ADDIS ABABA UNIVERSITY

COLLEGE OF NATURAL SCIENCES

Semantic Relation Extraction for Amharic Text Using Deep Learning Approach

Aschenaki Abi Abera

A Thesis Submitted to the Department of Computer Science in Partial Fulfillment for the Degree of Master of Science in Computer Science (in Data and Web Engineering)

Addis Ababa, Ethiopia

October, 2020

ADDIS ABABA UNIVERSITY

COLLEGE OF NATURAL SCIENCES

Aschenaki Abi Abera

Advisor: Yaregal Assabie (PhD)

This is to certify that the thesis prepared by Aschenaki Abi Abera, titled: *Semantic Relation Extraction for Amharic Text Using Deep Learning Approach* and submitted in partial fulfillment of the requirements for the Degree of Master of Science in Computer Science (in Data and Web Engineering) complies with the regulations of the university and meets the accepted standards with respect to originality and quality.

Signed by the Examining Committee:

Name	Signature	Date
Advisor: Yaregal Assabe (PhD)	_____	_____
Examiner: Mesfin Kifle (PhD)	_____	_____
Examiner: Ayalew Belay (PhD)	_____	_____

Abstract

Relation extraction is an important semantic processing task in the field of natural language processing. The task of relation extraction can be defined as follows. Given a sentence *S* with a pair of annotated entities *e*1 and *e*2, the task is to identify the semantic relation between *e*1 and *e*2 following a set of predefined relation types. Semantic relation extraction can support many applications such as text mining, question answering, information extraction, etc. Some state-of-the-art systems in foreign languages still rely on lexical resources such as WordNet and natural language processing tools such as dependency parser and named entity recognizers to get high-level features. Another challenge is that important information can appear at any position in the sentence. To tackle these problems, we propose Amharic semantic relation extraction system using a deep learning approach. From the existing deep learning approaches, the bidirectional long short-term memory network with attention mechanism is used. It enables multi-level automatic feature representation learning from data and captures the most important semantic information in a sentence.

The proposed model contains different components. The first is a word embedding that maps each word into a low dimension vector. It is a feature learning techniques to obtain new features across domains for relation extraction in Amharic text. The second is BLSTM that helps to get high-level features from embedding layer by exploiting information from both the past and the future direction. The single direction of relation may not reflect all information in context. The third is attention mechanism that produces a weight vector, and merges word-level features from each time step into a sentence-level feature vector, by multiplying the weight vector.

To evaluate our model, we conduct experiments on Amharic-RE-Dataset, which is prepared from Amharic text for this thesis. The commonly used evaluation techniques precision, recall, and F-score are used to measure the effectiveness of the proposed system. The proposed attention based bidirectional long short term memory model yields an F1- score of 87.06%. It performs good result with only word embedding as input features, without using lexical resources or NLP systems.

Keywords: Amharic Text Semantic Relation Extraction, Deep learning, Word Embedding, Attention based Bi directional long short term memory.

Acknowledgments

First, I would like to thank God, who makes everything possible throughout my life and this research.

I would also like to thank my advisor **Dr**. **Yaregal Assabie** for his valuable comments starting from the proposal to this end.

I would like to thank Ato Messele Gulilat from linguistic department for his willingness to give comments on linguistic aspects and dataset preparation.

My special gratitude goes to all my classmates, friends, and family for motivating and supporting me.

Table of Contents

List of Tables

List of Figures

List of Algorithms

Acronyms and Abbreviations

NLP	Natural Language Processing
e1	Entity 1
e2	Entity 2
IE	Information Extraction
RE	Relation Extraction
MUC	Message Understanding Conference
DL	Deep Learning
POS	Part Of Speech
NER	Named Entity Recognizers
ML	Machine Learning
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short Term Memory
BLSTM	Bidirectional Long Short Term Memory
Att-BLSTM	Attention-based Bi-directional LSTM networks
RNN	Recurrent Neural Network
CBOW	Continuous Bag of Words
PI	Position Indicator
r	Relations
TP	True Positives
FP	False Positives
TN	True Negatives
FN	False Negatives
P	Precision
R	Recall
F1	F1-Score
SOV	Subject Object Verb

Chapter 1: Introduction

1.1 Background

Natural language processing is the computational techniques that process written or spoken human language[1]. It focuses on developing systems that enable computers to communicate with people using human language. Currently, a large amount of unstructured data exists on the internet. This abundant data is useful only if suitable techniques are available to process the data and extract knowledge from it. This process needs a natural language processing application named relation extraction. Relation extraction (RE) is a subtask of information extraction. It transforms unstructured textual data into structured data that can be understood by machines. Semantic relation extraction aims to assign a relation label, from the pre-defined classes, between the marked nominals in the given sentence. The semantic meaning of a relationship is formed in the context of the two target nominals or relation arguments, including the word sequence between them and a window of proceeding and following words. Generally, the task of relation extraction can be defined as follows. Given a sentence *S* with a pair of annotated entities *e*1 and *e*2, the task is to identify the semantic relation between *e*1 and *e*2 by a set of predefined relation classes (e.g. Content-Container, Cause-Effect, Component-Whole, Entity-Destination, Entity-Origin, Instrument-Agency, Member Collection, Message-Topic, Product-Producer and Other).

The Amharic documents are also increasing from time to time on the web and in other machine-readable forms. Because of this growth, getting the right information for decision making from existing abundant unstructured textual data is a challenging task. The non-availability of tools for extracting and exploiting the valuable information that is effective enough to satisfy the users need have been a major problem for years. Semantic relation extraction is a challenging task especially for languages having low resources and sophisticated linguistic structures like Amharic. The complex structure of languages also needs the design of important features and the best combination of these features. Amharic is one of the low resourced and morphologically rich languages that share the above challenges. Considering the development of Amharic semantic relation extraction will greatly enhance the performance of Amharic NLP tasks like automated knowledge base construction, semantic parsing, question answering, and so on [2].

As an approach to relation extraction, many researchers used rule-based, machine learning, and deep learning (DL) for different languages. However, those approaches have their limitations. Rule-based approaches depend on handcrafted rules. A rule-based method for relation extraction lacks portability and robustness. To address these problems, machine learning approaches are proposed, which can be further classified as supervised, semi-supervised, and unsupervised. In supervised approaches, the machine is presented with huge amounts of manually annotated data. However, it takes time and effort for the development of a suitably tagged corpus. Unsupervised approaches do not need any structural information about the data. In contrast, semi-supervised approaches used both labeled and unlabeled text, which is expected to decrease the labor intensive, and the cost of developing such a system. For this thesis, a deep learning approach is proposed which can learn effective entities and relation features from the given sentences without complicated manual feature engineering. The goal of deep learning is to tackle the feature representation by creating a big meaningful network that can capture the features in the structured data[3, 4, 5, 6].

1.2 Motivation

Amharic is a working language for the federal democratic republic of Ethiopia. It is also an official language for several regions in Ethiopia such as Amhara, Addis Ababa, Southern Nations Nationalities and Peoples, Gambella, Dire Dawa, etc. It is also used as a medium of instruction for primary and junior secondary schools. Besides, several literary works, newspapers, magazines, educational resources, official credentials, and religious documents are published and available in the language. The availability of a huge amount of Amharic documents requires the development of natural language processing applications that can automatically process Amharic texts. A relation extraction system can be used as an input for different natural language applications such as semantic search, machine translation, question answering, paraphrasing, information retrieval, information extraction, etc. The necessity of relation extraction for many natural language processing applications initiated us to develop a system that automatically extracts the relation between entities in Amharic text.

On the other hand, in comparison with foreign languages, Amharic is one of the most resource-scarce languages in the context of NLP. Today the improvement in modern technology raises the availability of digital information on the Internet, which is written in the Amharic

language. Even though Amharic lacks NLP resources and we are drowning with much data. Amharic semantic relation extraction got less attention and to the best of our knowledge, no relation extraction system is developed for the Amharic language.

Deep learning has led to breakthroughs in several fields. Many of the key breakthroughs have come when researchers discovered efficient ways of learning to solve the complete problem, instead of breaking it down and trying to solve sub problems. In general, lack of active research on the automatic relation extraction and dramatic growth of electronic Amharic documents from time to time is a motivating factor for this work to come up with deep learning approaches that can minimize these problems.

1.3 Statement of The Problem

A lot of work has been done and significantly well on relation extraction for technologically advanced languages like English [7, 8, 9, 10, 11, 12], Chinese[13, 14, 15], and Indian[16, 17]. The same does not hold good for the Amharic language due to its morphologically rich nature and agglutinative structure[18]. Those researches done so far on the area of relation extraction are designed and implemented for a specific language. It is because they relied on immediate language specific lexical resources and NLP tools such as part-of-speech taggers, word embedding, dependency parsers, named entity recognizers, and others that used to extract features and constraints for relations. Besides, due to the complexity of natural languages and differences in linguistic characteristics and rules, it is difficult to design a solution that would be universal and applicable to any language. Thus, relation extraction systems developed for English, Indian, Chinese text will not be applicable and will not give the same level of performance for Amharic text. Currently, the amount of Amharic electronic information is increases from time to time. Identifying relevant information from a given text manually is time consuming and tedious task. Thus, it is better to apply the semantic relation extraction system for Amharic text.

Accordingly, as a component of the Amharic information extraction system, Bekele Worku[19] has introduced relation extraction for Amharic texts. This work has the following gaps**:** It works only for the infrastructure domain and it extracts relations between named entities within the given domain only. It uses a gazetteer for the identification of those entities, which cannot have any fixed pattern such as organization and location named entities. Thus, the extraction cannot work beyond the entities in the list or gazetteer. Manual feature

engineering is time-consuming and performing poorly on generalization due to the low coverage of different training datasets. Thus, the manually constructed features may not capture all the relevant information**.** The other downside of traditional relation extraction methods is that many traditional NLP systems are utilized to extract high-level features, such as part of speech tags(POS), shortest dependency path, and named entities, which consequently increases computational cost and additional propagation errors[20, 21, 22]. Recently, deep learning methods provide an effective way of reducing the number of handcrafted features. However, some deep learning approaches still use lexical resources such as WordNet and NLP systems like dependency parsers and NER to get high-level features [7, 23]. To solve the above limitations, we propose a semantic relation extraction system for Amharic text using a deep learning approach. Deep learning automatically learn features from the complex morphology structure of Amharic data, by reducing the number of handcrafted features[4 ,8 ,9]. This is the other pushing factor to come across with deep learning for Amharic semantic relation extraction.

1.4 Objectives

General Objective

The general objective of this research is to develop a model of semantic relation extraction for Amharic text using a deep learning approach.

Specific Objectives

The following are specific objectives, to achieve the general objective.

- Conducting a literature review for a better understanding of relation extraction.
- Collecting a corpus of Amharic text.
- Preparing a dataset for Amharic semantic relation extraction.
- Generating a word embedding that can be used as features for relation extraction.
- Analyzing the relationship between entities in Amharic text.
- Designing a model for Amharic text semantic relation extraction.
- Developing a prototype system based on the model.
- Evaluating the proposed system through the prototype.

1.5 Methods

The following methods will be applied to achieve the objectives.

Literature Review

A literature review will be conducted on the area of relation extraction related works, relation extraction approaches, feature extraction techniques, and Amharic language to have a full point of view on relation extraction. This will be reviewed from different sources such as conference proceedings, journal articles, master's thesis, and books.

Data Collection and Preparation

Amharic text corpus will be collected from Amharic bible, news agencies, broadcasting media, online newspapers, Wikipedia, and magazines. These data will be organized and structured through cleaning, tokenization, and stop word removal in a way that they are suitable for experimentation. For experimentation, both the labeled and unlabeled Amharic text will be prepared and used.

Development tools

For the development of a prototype system, Python programming language will be used as a backbone. Tensorflow machin learning library, Anaconda, Genism**,** keras machin learning library and NLTK tools will be used for prototype development, text preprocessing, and word embedding generation.

Prototype development

Based on the selected tools and programming languages, a prototype will be developed to evaluate the performance of the system. It helps to check whether the study is under the ideas and theories of relation extraction.

Testing and Evaluation

The proposed system will be tested and evaluate through the prototype using testing data (unseen data) to know how well it predicts the relation. It will be evaluated using the standard measurement metrics such as precision, recall, and F-score

1.6 Scope and Limitations

In this study, only a semantic relation extraction system is developed using a deep learning approach for Amharic text. It considers only text data types. This means it does not consider other data types such as image, video, audio, graphic, figure, Table, or any pictorial representations.

