On the Entropy of Written Afan Oromo

Dereje Hailemariam Woldegebreal[1](✉), Tsegamlak Terefe Debella[1], and Kalkidan Dejenie Molla[2]

[1] School of Electrical and Computer Engineering (SECE), Addis Ababa University, Addis Ababa, Ethiopia
{dereje.hailemariam,tsegamlak.terefe}@aait.edu.et
[2] SECE, Debretabor University, Debretabor, Ethiopia
dejeniekal@gmail.com

Abstract. Afan Oromo is the language of the Oromo people, the largest ethnolinguistic group in Ethiopia. Written Afan Oromo uses Latin alphabet. In electronic communication systems letters in the alphabet are represented with standard ASCII-8 code, which uses 8 bits/letter, or UTF-8 fixed length encoding, which uses 16 bits/letter. Moreover, the language uses gemination (i.e., doubling of a consonant) and long vowels are represented by double letters, e.g., "*dammee*" to mean sweet potato. From information theoretic perspective, this doubling and fixed length encoding schemes add *redundancy* in written Afan Oromo. This redundancy, in turn, contributes for inefficient use of communication resources, such as bandwidth and energy, during transmission and storage of texts written in Afan Oromo. This paper aims at utilizing information theory to estimate entropy of written Afan Oromo. We use higher-order Markov chain, also called N-gram model, to compute the entropy of a sample text corpora (or written source) by capturing the dependencies among sequence of letters generated from the corpora. Entropy measures average information in bits per letter or block of letters, depending on the N-gram considered. Entropy also indicates the achievable lower bound for compression when using lossless compressions such as Huffman coding. When modeled as a first order Markov chain (i.e., assuming *memoryless* source where sequence of letters from the source are occurring independent of each other), the entropy of the language is 4.31 bits/letter. When compared with ASCII-8, the achievable compression level is about 46%. When $N = 19$ the estimated entropy is as low as 0.85 bits/letter; this corresponds to about 89% compression level. Huffman and Arithmetic source coding algorithms are implemented to check the achievable compression level. For the collected sample corpora, the average compression by Huffman algorithm varies from $42.2\% - 64.9\%$ for $N = 1 - 5$. These compression levels are closer to the theoretical entropy. With increasing demand of the language in telecom services and storage systems, the entropy results show the need to further investigate language specific applications, like compression algorithms.

Keywords: Compression · Entropy · Encoding · Written Afan Oromo

Y. H. Sheikh et al. (Eds.): AFRICOMM 2021, LNICST 443, pp. 25–46, 2022.
https://doi.org/10.1007/978-3-031-06374-9_3

1 Introduction

Afan Oromo is the language of the Oromo people, the largest ethnolinguistic group in Ethiopia. The 2007 official census of Ethiopia indicated that the Oromo people constitute about 34.6% of the total population, indicating that the language is spoken by more than one third of the Ethiopian population [1]. According to a recent decision by the Government of Ethiopia the language will soon become one of the five official working languages of the country [2]. Moreover, Afan Oromo is becoming one of the main communication languages in Ethiopia's telecom services, such as in short message services (SMS), and for digital documentations. From a continental perspective, the Oromo people also reside in Kenya and some parts of Somalia, making Afan Oromo as one of the most widely spoken African languages.

Qubee is the written form of Afan Oromo and a considerable portion of the language speakers are capable of understanding it. Qubee uses Latin alphabet and gemination (i.e., doubling of a consonant) and long vowels are represented by double letters; an example for doubling is the word "*dammee*" which translates to sweet potato [3]. Contrary to the large number of language users residing over a widespread geographic area as well as the socio-economic and political importance of the language, Afan Oromo is not well investigated from *information theoretic* point of view. Information theory helps to understand the language's statistical properties and develop language specific applications, e.g., source compression and text generation applications. The investigation requires having the right language *model*.

For written natural languages, such as written Afan Oromo, information source models are broadly categorized as *morphological* model and *statistical* model [4]. Morphological language models capture structure of a language at hand, which is defined based on linguistic knowledge. Statistical language models, on the other hand, are based on the language's statistics. One statistical model is *assigning probabilities* to letters or block of letters for a text corpora organized by collecting different written sources, for later use to compute entropy and design source coding algorithms.

Information theory-based research on natural languages, that utilize languages' statistics, has been the subject for more than 70 years. Early in the 1950's, in his groundbreaking paper Claude E. Shannon developed the N-gram as the basic statistical language model for written natural languages [4, 11]. According to Shannon's work [5]:

- Discrete information sources, such as written natural languages, can be modeled as generating messages on a letter-by-letter basis in a random manner. In latter investigations, such as in text generations, sources are assumed to generate *block of letters* or *words* at a given instant of time.
- Each letter is generated according to certain probability distribution depending, in general, on preceding letters, as in N-gram, where the N indicates the dependency of a letter on previous N sequence of letters. The N-gram is also called higher-order Markov chain. Discrete memoryless sources (DMS) are special cases of N-gram model when $N = 1$. If the source generates words, then N indicates previously generated words.

Shannon has showed ways to estimate entropy of English language [5, 11]. Later on David Albert Huffman, a student of Shannon, showed that one can design source encoders that closely approach the estimated entropy [6]. Subsequent researches focused on technologically favored languages like English, French, Russian, Spanish, German and Japanese [4]. Regarding Ethiopian languages, [4] took the initiative to investigate entropy and entropy-based compression for written Amharic language, another popular language in Ethiopia [4]. Though non-entropy based, [13] proposed using Prediction by Partial Matching (PPM) algorithm to compress written Amharic language in Table 1 summarizes results of previous studies.

