ADDIS ABABA UNINIVERSITY
SCHOOL OF GRADUATE STUDIES
COLLEGE OF NATURAL SCIENCES
DEPARTMENT OF COMPUTER SCIENCE

Language Modeling for Amharic Automatic Speech Recognition Systems

BY

MULUGETA MEKONNEN

A THESIS SUBMITED TO
THE SCHOOL OF GRADUATE STUDIES OF ADDIS ABABA UNIVERSITY
IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE DEGREE OF MASTER
OF SCIENCE IN COMPUTER SCIENCE

MARCH, 2013

ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
COLLEGE OF NATURAL SCIENCES
DEPARTMENT OF COMPUTER SCIENCE

LANGUAGE MODELING FOR AMHARIC AUTOMATIC SPEECH RECOGNITION SYSTEMS

BY

MULUGETA MEKONNEN

ADVISOR: SEBSIBE HAILEMARIAM (PhD)

Signature of the board of examiners for Approval

	Name	**Signature**
1.	Dr. Sebsibe Hailemariam, Advisor	_____
2.	_____	_____
3.	_____	_____

ACKNOWLEDGMENTS

I would like to thank my advisor Dr. Sebsibe Hailemariam. His advice, support and enthusiasm have guided this research from the outset. It has been both great privilege and enormous fun to have worked with him.

My thanks also go to fellow workers of Ethiopian News Agency for good cooperation they have shown during text corpus gathering. Especially, the ICT office head, Mr Debebe helped me a lot on this matter. Thank you. I would also like to address my thankfulness to Andreas Stolcke and Dmytro Prylipko for the technical advices and support on the speech recognition experiments.

I feel obliged to thank all my colleagues for all the good times in and out of class. Particular thanks must go to Aynadis, Birhanu, Desalegn, Genet and Honelet for being wonderful friends.

This research would have not been possible without computing facility, financial, and moral supports from my brother Mikias Mekonnen. I gratefully acknowledge all the good ways you have been showing me since my childhood. “YIBRALIH”.

Finally, I would like to articulate my heartfelt thankfulness to my mother, Amsale Teketel, and the whole family who have been backing me up in all good and bad days of mine.

Mulugeta Mekonnen Belete

March, 2013

Dedicated to

My father Mekonnen Belete Bekele

and

My mother Amsale Teketel Mekuria

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

Acronym	Description
AASR	Amharic Automatic Speech Recognizer
AMI	Average Mutual Information
ARPA	Advanced Research Projects Agency
ASC	Automatic Spelling Correction
ASR	Automatic Speech Recognition
ATC	Amharic Text Corpus
DC	Document classification
ENA	Ethiopia News Agency
FLM	Factored Language Model
HMM	Hidden Markov Model
HTK	Hidden Markov Tool Kit
LM	Language Model
LMs	Language Models
MBLM	Morphology Based Language Models
MFCC	Mel Frequency Cepstral Coefficient
NLP	Natural Language Processing
OCR	Optical character recognition
OOV	Out of vocabulary
POS	Part Of Speech Tag
SCTLM	Speech Corpus Text Language Models
SLM	Statistical Language Model
SMT	Statistical Machine Translation
SRILM	Stanford Research Institute Language Model
SRS	Speech Recognition System

ABSTRACT

For automatic speech recognition and other NLP tasks to be effective, a language model plays a critical role by assigning a probability to hypothesized word sequence. Various researches have been done on acoustic modelling to improve performance of Amharic SRS with no considerable effort to supplement it with proper LM. The aim of this research is to build LM for Amharic, official language of the federal government of Ethiopia, and study how it improves the performance of Amharic SRS.

Accordingly, text corpus consisted of 9,079,766 tokens is prepared and various word n-gram, class-based and interpolated LMs are built using SRILM tool. Both perplexity and WRA metrics are used to evaluate the LMs. Though LMs of order 2 to 7 were built, a tetra-gram (4-gram) LM happens to be the best n-gram LM. Relative performance of different smoothing algorithms is also compared and unmodified Kneser-Ney smoothing out smarted all others. Moreover, interpolated models performed better than back-off models. With the aim of tackling data sparsity problem, different class-based LMs are also developed using IBM clustering algorithms for automatic grouping of words into clusters. Eventually, class-based LMs performed worse than word based LMs due to its generic nature. However, interpolating class-based with word-based models leads to considerable perplexity reduction over the pure word-based and class-based LMs.

The word n-gram, class-based and interpolated LMs are then finally integrated to the baseline speech recognizer which has 74.52% WRA in a lattice rescoring framework. Consequently, WRA results of 80.9%, 66.0% and 82.7% have been achieved using word based n-grams, class-based and interpolated LMs respectively. Overall, an absolute 8.18% WRA gain has been achieved as a result of applying the interpolated class-based LMs to the baseline recognizer and this clearly shows LM is an indispensable part of speech recognition task. Class-based language models resulted in improved perplexity and WRA results only when combined with word-based models. Therefore, using class-based language models as a complementary tool to the word-based models is rewarding.

Keywords: Amharic language modeling, Amharic class-based language modeling, Amharic text corpus.

CHAPTER ONE
INTRODUCTION

Language is one of the distinct character of human being and is an important component of our lives. In written form it serves as a tool to transfer knowledge from one generation to the next. In spoken form it serves as a primary means of communication with others to accomplish our day-to-day activity. Natural Language Processing (NLP), therefore, is an area of research and application that explores how computers can be used to understand and manipulate natural language text or speech to carry out useful tasks[15]. Some of these useful tasks are Automatic Speech Recognition (ASR), Machine Translation (MT), optical character recognition (OCR), question answering (QA) and so on.

One important application area of NLP is automatic speech recognition (ASR). ASR is a task that involves the conversion of speech waveform (automatically) into sequence of words (texts)[12]. The goal, thus, is to create machines that can receive spoken information and act appropriately upon that information [34]. ASR has many real world applications such as automated call centres (eg. conversational agent that guide travellers to reserve and get arrival and departure information), embedded systems (eg. speech recognizer embedded in cars that enables drivers to control their environment by voice), voice enabled searching and so on[15]. ASR systems may be categorized as isolated or continuous speech, speaker dependent or independent, small, medium or large vocabulary system, read or spontaneous speech and noisy or noise free speech [34].

Isolated word recognition systems are intended to recognize individual word (word surrounded by long pause). In such systems, like digit recognizer, words are equally probable. Systems that do not require pauses between the words and that allow complete sentences to be spoken are called continuous speech recognition systems.. However, in this case all words are not equally probable and a Language model, therefore, is often required to predict the likelihood of word sequences.

speaker independent ASR systems are able to recognize speech from people whose speech has never been used during system training. This requires a lot of speech corpus recorded from different speakers. On the contrary, speaker dependent ASR systems recognize speech of a speaker whose speech is the only training data used for the development of the recognizer. These systems perform worse for other speakers.

Vocabulary size used by the recognizer is the other parameter that differentiate ASR systems. According to [34], small vocabulary tasks have less than 100 vocabulary entry while medium vocabulary have between 100 and 999 words. Large vocabulary systems are those having a vocabulary of roughly more than 1000 words. Although, there existed other approaches, the dominant paradigm for large vocabulary continuous speech recognition (LVCSR) is the HMM.

1.1 Speech Recognition Model

This thesis is about language modelling for speech recognition. This section intends to show the role of language model (LM) for a typical speech recognition system.

Given some acoustic input O the goal of speech recognizer is to output the best string of words W which it can find to match this input. The assumption is that a specified word sequence, W, produces an acoustic observation sequence O. This intuition could be modelled via the use of a statistical model of the joint distribution p(W,O) of the sequence of spoken words W and the corresponding observed sequence of acoustic information O. The goal is then to decode the word string, based on the acoustic observation sequence, so that the decoded string has the maximum a posteriori probability. Accordingly, The recognition task can be expressed by :

$$\widehat{W} = arg\max_{W} P(W|O) \qquad (1.1)$$

According to Bayes rule, any probability P(x|y) can be broken down into a form

$$\mathrm{P(x|y)} = \frac{\mathrm{P(y|x)P(x)}}{\mathrm{P(y)}} \qquad (1.2)$$

Substituting equation 1.1 into 1.2

$$\widehat{W} = arg\max_{W} \frac{p(W)p(O|W)}{p(O)} \qquad (1.3)$$

P(O) can be ignored since for each potential sentence the same observations O will be examined, which must have the same probability P(O). Thus equation 1.3 can be reduced to:

$$\widehat{W} = arg \max_{W} P(W)P(O|W) \qquad (1.4)$$

Where p(W) is the probability of the sequence of n words ($w_i..w_n$) and p(O|W) is the probability of observing the acoustic evidence O when the sequence W is spoken. The a priori distribution p(W) of what words might be spoken is referred to as a language model (LM). The observation probability model p(O|W) is called the acoustic model. One can see from this equation that language model is an indispensable part of a speech recognizer and plays a critical role for the performance of the speech recognition system.

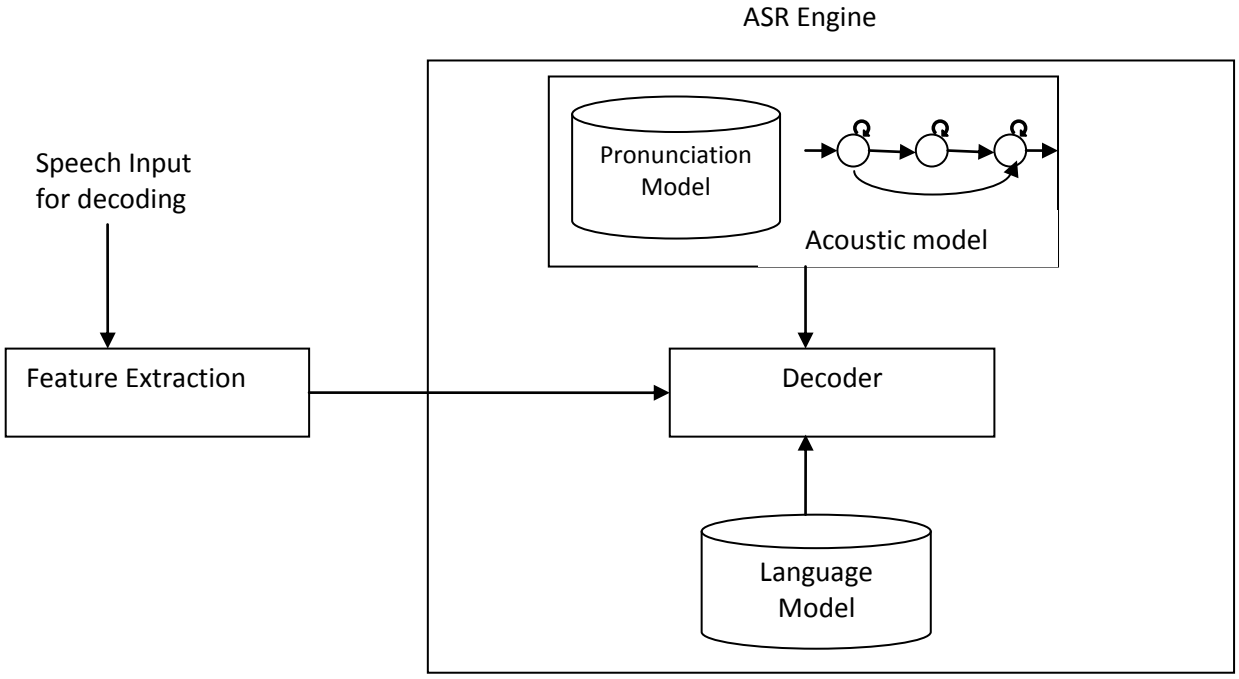


Figure 1.1 Architecture of ASR Systems [9]

Figure 1.1 indicates components of a typical speech recognition system and a brief description for each of the components given below:

Feature extraction: Generally, it is assumed that the speech signal and its features are a realization of some semantic or linguist message encoded as a sequence of linguistic symbols. The first step in speech recognition would be converting the analog speech waveform into an equivalent digital signal. This is done in two steps: sampling and quantization. A signal is sampled by measuring its amplitude at a particular time. And quantization is a process of representing real-valued amplitude measurements as integers. Once the quantized representation of the waveform is obtained, features that have a maximum of information

relevant for classification is extracted. The most commonly used feature for speech recognition is mel frequency cepstral coefficients (MFCC) which is based on a concept cepstrum[1].

The feature extraction module, therefore, converts the continuous speech signal into a sequence of feature vectors, each vector representing the information in a small time window of the signal. Feature vectors then would be labelled with a sequence of discrete symbols (usually phonemes in continuous speech recognition case) by the acoustic model.

Acoustic model: which is given by P(O|W) in a speech recognition task captures the correspondence between a short sequence of acoustic vectors and an elementary unit of speech (phoneme). In other words, it converts the signal represented by acoustic vectors to a sequence of hypothesized phonemes. The sequence of phonemes, during decoding, are converted into hypothesized word sequences using a pronunciation model. A phoneme is minimal unit of sound in a given language.

Pronunciation dictionary: is simply lists of words, with a pronunciation for each word expressed as a phone sequence. While some words have single pronunciation, some may have multiple pronunciations. Pronunciation model, therefore, generates hypothesized word sequences from the hypothesized sequence of acoustic signal. In other words, it specifies the finite set of words that may be output by the speech recognizer.

Language model (LM): LM**,** provided by P(W) in a speech recognition task, provides word-level structure for a language. The language model computes the prior probabilities of word sequences in a given language. In continuous speech recognition where the recognizer is expected to recognize sequence of words, the language model tells from the resulting list of phrases which sequence of words is more probable in a given language than others.

The decoder: Given acoustic model, LM and pronunciation model the task of the decoder would be to search a sequence of words that maximizes the product of the probabilities given by the acoustic and the language models. Alternatively, the decoder may provide N

[1] A cepstrum is the result of taking the Fourier transform (FT) of the logarithm of the estimated power (energy) spectrum of a signal.

hypotheses, represented either as an N-best list or a lattice, instead of a single most likely hypothesis [9].

1.2 Current status of Amharic LMs and Speech Recognition Systems

Amharic, which belongs to the Semitic language family, is the official working language of the federal government of Ethiopia. It has 19.9 million (comprise 26.9% of the country's population) native speakers according to the 2007 national census [5]. It is also a language that widely spoken by people across different regions of the country. Consequently, many NLP researches have been carried out on this language by different scholars; specifically by graduate and post graduate students. The brief of these research works related to speech recognition and language modeling are presented in this section.

Noticing speech recognition has a wide real world applications, different attempts have been made to build Amharic speech recognition system [4,7, 46, 47, 49, 51]. Though research on ASR for other languages began in 1960's, it is in 2001 Solomon[51] developed isolated, Consonant-Vowel (CV) syllable recognition system for Amharic. Kinfe [49] then attempted to find the best acoustic modeling unit by developing isolated Amharic SRS using phones, tri-phones and CV-syllables as modeling units. His experiment revealed that using phones as a modeling unit out smarted others. Later on, Zegeye[47] developed the first large vocabulary, speaker independent continuous speech recognition system. He trained phone-based and tri-phone models, eventually, his experiment demonstrated that tri-phones are the best modeling unit for continuous SRS. On the other hand, Martha [53] demonstrated the possibility of using Amharic ASR to command and control Microsoft Word application. Molalgne[52] compared HTK and MSSTATE SR modeling tools and revealed that HTK is the best in terms of both recognition speed and WRA. Unlike others, Hussien [46] explored the possibility of using ANN/HMM hybride approach to develop Amharic speaker independent continuous speech recognizers. Solomon [4] revisited the possibility of using syllable as modelling unit. However, the tri-phone model showed best performance at a marginal cost of memory and recognition time. Abraham[4], developed speaker dependent continuous speech recognizer and he demonstrated that introducing alternate pronunciations of words into the lexicon would benefit Amharic SRS.

To the best of the researcher's knowledge only [3] worked on Amharic language modeling. [3] built sub-word based LMs with a motivation that using sub-words as language modeling unit addresses data sparsity problem of Amharic that emanates from the morphological richness of the language. Using this approach, she has achieved a considerable out of vocabulary (OOV) and perplexity rates reduction and demonstrated that sub-word modeling approach is competent to word based n-gram modeling for small vocabulary task.

Despite these attempts, there is significant performance gap between ASR systems of developed languages and Amharic language as the word recognition accuracy of ASR in the technologically favoured languages is nearly 100% [6]. Therefore, attempt should be made on Amharic language to build ASR system that serve the intended purpose as other languages do.

1.3 Statement of the problem

There has been few attempts to build Speech Recognition System for Amharic. However, Amharic SRS has not reached its maximum possible performance when compared to technologically favored languages [4,6,7]. Among other things, lack of well prepared and adequate linguistic resources such as speech and text corpus, dialects (pronunciation variation), lack of quality LM and the morphological richness of the language itself contributed to the performance degradation. Therefore, much has to be done to every component (pronunciation, acoustic and LM components) of the system so as to make it usable for real world applications such as voice enabled environment control, search, dictation systems and so on.

As briefly discussed in section 1.2, the majority of the previous works mainly focused to address the various issues related to the acoustic model component. To that end, [4 and 7] attempted to address problems related to pronunciation variation, [7 and 47] attempted to improve performance of the recognizer using different acoustic modeling units. While all of other works are using HMM approach, [46] attempted a hybrid approach (ANN/HMM). However, in addition to an acoustic model, the LM component plays a massive role to enhance the performance of speech recognizers [9]. To this end, [7 and 47] applied bi-gram LM trained on small text corpus that is used to prepare the speech corpus. Nonetheless, bi-gram is poor in predicting the next word since its prediction is based on a single word history.

On the other hand, for the language models to make reliable prediction there shall be large size corpus to learn the parameters and robust smoothing algorithm needs to be applied [14]. Since Amharic is a morphologically complex language, the data sparseness problem of statistical language modeling becomes more serious. Martha[3] applied sub-word based modeling to address this issue. Using this approach, reduction in OOV and perplexity rate achieved. However, applying these models to word-based recognizer did not lead to WRA improvement. That is because, sub-word based models are poor in capturing word level dependencies in sentences. For example, a word might be segmented into 3 or more morphemes and if a tri-gram morpheme based LM is trained then the span of the n-grams might be limited to a single word. Building higher order n-gram might be proposed to tackle this problem, however, doing so increases the complexity of language models. Taking the above example, one need to build 6 gram language model to capture dependency between two words. An alternative to overcome this problem is to build sub-word based recognizer. This approach has its own drawback that since the recognizer recognizes sub-words, these sub-words need to be concatenated to form words and/or sentences. The generated (concatenated) words should be valid with respect to the morphological rules and semantics of Amharic. However, the language model itself does not guarantee that these rules will be satisfied. Therefore, illegal morpheme sequence might be concatenated [1]. Therefore, other approach that takes into account data sparsity problem of Amharic need to be studied. One promising direction is to cluster words that have similar syntactic and/or semantic functionality to same class which normally is called class-based language modeling and it proved to perform better for languages such as French, British English, German, Italian and Spanish [32].

The degraded performance of Amharic speech recognizer, among other things, should be due to the lack of high quality LM. The speech recognition equation (see equation 1.4) shows that the task is highly dependent on the quality of the LM being used in addition to the acoustic model component. Therefore, high quality language model in addition to the acoustic model has to be primed in order to have a better SRS. The rationale of this study is, therefore, to explore the best way of modeling the Amharic language using words as a language modeling unit. To this end, the research is going to answer the following questions.

- Do language models yield a performance improvement in Amharic speech recognition system?

- Which n-gram gives the best quality language model in the n-gram language model?
- Which smoothing techniques give the best Language model?
- Are Amharic class-based language models superior over word based models ?
- Does combining class-based models with word based models lead to performance improvement ?
- Do lower perplexity language models provide word recognition improvement in speech recognition task ?

1.4 Motivation

Ever since humans have been interacting with computers, it has been tried to make the communication between the two to take place in convenient way. This evolved from early days of punched card to the recent touch on screen and human-like mode of communication. Users would like computers to understand human language such as speech and handwriting. However, none of this can be achieved with a reasonable success unless the computer is having knowledge of human language [15 and 36]. And, language modeling is a tool to equip computer systems with knowledge of human language.

ASR for other technologically favored languages become fruitful since so many research have been done in language model and other components of it [15]. Despite of that only very limited effort exerted on Amharic language modeling. If speech recognition and other natural language processing tasks of a specific language wanted to be worthy, building language model is essential. Every language has its own set of distinguishing characteristics that makes it different from others. Approach working good for one language may not show up the same performance on other languages due to differences in the nature of the languages (like grammatical structures). It is to mean that there is no an approach that fit every language. Hence, the best approach for Amharic language modeling should be studied out in its way.

1.5 Objectives

The general and specific objectives of this research are described below.

1.5.1 General Objectives

The general objective of this research is to build statistical Amharic language model and assess its significance on the performance of a continuous Amharic speech recognition system.

1.5.2 Specific Objectives

The specific objectives of this research work are:

- Studying and selecting appropriate language modeling method for Amharic ASR.
- Preparing large corpus for Amharic Language to better model the language
- Building and integrating Amharic language model into the ASR system
- Compare performance of deferent language models
- Testing the performance improvement of the ASR system due to the language model

1.6 Scope and Limitations

The scope of this research project is limited to building a statistical language model and evaluating its quality using both intrinsic and extrinsic evaluation methods. Even though there existed range of natural language processing tasks [15] that could be used for extrinsic evaluation, in this research speech recognition task is considered for that purpose.