1.7 Application of Results

Relation extraction can convert unstructured information that exists in the Amharic text into structured, which allows different users to access it easily and quickly. The extracted structured information can be used as an input for the other NLP applications such as automatic question answering, information retrieval, ontology learning, semantic web labeling, knowledge base construction, mining bio text, information extraction, document summarization, machine translation, construction of thesauri and semantic networks, word sense disambiguation, language modeling, paraphrasing, and recognizing textual entailment[2, 24, 25]. The way of using RE as input for those NLP applications are described as follows:

In the automatic question answering system: Relation extraction automatically links related questions and answers. In information retrieval: The implementation of relation extraction semantic retrieval function is possible. In ontology learning: Relation extraction can discover new relationships between entities to enrich the structure of the ontology. In semantic web labeling: Relation extraction can automatically associate semantic web knowledge units. In mining biotext: Relation extraction is beneficial to determine protein-protein relations, and gene-binding conditions. These protein-protein relations are useful for applications like drug discovery in a large corpus of bio text. In knowledge base construction: Construction of knowledge bases is a laborious, time-consuming project that demands domain expertise. Automatic extraction of relationships from text helps to reduce the time and domain expertise needed for the task.

Generally, the significance of this research is summarized as follows: firstly, it will be significant for us to gain deep knowledge on relation extraction and natural language processing; secondly, it will be significant for other researchers who will develop NLP

application for Amharic text. Finally, it also helps society and organizations to extract structured information from an unstructured one. Hence, Amharic relation extraction plays an important role in Amharic language users.

1.8 Thesis Organization

The rest of the thesis is organized as follows. Chapter 2 presents a general understanding of relation extraction. This chapter introduces reviews made on different works of literature regarding relation extraction and the related subject areas together with its approaches. It discusses the Amharic language. Chapter 3 discusses related work on relation extraction using different approaches and different domains in different languages. Chapter 4 presents the Amharic text relation extraction model. This chapter also presents how the main components can be implemented. Chapter 5 presents the details of the experimentation and evaluation of the system. Chapter 6 concludes the thesis by outlining the contribution of the research. It also shows some research directions that can be used in the future to improve the RE system for Amharic text.

Chapter 2: Literature Review

2.1 Information Extraction

Nowadays, digitally stored information is available in abundance and in numerous forms. Much of this abundance of information is presented in the form of unstructured or semi-structured texts. Software tools are not able to analyze such texts and take much time for humans to manipulate, process, and store data in a structured form. Information extraction is emerged as a solution to deal with this information overload problem. Information Extraction is a sub-field of natural language processing that is concerned with identifying predefined types of information from the document and transforms them into a structured representation. It extracts relevant information from the fragments, and then pieces together the information in a coherent framework [26].

The goal of an information exraction (IE) system is to find and link the relevant information while ignoring the extraneous and irrelevant ones and making the information more suitable for information processing tasks. The extracted information can be used for several purposes, for example, to prepare a summary of texts, classifying text items, populate databases, fill-in slots in frames, identify keywords and phrases for retrieval, and so on [2, 25, 27]. Information extraction is done for Amharic text, but relation extraction is out of scope.

2.2 Information Extraction Tasks

Based on Message Understanding Conference-7(MUC-7) [28], information extraction activities are decomposed into five tasks as described below.

Named Entity Recognition: The first step in most IE systems is the detection and classification of named entities, which are proper nouns, in a natural text. Named entity types to refer to places, persons, and organizations, and so on. Some applications may require the identification of other entity types, including products, proteins, genes, weapons, and others. It is all about finding entities.

Relation Extraction: After detecting and classifying the named entity types the next step in IE is to detect and classify the relations that exist between two or more entities of a certain type (e.g. Person, Organization, Location) and fall into several semantic categories(Cause-Effect, Component-Whole, Content-Container, Entity- Destination, Entity-Origin,

Instrument-Agency, Member-Collection, Message-Topic, and Product-Producer). This study focuses on relation extraction task and a detailed description of this task will be provided later.

Temporal and Event Processing: After detecting and classifying the different named entity types and the relations that exist between them the next step for any IE system is finding and analyzing the events in which those entities take part and how those events relate to time are important for comprehensive extraction of information from a given text.

Template Filling: The task of template filling is to find documents that invoke particular scripts, which consists of prototypical sequences of sub-events, participants, roles, and properties, and followed by filling the slots in the associated templates with fillers extracted directly from the text. The fillers may consist named entities or text segments extracted from a text.

2.3 Relation Extraction

Research in the field of relation extraction dates back as far as the late 1970s [29]. Relation extraction is one of the most widely studied topics in natural language processing. It is a very useful step in IE. The task of relation extraction can be divided into two phases: the task of detecting relations if a relation occurs between the corresponding entity mention and then classifying the detected relation mentions into some predefined classes. In other words, RE is the way of transforming unstructured (free) text into a structural form, which can be used in web-search, question answering, and a lot more. How many different relations between two concepts can be found in a text is still not answered. From the definition, we can also say that the detection and extraction of semantic relations are not easy: we are dealing with information that is not explicit, so we will need to use sophisticated techniques that allow us to extract this kind of information.

A relation usually indicates a well-defined relation between two nominals. The most recent work in RE focuses on binary relations, i.e. relations that hold between two nominals, where nominals represent concepts, objects, or people in the real world and the relation predicate describes the type of stative association or interaction that holds between the things represented by the nominals. For example, the following sentence has the relation of Entity-Destination between the nominals (Flowers and chapel). ⟨e1⟩ Flowers ⟨/e1⟩ is carried into the

⟨e2⟩ chapel ⟨/e2⟩. ⟨e1⟩, ⟨/e1⟩, ⟨e2⟩, ⟨/e2⟩ are four position indicators which specify the starting and ending of the nominals [24]. The same is true for Amharic with many language structure differences. For example, "<e1>ውሃው</e1> ወደ <e2>ሐይቁ</e2> ይጓዛል።". This Amharic sentence is a type of Entity-Destination (e1, e2) relation. Because, the destination of the entity <e1>ውሃው</e1> is <e2>ሐይቁ</e2>.

In this thesis, Amharic semantic relation extraction dataset is prepared. The dataset is divided into 9 directed relations, and one an undirected relation (Other). The 9 directed relations are Product-Producer, Content-Container, Cause-effect, Instrument-Agency, Entity-Origin, Entity-Destination, Component-Whole, Member-Collection, Message-Topic, and Other class. For example, the cause-and-effect relation in the phrase “smile lines”, the product-producer relation in the phrase “honey bee”, and the content-container relation in “the apples in the basket”.

The input to the relation extraction task is usually a set of training data, and a set of testing data. The training and testing data can be supervised unsupervised or semi-supervised text data. In the supervised and semi-supervised text, entities have already been recognized. For each co-occurring pair of entities, the challenge is to determine whether one of a set of pre-defined relations holds. Relation feature extraction defines which relations are to be extracted and how they are defined, i.e. how many arguments they have and what concepts those arguments belong to.

2.4 Approaches for Relation Extraction

There are three broad lines of RE approaches: Rule-based approaches, machine learning-based approaches, and deep learning-based approaches. In the following, we go through those approaches and highlight the strengths and weaknesses of each approach.

2.4.1 Rule-Based Approaches

In rule-based approaches, humans manually gather a set of rules by lexical syntactic features that determine whether an entity pair is an instance of a relation. Since these sets of rules are often based on pattern-matching, rules are also thought of as extraction patterns. To be able to write good extractors, humans want to know what types of features are involved and how a relation is expressed in a text. Therefore, rule-based RE is considered knowledge-driven, as

opposed to the data-driven techniques. Depending on the RE system, matching rules are executed sequentially or even in a cascading fashion in which rules are embedded within other rules [30].

Typically, rule-based RE systems must strike a balance between the expressive powers they offer in creating rule-sets. If rule-sets become too complex, it becomes less humanly readable and therefore more costly to manage, debug, or extend. Because of this, rule-based RE approaches have well-known difficulties in scalability. Tailor crafted rules have shown to have high precision but on the other hand, suffer in recall as it only can discover relations that followed the exact crafted pattern. The ability to generalize to other domains is also hard because new handcrafted patterns would be needed. Thus, to alleviate these limitations machine learning approach for RE was formulated.

2.4.2 Machine Learning-Based Approaches

In machine learning (ML-based) approaches, classifiers are trained using a large set of positive and negative training examples that takes the form of text in which all entities and relation labels are marked. The main challenges in ML-based RE are twofold: Firstly, for a relation of interest a good classification algorithm and feature-set must be identified, typically through a process of experimentation with gold data. Secondly, to train and evaluate extractors large amounts of data need to be prepared. Since the costs of manually creating gold data tend to be very high and a crucial bottleneck to ML-based RE. In the following, we introduce some approaches to ML-based RE such as supervised, unsupervised, and semi-supervised, which address this problem in different ways.

A. Supervised Approaches

In the Supervised ML approach, classifiers are trained using fully hand-labeled data where entities and the relations between them were annotated. This involves labeling data in a corpus with entities and relations between them and combining various lexical, syntactic, and semantic features with machine learning classifiers [22]. It has the advantages that cross-validation can be used to automatically evaluate different feature sets and classification approaches. Even though these approaches can achieve high performance, they need expensive data annotation and the selection of features and the effective integration of

knowledge sources into relation extraction seem to be difficult. Furthermore, they are usually biased towards the domain of the text corpus. The other disadvantages are the high costs associated with manually labeling gold data if none exists. Generally, supervised learning for relation extraction is preferred over unsupervised learning due to ease of evaluation. However, it is expensive to create annotated data.

B. Unsupervised Approaches

Unsupervised approaches use massive amounts of data and extract more relations, and the subsequent associations may not be easy to map to the relations required for a particular knowledge base. Unsupervised relation extraction algorithms collect pairs of co-occurring entities as relation instances, extract features for these instances and then apply different clustering techniques to find the major relations in a corpus. These algorithms obtain a string of words between the entities in large amounts of text and clusters and simplify these word strings to produce relation-strings and rely upon tagging in advance, a predefined set of entity types, such as Person, Organization, and Location.

Unsupervised approaches can extract relations from web text without parses or named entity tags. It is very flexible compared to (distant) supervised approaches because it can be applied to a very large variety of text types and does not need a predefined set of relations. The downside is that it is not possible to predict what relations will be found, and human judgment is needed to determine whether a found relation is informative. The system is not very useful for a specific kind of relation, and it is often not possible for the system to label the relations in a way that is informative for a human. Another limitation of the unsupervised approaches is that the relations they produce might not be compatible with relations that exist in knowledge bases. This makes the automatic enhancement of knowledge bases non-trivial.

C. Semi-Supervised Approaches

The semi-supervised approach is called a weakly supervised learning method. Its basic idea is to label the unlabeled sample, which mainly solves the problem of learning. It improves the ability of model generalization in the label sample. It can effectively reduce the dependence of human participation and corpus annotation. It has been widely used for large-scale text relational extraction tasks. Another semi-supervised approach used for relation extraction is

distant supervision [21]. Distant supervision attempts to create training data automatically by leveraging large knowledge bases of facts and corpus to learning extractors. A database annotates rather than humans annotate the data, which allows a larger dataset. A small set of hand crafted data is used to collect training data. This data consists of a small set of well-known entity pairs in a certain relation. The manually collected relations and entities help as a knowledge base in a semi-supervised approach. This approach does not achieve high precision and suffers from semantic drift due to the small number of initial seeds and patterns.

2.4.3 Deep Learning-Based Approaches

The traditional non-deep learning approaches in relation extraction depend on NLP tools for feature extraction. The errors that happen on those NLP tools can be amplified in relational extraction and affect the performance of relational extraction. Such errors can be mitigated using deep learning approaches. Deep learning is also known as deep structured learning or hierarchical learning and an extension of machine learning approaches that attempts to learn a layered model of inputs commonly known as neural nets. Deep learning algorithms are capable of discovering complex structures from large datasets using the backpropagation algorithm to update their internal parameters [3, 4, 6]. This peculiarity of deep learning lets the model extract relations with better accuracy. Recently, many researchers have applied depth learning techniques to relational extraction. As shown in Figure 2.1 [16], the current trend deep learning models for relation extraction are classified into end-to-end models, dependency models, and distantly supervised models. Among them, we briefly discuss only the models used in this thesis.