Table 1. Entropy estimations of different natural languages [4].

N	Russian	French	Rumanian	Hebrew	Arabic	Portuguese	German	Amharic
1	5.01	4.76	4.76	4.52	5.00	4.76	4.70	5.27
2	4.35	3.90	–	4.12	4.20	3.90	4.10	4.20
3	3.52	–	–	3.71	3.80	3.50	–	2.78
4	3.00	2.83	2.69	3.12	2.5	3.10	2.95	1.99

This paper investigates Afan Oromo, the next popular language in Ethiopia and one of the cross-border languages in the African continent. Afan Oromo has been studied in a limited manner in areas of natural language processing such as part of speech tagging [7], automatic classification of news text, word sequence prediction, machine translation and the likes. A multitude of related researches are available at the Addis Ababa University Institutional Repository [8].

Although the mentioned researches might have contributed their parts to bring Afan Oromo to the attention of natural language processing, to the best of our knowledge no prior study is made that establishes entropy measure and apply source encoding (compression) for the written form of the language. Aside the mere desires to investigate Afan Oromo using tools from information theory, key observations that motivate us to investigate the language are summarized below.

- Because of the language's structure and grammatical rules, the written form of the language contains repetitive/redundant letters, especially vowels. An example is the word "Shaampiyoonaan" which is Oromo Language word for champion. A second example is a sample SMS shown in Fig. 1 that indicates the degree of redundancy in vowels and consonants because of gemination. While acknowledging the redundancy requirement because of the language's structure, from information theory perspective this redundancy contributes to inefficient transmission and storage in communication systems. The 4.31 bits/letter entropy result we obtained for a first order Markov chain assumption is an indicative of the redundancy level.

Sa'aa murtaahe qofatu hafe!
Abbaa SUUQII! Abbaa
qabeengnaa taaa! LIDATAA
GIDDU GALEESSSA GABAA MARKAATOOTTI SUUQII
mohadhaa!Ammaan tana ergaa! MILKAAHA!

Fig. 1. Sample Afan Oromo SMS message.

- Like other Latin alphabet-based languages, UTF-8 and ASCII-8 fixed length encodings are used to represent the language's letters for electronic communication. While such representations are good for standard communication, both encodings poorly model written natural languages in general, as the actual entropy is much lesser that the bits used in these representations. See Table 1. Written Afan Oromo is not an exception to that.
- Nowadays written Afan Oromo is becoming one of the main communication means in Ethiopia's telecom services and huge traffic is expected to be generated with this language as a source. Moreover, given the explosion of over-the-top (OTT) services, where data compression is one backbone, developing resource-efficient and language-specific OTT services may not be far-fetched imagination. With knowledge of the language's entropy, more OTT services can be developed.

This paper sets to achieve the above two objectives, i.e., investigating entropy and entropy-based compression algorithms of written Afan Oromo. To achieve these objectives, we first formed corpora from ten literatures, then estimated the language's entropy and finally investigated achievable compression level using Huffman and Arithmetic encodings.

The rest of the paper is organized into seven sections. Section two presents source models, source coding and review of entropy estimation. Then, basic concepts in Markov chain are covered in Section three. Section four slightly talks about two of the most common lossless compression algorithms; namely, Huffman coding and Arithmetic coding. Sections five presents methodology used in this work. Finally, results and discussion are presented in Section six, followed by conclusion in Section seven.

2 System Model, Entropy Estimation and Source Encoding

2.1 Written Afan Oromo

Qubee is defined using Latin alphabets that consist of thirty-three basic symbols representing distinct sounds [3]. Qubee utilizes capital and small letters, the vowels are sound makers and are sound by themselves, as is in English. Afan 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 (see the text in Fig. 1). Furthermore, sometimes an "H", which represents the closest glottal sound, is used in place of an apostrophe. The apostrophe will be considered as a distinct symbol (say, as space is considered as

the 27th letter of written English). Additionally, in Afan Oromo writing system, geminated consonants and long vowels are represented by double letters in addition to seven compound symbols. Figure 2 shows Qubee alphabets.

2.2 Information Source Model

Information source and source coding are two key building blocks for a generic communication system [9]. These two blocks are the focused of this paper; information source modeling is the focus of this subsection while source encoding is explained in the next subsection.

A generic information source randomly generates signals that represent voice, text, data or multimedia service. The information source can be either discrete in time with finite (countable) alphabet (amplitude space), such as written languages, or analog (continuous in time with uncountable alphabet) such as audio or video. For analogue sources like voice, generated signals are converted to discrete samples by properly applying the Nyquist Sampling Theorem, followed by quantization [9].

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]	[tʃ']	[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]	[p']	[k']	[r]	[s]	[ʃ]	[t]	[ʊ]	[v]
W w	X x	Y y	Z z	'				
wa	xa	ya	za					
[w]	[t']	[j]	[z]	[ʔ]				
aa	ee	ii	oo	uu				
[ɑː]	[eː]	[iː]	[oː]	[uː]				

Fig. 2. The Qubee alphabets.

Depending on the statistical dependency among successive sequence of letters, the information source can be modeled as memoryless (also called *discrete memoryless source*) or as *Markov chain* for a source having memory.

Discrete Memoryless Source: it is assumed that the current output letter is statistically independent of all past and future outputs. Thus, two possible statistical directions are taken to model the source. In the first approach, the discrete memoryless source is assumed to generates a sequence of independent and identically distributed (i.i.d.) letters taking values form alphabet, say $\mathcal{A}$ [10]. This is the approach utilized in most of today's fixed length encoding systems such as ASCII and UTF-8. In an alternative approach, if the probability of each letter is known, a discrete memoryless source can model sequence of letters as having statistical independence, but distributed according to the nature of the source at hand.