The data being collected and used for this research work is from News domain. Even though non-news texts such as dialog, interview, fiction and others are required to build general purpose LM, in this research only news corpus is considered. Nonetheless, in a news corpus contents of several domain such as health, sport, politics, agriculture and so on are common. Therefore, it enables to build LM with good coverage.

1.7 Methodology

This section describes the research methodology that will be used in order to achieve the stated objectives of this research work.

1.7.1 Literature review

Reviewing of literatures is done so as to reveal underlying theories/principles, methodologies, algorithms and tools which could be used in language modeling tasks. Review on Amharic language modeling researches will be explored to a greater depth to comprehend what have already being done and what is missing. Furthermore, it would be helpful to better understand the problem and come up with an elegant solution. Moreover, a comprehensive investigation and analysis on language modeling for speech recognition will be conducted.

1.7.2 Data collection

At the heart of building statistical language model there exists a large text corpus. In order to build a robust statistical language model, one ought to consider two aspects of the data being used to train the model [45]. The first one is quality which refers to acquiring suitable sample of domain specific data from which to train the models. The other is quantity that refers to the volume of the corpus used to build the model. Meaning that a corpus that would have good coverage as well as in-domain content is required.

The researcher had to determine the type of corpus that would fit the intended objective which is building a language model that could be used to improve quality of a given speech recognition system. To the best of the researcher's knowledge, there is no a publicly available readymade corpus for Amharic that could be used for language modeling task. Thus, the researcher identified the possible sources to acquire the news corpus with good coverage. As a result other researchers corpus used for Amharic language modeling related matter and Ethiopian News Agency data are considered.

1.7.3 Modeling Techniques

ASR modeling

The dominant technology used in ASR is called the Hidden Markov Model (HMM) [9]. This technology recognizes speech by estimating the likelihood of each phoneme at contiguous, small regions (frames) of the speech signal. It converts those phoneme sequences into hypothesized word sequences using a pronunciation model (vocabulary) where Each word in a vocabulary list is specified in terms of its component phonemes. Finally, it determines from the resulting list of phrases which one was intended, using a language model.

Language Modeling

Two of the most commonly known approaches to language modeling are statistical and rule-based (grammar-based) methods. Rule based approach works based on the grammatical rules that define words and rules that characterize how those words could be combined to form grammatical sentence in the language[2]. Meaning that sentences whose rule is not included in the model and ungrammatical constructs are going to be rejected.

However, it is very difficult and time consuming to define all the rules of a given language plus it will not be fair to discard ungrammatical constructs. Human beings do not always produce grammatical sentences. For example, speech recognition system used to provide subtitle for a television program should be able to deal with ungrammatical utterances[2]. Hence, the application of rule based approach is limited to specific area like command and control system where a sort of formal mode of communication is applied.

In this research work, therefore, the more robust language modeling technique which is statistical language modeling approach is employed. In a nutshell, statistical models are constructed from text corpus in a process called training. They are meant to provide an estimate P(w) for words in the language based on patterns and regularities seen in the corpus. There exists different approaches in statistical language modeling framework [16]. In this study, word n-gram and class-based language modeling methods are applied.

1.7.4 Tools

There are different language modeling tools for statistical language modeling approach. Stanford Research Institute Language Model (SIRLM) [38], a publicly available tool, is used for language modeling task. Hidden Markove Modeling Toolkit (HTK), which is widely used in ASR modeling, is used for the development of Amharic ASR. The selection of these tools is based on their popularity, appropriateness and researcher's convenience.

1.7.5 Evaluation

Language model evaluation is either intrinsic or extrinsic. In extrinsic evaluation, the model is evaluated in terms of the magnitude of the effect it brings on a specific application for which it was designed where as in intrinsic evaluation the model is assessed how good it is independent of any application [3].

In this research both the intrinsic and extrinsic methods are being used to assess the quality of the LMs built. For the purpose of extrinsic evaluation, the LMs are applied on Amharic ASR task. For intrinsic evaluation, the perplexity2 measure which is in built to most language modeling tools is used.

[2] A measure of on average how many different equally most probable words can follow any given word. Lower perplexities represent better language models.

1.8 Application of Results

For NLP tasks, the application of language modeling is usually presented in the context of the noisy channel model [10]. The noisy channel model (Figure 1) considers many NLP tasks in the form of decoding problems. The idea is, input I corrupt as it travels through the noisy channel, resulting in the observed output O which assumed to be different from what is given as input. This corrupted output needs to be corrected so as to deliver the initially intended message. The task of the decoder, therefore, is to find the most likely input Îgiven the corrupted version of the input, which is O. The decoder then proposes list of possible alternatives of the likely input Î and the language model which attempts to capture regularities of human language helps the decoder to figure out what can be the most likely input from the alternatives looking at the some previous contexts.

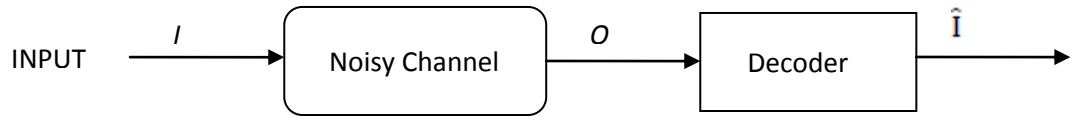


Figure 1. 2 The noisy channel model.

Initially language model came to live to enhance ASR systems as the input speech sounds are very confusable and many words sound extremely similar. However, it is now used by a variety of NLP tasks in addition to ASR systems[15]. Some of these are:

- optical character recognition (OCR),
- Handwriting recognition,
- statistical machine translation (SMT),
- automatic spelling correction (ASC) and other applications such as information retrieval [10].

1.9 Organization of the Thesis

The thesis is organized into six chapters including this chapter. This chapter offers background information on Amharic language modelling and speech recognition systems,

states the problem, describes the objective and methodology of the study. Chapter two provides the detail of statistical language modelling such as approaches to language modelling, smoothing algorithms, language model adaptation and evaluation techniques. In Chapter three, the preparation of Amharic text corpus (ATC) is presented. Chapter four shows the detail of Amharic n-gram and class-based language models construction. Results of both intrinsic and extrinsic evaluation of class-based and word based n-gram language models are discussed in Chapter Five. The final chapter, Chapter six, gives concluding remarks of the study and points out possible extension of the work.

CHAPTER TWO
LITRATURE REVIEW AND RELATED WORKS

2.1 Introduction

The need for a language model in large vocabulary speech recognizers has already been outlined, together with the relationship of the language model with other components of a contemporary speech recognition system. This chapter deals with the theoretical foundations of statistical language modeling. The first section describes the fundamental concepts of language modeling and discusses various language modeling approaches. The maximum likelihood (ML) probability estimation has a problem of assigning negligible probability for rare events in the training corpus. The second section describes the techniques used to overcome weakness of ML estimation and data sparseness problem. Perplexity which is the common LM evaluation metric is presented in the third section. Language models work fine when the training data is from the same domain of the NLP task for which the LM is built. However, whenever there is a problem of domain mismatch between the LMs training data and the task for which the LM is built, LM adaptation is carried out to complement the mismatch. The fourth sections describes the statistical LM adaptation technique. Finally, an exhaustive review of works done related to Amharic language modeling are presented.

2.2 Language modeling

Language modeling is the task of estimating the probability distribution of linguistic units such as words, sentences, queries, utterances, or even complete documents [13]. The probability distribution itself is referred to as a language model. Over the past couple of decades, with the increased availability of text data online, the quality of language models has increased vastly, and it has become possible to use language models in a wide variety of natural language processing (NLP) tasks including speech recognition, document classification, optical character recognition, and statistical machine translation[14]. In this section, a number of language modeling techniques are reviewed.

2.2.1 N-gram language modeling

The most popular technique for building language models is using N-grams. An n-gram is a sequence of n symbols and an n-gram language model (LM) is used to predict each symbol in the sequence given its n-1 predecessors. It is built on the assumption that the probability of a specific n-gram occurring in some unknown test text can be estimated from the frequency of its occurrence in some given training text [9]. Basically, it is about estimating the probability of a word w given history h or the probability of an entire word sequence W (W= w_1, w_2, w_3……w_k). The probability of the word sequence P (W) can be computed using the chain rule as follows

$$\begin{aligned} P(W) &= P(w_1, w_2, \ldots, w_k) \\ &= P(w_1)P(w_2|w_1)P(w_3|w_1w_2),\ldots,P(w_k|w_1 \ldots w_{k-1}) \\ &= \prod_{i=1}^{k} P(w_i \mid w_i, \ldots, w_{i-1}) \end{aligned} \tag{2.1}$$

Equation (2.1) shows that one can compute the joint probability of the entire sequence as a product of conditional probabilities. Chain rule does not work well as it tries to compute the probability of the i^{th} word looking the entire history (preceding word sequence) of that word, but any particular context (history) might have occurred only few times or might have never occurred at all [15]. Thus, N-gram makes use of the Markov assumptions which says that it is possible to predict the probability of some future units without looking into the entire history or too far in the past [15]. So, from the above equation (2.1), one can estimate P ($w_k|w_1$, w_2, $w_3 \ldots w_{k-1}$) as :

$$P(w_k|w_1, w_2, w_3 \ldots w_{k-1}) \approx P(w_k|w_{k-n+1,}\ldots,w_{k-1}) \tag{2.2}$$

for some $n \geq 1$. In practice, the largest n in wide use is n=3; this model (with n=3) is called trigram model. As the value of n gets larger, the model will suffer more from data sparsity[3] problem and becomes computationally complex as it demands more memory and computing time.

The probabilities of N-grams are usually estimated using a maximum likelihood estimation (MLE) approach. The MLE for the parameters of an N-gram model obtained by taking counts

[3] Any particular sequence of word that occurs in the test set may simply never have occurred in the training corpus.

from a training corpus, and normalizing them so that they lie between 0 and 1 [15] and that could be described in the following equation:

$$P\left(w_k | w_1, w_2, w_3 \ldots w_{k-1}\right) = \frac{C(w_{k-n+1},\ldots,w_k)}{C(w_{k-n+1},\ldots,w_{k-1})} \quad (2.3)$$

where C (.) is the count of a given word sequence in the training corpus.

In an n-gram model the combination of the finite vocabulary V and fixed length history limits the number of unique events to $|V|^n$, where V is vocabulary size. For any n>1 it is highly unlikely to see all word sequences from the training corpus, though, the corpus is large in size. On the other hand, MLE primarily depends on a specific set of training data and assigns zero probability for those not found in the training corpus even though they are meaning full constructs of the given language. But it is not feasible, because an event has never been observed in training data does not mean it cannot occur in test data. Furthermore, MLE produces poor estimates for less frequently observed events [15]. To this end, some of the common techniques used to overcome the shortcomings of MLE are covered in the forthcoming section (See section 2.3) of this chapter.

2.2.2 Class Based Language Modeling

Class-based language modeling is a variant of normal word-based model that attempts to make use of the similarities between words. One of the problems that N-gram language models suffer from is the training data sparsity to reliably estimate all conditional probabilities $P(w_k|w_{1..k-1})$ [19]. Class-based n-gram models are intended to help overcome this data sparsity problem by grouping words into equivalence classes rather than treating them as distinct words and thus reducing the number of parameters of the model [8]. Grouping of words to classes could be done either using linguistic knowledge such as based on POS and semantics of words or using statistical methods [41].

Generally, grouping of related words is implemented by mapping a set of words to a word class c ϵ C by using a classification function C(w) = c. If any class contains more than one word then this mapping will result in less distinct word classes than there are words, |C| <|V|,

thus it shall reduce the number of separate contexts, in other words, reduces the storage and training data requirements [9].

By partitioning all words in the vocabulary |V| into |C| sets of words, a typical class-based n-gram model approximates $P(w_k|w_{1\ldots}w_{k-1})$ with the two following component probabilities [8]:

$$P(w_k|w_1 \ldots w_{k-1}) = P\big(w_k\big|c(w_k)\big) * P\big(c(w_k)\big|c(w_1 \ldots w_{k-1})\big) \qquad (2.4)$$

where $P\big(w_k\big|c(w_k)\big)$ is a class map with associated word given class probability and $P\big(c(w_k)\big|c(w_1 \ldots w_{k-1})\big)$ is an n-gram which models class sequences.

Class based language model has got some advantages over a word n-gram model. At the expense of a slightly greater perplexity, the 3-gram model with word classes requires only about one-third as much storage as the 3-gram language model in which each word is treated as unique individual[8]. Moreover, it also tackle data sparseness problem by defining clusters for similar words in a corpus. So that the model can make more confident assumptions about infrequent words based on other more frequent words in the same class. For the same reason, it is able to make generalized assumptions about words used in contexts which are not explicitly encountered in the training text [9].

Class-based LMs have often been shown to improve the performance of speech recognition systems when combined with word-based language models [19]. However, this model has a difficulty of distinguishing different histories as classes are used instead of actual words when computing history of a word. This problem could be counteracted by using higher order n-grams [41].

2.2.3 Mixture Language model

The mixture language model was originally motivated by an observation that the likelihood of different words could be very different in different sub-domains [21]. Moreover, the conventional n-gram language models have a limitation of taking advantage of long distance dependencies in natural language as their estimate are merely dependent on very few local words. A human listener, in contrast, condition his expectation of what will be said on a long-ranging history of what has been said.

[23] proposed a mixture model that capture properties of text (words) such as text in different domains (politics or health) , style (news text or database queries) and vocabulary (general or technical). N-grams of these specific models are then linearly interpolated using interpolation parameters that reflect the type of the preceding part of the text. A mixture LM, denoted by M, is constructed as the weighted sum of J component LMs derived from J partitioned corpus [22]. The adaptive interpolation parameters are chosen to be the maximum-likelihood estimates obtained from the text passage preceding the word under consideration.

Each of the component M language models are trained on different text category such as education, politics news text, etc. For simplicity of notation, a bigram model ($P_i(w_k|w_{k-1})$) which is the conditional probability of observing word w_k after word w_{k-1} is considered. Each of the M specific models contributes to the interpolated model via [23]:

$$\mathbf{P(w_k|w_{k-1}) = \sum_{i=1}^{J} \lambda_i P_i(w_k|w_{k-1})} \qquad (2.5)$$

Where λ_i interpreted as the probability that it is language model M which produces the current word. Particularly, $0 \leq \lambda_i \leq 1$ and $\sum_i \lambda_i = 1$

Two of the important issues in building mixture models are classifying the corpus in to the corresponding topics and estimating interpolation factor for the M components. With regard to corpus clustering, it can be done either manually if the corpus is especially labeled corpus or automatically using classification algorithms such as Agglomerative clustering [24].

2.2.4 Factored Language Models (FLM)

Factored language model is an extension of a conventional n-gram language models. It was first introduced in [25] for incorporating different morphological information in Arabic language modeling. The assumption is that incorporating other sources of knowledge about the language would improve the quality of the language model. For example, incorporating morphological class information, for instance, might improve the quality of a language model as knowing the POS of a word can tell us what words are likely to occur in its neighborhood [15]

FLM consider words as a bundle of features commonly called factors, one of which may be the actual surface form of the word. As described by [25] a word w is a vector of p (parallel) factors such that:

$$\mathbf{W_k} \equiv \{\mathbf{w_k^1}, \mathbf{w_k^2}, \dots, \mathbf{w_k^p}\} \qquad (2.6)$$

Where f^i, is factor that the word can decomposed to.

Apart from the word itself, factors of a given word can be anything, including morphological classes, stems, roots, and any other linguistic features that might correspond to or decompose a word; so that the probabilistic language model is over both words and its decompositional factors. Once a set of factors for words has been selected, the goal of an FLM is to produce a probability model over a sentence of K words, each with p factors, namely:

$$P(w_1, w_2, \dots, w_K) = \boldsymbol{P}(\mathbf{f_{1:K}^{1:p}}) \qquad (2.7)$$

Being an extension of n-gram, FLM use the same Markov independence assumption. Given the history of n-1 words, the likelihood of the next word W_k, composed of factors$f_k^{1:p}$, can similarly be written as:

$$P(w_1, w_2, \dots, w_k) \equiv P(f_k^{1:p} \mid f_{k-n+1}^{1:p}, \dots, f_{k-1}^{1:p}) \quad (2.8)$$

There are two main design decisions one must come across in building an FLM [26]:

- First, choosing appropriate set of factors with which to represent words. This task can be done either using data-driven techniques or applying linguistic knowledge
- Second, finding the best statistical model over these factors.

Factored language model provides sort of flexibility. For example, in n-gram models probability of a given word is computed from a linearly ordered word sequences, however, factored language model does not impose a particular linear ordering on the factors in a word bundle. Furthermore, not all available features have to be used in a factored language model. The relevant history for a word can be defined to be any subset of the available n factors [27].

2.2.5 Skip language model

Skip language model is an extension of the n-gram language model in which the probability of a word is estimated, unlike n-gram, from a different context than the immediate n-1 previous words. Skip n-gram allows tokens to be skipped in addition to allowing adjacent sequences of words. For example the sentence "አበበ በእረፍቱ ቀን ትምህርትቤት ሄደ" (Abebe went to school on his vacation time) has three word level tri-grams:

"አበበ በእረፍቱ ቀን"
"በእረፍቱ ቀን ትምህርትቤት"
"ቀን ትምህርትቤት ሄደ "

However, one might argue that an equally important tri-gram that could be implied from the sentence but not normally captured by the tri-gram are: "በእረፍቱ ትምህርትቤት ሄደ" and "አበበ በእረፍቱ ሄደ". Even though, skip n-gram uses the same number of predecessor words as the normal n-gram model, it utilizes long range dependencies within sentences so that those likely tri-grams could be generated. For example, words "ቀን" and "ቀን ትምህርትቤት" be skipped, enables these trigrams to be formed. This approach is useful to train language models from limited corpus as it would help to withstand data sparsity by extracting more events than regular n-grams [30].

2.2.6 Morphology based language model

The classical approach, very often, in language modeling is to use the whole word as a modeling unit. And, the probability of a word depends only on the previous 2 or three words. Despite of this, the task of statistical language modeling is challenging as many word contexts are observed infrequently or not at all [35]. The problem gets even worse for morphologically rich languages like Arabic, Turkish etc… [37]. As these languages can generate large number of word forms using productive word formation processes (in-flection, derivation, compounding etc)[4] and it is very difficult to include all these word forms into a dictionary. Consequently, this would result in high out-of-vocabulary (OOV) rates and high perplexity language models. Morphology based language model, therefore, is intended to overcome these problems by using sub-words as a modeling unit.

[4] These are morphological rules that govern how word-forms could be generated from morphs

Morphology is a field of study in linguistics and it deals with how words are built up from smaller meaning-bearing units, morphemes[5] [15]**.** Morphology based language modeling (MBLM) use sub-word as a modeling unit instead of the whole word (word-form). These models, therefore, are based on the decomposition of word forms into smaller morphological components. Word forms could be decomposed in to stems, stem plus word endings, root forms, syllable forms and some combination of these. Two common approaches to decompose words into sub-words are known to be knowledge-based and data-driven methods.

Using morphology based language modeling a significant reduction in perplexity result and OOV rate is reported [3, 37, 50]. the reduction in perplexity and OOV rate, however, may not always lead to improvement in word recognition accuracy [50].

2.3 Language model smoothing methods

As shown by equation 2.3, the primary problem with maximum likelihood estimation (MLE) is assigning a zero probability for unseen events in the training corpus. Consequently, word sequence with zero probability part ($P(w_k|\text{history})=0$) will end up in the zero probability for p(W). This commit sever mistakes in statistical analysis. Thus, the solution for such problem is called smoothing which assign non-zero probably to all events whether observed in the training data or not. Smoothing is a mechanism used for adjusting the observed counts and these counts are replaced by some other representative counts so as to build robust language models [17]. Not only do smoothing methods prevent zero probabilities, but it also attempt to improve accuracy of the model as a whole. Especially, when the training corpus is not large enough, smoothing has the potential to significantly improve estimation [16]. Some of the smoothing techniques are discussed in the following subsections considering a bi-gram case.

2.3.1 Add 1 smoothing

This is also called Laplace smoothing. The idea here is add one to each occurrence of the n-grams. In other words, Laplace smoothing algorithm pretends that each n-gram occurs once more than it actually does in the training data set [16]. For example the following equation shows each bigram occurs more than once than it actually does:

[5] Morpheme is a minimal unit of a language that bears meaning or has some grammatical function

$$P(w_i|w_{i-1}) = \frac{1+C(w_{k-1}w_k)}{|V|+C(w_{k-1})} \qquad (2.9)$$

Where |V| is the vocabulary size of the training corpus and C(.) is the count of n-grams. |V| is a normalization factor added because 1 is added to every word count that followed w_{i-1}.

The problem with this algorithm is that it gives too much probability to unseen N-grams [17, 14]. At the cost of enormously reducing the probability estimates of more frequent events. Thus, it is not widely used for practical applications.