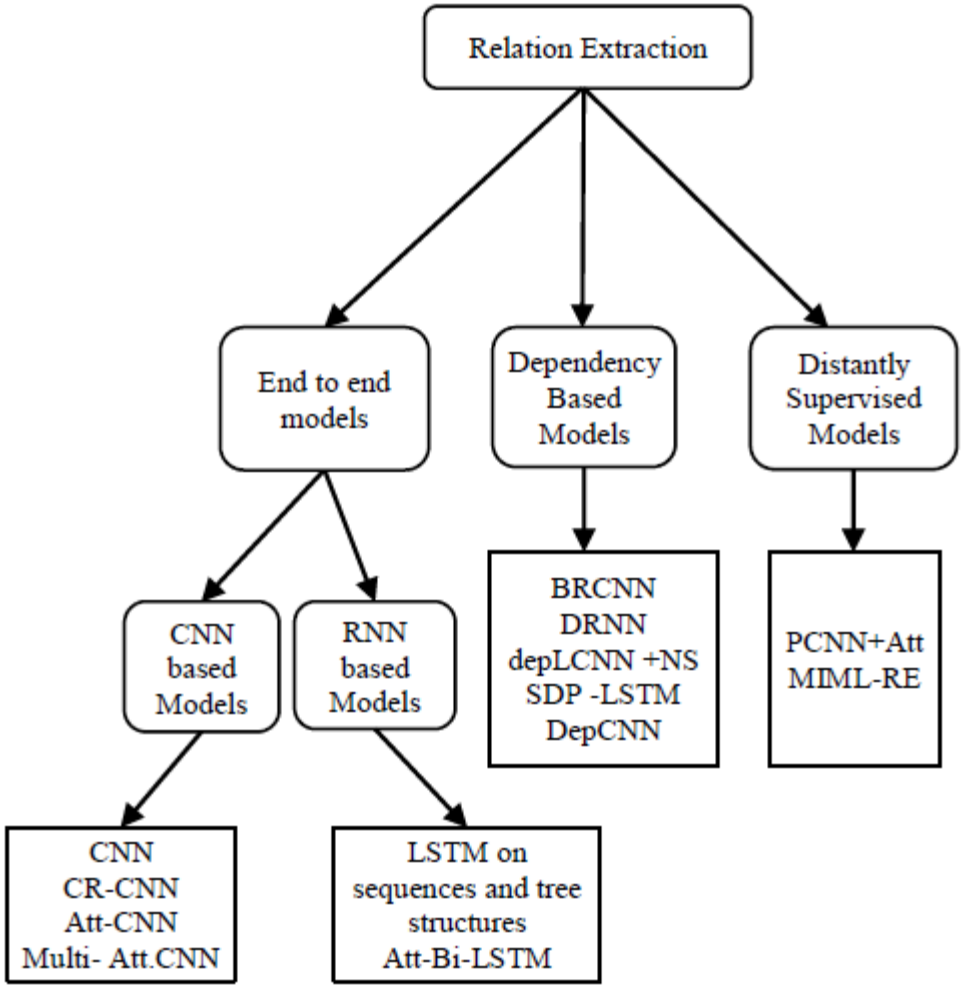

Figure 2. 1 Classification of relation extraction models using deep learning

End-To-End Models

End-To-End models are efficient ways of learning to handle the problems once, instead of breaking it down and trying to solve sub problems [31]. For instance, traditional methods for relation extraction are often based on a pipeline of two isolated subtasks: entity recognition and relation extraction. First, it detects the named entities and then performing relation extraction on the detected entity mentions. The output of entity recognition is fed as input to relation classification. The two subtasks together and ignoring the underlying interdependencies are known as end-to-end relation extraction. Recently end-to-end models without high-level features are proposed to propagate errors from the entity recognition to relation extraction. The commonly used End-to-End deep learning models for relation extraction are convolutional neural networks (CNN) based and recurrent neural network

(RNN) based. CNN is not suitable for learning long-distance semantic information [11, 23]. Thus, our approach is RNN based [32] specifically Attention-based Bi-directional LSTM model.

Recurrent Neural Network (RNN) Based Models

Traditional neural networks have a major limitation in considering the sequential relation of inputs and outputs. It is assumed that each input and output is independent of each other. To overcome this limitation recurrent neural networks (RNN) are proposed which shown great success in many NLP tasks [32]. Recurrent neural networks have a memory about what has been calculated so far and use it on current output computation. A typical recurrent neural network is shown in Figure 2.2 [3].

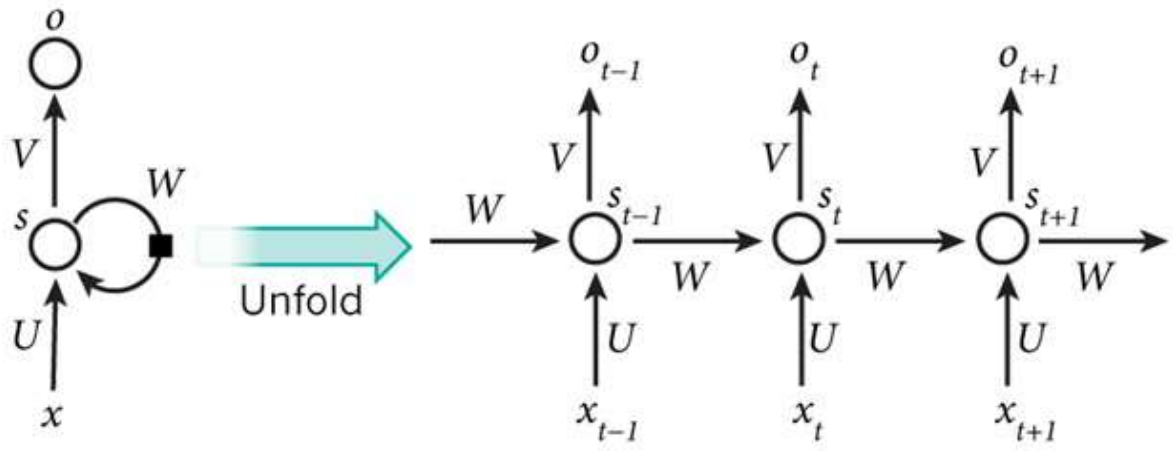

Figure 2. 2 recurrent neural network and the unfolding in time of its forward computation

The artificial neurons (for example, hidden units grouped under node *s* with values s_t at time *t*) get inputs from other neurons at previous time steps (this is represented with the black square, representing a delay of one-time step, on the left). In this way, a recurrent neural network can map an input sequence with elements x_t into an output sequence with elements $o_{t'}$ with each o_t depending on all the previous $x_t{}'$ (for $t' \leq t$).

The backpropagation algorithm can be directly applied to the computational graph of the unfolded network on the right, to compute the derivative of a total error (for example, the log-probability of generating the right sequence of outputs) concerning all the states s_t and all the parameters. Unlike a traditional deep neural network, which uses different parameters at each layer, RNN shares the same parameters (U, V, and W above) across all steps. This reacts that

it is performing the same task at each step, just with different inputs. This greatly reduces the total number of parameters the network needs to learn [3]. Although bidirectional RNN has access to both past and future context information, the range of context is limited due to the vanishing gradient problem [32]. In conventional Back Propagation Networks or Real-Time Recurrent Learning Networks, the error signals flowing backward in each time step tend to blow up or vanish. Learning to store information over extended time intervals via recurrent backpropagation takes a very long time, mostly due to insufficient, decaying backpropagation error. Since the RNN model can deal with long-distance patterns [33], LSTMs will be suitable for learning long-span relations.

Long Short Term Memory (LSTM)

LSTM has achieved the best known results in handwriting recognition [34], speech recognition [35], and relation extraction [36]. LSTM units are firstly proposed by Hochreiter and Schmidhuber[37] to overcome the gradient vanishing problem. Long Short-Term Memory (LSTM) is a special kind of RNN model designed to overcome backflow problems [37]. The LSTM model can alleviate the long-distance dependence problem of RNN and CNN, and it has been improved and promoted recently by Graves [38].

LSTM layer consists of a set of recurrently connected blocks, known as memory blocks. Each one contains one or more recurrently connected memory cells. They are explicitly designed for handling the problem of long term de-pendency where longer context information is needed for the current task. The LSTM can remove or add information to the cell state, which is carefully regulated by structures called gates. Gates control the flow of information. Usually, LSTM units are implemented in blocks with several units. These blocks have three gates: input gate, forget gate, output gate that provides continuous analogs of write, read and reset operations for the cells and that control information flow drawing on the logistic function.

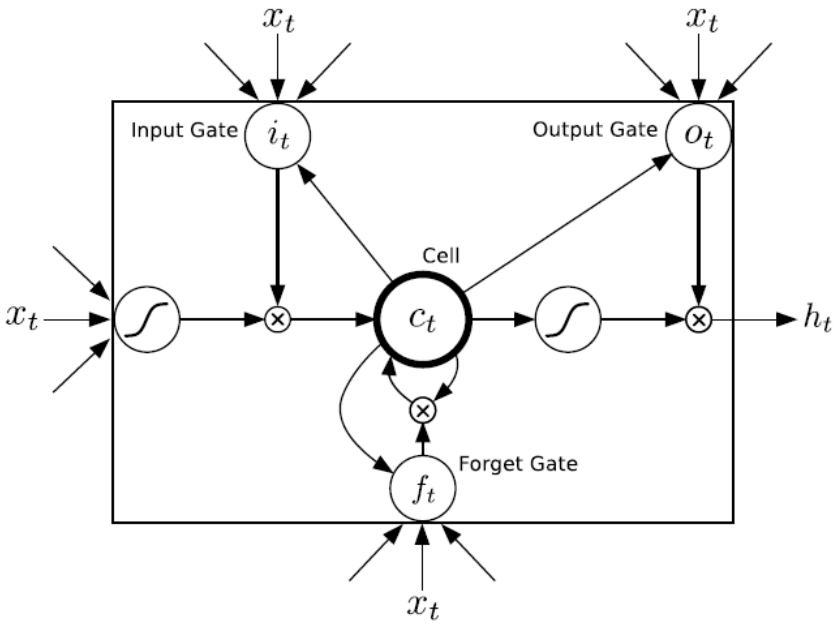

Figure 2. 3 Long Short-term Memory Cell

Figure 2.3 [38] shows one cell of the LSTM memory block. More precisely, the input x_t to the cells is multiplied by the activation of the input gate, the output to the net is multiplied by that of the output gate, and the previous cell values are multiplied by the forget gate. The net can only interact with the cells via the gates.

LSTM units keep the previous state and memorize the extracted features of the current data input. Among many LSTM variants, a variant presented by Graves et al [35] is adopted for this study, which adds weighted peephole connections from the Constant Error Carousel (CEC) to the gates of the same memory block. By directly using the current cell state to generate the gate degrees, the peephole connections allow all gates to inspect into the current cell even when the output gate is closed [38].

The following components composite the LSTM-based recurrent neural networks. The input gate i_t with a corresponding weight matrix $W_{xi}, W_{hi}, W_{ci}, b_i$. The forget gate f_t with a corresponding weight matrixW_{xf},W_{hf},W_{cf},b_f. The output gate o_t with a corresponding weight matrix $W_{xo}, W_{ho}, W_{co}, b_o$. All of those gates are set to generate some degrees, using the current input x_i, the state h_{i-1} that previous step generated, and the current state of this cell c_{i-1} (peephole), for the decisions whether to take the inputs, forget the memory stored before, and output the state generated later. Just as these following equations demonstrate:

$$i_t = \sigma(w_{xi}x_t + w_{hi}h_{t-1} + w_{ci}c_{t-1} + b_i) \quad (2.1)$$

$$f_t = \sigma(w_{xf}x_t + w_{hf}h_{t-1} + w_{cf}c_{t-1} + b_f) \quad (2.2)$$

$$g_t = tanh(w_{xc}x_t + w_{hc}h_{t-1} + w_{cc}c_{t-1} + b_c) \quad (2.3)$$

$$c_t = i_t g_t + f_t c_{t-1} \quad (2.4)$$

$$o_t = \sigma(w_{xo}x_t + w_{ho}x_{t-1} + w_{co}c_t + b_o) \quad (2.5)$$

$$h_t = o_t \, tanh(c_t) \quad (2.6)$$

Where σ is the logistic sigmoid function and h is the hidden vector. Hence, the current cell state c_t will be generated by calculating the weighted sum using both previous cell state and current information generated by the cell.

Bidirectional Long Short-Term Memory (BLSTM)

Bidirectional long short-term memory recurrent neural networks are extended versions of unidirectional LSTM networks. An inevitable challenge in relation extraction is that the important information can appear at any position in the sentence. However, standard LSTM networks process sequences in temporal order, they ignore future context. BLSTM network is designed to capture information of sequential data set and maintain contextual features from the past and future. Bi-directional LSTM networks are used to model the sentences with complete sequential information about all words before and after it [7, 8].

For the semantic relation extraction task, it is beneficial to have access to the future as well as past context. In this paper, we use BLSTM to get high-level features by accessing the future as well as past context.

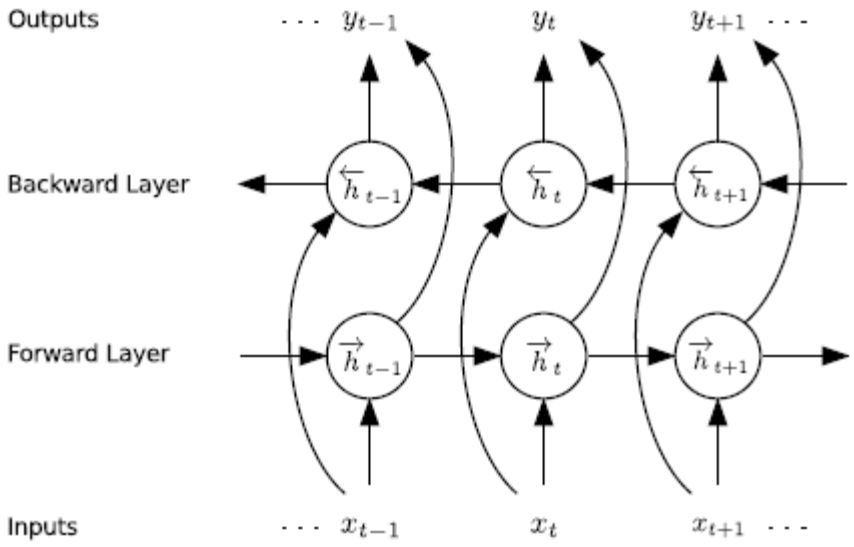

Figure 2. 4 Architecture of a Bidirectional Long Short Term Memory RNN

As shown in Figure 2.4 [35], the network includes two sub-networks, which are forward and backward pass for the left and right sequence context respectively. The output of the t^{th} word is shown in the following equation:

$$h_t = [\overrightarrow{h_t} \oplus \overleftarrow{h_t}] \quad (2.7)$$

Here, we use the element-wise sum ($\oplus$) to combine the forward and backward pass outputs.

