learning to filter netnews," in *Proc. of the 12th International Conference on Machine Learning*, pp. 331–339, 1995.

[60] S. Chakrabarti, B. Dom, R. Agrawal, and P. Raghavan, "Using taxonomy, discriminants, and signatures for navigating in text databases," in *Proc. VLDB*, pp. 446–455, 1997.

[61] B. Liu and L. Zhang, "A survey of opinion mining and sentiment analysis," in *Mining Text Data* (C. C. Aggarwal and C. Zhai, eds.), pp. 415–463, Springer Science and Business Media, 2012.

[62] M. Lui, *Generalized language identification.* PhD thesis, the University of Melbourne, Australia, 2014.

[63] I. H. Witten, A. Moffat, and T. C. Bell, *Managing gigabytes: Compressing and indexing documents and images.* San Francisco, USA: Morgan Kaufmann, 1999.

[64] F. Sebastiani, "Machine learning in automated text categorization," *ACM Computing Surveys (CSUR)*, vol. 34, no. 1, pp. 1–47, 2002.

[65] P. McNamee, "Language identification: A solved problem suitable for undergraduate instruction," *Journal of Computing Sciences in Colleges*, vol. 20, no. 3, pp. 94–101, 2005.

[66] J. Tiedemann and N. Ljubešić, "Efficient discrimination between closely related languages," in *Proc. COLING 2012*, pp. 2619–2634, 2012.

[67] M. Zampieri, "Using bag-of-words to distinguish similar languages: How efficient are they?," in *Proc. of the 14th International Symposium on Computational Intelligence and Informatics (CINTI)*, pp. 37–41, 2013.

[68] G. Grefenstette, "Comparing two language identification schemes," in *Proc. of Analisi Statistica dei Dati Testuali (JADT)*, pp. 263–268, 1995.

[69] I. Suzuki, Y. Mikami, A. Ohsato, and Y. Chubachi, "A language and character set determination method based on n-gram statistics," *ACM Transactions on Asian Language Information Processing (TALIP)*, vol. 1, no. 3, pp. 269–278, 2002.

[70] H. Takçi and E. Ekinci, "Minimal feature set in language identification and finding suitable classification method with it," *Procedia Technology*, vol. 1, pp. 444–448, 2012.

[71] H. Takçı and T. Güngör, "A high performance centroid-based classification approach for language identification," *Pattern Recognition Letters*, vol. 33, no. 16, pp. 2077–2084, 2012.

[72] J. M. Prager, "Linguini: Language identification for multilingual documents," in *Proc. of the 32nd Annual Hawaii International Conference on Systems Sciences. HICSS-32*, pp. 11–16, 1999.

[73] W. B. Cavnar, J. M. Trenkle, *et al.*, "N-gram-based text categorization," in *Proc. of the 3rd Symposium on Document Analysis and Information Retrieval*, pp. 161–175, 1994.

[74] G. Adams and P. Resnik, "A language identification application built on the java client/server platform," in *Proc. of the ACL/EACL'97 Workshop on From Research to Commercial Applications: Making NLP Work in Practice*, pp. 43–47, 1997.

[75] Y. C. Chew, Y. Mikami, C. A. Marasinghe, and S. Nandasara, "Optimizing n-gram order of an n-gram based language identification algorithm for 63 written languages," *International Journal on Advances in ICT for Emerging Regions ICTer*, vol. 2, no. 2, pp. 21–28, 2009.

[76] T. Baldwin and M. Lui, "Language identification: The long and the short of the matter," in *Proc. of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT 2010)*, (Los, Angeles, USA), pp. 229–237, 2010.

[77] P. Vojtek and M. Bieliková, "Comparing natural language identification methods based on Markov processes," in *Proc. of the 4th International Seminar on Computer Treatment of Slavic and East European Languages*, pp. 271–282, 2007.

[78] P. Mcnamee and J. Mayfield, "Character n-gram tokenization for European language text retrieval," *Information retrieval*, vol. 7, no. 1-2, pp. 73–97, 2004.

[79] R. D. Brown, "Finding and identifying text in 900+ languages," *Digital Investigation*, vol. 9, pp. S34–S43, 2012.

[80] A. Simões, J. J. Almeida, and S. D. Byers, "Language identification: A neural network approach," in *Proc. of the 3rd Symposium on Languages, Applications and Technologies (SLATE 2014)*, pp. 251–265, 2014.

[81] Y. Yang and J. O. Pedersen, "A comparative study on feature selection in text categorization," in *Proc. of the 14th International Conference on Machine Learning ICML-97*, pp. 412–420, 1997.

[82] A. N. Aizawa, "Linguistic techniques to improve the performance of automatic text categorization.," in *Proc. of the 6th Natural Language Processing Pacific Rim Symposim NLPRS*, pp. 307–314, 2001.

[83] F. Peng, D. Schuurmans, and S. Wang, "Augmenting naive Bayes classifiers with statistical language models," *Information Retrieval*, vol. 7, no. 3-4, pp. 317–345, 2004.

[84] D. D. Lewis and M. Ringuette, "A comparison of two learning algorithms for text categorization," in *Proc. of the 3rd Annual Symposium on Document Analysis and Information Retrieval SDAIR 94*, pp. 81–93, 1994.

[85] G. Forman, "An extensive empirical study of feature selection metrics for text classification," *The Journal of Machine Learning Research*, vol. 3, pp. 1289–1305, 2003.