2.3.2 Add λ smoothing

Add λ smoothing (also called Lidstone's law) came in place to overcome the problem of assigning much probability to unseen events by Laplace smoothing. The idea here is to add a fraction of 1 which is $0<\lambda<1$ rather than 1 to the counts so that reduce the probability assigned to unseen events. Accordingly, the bi-gram probability is computed as:

$$P(w_i|w_{i-1}) = \frac{\lambda + C(w_{k-1}w_k)}{|V|\lambda + C(w_{k-1})} \qquad (2.10)$$

This algorithms seems to avoid the drawback of Add 1 smoothing. However, it still needs an efficient way to find appropriate value for λ dynamically. On top of this, discounting using Lidstone's law always gives probability estimates linear in the MLE frequency and this is not a good match to the empirical distribution at low frequencies [14].

2.3.3 Good-Turing Smoothing

Good-Turing is discounting algorithm which is based on counting of N-grams seen once and using that count to estimate the count of other N-grams that never been seen at all. The basic insight of Good-Turing, therefore, is to re-estimate the amount of probability mass to assign to N-grams with zero counts by looking at the number of N-grams that occurred one time [15]. Thus, for any N-gram that occurs r times, an adjusted count r* is computed as:

$$r^* = (r+1)\frac{n_{r+1}}{n_r} \qquad (2.11)$$

and where n_r is the number of n-grams that exactly appears r times in the training corpus. The Good-Turing estimate replaces an MLE count r with a corrected count r∗. To convert this count to probability, normalization is required: for N-gram α with r counts probability is computed as:

$$P_{GT}(\alpha) = \frac{r*}{N} \qquad (2.12)$$

where N is the total (actual) counts of the n-grams in the distribution.

Even if smoothing algorithms (like Witten-Bell discounting and Kneyser-Ney and katz smoothing) are based on Good Turing, practically, it is not used for N-gram smoothing by itself, because it does not include the combination of higher order model with lower order models necessary for good performance [16]. Moreover, as can be seen from equation 2.11 above Good Turing method cannot be used when n_r is 0 (that is count of count is 0) which is common for high r. Therefore, usually a threshold is set for r, n-grams above the threshold will not be smoothed.

2.3.4 Jelinek-Mercer smoothing (Interpolation)

The basic intuition of this method is that combining higher order N-grams with lower order N-grams that generally suffer less from data sparseness problem yields in a better model. In any case where there are multiple probability estimates, a linear combination of them can be taken, provided that the contribution of each is weighted so that the result is another probability function [14]. For example, if count(ነዉ | አገር)= 0 and Count(ነው | አገር) =0, then using MLE P(ነዉ | አገር)=P(ነው | አገር) =0. However, this seems to be wrong as one expects P(ነዉ | አገር)>P(ነው | አገር) since ነዉ is much more common than ነው in Amharic language. Interpolating bigram and unigram models would give reasonable estimate for such cases.

Thus the bi-gram probability $P(w_k|w_{k-1})$ is estimated by:

$$P(w_k|w_{k-1}) = \lambda_1 P(w_k|w_{k-1}) + \lambda_2 P(w_k) \qquad (2.13)$$

where the sum of λ must be 1. The λ values are set using held-out[6] data. And generally one wants to find the combination of weights that works best.

[6] Held-out data is an additional training corpus that we use not to set the N-gram counts, but to set other parameters.

2.3.5 Back-off

Back-off use N-gram hierarchy in a way that if there exists non-zero counts for higher order N-grams then that count is solely used [14]. But when N-grams cannot be estimated reliably because their count is too small, a more general distribution, normally based on N-1 gram is used instead. In this method, generally, the longest history that is felt to be reliable is used. One of such algorithms is Katz smoothing.

Katz smoothing applies Good-Turing estimates to the problem of back-off language models. It uses a form of discounting in which the amount of discounting is proportional to that predicted by the Good-Turing estimate. Then, count mass subtracted from nonzero counts is redistributed among the zero count n-grams according to next lower-order distribution (n-1 grams). Thus, It uses discounting to know how much total probability mass to set aside for all unseen events, and back-off to distribute this probability. Accordingly, bigram estimate with r count computed as:

$$P_{Katz}(w_k|w_{k-1}) = \begin{cases} \dfrac{C(w_{k-1}w_k)}{C(w_{k-1})} \; if \; r > i \\ d_r \dfrac{C(w_{k-1}w_k)}{C(w_{k-1})} \; if \; i \geq r > 0 \\ \alpha(w_{k-1})P(w_k) \qquad r = 0 \end{cases} \qquad (2.14)$$

Where i is a count threshold (large counts taken to be reliable), d_r is discounting ratio which is proportional to Good-Turing discount. Specifically, if Good-Turing estimates count C as C^*, then d_r is approximated by $\frac{C^*}{C}$ where C is the actual count of the n-gram[7]. Then the back-off weight, $\alpha(w_{k-1})$, is computed as:

$$\alpha(w_{k-1}) = \frac{1 - \sum_{w_k:r>0} P_{Katz}(w_{k-1}w_k)}{1 - \sum_{w_k:r>0} P_{Katz}(w_k)} \qquad (2.15)$$

2.3.6 Whitten-Bell smoothing

Whitten-Bell smoothing is considered to be an instance of Jelinek-Mercer smoothing. Particularly, the n^{th}-order smoothed model is defined recursively as a linear interpolation

[7] The full derivation of discounting ratio d_r is given by [15]

between the n[th]-order maximum likelihood model and the (n-1)th-order smoothed model. The bigram model is computed by:

$$P_{WB}(w_k|w_{k-1}) = \lambda_{w_{k-1}} P_{ML}(w_k|w_{k-1}) + \left(1 - \lambda_{w_{k-1}}\right) P_{WB}(w_k) \qquad (2.16)$$

The parameter $\lambda_{w_{i-1}}$ is computed on the basis of the number of unique words that follow the history w_{i-1}, which is normally the number observed bi-gram types.

$$1 - \lambda_{w_{k-1}} = \frac{|\{w_k : C(w_{k-1}w_k) > 0\}|}{|\{w_k : C(w_{k-1}w_k) > 0\}| + N} \qquad (2.17)$$

Where N is the total number of tokens that follow the history (w_{k-1}).

2.3.7 Absolute discounting

This method involves the interpolation of higher and lower order models and the higher order maximum likelihood distribution is created by deducting a fixed discount d ≤ 1 from each non zero count [16]. The intuition is that since higher counts have good estimates, small discount d will not affect them much. Rather it will mainly modify the smaller counts, for which the estimate is unreliable[15]. Absolute discounting equation as applied on bigrams:

$$P_{Absolute}\ (w_n|w_{n-1}) = \begin{cases} \dfrac{C(w_{k-1}w_k) - d}{w_{k-1}}, & if\ C(w_{k-1}w_k) > 0 \\ \alpha(w_{k-1}) P_{Absolute}\ (w_k), & if\ C(w_{k-1}w_k) = 0 \end{cases} \qquad (2.18)$$

Where C(.) is the count of the n-gram, D is the discounting value which is commonly 1<D<0 and α is the back-off weight.

$$\alpha(\mathrm{w_{k-1}}) = \frac{1 - \sum_{\mathrm{w_k}} \mathrm{P_{Absolute}}\ (w_{k-1}w_k)}{1 - \sum_{\mathrm{w_k}} \mathrm{P_{Absolute}}\ (\mathrm{w_k})} \qquad (2.19)$$

Absolute discounting is a much better method of computing a revised count c∗ than the Good-Turing discount formula given by Equation (2.7), based on frequencies-of-frequencies [15].

2.3.8 Kneser-Ney Smoothing

Kneser-Ney Smoothing is an extension of Absolute Discounting method described in Section 2.3.7 above and it uses a different approach to handle the back-off distribution. In this method, the estimation of $p(w_k)$ is based on the number of different contexts word w_k has appeared in. The intuition, therefore, is words that have appeared in more contexts are more likely to appear in some new contexts as well. For clarity, let us see an example given by [15]. Consider the job of predicting the next word in the following sentence, assuming we are backing off to a unigram model:

I can't see without my reading _____

The word *glass* seems much more likely to follow here than the word *Francisco*. But *Francisco* is in fact more common, so a unigram model will prefer *glass* as a next word. Although *Francisco* is frequent, it is only frequent after the word *san*, i.e. in the phrase *san Francisco*. On the other hand, the word *glass* has a wider distribution (appears following many other words). Thus instead of backing off to the unigram MLE count (the number of times the word w has seen), we want a heuristic that more accurately estimates the number of times we might expect to see word w in a new unseen context. Thus, assuming a proper coefficient α on the back-off to make everything sum to 1, Kneser-Ney bi-gram back-off probability is computed as:

$$P_{KN}(w_k|w_{k-1}) = \begin{cases} \dfrac{C(w_{k-1}w_k) - D}{C(w_{k-1})}, & if\ C(w_{k-1}w_k) > 0 \\ \propto (w_k) \dfrac{|\{w_{k-1}: C(w_{k-1}w_k) > 0\}|}{\sum_{w_k} |\{w_{k-1}: C(w_{k-1}w_k) > 0\}|}, & Otherwise \end{cases} \quad (2.20)$$

2.3.9 Modified Kneser-Ney

This smoothing algorithm is a variant of Kneser-Ney algorithm described in section 2.7. Unlike Kneser-Ney that uses a single discount D for all nonzero counts, it has three different parameters, D_1, D_2, and D_{3+}, that are applied to n-grams with one, two and three or more counts, respectively [16]. The motivation behind this modification was that the ideal average discount for n-grams with one or two counts is substantially different from the ideal average discount for n-grams with higher counts. It is reported that this algorithm performed better

than the regular Kneser-Ney smoothing [16,20]. The probability computation for the bigram language model is given as follows:

$$P_{MKN}(w_k|w_{k-1}) = \frac{C(w_{k-1}w_k) - D(C(w_{k-1}w_k))}{C(w_{k-1})} + \beta(w_{k-1}w_k)P_{MKN}(w_k) \qquad (2.21)$$

Where D(c):

$$D(c) = \begin{cases} 0, & if\ c = 0 \\ D_1, & if\ c = 1 \\ D_2, & if\ c = 2 \\ D_{3+}, & if\ c \geq 3 \end{cases} \qquad (2.22)$$

And the value of D_1, D_2 and D_{3+} is determined as:

$$\begin{aligned} \mathrm{Y} &= \frac{n_1}{n_1 + 2n_2} \\ \mathrm{D1} &= 1 - 2Y\frac{n_2}{n_1} \\ \mathrm{D2} &= 2 - 3Y\frac{n_3}{n_2} \\ \mathrm{D3} &= 3 - 4Y\frac{n_4}{n_3} \end{aligned} \qquad (2.23)$$

where the n_i's have the same meaning as in absolute discounting and Kneser-Ney smoothing. If one of the n_i's (count of counts) is zero, then it is not possible to use this smoothing technique. On the other hand, $\beta(w_{k-1}\ w_k)$ is computed as follows so as to make the distribution to sum to 1.

$$\beta(w_{k-1}w_k) = \frac{\sum_{j=1}^{3+} D_j \left|\{w_{k-1}: C_j(w_{k-1}w_k) > 0\}\right|}{\sum_{w_k} \left|\{w_{k-1}: C(w_{k-1}w_k) > 0\}\right|} \qquad (2.24)$$

2.4 Language Model Evaluation

How can one knows the language model is a good one? In a very common sense, a good language model ought to prefer good sentences than bad once. In other words, a good language model is the one that assign higher probability to "real" or "frequently observed

sentences" than "ungrammatical" and "rarely observed" sentences. For example, a good Amharic language model shall assign higher probability for a sentence (አበበ በሶ በላ) /Abebe ate beso/ than (አበበ በሶ አኘከ) /Abebe chewed beso/.

After training the parameters of the LM using training data, its performance is then tested using some other data that is not used for training the model (test set). Then, an evaluation metric tells us how well the model does on the test set. There existed two types of evaluation metrics for evaluating goodness of the model and these are extrinsic and intrinsic evaluation metrics.

The first one, extrinsic approach, claim that the ultimate demonstration of success is showing improved performance in the application task, be that spelling correction, machine translation, information retrieval etc which the model is built for [14]. So as to compare two language models, each model is put in a task like speech recognizer and make comparative analysis on the performance difference and conclude the difference is due to the integrated LM.

Extrinsic evaluation method of LMs is expensive [15]. On one hand, an external application such as speech recognizer is required just to evaluate a LM performance. Moreover, these tasks require too much time to run and may take hours and even days depending on the complexity of the task. Therefore, a metric that would allow evaluation of a LM with less cost and independent of any external application is important. An intrinsic evaluation is one which measures the quality of the language model independent of any application [15]. A commonly used intrinsic LM evaluation method is perplexity. The approach is described in detail in the following sub section.

2.4.1 Perplexity

The intuition of perplexity is that the probability that a language model assigns to some test text, which did not form part of the data used to train the model, gives an indication of how well that model predicts or models unseen data. For example, given two LMs, the better model is the one that predicts the details of the test data better. That prediction is measured by the probability the model assigns to the test data; the one that assigns higher probability is the better model.

More formally, the perplexity (sometimes called PP for short) of a language model on a test set is a function of the probability that the language model assigns to that test set. For a test set W = $w_1w_2 \ldots w_k$, the perplexity is the probability of the test set, normalized by the number of words:

$$PP(W) = P(w_1w_2, \ldots, w_k)^{-\frac{1}{N}}$$

$$PP(W) = \sqrt[N]{\frac{1}{P(w_1w_2, \ldots, w_k)}} \qquad (2.25)$$

Applying chain rule to expand the probability of W, we find:

$$PP(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_k|w_1, \ldots, w_{k-1})}} \qquad (2.26)$$

Suppose if bi-gram LM is considered, then the perplexity of W would be computed as:

$$PP(W) \cong \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_k|w_{k-1})}} \qquad (2.27)$$

As shown on the above equation (Equation 2.27), perplexity is an inverse of the average probability assigned to each word in the test set by the language model. That means, if the perplexity is low meaning that the prediction probability is high and it is a better language model. Nonetheless, low perplexity only indicates the goodness of the language model on the test set and it doesn't necessarily show performance improvement over the applications it is built for. For example, improvement in perplexity of the langue model my not necessarily improve the word error rate of the speech recognizer. However, it often correlates with such improvements. Perplexity also considered to be the average branching factor of a language model. The branching factor is the number of possible next words that can follow any other word [15].

Using this approach, given two language models built using same vocabulary and training set, their relative performance is evaluated based on their perplexity results on a given test set data. Whichever shows less perplexity on the test data is considered to be the better model.

2.5 Language Model Adaptation

Language model adaptation refers to the process of exploiting specific, but limited, knowledge about the recognition task to compensate for any mismatch between training and recognition[42]. Meaning that the performance of the speech recognizer affected due to differences in lexical, syntactic, or semantic characteristics of the corpus used in training the LMs and recognition tasks. It is an inevitable problem and LM adaptation is one of the remedies. Figure 2.1 shows the general adaptation framework for statistical language models.

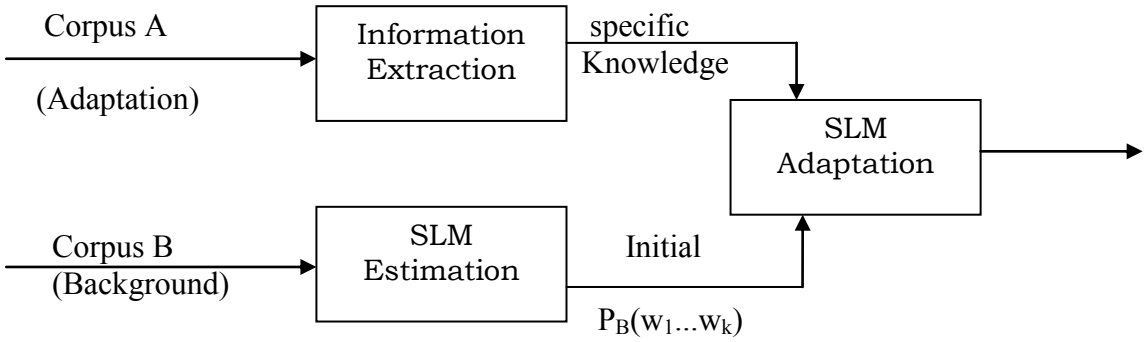


Figure 2.1. General framework for SLM adaptation [42]

The adapted language model make use of both the well-trained, but possibly mismatched, background SLM ($P_B(w_1...w_k)$) and the adaptation data (Corpus A), possibly a small size corpus, but has got specific information relevant to the current task. If the adaptation data is not readily available, either it can be prepared from scratch or can be generated artificially from baseline systems [43]. For example, text corpus used to prepare speech corpus can be used as an adaptation data.. Another issue is how to use corpus A to obtain task relevant information. [42] reviewed different approaches of doing LM adaptation and one is model interpolation technique proposed by [44].

2.6 Related Works

This section presents previous works done on language modeling for Amharic and other languages. Unlike other languages, there are only few attempts on Amharic language modeling and the researcher tried to make exhaustive review of these attempts. On the other

hand, only works related to class-based language modeling are given emphasis for other languages.

[41] have carried out an empirical study of clustering techniques for Asian languages (Chinese and Japanese). The researchers experimented with class-based trigram models on a Japanese newspaper corpus and on a Chinese heterogeneous corpus using both statistical and knowledge based approaches for words clustering. The Chinese corpus consisted of 7 million words and the Japanese corpus consisted of ten million words were used to carry out the experiments. Accordingly, they have developed trigram models using both automatically derived class and human defined classes. Their experimental results revealed that both of the class based LMs (trained using automatically derived class and using human defined classes) showed relatively higher perplexity than the word based models. However, both provided considerable perplexity improvement when combined with word n-gram LMs. Furthermore, the human defined classes performed worse than automatically derived class-models. That is because human defined classes are fewer in number and that makes the model more generic. They have also explored different ways of using clusters to reduce the size of the LM. To this end, they have achieved around up to more than 40% LM size reduction.

[32] studied the effectiveness of class-based models when linearly combined with word-based models for five different languages namely French, British English, German, Italian and Spanish. While statistical clustering is applied for all the languages, POS based clustering is tested only for English, French and Italian languages. For each languages, they have trained language models for varying corpus size and reported significant improvement in perplexity and word error rate reduction for all the five languages irrespective of the size of the training corpus. Moreover, grammatically derived class LMs showed relatively smaller improvement when compared with statistically derived class LMs. Eventually, they make a conclusive remark that even though class-based language modeling is introduced as a solution to train robust LMs from sparse data, it provides both WRA and perplexity improvement when combined with word n-gram LMs even on larger text corpus.

2.6.1 Previous works on Amharic language modeling

Zegaye [47] attempted to build bi-gram language model as part of his speaker dependent, continuous Amharic speech recognizer. He used text corpus which consisted of 800 sentences that were used to record the 800 speech utterances. The Katz back-off smoothing is applied.

Finally, the LM were evaluated using different test sets that are automatically generated by the HTK tool and scored an average of 52.5 perplexity result. The minimal perplexity score was due to a very small corpus used to train and evaluate the LM.

Solomon [7] trained bigram language models using the HTK statistical language model development tools as component for his LVC Amharic speech recognizer. The model is built from text corpus consisting of less than 75,000 sentences (900,000 tokens), including the 10, 875 sentences recorded for training speech data and it showed perplexity value of 167.889 and 165.589 on development and evaluation sets for 20,000 word vocabulary size.

The test set used for evaluation is not a separate text rather it is an automatically generated text using HTK tool from the text used for training. His lower perplexity result, therefore, might be associated with the type of the artificially generated test text.

Martha [3] experimented with various sub word based language models including morph based and factored language models. She built morpheme based language models using both statistical derived and linguistic morphs. Morpheser [28] is used to derive statistical morphs from text corpus composed of 77,844 sentences (868929 tokens or 108523 types). On the other hand, manually segmented corpus is prepared to build linguistic morpheme based language models. This corpus has 1,141,434 tokens or 11,154 distinct morphs. Using SRILM language modeling tool, different statistical morpheme based n-gram language models are built and perplexity values of 18.39 and 64.22 are reported to be the best models for linguistic and language model respectively.

Factored language models, apart from morpheme based models, are also built using the manually segmented corpus. Although words can have more than one prefix or suffix, each word considered as having only zero or one prefix and/or suffix by concatenating a sequence of affixes into a single unit. Language models with two parents, which the estimation of the probability of each word depends on the previous word/s and its/their POS, and a language model with four parents that consider factors such as the word itself, POS, prefix, root, pattern and suffix as histories and that uses a fixed back-off is built using SRILM. And, perplexity value of 17.03 and 115.89 are reported for four factors and two factors language models.

All the sub word based language models built are then tested on Amharic speech recognizer. The baseline speech recognizer which has a word based bigram language model along with the lexical and acoustic models have got 91.67% word recognition accuracy. After application of the above sub-word language models to the baseline, they obtained 0.18% and 0.25% absolute WRA gain for statistical and linguistic morpheme based models and 1.25%, 1.08% for two and four factor factored language models respectively.

