ENTROPY ESTIMATION AND ENTROPY BASED ENCODING OF WRITTEN AFAAN OROMO FOR ITS EFFICIENT DIGITAL TRANSMISSION AND STORAGE

A Thesis Submitted to:

Addis Ababa University

Addis Ababa Institute of Technology

School of Electrical and Computer Engineering

In Partial Fulfillment of the Requirements for the Degree of

Masters of Science in Electrical Engineering

(Electronic Communication Engineering)

By:

Kalkidan Dejene Molla

Adviser:

Dr. Ing. Dereje Hailemariam

February, 22, 2021

Declaration

I declare that this thesis titled, "***Entropy Estimation and Entropy Based Encoding of Written Afan Oromo for its Efficient Digital Transmission and Storage.***", is my own work and all sources of materials used for the thesis have been clearly stated and attributed.

Student Name: Kalkidan Dejenie

Signature: ———————————-

Date of Submission: ———————————

This thesis has been submitted for examination with my approval as a university adviser.

Supervisor: Dr. Ing. Dereje Hailemariam

Signature: ———————————

Acknowledgments

First and foremost, I would like to praise the almighty for giving me endurance and persistence. Secondly, my special thanks also go to my advisor, Dr. Ing. Dereje Hailemariam, for his initial inspiration with the topic of this thesis, careful guidance, constant support and sustained encouragement. I consider working with him to be one of my best opportunities. I have learned many things, not only academic knowledge, but also methodologies of scientific research; all these, for sure, will benefit me throughout my life and career.

Furthermore, I would like to extend my thanks to Mr. Tsegamlak Terefe for sharing resources, guidance, unreserved critical comment and constructive moral value from the inception of my research proposal up to this end. I would also like to extend my heart felt appreciation and acknowledgement for Dr. Feda Negesse who supported me to get Afan Oromo corpus from Oslo Text Laboratory. Furthermore, he has advised me on how to get the corpus from the laboratory easily. Additionally, I am also indebted to extend my acknowledgment to Mr. Bekele Gerba, who supported me by providing the original copy of his book which was used in entropy extraction. Furthermore, he also facilitated the link with other authors to get additional book. In this regard, I would also want to extend my gratitude to Mr. Alemu Yadessa who also provided the original copy of his book and for providing an access to data from "Chefe Oromia".

Last but not least, my heartfelt appreciation goes to my father Mr. Dejenie Molla and my mother W/zro Haymanot Addisu who have always prayerfully supported me in whatever challenge I have taken up in my life. You have been undying stream of motivation. Finally, I take the opportunity to express my very sincere thanks to my sisters, brother, teachers, colleagues and friends and to whom I am not able to mention in the name list.

Abstract

According to Ethiopian population census, Oromo Language is estimated to be spoken by 36.4% of the local population. Furthermore, in addition to the local population the language is spoken outside of Ethiopia, for instance in small portion of Kenya. Thus, taking this into account the language is estimated to be spoken by around fifty million people. In addition to the spoken form, a considerable portion of the language's speaker are capable of understanding its written form known as Qubee. The introduction of Qubee, in the mid-nineties has opened doors for its utilization in modern day communication systems. Leaving this argument aside, in the eyes of information theory and communication channels both symbol utilization schemes are found to be inefficient. This is because, Latin or Amharic symbols are represented by ASCII8 and UTF 16 fixed length encoding mechanisms poorly model written natural language.

With the expected increasing demand of the language in telecom services in mind, in this thesis we mainly aim at estimating the Oromo Language Language's entropy. The estimation will set the optimum number of bits per symbol needed to efficiently transmit written Oromo Language in communication systems. To achieve our objective, we have modeled the sources, i.e., written Oromo Language, as N^{th} order Markovian chain random process. Based on the modeling scheme we have studied the distribution of symbols in ten literature written in Oromo Language. The study reveals the Language can be transmitted using 4.31 bits/symbol when modeled as first order Markovian Chain source. Whereas, the zero crossing entropy of the source was estimated to be in average at N=19.5; which gave an entropy estimation of 0.85 bits/symbol with a redundancy of 89.36%. Additionally, we have conducted two entropy-based compression algorithms, namely, Huffman and Arithmetic coding, to test the validity of our estimation. The Huffman algorithm was able to compress our sample corpora in average from $42.17\% - 64.88\%$ for $N = 1 - 5$. These compression results confirm the results of our N^{th} order estimation of the Language's entropy by approaching their theoretical limits.

Key Words: *Entropy, Encoding, Oromo Language, Language.*

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

- **ASCII**: American Standard Coding for Information Interchange.
- **DMS**: Discreet Memoryless Source.
- **IPA**: International Phonetic Writing.
- **NLP**: Natural Language Processing.
- **SLM**: Statistical Language Modeling.
- **SMS**: Short Message Service.
- **SOV**: Subject Object Verb.
- **SVO**: Subject Verb Object.
- **UTF**: Universal Transformation Format.

CHAPTER 1

INTRODUCTION

In communication systems, information theory focuses on obtaining the optimal representation of source symbols in terms of bits [1]. Here, optimality implies bit/symbol allocation below which data loss will occur and above which efficient resources utilization is undermined. In this context, information theory focuses on estimating the average optimum bits needed by source symbols [2]. Following this measurement, information theory utilizes the process of compression, which is also known as source encoding, to achieve the estimated optimum bits allocation. Thus, this way information theory tries to addresses the two fundamental questions of communication theory [3], namely,

1. What is the ultimate achievable data compression for a given discreet source?
2. What is the ultimate transmission rate of a communication channel?

According to Claude Shannon the answer to the first question lies with the concept of source entropy. Whereas, the second question is answered by defining the channel capacity, which could be considered as the channels entropy [4].

In the field of Thermodynamics, the term entropy is defined as the degree of disorder associated with particles within a given system. Bringing this concept to communication systems, entropy is related to the degree of randomness associated with a sources symbol generation. To this end, [4] acknowledges that a more random source is highly likely provides extra information on each symbol generation. Thus, the source requires more bits per symbol to precisely define unique symbols and combinations of symbols. Comparatively, a less random source which is more or less identified by generating repetitive symbols is predictable. Hence, source requires a lesser amount of bits per symbol to represent the generated symbols or their combinations. Shannon also extended this concept to channels in [4]; in the paper channel capacity was defined

as the maximum data rate that is reliably supported by a communication channel. Thus, most communication system designs aim at achieving the channel capacity for enhanced service provision.

In this context, information theory suggests means of achieving the theoretical limits by systematically allocating bits to source symbols. This is because, if a communication system is expected to transmit at channel capacity, then it should transmit the right number of bits per unit of time. This, on the other hand, requires allocating the optimum number of bits per source symbol. To this end, information theory introduces data compression, which is used to represent signals (or more general data) in a more efficient form by removing redundancy in data. Shannon in [4] acknowledged that achieving this objective requires answering the below three critical questions.

1. How is information source modeled?
2. How is the source measured?
3. How many bits/symbol is enough?

The first two questions were answered by Shannon in [5]; accordingly, the paper states that most discreet sources can be modeled as a random process, where, the type of process depends on the source at hand. For instance, the paper showed that written English can be modeled as a Markov Chain random process. On the other hand, the paper also showed that such kind of sources can be measured by their entropy. Finally, Huffman, a student of Shannon, showed that there could be source encoding mechanisms that could closely follow the estimated entropy [6].

With this understanding, in this thesis, we mainly aim at utilizing information theory to estimate the entropy of written Oromo Language language. We strongly believe the current fixed source coding mechanism of the language, i.e., ASCII8, is not efficient to represent the Language given the redundancy observed in Oromo Language word formation. Thus, we believe by studying the Language's entropy we could show that it can be transmitted in telecom networks in a cost efficient manner. This is expected to impact a considerable portion of the population residing in Ethiopia.

Furthermore, we aim at conforming our findings by utilizing entropy based compression techniques, mainly Huffman and Arithmetic codings. We choose these mechanisms since theoretically they are known to achieve compression results near to the entropy. To the best of our understanding, the work done in this thesis is first of its kinds for the language.

1.1 Oromo Language Language

Written Oromo Language (Qubee) is defined using Latin based alphabet that consists of thirty-three basic symbols. Out of these thirty-three symbols, five are vowels, twenty-four are consonants, out of which seven are paired letters (a combination of two consonant characters such as 'CH', 'DH', 'NY', 'SH', 'TS'). The written Oromo Language, like its English counterpart, utilizes capital and small letters, the vowels are sound makers and are sound by themselves, as is in English. However, Afaan Oromo has a considerable amount of glottal stops. In this regard, an apostrophe, and less commonly a hyphen, is used to represent these sounds in writing. Furthermore, sometimes an 'H', which represents the closest glottal sound is also used in place of an apostrophe. For a reason to be apparent later, the apostrophe will be considered as a distinct symbol (say, as space is considered as the 27^{th} letter of written English). Additionally, in Afaan Oromo writing system, geminated consonants and long vowels are represented by double letters in addition to seven compound symbols (6 Qubee). However, it should be noted that not all the 26 letters correspond with their English sound representation. This is shown in Table 1.1 [7].

Table 1.1: The Qubee symbols in international phonetic writing (IPA) [7].

Qubee		IPA	Qubee		IPA	Qubee		IPA
A	a	/a/	L	l	/l/	W	w	/w/
B	b	/b/	M	m	/m/	X	x	/t'/
C	c	/č'/	N	n	/n/	W	w	/w/
D	d	/d/	O	o	/o/	Z	z	/z/
E	e	/e/	P	p	/p/	CH	ch	/č/
F	f	/f/	Q	q	/k'/	DH	dh	/d/
G	g	/g/	R	r	/r/	NY	ny	/ñ/
H	h	/h/	S	s	/s/	PH	ph	/p'/
I	i	/i/	T	t	/t/	SH	sh	/š/
J	j	/ǧ/	U	u	/u/	TS	ts	/s'/
K	k	/k/	V	v	/v/	ZH	zh	/ž/

The complete list of the Language's alphabets is shown in Figure 1·1, the basic

symbol set does not contain ‘p’, ’v’ and ‘z’, because there are no native words in the languages morphology that are formed from these characters. However, in writing Oromo Language Language, they are used to refer to foreign words such as “polisii” (“police”) [8].

A a	B b	C c	Ch ch	D d	Dh dh	E e	F f	G g
a	ba	ca	cha	da	dha	e	fa	ga
[ɐ]	[b]	[tʃʼ]	[tʃ]	[d]	[ɗ]	[ɛ]	[f]	[g]
H h	I i	J j	K k	L l	M m	N n	Ny ny	O o
ha	i	ja	ka	la	ma	na	nya	o
[h]	[ɪ]	[dʒ]	[k]	[l]	[m]	[n]	[ɲ]	[ɔ]
P p	Ph ph	Q q	R r	S s	Sh sh	T t	U u	V v
pa	pha	qa	ra	sa	sha	ta	u	va
[p]	[pʼ]	[kʼ]	[r]	[s]	[ʃ]	[t]	[ʊ]	[v]
W w	X x	Y y	Z z	'				
wa	xa	ya	za					
[w]	[tʼ]	[j]	[z]	[ʔ]				
aa	ee	ii	oo	uu				
[ɑː]	[eː]	[iː]	[oː]	[uː]				

Figure 1·1: Oromo Language symbol set [7].

1.1.1 Word and Sentence Boundaries

In Oromo Language, like in other languages, the blank character (space) shows the end of one word. Moreover, parenthesis, brackets, quotes are being used to show word’s boundary. Furthermore, sentence boundaries punctuations are almost similar to English language i.e., a sentence may end with a period (.), a question mark (?), or an exclamation point (!) [9].

1.1.2 Word Segmentation

The word, in Oromo Language “jecha”, is the smallest unit of a language. There are different methods for separating words from each other. This method might vary from

one language to another. In some languages, the written or textual script does not have whitespace characters between the words. However, in most Latin languages a word is separated from other words by white space characters [10]. Since Oromo Language is one of Cushitic family that uses Latin script for textual purpose, it uses white space character to separate words from each other's. Therefore, the task of taking an input sentence and inserting legitimate word boundaries, called word segmentation, is performed using the white space character.