[86] H. Schütze, D. A. Hull, and J. O. Pedersen, "A comparison of classifiers and document representations for the routing problem," in *Proc. of the 18th annual International Conference on Research and Development in Information Retrieval*, pp. 229–237, 1995.

[87] E. Wiener, J. O. Pedersen, A. S. Weigend, *et al.*, "A neural network approach to topic spotting," in *Proc. of the 4th Annual Symposium on Document Analysis and Information Retrieval SDAIR-95*, pp. 332–344, 1995.

[88] Y. Yang, "Noise reduction in a statistical approach to text categorization," in *Pro. of the 18th Annual International Conference on Research and Development in Information Retrieval*, pp. 256–263, 1995.

[89] Y. Yang and J. Wilbur, "Using corpus statistics to remove redundant words in text categorization," *JASIS*, vol. 47, no. 5, pp. 357–369, 1996.

[90] I. Moulinier, G. Raskinis, and J. Ganascia, "Text categorization: A symbolic approach," in *Proc. of the 5th Annual Symposium on Document Analysis and Information Retrieval*, pp. 87–99, 1996.

[91] W. Shang, H. Huang, H. Zhu, Y. Lin, Y. Qu, and Z. Wang, "A novel feature selection algorithm for text categorization," *Expert Systems with Applications*, vol. 33, no. 1, pp. 1–5, 2007.

[92] E. Moyotl-Hernández and H. Jiménez-Salazar, "Enhancement of DTP feature selection method for text categorization," in *Proc. of the Computational Linguistics and Intelligent Text Processing*, pp. 719–722, 2005.

[93] S. Li and C. Zong, "A new approach to feature selection for text categorization," in *Proc. International Conference on Natural Language Processing and Knowledge Engineering, 2005. IEEE NLP-KE'05*, pp. 626–630, 2005.

[94] R. D. Brown, "Selecting and weighting n-grams to identify 1100 languages," in *Proc. of the 16th International Conference on Text, Speech and Dialogue (TSD 2013)*, pp. 475–483, 2013.

[95] T. Dunning, *Statistical identification of language.* Technical Report MCCS 940-273, Computing Research Laboratory, New Mexico State University, 1994.

[96] W. J. Teahan, "Text classification and segmentation using minimum cross-entropy," in *Proc. of the 6th International Conference Recherche d'Information Assistee par Ordinateur (RIAO' 00)*, pp. 943–961, 2000.

[97] W. J. Teahan and D. J. Harper, "Using compression-based language models for text categorization," in *Language Modeling for Information Retrieval* (B. Croft and J. Lafferty, eds.), pp. 141–165, Springer Science and Business Media, 2003.

[98] M. Majliš, "Yet another language identifier," in *Proc. of the Student Research Workshop at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, (Avignon, France), pp. 46–54, 2012.

[99] T. Mandl, M. Shramko, O. Tartakovski, and C. Womser-Hacker, "Language identification in multi-lingual web-documents," in *Proc. of the 11th International Conference on Applications of Natural Language to Information Systems (NLDB 2006)*, (Klagenfurt, Austria), pp. 153–163, 2006.

[100] E. Tromp and M. Pechenizkiy, "Graph-based n-gram language identification on short texts," in *Proc. of the 20th Machine Learning Conference of Belgium and The Netherlands*, (The Hague, Netherlands), pp. 27–34, 2011.

[101] S. Johnson, "Solving the problem of language recognition," tech. rep., Technical report, School of Computer Studies, University of Leeds, 1993.

[102] R. D. Lins and P. Gonçalves, "Automatic language identification of written texts," in *Proc. ACM Symposium on Applied Computing*, pp. 1128–1133, 2004.

[103] J. Vogel and D. Tresner-Kirsch, "Robust language identification in short, noisy texts: Improvements to liga," in *Proc. of the 3rd International Workshop on Mining Ubiquitous and Social Environments (MUSE 2012)*, (Bristol, UK), pp. 43–50, 2012.

[104] O. Giwa and M. H. Davel, "Language identification of individual words with joint sequence models," in *Proc. 15th Annual Conference of the International Speech Communication Association, INTERSPEECH*, (14-18 September, Singapore), pp. 1400–1404, 2014.

[105] M. Lui and T. Baldwin, "langid. py: An off-the-shelf language identification tool," in *Proc. of the ACL 2012 System Demonstrations*, (Jeju Island, Korea), pp. 25–30, 2012.

[106] S. Bergsma, P. McNamee, M. Bagdouri, C. Fink, and T. Wilson, "Language identification for creating language-specific twitter collections," in *Proc. of the Second Workshop on Language in Social Media*, pp. 65–74, 2012.

[107] S. Carter, W. Weerkamp, and M. Tsagkias, "Microblog language identification: Overcoming the limitations of short, unedited and idiomatic text," *Language Resources and Evaluation*, vol. 47, pp. 195–215, 2013.

[108] M. Goldszmidt, M. Najork, and S. Paparizos, "Boot-strapping language identifiers for short colloquial postings," in *Proc. of the Machine Learning and Knowledge Discovery in Databases*, pp. 95–111, 2013.

[109] B. King and S. P. Abney, "Labeling the languages of words in mixed-language documents using weakly supervised methods.," in *Proc. of the NAACL-HLT*, pp. 1110–1119, 2013.