2.6.2 Summary

As can be noted from the review above [see section 2.6.1], only very few works are there in the construction of language model for Amharic language, though, it is a very essential component of many NLP applications and research works. Statistical language model, by its nature, demands large volume of data, however, most of the previously done language models for Amharic language were built from small size corpora. In spite of this, most of the researchers did not go beyond bi-gram language models. Furthermore, only few language modeling approaches are explored yet. Sub word based language models seen as a solution for morphologically rich languages like Amharic. In fact, these models yield in a very minimal perplexity values. Despite, the minimal perplexity, when the models are applied on real applications (speech recognizer) only minimal improvement have been observed. This is attributed to known challenge of sub word based models namely loss of dependency between words. These models better predict the co-occurrence of the prefix/suffix and the stem words instead of word sequences (word co-occurrence).

Thus, other approach that handle the data sparsity problem of Amharic need to be studied. Furthermore, it is indispensable to prepare or build larger text corpus so as to build a more robust language model.

CHAPTER THREE
AMHARIC TEXT CORPUS (ATC)

3.1 Introduction

Amharic, the primary language of Ethiopia, is the second largest Semetic language next to Arabic in terms of the number of speakers. According to [5], it is estimated that more than 20 million persons speak it as first language and used as a second language by large number people across different regions of the country. Despite the large number of speakers, there have been few efforts to create language processing tools and resources such as speech and text corpus for Amharic. This chapter describes the collection, normalization and statistics of ATC.

A corpus is a collection of transcribed speech or written text which have been selected and brought together as a source of data for research studies. Statistical NLP tasks are based on counting of things or events in a training corpus. To get the statistical model of these events, the first task would be to have corpus which consisted of the events needed. Corpus could be domain specific in which case all its content is about specific subject or general purpose in which case the corpus have contained of content from different subjects or areas. Corpus can also be distinguished as tagged or untagged. Tagged corpus provides additional information by annotating the plain text at different levels such as at morphological level in terms of prefix, stem and postfix, at lexical level with Part-Of-Speech (POS tagging and so on. For this study untagged text corpus is built.

There are two phases in corpus preparation, design and implementation [48]. In the design phase, details such as the quality and quantity aspects of the corpus are specified and in the implementation phase, series of activities are executed to accomplish the design. The quality of the corpus refers to how fit the text is for the particular purpose it is being built. On the other hand, quantity refers to the corpus size. Even though the quality aspect can partially be evaluated through manual inspection, there is no an easy way to measure if the corpus size is sufficient or not for the task at hand.

Even though there existed well organized and freely available corpus for other languages like English, there is no a publicly available text corpus that would be used for language modeling researches for Amharic. To this end, corpus used by other researchers[3] were considered.

However, there are two limitations with regard to using that corpus: it is smaller in size and the corpus content is not appropriate to the task being undertaken according to our selection criterion described in the next section [see section 3.2]. [45] experimented on the effect of corpus size and homogeneity on the quality of LM and they have shown that a language model trained on only 2 million words in-domain corpus can perform better than one trained on a generic corpus of over 100 times that size. Therefore, the researcher had to prepare a corpus by himself that considers both quality and quantity aspects.

3.2 Corpus building

At the heart of building statistical language model there exists a large text corpus. Building such large corpus is a project by itself, however, in cases where the required corpus is not available, it is inevitable to prepare one from scratch. The corpus construction process involves two important steps: the planning (design stage) and the implementation (creation stage) as described below:

3.2.1. Design stage

The process of collecting a corpus to its broadest and widest range should be based on its purpose. Our goal was to build corpus that is to be used to study the possible ways of building Amharic language models so as to improve speech recognition task. Thus in the design stage, selection criteria is considered.

Two main selection criteria, namely domain and Availability were considered during the design phase. Domain is one of the parameter used for corpus classification. Domain mainly shows the writing style it contains and selection of domain directly related with the purpose of corpus building. The objective is to build LM for speech recognition task which is trained from read corpus which are mainly statement type sentences [4]. Therefore, news domain is identified for this research since text from news domain is the possible source from which statement type sentences can be collected.

The other consideration, in the design of corpus building, is the availability issue. In contrast with other domains, Amharic news text can be acquired more easily with reasonable size. However, there is no an easy way to know the appropriate corpus size to be collected for a

certain task[48]. The general rule, thus, is acquiring as large as possible from the selected domain and that is actually what has been done in our case too.

3.2.2. Creation stage

Creation of Amharic text corpus includes activities such as copy right permission, corpus acquisition and data pre-processing.

a) Copyright Permission

In order to make the corpus accessible for the entire research community, copyright clearance was obtained for all text included in the corpus. The approval[8] was obtained from the right owner (ENA) on conditions listed below:

- to allow the materials to be used for creation of text corpus,
- that the text corpus be used for the academic research and not for commercial purpose

b) Text Collection (Corpus acquisition)

Even though other sources were also considered, eventually, the researcher succeeded in acquiring 5GB MSSql server database file that holds text from Ethiopian News Agency (ENA). The database consisted of news, articles and other text produced over seven years [2003 - 2009]. The database includes those text that were published in the ENA official website[9] and news prepared for other media such as Ethiopian national television and radio [ERTA].

c) pre-processing

Preprocessing is a task of conditioning the corpus in the way it could be used for subsequent tasks through the course of language model development. The 'raw' Amharic text corpus was not in a form that could directly be used for the task. Hence, it has gone through preprocessing phase. While some of the preprocessing task is done automatically, some are carried out just manually. The major preprocessing tasks done on the ENA data are the following:

[8] The conditions to use the corpus is stated in appendix E.
[9] The official website of ENA : www.ena.gov.et

i. Extraction of the news from MSSQL server

The news was stored in the news column along with font type, color and other text formatting information. Hence, it was not possible to get the text by executing an SQL statement on the server directly. Rather, a program is written to extract the plain text and write it into a file. The remaining pre-processing tasks mentioned below are carried out to clean the corpus on the text file.

ii. Removal of Non-Amharic Script

The news database had English news in addition to Amharic news. When the news are being extracted from the database, it was based on a "language" attribute of the news table. However, there was English news whose language value was entered to be Amharic mistakenly. This English text is removed automatically using regular expressions on Linux text editors like vi. Other task such as removal of English terms found in the middle of Amharic text was done manually provided that if it does not affect the meaning of the sentence.

iii. Removal of punctuation marks

Punctuation marks such as commas, periods, and colons are critical for finding boundaries of things, and others like question marks, exclamation marks, and quotation marks are important for identifying some aspects of meaning. Furthermore, in some tasks, like part-of-speech tagging or parsing or speech synthesis, punctuation marks are treated as if they were separate words. On the contrary, these punctuation marks are not important to predict the next word using statistical language models. Thus, all Amharic punctuation marks are removed.

iv. Separation of News text from other articles

As mentioned above, the domain of the corpus needed is to be a news corpus. However, other contents such as feature articles, interviews, minutes and other non news content are in the database. Removal of these non news content is done manually. Thus, the researcher do not claim that all of them are removed due to the big size of the text.

v. Expansion of Acronyms

The text extracted from the news database was having lot acronyms. Expanding the acronyms were done manually and found to be one of the difficult tasks in the pre-processing phase as

there were inconsistency in using different acronyms referring to same word or phrase. For example, both ዓ.ም and ዓ.ም /A.C or After Christ/ read and mean same thing. However, different alphabets are being used. These are expanded to same grapheme string.

Furthermore, some are shortened using different alphabets and read differently as well. For example, እ.አ.አ , እ.ኢ.አ and እ.ኤ.አ all refers to same thing: European Calendar. They are same in their semantics, but read differently. Since this might shows and represents interpersonal variation between people in their writing style and speech, such terms are expanded keeping the way they appear in the corpus.

The preprocessed Amharic text corpus, therefore, is the subset of the 'raw' text with all acronyms expanded, punctuation removed, non-Amharic contents avoided and so on. Generally, a clean Amharic corpus with a content relevant to our task. However, it still cannot directly be used to build the language model as the tool used to build the model does not support non-English alphabets. Therefore, Amharic-to-English transliteration is being done as described below.

3.2.3. Transliteration

One of the late preprocessing tasks done on the Amharic text corpus is transliteration. Transliteration is the task of mapping one set of characters in one language to other set of characters in another language. Transliteration could be done either manually or automatically. Manual transliteration might work fine when the text to be transliterated is small in size. However, since ATC is millions of words of corpus, the manual method is not feasible. Thus, the researcher has developed his own Translitrator.

Translitrator is a program which does mapping of Amharic grapheme text to its equivalent Latin grapheme text. This was done due to the requirement imposed by the modeling tool [38] being used for experimentation. Translitrator[10] is written in Python language and uses the transliteration scheme[11] proposed by [54]. The task of the Translator includes mapping of Amharic characters to English characters and conversion of numbers and date to their citation form. For, example number 500 is transliterated to "amst meto" [Five hundred],

[10] The source code of the Translitrator can be found attached in appendix A
[11] Transliteration scheme used by Translitrator is attached in appendix B

3.2.4. Corpus statistics

After doing all the preprocessing tasks described above, 79MB size of transliterated Amharic news corpus is obtained. This could show how noisy our text database was plus the time and efforts it took to filter out 79MB corpus from 5GB database. As can be seen from table 3.1, the final corpus used for this study consists of 409,398 distinct sentences which in turn having a total of 9,079,766 tokens and 315,206 distinct words. To the best of our knowledge, the largest Amharic corpus used for natural language processing research is the one used by [3] which consists of 120,262 sentences which is 29% of the corpus prepared for this study. To this end, our corpus is fairly large[12] as it consists of 409, 398 sentences (which is named to be ATC-409K after wards). But still the researcher do not claim that it is sufficiently large and even it is smaller when compared with corpora of other languages such as the 127 million words of Estonian and 150 million words Finnish languages [1].

Table 3. 1 General statistics of ATC-409K corpus

No	Parameter	Size
1	Number of Sentences	409, 398
2	Number of Word tokens	9,079,766
3	Number of Word types	315,206

Word frequency distribution of ATC-409K corpus is shown in Table 3.2. More than 50% of the words appear only once in the corpus. This implies that there exist a lot of infrequent words in the corpus and some of this infrequent words are due to typographical error in the text and some words might be concatenated and hence contribute to the size of the infrequent words. Words appeared to be frequent, for example, 105 words appear more than 10,000 times and most of these words are action words which are common in a news corpus such as ገለፀ /gelexxe/, ነዉ /new/, ተገረ /tenegere/ etc. Moreover, it also consisted of frequent words from various domains like health, politics, tourism and so on.

On the other hand, some words show only modest frequency. This frequency characteristic of natural language is described in zip's law as: "there are a few very common words, a middling

[12] Though there is no an agreed up notion on how much of data is said to be large and small, corpus consisting of less than 1 million words is said to be small.

number of medium frequency words, and many low frequency words" [14]. To this end, our corpus exhibited the characteristics described in zip's law.

Table 3. 2 Word frequency distribution of ATC-409K corpus

No	Word Frequency	Number of words	Percentage (%)
1	1	158406	50.25
2	2 – 10	113886	36.13
3	11 – 100	33581	10.65
4	101 – 1000	7926	2.51
5	1001 – 10, 000	1302	0.42
6	> 10,000	105	0.04
Total		315206	100

3.2.5. Splitting the corpus

Very often, there is a trade off in dividing corpus in to training and test sets in statistical models since large amount of both training and test data sets are required to build robust model. Test data set should be as large as possible because a small test set may be accidentally unrepresentative. Though, it is hard, in principle it is required to pick the smallest test set that is representative and gives enough statistical power to measure a statistically significant difference between two potential models [14]. To this end, literatures [3,14] shows that data usually divided into 80% training, 10%held-out, and 10%test sets.

However, to make more reasonable and acceptable generalization k-fold validation is carried out. To this end, 10-fold validation experiment were performed as part of the preliminary experiments. The detail of this experiment is presented in Chapter 5. The corpus is first sorted out in ascending order so as to make random sampling. It is then partitioned into 10 disjoint set of corpus using a small python program (splitter[13]) developed for this purpose. Given the entire corpus, splitter systematically split the corpus into ten equal and disjoint data subsets containing 1/10 of the total corpus. The basic pseudo code of the program is given in figure 3.1.

[13]Appendix [A] shows small Python script used for dividing the corpus into training, held-out and test sets

```
Input: file containing list of sentences labeled by
their line number
For  i= 1 to 10
        for j=i to the total # lines in the input file
                write line j to file i
                j=j+10
        End
End
```

Figure 3.1 pseudo code of the splitter script

The input text file is list of sentences labeled by their line number according to their position in the file. Data subset k takes every 10th sentence of the input corpus starting at line k where the value of k is 1 to 10. Accordingly, 10 data subset are generated and these data subsets are arranged into test, held-out and training data with 10:10:80 proportion in 10 different combinations. The i^{th} combination consists of a test data set of data subset i; held-out data set of ((i mod 10)+4) mod 10) and the rest data subsets are merged together and used as training data set.

Making sure that every dataset is used for training and evaluation, 10 different tri-gram LMs were built using the 10 different combination of the datasets. The data sets that provides a perplexity result closer to the average is taken to be used for the entire experiments. *Table 3.3* shows statistics on training, test and held-out corpus used for this study.

Table 3.3 Training, test and held-out data sets.

Data set	Number of lines
Training (ATC-train-327.5K)	327,518
Test(ATC-test-41K)	40,940
Held-out(ATC-heldout-41K)	40,940

The training set is made up of 327, 518 sentences (here after named to be ATC-train-327.5K). On the other hand, each the held-out and test sets have got 40,940 sentences and named to ATC-heldout-41K and ATC-test-41K here after respectively. These data sets are used to build the different language models throughout this study.

3.3 Summary

Even though Amharic is a widely spoken language in the country, no computational resources such as text corpus have been made available for research works. In this chapter, the researcher showed the collection and compilation of Amharic text corpus (ATC) from the news domain. The corpus consisted of 409, 398 lines and 9,079,766 tokens. It is being used for language modeling task, however, by doing more works on the corpus such as corpus annotation with part of speech tag shall make it more useful for other tasks such as spelling and grammar checker, semantic network building and ontology learning and so on.

CHAPTER FOUR
AMHARIC LANGUAGE MODELS CONSTRUCTION

In the previous chapter the development of Amharic text corpus that are used to build different LMs in this thesis is discussed. This chapter delves into the design of word n-gram and class-based Amharic language models. Moreover, the tools and algorithms used in the study are also described.

4.1 Modeling Tool

SRILM [38] is one of publicly available Language Modeling toolkit on the internet developed by Stanford University Research Institute. Basically, it is composed of C++ libraries, executable programs and other scripts that allow the development and experimentation of statistical language models. It came to existence out of dissatisfaction with previously available LM tools like CMU-Cambridge toolkit and the HTK Lattice toolkit. It is designed with goals of building state of the art LM algorithms, providing flexibility and extensibility in language modeling and coming up with convenient commands for LM development and testing

In addition to the N-gram language model, SRILM supports several other LM types including Class-based models, Cache models, skip language models and interpolated language models. Furthermore, it conveys other LM applications arising in speech recognition such as a word boundary tagger, lattice tool, helper script that convert LMs to word graphs and others [39].

4.2 Amharic N-gram language models

N-gram language modeling is the state of the art language modeling approach. Despite the criticism for vulnerability of data sparseness and failure to capture long range dependency, it gives quite reasonable performance provided that appropriate smoothing and reasonable size training corpus are used [15]. Moreover, n-gram language model is much easier to build than other approaches. However, the researcher observed that an n-gram language models with an appropriate smoothing and reasonable corpus size is not built so far Amharic.

Therefore, our experiment on Amharic language modeling began by building the standard word N-gram language models. Detail explanation of N-gram language models is given in

Chapter 2 of this document. The block diagram (see figure 4.1) below shows our n-gram language model construction process based on the SRILM tool.

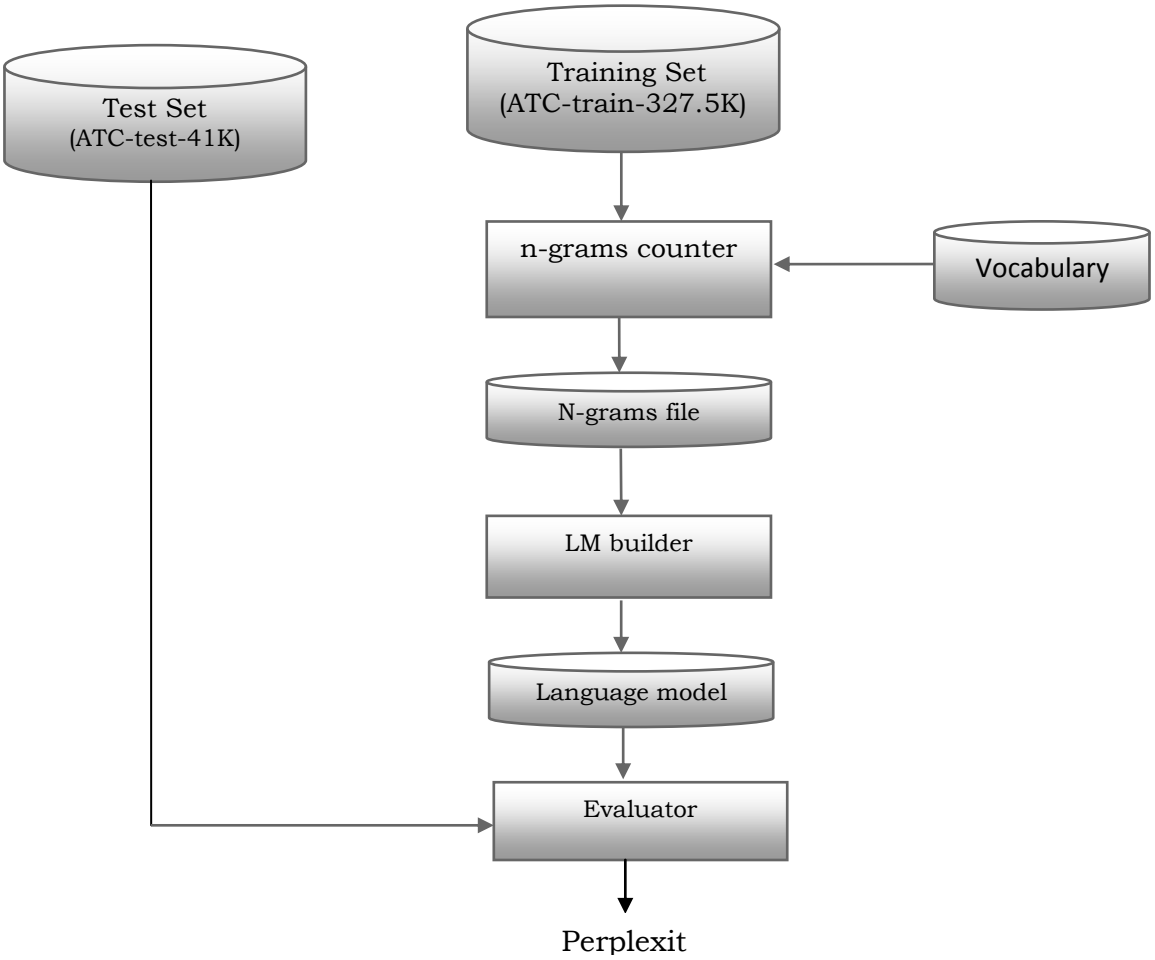


Figure 4. 1 Block diagram of Amharic N-gram language mode

4.2.1 Training set

Language model encodes the likelihood of word sequences of a given language from the training data. Therefore, the training data has a big impact on the quality of the model built. For this study, as shown in chapter 3, a training corpus (ATC-train-327.5K) that consisted of 327, 518 sentences is prepared. Therefore, it is used to train all the n-gram models.

4.2.2 Vocabulary

A preliminary step in language modeling is to prepare an appropriate vocabulary. The selection of vocabulary entry is constrained by different factors such as the task to which the language model is being built for. For example, a language model specifically designed for

speech recognition system could have set of words which exist in pronunciation dictionary of the recognition system. Often, nonetheless, the vocabulary consists of most frequent words from the training corpus [9, 36].

Language models have finite vocabulary words, thus, they are subject to out of vocabulary problems (OOV). For example, the test set might contained of words that are not found in the vocabulary and these words are said to be OOV words. Depending on how language models deal with OOV words, its vocabulary is named as either open or closed.

A closed vocabulary model assumes all the words that will be encountered in the test set are in the vocabulary. Thus, closed vocabulary tasks take no assumption of unknown words and the occurrences of such words in training and test data will cause an error. Such systems could be used in limited vocabulary tasks such as command/control environment where the vocabulary contains commands used to operate the system [15].

An open vocabulary system mimic the potential unknown words in a test set by adding pseudo-word like <unk>. For example, a tri-gram አበበ ፓስታ በላ meaning /abebe ate pasta/ would be converted to አበበ <unk> በላ if the word ፓስታ /pasta/ is not in the vocabulary. In other words such systems consider <unk> symbol just like other words in the vocabulary and its probability is computed as:

I. Choose vocabulary in advance (fixed size)
II. Map all words in the training set to <unk> if they are not found in the dictionary
III. Estimate the probabilities for <unk> just like other in-dictionary words

That means, every possible word that the system might encounter will have some non-zero probability.