There are three motives for selecting BLSTM for relation extraction. First, BLSTM shows better performance. Second, LSTM-based models have meanings that are more explicit in the attention mechanism that is used in the next step than CNN-based models. Last, BLSTM is quite simple compared with other complex models, which means it has fewer parameters and faster calculating speed.

Attention Mechanism

The attention mechanism is widely used in NLP tasks such as question answering, machine translation, relation extraction, and others for different languages [6, 13]. Attention mechanism was introduced to address the limitation of modeling long dependencies and the efficient usage of memory for computation. The attention mechanism intervenes as an intermediate layer between the encoder and the decoder, having the objective of capturing the information from the sequence of tokens that are relevant to the contents of the sentence. This

mechanism helps the decoder to pick only the encoded inputs that are important for each step of the decoding process. It aims to select the most relevant part concerning the given text.

Since raw sentences contain some irrelevant information. For concentrating on the words that are important to predict the relationship of entities, it can be a good strategy to pay more attention to these words. The attention mechanism enables the model to differently attend over the hidden vectors of BLSTM along a context subsequence and produces a weighted representation. The ultimate goal of the Attention Model is to help frameworks like Encoder-Decoder learn the interrelationships between multiple contents to better represent this information.

Softmax Classifier

Softmax is used for the multi-classification task.it assumes that each example is a member of exactly one class. A fully connected softmax layer accepts the sentence representation obtained from the attention layer to predict relations from the pre-defined relation set. In this thesis, we use a softmax classifier to predict a label $\hat{y}$ from a discrete set of classes Y for a sentence S. It produces the probability distribution p over relation types conditioned on the sentence representation s.

The classifier takes the hidden state h^* as input:

$$\hat{p}\,(y|S) = softmax(W^{(s)}h^* + b^{(s)}) \qquad (2.8)$$

$$\hat{y} = \underset{y}{arg\,max}\ \hat{p}\,(y|S) \qquad (2.9)$$

The cost function is the negative log-likelihood of the true class labels$\hat{y}$:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m} t_i log\,(y_i) + \lambda\, /\!/\theta\,/\!/^2\, F \qquad (2.10)$$

Where t $\in$ $\Re^m$ is the one-hot represented ground truth and y $\in$ $\Re^m$ is the estimated probability for each class by softmax (m is the number of target classes), and λ is an L2 regularization hyper parameter. The feature vector is fed into softmax classifier to get the probability distribution of relation types [7].

2.5 Feature Extraction

Feature extraction is the name for methods that select or combine variables into features, effectively reducing the amount of data that must be processed, while still accurately and

completely describing the original data set. A key difference between deep learning and traditional machine learning is how features are extracted. Traditional machine learning approaches use handcrafted engineering features by applying several feature extraction algorithms and then apply the learning algorithms. Additionally, other boosting approaches are often used where several learning algorithms are applied to the features of a single task or dataset and a decision is made according to the multiple outcomes from the different algorithms.

On the other hand, in the case of deep learning, the features are learned automatically and are represented hierarchically at multiple levels. This is the strong point of deep learning against traditional machine learning approaches. The most common features for the task of relation extraction using deep learning models are described as follows.

2.6 Word Embeddings

Word embedding is the technique of mapping words to vectors of real numbers. It is a recent approach to feature learning techniques in relation extraction. It helps to generate the semantic and syntactic similarities of words by learning from huge data. The basic idea of word embeddings is that any two words that have similar meaning will also have similar context words. Word embeddings can be considered as unsupervised feature extraction. Thus, it helps to reduce the need for linguistic resources and hand-coding feature extractors for feature extraction [39]. The most common example to demonstrate the semantic embedding capabilities of word embedding is vector (" King") – vector (" Man") + vector (" Woman") ≈ vector ("Queen").

word2vec [39] and glove [40] are two methods for learning word embeddings from raw text that have shown their advantages in many NLP tasks like relation extraction. The glove is used to capture the meaning of one word embedding with the structure of the whole observed corpus using the word frequency and co-occurrence counts as the main measures. Word2vec is a two-layer neural net that processes text by "vectorizing" words to establish similarities between terms. There are two main models for word2vec, the continuous bag-of-words (CBOW) model and the skip-gram model. Let us discuss both these models separately.

1) The Skip-Gram Model

The skip-gram model is usually used to predict all surrounding words (context) given a word [39]. With skip-gram, the representation dimension decreases from the vocabulary size to the length of the hidden layer. Additionally, the vectors are more meaningful to describe the relationship between words. The skip-gram model considers that context words are generated based on the central target word. For example, the text sequence is “the”, “man”, “loves”, “his”, and “son”. The central target word is “loves”.

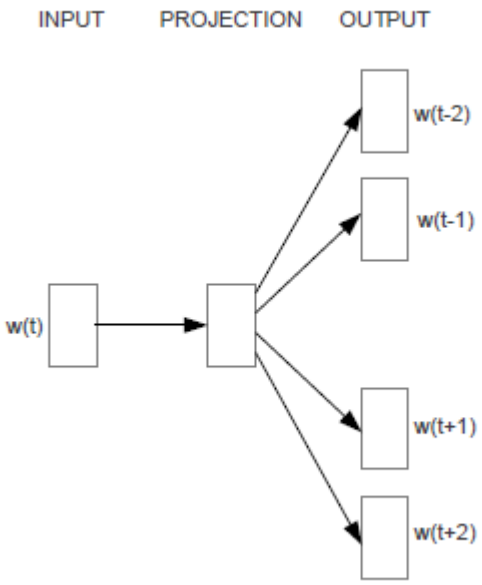

Figure 2. 5 The skip-gram model

As shown in Figure 2.5 [39], the model accepts a word W(t) and predicts the words around a given word W(t), which context words W(t-2), W(t-1), W(t+1), W(t+2). Some words can be skipped within a given window size to look forward and backward from the target word. The Skip-gram model has one projection layered neural network. The input layer consists of a one-hot encoded vector of the vocabulary.

2) The Continuous Bag of Words (CBOW) Model

CBOW creates a sliding window around the current word, to predict it from context (the surrounding words) [41]. The context can be a word or a group of words. In the CBOW

model, the central target word is generated based on the context words. based on the context words “the”, “man”, “his” and “son”, the CBOW model can generate the target word “loves”.

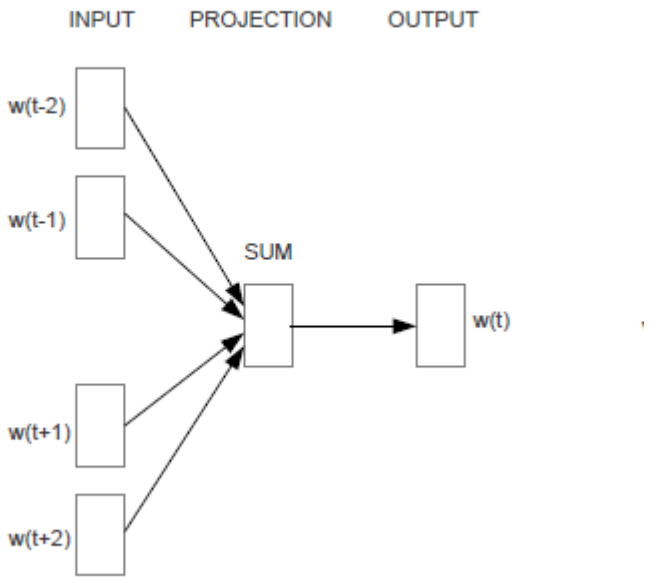

Figure 2. 6 The Continuous Bag of Words (CBOW) Model

A continuous bag of words is the reverse of the skip-gram model. As shown in Figure 2.6 [41], the model accepts the context W(t-2), W(t-1), W(t+1), W(t+2) the task is to predict the target word W(t). The continuous bag of words model (CBOW) takes the average of the vectors of the input context words to compute the output of the production layer. As shown in Figure 2.6 and Figure 2.7, the difference between skip-gram and CBOW is the way the word vectors are generated. In CBOW, the target word with all the examples are fed into the model and taking the average of the extracted hidden layer.

Given a sentence consisting of T words S = { $x_1, x_2, \ldots, x_T$ }, every word x_i is converted into a real-valued vector e_i. For each word in S, we first lookup the embedding matrix $W^{wrd} \in R^{d^w |V|}$, where V is a fixed-sized vocabulary and d^w is the size of word embedding. The matrix W^{wrd} is a parameter to be learned and d^w is a hyper-parameter to be chosen by the user. We transform a word x_i into its word embedding e_i by using the matrix-vector product:

$$e_i = W^{wrd} v^i \quad (2.11)$$

Where v^i is a vector of size |V | which has value 1 at index e_i and 0 in all other positions. The output of this phase is a fixed-sized vector representation for each word. Then the output sentence is feed into the next layer as a real-valued vector. This output file will be the source of features in the next stages of this proposed architecture.

$$\text{emb}_s = \{e_1, e_2, \ldots, e_T\} \quad (2.12)$$

2.7 Position Indicators

In relation extraction, position indicators are needed to let the algorithm know the target nominals. Position indicators (PI) also highly important to increase relation classification accuracy. For instance, the `input word sequence S is represented below, including four position indicators (PI) to specify the starting and ending of the nominals. The following is an example:

"<e1> people </e1> have been moving back into <e2> downtown </e2>".

Note that people and downtown are the two nominals with the relation 'Entity-Destination (e1, e2)', and <e1>, </e1>, <e2>, </e2> are the four position indicators which are regarded as single words in the training and testing process.

The four PIs are interpreted as:

- '<e1>': a word before the first relation argument in the word sequence
- '</e2>': a word after the first relation argument in the word sequence
- '<e2>': a word before the second relation argument in the word sequence
- '</e2>': a word after the second relation argument in the word sequence

2.8 Evaluation Metrics

The effectiveness of the proposed system is demonstrated by evaluating the model on the newly developed Amharic-RE-Dataset for the relation extraction task. The evaluation metrics to evaluate relation extraction systems are precision, recall and the f1- Score. Precision is the fraction of correctly retrieved relationships of type r in the text, over the total number of retrieved relationships of type r. recall is the fraction of relationships of type r that are correctly extracted, over the total number of relationships of type r present in the text. An attempt to

combine these measures is the F1-Score, which corresponds to the harmonic mean of both precision and recall.

Where r refers to a type of relationship, and N refers to the number of relationship types r.

True Positives (TPr) as the number of successfully extracted relationships of type r;
False Positives (FPr) as the number of extracted relationships that are said to be of type r but are not from type r.
True Negatives (TNr) as the number of successfully extracted relationships that were not of type r.
False Negatives (FNr) as the number of extracted relationships that are of type r but are said to be other type.
Accordingly, Precision, Recall, and F1-Score can be defined as:

$$Precision\ (Pr) = \frac{Number\_of\_correctly\_extracted\_relations\ r}{Total\_Number\_of\_extracted\_relations\ r} = \frac{TPr}{TPr + FPr} \qquad (2.\ 13)$$

$$Recall\ (Rr) = \frac{Number\_of\_correctly\_extracted\_relations\ r}{Total\_Number\_of\_Relations\_types\ r\ in\_Text} = \frac{TPr}{TPr + FNr} \qquad (2.\ 14)$$

$$\text{F1-Score (F1r)} = 2 \times \frac{Pr \times Rr}{Pr + Rr} \qquad (2.\ 15\ )$$

These measures are naturally defined per class, but it can be useful to extend them in order to compose a final score in the case of multiclass classification problems. A direct extension is obtained by averaging over Precision and Recall, which is what Macro-average and Micro-average methods try to implement. Macro-averages given by Equations (2.16) and (2.17) takes the average of the precision and recall of the system on different sets. Micro-averages given by Equations (2.18) and (2.19) sum up the individual true positives, false positives, and false negatives returned by the system for different classes, and the measures are computed by applying the formula of the single class measures (i.e. Precision or Recall) directly using the sums of the individuals. The last performance metric that is used for evaluation is the macro-averaged F1 score prediction. The F1 score is defined as the harmonic mean between the precision and recall of the system. Macro-averaged means that the evaluation for precision and recall is done on a per-class basis then averaged [42, 43].

$$Macro\text{-}average\ Precision = \frac{\sum_r^N Pr}{N} \quad (2.16)$$

$$Macro\text{-}average\ Recall = \frac{\sum_r^N Rr}{N} \quad (2.17)$$

$$Micro\text{-}average\ Precision = \frac{\sum_r^N TPr}{\sum_r^N TPr + \sum_r^N FPr} \quad (2.18)$$

$$Micro\text{-}average\ Recall = \frac{\sum_r^N TPr}{\sum_r^N TPr + \sum_r^N FNr} \quad (2.19)$$

Confusion matrix: confusion matrix is one of the most intuitive and descriptive evaluaion metrics used to find the accuracy and correctness of machine learning classification models where output can be two or more types of classes. One benefit of a confusion matrix is that it is easy to see if the system is confusing two classes (i.e. commonly mislabeling one as another). Confusion matrix is represented through rows and columns, where rows are the actual classes of the instances, and columns are the predicted classes of the instances. For the binary classification, a confusion matrix is represented as $2 * 2$ matrix. For the multiclass classification, a confusion matrix with the k class has a $k * k$ confusion matrix. Four confusion matrix measures namely TP, FP, TN, and FN are used for the classification of the model as shown in the below table 2.1 [44].