[110] M. Lui, J. H. Lau, and T. Baldwin, "Automatic detection and language identification of multilingual documents," *Transactions of the Association for Computational Linguistics*, vol. 2, pp. 27–40, 2014.

[111] T. Baldwin and M. Lui, "Multilingual language identification: Altw 2010 shared task dataset," in *Proc. of the Australasian Language Technology Workshop (ALTW 2010)*, (Melbourne, Australia), pp. 5–7, 2010.

[112] T. Gottron and N. Lipka, "A comparison of language identification approaches on short, query-style texts," in *Advances in Information Retrieval*, pp. 611–614, Springer, 2010.

[113] H. Ceylan and Y. Kim, "Language identification of search engine queries," in *Proc. of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, (Singapore), pp. 1066–1074, 2009.

[114] D. Nguyen and A. S. Dogruoz, "Word level language identification in online multilingual communication," in *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, (Seattle, USA), pp. 857–862, 2014.

[115] K. Stasinos, "What's in a name? quite a lot," in *Proc. of the 2007 Conference on Recent Advances in Natural Language Processing (RANLP-07)*, (Borovets, Bulgaria), 2007.

[116] T. Fawcett, "ROC graphs: Notes and practical considerations for data mining researchers," tech. rep., Technical report hpl-2003-4, HP Laboratories, Palo Alto, CA, USA, 2003.

[117] Y. Yang and X. Liu, "A re-examination of text categorization methods," in *Proc. of the 22nd Annual International Conference on Research and Development in Information Retrieval, (ACM SIGIR)*, pp. 42–49, 1999.

[118] C. J. van Rijsbergen, *Information retrieval*. 2nd ed., 1979.

[119] Y. Yang, "An evaluation of statistical approaches to text categorization," *Information Retrieval*, vol. 1, no. 1-2, pp. 69–90, 1999.

[120] K. R. Beesley, "Language identifier: A computer program for automatic natural-language identification of on-line text," in *Proc. of the 29th Annual Conference of the American Translators Association*, pp. 47–54, 1988.

[121] H. Yamaguchi and K. Tanaka-Ishii, "Text segmentation by language using minimum description length," in *Proc. of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012)*, (Jeju Island, Korea), pp. 969–978, 2012.

[122] K. P. Scannell, "The crúbadán project: Corpus building for under-resourced languages," in *Proc. of the 3rd Web as Corpus Workshop : Building and Exploring Web Corpora*, (Louvain-la-Neuve, Belgium), pp. 5–15, 2007.

[123] W. D. Lewis and F. Xia, "Developing odin: A multilingual repository of annotated language data for hundreds of the world's languages," *Literary and Linguistic Computing*, vol. 25, no. 3, pp. 303–319, 2010.

[124] P. Resnik, "Mining the web for bilingual text," in *Proc. of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics*, (College Park, USA), pp. 527–534, 1999.

[125] E. Tromp, *Multilingual sentiment analysis on social media.* Master's thesis, department of mathematics and computer science, Eindhoven University of Technology, 2011.

[126] H. C. Hyde-Thomson and R. Liron, "Unified messaging system with automatic language identification for text-to-speech conversion," Nov. 26 2002. US Patent 6,487,533.

[127] J. W. Thirion, M. H. Davel, and E. Barnard, "Multilingual pronunciations of proper names in a Southern African corpus," in *Proc. of the 23rd Annual Symposium of the Pattern Recognition Association of South Africa (PRASA 2012)*, pp. 102–108, 2012.

[128] J. Gustafson, "ONOMASTICA - creating a multi-lingual dictionary of European names," *Lund Working Papers in Linguistics*, vol. 43, pp. 66–69, 2009.

[129] H. van den Heuvel, J.-P. Martens, K. D'hanens, and N. Konings, "The Autonomata Spoken Names Corpus," in *Proc. LREC*, (Marrakech, Marocco), pp. 140–143, 2008.

[130] H. van den Heuvel, B. Réveil, and J.-P. Martens, "Pronunciation-based ASR for names," in *Proc. the 10th Annual Conference of the International Speech Communication Association (INTERSPEECH 2009)*, pp. 2991–2994, 2009.

[131] E. Barnard, M. Davel, and C. van Heerden, "ASR corpus design for resource-scarce languages," in *Proc. the 10th Annual Conference of the International Speech Communication Association (INTERSPEECH 2009)*, pp. 2847–2850, 2009.

[132] P. Zulu, G. Botha, and E. Barnard, "Orthographic measures of language distances between the official South African languages," *Literator: Journal of Literary Criticism, Comparative Linguistics and Literary Studies: Human Language Technology for South African Languages: Special Issue 1*, vol. 29, pp. 185–204, 2008.

[133] N. J. de Vries, J. Badenhorst, M. H. Davel, E. Barnard, and A. de Waal, "Woefzela-An open-source platform for ASR data collection in the developing world," in *Proc. the 12th Annual Conference of the International Speech Communication Association (INTERSPEECH 2011)*, pp. 3177–3180, 2011.

[134] M. Kgampe and M. Davel, "Consistency of cross-lingual pronunciation of South African personal names," in *Proc. of the 21st Annual Symposium of the Pattern Recognition Association of South Africa (PRASA 2010)*, (Stellenbosch, South Africa), pp. 123–127, Nov. 2010.

[135] M. Davel and O. Martirosian, "Pronunciation dictionary development in resource-scarce environments," in *Proc. of the 10th Annual Conference of the International Speech Communication Association (INTERSPEECH 2009)*, (Brighton, United Kingdom), pp. 2851–2854, Sep. 2009.