To train the Amharic n-gram LMs a vocabulary that consisted of frequent 151, 049 [151K-V] words from the ATC-409K corpus were prepared. Only frequent words are considered because the task of language modeling is to model the pattern and regularities of word sequences, much information cannot be gained from rare events such as things seen only once.

4.2.3 n-grams counter

N-grams counter collects and store n-gram counts into n-gram files. The input to this tool is the training data, the order of n-grams and the vocabulary. After scanning of the input training text, it generates counts of n-grams. These counts can be vocabulary independent, and contain frequency of all n-grams(for all the words in the training set) up to specified order. For example, if one wanted to count 4-grams, all unigram, bi-gram, tri-gram and tetra-gram counts are generated. The lower order counts are collected since they are useful for LM smoothing. On the other hand, it can be vocabulary dependent, in which case all the OOVs are mapped to special symbol such as <unk>. Therefore, the output of this module is a file that holds n-gram statistics of the training data. Later on, the counts in these files are used to build the language model.

N-grams file is a count file that consisted of frequencies of n-grams obtained from the training corpus. Figure 4.2 shows sample for vocabulary dependent 3-gram count file (n-grams followed by their count). For example, the 9th entry of the list shows that the symbol <unk> is used since the word that comes after 'yemiiyanesaw yejermenu' is missing in the dictionary. It is this file primarily used to build the language model.

```
yemiiqombet                          4
agoberun                             7
yasayut                  111
mesariiyana xxere                      1
mesariiyana bebomb                     1
mesariiyana balemuyawoc      2
mesariiyana yesew                      3
mesariiyana medheniit        6
yasayut tesatfo yemiideneq                3
yemiiyanesaw yejermenu <unk>  1
ixndargalen ixnj keesxebariinet   4
```

Figure 4. 2 Sample N-gram count file

4.2.4 LM Builder

Taking the n-gram count file generated in the previous phase, the task of the language model builder is to compute n-gram probabilities which are stored in the language model file. To address the issue of data sparsity problem, smoothing algorithms needs to be used instead of MLE during language model construction.

Smoothing algorithms are described in section 2.2 of chapter 2 of this document. [16] stated that while some are easy to implement others are relatively difficult and while some perform better others do not. In this thesis, Whitten-Bell, Kneser-Ney, Absolute discounting, Good-Turing and Modified Kneser-Ney smoothing algorithms are being applied so as to compare their relative performance and pick one that gives the best performance over others. The experiment that shows relative performance of these algorithms are given in the next chapter (see chapter 5).

4.2.5 n-gram LM

An n-gram is a sequence of n symbols and an n-gram LM computes probability of each symbol given the preceding n-1 symbols. The figure below (see Figure 4.3) shows sample back-off tri-gram language model. A back-off language model specifies the likelihood and back-off probabilities for n-grams up-to the specified order. For example, a trigram language model has probabilities for unigram, bigram and trigrams and back-off probabilities for unigram and bigram models. The back-off probabilities are used to compute the probability of the n-gram from the next lower order n-gram when it is not actually seen during the training phase.

```
\data\
ngram 1=10000
ngram 2=4500

\1-grams:

-0.119325    yalacewn    0.00311902
...

\2-grams:

-3.332524    yalacewn ixrkatana    -0.1731701
-3.848996    yalacewn ixweqet
-2.072269    yalacewn ixwqet
-2.120574    yalacewn ixwqetna    0.005117089
...
\3-grams:

-0.2198163    tesatfo bemetekel lay
-0.2198163    gedeb bemetelalefacew yetxfatenxenet
-0.2401845    <s> bemetema wereda
-0.3778298    bemezewawer bemetema bekul
-0.3778298    beteley bemetema wereda
-0.5539211    <unk> bemetemamen new
...
```

Figure 4. 3 Sample back-off tri-gram language model

A typical ARPA format Back-off LM, starts with a header keyword "\data\", listing the number of N-grams of each length. n-grams are listed one per line preceded by the group label "\N-gram:" where N is the length of the n-grams to follow. Each N-gram line starts with the logarithm (base 10) of conditional probability P of that n-gram, followed by the words $W_1...W_n$ making up the n-gram. These are optionally followed by the logarithm (base 10) of the back-off weight for the n-grams in the model. For example, in a tri-gram LM, all tri-grams will not have Back-off weights as there is no other words that follows. Meaning that there is no an n-gram that might back to the tri-grams.

4.2.6 Evaluator

Evaluator applies the intrinsic evaluation metric which is known to be perplexity. Given a LM and test set which does not form the training data, evaluator calculates the perplexity of the model against the test set data. The detail of the perplexity computation is given in chapter 2 of this document. According to this metric, a language model that shows lower perplexity value is the better one since perplexity is inverse of the probability the model assigns to the test sentences.

4.3 Amharic Class-Based Language Models

In the previous section [section 4.2], Amharic n-gram language model design is presented. Though, n-gram models provide reasonable performance they are subject for data sparsity problem and different smoothing algorithms were also used to address the problem. Basically smoothing techniques tries to overcome data sparseness problem of standard n-gram models by employing two basic strategies known to be backing-off and interpolation [15]. However, these methods do not deal with other problems of n-gram model such as its large number of parameters and high dependence on discourse domain [32]. A supplement to the above mentioned partial solutions for data sparseness is class-based language modeling [32]. In this section, the class-based LM construction is discussed.

Data sparsity is a known problem of n-gram LMs even having very large corpus; though it is sever on smaller size corpus. An alternate solution then would be to cluster similar words (having similar syntactic and/or semantic functions) into same class and the approach is called class-based language modeling [8]. This approach uses fewer contexts than word n-grams as it predicts the next word looking at class histories which are normally fewer in

number than distinct words, thus generalize better for rare or unseen higher order n-grams. For example, let us say our training data has the following 4 sentences (each separated with comma)

From addis, To Bahirdar, To Mekele, from school

Suppose Addis, Bahirdar and Mekele are in class “city” and To and From are in class “preposition”. For certain flight reservation system, one wanted to compute the bigram “to addis“ and if MLE P(addis|to) is applied, it is evaluated as: P(addis|to) = $\frac{C(\text{to addis})}{C(\text{to})}$ =0. Even though the bigram "to addis" is not in the training set, it is highly likely that one can pose the enquiry "to addis" to the system and LM should not return 0. However, class-based language model gives non 0 probability despite of the fact that it didn't see the actual bi-gram and the probability is computed as Pc(addis|to)= p(addis|city) * p(City|preposition) which means 0.33 * 0.75. Accordingly, the bi-gram receives non zero probability and that is reasonable.

As can be seen from the above example, class-based language model use the relationship between words by clustering them into classes. As a result, it tackles data sparseness by using the probabilities of word classes instead of those of individual words and thereby reduces number of parameters. This helps to build compact LMs [8]. Generally, introducing the concept of classes into a language model brings the following advantages [40]:

- Grouping words into classes reduce the number of contexts in a model, and thus reduce data sparsity. Thus, it serves as a solution for the data sparsity problem of word n-gram models. Languages that might suffer more from data sparsity problem due to their complex morphology can benefit from this approach.
- Reducing the number of contexts generally results in a smaller parameter space. Meaning that it minimizes the memory requirement and can be used by devices having computational resource constraints like mobile Phones. For example, [8] has shown a trigram class-based model requires only about one-third as much storage as the word trigram models. Meaning, it enables the use of a larger context with less cost than n-gram models.
- Class-based models combine information for all words within the same class, and therefore are expected to generalize better to unseen words. This ability to generalize

to unseen events is known as language model robustness, which is important for large vocabulary continuous speech recognition tasks that generally expected to have high OOV rate.

Our goal, thus, would be to build class-based language model and see if Amharic could really gain the aforementioned benefits from the approach.

4.3.1 Detail of our class-based language model approach

Data sparsity is an issue for morphologically rich languages like Amharic [3]. One way to tackle this problem is to use a class-based model that uses frequency of sequences of classes to help produce a more knowledgeable estimate of the probability of word strings [41]. This is because, words are clustered either based on the syntactic/semantic (in case of knowledge based clustering) information about the language or based on their statistical distribution in case of data driven clustering approach. Even though data driven approach is based on statistical distribution of words, it has proven that words that have related syntactic role/semantics are kept in same class. For example, in our primary experiment and in [8] days of the week are kept in the same class. Therefore, the primary assumption is that using such knowledge would be helpful to build language models.

In this study, the researcher wanted to build and evaluate class-based language models and examine if it results in performance improvements over the standard n-gram language models. Moreover, its performance when combined with word n-gram LMs is also investigated.

In this study, class-based LMs that determine the probability of a word w_k based on its history as the product of the two factors; the probability of the class given the preceding classes, and the probability of a particular word given the cluster is built[8]. Thus, our class-based language model shall be generalized by:

$$P(w_k|w_{k-n+1} \dots w_{k-1}) \equiv \prod_{k=1}^{K} P_w(w_k|c_k) P_h(c_k|c_{k-n+1} \dots c_{k-1}) \qquad (4.1)$$

Where K is the length of a sentence and the probability P_h of class c_k is computed as:

$$P_h(c_k|c_{k-n+1} \dots c_{k-1}) = \frac{C(c_{k-n+1} \dots c_k)}{C(c_{k-n+1} \dots c_{k-1})} \qquad (4.2)$$

Where $C(c_k)$ is count of class c_k.On the other hand, the probability of word w_k given its classes is computed as:

$$\mathbf{P_w(w_k|c_k) = \frac{C(w_k)+1}{S(c_k)+C(c_k)}} \qquad (\mathbf{4.3})$$

Where $C(w_k)$ is the count of the predicted word w_k and by $C(c_k)$ means that the number of words in the training text for which the class is c_k . And, $S(c_k)$ is the size of class c_k which is introduced as a normalization factor since 1 is added to the count of each word more than it actually occurs.

4.3.2 Class-based language model construction

This section provide over view of the overall process of building class n-gram language model based on the SRILM [38] toolkit. As described in Section 4.3.1, a class-based language model consists of two separate components. The first is an n-gram which models the sequence of classes (i.e. $P_h(c_k|c_{k-n+1} \dots c_{k-1})$) and the second is a class map with associated word counts or probabilities within classes allowing the word given-class probability bigram $P_w(w_k|c_k)$ to be evaluated. Figure 4.4 illustrates the block diagram for class-based language models.

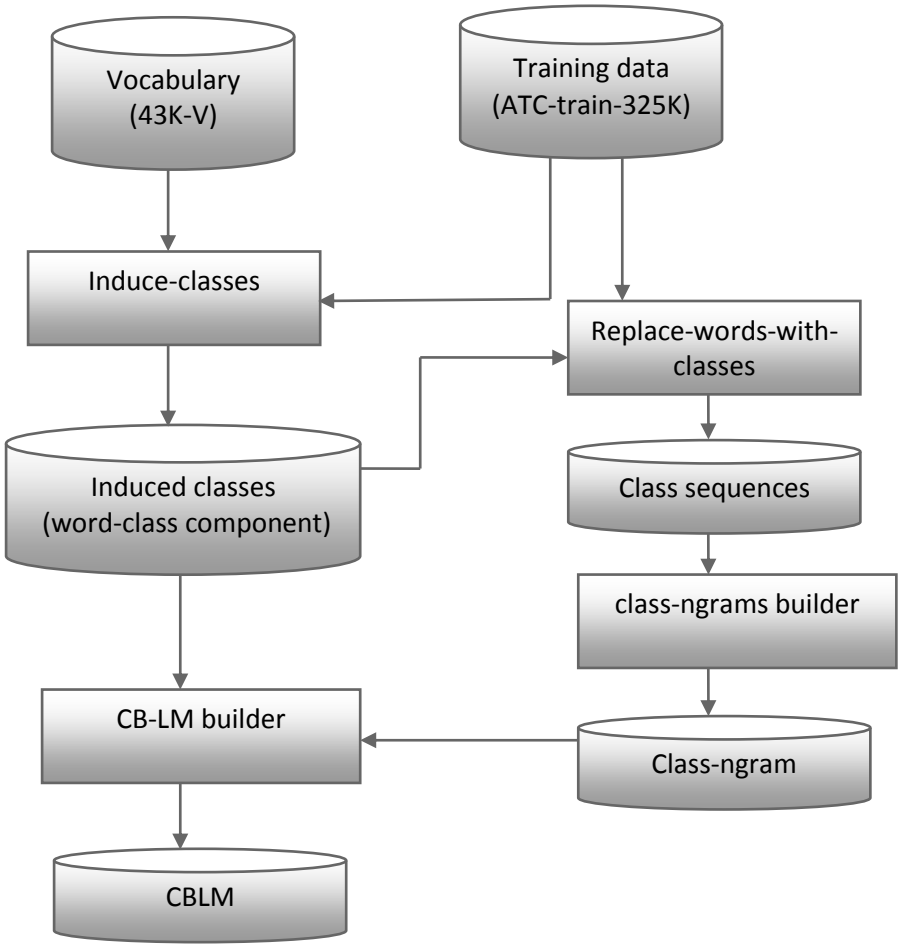


Figure 4. 4 Block diagram of class-based language models

A class-based n-gram model is similar to a word-based n-gram in that both store probabilities of n-tokens; except in the class-based model case, these tokens are word classes instead of words. Thus building class-based language model is not by far different from building word based n-grams. Thus, components that are common to word n-gram models (see section 4.2) are not described. However, some components are briefly discussed to indicate the difference with word-based n-gram models.

Vocabulary

The first task in class-based language modeling is to prepare a vocabulary. Therefore, a new vocabulary that consists of 43,000 most frequent tokens [here after named to be 43K-V] is prepared from the ATC-409K corpus.

Training data

To train the parameters of the class-based language models, the training data used for n-gram language modeling (i.e. ATC-train-327.5K) is used. Specifically, the training data is used to induce classes automatically and to compute the class n-gram component of class-based LMs

Induce-classes

One of the important aspects in building class based language model is the decision made on how to induce the classes. So far there exist two widely used mechanisms to do it: the first one is a knowledge based approach where Part Of Speech Tagger (POS) is commonly used as a tool and the other one is a statistical approach by which words are clustered based on statistical evidences obtained from the corpus. The former one is generally used to produce a small number of clusters and that might lead to increased perplexity [41].

In this study, the researcher used the later approach due to unavailability of a working Amharic Part Of Speech Tagger (Amharic POS) and it is infeasible neither to do it manually as the size of the corpus is big nor to develop Amharic tagger due to the task requirement to have a good linguistic expertise. Furthermore, literatures [41 and 32] reported that automatically derived clusters outperform knowledge based (part-of-speech tags) clusters, at least when there is enough training data.

Induce-classes construct a class map which defines which class each word belongs to. It takes the training text and list of words that are to be partitioned into classes. It is also used to generate the word-given-classes component of class-based LM. Clustering is an important aspect that makes class-based language models different from the word based n-gram models. There exists different clustering algorithms that use differing criterion to map words to classes and one of them is IBM clustering [8]. In this study, IBM clustering algorithm is used due to its popularity and the researchers convenience.

IBM clustering

IBM clustering algorithm [8], also named Brown's class joining algorithms, is one of the most well-known and effective clustering algorithms in LM. The algorithm tries to find

partitions that maximize the average mutual information (AMI)[14] of adjacent classes as much as possible. And the AMI function is given by :

$$AMI=\sum_{c1c2} P(c1c2)\log\frac{P(c2|c1)}{P(c2)} \quad 4.4$$

where c1 and c2 are classes resulting by mapping of words to classes and P(.) is probability. This algorithm put all words in a vocabulary into its own distinct class, and then successively merges pair of classes temporarily and compute AMI. Thereafter, the best cluster pair, which resulted in a minimum AMI loss, is permanently merged. After having derived a set of classes from successive merges, it cycles through the vocabulary moving each word to the class for which the resulting partition has the greatest average mutual information.
Figure 5.3 shows the detail of the algorithm.

IBM clustering Algorithm
Initialize: put a single word in each cluster
compute the initial AMI of the collection
Iterate until desired number of classes are reached
for each pair of clusters **do**
merge the pair of clusters temporarily
compute the AMI of the collection
end for
merge a pair resulted in minimal loss of AMI
compute AMI of the new collection
repeat
Move each term to the cluster for which the resulting partition has the greatest AMI
until no more increment in AMI

Figure 4. 5 IBM clustering Algorithm

Replace-words-with-classes

Class-based LM has two components; the first one is the word-given class component which is computed by induce class component. The other one is the class n-gram component. The class n-gram is simply the class sequences of words in the training data. Thus to obtain the class sequence, words in the training data need to be replaced by their class label. Therefore, Replace-words-with-classes takes induced classes to look up the class label for words in the training data**.**

[14] The derivation of the average mutual information can be found on Brown et al., 1992

Class-sequences

It is the training corpus with all words are replaced with their respective class label. This is the input to build the class n-gram component of class-based LMs.

class n-gram builder

It is a tool that compute the class probability given previous classes. Using this tool, any of smoothing algorithms that are used to smooth word n-gram models can be applied. The difference between class n-gram and word n-gram is that while the former models class sequences, the later models word sequences.

CB-LM builder

Class-based Language model builder (CB-LM builder) compute class-based probability of a word taking the product of the word-given-class and class-n-gram components.

4.3.3 Analysis of words in clusters

IBM clustering described above (see section 4.3.2) is employed in order to automatically cluster words in the 43K-V vocabulary. Accordingly, 700 classes are induced which is proportionally reasonable to what is being recommended for 64K vocabulary by [9]. Table 4.1 shows examples of 10 classes that are found to be particularly interesting. The number of words in each cluster is highly varied from 1 to more than 500. To make the table manageable, a maximum of 15 words are included here and full member of these sample classes is attached to appendix D.

Table 4. 1 ***examples of automatic word groupings***

Class labels	Member words
Class 7	yemabazat, yemadaqel,yemarabat,yemasefafat,yemadares, yemaseracxet,yemaberetatat,yemabqat,yemadaber, yemaseltxen,yemagolbet,yemalamed, yemasadegu, yemamecacet, yemamualat,
Class 8	sera, sra, sracew, sram, sraw, srawu
Class 11	Arb, hamus, hemus, ixhud, ixrob, kremt, maksenxo, qdamie, rebuix, rob, samnet, samnt, semon, senxo, wer
Class 154	maletacewnm, maletie, maletu, melixktacn, melsu, siilu, siilum, sl, stl, stlm baybalm, bayoc, bemalet, bemaletacew, eluna
Class 306	gelexxec, gelexxewal, gelxxalec, gelxxewal, txequmewal
Class 309	baleend, bebzu, yemulu, legmasx, yerub, beend, beerat, beesr, lemejemeriiyaw behulet, ixskeasraarat, bemecxeresxaw, besdst, snt, besostenxaw
Class 317	beixqd, betaqedew, hlmacewn, ixqd, ixqdm, ixqdn, ixqdoc, ixqdocacewn, ixqdocn, ixqdun, leixqdu, poliisiiwocu, raixyacewn, raixyun, stratiejiiwn
Class 322	bealemacn, beehguru, beegeriitu, begzatiitu, bemediinawa, beketemacn, beweredawoc, beekababiiwocu, bewenzu, betxabiiyaw, beyekllocu, bekampu bemuziiyemu, beparku, beierportu
Class 548	balebietu, balemoyaw, balemuyaw, danxaw, danxocu, dayriekterua, diinu, dokteru, driekteru, temeramariiwocu, tewekaywa, preziidantu, priezdantu, jienieralu, profieseru
Class 550	balie, beborenana, behoro, belesa, butajra, driedawana, ebaya, efar, emara, ermacxho, ersiina, gambielana, godie, gojamna, gondern

The degree to which the classes capture both syntactic and semantic aspects of Amharic is quite impressive. Furthermore, pronunciation variation, which is a critical issue for speech recognition task, is also captured quite interestingly. For example, {(balemoyaw, balemuyaw) ,{(ixrob , rebuix) } are kept into same class. Furthermore, in most cases classes hold words which are semantically related with one another. For example, class 11 consisted of only weekdays and words related to date.

CHAPTER FIVE
EXPERIMENT AND EXPERIMENTAL ANALYSIS

5.1 Introduction

This chapter presents various experimental results of our Amharic language models. In a nutshell, three set of experiments are conducted on Amharic LMs. The first series of experiments are perplexity experiments on Amharic word n-gram LMs [see section 5.2]. To that end, test set perplexity results on higher order LMs, comparison of smoothing algorithms and interpolation of LMs are presented.

The second series of experiment is on Amharic class-based LMs. The performance of word-based and class-based LMs are compared based on perplexity measure. Combined models where the class-based models are interpolated with word-based n-gram models are also analyzed in this experimental group.

The third series of experiment is an extrinsic evaluation where our LMs are applied on Amharic speech recognition task. Specifically, word recognition accuracy (WRA) experiments are carried out using Amharic word n-gram, class-based and combined LMs. These experiments are crucial since the best way of evaluating LMs is putting them on the application for which they are meant for. Therefore, the relative contribution of word-based, class-based and interpolated models for the improvement of the baseline speech recognition system is evaluated.

5.2 Experiments on Amharic N-gram Language models

This section presents different perplexity experiments done on Amharic N-gram language models.