Table 2. 1 Confusion matrix

		Predicted class	
		Positive	Negative
Actual class	Positive	TP	FN
	Negative	FP	TN

2.9 Amharic Language

For this research since Amharic texts are considered, it is important to investigate the characteristics of the language. Hence, under this section, the structure of the Amharic language those are believed to be pertinent to the current research will be reviewed.

2.10 Overview of Amharic language

Amharic is an official working language of the Federal Democratic Republic of Ethiopia and the second most widely spoken Semitic language in the world, next to Arabic [45]. It is also a working language of several regional states like Amhara, Addis Ababa, South Nation Nationalités and people, Dire Dawa, *etc.* It is also a medium of instruction in primary and secondary schools in the country. It is spoken by around a million people as the first language and 7-15 million people as a second language [46]. Currently, different Mass media like radio, television broadcasts, and the press are also using it for disseminating information to the public. Despite it has a large speaker population, the language has little computational linguistic resources. Like other Semitic languages, Amharic is one of the most morphologically complex languages [18].

2.10.1 Amharic Writing System

Amharic is written using a writing system called Fidel, adapted from the one used by Ge'ez language [45]. Amharic took the whole Geez alphabet including redundant characters and use it in the writing system. For instance, ሐ, ሀ, ኀ and ኸ are pronounced as (hä), ሰ and ሠ as (sä), አ and ዐ as (a) and, ጸ and ፀ as (tsä) [47]. The Amharic alphabet does not have capital and lower case distinctions. Amharic differs from the structure of Semitic languages, especially in syntax. It is conveniently written in a tabular format of seven columns. The basic form is represented in the first column and the rest are derived from it by simple modifications. Amharic has 34 base characters and a total of 435 characters. In Amharic language, one symbol or character represents both a consonant and a vowel. The six orders are formed by attaching diacritic markings to the basic symbol to combine consonants with vowels. The Ethiopic script is a syllabary rather than an alphabet because it doesn't have separate characters for vowels unlike alphabets in other languages but for the sake of convenience, it is named as an alphabet [48].

2.10.2 Amharic Punctuation Marks and Numerals

The Amharic language has its punctuation. There are varieties of punctuation marks used in handwriting. The most known punctuations are:

- Colon (**:**), referred to as "hulet netib" in Amharic, is used to separate two words. Nowadays the two dots are replaced with whitespace.
- Four dot (**::**): is used as a sentence delimiter which is equivalent to the symbol "." in English.
- Netela serez (፣): is used to separate lists or ideas just like the comma in English.
- Dirib serez (፤): has the same purpose as semi-colon has in the English language.

In addition to those listed above, the language uses other punctuation marks taken from other languages like ? ,! , /, \, "", "", ' etc.

In the Amharic writing system, two types of numbering systems are used. The first one is taken from the Geez language. However, these numbers are not suitable for mathematical computation since there is no symbol for the number zero in this numbering system. It has mostly been used for page numbering and dates. The second numbering system is the one used in the English language. This one is more suitable to use for automatic Amharic document processing applications.

2.10.3 Amharic Sentence Structure

Amharic is one of the most morphologically complex languages [18]. In this language, different affixes are used to make inflectional and derivational morpheme. The derivation is achieved by affixation (prefix, infix, and suffix) or compounding. Inflection is achieved by either changing vowels or repeating consonants; and then adding the necessary affixes or suffixes. The word order in Amharic clauses is generally subject object verb (SOV) arrangement [49]. The grammatical structure of Amharic sentence is a combination of a noun phrase and verb phrase. The noun phrase comes first and then the verb phrase.

In Amharic language, one word can be a sentence by implicitly combining the object, subject, and verb. For example, the word "አፈቀራት" can be taken as a sentence; the subject is "እሱ", the object is "እሷ" and the verb is "አፈቀረ". Therefore, it is difficult to identifying morphemes from a word. Besides, plural nouns, plural verb formation in Amharic language is different from

English language. According to Baye Yimam [47], Amharic words are categorized under five categories. This classification is based on the use of morphology and position of the word in a sentence. Amharic word categories are noun, verb, adjective, adverb, and preposition.

In this thesis, the relation extraction model is also based on these Amharic sentence structures. In Amharic sentences, the types of relations are discussed in the following examples.

Example1: <e1>ቁልፉ</e1> በሰውየው <e2>ኪስ</e2> ውስጥ ነው ።።

This sentence is a type of Content-Container (e1, e2) relation. Where e1 refers to ቁልፉ (key) and e2 refers to ኪስ (pocket). In this case, ቁልፉ (the key) is content, and ኪስ (pocket) is a container. Thus, a container must enclose the thing it contains.

Example2 : <e1>መሳሪያዎቹ</e1> <e2>በፋብሪካ</e2> ተመረቱ ።።

This sentence is a type of Product-Producer (e1, e2) relation. In this case, መሳሪያዎቹ (instruments) is a product, and በፋብሪካ (factory) is a producer. The producer ፋብሪካ should be actively involved in the process of bringing the product መሳሪያዎቹ into existence and not just serve as a raw material.

Chapter 3: Related Work

In this chapter, we present some works done so far on relation extraction. Among them, we have chosen the most relevant ones, which are related to our work those are done in different languages.

3.1 Relation Extraction from English Language

The main challenge in RE is the fact that important information can appear at any position in the sentence. Therefore, Zhang *et al.* [7] proposed bidirectional long short-term memory networks (BLSTM) to model the sentence with complete, sequential information about all words. BLSTM network helps to model the sentence level representation. For every word in a given sentence, BLSTM has complete, sequential information about all words before and after it. Long distance relationships may be solved to some extent in these networks. At the same time, they also use features derived from the lexical resources such as WordNet or NLP systems such as dependency parser and named entity recognizers (NER). They aim to grasp more features that may indicate the relationship between the pair of two nominals. They extract and concatenate sentence level features and lexical level features to form the finally extracted feature vector. A multilayer perceptron (MLP) is used for combining sentence level features and lexical features into the final extracted feature vector. Finally, the final extracted features are fed into a softmax classifier to predict the semantic relation labels.

Experiments are conducted on the SemEval-2010 task 8 dataset [24]. It proves that the BLSTM-based method is effective to mine the relationship between two nominals. With more features, the performance achieves F1 of 84.3, which testifies general features gotten from NLP tools could improve the classification performance. As shown in the experiment, using only word embedding as input features are sufficient to achieve state-of-the-art results. They prove the effect of different features for the classification by ignoring a feature from the set of features. For example, by removing position and NER features. Maybe other features represent the information they contained. It shows that BLSTM has a better representation of sentence level relationships without position features. The whole features are considered from lexical and sentence levels. As a limitation of this work, it requires more NLP resources to train the model than attention-based models.

Wu *et al.* [8] introduces semantic relation classification using a Bi-directional LSTM network. This network helps to solve the over-fitting problem. Sequence padding is used to fill in the missing words if the sentence length is less than the standard length. The bi-directional LSTM network is used to enhance the accuracy, and another novel point is input layer processing, including max-length processing and sequence padding. At the same time, they use deleting "nonsense" word method, article, preposition, conjunction, interjection. Word2vec is used in every sentence.

Maximum-likelihood loss function or cross-entropy cost function is chosen to convergence and then the parameters go backpropagation to adjust parameters. There are some sentence level features in this model, and then they also need to collect position features. Both entity features and label features are used. The SemEval-2010 Task 8 public dataset is used to evaluate this system [24]. Finally, they have 83.1 accuracies of the result and value 80.3 F1-score on their test set. The limitation of this work is that the performance of the encoder-decoder network degrades rapidly as the length of the input sentence increases.

Traditional relation extraction methods are usually based on pattern matching and designing features manually is time-consuming, and performing poorly on generalization due to the low coverage of different training datasets [20, 21, 22]. It depends on lexical resources or NLP systems to get high-level features. Another downside is that important information can appear at any position in the sentence. To solve these problems, *Zhou et al.* [9] proposed attention-based bidirectional long short-term memory networks (Att-BLSTM) for relation classification. This model employs an attention mechanism with BLSTM to capture the most important semantic information in a sentence. The model contains two sub-networks, which are forward and backward pass, for the left and right sequence context respectively. They used the element wise sum to combine the forward and backward pass outputs. They used the attention mechanism for relation classification tasks. They employed dropout on the embedding layer, LSTM layer, and the penultimate layer to prevent co-adaptation of hidden units by randomly omitting feature detectors from the network during forward propagation. They used word embedding for feature extraction.

This model does not utilize any features derived from lexical resources or NLP tools; it uses raw text with position indicators as input. It can automatically focus on the words that have a

decisive effect on classification. To chuck, the effectiveness of Att- BLSTM, experiments are conducted on SemEval-2010 Task 8 dataset [24], which is based on a macro-averaged F1-score for the nine actual relations (excluding the Other relation) and considers the directionality. It achieves an F1-score of 84.0% with only word vectors.

Lee *et al.* [10] present an end-to-end recurrent neural model that combines an entity-aware attention mechanism with a latent entity typing (LET) method. This model is not only effectively utilizes entities and their latent types as features, but also builds word representations by applying self-attention based on the symmetrical similarity of a sentence itself. The self-attention mechanisms and the recurrent neural architecture with BLSTM are applied to capture the context of the sentence. Entity-aware attention focuses on the most important semantic information by considering entity pairs along their latent type representation obtained by LET which constructs a latent type vector of an entity according to its characteristic.

The model contains four main modules: Word representation, to link each word into the corresponding vector representations. Self-Attention, to capture the meaning of the correlation among words using multi-head attention. BLSTM, to encode sequentially the representations of the self-attention layer. Entity-aware Attention, to calculate attention weights regarding the word positions, entity pairs, and their type representations. The obtained features are then averaged with time steps to produce sentence-level features. The proposed model is evaluated on the SemEval-2010 Task 8 dataset[24], this proposed model with LET achieves an F1-score of 85.2%. It is using only raw sentence and word embeddings without any high-level features from NLP tools, and it outperforms existing state-of-the-art methods.

T. H. Nguyen and R. Grishman [11] introduced a convolutional neural network for relation extraction that automatically learns features from sentences. It emphasizes an unbalanced corpus and minimizes the usage of external supervised NLP toolkits and resources for features. Because errors in these external NLP toolkits lead to errors of relation detection and classification. Rather than using exterior features to enrich the representation, this model, utilize word embeddings that can be trained automatically in an unsupervised framework as the only external resource for the whole system. Besides, they used pre-trained word embeddings for initialization and optimization of both word embeddings and position

embeddings as model parameters. This network takes advantage of multiple window sizes for filters, position embeddings for encoding relative distances, and pre-trained word embeddings for initialization in a non-static architecture.

The elements that they derive from this structure are the words, the n-grams, and their positions in the sentences, suggesting a paradigm in which relation mentions are represented by features that depend on these elements. All the parameters for the presented CNN are the word embedding matrix, the position embedding matrix, the m filter matrices, the weight matrix for the fully connected layer. The gradients are computed using back-propagation while training is done via stochastic gradient descent with shuffled mini-batches.

They evaluate the models on the SemEval-2010 Task 8 dataset[24] for relation classification. The experimental results show that this model outperforms the state-of-the-art systems for relation classification by achieving 82.8%. They emphasize that the relation extraction problem with an unbalanced corpus. The ACE dataset is much more biased toward the Other class than the SemEval dataset and thus more appropriate for relation extraction experiments. However, CNN is not suitable for learning long-distance semantic information.

Xiaoyu *et al.* [12] proposed a novel Att-RCNN model using a combination of recurrent neural network (RNN) and convolutional neural network (CNN). This is used to extract higher level contextual representations of words and to obtain sentence features respectively. Att-RCNN utilizes both advantages of RNN and CNN to capture features by embedding the relation information in texts. They used bidirectional RNN with gated recurrent units (GRU) cells to learn contextual features of each word by using word embeddings as cell input. They employ both a word level attention dealing shortest dependency path (SDP) information and a sentence level attention applying to max-pooling procedure.