1.1.3 Language structure

Oromo Language has a very rich morphology like other African and Ethiopian languages. The writing system of the language is straightforward, which is designed based on Latin script. Thus letters in the English language are also in Oromo Language except the way it is spelled. A detailed description of Oromo Language Writing System can be found in any text related to the language [9].

1.1.4 Sentence Structure

Oromo Language and English are different in sentence structuring. Oromo Language uses subject-object-verb (SOV) language. SOV is the type of language in which the subject, object and verb appear in that order. Subject-verb-object (SVO) is a sentence structure where the subject comes first, the verb second and the third object. Therefore, it has SOV structure. There is also a difference in the formation of adjectives in Oromo Language and English. In Oromo Language adjectives follow a noun or pronoun; their normal position is close to the noun they modify while in English adjectives usually precede the noun. For instance, namicha gaarii (good man), gaarii (adj.) follows namicha (noun).

1.1.5 Punctuation Marks

The use of punctuation marks in written Oromo Language are more or less similar to English language with the exception of the apostrophe. Apostrophe mark (‘) in English shows possession, but in Oromo Language it is used in writing to represent a glitch (called hudhaa) sound. It plays an important role in Oromo Language reading

and writing system. with the exception of some words like “ja’a” to mean “six” which is identified from the sound created. Moreover, sometimes apostrophe mark (‘) in written Oromo Language is interchangeable with the spelling “h”. For instance, “ba’e”, “ja’a” can be replaced by the spelling “h “bahe”, “jaha” respectively without changing the meaning of the words [7, 10].

1.2 Entropy Estimation and Source Encoding

A generic digital communication system consists of basic blocks as shown in Figure 1·2. Out of these major blocks the concept of information theory extends mainly itself up to the channel encoder [11]. But in this thesis, we mainly give emphasis to the first two building blocks, i.e., the information source and the source encoder. To this end, we assume written Oromo Language will propagate through these block and it should be modeled and encoded for efficient utilization of the channel. In this context, the language should be modeled and measured as given in [4]. Following this, appropriate encoder models should be utilized to find the optimal allocation bits to the symbols of the Language. With this understanding, we will give a brief overview of the major tasks to be accomplished, i.e.,source modeling, entropy estimation, and source coding, in subsequent sub sections.

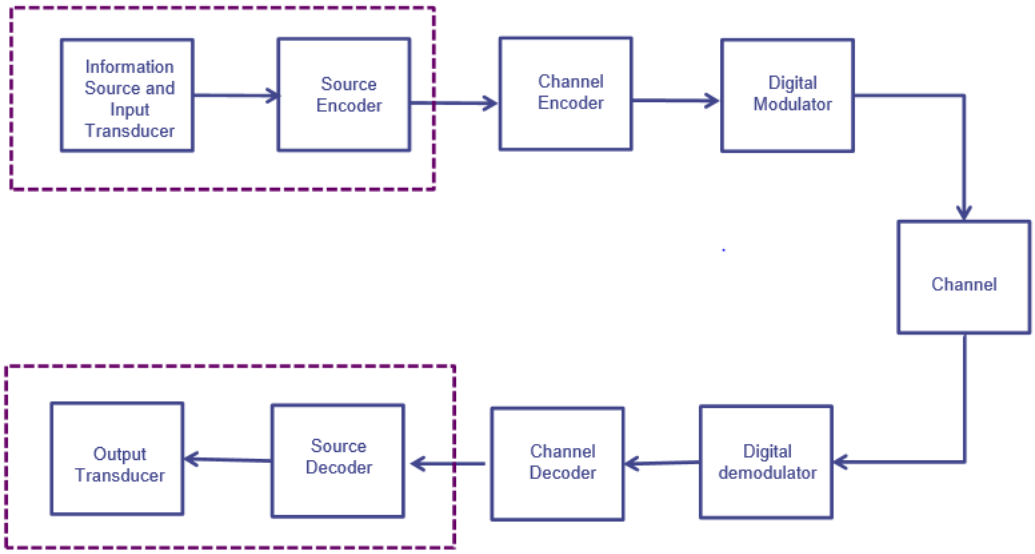


Figure 1·2: A generic digital communication system of basic blocks [11]

1.2.1 Entropy Estimation

The basic idea behind information theory is the more one knows about a topic, the lesser its information content becomes. This is because, if an event is probable, it is no surprise when it happens and thus provides little new information [11]. Inversely, if the event is less probable, it is much more informative that the event happened. Therefore, the information content is an increasing function of the inverse of the probability of the event (1/p). Thus, if more events may happen, entropy measures the average information content you can expect to get if one of the events actually happens. This implies that casting a die has more entropy than tossing a coin because each outcome of the die has smaller probability than each outcome of the coin. Thus, entropy is a measure of unpredictability of the state, or equivalently, of its average information content [3]. To get an intuitive understanding of these terms, let's consider the example of a political poll. Usually, such polls happen because the outcome of the poll is not already known. In other words, the outcome of the poll is relatively unpredictable, and actually performing the poll and learning the results gives some

new information; these are just different ways of saying that the a priori entropy of the poll results is large. Now, consider the case that the same poll is performed a second time shortly after the first poll. Since the result of the first poll is already known, the outcome of the second poll can be predicted well and the results should not contain much new information; in this case the priori entropy of the second poll result is small relative to that of the first. Hence, entropy tell us how to measure the content of a source. In this regard, entropy of a language is a statistical parameter which measures, in a certain sense, how much information is produced on the average for each letter of a text in the language [4]. For instance, imagine that we wish to transmit sequences comprising the 4 characters 'A', 'B', 'C', and 'D'. Thus, a message to be transmitted might be 'ABADDCAB'. Information theory gives a way of calculating the smallest possible amount of information that will convey this. If all 4 letters are equally likely (i.e., occurring 25% of the time or possible outcomes), we can do no better (over a binary channel) than to have 2 bits encode (in binary) each letter: 'A' might code as '00', 'B' as '01', 'C' as '10', and 'D' as '11'. Now suppose 'A' occurs with 70% probability, 'B' with 26%, and 'C' and 'D' with 2% each. We could assign variable length codes, so that receiving a '1' tells us to look at another bit unless we have already received 2 bits of sequential 1s. In this case, 'A' would be coded as '0' (one bit), 'B' as '10', and 'C' and 'D' as '110' and '111'. It is easy to see that 70% of the time only one bit needs to be sent, 26% of the time two bits, and only 4% of the time 3 bits. On average, then, fewer than 2 bits are required since the entropy is lower (owing to the high prevalence of 'A' followed by 'B' – together 96% of characters). The calculation of the sum of probability-weighted log probabilities measures and captures this effect [12].

1.2.2 Source Coding/Compression

The goal of source coding or data compression is to represent outputs symbols/letters from a source with the fewest bits such that best recovery of the source's symbol from the compressed data is possible. Data compression can be broadly classified in to lossless and lossy compression [13]. In lossless compression the goal is to minimize the number of bits in such a way that perfect (lossless) reconstruction of the source's

symbol from compressed data is possible [14]. Typical example for a lossless compression is text [15]. On the contrary, in lossy data compression the data are compressed subject to a maximum tolerable distortion.

We would like to finalize our brief discussion on source coding by presenting the first theorem of lossless source coding. Shannon's first lossless source coding theorem gives the fundamental limit on lossless source coding and shows that the entropy of a Discrete Memoryless Sources (DMS) plays a fundamental role in lossless compression of information sources. In this regard, Shannon's first theorem for lossless source codding theorem states, given X denoting a DMS with an entropy H; it can be compressed at a rate R, $R \geq H(X)$. Two such typical compression algorithms are the Huffman and the Lempel-Ziv compression algorithms. The Huffman coding algorithm is an example of a variable-length coding algorithm, and the Lempel-Ziv algorithm is a fixed-length coding algorithm. Comparatively, when the source symbols are not equally probable, an efficient encoding method is to use variable-length code words [16].

1.3 Statistical Language Models and Entropy Estimation

Shannon in [4] acknowledged that written natural languages can be modeled as discrete information sources, and written Oromo Language is not an exception. These information sources can be modeled as memoryless, and so called DMS, or as having memory or as Markovian Chain. In DMS it is assumed that the current output symbol is statistically independent of all past and future outputs. Thus,a DMS source can be considerd to follow an identically independently distributed (iid) distribution. But in reality natural languages are Markovian Chain where each current output of the source is dependent on the previous outputs. After establishing the right source model, the next phase requires measuring the probabilistic behavior of the source for proper bit allocation [17]. To this end, the measure of 'how uncertain we are of the outcome' of a discrete source is what we call entropy, designated as H. Such a measure is dependent on the probability distribution of the set of possible symbols of the source, and can be represented as $H(p_1, p_2, \ldots, p_n)$, where p_i is the probability of occurrence for letter

i. The first paper on the area states that such a measure should have the following properties [4]:

1. H should be continuous in the p_i.
2. If all symbols are equal, then H should be a monotonic increasing function . With equally likely events there is more choice, or uncertainty, when there are more possible events.
3. If a choice is broken down into two successive choices, the original H should be the weighted sum of the individual values of H.

And the paper proved that the only H that satisfies the above three equations as (1.1)

$$H = -K \sum_{i=1}^{n} p_i log p_i \tag{1.1}$$

where, the constant K amounts to a choice of a unit of measure.

1.4 Estimated Entropy for Natural Languages

Regarding studying the statistical behavior of local languages in relation to entropy estimation, [18] is a pioneer work. In the research, the authors estimated possible entropies of written Amharic language using a number of different block sizes. Furthermore, they investigated the entropy-based compression of the written form of the language so that cost efficient digital transmission of the language's text can be made. On the contrary, when it comes to Oromo Language, we were able to observe studies related with morphological studies of the language [10]. Hence, making the statistical study of Oromo Language, which helps to estimate the language's entropy, one of the untouched topic to the best of our knowledge. On the contrary, when it comes to non-local languages numerous researches have been conducted with regards of entropy estimation. We have summarized the results of those studies in Table 1.2 [19].

Table 1.2: Entropy estimation of different language [19]

N-gram	Russian	French	Rumanian	Hebrew	Arabic	Portuguese	German	Amharic
0	5.01	4.76	4.76	4.52	5.00	4.76	4.70	5.271
1	4.35	3.90	-	4.12-4.12	4.20	3.90	4.10	4.200
2	3.52	-	-	3.71	3.80	3.50	-	2.784
3	3.00	2.83	2.69	3.212	2.50	3.10	2.95	1.992

1.5 Statment of the Problem

Written Oromo Language is one of the major communication language in Ethiopia. Thus, huge traffic is expected to be generated with this language as a source. But, contrary to this fact the language's important property called entropy is not studied yet. Furthermore, the study of other Latin alphabet based languages, like written English, does not directly help the estimation of the language's entropy since the language has its own structure. Furthermore, to make matters more interesting, the presence of repetitive letters in the languages structure makes it an ideal candidate for lossless compression algorithms. For example, if we take "Shaampiyoonaan" which is Oromo Language word for champion and model it as a zero order Markov chain, i.e., assuming as if each letter is chosen independently and have no relation with each other. With this assumption, we can see that for the given word the zero order Markovian model will allocate the frequency of the letter as, $s = 1/14, h = 1/14, a = 2/7, p = 1/14, i = 1/14, o = 1/7, n = 1/7$. Thus, making the Oromo Language "a's" more compressible as compared to the English version. We strongly believe this structure of the language can open an opportunity to transmit and store the digital textual file of the language more efficient manner.

To further make matters more interesting, now a days written Oromo Language is becoming one of the main communication means in our country's telecom services, such as short message services (SMS), national security system and digital documentation. Thus, if an Oromo Language texts is written in the Latin language's symbol sets and stored or transmitted using ASCII encoding it will be a resource miss utilization. For instance, a single SMS message is expected to have a maximum of 140 bytes or 160 seven bits [18]. Hence, given the 8 or 7 bits requirement of English symbols, the 157 in an example SMS shown Figure 1·3 requires 138 bytes i.e., 7 bits/symbol

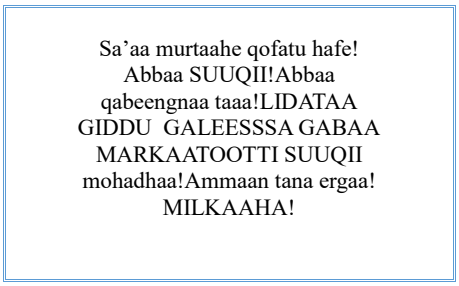


Figure 1·3: Oromo Language SMS.