Markovian Information Source: in reality natural languages are Marko chain where each current output letter (word or block of letters which may or may not be meaningful) of the source is dependent on the previous sequence of output letters (or words).

For our work, the information source is a written Afan Oromo language source that randomly emits sequence of letters drawn from Qubee alphabet, based on a message written in Afan Oromo. Randomness observed in the output letters is dictated by factors such as language's structure, context and the writer's style of writing. We take note of the following two key points: first at a given instant the source's outputs could be a letter or word. Second, the statistical dependencies among successive letters or words follow the memoryless or Markovian assumption.

In summary, the information source modeling in our work results in two things: assigning probability distribution for each letter or block of letters taking the statistical dependency into account and, then computing entropy of the source. This is identical to Shannon's approach in modeling language sources [5]. To extract the statistical distributions (or probabilities), one can utilize corpora i.e., a collection of texts written with a specific language's symbol set [4]. For discrete memoryless sources computing the probability is easier, thanks to the statistical independency. However, Markov chain is used to simplify probability computations for sources whose output letter or words are dependent. The flowchart in Fig. 4 explains the approach used in this paper.

2.3 Source Encoding

With knowledge of source's entropy, source encoding (also known as compression[1]) optimally represents each letter (or word) with sequence of bits [9]. Codeword is a term coined to name the sequence of bits. Entropy shows a lower bound for the average codeword length to ensure error free and efficient transmission of bits in a communication system [9]. The optimality in source coding is meant to say that, if the average codeword length is below the entropy, data loss will occur during source decoding. Such encoding/compression scheme is called *lossy compression* [10]. On the other hand, if the average codeword length exceeds the entropy, there is no loss in information but efficient resources utilization is undermined, as redundant information is further processed by the communication system [9]. Such data compression is termed as *lossless compression* [10]. Typical application requiring a lossless compression is written text, as in written Afan Oromo language [10]. Options to achieve optimal representation of source letters are: assigning variable length codewords based on knowledge of probability of occurrence of letters and mapping block of letters to a codeword, instead of mapping a letter to a codeword [9].

[1] Source coding is a term used communication engineering literatures, while compression is terms used computer science and information systems literatures. In this paper we may be using these terminologies interchangeably.

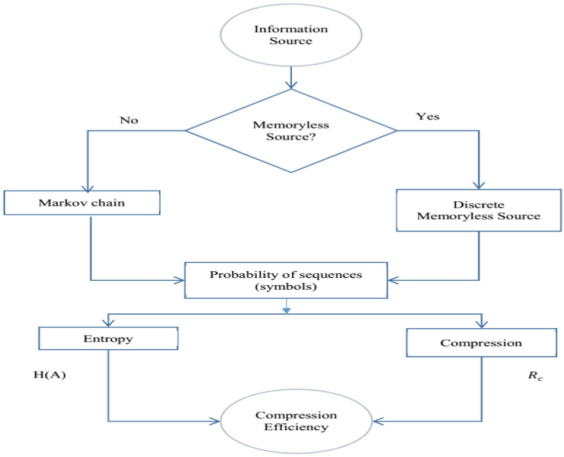

Fig. 3. Steps taken to model and study written natural languages.

For the purpose of this work, Huffman encoding and Arithmetic encoding algorithms are used to see the level of lossless compression we can achieve for written Afan Oromo language.

2.4 Entropy and Source Coding for Discrete Memoryless Sources

If we assume that the discrete information source generates random letter (in our case letters for written Afan Oromo) on a letter-by-letter basis and drawn from a finite alphabet set $\mathcal{A}$,

$$\mathcal{A} = \{a_0, a_1, \ldots, a_{N-1}\}$$

of size N, with probabilities

$$P(\mathcal{A} = a_k) = p_k,\ k = 0, 1, \ldots, N-1$$

In the followings we present fundamental concepts from information theory to quantify the amount of information emitted by the source and the degree of compression we can expect by using efficient source encoding techniques.

Self-information: or the Shannon information content associated with a letter a_k, which occurs with probability p_k, is given by:

$$I(a_k) = \log_2 \frac{1}{P(A = a_k)} = \log_2 \frac{1}{p_k} = -\log_2 p_k [bits] \tag{1}$$

We note that letters that are highly likely to occur contain lesser information and vice versa [9].

Entropy: of the discrete information source, designated as $H(A)$, measures the average (expected) information content per letter and is given by:

$$H(A) = \mathrm{E}[I(a_k)] = \sum_{k=1}^{N} p_k I(a_k) = -\sum_{k=1}^{N} p_k \log_2 p_k \left[\frac{bits}{letter}\right] \tag{2}$$

where $\mathrm{E}[I(a_k)]$ is the expected value of the self information $I(a_k)$.

Variable-Length Codes: for a letter a_k from the alphabet $\mathcal{A}$, if the source encoder assigns a codeword of length l_k, measured in bits, the average codeword length $\overline{R}$ is:

$$E[l_i] = \overline{R} = \sum_{i=0}^{N-1} p_i l_i \tag{3}$$

Source Coding Theorem –Shannon's source coding theorem states that for an optimal source encoder, the average codeword length, $\overline{R}$, will be within one bit from the entropy, i.e.,

$$H(A) \leq \overline{R} \leq H(A) + 1 \tag{4}$$

If $\overline{R} < H(A)$, the error probability will be bounded away from zero, independent of the complexity of the source encoder and decoder employed [9]. According to the theorem, entropy $H(A)$ of the source gives a lower bound in the rate at which the source can be compressed for reliable reconstruction.