5.2.1 Preliminary Experiments

In this series of experiments, the researcher tried to address basic issues related to language modeling. The first one is experiment on the corpus based on which the training, test and held-out datasets are chosen. Moreover, other experiments are conducted to see perplexity results of lower order word n-gram LMs and to compare open and closed vocabulary models.

i. Experiment on the corpus

In statistical models, there is usually an issue of how the results of a statistical analysis will generalize to an independent data set. That means sometimes the test set data keenly fit the model's parameters and an exaggerated result may be reported, for example if the test set is taken from the training data. On the other, hand the test set may be too far from the models expectation; for example, if the training and test sets are taken from different domains. Therefore, techniques like k-fold validation is used to see performance variability among the K different experiments. In this experiment, 10-fold validation is done on tri-gram LM. This experiment allows us to see variability and to select training and test set data that behave near the central tendency. The selected training, test and held-out data sets are used for further experiments in this research work.

The transliterated Amharic text corpus that consisted of 409,398 lines (ATC-409K corpus) is prepared for this research work. Furthermore, A vocabulary that consisted of 151,046 words (we call it here after as 151K-V) that appeared twice in the ATC-409K corpus is prepared. The ATC-409K is then partitioned in to 10 non-overlapping data sets (DS1 to DS10). To divide the corpus into training, test and held-out data according to the 8:1:1 ratio, 10 random combination of these 10 data sets are taken to build 10 tri-gram LMs where in each case 8 out of 10 data sets are used for training and 1 data set is used for each test and held-out data. Table 5.1 shows the perplexity results of ten experiments.

Table 5.1Perplexity result of a tri-gram LM on 10 different data sets

Experiments	Test set	Held-out set	Training set	Perplexity
EXP1	DS1	DS5	DS2…DS4, DS6…DS10	256.269
EXP2	DS2	DS6	DS1, DS2…DSs, DS7…DS10	258.815
EXP3	DS3	DS7	DS1…DS2, DS4…DS6, DS7…DS10	255.454
EXP4	DS4	DS8	DS1…DS3, DS5…DS7, DS9…DS10	254.947
EXP5	DS5	DS9	DS1…DS4, DS6…DS8,DS10	255.41
EXP6	DS6	DS10	DS1…DS5, DS7…DS9	255.782
EXP7	DS7	DS1	DS2…DS6, DS8…DS10	254.117
EXP8	DS8	DS2	DS1,DS3…DS4, DS9…DS10	255.96
EXP9	DS9	DS3	DS1,DS2,DS4...DS8,DS10	256.956
EXP10	DS10	DS4	DS1...DS3, DS5...DS9	261.425
Average				256.514

As can been seen from the table, varying perplexity results are observed due to the variation in the training and test set data considered in each experiment. As can be seen from the table, EXP7 resulted in lowest perplexity while EXP10 shows the highest of all the 10 models. According to k-fold technique, the average of the 10 experiments is taken to be reliable. Therefore, the data sets that give perplexity value closer to the average perplexity shall be taken as reliable and therefore all other experiments that follows are done using these data sets. Accordingly, EXP1 with perplexity value of 256.269 is closest to the average which is 256.514. Therefore, the datasets in EXP1 which consisted of 327,518(ATC-train-327.5K), 40940(ATC-test-41K) and 40940(ATC-heldout-41K) lines of text for training, test and held-out respectively are picked up for other experiments carried out in this thesis

ii. Experiment on lower order LMs and vocabulary type

N-gram LM of order up to 3 gram is usually considered to be lower order models. Using 151K-V and ATC-train-327.5K training set, a set of open and closed vocabulary models are trained. In the open vocabulary models, OOV words are mapped to <unk> symbol whereas in closed vocabulary models OOV words are simply ignored. In this experiment, unigram, bigram, and trigram open and closed vocabulary models are trained and their relative performance is compared using perplexity measure. Good-Turing discounting is applied for all models and Table 5.3 shows the perplexity results obtained.

Table 5.2 Perplexity results of lower order LMs

Order of LMs	Open vocabulary models	Closed vocabulary models
Uni-gram	5411	5758.61
Bi-gram	319.052	329.583
Tri-gram	256.269	260.516

All the models are then evaluated using ATC-test-41K test set. Generally, open vocabulary models performed better than closed vocabulary models. Furthermore, as the order of n-grams increases there happened reduction in perplexity and in both closed and open vocabulary models, trigram LM performed best. The reduction in perplexity as the order of N increases motivated us to do further experiment with higher order N-grams [see section 5.2.2]

On top of the experimental result, closed vocabulary models are not appropriate for speech recognition systems where it is difficult to know what a speaker likely about to say. That is

because, closed vocabulary systems end up in error if they encounter words that are not in their vocabulary. Hence, closed vocabulary systems shall be used only in the environment/system where all possible words are known. Since our language models are going to be applied on speech recognition system where words that speaker utter are potentially infinite, open vocabulary models are more appropriate. Open vocabulary models are more natural and realistic as they leave space for words which are not included in the vocabulary and training data. Therefore, open vocabulary language modeling approach is applied throughout this thesis work.

5.2.2 Higher Order N-Grams

One of the pit falls of lower order N-grams is that they only depend on shorter context to predict the next word. Amongst other methods, higher order n-gram is one to tackle this problem, because it permits us to base our prediction on relatively longer history. In line with this, the results of the preliminary experiment shows up that as the order of n-gram grows there happens reduction in perplexity.

To see if further reduction in perplexity could be obtained, Tetra-gram[15] (4-gram), Pentagram (5-gram), Hexagram (6-Gram), Hepta-gram (7-Gram) language models are trained and their perplexity is evaluated on the test set sentences. The higher order N-Grams are built on the same set of parameters[16] used to train the lower order language model (see section 5.2.1) since the aim is to compare the results of these two sets of experiments.

Table 5.4 shows perplexity values of the higher order n-gram models. It turns out that Tetra-gram (4-gram) language model is the best performing model and it is slightly better when compared with the best performing lower order language model (trigram, with 256.269 perplexity value). One can also note that there is a slight increment in perplexity for LMs beyond tetra-gram.

The reason for this increment might be attributed to the very sparse nature of the data used for training. Meaning that as the length of history considered gets longer, the chance of observing that sequence in the training corpus gets to be lesser. Thus, our models will not

[15] There is no an agreed up naming convention of ngrams orders.
[16] We used same training and test data, vocabulary, and smoothing method used to train the baseline model

have robust estimates of strings. However, still there is a slight improvement when comparing the best performing lower order model (tri-gram) and the worst performing model (Hepta-gram) from the higher order LMs.

Table 5.3 Perplexity result for the higher order Amharic Language models trained on ATC-train-327.5K corpus.

Order of LMs	Test Set Perplexity	held-out Set perplexity
Tetra-gram(4-gram)	252.768	251.875
Penta-gram(5-gram)	253.888	252.971
Hexa-gram(6-Gram)	255.122	254.201
Hepta-gram(7-Gram)	256.091	255.129

In an effort to justify if the increment in perplexity really comes due to data sparseness, additional data to be used for the training and testing purposes is required. However, it was not easier to find additional corpus for this purpose. As a result, a new corpus from the existing corpora is formed by merging sentences of the training and held-out sets together to be used as a training set for the new language models.

This new training corpus is having a total of 368,458 sentences [after wards named to be ATC-368K] or 8,171,805 word tokens. Using the new larger corpus, new Tetra-gram (4-gram), Pentagram (5-gram) and Hexagram (6-Gram) language models are trained. All of these models were smoothed using Good-Turing discounting along with back-off method to make them comparable with the earlier models whose perplexity values are shown by Table 5.3. Both set of models were tested using same test set data (ATC-test-41K).

Table 5.4 Perplexity result for the higher order Amharic Language models trained on ATC-368K corpus.

NO	Order	Test Set Perplexity
1	Tetragram(4-gram)	242.624
2	Pentagram(5-gram)	243.64
3	Hexagram(6-Gram)	244.815
4	Heptagram(7-Gram)	245.732

As can be seen from 5.4, all of the models trained from ATC-368 have shown reduction in perplexity when compared with the models trained using ATC-train-327.5K corpus. Generally, the new models have shown an average of more than 4% perplexity reduction over

the previous models shown on table 5.3. On the other hand, in both set of models tetra-gram happens to be the best model. Furthermore, it is possible to see from the trend of the two set of models that there is a sort of similarity as perplexity value shows a slight increment for the models above tera-gram.

The obtained 4% perplexity reduction as a result of adding more data to our training set indicates that when the training data gets larger, there is a more likely chance that longer histories could be observed in a statistically significant manner and consequently better higher order models could be trained. Therefore, the slight perplexity increment beyond tetra-gram could possibly be attributed with the sparseness of the training corpus.

5.2.3 Experiment on smoothing Techniques

Smoothing describes those techniques used for adjusting the maximum likelihood estimate of probabilities to more accurate probabilities. These techniques tend to make distributions more uniform, by adjusting low probabilities such as zero probabilities upward, and high probabilities downward. Not only do smoothing methods generally prevent zero probabilities, but they also attempt to improve the accuracy of the model as a whole [16]. In the previous experiments above, n-grams LM trained with the Good-Turing discounting and back-off algorithms which are default to SRILM. However, as indicated by [16], there exists an enormous number of techniques that have been proposed for smoothing n-gram models.

The aim of this experiment is to investigate the relative performance of some of these smoothing techniques over our data sets. For that purpose, the smoothing algorithms are applied over Tetra-gram [4 gram] language model as it shows less perplexity value from our couple of earlier experiment (see Section 5.2.1). In this experiment, absolute discounting, Witten-Bell smoothing, modified and unmodified Kneser-Ney algorithms [see chapter 2][17] are applied. The perplexity result of the tetra-gram LM with these smoothing algorithm is shown in Table 5.5

[17] The theoretical background for all these and other smoothing techniques is presented in section 3.12.2.2 of this document.

Table 5.5 Perplexity results of tetra-gram LM with varied smoothing techniques

NO	Smoothing technique	Perplexity
1	Absolute Discount with- 0.7 discounting factor	249.966
2	Witten-Bell	246.065
3	Modified Kneser-Ney	238.74
4	Unmodified Kneser-Ney	222.564

As the experimental results show, Kneser-Ney smoothing out smarted over other smoothing techniques and our finding conforms to what [16 and 3] have reported.

5.2.4 Interpolated models

High-order and low-order n-grams have different strengths and weakness. Higher order n-grams are more sensitive to contexts. Meaning that their prediction is more reliable. However, since they consider longer history their count is sparse. On the other hand, lower order n-grams are less sensitive for contexts since they consider only short context. They have a difficulty to provide reliable prediction. But, lower order n-grams have robust counts. Thus, interpolating low-order and high-order n-grams shall give better model. And the interpolation scheme for our tetra gram models is given by:

$$P(w_i|w_{i-3}w_{i-2}w_{i-1}) = \lambda_1 P(w_i|w_{i-3}w_{i-2}w_{i-1}) + \lambda_2 P(w_i|w_{i-2}w_{i-1}) + \lambda_3 P(w_i|w_{i-1}) + \lambda_4 P(w_i)$$

Where

$$\sum_i \lambda_i = 1$$

Interpolation of n-gram probability estimates has been tried for the four smoothing techniques for which SRILM supports interpolation. Thus, four tetra-gram models (that appeared to be the best in our experiments) are trained using absolute discounting, Witten-Bell, modified and unmodified Kneser-Ney smoothing algorithms. So as to compare the performance of the interpolated models fairly, same parameters are used for all models except varying smoothing techniques. To that end, all the interpolated models are in the same order (Tetra-gram models) and the perplexity result on the test set is shown in Table 5.6.

Table 5.6 Perplexity result of a tetra-gram interpolated LM.

NO	Smoothing technique	Perplexity
1	Absolute Discount with- 0.7 discounting factor	254.315
2	Witten-Bell	244.837
3	Modified Kneser-Ney	219.499
4	Unmodified Kneser-Ney	218.863

As the result shown on Table 5.6 indicates, a Tetra-gram model with unmodified Kneser-Ney smoothing yields the best model with a perplexity value of 218.863. Furthermore, modified Kneser-Ney has also shown a significant reduction in perplexity when compared with its perplexity value without interpolation shown on 5.5. Therefore, the interpolated version of the Kneser-Ney algorithms are better than their back-off version.

On the contrary, the language models with Witten-Bell and Absolute discounting smoothing techniques did not benefit from the interpolation. Specifically, the model with absolute discounting, even, performed worse as the result of the interpolation.

5.3 Class-based language models

This section presents perplexity results of Amharic class-based language models. Class-based language models refers to those models that make use of word classes with the goal of improving their performance [32]. These models compute probability of a word given its history as a product of the probability of the word given its class and the probability of the cluster given the preceding clusters. Class-based language modeling, therefore, is a variant of word based n-gram language modeling except its assumption that similar words appear in similar contexts. Two of the classical tasks in class-based language modeling are: word clustering and computing of word probabilities. Word clustering could be done based on linguistic knowledge or just using statistical method.

For this experiment a new vocabulary that consisted of 43,000 (we refer to this vocabulary as 43K-V after wards) most frequent words (words that appeared more than 10 times in the ATC-409K corpus) is prepared and used. In other words, it is a subset of the 151K-V used for word based LMs. The only reason to use smaller vocabulary was the clustering algorithm's high time complexity. IBM clustering algorithm [8], which uses a statistical approach for

inducing classes automatically, is employed. Using clusters derived using this algorithm, LMs of varying order are trained. While these models are trained using ATC-train-327.5K training corpus, perplexity of the LMs is measured using ATC-test-41K test set that were used for word based LM models evaluation.

So as to compare[18] the class-based LMs with the standard n-gram LMs, word based n-gram LMs are trained using same data sets used for class-based models (i.e 43K-V vocabulary and ATC-train-327.5K training corpus). Both word-based and class-based models are open vocabulary models and unmodified Kneser-Ney smoothing algorithm is applied in all cases since it provided in lower perplexity models from our experiments described in section 5.2.

Table 5.7 shows perplexity results of the word and class-based models on ATC-test-41K test set. As can be noted, in all cases class-based models have showed worse perplexity than their word based counter. The results are consistent with those of [9, 32], who observed that class-based LMs show higher perplexity than word based models for English language. The relative high perplexity can be justified by the nature of class-based language model that it is more general[19] than n-gram models. Because, they do prediction of words using class sequences as history rather than word sequences.

Table 5.7 Class-based language models perplexity results

Order of LMs	**Word based LMs perplexity**	**Class-based LMs perplexity**
Bi-gram	213.953	415.243
Tri-gram	164.425	298.282
Tetra-gram	159.558	277.402
Penta-gram	158.984	274.539
Hexa-gram	158.989	274.595

Another interesting point about word and class based LMs could be the perplexity variation as the order of n-gram increases from bi-gram to hexa-gram. For example, the variation

[18] To compare two language models they must have used same vocabulary, training and evaluation resources
[19] Perplexity measures the average branching factor of a model and the history of a word is classes which is more general than words

between bigram and tri-gram models in word-based models is 49.528 where as in class-based models it is 120.456. It is same for tri-gram and tetra-gram models as well. This might shows that class-based models generalize better for higher order n-grams than word-based models. On the other hand, beyond penta-gram both word and class-based models resulted in perplexity increments.

5.3.1 Combining word-based and class-based language models

Works of [9, 11, 32, 41 and 8] showed that class-based language models perform better when combined with word-based language models. Class-based LM gives more confidence for infrequent words by relying on other more frequent words in the same class. However, class-based n-gram has problem to recognize different histories. On the other hand, n-gram models have a good capability to discriminate different contexts but have good estimate for infrequent words (i.e suffers much from data sparsity). Thus, combining these two LMs shall lead to further improvements.

The common means to combine two LMs is by interpolating them linearly and linear interpolation takes weighted sum of probabilities given by component models. Thus, in this series of experiments, the combined models of equation 5.1 is being used. The word-based and class-based LMs which are described in the section above (see section 4.2) are interpolated linearly and the linear interpolation scheme is given by:

$$\boldsymbol{P(W) = \gamma_W P_W(W) + \gamma_C P_C(W)} \qquad (5.1)$$

Where $\gamma_W + \gamma_C = 1$ and the value of $\gamma_.$ is optimized on held out data.

The value for $\gamma_.$ is optimized on held out data. Table 5.8 shows the perplexity results of bi-gram, tri-gram, tetra-gram, penta-gram and hexa-gram interpolated models. Furthermore, the perplexity results of word-based models is also included for the sake of comparison.

Table 5.8 Perplexity results of interpolated class based language models

Order of LMs	Word based LM perplexity	Interpolated model
Bi-gram	213.953	200.91
Tri-gram	164.425	147.275
Tetra-gram	159.558	141.105
Penta-gram	158.984	140.377
Hexa-gram	158.989	140.402

As can be seen from Table 5.8 it turns out that the interpolated models outsmarted both pure word-based and pure class-based models as shown above in Table 5.7. While it shows an average of 10.3% perplexity reduction over word based models, it resulted in 44 % perplexity reduction over the pure class-based models. This might shows that the two types of models (word-based and class-based LM) have different qualities and combining them would give better performance than using each individually.

Like the case in Table 5.2, the interpolated models also showed a slight perplexity increment beyond penta-gram model and that might be justified by the interpolation scheme that in interpolated models probability of a word is computed as a weighted sum of component models.

5.4 Speech Recognition Experiment

In this section, Amharic n-gram, class-based and combined language models are evaluated by putting them in to speech recognition system. The language models are used in the decoding step, when the recognizer attempt to recognize the words contained in test utterances. For that purpose, the LMs are inculcated (one at a time) into the baseline recognition system using lattice rescoring framework (see section 5.4.1 below) and the obtained results are reported accordingly.

Generally, this section presents the method employed to do word recognition experiments, the tools and resources used to build the baseline speech recognizer and word recognition experiment results. Since, the objective of this thesis is to build language models and see how

it improves the performance of speech recognizer, the details of the speech recognizer are not discussed.

5.4.1 Speech Recognition Experimentation Method

Thus far, perplexity experiments are used to evaluate the quality of the language models. For the case of word recognition experiment, lattice rescoring framework [36] is being employed. A lattice is a directed acyclic graph which contains the paths through the search space which were considered most likely by the acoustic and/or language models used in the initial decode [36]. Lattice rescoring is a mode of performing speech recognition experiments where rather than decoding the best path, set of alternative paths (n-best lists) are output in the decoding phase (Usually called N-best lists). The lattices from which the N-best lists are generated put the language and acoustic model scores separately.

Once the lattices generated from the baseline recognition, the acoustic and/or LM scores are then can be rescored/re-computed by the scores of some improved models and the best path can then be decoded finally. The primary advantage of this approach is that it is not required to compute both acoustic score and language model score if either these scores is changed. Thus, it allows us to save a lot of time. This approach is also commonly called multi-pass decoding.

5.4.2 Description of Baseline Amharic ASR

This section provides only the brief of the speech recognizer used for word recognition experiment (WRE). Specifically, the highlights of the tools, resources and methods used for building the baseline speech recognizer is given. The same resources and methods used by [4] is used with an exception of the task grammar. Thus, the detail of the recognizer can be found from [4].

Tools

HTK and Sclite are used for training and evaluation of the speech recognizer. While HTK is being used to train, generate lattices and evaluate the baseline recognizer. Sclite is used for evaluating the rescored lattice in lattice rescoring framework. Brief account of these two tools presented as follows:

i. Hidden Markov Model Toolkit (HTK)

The acoustic model for Amharic Speech Recognition System (ASRS) is built using HTK[20] [9]. HTK is a toolkit for building Hidden Markov Models (HMMs). Though it can be used to model any time series, it is specifically meant to build HMM-based speech recognizers. Generally, it comprises of two sets of tools. These can be named as training and evaluation tools.

The training tools are used to estimate the parameters of a set of HMMs from the training speech corpus and their associated transcriptions. On the other hand, the recognition tools are used to transcribe unknown utterances (test data) based on the parameters learnt in the training phase of speech recognizer development processes.

ii. Sclite

For the purpose of evaluating the performance of the speech recognizer, sclite which belongs to NIST's[21] speech recognition scoring toolkit (SCTK)[22] is used. Sclite compares the hypothesized transcription output by the speech recognizer with the correct or reference text. After comparing hypothesis to reference, statistics are collected during the scoring process and different kind of reports could be generated to summarize the performance of the recognition system.

In this research project, to analyze the performance of the speech recognizer, two basic reports are generated. The first one is called "detail report" which gives summary of sentence and word recognition performance; summary of substitution, insertion and deletion statistics and confusion pairs report.

The word recognition accuracy (WRA), typical metric to measure ASR performance, and sentence recognition accuracy (SRA) are computed as:

$$\text{WRA} = \frac{correctly\ \ recognized\ \ words}{number\ \ of\ Reference\ \ words} * 100 \qquad (5.2)$$

[20] HTK has been involved though various versions and we used the latest one (version 3.4)

[21] National Institute of Standards and Technology(NIST) provides various tools used for natural language researches such as evaluation tools (like sclite and others), language technology tools(like syllabification software and others) and corpus building tools (like speech file manipulation software)

[22] It has got different versions and we used the latest version (2.4.0) and it can be downloaded for free from www.nist.gov

$$SRA=\frac{\#correctly\ \ recognized\ \ sentences}{\#reference\ \ sentecnces}*100 \qquad (5.3)$$

Furthermore, it also calculates percentage of substituted, inserted and deleted words during recognition.