[136] Thirion, Jan W.F. and van Heerden, Charl and Giwa, Oluwapelumi and Davel, Marelie H., "The South African Directory Enquiries (SADE) corpus." to be Submitted.

[137] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian network classifiers," *Machine Learning*, vol. 29, no. 2-3, pp. 131–163, 1997.

[138] S. Eyheramendy, D. D. Lewis, and D. Madigan, "On the naive Bayes model for text categorization," in *Proc. of the 9th International Workshop on Articial Intelligence and Statistics AISTATS 2003*, 2003.

[139] A. McCallum, K. Nigam, *et al.*, "A comparison of event models for naive Bayes text classification," in *Proc. Workshop on Learning for Text categorization AAAI-98*, pp. 41–48, 1998.

[140] Y. Yang, *Discretization for Naive-Bayes learning.* Ph.D. Thesis, School of Computer Science and Software Engineering of Monash University, 2003.

[141] C. C. Aggarwal and C. Zhai, "A survey of text classification algorithms," in *Mining Text Data* (C. C. Aggarwal and C. Zhai, eds.), Springer Science and Business Media, 2012.

[142] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," *Computer Speech and Language*, vol. 13, no. 4, pp. 359–393, 1999.

[143] F. Jelinek, "Self-organized language modeling for speech recognition," in *Readings in Speech Recognition* (A. Waibel and K.-F. Lee, eds.), pp. 450–506, Morgan Kaufman Publishers, 1990.

[144] J. M. Ponte and W. B. Croft, "A language modeling approach to information retrieval," in *Proc. of the 21st Annual International Conference on Research and Development in Information Retrieval*, pp. 275–281, 1998.

[145] D. Hiemstra, *Using language models for information retrieval.* PhD thesis, University of Twente, Enschede, The Netherlands, 2001.

[146] D. Jurafsky and J. H. Martin, *Speech and language processing.* Pearson Education India, 2000.

[147] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," in *Proc. of the 34th Annual Meeting of the Association for Computational Linguistics ACL*, pp. 310–318, 1996.

[148] S. Katz, "Estimation of probabilities from sparse data for the language model component of a speech recognizer," in *Proc. of the 12th Annual International Conference on Acoustics, Speech and Signal Processing ICASSP*, pp. 400–401, 1987.

[149] A. M. M. Hasan, S. Islam, and M. A. Rahman, "A comparative study of Witten Bell and Kneser-Ney smoothing methods for statistical machine translation," *Journal of Information Technology*, vol. 1, pp. 1–6, 2012.

[150] H. Ney, U. Essen, and R. Kneser, "On structuring probabilistic dependences in stochastic language modelling," *Computer Speech and Language*, vol. 8, no. 1, pp. 1–38, 1994.

[151] F. Jelinek, "Up from trigrams," in *Proc. EUROSPEECH*, pp. 1037–1040, 1991.

[152] T. Glasmachers and C. Igel, "Maximum likelihood model selection for 1-norm soft margin SVMs with multiple parameters," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, no. 8, pp. 1522–1528, 2010.

[153] V. N. Vapnik and S. Kotz, *Estimation of dependences based on empirical data.* Springer-Verlag New York, 1982.

[154] B. Schölkopf, K. Tsuda, and J.-P. Vert, *Kernel methods in computational biology.* MIT Press, 2004.

[155] J. Shawe-Taylor and N. Cristianini, *Kernel methods for pattern analysis.* Cambridge, MA: Cambridge University Press, 2004.

[156] B. Schölkopf and A. J. Smola, *Learning with kernels: Support vector machines, regularization, optimization, and beyond.* MIT Press, 2002.

[157] C. J. van Heerden, *Efficient training of support sector machines and their hyperparameters.* Ph.D. Thesis, North-West University, Potchefstroom, 2013.

[158] A. Ben-Hur and J. Weston, "A user's guide to support vector machines," in *Data Mining Techniques for the Life Sciences* (O. Carugo and F. Eisenhaber, eds.), pp. 223–239, 2010.

[159] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[160] N. Cristianini and J. Shawe-Taylor, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge, MA: Cambridge University Press, 2000.

[161] S. S. Keerthi and C.-J. Lin, "Asymptotic behaviors of support vector machines with Gaussian kernel," *Neural Computation*, vol. 15, no. 7, pp. 1667–1689, 2003.

[162] D. Anguita, S. Ridella, F. Rivieccio, and R. Zunino, "Hyperparameter design criteria for support vector classifiers," *Neurocomputing*, vol. 55, no. 1, pp. 109–134, 2003.

[163] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[164] H. Kamper, F. De Wet, T. Hain, and T. Niesler, "Resource development and experiments in automatic SA broadcast news transcription," in *Proc. of the third International Workshop on Spoken Languages Technologies for Under-resourced Languages (SLTÚ12)*, 2012.

[165] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *The Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.

[166] S. Deligne and F. Bimbot, "Language modeling by variable length sequences: Theoretical formulation and evaluation of multigrams," in *Proc. International Conference on Acoustics, Speech, and Signal Processing, Detroit, MI, USA*, pp. 169–172, 1995.

[167] M. H. Davel, W. D. Basson, C. van Heerden, and E. Barnard, "NCHLT Dictionaries: Project Report," tech. rep., Multilingual Speech Technologies, North-West University, May 2013.