Shannon's theorem shows that a symbol that is expected to occur with probability p is best represented in $-\log_2 p$ bits. In this manner a symbol with a high probability of occurrence is coded with fewer number of bits as compared to unlikely (low probability) symbols. Such assignment of codes produces variable-length codes. Moreover, Source Coding Theorem also states that if a random source generates M long and successive letters, the M letters can be roughly compressed to $MH(A)$ bits.

2.5 Source Model for Block of Letters

Assume that at a time the information source generates block of letters of length k, where the block may or may not be a meaningful work, and designated as $X = \{S_1, S_2, S_3, \ldots., S_k\}$. Each element of the sequence takes values from the alphabet $\mathcal{A}$ i.e., $S_i \in \{a_0, a_1, \ldots, a_{N-1}\}$. For a discrete memoryless source, the probability of the block is:

$$P(X) = P(S_1, S_2, S_3, \ldots., S_k) = \prod_{i=1}^{k} P(S_i) \tag{5}$$

where $P(S_i)$ is the probability of the letter S_i. On the contrary, for a Markovian source the block probability is given by:

$$P(X) = P(S_k|S_1, S_2, S_3, \ldots., S_{k-1})P(S_1, S_2, S_3, \ldots., S_{k-1}) \tag{6}$$

where,

$$P(S_1, S_2, S_3, \ldots., S_{k-1}) = P(S_{k-1}|S_1, S_2, S_3, \ldots., S_{k-2})P(S_1, S_2, S_3, \ldots., S_{k-2}) \tag{7}$$

Researchers have shown that in written natural languages, each letter is dependent on the occurrence of pervious N letters, which leads to N-gram modeling or higher-order Markov chain [4, 17]. In the following section, we will first explain key points from first-order Markov chain and higher-order or N-gram Markov chain.

3 Markov Chain for Discrete Sources

3.1 First-Order Markov Chain

First-order Markov chain, usually called Markov chain only, is a kind of stochastic model used to model randomly varying systems, like information source, whose future output has dependency only on current output only. This dependency on the current state only is sometimes characterized as "*memorylessness*". State, initial distribution, transition probability, transition matrix, and steady state distribution are few of common terms in Markov chain study and are briefly presented below.

State in Markov chain: assumes that real life systems are made of *states*, e.g., each letter generated from the information source can be viewed as a state. The state space is discrete (usually) and countable (finite state space). The system randomly transitions in discrete time from one state to another state according to certain *transition probability*. To formulate mapping between states in Markov chain and letters generated from an information source, suppose we have a discrete-time information source that generates sequence of letters $S_1, S_2, S_3, \ldots., S_n, \ldots$, where n in S_n indicates the discrete time index and $S_n \in \{a_0, a_1, \ldots, a_{N-1}\}$. drawn from the alphabet $\mathcal{A}$, as defined in Subsect. 2.5 above.

In Markov chain analogue we can think of S_n as one of those states of the information source, and every time the information source generates a letter, we can think of the chain is moving/transitioning from one state to another state in a random manner. For the information source of alphabet size $N-1$, the state space has $N-1$ discrete states. For simplicity, assume that the states are numbered from $0, 1, 2, \ldots, N-1$. Then $S_n = j$ for c can be interpreted as at time n the chain is in state j.

Transition Probability: indicates the probability with which the chain transitions in discrete time from one state to another state. The transitions represent/capture hidden patters in the system, e.g., transitions from one letter to the next letter in a text is used as an approximation of the true underlying nature of the language. For the first order Markov chain fulfilling the memorylessness property, the Markov property is expressed as:

$$\mathrm{P}(S_{n+1} = j|S_n = i, S_{n-1} = i-1, \ldots, S_0 = i_0) = \mathrm{P}(S_{n+1} = j|S_n = i) \tag{8}$$

We learn from the Markov property that only the most recent state (information) matters to predict the next or future state. Alternatively, the past and the future are conditionally independent given the present. From now on, the transition from state i to state j in:

$$P_{ij} = \mathrm{P}(S_{n+1} = j | S_n = i) \tag{9}$$

Transition Probability Matrix: collection of the transition probabilities P_{ij} form the *probability transition matrix*, P. Each element of the matrix represents the probability that the system switches its state or remains in the same state. Every state in the state space is included once as a row and again as a column [8]. P is a square matrix whose order is the same as the number of states.

Initial Distribution: of the states indicates probabilities of different states at a given (initial) time instant.

Steady State Distribution - One of the interesting things about systems that obey Markov chain is that, from which ever initial states the chain starts from, the chain will converge to *steady state*, *stationary*, *equilibrium* or *static* distribution after sufficiently large iterations/transitions. In steady state, the chain does not stop from transitioning; rather, from a given state the probability of outward transitions and inward transitions will be the same (or is in equilibrium) so that the probability distribution stays the same.

Theorem: Let P be the transition matrix of a Markov chain, and let U be the probability vector which represents the starting/initial distribution. Then, the probability distribution of the chain after k transition in the future is given by [2].

$$U^{(k)} = UP^k \tag{10}$$

P^k is the result of multiplying P k times by itself. Each element of $U^{(k)}$, designated as $P_{ij}^{(k)}$, is the probability of going from state i to state j in k iterations or transitions. As we keep iterating through state transitions by applying P^k, the probability vectors $U^{(k)}$ converge to some fixed value. That is called the *steady state probability* vector or an *equilibrium distribution.*

We note from (10) that based on knowledge of the state space, initial distribution and transition matrix, Markov chain can predict/estimate, in probabilistic sense, the future state of the system after certain transitions in time.