- **Substituted words:** are those reference words for which the recognizer supplied an incorrect word.
- **Inserted words**: those extra words added in the recognized sentence (falsely recognized words).
- **Deleted words**: those reference words omitted in the recognized sentence.

The other important information offered by "detail report" is the confusion pairs information. Confusion pairs are those pair of words which are being substituted one by another. This report allows us to see which words could possibly lead the recognizer to acoustic confusability. The following figure shows sample of Amharic confusion pair words.

Table 5.9 confusion pairs sample

Melaixkt --- >melixkt
Abeba --- > ixbab
Neber --- > nebere
Sewoc --- >wixcx
Sixtadiiyom --- > ietiiiem
Tadergalec --- > taqerbalec
Temare --- > kemar
Yegebre --- > yedebre

The Second type of report is the one that shows the actual alignments of the reference words against the hypothesis words. This report clearly shows the detail of correctly recognized, inserted, substituted and deleted words per each pair of sentences.

The Speech Corpus

Our baseline speech recognition system is an Amharic speaker dependent continuous speech recognition system[23]. The speech corpus used to develop the baseline speech recognition system is a read speech corpus prepared by [4]. It contains 900 recorded sentences and out of the 900 sentences 100 sentences are used as test sets while the remaining sentences are consumed for training the acoustic model. Most of the sentences in the corpus are from the news domain.

Compared to other speech corpora, for example, British National Corpus (1,500 hours of speech), that contain hundreds of hours of speech data for training, ours is obviously small in size and accordingly the models will suffer from a lack of training data. However, since our aim is to analyze the contribution of our language models in speech recognition performance improvement, it could fairly serve our purpose.

The Acoustic and Lexical Models

Acoustic model (Mono-phone and Tri-phones)

In an attempt to build the base line speech recognizer, two acoustic models namely a mono-phone and tri-phone models are built. Using mono-phone model the speech recognition engine tries to match the sound that it has heard to a single phone (single sound) and does not look the context of the phone. On the other hand, with a tri-phone acoustic model, the speech recognition engine essentially looks for a mono-phone in the context of other phones (the one immediately before and after it).

[9] stated that tri-phone models shall improve recognition performance, because the SRE is looking to match a specific sequence of 3 sounds together (tri-phones), rather than only one sound. Meaning that considering the context would reduce the possibility of error caused by confusing one sound with another, since a sequence of 3 distinct sounds are considered instead of one. It is pretty much like using a 3 word Google search rather than a single word search. Accordingly, the WRA of our mono-phone and tri-phone acoustic models are compared and it is found that the tri-phone model performed better. Therefore, tri-phone which is an intra-word model is used for subsequent experiments.

[23] It recognizes speech of a speaker whose speech is used during the development of the recognizer.

Pronunciation dictionary

Another important source of knowledge required to build SRS is pronunciation dictionary. To that end, [4] prepared 3 types of Amharic pronunciation dictionaries covering all words (3499) in the 900 sentences and they reported that the "Type III"[24]dictionary showed up the best improvement on recognition accuracy. Therefore, type III dictionary is used after the following changes are made:

- Alternative pronunciations that were not considered are added for some words. For example, all words that begin with 6th order /ር / /r/ could be pronounced as /እ ር //ixr/. For example a word /እ ር ግ ማን / which means curse could be pronounced as /ር ግ ማን / /እ ር ግ ማን /. However, still the researcher does not claim that all possible pronunciations of every word in the dictionary are considered.
- Some words having multiple pronunciations were entered as if they were different words and those words have been corrected accordingly. For Example, a word /ዩ ን ቨ ር ሲ ቲ/ which means university can be pronounced as ዩ ን ቨ ር ሲ ቲ or ዩ ን ቨ ር ስ ቲ and this word should have been kept in the dictionary in a form ፡

ዩ ን ቨ ር ሲ ቲ [ዩ ን ቨ ር ሲ ቲ] ዩ ን ቨ ር ሲ ቲ
ዩ ን ቨ ር ሲ ቲ [ዩ ን ቨ ር ሲ ቲ] ዩ ን ቨ ር ስ ቲ

Performance of the Baseline Speech Recognition System

Lattices generated from the 30 best alternatives for each sentence of the 100 test set using the HTK's HVite tool [9]. The word recognition accuracy of both the monophone and tri-phone acoustic models is indicated on Table 5.10. While the monophone model shows 47 % accuracy, the tri-phone model has got 74.52 % word recognition accuracy. Our result is in line with other findings [9,29] that tri-phone models perform better than monophone models. Thus, the tri-phone acoustic model is used as baseline for subsequent experiments in this chapter.

[24] Type III dictionary is a pronunciation dictionary that holds transcribed pronunciation of words based on acoustic evidences

Table 5.10 The baseline speech recognizer performance.

Acoustic models	**Word recognition accuracy (WRA)**
Mono-phone	47%
Tri-phone	74.52 %

5.5 Lattice rescoring experiments

Thus far, the description and intrinsic evaluation (perplexity results) of both class-based and word-based LMs is given. In this series of experiments, the word recognition accuracy (WRA) experiments of these LMs is presented in lattice rescoring framework. Consequently, four different contexts are considered. The first one is assessing WRA performance of word n-gram models, while the second one is considering class-based models. The third experiment shall consider interpolation of the two models and see if it results in WRA improvement. Because, the perplexity experiments revealed that interpolating these two types of LMs leads to further improvement.

i. Lattice rescoring with word-based n-gram models

Lattices, generated as indicated in section 4.3, have been rescored using the word-based n-gram language models and decoded using lattice-tool of SRILM toolkit to find the best path. The word based n-gram language models used for WRA experiments are described in section 5.3.1.

Table 5.12 shows the WRA experiment result of word n-gram models along with their perplexity. For the sake of comparison, the performance of the baseline speech recognition system is also shown on the same table. The baseline speech recognizer, with no language model recognized 74.52% of words given in the test utterances. As a result of the addition of the word-based LMs, a considerable WRA improvement is gained.

Table 5.11 WRA results of word based language models

Order of LMs	WRA (%)
Bi-gram	80.9
Tri-gram	80.4
Tetra-gram	80.6
Baseline ASR	74.52

It turns out that, though the bi-gram LM is the worst performing LM in terms of perplexity measure, it resulted in higher WRA accuracy. This is a good evidence that lower perplexity LM does not always guarantee higher WRA result. Language models beyond tetra-gram resulted in the same performance with that of the tetra-gram LM and that might be due to the short length sentences of the test utterances.

In a nutshell this experiment has shown that LMs are an indispensable component of speech recognition systems that plays a critical role for the improvement of speech recognizers' performance. To this end, the performance of the baseline recognizer is improved by an average of more than 6.1% as a result of the n-gram LMs addition.

ii. Lattice rescoring with class-based language models

Word recognition experiments are also carried out on class-based LMs which are presented in section 4.2 under lattice rescoring framework. IBM clustering algorithm were applied to the training corpus to determine suitable word clusters. Accordingly, 700 classes are induced automatically and these classes were then used to define a cluster-based LMs and to compute the perplexity on the test sets.

Again, the lattices generated from baseline ASR system is used. Similar to the perplexity experiment, class-based LMs have shown a degraded WRA performance than the word based LMs and the baseline recognition system as well. As can be seen from Table 5.13, the best performing class-based language model (tetra-gram) has shown only 66.0% WRA which is less than by 8.54% and 14.9% from the baseline performance and the best performing word n-gram LMs (bi-gram) respectively.

Table 5.12 WRA result of class-based LMs

Order of LMs	WRA %
Bi-gram	62.7%
Tri-gram	65.7%
Tetra-gram	66.0%
Baseline ASR	74.54

Unlike word based model where WRA drops beyond bi-gram, class based language models showed relative improvement on higher order LMs. This might shows that these models able to generalize better on large contexts or rare events.

iii. Lattice rescoring with Interpolated models

Combining word and class-based model lead to performance improvement for other languages such as Japanese and Chinese [41]. In this experiment, the researcher wanted to investigate if it leads to improvement for Amharic. Accordingly, the combined models where the class-based LMs are linearly interpolated with the word-based LMs as described by Equation 5.1 is used. The interpolation constant λ is optimized on held-out data. The results are shown in Table 5.14.

Table 5.13 WRA result of interpolated class-based LMs

Order of LMs	Perplexity	WRA %
Bi-gram	200.91	82.7
Tri-gram	147.275	82.1
Tetra-gram	141.105	82.5
Penta-gram	140.377	82.5
Baseline ASR		74.54

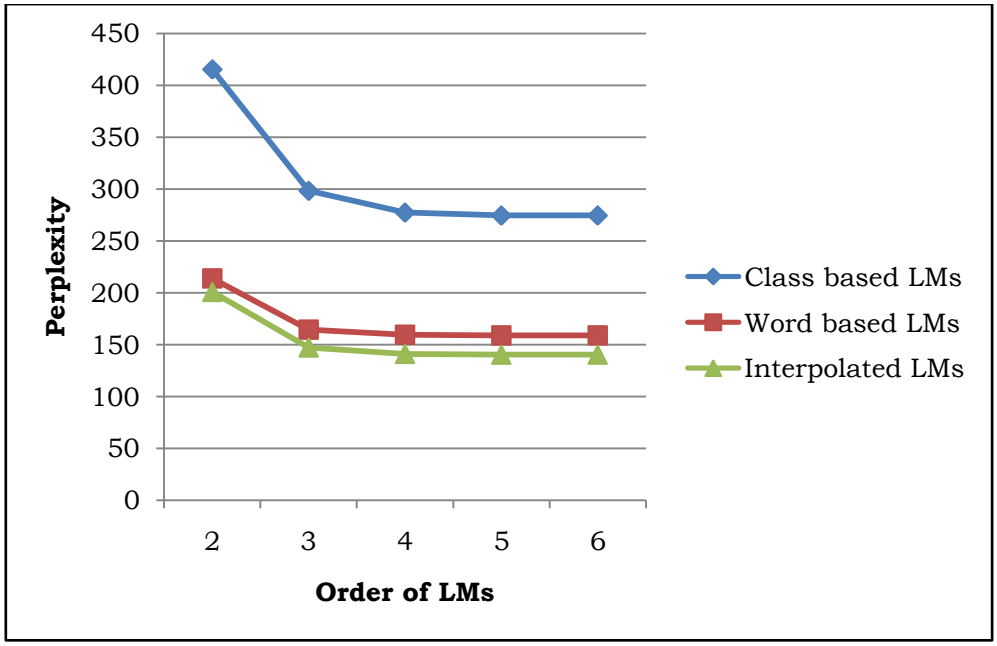
Alike the perplexity experiments, the combined models resulted in WRA improvement. As can be seen from the figure, a reasonable and consistent improvement in WRA has achieved when comparing interpolated class-based LMs with respect to word n-gram LMs and the baseline system. In terms of WRA, the interpolated models have resulted in average of 1.84 % absolute WRA gain over word n-gram model and 8.16 % over the baseline system. This clearly confirms the effectiveness of class-based LMs when combined with word based

models. Integrating the predictive power of word-based LMs, Class-based LMs can be, therefore, used to improve performance of speech recognition systems. Furthermore, it is found that our findings conforms with [18, 41].

5.6 Summary

This chapter presented perplexity and WRA results of Amharic word n-gram and class-based LMs. From the preliminary experiment the training, test and held-out data are determined. This experiment also carried out to see the perplexity of open vocabulary and closed vocabulary systems. Moreover, it is seen that lower order n-grams have higher perplexity that might arise from their short context. Higher order n-gram of order 2 up to 7 were also trained, however, tetra-gram model appeared to be the best model. Furthermore, experiment on various smoothing algorithms has been conducted and the unmodified Kneser-Ney smoothing out smarted others. Another experiment have also been conducted to see if our LMs benefit from interpolation of lower order models with higher order models. Eventually, the interpolation has improved perplexity for most cases.

On the other hand, class-based language models are trained and their perplexity result is compared with word-based n-gram LMs. IBM clustering algorithm is applied to induce classes. For all the cases class-based LMs have shown higher perplexity than the word n-gram LMs. Linear interpolation of word-based and class-based models is also tried. As it can be noted from figure 5.1, it turns out that the interpolated models consistently out smarted pure class-based and word-based models.



5.1 perplexity comparison of class-based, word n-gram and interpolated LMs

Finally, WRA experiments are carried out. Lattice rescoring approach were employed due to much flexibility it offers. Furthermore, the tools and resources used to build the baseline speech recognizer are also discussed briefly. Alike the perplexity experiment, the three types of LMs which are word n-gram, class-based and interpolation of these LMs considered were considered. Similar to the perplexity experiment, the class-based LMs alone showed poor WRA result (below the baseline performance) where as word n-gram LMs resulted in considerable improvement over the baseline speech recognizer. Moreover, combining word n-gram and class-based LMs gave higher WRA result than the individual components.

CHAPTER SIX
CONCLUSION AND RECOMMENDATIONS

This thesis has presented the development and experimental results of Amharic language models. In the previous chapters of this document, the theoretical basis of language modelling have been reviewed and as required part of building our Amharic LMs, the preparation of the Amharic text corpus (ATC) is showed. Moreover, the construction of Amharic n-gram and class-based LMs is also presented. Our main achievements have been the experimental results of our work, as shown in chapter 5. This chapter presents our conclusive remarks and recommendations for future work.

6.1 Conclusion

The primary objective of this thesis is to build Amharic language models with the aim of improving the performance of Amharic speech recognition systems. To this end, as the experimental results presented in chapter 5 showed, the objective is achieved suitably.

In the first series of experiments, various word n-gram models are trained. It is found that tetra-gram model to be the best model in terms of perplexity measure and LMs beyond tetra-gram resulted in higher perplexity. Our experimental result showed that the perplexity increment is because of the data sparsity problem. One of the measures to reduce this problem is to use smoothing technique. To this end, an experiment has been carried out to compare some of these smoothing algorithms. Even though all resulted in perplexity reduction, unmodified Kneser-Ney resulted in lowest perplexity model. Furthermore, interpolated version of some of the algorithms were also tried and while Absolute discounting and Whitten-Bell did not benefited much from the interpolation, Kneser-Ney family, however, have gained much in perplexity reduction.

Apart from n-gram LMs, class-based language models are also built using IBM clustering algorithm to induce word classes automatically. Generally, class-based language models showed higher perplexity when compared with word-based models. That is because, reducing the number of parameters makes the model coarser and thus the prediction of the next word is less precise. In order to take the advantages of the predictive power of word n-gram models and generalizing power of class-based models, the two types of models are then combined. It

turns out that the combined models provides better LMs than word based and class-based models alone.

The best way of determining the quality of LMs is to apply them to the application for which they have been designed and see if they brought performance improvement. Therefore, speech recognition experiments have been conducted in a lattice rescoring framework where the lattices are generated from the baseline speech recognizer which has 74.52% WRA. The lattices have been rescored with our word based, class-based and interpolated class-based LMs. Similar to the perplexity result, class-based LMs alone did not lead to WRA accuracy. However, it provides substantial WRA gain when combined with word-based models

To summarise, this thesis successfully demonstrated the construction of Amharic word based n-grams, class-based and interpolated LMs and the LM contribution to the performance of speech recognizer. Applying these LMs to the baseline speech recognizer which has 74.52% WRA, WRA results of 80.9%, 66.0%, and 82.7% have been achieved respectively. It means that the word n-gram LMs alone improved the baseline by 6.38%. Combining class-based models and word based n-gram LMs yields not only perplexity reduction but also WRA gain of 1.8% over the best performing word n-gram LM or 16.5% over the best performing class-based LM. Over all, by applying the best LM which is interpolated class-based LM to baseline, an absolute 8.08% WRA gain has been achieved. By this, the researcher makes a conclusive remark that building high quality LM is indispensible so as to get improved Amharic speech recognizers. On the other hand, though bi-gram LM seems to be bad compared with higher order LMs (above 2-gram) in perplexity metric, it happens to be the best model in WRA experiments. From this, it is possible to note that lower perplexity does not always guarantee improved WRA and perplexity does not correlate well with WRA measure. Moreover, the WRA experiments showed that class-based LMs have good generalizing and weak prediction power as it resulted in good WRA for longer contexts than shorter contexts. On the other hand, word-based models are good in prediction but more prone for data sparsity problem as the best performing LM is bi-gram. Therefore, the researcher makes a remark that combining these two types of LMs leads to better LMs in both perplexity and WRA results. Moreover, our experimental results showed that class-based LMs alone do not lead to both WRA and perplexity improvement. It is possible, however, to conclude that the use of class-based language models as a complementary tool (to the word based language models) is gainful for speech recognition systems.

6.2 Recommendations

There are a number of interesting directions in which to build upon the work of this research.

For the class-based LMs, hard clustering approach which oblige a word to be a member of one and only one class is applied. However, in real world one word might has multiple syntactic or semantic function thus may belong to multiple clusters. For example a word ገና /Christmas/ falls into two POS class like noun and adverb. One of the most interesting and useful extension would be to apply soft clustering where one word could belong to more than one cluster.

The other possible improvement could be clustering words using knowledge based clustering approach as only data driven approach is studied in this thesis. Though it has been reported for other languages like English POS based LMs results in a relatively higher perplexity, combining it with statistically induced classes LMs worth exploring.

Our language models are applied and tested on speech recognition task. One possible future work shall be to apply them on other NLP tasks such as machine translation and grammar checker and see their contribution for improving the tasks. Furthermore, our experiments revealed that perplexity does not always correlate well with WRA results. Thus, the other possible future work shall be to find better language modelling evaluation metric.

One of the important resources for many NLP researches is the availability of text corpus. As part of this research, a un-tagged text corpus is prepared. However, the corpus shall be more useful if it is annotated with additional linguistic information such as POS tags. POS tagging can be done either manually or automatically using tagger tools. Therefore, one possible future work would be to prepare tagged version of the corpus so that it could be used as a common resource for many NLP researches on Amharic language. Moreover, making it accessible for the research community through the Internet is also important.

Finally, the overall effort towards the development of Amharic LM is very limited compared to other languages. Therefore, exploring other language modelling approaches such as neural network language modelling [31] and others reviewed by [33] shall also be studied.

REFERENCES

[1]. E. Arisoy, "Statistical and discriminative language modeling for Turkish large vocabulary continuous speech recognition," Ph.D. dissertation, Dept. Elect. Eng., Bogazici Univ., 2009.

[2]. T. Kaufman, "A Rule-based Language Model for Speech Recognition," Ph.D. dissertation Dept. Informatik-Ing., Swiss Federal Institute Of Technology, Zurich, 2009.

[3]. M. Y. Tachbelie, "Morphology-Based Language Modeling for Amharic," Ph.D dissertation, dept. informatics, Universität Hamburg, 2010.

[4]. A. W. Zewoudie, " Enhanced Amharic Speech Recognition Systems," M.S. thesis, Dept. Computer Science, Addis Ababa Univ., 2011

[5]. Population census commission, "Summary and statistical report of the 2007 population and housing census", Addis Ababa, Ethiopia, 2008.

[6]. S. T. Abate, M. Y. Tachbelie, W. Menzel, "Amharic Speech Recognition: Past, Present and Future, " Proc. of the 16th International Conference of Ethiopian Studies, Trondheim, 2009, PP. 1391 - 1401

[7]. S. T. Abate, " Automatic Speech Recognition for Amharic," Ph.D dissertation, dept. informatics, Universität Hamburg, 2005.

[8]. P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. D. Pietra, J. C. Lai "Class-Based n-gram Models of Natural Language," Computational Linguistics, Volume 18, Issue 4, PP 467-479, December 1992.

[9]. S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D., Ollason, D. Povey, V. Valtchev, P. Woodland, The HTK book, Version 3.4. Cambridge University Engineering Department, March 2009.

[10]. A. Ramanathan, "Language Modeling for Information Retrieval," dissertation annual progress seminar report, dept. Comp. Sc., and Elect. Eng., IIT, Bombay, 2003.

[11]. E.W.D.Whittaker, P.C.Woodland, "Comparison of language modeling techniques for Russian and English," Proc. Int. Conf. on Spoken Language Processing, 1998

[12]. K. C. Sim, "Structured Precision Matrix Modeling for Speech Recognition," Ph.D Dissertation, submitted to the University of Cambridge for the degree of Doctor of Philosophy, 2006.

[13]. X. Huang and L. Deng, "Handbook of Natural Language Processing" Microsoft Corporation, 2007.

[14]. C. D. Manning and H. Schuatze, Foundations of statistical natural language processing, The MIT press, Cambridge, London, 2000.

[15]. D. Jurafsky and J. H. Martin, Speech and Language Processing: An introduction to natural language processing, computational linguistics and speech recognition, 2007.

[16]. S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," Harvard University: Computer Science Group, Rep. TR-10-98, 1998.

[17]. T. Gandhi, V. K. Antiwal, A. K. Jain, "Evaluation of Smoothed Language Models," International Journal of Research and Reviews in Computer Science (IJRRCS), Vol. 2, No. 2, April 2011.