[168] E. Barnard, M. H. Davel, C. van Heerden, F. de Wet, and J. Badenhorst, "The NCHLT speech corpus of the South African languages," in *Proc. of the 4th Workshop on Spoken Language Technologies for Under-resourced Languages*, (St. Peterburg, Russia), pp. 194–200, 2014.

[169] P. Auer, *Code-switching in conversation: Language, interaction and identity*. Routledge, 2002.

[170] O. Giwa, M. H. Davel, and E. Barnard, "A Southern African corpus for multilingual name pronunciation," in *Proc. Annual Symposium of the Pattern Recognition Association of South Africa (PRASA)*, (Vanderbijlpark, South Africa), pp. 49–53, 2011.

[171] M. H. Davel, C. J. van Heerden, and E. Barnard, "Validating smartphone-collected speech corpora," in *of the third International Workshop on Spoken Languages Technologies for Under-resourced Languages (SLTÚ12)*, pp. 68–75, 2012.

[172] M. H. Davel, C. J. van Heerden, and E. Barnard, "G2P variant prediction techniques for ASR and STD," in *Proc. the 14th Annual Conference of the International Speech Communication Association (INTERSPEECH 2013), pages=1831–1835, year=2013*.

[173] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*, IEEE Signal Processing Society, Dec. 2011. IEEE Catalog No.: CFP11SRW-USB.

[174] Wikipedia, "Germanic languages." `http://en.wikipedia.org/wiki/Germanic_languages`. Accessed: 2015-03-12.

Appendix A

Using G-Translate for language verification

The G-Translate technique employs Google Translate API[1]. The G-Translate method is used in this study for the preprocessing task and language verification of generic words. In this section, the term source-language is used to refer to the current classification of each word (as obtained using alternative techniques).

For preprocessing tasks, G-Translate confirms if a word indeed exists in the source language by translating it to one or more different languages, referred to from here onwards as the 'proxy languages'. These languages can be any of the Google Translate API languages. G-Translate can also be used to check for wrongly spelled words in the source language.

Selecting a suitable proxy language depends on its web availability. Google Translate API uses English as its default source language for any translation. To obtain good translation results, proxy languages should be selected outside the source language group. Proxy languages that have fewer borrowed words from the source language will be a better option. For example, translating from Afrikaans to Dutch (used as a proxy language) is not recommended, the reason being that most words from Afrikaans originated from Dutch and invariably classified in the same language group. A thorough understanding of language groups associated with the source languages is paramount for the best performance outcome [174].

A.1 Experimental set-up

The applicability of G-Translate to preprocessing task in the context of a two-language task is analysed. The effects of using varied counts of proxy languages are analysed. The initial task

[1] \https://cloud.google.com/translate/v2/using_rest

was to apply this technique to four languages namely Afrikaans, English, isiZulu and Sesotho, which are widely spoken in Gauteng, South Africa. Sesotho was excluded from the current task because the language is not available on Google API lists of languages (as at the time this experiment was conducted), while isiZulu (although present in the API language lists) was also excluded because of lack of first-language speakers when this experiment was carried out.

A.1.1 Data set

The particular task involves only two South African languages (English, Afrikaans, see section A.1 for reasons). A data set was obtained from the same source as discussed earlier in Section 4.6.1. During development, a word-list of 12 000 unique words was randomly selected from the original 15 000 words with character length greater than one.

The experimental word-list was previously classified using alternative LID techniques either manually or by any of the techniques previous discussed in Chapter 4.

A.1.2 Selecting proxy languages

Users' discretion is advised when selecting proxy languages. As discussed above, any language selected as proxy language must be present in the Google Translate API list of languages.

It should be noted that care must be taken when selecting languages that do not employ 'Latin alphabets'. Experiments show that using non-Latin alphabet languages such as Japanese, Chinese, or Russian, automatically changes the orthography of a word regardless of its existence in the source language. For example, the non-English word 'horosho' is translated to 'ХОРОШО' in Russian, although, this is the English alphabet-to-alphabet translation of the Russian word in context.

A.1.3 Experimental approach

In this experiment, the approach to affirming a word that indeed originates from a source language can be sub-categorised into two parts:

- Translating from non-English source languages: The threshold value, τ, for which a word is accepted to originate from a source language is given as:

$$\tau \geq X/2 \tag{A.1}$$

where τ is the threshold value that corresponds to the number of times a word changes orthography across X proxy languages. X represents the number of proxy languages employed.

- Translating from English source language: With the large-scale usage of English language words across all domains, translating from English takes a different perspective. The threshold value can be estimated as:

$$\tau \geq 1. \tag{A.2}$$

A.1.4 Evaluation metrics

Classification accuracy is employed as evaluation metrics for language verification using the G-Translate technique. As discussed earlier in section 2.6, classification accuracy is the percentage of correct (both negative and positive) instances that were predicted. For clarity purposes, the appropriate measures used for distribution of the data set is used.

A.2 Experiments and results

Words in the source language are translated to the X proxy languages without remapping them back to the source language. After translation, we obtain two result sets, 'error-labelled words' and 'correct-labelled words'. Data contained in either of the two sets depends on the source language and threshold value, τ. Error-labelled result sets are words G-Translate deems not from the source language, that is, words that keep the orthography of the original source language based on the threshold value. Correct-labelled refers to words G-Translate tags as a genuine word in that source language; that is, words that change orthography with respect to the threshold value.