But after the redundancy estimation for the above SMS the original data of 107 bytes could be compressed to 78.25 bytes. Thus, in terms of storage we could have saved 26% from a single SMS. This in aggregate would have a huge implication in telecom network infrastructures.

Thus, in this thesis we mainly aim at estimating the entropy of written Oromo Language which not only helps in studying the compression limit of the language but also opens the door for further studies on the language. Furthermore, to measure the quality of our entropy estimation we aim at compressing our sample sets using entropy based compression algorithms, i.e., Huffman and Arithmetic. Thus, this way not only we test the quality of our estimation but also show the achievability of the estimation.

1.6 Objectives

1.6.1 General Objective

The general objective of this thesis is to estimate the entropy of written Oromo Language.

1.6.2 Specific Objectives

To meet the requirment of the general objective we aim at,

- Identify and evaluate corpus based entropy estimation technique on written oromo Language.
- Select, pre-process and utilize oromo Language literature.
- Conform the estimation using entropy-based encoding to Oromo Language language.

1.7 Thesis Organization

The remainder of the thesis is organized as follows: In Chapter 2 after a brief introduction to information theory, we discuss the background and the existing works in the context of entropy estimation and entropy based encoding (compression or encoding). Following this, in Chapter 3 we provide our methodology in three main parts: namely, corpora collecting and formatting, extracting frequency counts of symbols or blocks of symbols and estimating entropy of the source based on frequency counts. Next, in Chapter 4. We have given the estimation results. Finally, in Chapter 5 we have given our summary and discusses the conclusion drawn from the results with the future work.

CHAPTER 2

FUNDAMENTAL CONCEPTS AND LITERATURE REVIEW

Information theory mainly focuses on the estimation of an optimal bits/symbol for sequences generated by a digital source [20]. Additionally, Information theory also emphasizes on finding a transformation technique that would achieve the estimated optimal bit/symbol allocation. In practice, most of the time, the transformation techniques are required to reconstruct the transformed representations with the lowest possible loss. In information theory, the optimal representation of the original data is estimated by its entropy. On the other hand, the process of achieving a representation that is close to its entropy is known as compression. On the contrary, the process of extracting the original data is decompression. In this thesis, we aim at estimate the optimal bit/symbol (entropy) of the Afan Oromo language. Additionally, we also aim at demonstrating the optimality of our estimation using renown entropy based compression algorithms. These algorithms are proved to provide results that are within the vicinity of the source's entropy.

To meet our objective, in this chapter, we try to address basic concepts in relation to source modeling, entropy estimation and entropy encoding. We give a higher emphasis on these topics due to the organization of the digital communication system as shown in Figure 1. In the figure, for a source to be transmitted it initially have to be processed into a format that is suitable to the architecture, i.e., the source has to be discretized. In our case, the source, i.e., written Afan Oromo, is already a discreet source which can easily be transformed into bit streams. However, directly converting sequences of symbols into bit streams is most of the time not efficient [21]. To this end, in digital communication systems, the source's characteristics have first to be captured via proper mathematical models that well represents the behaviors observed in symbol generation. We call this process source modeling. Following this, an efficient communication system needs to estimate the number of bits it needs to

allocate, we call this estimation (in our case entropy estimation). Finally, the system has to device mechanisms that would implement the estimated optimal bit/symbol, i.e. source coding. Hence, we will first present known source modeling approaches for written natural languages. Following this, we will present renown entropy estimation and compression techniques.

2.1 Natural Language Source Modeling

In communication system, an information source can be a text, video, image, sound or a combination of each format. In this paper we assume our information source to be a written natural language, i.e., Afan Oromo. When a written source is an information source, we have two possible source representations; i.e., statistical and morphological [21]. The statistical language model emphasizes on studying the statics and interdependencies between symbols, words and sentences. On the contrary, the morphological models emphasize on modeling a given language's grammatical structure [22]. Since, in this paper we mainly give an emphasis on the efficient transmission of a written language, i.e., Affan Oromo, we will give an emphasis on statistical language modeling. A statistical language model (SLM) models a language by uing the probability distribution of sequences or words observed in the language. In this regard, given a sequence, say of length m, it assigns a probability to the whole sequence by taking various assumptions as to be discussed in latter sub sections. The use of estimating such relative likelihood of different phrases is not limited to source modeling. On the contrary, it can be found in many natural language processing applications, especially those that generate text as an output. To this end, SLM is used in speech recognition, machine translation,part-of-speech tagging, parsing, optical character recognition, handwriting recognition, information retrieval and other applications [22].

Statistical natural language source modeling was first introduced by Shannon in [4]. The paper acknowledges written natural languages can be modeled as discrete information sources. Furthermore, it also acknowledges such information sources can further be modeled as memoryless, in the so called Discrete Memoryless Source mod-

eling (DMS); or as having memory (Markovian chain). A DMS assumes symbols are statistically independent and have no memory of all past and future inputs. Thus, the mathematical model for a DMS is a sequence of identically independently distributed (iid) random variables. But in reality natural languages have interdependencies between symbols, words and sentences due to language's structure. Thus, to model such sources appropriately [12] proposes Markovian chain approach. In Markov chain, each output of a source is dependent on its N previous outputs. With this concept in mind, we can express a DMS and a Markovian process mathematically as: Given a sequence X, such that $X = \{S_1, S_2, S_3, S_4, \ldots\ldots, S_n\}$, the probability of the sequence in DMS modeling becomes,

$$P(X) = \prod_{i=1}^{n} P(S_i) \tag{2.1}$$

Where, $P(S_i)$ is the probability of a symbol.

On the contrary, for a Markovian based modeling,

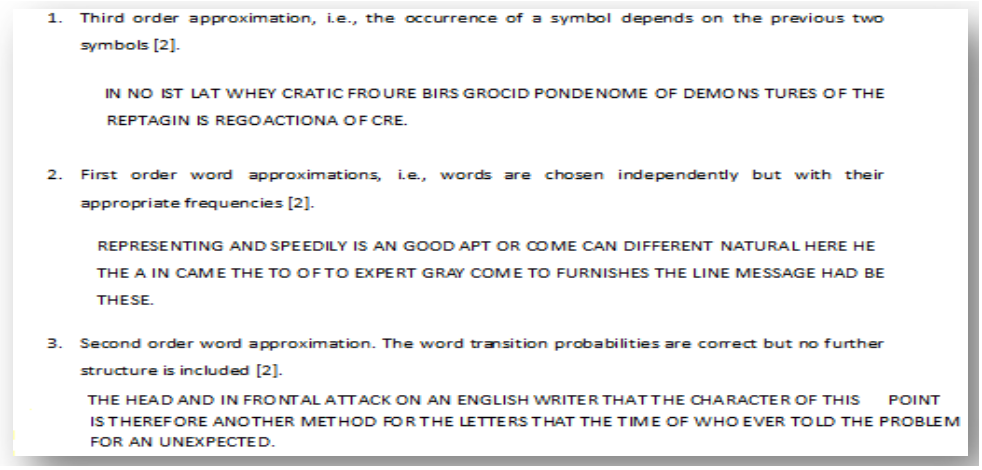
$$P(X) = P(S_1, S_2, , S_n) = P(S_n|S_1, S_2, ...S_{n-1})P(S_1, S_2, ..., S_{n-1}) \tag{2.2}$$

Where, $P(S_1, S_2, ..., S_{n-1}) = P(S_{n-1}|S_1, S_2, ...S_{n-2})P(S_1, S_2, ...S_{n-2})$

In fact, we can model the source (Written Afan Oromo) as a DMS or Markovian source. However, researches have shown that in written natural languages, each letter (Symbol) is dependent on the occurrence of pervious N symbols. In N-gram modeling, the following key points are kept in mind,

- For a sequence of length N, the occurrence of the Nth symbol is dependent on the previous N-1 symbols.
- To incorporate dependencies between words, we could use bigrams (N=2), trigrams (N=3), or more generally n-grams of any size; thus capturing short- and long-range correlation.
- Such increased "block" sizes are a key input to n-gram or block entropy estimation.

These points were better demonstrated in [23] as shown in Figure 2·1. The figure shows a machine generating sentences after being trained using N-gram Markovian models on English words. To generate the sentences, the conditional block probabilities of English words were given to the machine. Then, the machine was expected to generate the first block randomly. Following this, it generates the next word by taking various assumptions. In the first assumption, it assumes that the occurrence of a letter (symbol) is dependent on previous N symbols (1-grham). In another assumption, the machine assumes that words are independent of each other while having probability of occurrence. In the final assumption, words are assumed to follow a 1-gram model. The figure also shows that, as the assumptions are refined the generated sentences resemble the one generated by humans.


1. Third order approximation, i.e., the occurrence of a symbol depends on the previous two symbols [2].

   IN NO IST LAT WHEY CRATIC FROURE BIRS GROCID PONDENOME OF DEMONS TURES OF THE REPTAGIN IS REGOACTIONA OF CRE.

2. First order word approximations, i.e., words are chosen independently but with their appropriate frequencies [2].

   REPRESENTING AND SPEEDILY IS AN GOOD APT OR COME CAN DIFFERENT NATURAL HERE HE THE A IN CAME THE TO OF TO EXPERT GRAY COME TO FURNISHES THE LINE MESSAGE HAD BE THESE.

3. Second order word approximation. The word transition probabilities are correct but no further structure is included [2].

   THE HEAD AND IN FRONTAL ATTACK ON AN ENGLISH WRITER THAT THE CHARACTER OF THIS POINT IS THEREFORE ANOTHER METHOD FOR THE LETTERS THAT THE TIME OF WHO EVER TOLD THE PROBLEM FOR AN UNEXPECTED.

Figure 2·1: N gram approximation of the English language [3]

2.2 Natural Language Entropy Estimation

Using an appropriate source model by itself do not insure the efficient transmission of the source's symbols. To obtain efficient transmission, some logical steps have to be taken prior to the transmission of the symbols. In this aspect, the source's symbols have first to be represented by a logical stream of bits. Following this, the stream of bits has to be represented by a distinguishable physical baseband signal. Finally, the baseband signal has to be converted into a pass band signal that is

suitable to the channel at hand [11]. However, before the modulation of the logical bit streams, an optimal utilization of the underlying channel requires the determination of an optimal amount of bits/symbols. This is because, an inefficient bit allocation implies the requirement of additional base band signals to represent a symbol. This, if not managed properly, will significantly reduce the symbol rate. To overcome this challenge, the optimal amount of bits/symbol needed for a given source have to be properly defined. In this regard, [4] introduced the concept of information theory in his pioneering work. Shannon associated information to the uncertainty (probability of occurrence) of an event, for instance the probability of symbol genration. He named this association the information content of a symbol. In his argument, Shannon stated that if a symbol has a high probability of occurrence, then it has the lowest possible information content [4]. For instance, the information that the "Sun will rise tomorrow" is obvious and will add no value to the listener. However, the information that "there will be an earthquake tomorrow" will add a significant value to the receiver. To make this definition more formal, Shannon also imposed the following restrictions on the function measuring the information content of symbols. In this regard, given a set of symbols with probabilities of occurrences $P = \{p_1, p_2, p_3, \ldots\ldots\ldots, p_N\}$

- The function measuring the information content, H, should be a continuous function of p_i.
- H must be a monotonically increasing function of p_i and will have a maximum when symbols are equiprobabel, i.e., $p_i = \frac{1}{N}$.

The first requirement is an obvious one, since we have associated information measure with uncertainty, it must be a continuous function of p_i. The second restriction comes from the fact that; the unpredictability of the source becomes high when the source generates symbols at an equally likely probability. Shannon argues that a mathematical function that meets the above two requirements is a logarithmic function, i.e., $H \propto -logp_i$. However, information measure does not directly tell us the link between bit allocation and uncertainty. To establish the link, one must look the transmission of symbols from the receiver side. In this regard, we will adopt the approach used in [18] and we will assume we have a source with a symbol set $X = \{x_1, x_2, x_3, x_4\}$.