SemEval-2010 task 8 dataset[24], KBP37 dataset and SemEval-2018 task 7 dataset are used for experimentation. The experiments show that the proposed model (Att-RCNN) achieves nearly 4% higher than CNNC+ softmax, a CNN-based model, and nearly 3% higher than SDP-LSTM, a RNN-based model on *F*1 score. They improve the *F*1 score by 2.5% as Att-RCNN takes advantage of both CNN and RNN architecture. Att-RCNN also outperforms all the single models and achieves an *F*1 score of 86.42%, which reaches an increase of 1.5% than others.

3.2 Relation Extraction from Chinese Language

Ji Wen [13] presents, structure regularized bidirectional recurrent convolutional neural network(SR- BRCNN) model, to classify the relation of two entities in a sentence. BCRNN model is used to learn representations with bidirectional information along the shortest dependency path (SDP) forwards and backward at the same time. The SDP, RNN with LSTM units are employed to learn hidden representations of words and dependency relations respectively. The local features are captured from hidden representations of every two neighbor words using the convolution layer.

The information is gathered from local features of the SDP and the inverse SDP using the max-pooling layer. It has a softmax output layer after the pooling layer for classification in the unidirectional model RCNN. They used pre-trained word embeddings for feature extraction. The shortest dependency paths are used between two entities to capture the predicate argument sequences. Each two neighbor words are linked by a dependency relation in the shortest dependency path. When they inverse the shortest dependency path the corresponding relation keeps the same. Due to the limitations of recurrent neural networks to capture long term dependencies, they employ LSTM in this work. LSTM achieves well in tasks where long dependencies are required. In LSTM cell, some gating units are designed. Each of them is in charge of specific functions. They used two bidirectional LSTMs to capture the features of words and relations separately.

To get better SDPs, they proposed the Structure Regularized BRCNN. The most intuitive way to cut the dependency trees is to cut by punctuation. However, every sentence has no punctuation and the original dependency trees cannot be appropriately regularized. They tried decomposing the dependency trees using prepositions to overcome the drawback of random regularization.

The model is evaluated on two datasets. One is the Chinese Sanwen dataset and the other one is the SemEval2010 Task 8 dataset. They develop a corpus on Chinese Sanwen focusing on the task of Named Entity Recognition and Relation Classification. Experimental results show that this method outperforms the state-of-the-art approaches on the Chinese Sanwen task and performs as well on the SemEval-2010 Task 8 dataset [24]. The model archives 65.9% for Chinese Sanwen and 85.1% for SemEval-2010 Task 8 dataset.

Han *et al.* [14] proposed a neural network model named BLSTM-CCAtt (bidirectional-LSTM model using Character Composition Attention) model. This model is an attention-based neural network using the information provided by Chinese character compositions of entities. It uses the character composition of the entities to provide additional information. This model is constructed by starting with encoders and ending with a softmax classifier. In this model, there are three encoders, which are sentence encoders, and entity encoders, and BLSTM encoders. Finally, the BLSTM is chosen as all these three encoders. When the sentence is encoded, the attention mechanism employed the outputs of entity encoders as the query to provide weight to the words or characters and get the vector expression of this sentence. After a full connection layer, a softmax classifier is used to classify the relationship between the entities. In this paper, the authors create a large scale Chinese relation dataset based on a Chinese encyclopedia, to solve the problem of lacking large-scale open-domain datasets.

In this model, the first step is to transform the input tokens into low-dimensional vectors. When encoding sentences, the "input tokens" refers to words or characters according to whether the encoder is word-level or character-level. These input tokens are transformed into vectors by looking up the pre-trained embeddings. The position feature is used to specify the given entity pair. It also needs to be transformed into vectors by looking up the position embeddings. Extra information that cannot obtain directly from the sentence is verified to be helpful in relation extraction. Many Chinese characters have unique meanings. The characters that constitute the entities can provide additional information. Using this information, the attention mechanism can identify the critical part of the sentence. The proposed model achieves the best performance among all tested models. The F1 score of the proposed model is improved from 87.30 to 87.89 when using the multi-instance learning method.

Ouyang et al. [15] proposed an attention-based multi-instance multi-label bidirectional long short-term memory network (ATT+MIML+BLSTM) for distantly supervised Chinese text relation extraction. This paper focuses on distant supervision for relation extraction in Chinese texts. This model can obtain the rich semantic information of the Chinese sentences and handle overlapping relations. The BLSTM and character-level attention modules are used to obtain the important semantic information in each sentence, and then the final sentence vector representation is obtained through sentence-level attention. In addition, the multilabel loss function is used in the network model to deal with overlapping relations. Sentence-level

features are designed to construct a distributed representation for each sentence. The Chinese characters first convert in the sentence into real-valued vectors. The higher features of the sentence are then extracted by BLSTM. The final character-level attention weights the output of each time step of the BLSTM. The character-level features of each time step are then merged into a sentence level feature vector.

The input for each sentence consists of character embeddings and Position embeddings without Chinese word segmentation errors. character embedding is used to avoid introducing word segmentation errors into the relation extraction process. The model takes as input each raw Chinese character in a Chinese sentence. Position embeddings are used to represent the structural information of sentences. Then, the attention mechanism is used to extract richer Chinese character and sentence features. Character-level attention is used to capture important information in sentences and improve the accuracy of sentence representation. They handle the multi-label nature of relation extraction by using multi-label loss functions in the neural network classifier.

Distant supervision produces a lot of noise or mislabeled data, and direct use of supervised methods to classify relationships is very ineffective. To solve the problem of mislabeling, the author proposes the construction of sentence level relational attention on multiple instances by using the selective attention mechanism to weigh sentence vectors and weaken the weight of noisy instances dynamically. Chinese character embeddings are trained on the Chinese Wikipedia corpus by the word2vec tool.

Based on the idea of distant supervision, a new dataset is constructed for relation extraction in Chinese texts. The constructed dataset contains eight kinds of relationships (cooperation, friends, couples, parenthood, lovers, faculty-student, brother and sister, others). Experiments on this dataset show that the proposed method has achieved 95.7% accuracy for Chinese relation extraction relatively high performance. It can be seen that the attention mechanism can improve the performance of the model.

3.3 Relation Extraction from Indian languages

Ajees and Sumam [16] proposed a relation extraction system using a convolutional neural network. Five Indian languages are considered for this work. They are Hindi, Tamil, Kannada, Malayalam, and Telugu. The task of this work is divided into two subparts. The first one is the identification of named entities from the raw text and the second one is the extraction of relations amongst the entities in a sentence. Convolutional neural networks are used to build the model. The network is designed with four convolutional layers, two max-pooling layers, and two dense layers. The first layer is a convolutional layer for its ability to capture the local context. Relu is used as the activation function to bring nonlinearity. The final dense layer is associated with softmax activation units. CNN automatically learns the values of its filters based on the labels on the training samples. The Adam is used as the optimizer. Dropout is used to prevent overfitting. Word2vec model is built using the Skip-gram model with a context window size of 10.

The proposed approach was evaluated using the data set given by IECSIL@FIRE2018 shared task. The evaluation results show that the performance of the proposed system could outperform an average accuracy of 85.62%. One of the shortcomings with basic CNNs is their inability to model long distance dependencies, which is important for various RE tasks.

Thenmozhi *et al*. [17] presented a deep learning approach for Named Entity Recognition and relation extraction in five Indian languages namely Hindi, Kannada, Malayalam, Tamil, and Telugu. The proposed approach is commonly working for both NER and RE tasks. However, in this review, we give high consideration to the relation extraction part only. neural machine translation architecture is used to implement the methodology for these tasks. A deep learning approach based on sequence to sequence (Seq2Seq) Model is used. To build a deep neural network model, a multi-layer RNN (Recurrent Neural Network) with LSTM (Long Short Term Memory) as a recurrent unit is used. This neural network consists of several layers namely, embedding layer, encoding layer, decoding layer, projection layer, or softmax layer, and loss layer. The embedding layer learns weight vectors from the input sentence and NER / RE label input sequence based on their vocabulary. These embeddings are fed into multi-layer LSTM where encoding and decoding are performed. The attention mechanism is used to handle the longer sentences. The softmax layer or projection layer is a dense layer to obtain

the NER / RE label output sequence. The loss layer is used to compute the training loss during model building. Once, the model is built, the NER / RE label output sequences are obtained by using the model for sequence mapping. For the RE task, the target sequence is itself a RE label and thus it is not required to do any post process the output of the deep neural network. The input sentences and NER / RE label input sequences are constructed based on the delimiter "newline".

The proposed approach was evaluated using the data set given by IECSIL@FIRE2018 shared task. The proposed approach classifies the the relations between NEs to one of the classes namely information_1, information_2, information_3, information_4, information_per, information_quant, information_closed, information_so, information_neg, information_cc, action_1, action_2, action_3, action_per, action_so, action_quant, action_neg, and other. They have evaluated two deep learning models for RE by varying the directionality, depth, and number of training steps with and without attention mechanism called scaled-luong.

They have implemented several variations of the Seq2Seq model with a dropout of 0.2 and batch size of 128 to show the effectiveness of the proposed model of 8-layer, BLSTM with attention. They have implemented a total of five variations for finding the relation labels for the given sentences to show the effectiveness of the proposed model. The first is observed that except for the "Telugu" language, bi-directional LSTM with attention having 8 layers depth performs well for all other languages in RE task. For RE task, they have obtained the accuracies as 56.19%, 60.74%, 60.7%, 75.43% and 79.11% for Model 1, Model 2, Model 3, Model 4 and Model 5 respectively on pre-evaluation test data, and 79.44%, 76.01% and 61.11% for Run 1, Run 2 and Run 3 respectively on final-evaluation test data.

3.4 Relation Extraction from Amharic language

To the best of our knowledge, relation extraction system for the Amharic language is not done separately. However, Bekele Worku [19] proposed an information extraction system from Amharic language text using a knowledge-poor approach. In this work, relation extraction is considered as a sub-component of the information extraction task. But the work gives more attention to named entity recognition and less attention to relation extraction. Because they did not state the type of dataset they used, the type of relations, and the performance achieved for relation extraction. Generally, the work talks about relation extraction are finished only in

a single page but relation extraction is a complex task that needs a detailed analysis. Accordingly, it is better to say relation extraction is out of scope on this work.

However, the knowledge-poor approach is used for Amharic information extraction system. The approaches that they used do not apply to a general domain. The author used the infrastructure domain as a data source for training and testing of the proposed system. Multiple hand built rules are used, which specified different patterns, and then the text is matched against those patterns and if a match is found, the element of the text is extracted. Both rules and gazetteers are used for entities, which have a consistent pattern like person name, which proceeds by title. The work lacks generalization because of the manually designed classification features. The system achieves the F-measure of 89.1 % on the named entity recognition and overall, it achieves the F-measure of 89.8%.

3.5 Summery

In this section, we conducted a review of different research attempts to develop relation extraction for various languages. These relation extraction researches are language-dependent. The review also showed that two major approaches are followed for relation extraction. The first approach is the traditional non-deep learning models, and the second is deep learning models. The traditional non-deep learning models require more feature engineering than deep Learning models and a labor- and time-intensive task. The traditional non-deep learning methods depend upon existing natural language processing systems lexical resources for feature extraction. The feature space on which the model should be trained is sparse and high dimensional which decreases the performance of the model. Such a pipelined approach causes a cascaded error and retards the performance of the system. The manually constructed features may not capture all relevant information. Bekele Workus's [19] work is under the category of non-deep learning approach which has all the above limitations.

On the other hand, deep learning models have outperformed previous models by using automatically extracted features. In particular, many researchers attempted to solve problems that happened on the traditional non-deep learning methods based on end-to-end models using only raw sentences and word embedding as an input to the deep neural networks. Among the vast deep network types, convolutional neural networks (CNNs) and recurrent neural networks (RNNs) are more common in relation extraction tasks. The reason for this popularity

is the ability of CNNs in learning local patterns and the power of RNNs in sequential modeling. Although RNNs are suitable for relation extraction system, they suffer from vanishing and exploding gradients when there are long-term dependencies in the input data. Such dependencies are common in most NLP applications and specifically in relation extraction. To address this problem, long short-term memory (LSTM) network was introduced. Due to its potential to solve the problems of standard RNNs, LSTM has attracted the attention of RE researchers. To consider both the preceding and succeeding contexts, bidirectional LSTM (Bi-LSTM) was proposed. By combining the forward and backward hidden layers, these models can better address the sequential modeling problem.

Although Bi-LSTM is extensively used in semantic RE system, it has two main drawbacks: (1) the high dimensional input space common in text processing applications increases the complexity of the model and makes it difficult to optimize; (2) the model cannot focus on the important parts of the contextual information of the text. To address these problems, we proposed attention base Bi directional LSTM. Specifically, our proposed model extracts both long and short intra sentence relations, considers forward and backward contextual dependencies, selects the most important features, and pays more or less attention to different words in sentences.