In order to report performance analysis, we request source language first-language speakers to review each word in the two result sets. During the review process, words that are problematic (incorrectly classified in a particular set) are tagged while leaving out words that correctly belong to that set untagged. The newly reviewed word-list was used as reference set while our technique results were used as response set.

Beacuse of the size of the correct-labelled set, 50 words were randomly selected from the set for performance analysis; however, in the error-labelled set, all words deemed as erroneous were used during the analysis.

A.2.1 G-Translate Baseline

In order to obtain a baseline result, two proxy languages were used, where X = Spanish and French. These two are languages with significant web presence and an influence on

other languages. Moreover, English (believed to be the most influential language in the current age) was excluded from the proxy languages because it is already a source language and also the default source language for Google Translate API.

TABLE A.1: Performance achieved with G-Translate baseline on two proxy languages using English as the source language.

	TP	TN	FN	FP	*Accuracy*
Error-labelled	0	8	209	0	3.687
Correct-labelled	50	0	0	0	100.000

TABLE A.2: Performance achieved with G-Translate baseline on two proxy languages using Afrikaans as the source-language.

	TP	TN	FN	FP	*Accuracy*
Error-labelled	0	121	32	0	79.085
Correct-labelled	44	0	0	6	88.000

Tables A.1 and A.2 show the performance accuracy of G-Translate on two proxy languages for both Afrikaans and English source languages. In Table A.1, the true negative of '8' for the 'error-labelled' word-list indicates the number of correctly rejected words that exist in the reference and response sets. The false negative of '209' represents the counts of incorrectly rejected words that G-Translate tagged as wrong words in the source language.

In Table A.2, the true positive of '44' for correctly labelled shows the counts of the correctly identified words that G-Translate tagged as genuine. The false positive of '6' refers to the total counts of incorrectly identified words G-Translate tagged as genuine.

In Table A.1, low performance on the English 'error-labelled' set is obtained because most misclassified words are either regarded as the genuine word in both proxy languages (with a different meaning) or proper names. For example, words such as; original, musical, ex, animal, focal and max, carry the same orthography in the two proxy languages. In Table A.2, the Afrikaans 'error-labelled' set, good performance accuracy was obtained where most of the words G-Translate presumed wrong or incorrect were spelling errors.

Good performance accuracy was obtained on the 50 randomly selected words for the 'correct-labelled' result sets in Tables A.1 and A.2. A drop in performance accuracy value to 88% was due to spelling errors. For example, words such as stormwaterbestuur, assesseringsvereistes and omgewingsmagtiging were tagged as incorrect by the reviewers. According to the reviewers, these words were combinations of two separate words, hence should contain spaces for them to be correctly tagged as Afrikaans words.

A.2.2 Using more proxy languages with significant web presence

In this section, the X parameter value was set to 4, which means using four proxy languages. Proxy languages employed include German, Spanish, Italian and French, which

all have a significant web presence.

TABLE A.3: Performance achieved with G-Translate on four proxy languages using English as the source language.

	TP	TN	FN	FP	*Accuracy*
Error-labelled	0	7	62	0	10.145
Correct-labelled	50	0	0	0	100.000

TABLE A.4: Performance achieved with G-Translate on four proxy languages using Afrikaans as the source language. .

	TP	TN	FN	FP	*Accuracy*
Error-labelled	0	126	27	0	82.353
Correct-labelled	38	0	0	12	76.000

Tables A.3 and A.4 show performance accuracy of G-Translate on four proxy languages for both Afrikaans and English as source languages. In Table A.3, the true negative value of '7' for the 'error-labelled' word-list refers to the number of correctly rejected words that exist in both the reference and response sets. The false negative value of '62' represents counts of incorrectly rejected words that G-Translate tagged as wrong words in English.

In Tables A.3 and A.4, It was possible to improve the performance accuracy of the 'error-labelled' list from approximately '4%' in the baseline to over '10%'. Reasons for poor verification could be associated with high numbers of loan or borrowed words that exist with the same orthography across the four proxy languages.

Furthermore, for the 'correct-labelled' set in Tables A.3 and A.4, good performance is observed except for the increase in the number of incorrectly identified Afrikaans words. This surge in counts is associated with more misspelled words (mostly combination of two separate words), which were wrongly identified as Afrikaans words by the G-Translate technique.

A.2.3 Using more proxy languages with less web presence

This section poses the question whether languages with limited web presence will help improve the performance accuracy over the result obtained in section A.2.2. Most languages that fall into this category(languages with less web presence) are under-resourced languages such as African, East European and a few Asian languages. For this experiment, the X parameter value was set to 4. Languages used include Polish, Irish, Indonesian, and Turkish.

Tables A.5 and A.6 show results obtained by comparing reference and response sets on four proxy languages with limited web presence.

TABLE A.5: Performance achieved with G-Translate on four proxy languages using English as the source language.

	TP	TN	FN	FP	*Accuracy*
Error-labelled	0	7	33	0	17.500
Correct-labelled	50	0	0	0	100.000

TABLE A.6: Performance achieved with G-Translate on four proxy languages using Afrikaans as the source language.