Additionally, we will also assume that the symbols are independent and have a probability of occurrence, $\{p_1, p_2, p_3, p_4\}$; where, $p_1 > p_2 > p_3 > p_4$ and $\sum_{i=1}^{4} p_i = 1$. Furthermore, if we assume a transmitter and a receiver are communicating using a "Yes" or "No" answers. In this context, the transmitter will first send a symbol and wait for the receiver to ask a question. To this end, if the receiver has a knowledge of the distribution, after the reception of a symbol, the first logical question a receiver asks will be is it x_1? This is because, the receiver will expect to receive x_1 most of the time. If the symbol transmitted is x_1, then the transmitter will reply with "Yes" and the transmission ends . However, if the received symbol is not x_1 then, the receiver will ask the next logical question, i.e., is it x_2? Using this approach, the transmission of symbols follows the tree structure shown in Figure 2·2. In the figure, if we substitute the "Yes" answers with a logical "1" and the "No" answers with a logical "0"; then, this will tell us the bit allocation for each symbols while taking the probability of occurrence into account. Thus, in this approach, the bit streams become; $X = \{x_1 = 1, x_2 = 01, x_3 = 001, x_4 = 0001\}$. If we think of it, this bit allocation is an efficient one. This is because, the rare symbols are allocated with larger bit streams. Thus, the transmitter will be transmitting a single bit most of the time. Moreover, this allocation mechanism has a direct link to the definition of information. In this context, $Number of questions = bit/symbol \approx log_2 1/p_i \approx H$.

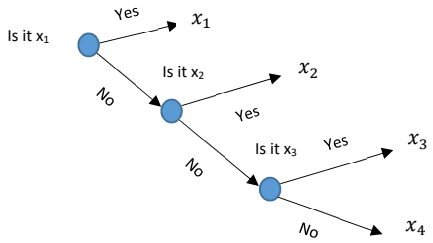


Figure 2·2: Transmission of symbols based on "Yes" or "No" questions

Hence, assuming independence between letters, given N symbols; the average optimal bit/symbol becomes,

$$H = -\sum_{i=1}^{n} p_i log_2 p_i \tag{2.3}$$

Equation (2.3) defines the self-entropy of a given source while assuming independence between symbols. However, in natural languages, symbols are assumed to be interdependent in a Markovian basis. Shannon defined the entropy of such sources using the conditional probabilities of symbols [3]. In this regard, given a block with N symbols, $X_i = x_1, x_2, x_3, \ldots\ldots\ldots, x_N$, having a block probability of p_{B_i}. Moreover, if the conditional probability of a symbol is defined as $p_{B_i}(x_N) = p(x_N|x_1, x_2, \ldots., x_N)$.

$$F_N = -\sum_i p_{B_i} log_2 p_{B_i}(x_N) \quad bits/symbol \tag{2.4}$$

Where, p_{B_i} is all the possible blocks generated by a source. Whereas, $p_{B_i}(x_N)$ is all the possible conditional probabilities. However, accounting for all possible conditional probabilities is a challenging task most of the time. To overcome this problem, Shannon proposed to utilize Bayesian inference, i.e.,

$$p_{B_i}(x_N) = p_{B_i}/p_{B_{i-1}} \tag{2.5}$$

Where, $p_{B_{i-1}} = p(x_1, x_2, x_3, \ldots\ldots\ldots, x_{N-1})$

Substituting this in (2.5), we get

$$F_N = -\sum_i p_{B_i} log_2 p_{B_i}/p_{B_{i-1}} \tag{2.6}$$

$$F_N = -\sum_i p_{B_i} log_2 p_{B_i} + \sum_i p_{B_i} log_2 p_{B_{i-1}} \tag{2.7}$$

Now if we define $G_N = -1/N \sum_i p_{B_i} log_2 p_{B_i}$, then

$$F_N = N\ G_N - (N-1)\ G_{N-1} \tag{2.8}$$

Researchers have utilized (2.4) and (2.8) to estimate the entropy of various languages. In this regard, Shannon was the pioneer on estimating the entropy of the English language using (2.4 [4]. To estimate the entropy, Shannon has utilized the language's

Table 2.1: Conditional entropy of written English [12]

N	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	100
Upper	4.03	3.42	3.0	2.6	2.7	2.2	2.8	1.8	1.9	2.1	2.2	2.3	2.1	1.7	2.1	1.3
Lower	3.19	2.5	2.1	1.7	1.7	1.3	1.8	1.0	1.0	1.0	1.3	1.3	1.2	0.9	1.2	0.6

speakers to estimate the conditional probabilities. In this context, Shannon first selected a range of books and prepared snippets as shown in Figure 2·3 [18].

"Th_ onl_ wa_ to ge_ ri_ of a tempta____ is to yie__ to it. Resi__ it, an_ you_ soul gro__ sic_ wi__ longi__ fo_ th_ thin__ it ha_ forbi____ to itse__."

"The only way to get rid of a temptation is to yield to it. Resist it, and your soul grows sick with longing for the things it has forbidden to itself."

Figure 2·3: A snippet used for conditional probability estimation [18]

Shannon then asked test subjects (language speakers) to predict missing symbols and blocks of symbols. This way, Shannon was able to estimate the conditional probabilities of English symbols. Thus, paving the way for the estimation of the language's conditional entropy. Using this approach, Shannon was able to estimate written English has an entropy of English upto N=100 as shown in Table 2.1 [3]. In recent years, various researchers have utilized the advancement made in computers to estimate the entropy of various natural languages using (2.8) [18, 24]. For instance, [24] has used (2.8) to estimate the entropy of written English and Spanish. In the paper, the author collected a range of literature written in both languages and utilized a sliding window of size N to count the occurrences of symbols and blocks of symbols. Following this the author estimated the languages entropy using the frequency counts and (2.8). Similarly, [18], has used the same approach to estimate the entropy of written Amharic up to N=20. In general, in a different perspective, the entropy of a source tells us the amount of redundant data that can be removed. In this context, we can utilize the entropy of the source to define the compression ratio which is given

as,

$$R = 1 - \frac{G_N}{G_0} \tag{2.9}$$

Where, G_N is the conditional block entropy . Whereas, G_0 is the entropy of the source while symbols are assumed to be independent.

2.3 Source Coding/Compression

The source's entropy tells us the minimum obtainable bit/symbol [20]. The question of obtaining this lower bound is left to source coding. Source coding (data compression) is The process of efficiently allocating streams of bits to symbols [21]. Data compression is now almost a common requirement for every applications as it is a means for saving the channel bandwidth and storage space. Data Compression is an art of allowing a technique to reduce the volume of data i.e. excess information, by maintaining the quality of data. There a number of algorithms available for compression of files of different formats. But, the choice of an algorithms is dependent on the data at hand and the qualities of the chosen method, for instance, achievable compression ratio, data loss, etc. In this context, even for a given data type, there are a number of approaches are available. However, the best one depending upon the need is very important and most of the time challenging. In general, compression methods can be categorized in two broad categories, i.e., lossy and losseless [25].

Lossy Compression: In lossy data compression (perceptual coding), the loss of some fidelity is acceptable. The Lossy technique is a data compression method which compresses data by discarding (losing) some of it. The procedure aims to minimize the amount of data that needs to be handled, and/or transmitted by a computer [25]. These compression techniques rely on the capability of the receiver on filling up missing data. For instance, the loss of some speech segments is acceptable since a human being is capable of filling up lost parts using the context [14]. Such scenarios are also acceptable in image compression.

Lossless Compression: In lossless data compression, the integrity of the data

is preserved. The original data and the data after compression and decompression are exactly the same. This is because, for methods under this subcategory, the compression and decompression algorithms are exact inverses of each other: thus, no part of the data is lost in the process [26]. Text (written natural language) Compression is considered a Lossless type. For instance, one of the popular file format i.e., ZIP, is highly utilized for compression of data in modern computers. Lossless data compression highly exploits statistical redundancy to express data more precisely without any loss in information [11]. In this regard, the two main lossless source coding are the Huffman coding algorithm and the Lempel-Ziv algorithm. The Huffman coding algorithm is an example of a variable-length coding algorithm, i.e., code words vary in length. On the contrary, Lempel-Ziv algorithm is a fixed-length coding algorithm. Comparatively, when the source symbols are not equally probable, an efficient encoding method uses variable-length code words [15]. In general, lossless encoding algorithms can further be broadly divided into Dictionary based, Entropy based as shown in Figure 2·4 [18]. However, due to the objectives of this paper we will mainly focus on entropy based encoding algorithms, specifically Huffman and Arithmetic. This is because, entropy based encoding algorithms are known to achieve compression results that are close to the entropy of the source. For instance, Huffman encoding is known to achieve compression ratio that is near to the entropy [18].

$$C_{R_huffman} = H + \delta \tag{2.10}$$

Where, δ is the maximum probability of symbol.

Generally, we have to see lossless compression techniques in two perspectives. First, they can be seen as techniques reducing the size of the file to save storage space.

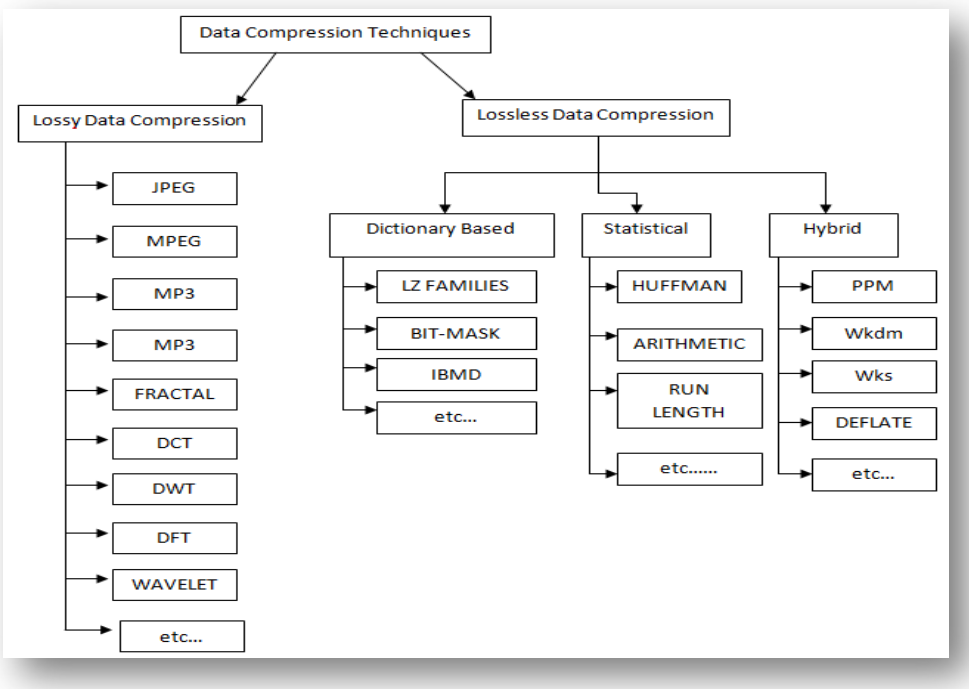


Figure 2·4: Categories of source encoding techniques [18]

In another perspective, they can be seen as techniques used for faster data transmission. In both aspects, the quality of the compression techniques is measured by two key parameters, i.e., the compression ratio (C_R) and the compression factor (C_F) [27].

$$C_R = \frac{compressedfilesize}{orginalfilesize} \tag{2.11}$$

$$C_F = \frac{1}{comressionratio} \tag{2.12}$$

To achieve high compression ratio, entropy encoding techniques first count the frequency of symbols (blocks of symbols) according to their occurrence [13]. Then, the algorithms generate code words that are variable or fixed in length that are dependent on the frequency counts. On the contrary, dictionary based encodings do not encode based on their frequency count. In such compression methods, most of the time, a data structure known as a dictionary is maintained [28]. While compressing, the encoder looks into the dictionary to identify if a symbol (blocks of symbols) is within the

dictionary. If a match is found, the symbols are replaced with the binary values of the index within the dictionary. We will not further dive into dictionary based encoding techniques. However, as stated earlier, since our objective, i.e., entropy estimation, is highly correlated with entropy encoding techniques, we will further our discussion with two selected entropy encoding techniques, i.e., Huffman and Arithmetic.