In conclusion, deep learning models are becoming important due to their demonstrated success in tackling complex learning problems. Relation extraction has the potential of employing deep learning models with the creation of huge datasets. Thus, we hypothesis that a deep learning approach can achieve good performance for Amharic semantic relation extraction system. Generally, the Major differences between all the above approaches are available resources, degree of pre-processing, features used, classification algorithm, and the nature of training/test data.

Chapter 4: The Proposed Solution

4.1 Introduction

In this section, we discuss the proposed model for the Amharic semantic relation extraction system using Attention-based Bi-directional LSTM networks (Att-BLSTM). The proposed approach aims at automating the process of relation extraction by automatically learning features from given data. In this chapter, the details of the model are discussed. Firstly, the proposed model is introduced. Secondly, text pre-processing activities such as cleaning, tokenization, and stop-word removal are discussed. Thirdly, the interconnection and usage of each component in the model are discussed in its subsection.

4.2 The Proposed Model

In this section, the Amharic semantic relation extraction model is designed as shown in Figure 4.1. Attention based Bi-directional Long Short Term Memory (Att-BLSTM) model is used as a deep learning approach. In the proposed model, word embedding, bidirectional LSTM, and attention mechanism are used to better capture both long-term dependencies and local features. The proposed model contains different components. These are the pre-processing, word embedding, feature extractor, model builder, and relation extractor. These components are briefly described in the next sections.

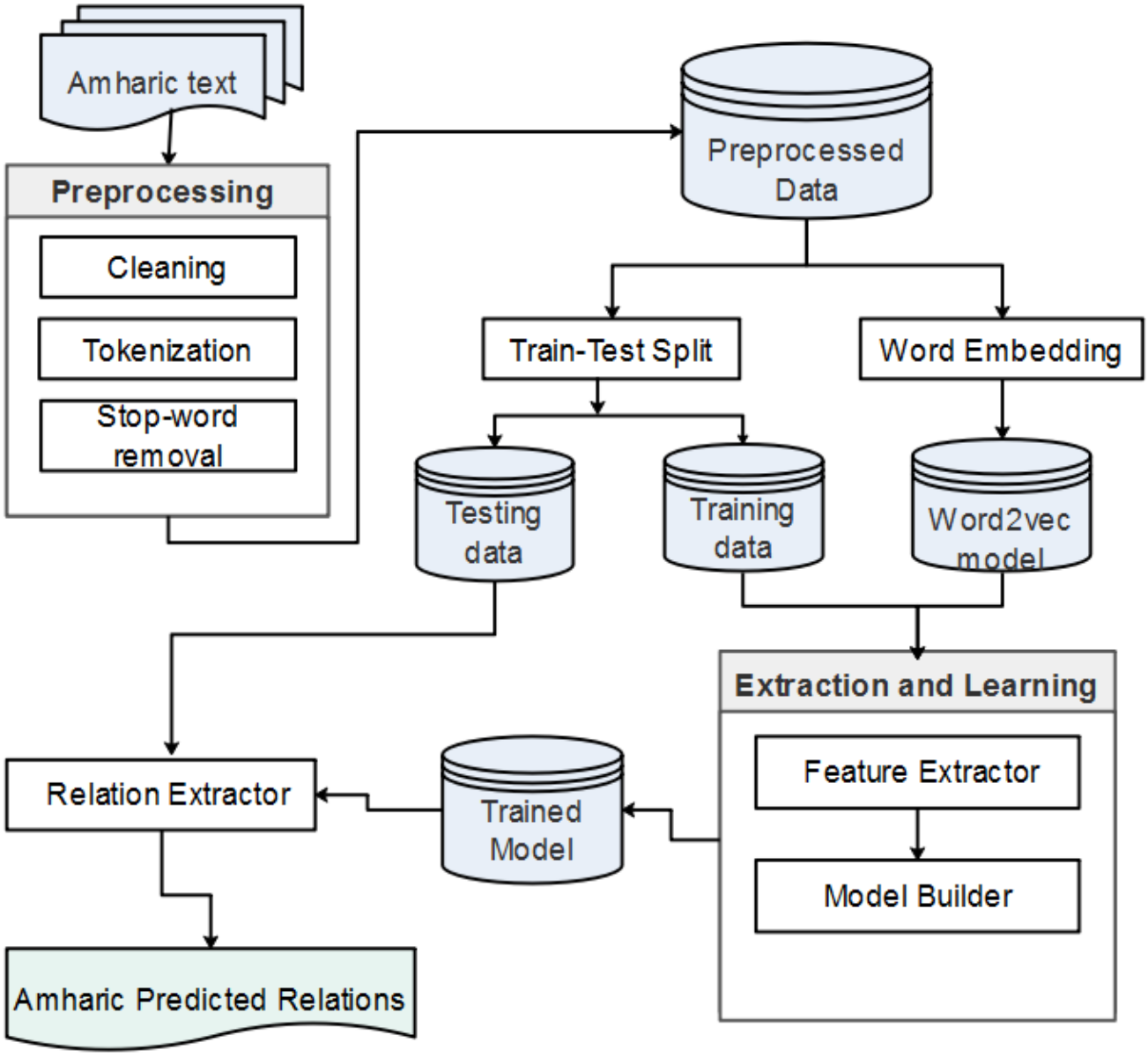

Figure 4. 1 Proposed model

4.2.1 Preprocessing

The input to the relation extraction task is the training dataset, the testing dataset. Using train/test splitter the 80% of the corpus is assigned as training data and 20% of the corpus is assigned as test data. The training datasets are also classified into two types, which are unlabeled (word embedding dataset) and labeled. Before proceeding to the next stages, the corpora have to pass through the main preprocessing stages. Preprocessing is about data cleaning or the task that is used to make the text data usable for analysis. Not all words in the text are important to create a word vector. For instance, text data often contains some special formats like number formats, date formats in language varied texts, and the most common words that do not help in relation extraction. Thus, text data must be preprocessed before usage. The proposed system preprocessing activities include cleaning, tokenization, and stop-word removal.

a. Cleaning

The first stage is the cleaning process, which removes non-Amharic characters from the collected Amharic corpus. Misspelled Amharic words were corrected based on the errors obtained in the quarter of the corpus. Because the existence of non-Amharic text affects the performance of automatic feature generation. Thus, all non-Amharic texts are removed from the corpora.

Algorithm 4.1: Cleaning

```
Input : Amharic-corpus
 Begin
   Read List of Amharic character
   For i=1 to number_of_character in Amharic-corpus
      If (Amharic corpus[i] !='Amharic character')
                Remove  Amharic-corpus[i];
       End If
   End For
End
Output: Cleaned Amharic Text
```

b. Tokenization

Tokenization is the process of breaking up the given text into units named as tokens or it describes splitting texts into sentences, or sentences into individual words. This is done by locating word boundaries (the ending point of a word and the beginning of the next word). The tokens may be words, numbers, punctuation marks, special symbols, etc. For example, in Amharic, a common way to split a text is by using a set of rules as a marker, like a whitespace or punctuation characters. In the old Amharic writing systems two dots '**:**' *ሁለት ነጥብ* was used to separate words and now it is replaced by white space. In this work, we used Amharic punctuation marks and white spaces for token identification. It also considers abbreviations and hyphenated words. For example, words like ጠ/ሚኒስትር, ቤተ-እስራኤል are taken as a single word. This tokenization is done using the NLTK word splitter.

Algorithm 4.2: Tokenization

```
Input: Amharic text corpus (F)
Begin
 For every word in F
 Find punctuation marks :: or ! or ? Or White space
         If punctuation marks or White space found
                    Split sentence
          End if
  End for
End

Output: Tokenized Amharic text into list of sentence
```

c. Stop-word Removal

Stop-words are words that occur most frequently in text data, but are not relevant or have no impact to discriminate among texts. The common stop-words in English such as *of*, *a*, and the, are not used to discriminate the text. Similar to other languages, the Amharic language also has different stop-words such as articles, prepositions, and conjunctions. For instance, there are common stop words in Amharic like **ስለ, ያለ**, *እና, ነገርግን,* etc., that are non-informative for generating word embedding. Using these words in a dataset as it is will have an impact on the

performance of the system, which means, they would degrade the speed and takes much memory space. Accordingly, Amharic stop-words are removed from a corpus of word embedding during pre-processing to reduce file size and processing time.

Algorithm 4.3: Stop-word Removal

```
Input: Preprocessed Amharic text corpus
Begin
  Define: List of stopword removal
  For i=1  to Number_of_words_in_the_document do
    For j=1 to Number_of_words_in_stopwords_list do
              If
              Words(i)== stopwords(j) THEN
                  Eliminate Words (i)
              End If
    End For
   End For
End
Output: Amharic text without stop-words
```

4.2.2 Word Embedding

After the unlabeled free text is preprocessed, the word embedding technique (word2vec) is used to map each word into a low dimension vector for automatic feature generation. The input is large unlabeled data, and this data will be tokenized and transformed into vectors by looking up word embeddings before it is used for training. From the word2vec result, we obtain contexts of words in a text by the interconnection of the different conceptual terms employed in a text.

For example: አምላክ -0.80265015 -4.579873 -1.8924758 2.6266816 -1.2619977 4.5167956 -4.8453174 4.860985 0.25822496 4.9696984 -7.1417923 6.1569505 2.5701008 -0.34958035 -0.9509108 1.5546086 -0.07761205 1.7100661 -3.4300444 0.42643702 2.0351622 -4.813588 -8.5975065 -4.3541837 -1.1125429 -1.9960092 3.5772793 1.477576 1.8352228 -1.5485846 -1.3787919 2.5427954 -0.28419626 -3.3074884 -4.8087296 3.8217275 -0.56678814

0.14878649 -1.1845628 -3.6746871 3.1566443 -3.9175317 2.345862 1.663101 0.967679 -1.1112202 -0.7067857 -0.68540454 1.1258357 4.7810698 -0.82176614 0.15093876 3.5671198 0.24816103 -9.72841 -6.454803 -1.0030937 -1.492264 -4.14333 0.5066791 -0.68482125 -3.7299206 -5.23864 -1.4855512 -0.5239402 -7.1987276 0.6864746 -1.8605227 0.63035506 -2.8144867 6.7934465 0.3532275 -0.5433455 -2.354061 -3.1559749 6.1305223 -1.5361226 1.2906698 5.0474114 -5.3004446 0.018039592 -0.6841162 3.070167 0.34988788 2.364217 3.83013 -8.871779 -0.556482 7.4700646 2.3798497 0.6020115 -3.2397583 -5.894556 0.37060094 -1.4711056 3.310762 0.84031534 -2.6047578 -0.023461768 3.1727958

The above vector output is generated for the single word አምላክ which is related to እግዚአብሔር, ቤተ ክርስቲያን, ክርስቶስ, ኢየሱስ , ጌታ and so on. This stage is where the words semantic and syntactic relations are learned. Then the output sentence is feed into the next layer as a real-valued vector. This output file will be the source of features in the next stages of this proposed architecture.

Algorithm 4.4: Training Word Embedding

```
Begin
        Input: Amharic Text Corpus
                  Tokenize the text ()
                  add all text in one file F ()
                  Call Word Vector Function()
                       train F (Word2Vec (F)) ()
                  save the trained Model
End
        Output: Amharic word embedding model
```

4.2.3 Feature Extractor

The feature extractor is designed to identify and extract all the necessary features from the data. It helps to increase the confidence level of predicting a sentence as a set of predefined relation classes by providing the necessary information (features) for model builder during model building.

The traditional lexical level features primarily include the nouns themselves, the types of the pairs of nominals, and word sequences between the entities, the quality of which strongly depends on the results of existing NLP tools. Alternatively, this paper uses generic word embeddings as the source of base features. The feature extractor takes tokenized words from the preprocessing stage and retrieves the corresponding feature vector from the output of Word2vector model. For training purposes, it reads words and retrieves their word vector, then combines it with its relation extraction for training. For plain text, it also reads tokenized words and retrieves their feature vector. The extracted sentences are classified according to the relation type. The sentences then pass to a features extraction module which extracts the different features using word embedings.

For the relation extraction task, it is beneficial to access the future as well as the past context. However, standard LSTM networks process sequences in temporal order, they ignore future context. In this paper, we use BLSTM to get high-level features from the embedding layer. The network includes two sub-networks, which are forward and backward pass for the left and right sequence context respectively. A forward LSTM network, which encodes the context of an input sentence, and a backward LSTM network, which encodes the context of the reversed sentence. They are run in parallel: the forward LSTM inputs the words from x_1 , to x_T , and the backward LSTM inputs in reverse order from x_T back to x_1 .

Example: ለፍቺ ዋነኛው ምክንያት የገንዘብ ችግር ነው ::

In the sentence the first word ለፍቺ refers to x_1 , the second word ዋነኛው refers to x_2 ….and the last word ነው refers to x_T .

In this section, we propose the attention mechanism for relation extraction tasks which can find the crucial parts of the sentence. The words are the crucial parts of the sentence through which we can infer the relationship between the two entities.