	TP	TN	FN	FP	*Accuracy*
Error-labelled	0	124	25	0	83.222
Correct-labelled	37	0	0	13	74.000

In Tables A.5 and A.6, it was possible to improve the performance accuracy of the error-labelled list from approximately 4% in the baseline to over 17%. For the 'correct-labelled' set, good performance was observed with English while performance dropped on the Afrikaans data set. After reviewing the output, it was noticed that more misspelled words were wrongly identified as correct Afrikaans words.

TABLE A.7: Accuracy obtained on comparison between G-Translate baseline and 'using more proxy languages'.

	Error-labelled		Correct-labelled	
Technique	Afrikaans(%)	English(%)	Afrikaans(%)	English(%)
2-proxy	79.085	3.687	88.000	100.000
4-proxy (strong web)	82.353	10.145	76.000	100.000
4-proxy (limited web)	83.222	17.500	74.000	100.000

Table A.7 shows a comprehensive comparison between G-Translate baseline and 'four-proxy languages' experiments. Accuracy increases gradually on the 'error-labelled' set for both source languages as X increases from 2 to 4, with the highest accuracy value obtained on $X = 4$ with less web presence.

In conclusion, overall, it is observed that performance decreases on Afrikaans 'correct-labelled' as X increases from 2 to 4, where the baseline produces the best outcome. This reduction in accuracy could be associated with more misspelled words being added as we increase the X parameter value from two to four languages. Some of the misclassified words are also observed as loan words in languages selected for limited web presence, which has a drastic effect on the final accuracy.

In conclusion, we can indirectly deduce that language verification of under-resourced languages perform well with a lower number of proxy languages with significant web presence. However, source languages with a considerable proportion of web presence will result in

better performance when verified on a higher number of proxy languages with limited web presence.

Appendix B

Phoneme set

The 'detailed' and 'combined' phonemes are show in Table B.1.

	X-SAMPA				
Description	NCHLT	Multipron	SADE	Detailed	Combined
Dental click	!\	!\	!\	!\	!\
Voiced dental click	!\g_0	-	-	!\g_0	!\ g
Aspirated dental click	!\h	-	-	!\h	!\
Rounded mid-high	2:	2:	-	2:	i 9
Rounded mid-low central vowel with duration	3:	3:	-	3:	9
Rounded mid-low front vowel	9	9	9	9	9
Diphthongs	9y	9y	-	9y	9 i
Central vowel (schwa)	@	@	@	@	@
Diphthongs	@i	@i	-	@ i	@ i
Diphthongs	@u	@u	-	@ u	@ u
Unrounded low back vowel with duration	A:	A:	A:	A:	A:
Voiced bilabial postalveolar fricative	BZ	-	-	BZ	b Z
Voiced dental fricative	D	D	D	D	D
Unrounded mid-low front vowel	E	E	E	E	E
Palatal nasal	J	J	-	J	J
Voiceless alveolar lateral fricative	K	K	K	K	K
Voiced alveolar lateral fricative	K\	K\	K\	K\	K\
Velar nasal	N	N	N	N	N
Rounded mid-low back vowel	O	O	O	O	O
Rounded mid-low back vowel with duration	O:	-	-	O:	O
Diphthongs	Oi	Oi	-	O i	O i
Rounded low back vowel	Q	Q	Q	Q	Q
Voiceless post-alveolar fricative	S	S	S	S	S

Voiceless dental fricative	T	T	T	T	T
Rounded near-high near-back vowel	U	-	-	u	u
Voiced post-alveolar fricative	Z	Z	Z	Z	Z
Unrounded low front vowel	a	a	a	a	a
Diphthongs	ai	ai	-	a i	a i
Diphthongs	au	au	-	a u	a u
Voiced bilabial plosive	b	b	b	b	b
Voiced bilabial implosive	b_<	-	-	b_<	b
Voiced alveolar plosive	d	d	d	d	d
Voiced post-alveolar affricate	d_0Z	-	-	d Z	d Z
Voiced alveolar affricate	dz	-	-	d z	d z
Diphthongs	e@	e@	-	e @	e @
Voiceless labiodental fricative	f	f	f	f	f
Labiodantal-postalveolar fricative	fS	-	-	f S	f S
Voiced velar plosive	g	g	g	g	g
Voiceless glottal fricative	h	h	h	h	h
Voiced glottal fricative	h\	-	-	h	h
Unrounded high front vowel	i	i	i	i	i
Unrounded high front vowel with duration	i:	-	-	i:	i
Diphthongs	i@	i@	-	i @	i @
Palatal approximant	j	j	j	j	j
Voiceless velar plosive	k	k	k	k	k
Voiceless velo-alveolar lateral affricate	kK_>	-	-	k K\	k K\
Ejective velar plosive	k_>	-	-	k	k
Voiceless aspirated velar plosive	k_h	-	-	k_h	k
Voiceless velar affricate	kx	kx	-	k x	k x
Alveolar lateral approximant	l	l	l	l	l
Bilabial nasal	m	m	m	m	m
Alveolar nasal	n	n	n	n	n
Rounded low back vowel	-	-	o	Q	Q
Voiceless bilabial plosive	p	p	p	p	p
Voiceless labio-palatal ejective affricate	pS_>	-	-	p S	p S
Ejective bilabial plosive	p_>	-	-	p	p
Aspirated bilabial plosive	p_h	-	-	p	p
Aspirated labio-alveolar affricate	ps_h	-	-	p s_h	p s
Alveolar trill	r	r	r	r	r
Alveolar approximant	r\	r\	-	r\	r
Voiceless alveolar fricative	s	s	s	s	s
Voiceless alveolar plosive	t	t	t	t	t