2.3.1 Huffman Coding

Huffman compression was first developed by David Huffman as a solution to his class assignment. The class was the first ever in the area of information theory and was taught by Robert Fano at MIT [29]. Codes generated using this technique or procedure are called Huffman codes. These codes are prefix free codes, i.e., no code word is not a prefix of another code word. Furthermore, the code words are optimum for a given source model (set of probabilities). The Huffman procedure is based on two observations regarding optimum prefix free codes [30].

- In an optimum code, symbols that occur more frequently (have a higher probability of occurrence) will have shorter code words than symbols that occur less frequently.
- In an optimum code, the two symbols that occur least frequently will have the same length. It is easy to see that the first observation is correct. If symbols that occur more often had longer code words as compared to their counterpart; then, the source will be transmitting longer code words more often. Thus, the average number of transmitted bits per symbol would be larger than if the conditions were reversed [29].

Huffman and Arithmetic compression techniques have found various applications. For instance, in [examples in Amharic document], Huffman and Arithmetic compression techniques have been utilized to compress books written in English and French. The compression was utilized to minimize the storage space requirement as summarized in Table 2.2.

Table 2.2: Huffman and Arithmetic compression of books written in English and French [18].

Language	Original File Size In bytes	Encoded File Size In bytes	Algorithm	Compression ratio
English	7,233,536	86,016	Huffman & Arithmetic	84.0952
	4,050,944	2,220,032	Huffman	1.8247
		2,199,552	Arithmetic	1.8417
	2,473,984	1,966,080	Huffman	1.2583
		1,548,288	Arithmetic	1.5979
	483,328	278,528	Huffman	1.7353
		274,432	Arithmetic	1.7611
	430,080	253,952	Huffman & Arithmetic	1.6935
	126,976	77,824	Huffman & Arithmetic	1.6316
	200,889	43,446	Adaptive Huffman	4.6239
		41,338	Adaptive Arithmetic	4.8597
	152,089	87,795	Adaptive Huffman	1.7323
		87,136	Adaptive Arithmetic	1.7454
	108,732	84,553	Adaptive Huffman	1.2859
		82,615	Adaptive Arithmetic	1.3161
French	133, 754	77,125	Adaptive Huffman	1.7342
		76,744	Adaptive Arithmetic	1.7429

In another application domain, Huffman algorithm has been utilized in minimizing the bit/symbol requirement of short messages texts (SMSs) written in English [18]. The paper has shown that, N-SMS messages can at least be reduced to N-1 messages as summarized in Table 2.3.

Table 2.3: Huffman compression of English SMS Messages [18].

No.	initial characters	compressed characters	initial message size	compressed message size	Raw size (bits)	Compressed size (bits)	Compression ratio
1	118	74	1	1	862	592	1.46
2	44	27	1	1	308	216	1.43
3	61	39	1	1	427	312	1.37
4	131	83	1	1	917	664	1.38
5	78	50	1	1	546	400	1.37
6	194	122	2	1	1358	976	1.39
7	202	126	2	1	1414	1008	1.40
8	216	133	2	1	1512	1064	1.42
9	213	131	2	1	1491	1048	1.42
10	197	122	2	1	1379	976	1.41
11	338	211	3	2	2366	1688	1.40
12	401	254	3	2	2807	2032	1.38
13	384	237	3	2	2688	1896	1.42
14	345	211	3	2	2478	1688	1.47
15	415	254	3	2	2905	2032	1.43

On the contrary, in [18], the bandwidth utilization rate of E-mails in the compressed and uncompressed state were shown. Huffman compression technique was used to compress the E-mails exchanged by four simulated links, i.e. links from,

- Mumbai-Trivandrum
- Mumbai-Bangalore
- Hyderabad-Bangalore
- Trivandrum-Hyderabad

The evaluation of the link utilization is summarized in Table 2.4. Additionally the

paper also investigated the improvement on packet drop rate due to the compression techniques. The work showed that the compressed E-mails have improved packet delivery rate as shown in Table 2.5.

Table 2.4: Performance evaluation of a link transmitting E-mails using Huffman Encoding [18].

E-mail (Huffman)	Link Utilization (%)	
Link	Compressed Data	Uncompressed Data
From Mumbai-Trivandrum		
Bandwidth utilization		
90%	40.33	71.60
From Mumbai- Bangalore		
90%	46.68	84.13
Hyderabad-Bangalore		
90%	61.60	91.40
Trivandrum-Hyderabad		
90%	58.03	88.80

Table 2.5: Packet Delivery and Drop Due to Huffman Compression [18].

Application	Ranges	
E-Mail	Compressed Data	Uncompressed Date
Packet Delivered	330-750	225-610
Packet Dropped	129-347	161-435
Drop Rate (%)	30-75	41-142

With this motivations in mind, we can generalize the Huffman compression using the following steps.

- Sort symbols in descending order of probability of occurrence.

- Locate two nodes with the lowest probability of occurrence.
- The two least node is added and an internal node of this two node is created and the added sum of the two node is given as its weight.
- The internal node is now added to the list and the two node as it's child.
- One of the child node is assigned 1 and other as 0 during coding.
- Previous steps are repeated till there is no other node left in the tree. The free node is root of the tree.

To elaborate on the algorithm, let's consider the encoding of symbols given in Table 2.6. In this example, assume that the frequency of the characters is as shown below.

Table 2.6: Frequency counts of letters.

Characters	A	B	C	D	E
Counts	17	12	12	27	32

In the table, symbols B and C are found to be the least frequent symbols. Thus, Huffman first group these symbols and aggregate their count, i.e., 24, as shown in Figure 2·5b. Following this, the algorithm identifies symbol A to be the least frequent symbol. Thus, Huffman moves up and aggregate A, B, C as shown in Figure 2·5. The algorithm continues in this way until it builds up the tree shown in Figure 2·6.

Figure 2·5: Building up Huffman code trees

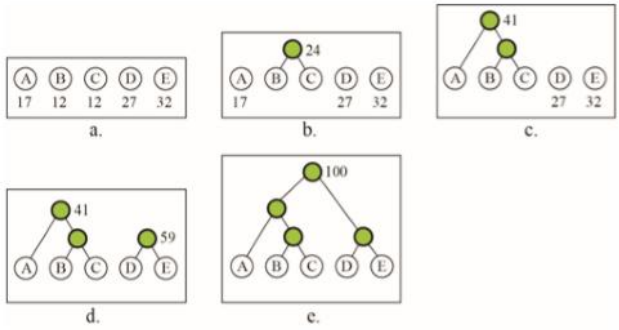
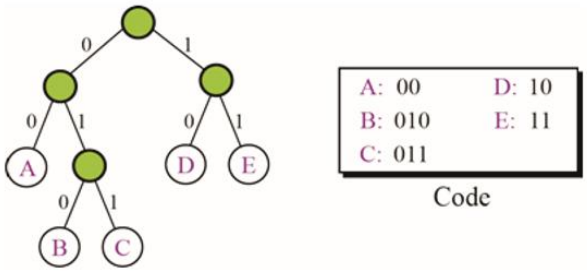


Figure 2·6: Reduced Huffman code



2.3.2 Arithmetic Coding

Arithmetic coding is a technique for coding that allows the information from the messages in a message sequence to be combined to share the same bits [13]. The technique allows the total number of bits sent to asymptotically approach the sum of the self-information of the individual messages (recall that the self-information of a message is defined as $log_2\frac{1}{p_i}$). To see the significance of this, consider sending a thousand messages each having probability .999. Using a Huffman code, each message has to take at least 1 bit, requiring 1000 bits to be sent. On the other hand the self-information of each message is $log_2\frac{1}{p_i} = .00144\ bits$, so the sum of this self-information over 1000 messages is only 1.4 bits. It turns out that arithmetic coding will send all the messages using only 3 bits, a factor of hundreds fewer than a Huffman coder. Of course this is an extreme case, and when all the probabilities are small, the gain will be less significant. Arithmetic coders are therefore most useful when there are large probabilities in the probability distribution. The main idea of arithmetic coding is to represent each possible sequence of n messages by a separate interval on the number line between 0 and 1, e.g. the interval from .2 to .5. For a sequence of messages with probabilities $p_1, p_2, \ldots\ldots\ldots, p_n$ the algorithm will assign the sequence to an interval of size$\prod_{i=1}^{n} p_i$, by starting with an interval of size 1 (from 0 to 1) and narrowing the interval by a factor of p_i on each message i. We can bound the number of bits required to uniquely identify an interval of size s, and use this to relate the length of the representation to the self-information of the messages. In the following discussion we assume the decoder knows when a message sequence is complete either by knowing the length of the message sequence or by including a special end-of-file message. This

was also implicitly assumed when sending a sequence of messages with Huffman codes since the decoder still needs to know when a message sequence is over. We will denote the probability distributions of a message set as$p(1), \ldots\ldots\ldots\ldots, p(m)$ and we define the accumulated probability for the probability distribution as [13]:

$$f(j) = \sum_{i=1}^{j-1} p(i) \qquad (j = 1\ldots\ldots\ldots, m) \tag{2.13}$$

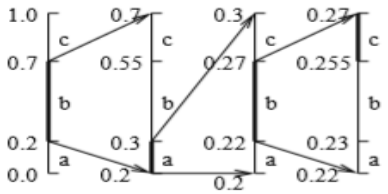


Figure 2·7: Generating an arithmetic code [13]

As noted from Figure 2·7 the appearance of a symbol restricts the tag to sub interval that is disjoint from any other subinterval that may have been generated using the above process. Hence, any number residing in the interval of the tag can be used as a tag of the sequence.But a practical choice would be to choose the tag value to be at the midpoint of the interval [31]. We could summarize the above procedure by first making the following definitions, assumptions and steps [13, 31].

- Given the sequence $S = \{X_1, X_2, X_3, \ldots.X_n\}$ define each sequences to be $X_1 = 1, X_2 = 2, X_3 = 3, , , , X_n = n$
- Define $U^{(n)}$ the upper bound tag limit of a sequence of length n
- Define $L^{(n)}$ as the lower bound tag limit of a sequence of length n
- Define$[U^{(0)}, L^{(0)}....0, 1]$

$$L^n = L^{n-1} + (U^{n-1}F_x(x_n) \tag{2.14}$$

$$U^n = L^{n-1} + (U^{n-1} - L^{n-1})F_x(x_n) \tag{2.15}$$

$$T^n = \frac{U^n + L^n}{2} \tag{2.16}$$

Where, $U^(n-1), L^(n-1), T^n$ are the upper and lower bound of the previous sequences and the tag of the sequence respectively.

To make the tag generation process a bit clear let us consider the tag generation process for the sequence with symbol probabilities of $S = a_1 = 0.7, a_2 = 0.1, a_3 = 0.2$ [18]. With this assumption, the tag for the symbols $\{a_1\}$, $\{a_1, a_2\}$ and $\{a_1, a_2, a_3\}$ become; $\{a_1\}$:

$$L(a_1) = L^{(1)} = L^{(0)} = L^{(0)} + (U^{(0)} - L^{(0)})F_x(0) = 0 + (1-0) * 0 = 0$$
$$U(a_1) = U^{(1)} = L^{(0)} = L^{(0)} + (U^{(0)} - L^{(0)})F_x(1) = 0 + (1-0) * 0.7 = 0.7$$

$\{a_1, a_2\}$:

$$L(a_1a_2) = L^{(2)} = L^{(1)} + (U^{(1)} - L^{(1)})F_x(1) = 0 + (0.7-0) * 0.7 = 0.49$$
$$U(a_1a_2) = U^{(2)} = L^{(1)} + (U^{(1)} - L^{(1)})F_x(2) = 0 + (0.7-0) * 0.8 = 0.56$$

$\{a_1, a_2, a_3\}$:

$$L(a_1a_2a_3) = L^{(3)} = L^{(2)} + (U^{(2)} - L^{(2)})F_x(2) = 0.49 + (0.56 - 0.49) * 0.8 = 0.546$$
$$U(a_1a_2a_3) = U^{(3)} = L^{(2)} + (U^{(2)} - L^{(2)})F_x(3) = 0.49 + (0.56 - 0.49) * 1 = 0.56$$

Thus, the Tag for the sequence $\{a_1a_2a_3\}$ becomes,

$$T(a_1a_2a_3) = T^3 = \frac{0.546+0.56}{2} = 0.553$$

Even though, the presented version of the Algorithm can indeed encode large sequences of symbols, the precision required to represent the tags become beyond capability of most computational devices. This is because as more symbols are encoded tag intervals become smaller and smaller, and one has to store these small fractions as one and zero [13]. To overcome this limitation the integer implementation of the algorithm is preferred [32, 13]

From source coding techniques, Arithmetic code is a uniquely decodable code that provides a rate close to the entropy for long stationary sequences. in our paper result and

discussion part we have discussed and showed This ability to encode sequences directly instead of as a concatenation of the codes for the elements of the sequence makes this approach more efficient than Huffman coding for the given probabilities [**39**].