3.2 Higher-Order Markov Chain

While the dependency of future state on present state only, i.e., the memorylessness assumption, is valid for many sources, there are sources where the future state depends on $N - 1$ past and present states. This leads to the name Higher-order Markov chain or N-gram model [16]. As an example in natural languages, because of hidden structures in the language and grammatical rules, the likelihood of a letter generated from a source is dependent on previously generated $N - 1$ contiguous sequence of letters. [In most

natural language processing studies, the N-gram indicates sequence of meaningful words instead of letters.]

We can think of N-gram as sequence of N letters, and a statistical model assigns probabilities to the sequences of letters. Common values of N have special names: a 1-g (or unigram) is a one-letter sequence of letters; a 2-g (or bigram) is a two-letter sequence of letters; and a 3-g (or trigram) is a three-letter sequence of letter and the likes. The below table show various sequences for the word "entropy".

N = 1	Unigram	"e", "n", "t", "r", "o", "p", "y'
N = 2	Bigram	"en", "nt", "tr", "ro", "op", "py"
N = 3	Trigram	"ent", "ntr", "tro", "rop", "opy"

The N-gram assigns probabilities to letters and sequence of letters, considering the dependency among the letters. As an example, if we have three sequence of letters S_1, S_2, S_3 where $S_i \in \{a_0, a_1, \ldots, a_{N-1}\}$ drawn from the alphabet $\mathcal{A}$. The N-gram model assigns probability to the sequence as

$$P(S_1S_2S_3) = P(S_1)P(S_2|S_1)P(S_3|S_1S_2) \tag{11}$$

Unigram Model: writes the likelihood in (11) in terms of the probabilities of the three letters as:

$$P_{uni}(S_1S_2S_2) = P(S_1)P(S_2)P(S_3) \tag{12}$$

One way to estimate the probability in (12) is through the relative frequency count approach by taking a substantially large corpus. If M is the total number of three-letter sequences count in the corpus, the maximum likelihood estimate of the sequence $S_1S_2S_2$ is computed from the number of counts as:

$$P_{uni}(S_1S_2S_2) = \frac{count(S_1S_2S_2)}{M} \tag{13}$$

N-gram Model: in a case that the sequence has large number of letters, getting a good estimate for the probability of the sequence is a bit difficult, as the training corpora may not be sufficient enough to bring out all the possible combinations of letter sequence in the source [4]. One approach to simplify the probability estimate is to combine chain rule and the N-gram assumption, so that probability of observing sequence of letters $S_1, \ldots, S_h$ is approximated as [16]:

$$P(S_1, \ldots, S_h) = \prod_{i=1}^{h} P(S_i|S_1, \ldots, S_{i-1}) \approx \prod_{i=1}^{h} P\big(S_i|S_{i-(N-1)}, \ldots, S_{i-1}\big) \tag{14}$$

Given a large corpora, the maximum likelihood estimation of the conditional probabilities in (14) are computed via the frequency count as:

$$P\big(S_i|S_{i-(N-1)}, \ldots, S_{i-1}\big) = \frac{count\big(S_{i-(N-1)}, \ldots, S_{i-1}, S_i\big)}{count\big(S_{i-(N-1)}, \ldots, S_{i-1}\big)} \tag{15}$$

Entropy Estimation: to estimate the self-information in (1), we need to substitute the probability p_k in (1) with $P_{B_j}(S_i)$, which is the probability of the letter S_i given the past history, i.e., $N-1$ sequence of $B_j = S_{i-(\mathrm{N}-1)}, \ldots, S_{i-1}$ letters. In this context, if we define:

$$\begin{aligned} P_{B_j}(S_i) &= P\left(S_i | S_{i-(\mathrm{N}-1)}, \ldots, S_{i-1}\right) \\ \text{and} \quad P\left(B_j\right) &= P\left(S_{i-(N-1)}, \ldots, S_{i-1}\right) \end{aligned} \tag{16}$$

Moreover, the joint probability of the history and the letter S_i is:

$$P\left(S_{i-(N-1)}, \ldots, S_{i-1}, S_i\right) = P\left(B_j, S_i\right) \tag{17}$$

With this definition the information content of a Markovian Higher order Markovian modeled source becomes [4]:

$$I(S_i) = \log_2 \frac{1}{P(S_i)} = -\sum_j P\left(B_j, S_i\right) \log_2 P_{B_j}(S_i)[bits] \tag{18}$$

Thus, the average conditional entropy is the expectation of the self-information:

$$H = F_N = \mathrm{E}[I(S_i)] = -\sum_i \sum_j P\left(B_j, S_i\right) \log_2 P_{B_j}(S_i) \tag{19}$$

From practical perspective, the conditional probability $P_{B_N}(S_i)$ is estimated using Bayesian theorem, i.e.,

$$P_{B_j}(S_i) = \frac{P\left(B_j, S_i\right)}{P\left(B_j\right)} \tag{20}$$

Thus,

$$H = \lim_{N \to \infty} F_N = \mathrm{E}[I(S_i)] = -\sum_i \sum_j P\left(B_j, S_i\right) \log_2 \frac{P\left(B_j, S_i\right)}{P\left(B_j\right)} \tag{21}$$

$$F_N = -\sum_i \sum_j P\left(B_j\right) \log_2 P\left(B_j, S_i\right) + \sum_i \sum_j P\left(B_j, S_i\right) \log_2 P\left(B_j\right) \tag{22}$$

Since,

$$F_N = -\sum_i \sum_j P\left(B_j, S_i\right) \log_2 P\left(B_j, S_i\right) + \sum_i P\left(B_j\right) \log_2 P\left(B_j\right) = NG_N - (N-1)G_{N-1} \tag{23}$$

Where

$$G_N = -\frac{1}{N} \sum_i P\left(B_j\right) \log_2 P\left(B_j\right)$$

Equation (23) is simpler to implement, as it allows a successive approximation to the optimal (minimum) entropy.