Post-alveolar affricate	tS	-	-	t S	t S
Ejective post-alveolar affricate	tS_>	-	-	t S	t S
Aspirated post-alveolar affricate	tS_h	-	-	t S	t S
Ejective alveolar plosive	t_>	-	-	t	t
Aspirated alveolar plosive	t_h	-	-	t_h	t
Voiceless alveolar lateral ejective	tl_>	tl_>	-	t l	t l
Voiceless aspirated alveolar lateral plosive	tl_h	-	-	t l	t l
Ejective alveolar affricate	ts_>	-	-	t s	t s
Aspirated alveolar affricate	ts_h	-	-	t s_h	t s
Rounded high back vowel	u	u	u	u	u
Rounded high back vowel with duration	u:	-	-	u	u
Diphthongs	u@	u@	-	u @	u @
Voiced labiodental fricative	v	v	v	v	v
Voiced labio-velar approximant	w	w	w	w	w
Voiceless velar fricative	x	x	x	x	x
Rounded high front vowel	y	y	y	y	y
Voiced alveolar fricative	z	z	z	z	z
Unrounded mid-low front vowel	{	{	{	{	{
Dental click	|	|	|	|	|
Voiced dental click	|\g_0	-	-	|\g_0	| g
Aspirated dental click	|\h	-	-	|\h	|
Alveolar lateral click	||\	||\	||\	||\	||\
Voiced alveolar lateral click	||\g_0	-	-	||\g_0	||\ g
Aspirated alveolar lateral click	||\h	-	-	||\h	||\

Table B.1: NCHLT, Multipron, SADE phones mapped to 'detailed' and 'combined'.

Appendix C

Grapheme set

Table C.1, C.2, C.3, C.4 show the set of graphemes extracted from NCHLT, SADE and Multipron corpora. Each table contains language-specific graphemes across the three (3) corpora. It also shows shared and differences in graphemes as obtained from each corpus. NOTE: this table only shows single character alphabet and do not include sequences of multiple letters, which equate to certain phonemes.

Sesotho				
	Description	NCHLT	SADE	Multipron
Common across corpora	Vowel	a	a	a
	Vowel	e	e	e
	Vowel	i	i	i
	Vowel	o	o	o
	Vowel	u	u	u
Common across corpora	Consonants	b	b	b
		c	c	c
		d	d	d
		f	f	f
		g	g	g
		h	h	h
		j	j	j
		k	k	k
		l	l	l
		m	m	m
		n	n	n
		p	p	p
		r	r	r
		s	s	s

		t	t	t
		w	w	w
		y	y	y
Differences across corpora	Consonant	q	q	-
	Consonant	v	-	-

Table C.1: List of Sesotho graphemes extracted from NCHLT, Multipron, SADE corpora.

Afrikaans				
	Description	NCHLT	SADE	Multipron
Common across corpora	Vowel	a	a	a
	Vowel	e	e	e
	Part of a Diphthong	ë	ë	ë
	Vowel	i	i	i
	Vowel	o	o	o
	Vowel	u	u	u
	Vowel	y	y	y
Differences across corpora	Vowel	ê	-	-
	Part of a Diphthong	è	-	-
	Part of a Diphthong	-	é	é
	Part of a Diphthong	ï	-	-
	Part of a Diphthong	-	-	ô
	Part of a Diphthong	ö	-	ö
	Part of a Diphthong	-	-	ü
Common across corpora	Consonants	b	b	b
		c	c	c
		d	d	d
		f	f	f
		g	g	g
		h	h	h
		j	j	j
		k	k	k
		l	l	l
		m	m	m
		n	n	n
		p	p	p
		r	r	r
		s	s	s

		t	t	t
		v	v	v
		w	w	w
Differences across corpora	Consonant	-	-	q
	Consonant	-	-	x
	Consonant	-	z	z

Table C.2: List of Afrikaans graphemes extracted from NCHLT, Multipron, SADE corpora.

Isizulu				
	Description	NCHLT	SADE	Multipron
Common across corpora	Vowel	a	a	a
	Vowel	e	e	e
	Vowel	i	i	i
	Vowel	o	o	o
	Vowel	u	u	u
Common across corpora	Consonants	b	b	b
		c	c	c
		d	d	d
		f	f	f
		g	g	g
		h	h	h
		j	j	j
		k	k	k
		l	l	l
		m	m	m
		n	n	n
		p	p	p
		q	q	q
		s	s	s
		t	t	t
		v	v	v
		w	w	w
		x	x	x
		y	y	y
		z	z	z

Table C.3: List of Isizulu graphemes extracted from NCHLT, Multipron, SADE corpora.

English				
	Description	NCHLT	SADE	Multipron
Common across corpora	Vowel	a	a	a
	Vowel	e	e	e
	Vowel	i	i	i
	Vowel	o	o	o
	Vowel	u	u	u
Common across corpora	Consonants	b	b	b
		c	c	c
		d	d	d
		f	f	f
		g	g	g
		h	h	h
		j	j	j
		k	k	k
		l	l	l
		m	m	m
		n	n	n
		p	p	p
		q	q	q
		r	r	r
		s	s	s
		t	t	t
		v	v	v
		w	w	w
		x	x	x
		y	y	y
		z	z	z

Table C.4: List of English graphemes extracted from NCHLT, Multipron, SADE corpora.