CHAPTER 3

METHODOLOGY

Entropy based estimation and entropy based encoding (Data compression) is now almost a common requirement for most applications as it is a means for saving the channel bandwidth and storage space. Data Compression is an art which allows the reduction of data volume, i.e. excess information, while maintaining the quality of data. The compression process includes a set of different coding algorithms which meets the goal of improving bits/symbol allocation. With this said, the methodology used to conduct this research work comprises four major phases. The first phase encompasses reviewing recent works and areas closely related to this research in order to acquire an in-depth understanding and knowledge. In this phase, the fundamentals of information theory concepts, source coding, related corpora based entropy estimation and corpora formatting process are reviewed. The remaining three phases realize the entropy estimation of Afan Oromo. ln general, the methodologies used to conduct this research work are summarized in Figure 3·1.

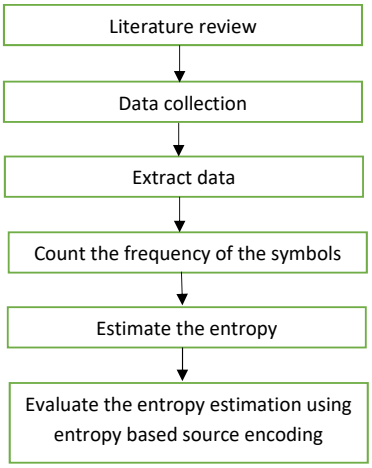


Figure 3·1: Summary of the methodology

Generally speaking, the three main phases realizing the entropy estimation of Afan Oromo can be summarized as:

- Corpora collection and formatting
  - Corpus (from Oslo text laboratory)
  - Literature (from different authors and researchers)
  - Corpora editing (using different editing software and scripts)
- Entropy estimation
  - Markovian chain (N-gram) based modeling
  - Programming Implementation of sliding window
- Crosschecking the estimated entropy using entropy based source encoding
  - Static Huffman
  - Arithmetic encoding algorithm

3.1 Corpora Collection and Corpora Formatting

To perform the Markovian based source modeling, we have collected ten literature written in Afan Oromo. The collected literature fall in different categories, i.e., fiction, political, historical, spiritual, scientific papers and a corpus prepared for Afan Ormo sentence tagging. However, since the literatures are prepared for different reasons some pre processing have to be conducted. In this regard, first the original document from different authors and researchers are converted into editable format. In this context, we were lucky since we were able to obtain the literature in Microsoft word. However, unnecessary contents such as images, page numbering, titles and decorative features have to be removed. Additionally, the corpus obtained from the Oslo text laboratory is originally prepared by breaking Afan Oromo literature into 31 million tokens for sentence tagging. Thus, we have manually re-edited the document into a formal literature by removing tokenization symbols. In general, to convert the corpora into a format that is suitable for frequency extraction, we have utilized different text

editing software such as notepad++, MATLAB language processing package kit and python. We finally saved the formatted document as a utf-8 ".txt" file for entropy extraction.

3.2 Probability extraction and Entropy estimation

In this paper, the entropy of Afan Oromo Language is estimated using frequency table list of symbols and blocks of symbols. Blocks are defined here to be unique symbols or sequence of symbols observed in the corpora; this is better demonstrated in Figure 2. The figure considers the words "Wallaaluun fagaata", from these words and with a block size of 1 (i.e., N=1) we are able to extract frequency counts for eight unique symbols. However, if we increase the block size to two we are able to extract frequency counts for more than fifteen unique symbol blocks.

To extract these block probabilities, a Matlab and python scripts were utilized. In this regard, the scripts utilize a sliding window technique to count the frequencies of symbols observed in the corpora. This is also better demonstrated in Figure 3·3. In the figure, the script sequentially read ten blocks at a time by incrementally sliding a window of size ten across the corpora.

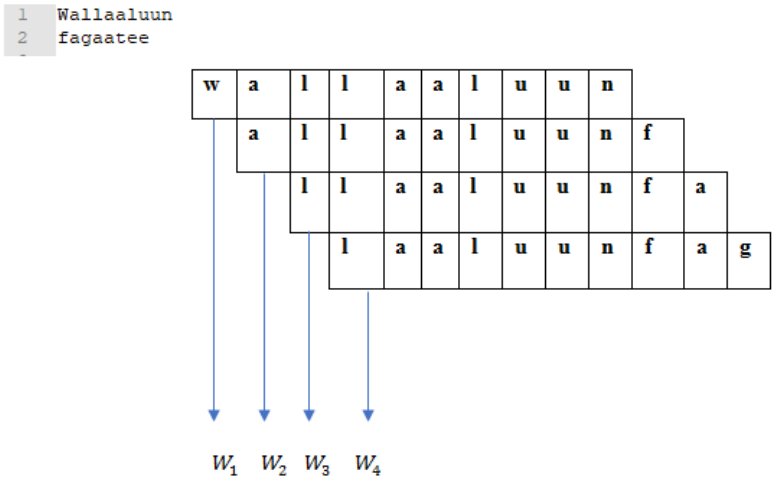


Figure 3·2: Demonstration of sliding window for N=10.

The script then counts the occurrence of each unique symbol blocs. It then estimates the probabilities of the blocks using the law of large numbers, i.e.:

$$P_{B_N} = \frac{countofB_N}{Numberofuniquesymbols} \tag{3.1}$$

Where, B_N is a block

Following this estimations, we have taken Shannon's two entropy estimation equations into account, i.e.,

$$G_N = \frac{1}{N}\sum P(B_i)logp(B_i) \quad (BlockEntropy) \tag{3.2}$$

$$F_N = NG_N - (N-1)G_N \quad (ConditionalEntropy) \tag{3.3}$$

In reality, (3.3) provides a proper estimation of the entropy; however, it is difficult to estimate it directly since it requires the marginal probabilities of symbols which on the other hand requires infinite data input. Thus, for our work we initially estimate the entropy using (3.2) and later calculate the conditional entropy using (3.3).

3.3 Matlab and Python Based Implementations

Now a day's different natural language processing software are available. We have used python script and malt lab language processing tools for probability extraction, to estimate the block entropy and perform entropy based source encoding. We have selected these tools for the following reasons:

- The scripts can easily be integrated with web based applications.
- Python and Matlab provide well developed text processing libraries toolkit. Matlab additionally has n-gram bagging for natural language processing.
- They treat primitive data type as object.

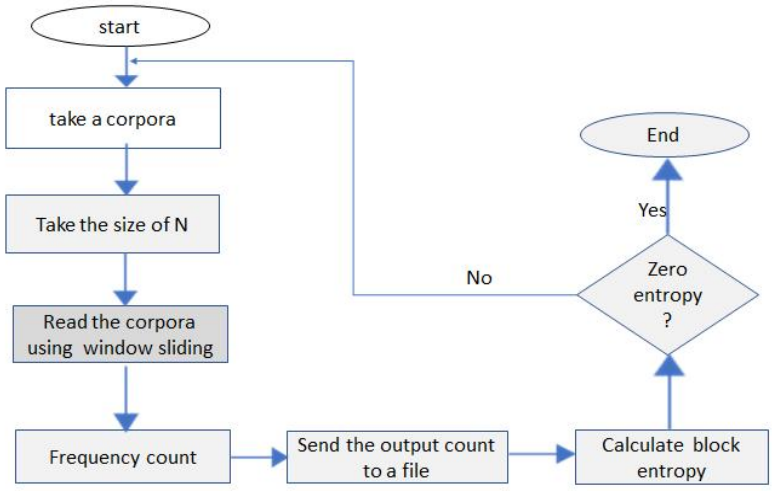


Figure 3·3: N gram based probability extraction.

3.4 Cross Checking the Entropy Estimation Using Source Coding

An alternative approach to entropy estimation is the utilization of entropy based source coding. This is because such source coding algorithms are proved to provide an average bit/symbol allocations that are close to the lower bound, i.e., the source's entropy. For instance, the Huffman source coding technique is known to provide an average bit per symbol allocation of $H + \delta$. Where, H is the source's entropy and δ to be the maximum probable symbol or blocks of symbols. With this in mind, we have selected two entropy based source encodings, i.e., Huffman and Arithmetic source encodings, as a means of cross checking our estimation. To achieve this objective, the compression and decompression process of the algorithms were implemented using Python scripts. We then compressed the collected literatures using the implementation. Following this, the original and the compressed file size was analyzed to estimate the average *bit/symbol* allocation.

CHAPTER 4

RESULT

We have compiled our result into two sections,i.e.,

- Block and conditional entropy estimation.
- Entropy based encoding.

4.1 Entropy Estimation

4.1.1 Selected Corpora

We have used equations (2.4) and (2.8) to estimate the block and conditional entropy of written Oromo Language. The estimations were made on the ten literature shown in Table 4.1. We have summarized the estimation results in Table 4.2,4.3 and Table 4.4,4.5 for the block and conditional entropy respectively.

Table 4.1: Selected Corpora

No.	Book (Corpora) Title	Authors	Category
1	Afan Oromo Corpus [33]	Oslo Text Lab.	Different Texts
2	Bible	various	Religious
3	Duka oromo(Duukaa Oromoo) [34]	Alemu Yadessa	History
4	Herra hajjii fii umrah [35]	Shekh Jamaal	Religious
5	mul'atan qaba [36]	Bekele Gerba	Political
6	MA. Thesis [37]	Sukkare Baqqle	Research
7	PhD. Dissertation	Taaddasaa Dirribaa	Research
8	Sagl Geda (Sagalee Gadaa)	Caalaa Soorii, et al.	Journal
9	Qaroomina Qaruu [38]	Amanuel Olijirra	scientific
10	Soomana Ramadaana	Heera, et al.	Religious

4.1.2 Block Entropy Estimation

We have calculated the block entropy for $N = 0 - 10$ as shown in Table 4.2 and 4.3 using (2.4). The graphical interpretation of the results is shown in Figure 4·1.