4 Source Coding for Discrete Information Source

4.1 Huffman Coding

The basic idea of behind Huffman coding is to assign each letter a_k of the alphabet $\mathcal{A}$ a sequence of bits, called codeword, whose length roughly equals to the amount of the self information of the letter, $-\log_2 p_k$. The end result is a uniquely decodable code whose average codeword length approaches the fundamental limit set by the entropy of the source [9]. The procedure to build the Huffman encoding is available in [9].

4.2 Arithmetic Coding

Arithmetic coding is a direct consequence of Shannon's source coding theorem which established that the number of bits per letter needed to encode the output of a source is arbitrarily close to the entropy of the source [10]. In order to prove his theorem, Shannon considered the following abstract coding scheme: List all likely sequences of letters and assign to each a binary code with the more probable sequences being assigned shorter codewords. The practical implementation of this coding method is called Arithmetic coding. Arithmetic coding is especially useful when dealing with sources with small alphabets, such as binary sources, and alphabets with highly skewed probabilities (a good example can be written text sources). Additional details about the coding scheme are available in [14] and other references.

5 Methodology

The methodology used in the course of this research follows the steps in Fig. 3 and the flow chart in Fig. 4 and comprises of corpora collection and formatting; computing probability and conditional probabilities for letters/symbols or sequence of letters/blocks of letters/symbols; estimating entropy of the source based on probabilities; implementing and testing entropy-based source encoding algorithms over the sample space; and measuring the efficiency of compression using. Brief description of the three tasks is indicated in the three subsequent subsections.

5.1 Corpora Collection and Formatting

In the process of building the corpora, corpus from the Oslo Text Laboratory [15] and ten literatures written in Afan Oromo were used (see Table 2). The collected literatures fall in different categories, i.e., political, historical, religious, scientific papers and a corpus prepared for Afan Oromo sentence tagging. 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 reedited the corpus by removing tokenization symbols. As more books become available in the future, the extracted frequencies and entropy estimation will be more refined.

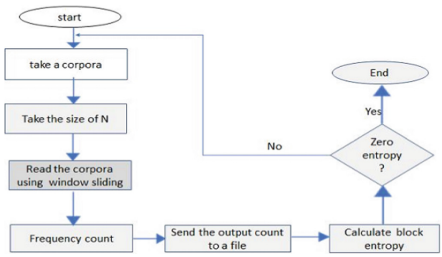

Fig. 4. N-gram based entropy estimation.

Table 2. Selected corpora.

No	Book (Corpora) Title	Authors	Category
1	Afan oromo corpus	Oslo Text Lab	Different Texts
2	Bible	Various	Religious
3	Duka oromo (dukkaa oromoo)	Alemu Yadessa	History
4	Herra hajjii fii umrah	Shekh Jamaal	Religious
5	Mal keba (Mul'kabaa)	Bekele Gerba	Political
6	PhD. Thesis	Taaddasaa Dirriba	Research
7	MA. Thesis	Sukkare Baggle	Research
8	Sagl Geda (Sagalee Gadaa)	Caalaa Soorii, et al	Journal
9	Qaaroomina Qaruu	Amanuerl Olijirra	Scientific
10	Soomana Ramadaana	Hera, et al.	Religious

5.2 Probability Extraction and Entropy Estimation

The entropy of Afan Oromo is estimated using frequency table list of sequence of letters. Blocks are defined here to be unique sequence of letters observed in the corpora, as demonstrated in Fig. 5. 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 letters. However, if we increase the block size and use a sliding as shown in Fig. 5 we are able to extract frequency counts for more than fifteen unique letters sequences. Python is used to implement this principle.

The script counts the occurrence of each unique sequence of letters, and then estimates the probabilities of the sequence using the law of large numbers, i.e.:

$$P_{B_N} = \frac{Count\ of\ B_N}{Number\ of\ unique\ symbols} \tag{24}$$

where B_N is a sequence (block) [12]. Following this estimations, the modified Shannon's estimation in (23) is used for the conditional entropy estimations.

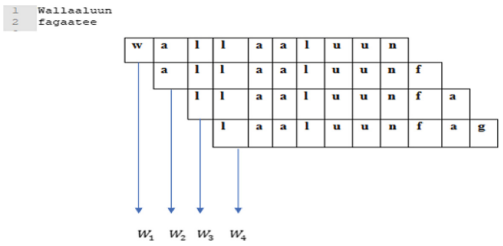

Fig. 5. Demonstration of sliding window for N = 1.

5.3 Source Encoding

We have developed a Python script to implement Huffman and Arithmetic encoding algorithms. The implementations were then tested on the selected sample books. The encoding schemes are intended to reveal how close we can get to the estimated entropy. This way we also provide a way of crosschecking the entropy estimation in (23).