[18]. E. W. D. Whittaker, "Statistical Language Modeling for Automatic Speech Recognition of Russian and English," PhD Dissertation, Cambridge, 1998.
[19]. J. Uszkoreit, T. Brants, "Distributed Word Clustering for Large Scale Class-Based Language Modeling in Machine Translation," in Proceedings of Association for Computational Linguistics-08:HLT, USA: Columbus, Ohio, PP. 755–762, June 2008.
[20]. J. T. Goodman, "A Bit of Progress in Language Modeling," Microsoft Research, Machine Learning and Applied Statistics Group: Redmond, Rep. MSR-TR-2001-72, 2008.
[21]. R. M. Iyer and M. Ostendorf, "Modeling Long Distance Dependence in Language: Topic Mixtures vs. Dynamic Cache Models," IEEE Transactions on speech and Audio processing, 7, January 1999.
[22]. Y. Gotoh and S. Renals, "Topic-based mixture language modeling," Natural Language Engineering, pp. 355-375, 2000
[23]. R. Kneser and V. Steinbiss, "On the Dynamic Adaptation of Stochastic LM," Proc. Int'l Conf. on Acoust., Speech and Signal Proc., Vol. 2, pp. 586-589, 1993.
[24]. R. Iyer, M. Ostendorf and J. R. Rohlicek, "Language Modeling with Sentence-Level Mixtures," in Proc. HLT '94, PP. 82-87, 1994
[25]. J. A. Bilmes and K. Kirchhoff, "Factored language models and generalized parallel back-off," in Proc. of HLT/NAACL, PP 4–6, 2003.
[26]. K. Kirchhoff, J. Bilmes, K. Duh, " Factored Language Models Tutorial," Seattle : Dept of EE, University of Washington, Rep. UWEETR-2008-0004, February 2008.
[27]. A. E. Axelrod, "Factored Language Models for Statistical Machine Translation," MSc Thesis, Division of Informatics, University of Edinburgh, 2006
[28]. M. Creutz, K. Lagus, "Unsupervised morpheme segmentation and morphology induction from text corpora using morfessor," Neural Networks Research Center, Helsinki University of Technology, Tech. Rep. A81, 2006.
[29]. E. Gouws, "Appropraite baseline values for HMM-based speech recognition," Proc. of the 15th Annual Symposium of the Pattern Recognition Association of South Africa, Grabouw, South Africa, November 2004.
[30]. D. Guthrie, B. Allison, W. Liu, L. Gutherie, and Y. Wilks, "A closer look at skip-gram modeling," In proceedings of the Fifth international conference on Language resources and Evaluation (LREC), 2006.
[31]. G. W. Blackwood, "Neural Network-based Language Model for Conversational Telephone Speech Recognition," M. Phil thesis, St. Catherine's College, 2005.
[32]. G. Maltese, P. Bravetti, H. Crepy, B. J. Grainger, M. Herzog, F. Palou, "Combining Word and Class-based language models: A comparative study in several languages using automatic and manual word-clustering techniques", INTERSPEECH, page 21-24. ISCA, 2001.
[33]. R. Rosenfeld, "Two decades of statistical language modeling: where do we go from here?," Proc. Of the IEEE, 88:1270-1278, August 2000.
[34]. J. R. Deller, Jr., John G. Proakis, J. H. L. Hansen, Discrete-time processing of speech signals, A JOHN WILEY AND SONS, INC, PUBLICATION, 2000

[35]. M. Moftah, W. Fakhr, S. Abdou and M. Rashwan, "Stem-based Arabic Language Models Experiments," Proceedings of the second international conference on Arabic language resources and tools, 2009.
[36]. P. R. Clarckson, "Adaptation of Statistical Language Models for Automatic Speech Recognition," PhD Dissertation, Engineering Department, university of Cambridge, 1998.
[37]. K. Kirchhoff, D. Vergyri, J. Bilmes, Kevin Duh, A. Stolcke "Morphology based language modeling for conversational Arabic Speech recognition," Computer Speech & Language 20.4 (2006): 589-608.
[38]. A. Stolcke,, "SRILM-an extensible language modeling toolkit,"Proceedings of the international conference on spoken language processing, Vol. 2. 2002.
[39]. A. Stolcke, J. Zheng, W. Wang, and Victor Abrash, "SRILM at Sixteen: Update and Outlook," in proceedings IEEE Automatic Speech Recognition and Understanding Workshop, IEEE SPS, December 2011.
[40]. W. Wang, "Statistical Parsing and Language modeling based on constraint dependency grammar," PhD dissertation, Purdue University, December 2003
[41]. J. Gao, J. T. Goodman, J. Mio, "The use of clustering techniques for language modeling – Application to Asian Languages," Computational Linguistics and Chinese Language Processing 6.1 (2001): pp 27-60
[42]. J. R. Bellegarda, "Statistical language model adaptation: review and perspectives," Speech Communication, pp 93–108, 2003
[43]. A. Kellner, "Initial language models for spoken dialogue systems," In: Proc. 1998 Internat. Conf. Acoust. Speech Signal Process., Vol. 1. Seattle, WA, May 1998.
[44]. K. Seymore, R. Rosenfeld, "Using story topics for language model adaptation," In: Proc. 1997 Euro. Conf. Speech Comm. Technol., Vol. 4. Rhodes, Greece, September 1997.
[45]. T. Rose, N. Haddock, "the effect of corpus size and homogeneity on language model quality," Interaction Technology Department, HP Laboratories Bristol, HPL-97-70, May 1997.
[46]. H. Seid, B. Gambdck, "A Speaker Independent Continuous Speech Recognizer for Amharic," INTERSPEECH, 2005 .
[47]. Z. Seyifu, "Large vocabulary, speaker independent, continuous Amharic speech recognition," M.Sc Thesis, Addis Ababa University Faculty of Informatics, 2003.
[48]. S. Atkins, J Clear and N Ostler, "Corpus Design Criteria," Literary and linguistic computing 7.1 ,1992, pp. 1-16
[49]. K. Tadesse, "Sub-word based Amharic speech recognizer: An experiment using Hidden Markov Model (HMM)," MSc Thesis, School of Information Studies for Africa, Addis Ababa University, Ethiopia, June 2002.
[50]. P. Geutner, "using morphology towards better large-vocabulary speech recognition system," ICASSP-95., Vol. 1, PP. 445-448, IEEE, 1995.
[51]. S. Berhanu, "Isolated Amharic Consonant-Vowel syllable recognition: An Experiment using the Hidden Markov Model," Msc Thesis, School of Information Studies for Africa, Addis Ababa University, Ethiopia, 2001.

[52]. Molalgne Girmaw' "An Automatic Speech Recognition System for Amharic," MSc Thesis, Dept. of Signals, Sensors and Systems, Royal Institute of Technology, Stockholm, Sweden, 2004.
[53]. Martha Yifiru, "Automatic Amharic Speech Recognition System to Command and Control Computers," MSc Thesis, School of Information Studies for Africa, Addis Ababa University, Ethiopia, 2003.
[54]. S. H/Mariam , S P Kishore, S.P. Alan W Black , Rohit Kumar and R. Sangal, "Unit Selection Voice for Amharic Using FESTVOX," In Fifth ISCA workshop on Speech Synthesis, Pittsburgh, USA, 2004.

APPENDIX A: TRANSLITERATION MAIN FUNCTION

```
## the main function where execution begins from
if __name__ == '__main__':
   ##input file
   rfile=open(r'C:\Documentsand Settings\user\My Documents\thesis\model\corpus.txt','r')
   input_file=rfile.readlines()
   ##translitrated file
   translitrated_file = open(r'C:\Documents and Settings\user\My
Documents\thesis\model\transed.txt','w', 0)

   #process each line in the input file
   for iline in input_file:
      iline = iline[:-1]
      iline = iline.strip()
      words = iline.split(" ")
    ## remove any blank spaces from the list
      while ("" in words):
         words.remove("")
    ## process word by word
      for word in words:
         count = 0
         word=word.split()
         myword=word[0]
        ## create character list from the word
         chars = list(myword)
         newword = ""
         if isnumeric(myword):
            print str(myword)+'  it is pure number'
            newword = int2word(myword) +  ' '
            translitrated_file.write(newword)
         elif isnumeric(chars[0]):
            #call function num_followecby_text(chars)
            print chars
            num_followecby_text(chars)
         elif isnumeric(chars[-1]):
            #call function text_followecby_num(chars)
            text_followecby_num(chars)
         else:
            textpart=alphabet(myword,count,newword)
            translitrated_file.write(textpart)
      translitrated_file.write('\n')
```

APPENDIX B: TRANSLITERATION SCHEME

(Adopted from Sebsbie *et al.* (2004) work)

Amharic symbol	Etop font	Amharic Word Example	
ፕ	p	/profieser/	professor'
ት	t	/tixmhixrt/	'education'
ክ	k	/kixb/	'circle'
ዕ	ax	/axda/	'credit'
ብ	b	/bixrr/	'Ethiopian Currency'
ድ	d	/dixmet/	'cat'
ግ	g	/gixl/	'private'
ጵ	px	/pxapxas/	'bishop'
ጥ	tx	/txixtx/	'cotton'
ጭ	cx	/cxixra/	'tail'
ቅ	q	/qixlliet/	'scandal'
ፍ	f	/fixndata/	'explosion'
ስ	s	/sixm/	'name'
ሽ	sx	/sxixnkurt/	'onion'
ህ	h	/hixnd/	'India'
ጽ	xx	/xxixnu/	'Determined'
ች	c	/cxixgixnx/	'seedling'
ጅ	j	/jixb/	'hyena'
ም	m	/mixn/	'what'
ን	n	/nixb/	'bee'
ኝ	nx	/monx/	'fool'
ል	l	/lixb/	'heart'
ር	r	/rixhruh/	'merciful'
ይ	y	/ayn /	'eye'
ው	w	/wixsxa/	'dog'
ቭ	v	/nerv/	'nerve'
ዝ	z	/zixnb/	'fly'
ዥ	zx	/zxereggege/	"stripped off"
አ	e	/deheye/	'he become poorer'
ኡ	u	/mulu/	'full '
ኢ	ii	/ iityopxya/	'Ethiopia'
ኣ	a	/arat/	'Four'
ኤ	ie	/meriet/	Land'
እ	ix	/ixnat/	'Mother'
ኦ	o	/bota/	'Land'

APPENDIX C: SRILM CONFIGURATION NOTES

- Download SRILM and put it in convenient directory
- extract it in to srilm folder (create srilm folder by your own)
- install all the required tools by SRILM by executing the following command

  sudo apt- get install g++ make gawk gzip tcl8.4 tcl8.4- dev csh
- Take backup of Makefile in srilm folder by the folliwng command

  cp $SRILM/Makefile $SRILM/Makefile.bak
- open Makefile like using gedit

  gedit srilm/Makefile
- add these two lines at top of the page

  SRILM=the absolute path to srilm folder (eg /home/mulie/thesis/tools/srilm)

  MACHINE_TYPE=i686 (depends on your machine; check using "$uname –m")
- We may also need to modify the machine-specific makefile

  cp ~/common/Makefile.machine.i686 ~/common/Makefile.machine.i686.bak

  gedit $SRILM/common/Makefile.machine.i686
- Look for CC and replace with the following:

  CC = /usr/bin/gcc$(GCC_FLAGS)
  CXX = /usr/bin/g++$(GCC_FLAGS)- DINSTANTIATE_TEMPLATES
- Look for TCL_INCLUDE and replace with the following:

  TCL_INCLUDE = - I/usr/include/tcl8.4
  TCL_LIBRARY = -L/usr/lib/tcl8.4
- now check if everything is working fine

  sudo make
- If no errors appeared, then we can proceed with the installation

  sudo make World
- add directories to the PATH variable

exportPATH=/home/mulie/thesis/tools/srilm/bin:/home/mulie/thesis/tools/srilm/bin/i686:$PATH

- add MANPATH (used for accessing the manual pages)
  export MANPATH=/home/mulie/thesis/tools/srilm/man:$MANPATH
- now test if every thin is fine

  make test
- this take some time and if you see IDENTICAL and few DIFFERS prove srilm compiled a success!

- Clean up files that we do not need anymore

  make cleanest

- Reference (this note is prepared using SRILM Install note as main reference)

APPENDIX D: SAMPLE AUTOMATIC WORD GROUPING

class 5
betekahiede, betekahiedew,betekahiedewu, bakahiedew,bakahiedewu, bakahiedut, bakehiedut, bazegajecw, bazegajew, bazegajewu, bazegajut, bekahiedut, bemekerew, bemiikahied, bemiikeberew, bemiikeberewu, bemiikefetew, bemiikenawenew, bemiimekrew, bemiinegagerew, bemiitxenaqeqew, bemiiyakahiedut, bemiiyakahiidew, bemiiyazegajew, bemiizegajew, betejemerew, betejemerewna, betekeberew, betekeberewu, betekefetew, betekenawene, betekenawenew, betekenawenewu, betetxeraw, betezegaje, betezegajew, betezegajewu, betxeraw, lemiikahied, lemiikeberew, lemiimeseretew, yakahiedewn, yakahiedutn, yemiikeberewn, yetekahiedewn, yetekeberewn, yetekefetewn,

maleda, msxt, txewat, txuat

class 7
yemabazat, yemaberetatat,yemabqat,yemadaber,yemadaqel, yemadares, yemaderajet, yemadleb,yemagolbet,yemalamed, yemalmat, yemamecacet, yemamualat, yemanesasat,yemaqreb,,yemaquaquam, yemarabat, yemaregagat,yemasadeg, yemasadegu, yemasatef, yemasaweq, yemascal,yemasefafat, yemaseltxen, yemasemarat, yemaseracxet,,yemasfafat, yemastekakel, yemastewaweq, yemasxasxal,yemedegef,yemegenbat

class 8

sera, sra, sracew, sram, sraw, srawu

class 11

arb, diey, erb, hamus, hemus, ixhud, ixhudn, ixrob, kremt, maksenxo, qdamie, rebuix, rob, samnet, samnt, semon, senxo, wer, werm

class 550

arebiiya, balie, beborenana, behoro, belesa bemekakelenxawna, bienc, butajra, debubawiina,,denbel, driedawana, ebaya, efar, efarna, efriika, efriikawiitua,,efriikawiiw, egamasxna, emara, emarana, erebiiya, ermacxho, ermacxhona, ersiina, etlantiik, faso, gambiela, gambielana,, gedariif geletie, giera,,gietxie, godie, gojamna, gondern, gonderna, gorie, hegen, hegenu, hererii, hereriina, iesya, iilubabor,, iitang, ixsiiya ixstie, ixsya, ixsyana, johansberg,,koriiya, koriiyan, koriiyana, majiina, mekakelenxawna, miilan, mixrabna, mseraq, msraq, msraqna, negelie, negielie, nejo, oromiiyan, oromiiyana, oromo,,pasfiik, prayz, riijiin, riiyad, saynt, serawiitacn, serawiitna, sltxie, somalie, somaliena, sumalie, sxebelie, terarawoc, teraroc, tesxagrew, tgray, tgrayna,,txeref, wcxalie, welegana, wetxere, yeewstraliiyana, yegambielana, yemekakelenxawna

class 548

balebietu, balemoyaw, balemoyawu balemuyaw,, balemuyawa, balemuyawu, danxaw, danxocu, dayriekterua, diinu, dokteru, driekteru, efegubaiew, efegubaiewa, efegubaiewu, emakariiw, emakariiwa, embasaderu, embasaderua, emerarocu, eseltxanxu, esetebabariiw, estebabariiw, estebabariiwu, estebaberiiw, estedadariiwocu, ezaZu,,ezegajocu, fexxamiiw, halafew, halafiiw, halafiiwa, halafiiwocu, halafiiwu, halefiiw, halefiiwu, helafiiw, helafiiwa, helafiiwocu, helafiiwu, iekspertu,,ixndeembasaderu, ixndehalafiiw, jienieralu, kebalemuyaw, kehalafiiw,, kentiibaw, kentiibawu, komanderu, komiisxneru,, komiisxnerua, liiqemenberua,,megenzebacewn, mehendiisu, mekonnu, merejawocu, mhuru, miiniisetru, miiniisteru, miiniistru, miiniistrua, miinstru, mncxocu, mskrocu, nxeriezedantu,,nxerieziidantu, ofiiseru, patriiyarku, qdusnetacew, patryarku,, bxxuixnetacew, preziidantu, priezdantu, priezdentu, priezedantu, prieziedantu,,prieziidantu, prieziidantua, prieziidentu, profieseru, sayntiistu, sebsabiiwa, sraeskiiyaju, temeramariiw, temeramariiwocu, tesfacewn, tetxeriiw,,tetxeriiwa, tewekayocu, tewekaywa

Class 322

bealemacn, bedega, beegeracn, beegeriitu, beegeriitua, beegerua, beehguriitu, beehguriitua, beehguru, beekababiiwocu, beekababiiyacn, beelemacn, begedamu, begzatiitu,,behageracew, behageracn, behageriitu, behegeracew, behegeracn, behegeriitu, behegeriitua, behegeru,, beheyqocu, beheyqu, behospiitaloc, beierportu,,beiikonomiiyacew, bekampu, beketemacn, beketemawa, beketemaytu, beketemocu, beklelu, bekllocu, bemediinawa, bemediinaytu, bemekakelacew, bemekakelenxana,,bemesmeru, bemoquadiisxo,

bemuziiyemu, beparku, beprieziidant, beqebeliew, berkatocu, besxampiiyonaw, betxabiiyaw, bewenzu, beweredawa, beweredawoc,,bewstxua, beyeegeratu, beyeegerocacew, beyekllocu

class 317

beixqd, betaqedew, ekefafay, eqd, gbn, hlmacewn, ixqd, ixqdm, ixqdn, ixqdna, ixqdoc, ixqdocacewn, ixqdocn, ixqdocun, ixqdu, ixqdun, ixqed, leixqdu, poliisiiwocu,,raixyacewn, raixyun, stratiejiiwn

class 309

baleend, bebzu,, beemset, beemst, beemstu, beend, beerat, beesr, behulet, beixreft, bemecxeresxaw, besdset, besdst, besdstu, besebat,, besebatu besmnt, besmntenxaw,,besoset, besost, besostenxaw, bezetxenx, bezetxenxu, ixskeasraarat, ixskeixkule, keyetnxawm, leend, legmasx, lehuletenxaw, lemejemeriiyaw, snt, yeemset,,yeemst, yeemstu, yeend, yeerat, yeesr, yegebabet, yegmasx, yehulet, yemulu, yerub, yesdst, yesebat, yesmnt, yesoset, yesost, yesostenxaw, yexxom, yezetxenx,wtxn, yeixqd

class 314

karie, karii, kekarie, kekiilo, kiilo, kiiyubiik, miilii, santii, sientii, skuyer, skuyier

class 306

emektewal, emelektewal, esmrewbetal, estxenqqewal, ewstewal, gelexxec, gelexxewal, gelxxalec, gelxxewal, txequmewal

CLASS233 233 47 IN
basalefnew, begnbotu,bemecxew, bemecxiiw, bemecxiiwu, bemecxwu, bemejemeriiyaw, bemejemeriiyawu, beqeriiw, beqetxay, beqetxayu, betetxenaqeqew, betetxenaqeqewu,beteyazew, bezenderow, bezendro, bezendrow, bezendrowu, ixskemecxiiw, ixskemiiqetxlew, kefiitacn, kemecxiiw, kemecxiiwu, kemecxw, keteyazew, keyaznewu, kezendro, kezendrow, lemiiqetxlew, leyaznew, lezendro, lezendrow, msretana, wedemiiqetxlew, yalefew, yefiitacn, yemecxiiw, yeqetxayu, yeqetxayun,yetetxenaqeqewn, yeyaznew, yezenderow, yezendro, yezendrow, yezendrown, yezendrowu, yezendrowun

class 154

baybalm, bayoc, bemalet, bemaletacew, bemaletm, biibal, biilum, bla, blachu, blen, blew, blewu, bye, byie, eluna, ixgelie, ixndemalet, ixndemiibalew, ixyalku, ixyaln, kemalet, kemiilew, lemalet, lemiilew, maganen, malet, maletacewnm, maletie, maletu, melixktacn, melsu, miistxiir, siilu, siilum, sl, stl, stlm, teblew, teblewu, yalenx,,yemiibalewu, yemiilew, yemiilewu, yemiilut, yemiilutn, yemiimeslew, yemnlew, yemtlew, yetebalena

APPENDIX E: CONDITIONS ON THE USE OF ATC-409K CORPUS

ATC-409 is an Amharic Text corpus which is prepared from the 7 years (2003 to 2009) of news archive obtained from the Ethiopia News Agency (ENA). The corpus consisted of a total of 409, 398 lines and 9,079,766 tokens. The text is obtained from the owner (ENA) under the condition stated below.

- The text shall only be used for the purpose of academic research and not for commercial purpose or any other non academic use.

Therefore, any form of usage of this corpus at odds with the condition stated above shall be considered to be illegal.

Declaration

I, the undersigned, declare that this thesis is my original work and has not been presented for a degree in any other university, and that all source of materials used for the thesis have been duly acknowledged.

Declared by:

Name: Mulugeta Mekonnen

Signature: _____

Date: _____

Confirmed by advisor:

Name: Sebsibe Hailemariam (PhD)

Signature: _____

Date: _____

Place and date of submission:
Addis Ababa University, Addis Ababa, March, 2013.