Table 4.2: NG_N *in bits/block* for $N = 0 - 10$

Book No.	0	1	2	3	4	5	6	7	8	9	10
1	7.99	4.39	7.58	10.01	12.22	13.49	14.57	15.53	15.93	16.21	16.71
2	7.99	4.39	7.58	10.01	12.22	13.49	14.57	15.53	15.93	16.31	16.62
3	7.99	4.39	7..58	10.09	12.02	13.49	14.57	15.37	15.94	16.35	16.63
4	7.99	4.25	7.52	10	11.66	12.65	13.22	13.59	13.82	13.98	14.08
5	7.99	4.34	7.52	10.03	11.98	13.57	14.73	15.18	15.63	15.9	16.15
6	7.99	4.39	7..58	10.09	12.02	13.49	14.57	15.37	15.94	15.9	16.15
7	7.99	4.37	7.56	10.05	11.86	14.08	14.78	15.23	15.59	15.77	16.05
8	7.99	4.37	7.58	10.08	11.9	13.18	14.12	14.82	15.51	15.89	16.15
9	7.99	4.27	7.51	10	11.81	13.1	14.01	14.99	15.2	15.88	16.05
10	7.99	4.31	7.56	10.05	11.86	14.08	14.78	15.23	15.59	15.89	16.15
Mean	7.99	4.35	7.57	10.03	12.04	13.785	14.675	15.38	15.76	16.05	16.43

Table 4.3: NG_N *in bits/block* for $N = 11 - 20$

Book No.	11	12	13	14	15	16	17	18	19	20
1	16.82	16.87	16.91	16.94	16.97	16.99	17.00	17.00	17.00	17.00
2	16.82	16.87	16.91	16.94	16.97	16.99	17.01	17.03	17.03	17.03
3	16.85	17.07	17.29	17.39	17.46	17.52	17.54	17.56	17.58	17.58
4	14.14	14.19	14.22	14.25	14.26	14.28	14.29	14.30	14.30	14.30
5	16.37	16.53	16.69	16.75	16.85	16.86	16.87	16.88	17.89	17.89
6	16.2	16.24	16.28	16.31	16.34	16.36	16.38	16.39	17.39	17.39
7	16.2	16.32	16.41	16.48	16.55	16.60	16.64	16.67	16.70	16.70
8	16.22	16.39	16.47	16.54	16.58	16.59	16.61	16.63	16.65	16.66
9	16.70	16.83	16.89	16.85	16.85	16.87	16.90	17.01	17.01	17.01
10	16.20	16.24	16.28	16.31	16.34	16.36	16.38	16.39	17.39	17.39
Mean	16.25	16.35	16.43	16.48	16.52	16.54	16.56	16.58	16.89	16.89

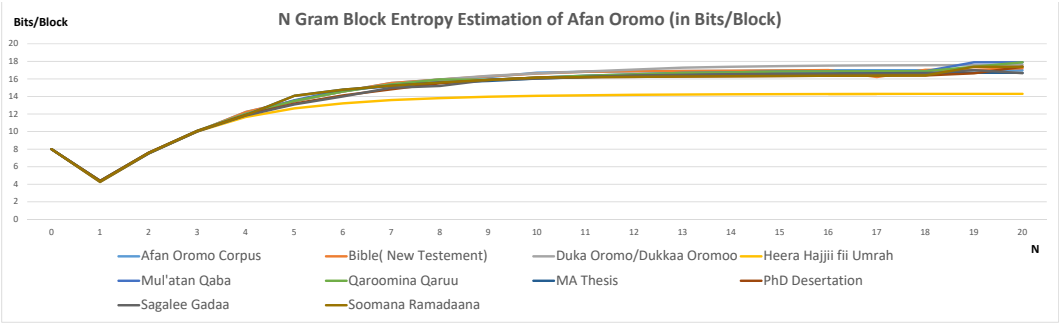


Figure 4·1: Graphical Interpretation of Affan Oromo block entropy estimation for $N = 0 - 20$

4.1.3 Conditional Entropy Estimation

We have calculated the conditional entropy based on the results of Table 4.2 and 4.3 and (2.8) for $N = 2 - 20$. Furthermore, the graphical interpretation of the the results is shown in Figure 4·2.

Table 4.4: F_N *in bits/block* for $N = 2 - 11$

Book No.	2	3	4	5	6	7	8	9	10	11
1	3.19	2.43	2.21	1.27	1.08	0.96	0.40	0.28	0.25	0.11
2	3.19	2.45	2.19	1.27	1.08	0.96	0.40	0.38	0.31	0.20
3	3.19	2.51	1.93	1.47	1.08	0.8	0.57	0.41	0.28	0.22
4	3.27	2.48	1.66	0.99	0.57	0.37	0.23	0.16	0.10	0.06
5	3.18	2.78	1.68	1.59	1.16	0.45	0.45	0.27	0.25	0.22
6	3.21	2.50	1.82	1.28	0.94	0.70	0.69	0.38	0.26	0.05
7	3.19	2.49	1.81	1.22	0.70	0.45	0.36	0.18	0.28	0.15
8	3.24	2.49	1.81	1.29	0.91	0.63	0.61	0.50	0.17	0.17
9	3.19	2.51	1.93	1.47	1.08	0.80	0.57	0.26	0.21	0.22
10	3.25	2.49	1.81	1.22	0.70	0.45	0.36	0.30	0.26	0.05
Mean	3.21	2.51	1.88	1.50	0.93	0.69	0.42	0.31	0.26	0.14

Table 4.5: F_N *in bits/block* for $N = 12 - 20$

Book No.	12	13	14	15	16	17	18	19	20
1	0.05	0.04	0.03	0.03	0.02	0.01	0.01	0	0
2	0.05	0.04	0.03	0.03	0.02	0.02	0.02	0	0
3	0.22	0.22	0.10	0.07	0.06	0.02	0.02	0.02	0.01
4	0.05	0.03	0.03	0.01	0.02	0.01	0.01	0	0
5	0.16	0.16	0.06	0.10	0.01	0.01	0.01	0.01	0
6	0.04	0.04	0.03	0.03	0.02	0.02	0.01	0	0.00
7	0.12	0.09	0.07	0.07	0.05	0.04	0.03	0.03	0
8	0.17	0.08	0.07	0.04	0.01	0.01	0.01	0.01	0.01
9	0.16	0.16	0.06	0.10	0.01	0.01	0.01	0.01	0.01
10	0.04	0.04	0.03	0.03	0.02	0.02	0.01	0	0
Mean	0.106	0.09	0.051	0.051	0.024	0.018	0.015	0.009	0.0039

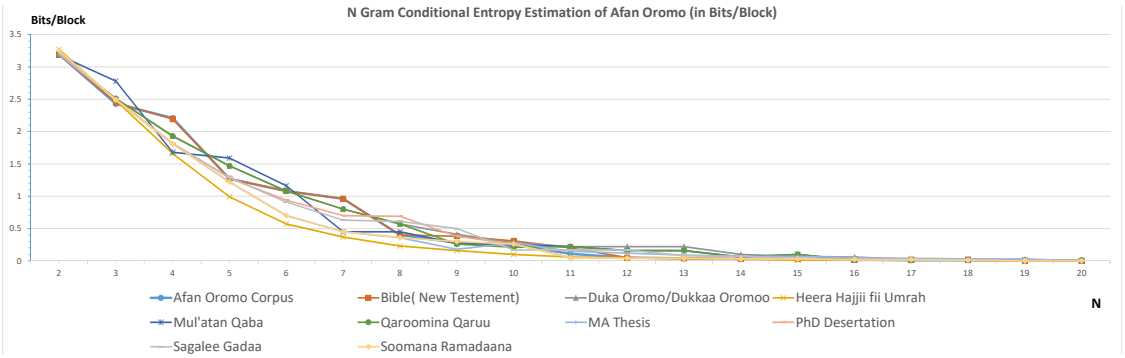


Figure 4·2: Graphical Interpretation of Affan Oromo conditional entropy estimation for $N = 2 - 20$

Based on Markovian chain modeling, entropy is a decreasing function of N. In this context, we say the entropy of a given natural language is properly captured when the estimation results approach zero, i.e., after this point we could say the language becomes completely predictable given N predecessor symbols. Thus, in this regard we aim at interpreting the results obtained in Tables (4.2,4.3) and Tables (4.4,4.5) by extracting the zero crossing entropy and estimating the redundancy of the language. To perform these analysis we have allowed a 1 % margin of error, i.e., we take two decimal points. To this end, we first extract the zero crossing entropy of the language by taking

the values of NG_N from Tables (4.2,4.3) that corresponds to an F_N approaching to zero. The values of NG_N and N are summarized in Table 4.6.

Table 4.6: Zero crossing entropy of the sample literature

NO	Name	N_z	$N_z G_{N_z}$
1	Afan oromo corpus	19	17.00
2	Bible	19	17.03
3	Duka oromo (dukkaa oromoo)	20	17.58
4	Herra hajjii fii umrah	19	14.30
5	Mal keba (Mul'kabaa)	20	17.89
6	Qaroomina Qaruu	20	17.01
7	PhD. Thesis	20	16.70
8	MA. Thesis	19	16.39
9	Sagl Geda (Sagalee Gadaa)	20	16.50
10	Soomana Ramadaana	19	16.39
Mean		19.5	16.68

With these results at hand we define the zero crossing entropy as [18];

$$H_z = \frac{N_z G_{N_z}}{N_z} \tag{4.1}$$

Where,

- N_z is the block size where F_N approaches zero.
- G_{N_z} is the block entropy at which F_N approaches zero.

Thus, with this definition and with the average estimate of Table 4.6 we can calculate the zero entropy of Oromo Language as;

$$H_z = \frac{16.68}{19.5} = 0.85 \; bits/symbol \tag{4.2}$$

Thus, when the Markovian chain structure of the language is fully captured at $N = 19.5$ the language only requires 0.85 bits/symbol on average. Furthermore, we can also estimate the redundancy observed in the language by taking the ratio of the zero crossing entropy and the entropy of the language when it is modeled as an iid

source [18]. Hence, the redundancy R_c becomes,

$$R_c = 1 - \frac{H_z}{H_{max}} \tag{4.3}$$

Where, H_max is the entropy of the language when it is modeled as an iid source (G_0).

Hence, for Oromo Language;

$$R_c = 1 - \frac{0.85}{7.99} = 89.36\% \tag{4.4}$$

Thus, for Oromo Language , if the language is modeled properly (i.e., at $N = 19.5$) one would only need to have the information about 10% of the symbols to complete the whole text. In other words, in terms of storage space, at the appropriate block size only 10% the data need to be saved and could successfully be recovered.

4.2 Evaluating the Estimations

To evaluate and confirm our estimations we have conducted entropy based encoding of the selected books. In this regard, we have selected Huffman and Arithmetic encoding as a tool. The summary of the compression results are summarized in Table 4.7 and 4.8 respectively. Furthermore, we have conducted the Huffman compression for using block sizes ranging upto five. Finally, we have summarized the Arithmetic Compression result as shown Table 4.9.

Table 4.7: Static Huffman compression of literature 1-7.

Book No.	Size in Bytes	Encoded Size in Bytes	N	Avg. Bits/Block	Comp. Ratio
1.	43,056,549	17,981,615	1	4.42	2.39
		14,092,412	2	7.62	3.05
		11,190,990	3	10.25	3.85
		10,090,232	4	12.45	4.26
		9,880,269	5	13.60	4.36
2.	1,540,542	983,040	1	4.39	1.567
		843,396	2	7.58	1.826
		746,693	3	10.03	2.060
		668,113	4	12.22	2.305
		604,106	5	13.49	2.550
3.	163,090	99,283	1	4.39	1.643
		86,213	2	7.58	1.892
		76,459	3	10.01	2.133
		68,274	4	12.22	2.389
		61,156	5	13.49	2.667
4.	21,280	13,019	1	4.25	1.634
		11,483	2	7.52	1.853
		10,144	3	10	2.098
		8,886	4	11.81	2.395
		7,706	5	14.08	2.761
5.	428,440	258,536	1	4.34	1.657
		225,720	2	7.52	1.898
		200,852	3	10.3	2.133
		179,799	4	11.98	2.383
		162,190	5	13.49	2.641
6.	108,752	65,716	1	4.37	1.659
		57,661	2	7.58	1.886
		51,184	3	10.08	2.125
		45,243	4	101.9	2.404
		40,125	5	13.18	2.710

Table 4.8: Static Huffman compression of literature 8-10.

Book No.	Size in Bytes	Encoded Size in Bytes	N	Avg. Bits/Block	Comp. Ratio
7.	130,460	78,605	1	4.37	1.659
		68,780	2	7.56	1.897
		61,076	3	10.05	2.136
		54,224	4	11.86	2.406
		48,086	5	13.49	2.713
8.	11,222	6,040	1	4.27	1.858
		5,339	2	7.8	2.102
		4,669	3	10	2.403
		4,111	4	11.81	2.729
		3,906	5	13.1	2.873
9.	158,345	93,263	1	4.39	1.698
		81,113	2	7.58	1.952
		56,254	3	10.09	2.815
		59,126	4	12.02	2.678
		52,555	5	13.49	3.013
10.	106,478	60,090	1	4.31	1.772
		52,228	2	7.56	2.039
		46,213	3	10.05	2.304
		38,274	4	11.86	2.782
		36,956	5	14.08	2.881

Table 4.9: Static Arithmetic compression of literature 1-10.