6 Results and Discussion

Equation (23) is used to estimate the block and conditional entropy of written Afan Oromo. We have summarized the estimation results in Tables 3 and 4 for the block entropy; these tables are plotted in Fig. 6. Moreover, Tables 5 and 6 are for the conditional entropy and the plots is given in Fig. 7. In these tables, the results for N = 0 and 1 indicates the entropy of the source given it is modeled as an i.i.d and first order Marko chain (for discrete memoryless source). Moreover, we drive the bit requirement for N = 0 by taking the average of unique symbols seen within the corpora.

Table 3. 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	10.00	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	15.90	16.15
6	7.99	4.39	7..58	10.09	12.02	13.49	14.57	15.37	15.94	15.90	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	11.90	13.18	14.12	14.82	15.51	15.89	16.15
9	7.99	4.27	7.51	10.00	11.81	13.10	14.01	14.99	15.20	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	13.79	14.68	15.38	15.76	16.05	16.43

Table 4. 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	16.20	16.24	16.28	16.31	16.34	16.36	16.38	16.39	17.39	17.39
7	16.20	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

According to Markov chain modeling, conditional entropy is a decreasing function of N as shown in Fig. 7. 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.

Table 5. F_N *in bits/symbol* 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 6. F_N *in bits/symbol* 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

Thus, in this regard we aim at interpreting the results obtained in Tables 5 and 6 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 NGN from Tables 3 and 4 that corresponds to an FN approaching to zero. The values of NGN and N are summarized in Table 7. With this results at hand the we can define the zero cross entropy as [4];

$$H_z = \frac{\widehat{N_z}\widehat{G_{N_z}}}{\widehat{N_z}} \, bits/symbol \tag{25}$$

Where, $\widehat{N_z}$ is the average block size at which NG_N approaches zero. $\widehat{G_{N_z}}$ is the average block entropy at which NG_N approaches zero. Using Eq. (25) and with the average estimate of Table 7 we can calculate the zero entropy of Afan Oromo Language as:

$$H_z = \frac{16.68}{19.5} == 0.85 \, bits/symbol$$

Thus, when the Marko chain structure of the language is fully captured, i.e., at N = 19.5, the language only requires 0.85 bits/symbol on average. Furthermore, we can also utilize H_z estimate the redundancy observed in the language using (26) as [4].

$$R_c = 1 - \frac{H_z}{H_{max}}\% \tag{26}$$

where, H_{max} is the entropy of the language when it is modeled as an i.i.d source.

Table 7. 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

The redundancy of a language estimates the maximum amount of compression that could be achieved without the loss of information. In this regard, the redundancy of written Afan Oromo as,

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

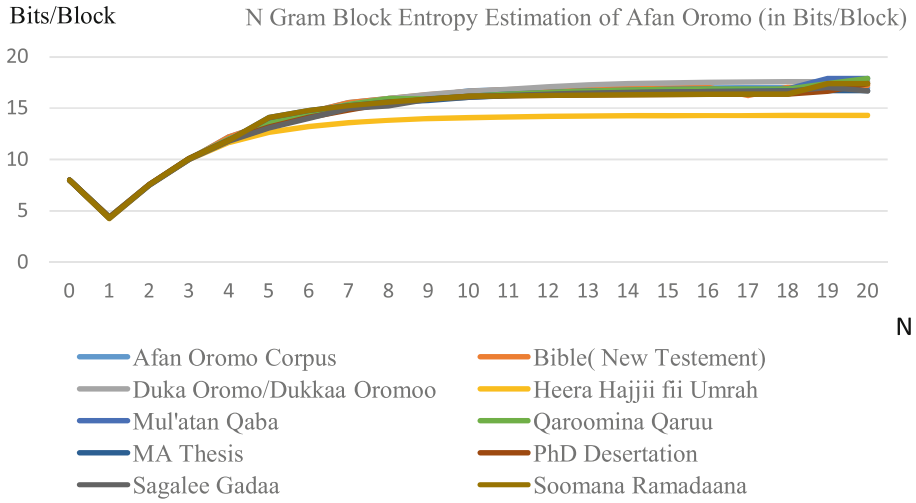

Fig. 6. Block entropy estimation of Afan Oromo for $N = 0$–20.

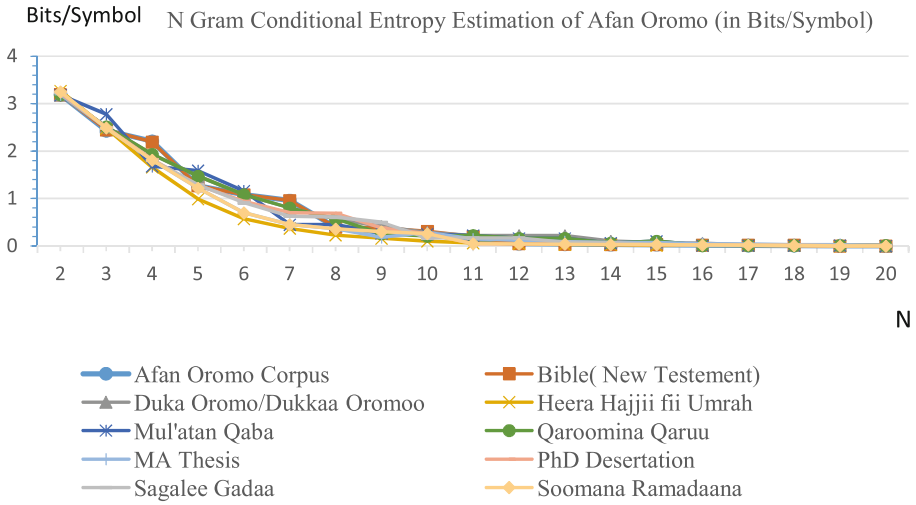

Fig. 7. Conditional entropy estimation of Afan Oromo for N = 2–20.