Book No.	Size in Bytes	Encoded Size in Bytes	Comp. Ratio
1.	43,056,549	23,259,082	1.851
2.	1,540,542	970,910	1.586
3.	163,090	98,489	1.655
4.	21,280	12,928	1.646
5.	42,8440	256,600	1.669
6.	108,752	65,278	1.665
7.	130,460	78,010	1.672
8.	11,222	7,525	1.490
9.	15,8345	94,322	1.678
10.	10,6478	64,412	1.653

The Huffman and Arithimetic compression results confirm the average bit/symbols estimated using the N-gram model. Moreover, we can also utilize these estimations to evaluate the redundancy estimation given in 4.4. In this regard, we will utilize

the results obtained by the Huffman algorithm for a compression block sizes ranging from $N = 1 - 5$. Moreover, we will define the redundancy of a language using the compression ratio as [18]. Thus, using this understanding we have computed the redundancy of the language in Tabel 4.10

$$R_c = 1 - \frac{1}{compressionRatio} \tag{4.5}$$

Table 4.10: Estimation of Oromo Language redundancy using Huffman compression technique for $N = 1 - 5$

NO	Name	R_c(N=1)	R_c(N=2)	R_c(N=3)	R_c(N=4)	R_c(N=5)
1	Afan oromo corpus	58.16	67.21	74.02	76.52	77.06
2	Bible	35.89	45.23	51.46	56.52	60.78
3	Duka oromo (dukkaa oromoo)	39.02	47.099	52.38	56.52	61.54
4	Herra hajjii fii umrah	38.82	46.04	52.33	58.24	63.79
5	Mal keba (Mul'kabaa)	39.66	47.31	53.12	58.03	62.14
6	Qaroomina Qaruu	41.10	48.77	64.47	62.66	66.81
7	PhD. Thesis	39.75	47.288	53.18	58.44	63.14
8	MA. Thesis	39.57	46.98	52.93	58.39	63.10
9	Sagl Geda (Sagalee Gadaa)	46.18	52.42	58.39	63.37	65.19
10	Soomana Ramadaana	43.56	50.95	56.59	64.05	65.29
Means		42.17	49.93	56.89	61.27	64.88

We have also summarized the redundancy estimated by the Arithmetic algorithm in Table 4.11. Both results are indeed in conformance with their theoretical redundancy estimation at the respective block sizes.

Table 4.11: Estimation of Oromo Language redundancy using Arithmetic compression technique $N = 1$

NO	Name	R_C (N=1)
1	Afan oromo corpus	45.97
2	Bible	36.95
3	Duka oromo (dukkaa oromoo)	39.58
4	Herra hajjii fii umrah	39.25
5	Mal keba (Mul'kabaa)	40.08
6	Qaroomina Qaruu	40.40
7	PhD. Thesis	40.19
8	MA. Thesis	39.94
9	Sagl Geda (Sagalee Gadaa)	32.88
10	Soomana Ramadaana	39.50
Mean		35.55

As a graphical comparison between the compression techniques we have given the bar graph given in Figure 4·3. We have removed "Oromo Language Corpus" and "Bible" because of the size of a corpora , so that the graph of the other literature becomes visible. In general, the Arithmetic compression technique is seen to beat the Huffman compression as expected.where as the compression

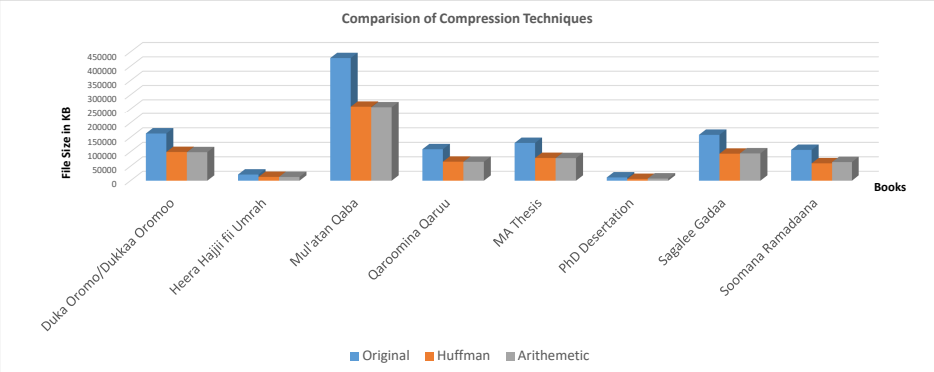


Figure 4·3: Graphical comparison of Huffman and Arithmetic compression techniques on the selected literature.

CHAPTER 5

SUMMARY AND CONCLUSION

We have set out to estimate the optimal average bits/symbol needed for written Oromo Language in the digital communication systems and storage space. In this context, we have estimated the entropy of the language using Markovian chain based source modeling. Furthermore, we have evaluated the quality of the estimation using entropy based encoding techniques, i.e., Huffman and Arithmetic. Our estimate shows that the language becomes completely predictable at $N = 19.5$ and at this block size the language entropy is estimated to require 0.85 $bits/symbol$ in average. Additionally, at this block size the language was found to have a redundancy of 89.36%.

We believe the outputs of this pioneer work can be used not only to develop compression systems for the language's text sources but also to broaden the research area of Oromo Language in statistical linguistics modeling, natural language processing (NLP), and even can act as a reference and a framework in developing other models for the language. As it has been discussed, the text source of the language contains a considerable redundancy and this should motivate development of source encoding applications. The applications can either be on the side of the telecom operator or be independent (such as android apps). We strongly believe this will minimize the cost of utilizing Oromo Language in the digital domain. For instance, nowadays bulk SMS business is becoming very common, and is expected to boom even more as subscriber number has not saturated yet and the business still looks at its infancy stage. Moreover, recently, the Ethiopian government invited interested operators to the telecom market which is expected to create more competition that in turn obliges the operators to look for efficient way of broadcasting SMS. And these facts make the integration and implementation of this work to the core SMS business of the telecom sector to be an important one. Furthermore, the other area this work could be of high importance is in context based modeling of the language.

REFERENCES

[1] Abbas El Gamal and Young-Han Kim. *Network information theory*. Cambridge university press, 2011.

[2] Kashfia Sailunaz, Mohammed Rokibul Alam Kotwal, and Mohammad Nurul Huda. "Data Compression Considering Text Files". In: *International Journal of Computer Applications* 90.11 (2014).

[3] David JC MacKay and David JC Mac Kay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[4] Claude E Shannon. "A mathematical theory of communication". In: *The Bell system technical journal* 27.3 (1948), pp. 379–423.

[5] Ben Purvis, Yong Mao, and Darren Robinson. "Entropy and its application to urban systems". In: *Entropy* 21.1 (2019), p. 56.

[6] Michał Stabno and Robert Wrembel. "RLH: Bitmap compression technique based on run-length and Huffman encoding". In: *Information Systems* 34.4-5 (2009), pp. 400–414.

[7] Ibrahim Bedane. "The Origin of Afaan Oromo: Mother Language". In: *Global Journal of Human-Social Science* 15.12 (2015), pp. 51–61.

[8] K Sayood. "Mathematical Preliminaries for Lossless Compression". In: *Introduction to Data Compression, 4th ed., Elsevier Inc* (2012), pp. 13–41.

[9] Workineh Tesema and Duresa Tamirat. "Enhancing the Text Production and Assisting Disable Users in Developing Word Prediction and Completion in Afan Oromo". In: *American Journal of Computer Science and Engineering Survey* 7.2 (2017).

[10] Temam Kedir Iteya. "Content Analysis of Faatee Blessing in Case of Arsi Oromo Peoples". In: *International Journal of Education, Culture and Society* 5.5 (2020), p. 95.

[11] John G Proakis. *Digital signal processing: principles algorithms and applications*. Pearson Education India, 2001.

[12] Claude E Shannon. "Prediction and entropy of printed English". In: *Bell system technical journal* 30.1 (1951), pp. 50–64.

[13] Khalid Sayood. *Introduction to data compression*. Morgan Kaufmann, 2017.

[14] Khalid Sayood. *Lossless compression handbook*. Elsevier, 2002.

[15] Tanvi Patel et al. "Survey of Text Compression Algorithms". In: *International Journal of Engineering Research & Technology (IJERT), india* (2015).

[16] Kashfia Sailunaz, Mohammed Rokibul Alam Kotwal, and Mohammad Nurul Huda. "Data Compression Considering Text Files". In: *International Journal of Computer Applications* 90.11 (2014).

[17] Athanasios Papoulis and S Unnikrishna Pillai. *Probability, random variables, and stochastic processes*. Tata McGraw-Hill Education, 2002.

[18] Tsegamlak Terefe and Dereje Hailemariam. "Entropy estimation and entropy-based encoding of written Amharic language for efficient transmission in telecom networks". In: *2017 IEEE AFRICON*. IEEE. 2017, pp. 238–244.

[19] Lev B Levitin and Zeev Reingold. "Entropy of natural languages: Theory and experiment". In: *Chaos, Solitons & Fractals* 4.5 (1994), pp. 709–743.

[20] John B Anderson and Seshadri Mohan. *Source and channel coding: an algorithmic approach*. Vol. 150. Springer Science & Business Media, 2012.

[21] Jonathan Perry, Hari Balakrishnan, and Devavrat Shah. "Rateless spinal codes". In: *Proceedings of the 10th ACM Workshop on Hot Topics in Networks*. 2011, pp. 1–6.

[22] Andrian Marcus and Jonathan I Maletic. "Recovering documentation-to-source-code traceability links using latent semantic indexing". In: *25th International Conference on Software Engineering, 2003. Proceedings.* IEEE. 2003, pp. 125–135.

[23] Jianhua Lin. "Divergence measures based on the Shannon entropy". In: *IEEE Transactions on Information theory* 37.1 (1991), pp. 145–151.

[24] Fabio Maselli, Claudio Conese, and Ljiljana Petkov. "Use of probability entropy for the estimation and graphical representation of the accuracy of maximum likelihood classifications". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 49.2 (1994), pp. 13–20.

[25] Oded Ghitza. "Auditory models and human performance in tasks related to speech coding and speech recognition". In: *IEEE Transactions on speech and audio processing* 2.1 (1994), pp. 115–132.

[26] Marta Karczewicz. *Variable length coding*. US Patent 6,696,993. 2004.

[27] James J Fallon and Steven L Bo. *System and method for lossless data compression and decompression*. US Patent 6,597,812. 2003.

[28] Neri Merhav, Gadiel Seroussi, and Marcelo J Weinberger. "Optimal prefix codes for sources with two-sided geometric distributions". In: *IEEE Transactions on Information Theory* 46.1 (2000), pp. 121–135.

[29] Lenore Blum, Manuel Blum, and Mike Shub. "A simple unpredictable pseudo-random number generator". In: *SIAM Journal on computing* 15.2 (1986), pp. 364–383.

[30] Konstantinos Krikellas, Stratis D Viglas, and Marcelo Cintra. "Generating code for holistic query evaluation". In: *2010 IEEE 26th International Conference on Data Engineering (ICDE 2010)*. IEEE. 2010, pp. 613–624.

[31] K Sayood. "Mathematical Preliminaries for Lossless Compression". In: *Introduction to Data Compression, 4th ed., Elsevier Inc* (2012), pp. 13–41.

[32] Mahnoosh Mehrabani, Srinivas Bangalore, and Benjamin Stern. "Personalized speech recognition for Internet of Things". In: *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*. IEEE. 2015, pp. 369–374.

[33] *Oslo Text Labratory*. `https://corpora.fi.muni.cz/habit/run.cgi/corp_info?corpname=orwac16/`. Accessed: 2019.

[34] Alemu Yadessa. *Duukaa Oromoo*. 2013.

[35] Jamaal Shekh and Muhammad Shekh. *Heera Hajjii fii Umrah*.

[36] Baqqalaa Garbaa. *Mul'tan Qaba*. 2015.

[37] Sukkaare Baqqalaa. "Qaaccessa Dhiyaannaa Xinjechaa:Kitaabilee Barnoota Afaan OromooKutaa Sagalfaafi Kurnaffaa Bara 2005 Qophaae Sukkaaree Baqqalaatiin Qophaaee Sadarkaa qorannoo Yunivarsiitii Addis Ababaa Kan guute tauu ni mirkaneessina." MA thesis. Addis Ababa: Kolleejjii Namoomaa, Qorannoo Afaanii, Joornaalizimifi Quunnamtii Yunivarsiitii Addis Ababa, 2016.

[38] Amanu'el Olijirraa. *Qroomina Qaruu*. 2020.