Thus, based on this estimation, if modeled properly Afan Oromo could be compressed in a lossless manner with only 10.64% of the total bit streams used by an i.i.d model (fixed length encoding). 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 Huffman compression results are given in Table 8, whereas Table 9 summarizes the arithmetic compression. We have conducted the Huffman compression using a block sizes ranging up to five.

The compression results confirm the average bit/symbols estimated using the N-gram model.

Table 8. Huffman compression of literature 1–10.

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	2.390
		14,092,412	2	7.62	3.050
		11,190,990	3	10.25	3.850
		10,090,232	4	12.45	4.260
		9,880,269	5	13.60	4.360
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	10.00	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
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	7.80	2.102
		4,669	3	10.00	2.403
		4,111	4	11.81	2.729
		3,906	5	13.10	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

We can use these compression results to define the redundancy of the language using Eq. (27) [4]. The result of this computation is summarized in Table 10.

$$R_c = 1 - \frac{1}{Compression\,Ratio} \tag{27}$$

The results in Table 10 are in conformance with their theoretical redundancy estimation at the respective block sizes. For instance, if we take the average entropy estimation for N = 1 and compute redundancy we would get a redundancy of 45.56%.

Table 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

Table 10. Estimation of compression ratio using the compression techniques.

Book No.	R_C (N=1) (Arithmetic)	R_c(N=1) (Huffman)	R_c(N=2) (Huffman)	R_c(N=3) (Huffman)	R_c(N=4) (Huffman)	R_c(N=5) (Huffman)
1	45.97	58.16	67.21	74.02	76.52	77.06
2	36.95	35.89	45.23	51.46	56.52	60.78
3	39.58	39.02	47.10	52.38	56.52	61.54
4	39.25	38.82	46.04	52.33	58.24	63.79
5	40.08	39.66	47.31	53.12	58.03	62.14
6	40.40	41.10	48.77	64.47	62.66	66.81
7	40.19	39.75	47.288	53.18	58.44	63.14
8	39.94	39.57	46.98	52.93	58.39	63.10
9	32.88	46.18	52.42	58.39	63.37	65.19
10	39.50	43.56	50.95	56.59	64.05	65.29
Means	35.55	42.17	49.93	56.89	61.27	64.88

7 Conclusion

We have set out to estimate the optimal average bits/letter needed for written Afan Oromo Language for possible investigation in digital communication and storage systems. In this context, we have estimated the entropy of the language using Marko 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 as the sequence or block size approaches 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 written Afan Oromo Language in statistical linguistics modeling, natural language processing, and even can act as a reference and a framework in developing other models for the language. 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. IFPRI, EDRI, CSA. Population & Housing Census Atlas of Ethiopia 2007, International Food Policy Research Institute (IFPRI) and Central Statistical Agency (CSA), Addis Ababa. https://www.ifpri.org/publication/population-housing-census-atlas-ethiopia-20072010, Accessed 20 June 2021
2. Africanews Homepage: One to five: Ethiopia gets four new federal working languages. https://www.africanews.com/2020/03/04/one-to-five-ethiopia-gets-four-new-federal-working-languages//, Accessed 24 July 2021
3. Bedane, I.: The origin of Afaan Oromo: mother language. Glob. J. Human-Social Sci. **15**(12), 51–61 (2015)
4. Terefe, T., Hailemariam, D.: Entropy estimation and entropy-based encoding of written amharic language for efficient transmission in telecom networks. In Proceeding of IEEE AFRICON, 2017, pp. 238–244, Cape Town, South Africa (2017)
5. Shannon, C.: A mathematical theory of communication. Bell Sys. Tech. J. **27**, 398–403 (1948)
6. Stabno, M. and Wrembel, R., RLH: bitmap compression technique based on run-length and huffman encoding. In Proceeding of Information Systems, pp. 400–414 (2009)
7. Wegari, G., Meshesha, M.: Parts of speech tagging for Afaan Oromo. Int. J. Adv. Comput. Sci. Appl. **1**, 1–15 (2015)
8. Addis Ababa University Institutional Repository Homepage. http://etd.aau.edu.et/, Accessed 24 July 2021
9. Proakis, J., Salehi, M.: Communication Systems Engineering, 2nd edn. Prentice-Hall, New Jersey (2002)
10. Sayood, K.: Introduction to Data Compression. Morgan Kaufmann, Burlington (2017)
11. Shannon, C.: Prediction and entropy of printed english. Bell Syst. Tech. J. **30**(1), 50–64 (1951)
12. Markov, A.: The Extension of the Law of Large Numbers onto Quantities Depending on Each other. Translation into English (2004)
13. Abate, Y.: Compression of Amharic Text Using The PPM Context-Modeling Algorithm. MSc thesis. http://etd.aau.edu.et/, Accessed 24 July 2021
14. Sayood, K.: Lossless Compression Handbook. Academic Press Series, USA (2003)
15. Oslo Text Labratory. https://corpora.fi.muni.cz/habit/run.cgi/corp_info?corpname=orwac16/. Accessed 2020

16. Jurafsky, D., Martin, J.: Speech and Language Processing, Draft of December 30, 2020. https://web.stanford.edu/~jurafsky/slp3/3.pdf, Accessed 8 July 2021
17. Bannayeva, A., Aslanov, M.: Development of the N-gram model for Azerbaijani language. In: 2020 IEEE 14th International Conference on Application of Information and Communication Technologies (AICT), pp. 1–5 (2020). https://doi.org/10.1109/AICT50176.2020.9368645