Example: <e1>ህመሙ</e1> የተከሰተው በተበከለ ውሃ በሚታጠብበት ወይም በሚጠጣበት ጊዜ ወደ ቆዳ በሚገባ <e2>ጥገኛ</e2> ምክንያት ነው።.

In this example where two nominals <e1>ህመሙ</e1> and <e2>ጥገኛ</e2> are relatively far apart from each other, the attention layer can still assign a relatively high salience score for these two nominals. Thus, the model improves its prediction with the special position indicators <e1>ህመሙ</e1> and <e2>ጥገኛ</e2> as compared to a model that does not know the exact nominal locations. Besides, this layer can automatically learn which words are more important under different relation types by applying attention mechanism. This Amharic sentence is a type Cause-Effect(e2,e1) relation, and the words with the highest salience scores apart from target nominals are the word ምክንያት. This word helps to predict the relationship between two nominals is Cause-Effect(e2,e1). The attention layer generates a weight vector, and merges word-level features from each time step into a sentence-level feature vector, by multiplying the weight vector. Finally, all the extracted features are supplied to the model builder which will estimate the parameters of the model.

4.2.4 Model Builder

The main objective of the model builder is to build a trained model using deep learning algorithms. All the previously extracted features are used for the actual training process. The model builder does the process of training the model. It is designed to estimate the model coefficients based on the extracted features and then generate a trained model. We use the training data constructed in the previous stage to train the softmax classifier, which classifies Amharic sentence containing the two entities into a relation type, after extracting its features. The model contains a set of values that are obtained from the calculations performed during the estimation. The trained model, here, is a set of parameters (known as weights) that corresponds to the importance of the features used in the task. It is an estimation of all the parameters that are obtained through training. The trained model is the final output of the learning process and is used as an inference for the prediction process.

4.2.5 Relation Extractor

The trained model helps to assist the relation extractor by supplying the necessary information to extract relations from the testing data. The features extracted and stored during training are supplied to the relation extractor through the trained model to identify the relation type from the text.

The relation extractor performs two main tasks: relation detection and relation classification. Relation detection is the first task in relation extraction and it is the process of detecting relations if a relation occurs between the corresponding entity mention. Relation classification is the second task in relation extraction and it is the process of classifying the detected relation mentions into some predefined classes. We use a softmax classifier to predict a label from a discrete set of classes for a sentence. It produces the probability distribution p over relation types conditioned on the sentence representation s. The relation extractor selects candidate relations based on the calculated probability by invoking the trained model.

Finally, the outputs of Amharic predicted relations are displayed as follows:

Cause-Effect(e1,e2)
Cause-Effect(e2,e1)
Component-Whole(e1,e2)
Component-Whole(e2,e1)
Content-Container(e1,e2)
Content-Container(e2,e1)
Entity-Destination(e1,e2)
Entity-Destination(e2,e1)
Entity-Origin(e1,e2)
Entity-Origin(e2,e1)
Instrument-Agency(e1,e2)
Instrument-Agency(e2,e1)
Member-Collection(e1,e2)
Member-Collection(e2,e1)
Message-Topic(e1,e2)
Message-Topic(e2,e1)
Product-Producer(e1,e2)
Product-Producer(e2,e1)
Other

Chapter 5: Experimentation and Evaluation

5.1 Introduction

Evaluation is the process of measuring the qualities of an algorithm or a system. It has an important role in the relation extraction system for us and other researchers. For instance, it helps us to know our system performance and compare it with previous research work. For other researchers also, provides them with the necessary information they need to easily compare alternative systems and choose the one that meets their requirements. In this chapter data collection, dataset preparation, development tools, the experimental scenarios, and evaluation results are discussed. We explored different hyper-parameters, here we present those for which the experiments are done. In this chapter, the results obtained from the models are discussed. The evaluation metrics are then applied to the results of the models. As we explained in section 2.6, using different evaluation metrics can reflect a better interpretation of the evaluation of the results.

5.2 Data Collection and Dataset Preparation

The experimentation of the proposed system begins with the collection of Amharic text corpora from different data sources. These are collected from the Amharic bible, news agencies, broadcasting media, online newspapers, Wikipedia, and magazines. The collected Amharic text corpora are from different domains such as politics, religion, technology, business, health, art, sport, and so on. Thus, the collected data is heterogeneous and writer independent. The collected Amharic text corpora are for word embedding, training, and testing. Two types of corpora are used for training purposes. The first corpus is unlabeled Amharic text for word vector generation and the second corpus is labeled Amharic text.

To the best of our knowledge, there is no authorized and publicly available dataset for the Amharic semantic relation extraction system. Accordingly, Amharic semantic relation extraction dataset is prepared in this study. This dataset is named as Amharic semantic relation extraction dataset (Amharic-RE-Dataset). Amharic-RE-Dataset shares some concepts from SemEval-2010 Task 8 dataset [24].

The Amharic-RE-Dataset contains 19 labels: 9 directed relations, and an undirected Other class. The 9 directed semantic relation types are Cause-Effect, Component-Whole, Content-Container, Entity- Destination, Entity-Origin, Instrument-Agency, Member-Collection, Message-Topic, and Product-Producer. All these relation types are explained with examples as follows.

Cause-Effect (CE): An event or object leads to an effect. The cause should be actively involved in the process of causing the effect. When an effect is caused by a combination of events, each such event is considered a separate cause for the effect.

Example: የባቡር <e1>አደጋው</e1> የተፈጠረው በሽብርተኞች <e2>ጥቃት</e2> ምክንያት ነው ።።

Instrument-Agency (IA): An agent uses an instrument. Both instrument and agency can be a physical object, an abstract object, or an organization. Properties, capabilities, attitudes, skills, etc. are not acceptable as instruments.

Example: <e1>ጠላቷ</e1> <e2>በብረት</e2> ገድሏታል ።።

Product-Producer (PP): A producer causes a product to exist. The producer should be actively involved in the process of bringing the product into existence and not just serve as a raw material. Tangible concrete objects, tangible substances, organizations, and positions in organizations are acceptable Products.

Example: <e1>ኩባንያው</e1> የፐላስቲክ <e2>ወንበሮችን</e2> ይሠራል።

Content-Container (CC): An object is physically stored in a delineated area of space. The word "contain" defines as "have within, hold, comprise, include". Thus, a container must enclose the thing it contains.

Example: <e1>ሳንቲሞች</e1> በፐላስቲክ <e2>ሳጥን</e2> ውስጥ ተቀምጠዋል።።

Entity-Origin (EO): An entity is coming or derived from an origin. The entity should have the ontologically type of entity (i.e., a physical or abstract object). The Origin can be abstract, physical, material, a person or a company and temporal.

Example: <e1>መረጃው</e1> <e2>ከፍላሽ</e2> ውስጥ ጠፍቷል።

Entity-Destination (ED): An entity is moving towards a destination. The Destination can be abstract, (but only if the Entity is abstract), physical (spatial/geographical) location, a person or company, and temporal.

Example: <e1>ስደተኛው</e1> ወደ <e2>ዲንካን</e2> ገብቷል ።

Component-Whole (CW): An object is a component of a larger whole. Where a component and whole is a physical object, an abstract object or an organization that has a certain structure or organization. Component has a clear boundary and could, in principle, be separated from the whole.

Example: <e1>በመጽሐፉ</e1> ውስጥ ያለው <e2>መግቢያ</e2> የመጽሐፉ ማጠቃለያ ነው።

Member-Collection (MC): A member forms a nonfunctional part of a collection. Members are not functional to the collection; they are not in a specific spatial/temporal position with respect to each other. Members and collection are separable; members can, in principle, be separated from the Collection.

Example: አንድ <e1>ማህበረሰብ</e1> በብዙ <e2>ግለሰቦች</e2> የተገነባ ነው ።

Message-Topic (MT): A message, written or spoken, is about a topic. A Message has descriptive, significate, or propositional content and has been produced to communicate this content.

Example: የመጨረሻው <e1>ምዕራፍ</e1> ስለ <e2>ሥነ-መለኮታዊ</e2> ጥናት ያቀርባል ።

An undirected **Other** class means that the relation does not belong to any of the nine relation types.

Example: አንድ <e1>ዘንግ</e1> አንድ<e2>ጡንቻን</e2> ከአጥንት ያያይዛል ።

The relation is directional, which means that Component-Whole (e1, e2) is different from Component-Whole (e2, e1). <e1>, </e1>, <e2> and </e2> are the four position indicators which help the model to know the position of target nominals. Our objective is to annotate instances of semantic relations that are true in the sense of holding in the most plausible truth-conditional interpretation of the sentence.

Finally, the goal is to create domain independent Amharic semantic relation extraction dataset. It helps to have the dilemma of lack of corpus in Amharic semantic relation extraction.

Thus, experiments of the proposed system are conducted on this Amharic-RE-Dataset. Hence, the gold-standard labels (Cause-Effect (e1, e2) and so on) should be provided to a system at training time but not at test time. For evaluation train test split using 80% of the corpus as training data and 20% of the corpus as test data. Categorical cross-entropy objective function with adam optimizer is used for the training process. The Sample Amharic-RE-Dataset is given in Annex A. The task is to predict, given a sentence and two tagged entities, which of the relation labels to apply. The predictions of the system are in the format as generated in Annex B.

5.3 Development Tools and Programming Languages

In the process of this research, many development tools are used. The tools that have been used include Tensorflow deep learning library, Anaconda, Genism**,** Keras deep learning library, etc.

a. Anaconda

In this experimentation, we have used the Anaconda programming tool. It is a free and open-source distribution of the Python and R programming languages for deep learning related applications.

b. Genisim

The Amharic text corpus is trained using Gensim for word vector generation, which is an open source vector space modeling and topic modeling toolkit implemented in Python. In Gensim a corpus is simply an object, when iterated over, returns its documents represented as sparse vectors.

c. Tensorflow

Tensorflow is Google's open-source machine learning library. It includes python API's. It has plenty of abstraction power but users might also be working with computational primitive wrappers such as matrix operations, element-wise math operators, and looping control. Tensorflow considers networks as a directed graph of nodes with data flow computation and dependencies encapsulated in it. Tensorflow is used as a back end for Keras library which is used for the development of deep neural network classifiers.

d. Keras

Keras is a Python machine learning library for deep learning that can run using Theano or TensorFlow as a back end. Its main focus is enabling the implementation of deep learning models as fast and easy as possible for research and development. Keras is used for the development of deep neural network classifiers for the proposed Amharic semantic RE system.

e. Pandas

Pandas is a Python package that presents an easy way of working with relational or labeled data by providing fast, flexible, and expressive data structures. It has two primary data structures: Series (1-dimensional) and DataFrame (2-dimensional). The 2-dimensional data structure is used in this study which then is converted into 3-dimensional data by using the NumPy package of Python. The LSTM and Bi-LSTM deep neural networks require a 3-dimensional input; hence, the 2-dimensional data was changed into 3-dimensional data.

f. Scikit learn

Scikit-learn is a Python machine learning library that includes a wide range of state-of-the-art deep learning algorithms for supervised and unsupervised problems. It focuses on bringing machine learning to non-specialists by providing a general-purpose high-level language. It is easy to use, has high performance. Scikit learn is used to evaluate deep neural classifiers by calculating performance metrics.

g. Python Programming Languages

Python is a powerful high-level, object-oriented programming general-purpose language. It has a wide range of applications from Web, scientific and mathematical computing to desktop graphical user Interfaces. In this experimentation process, we have used Python for deep learning relation extraction tasks.

h. Experimental setup

The laptop machine is used for experimenting with this study. It has a memory capacity of 4 GB RAM, 2.2 GHz processor, and 64-bit Operating system, x64 based processor.

5.4 Hyperparameters

Hyperparameters are properties of training data that will learn on its own during training by the classifier or other deep learning models. They include variables that determine the network structure (e.g: number of hidden units) and the variables which determine how the network is trained (e.g: learning rate). Hyperparameters are important since they directly control the behavior of the training algorithm, having an important impact on the performance of the model under training [50].

Choosing appropriate hyperparameters plays a key role in the success of neural network architectures, given the impact on the learned model. Choosing good hyperparameters provides benefits such as efficient search across the space of possible hyperparameters; and to make the experiment manageable [51, 52]. For example, the following are some model inbuilt configuration variables:

Learning Rate: The learning rate defines how quickly a network updates its parameters. A low learning rate slows down the learning process and the model will miss the important patterns in the data but converges smoothly. A larger learning rate speeds up the learning but may not converge.

Batch Size: Mini batch size is the number of subsamples given to the network after which parameter update happens.

Epochs: The number of epochs is the number of times the whole training data is shown to the network while training.

Hidden Layers and Units: Hidden layers are the layers between the input layer and the output layer. Many hidden units within a layer with regularization techniques can increase accuracy. A smaller number of units may cause underfitting.

Dropout: Dropout is a regularization technique to prevents the co-adaptation of hidden units by randomly omitting feature detectors from the network during forward propagation [53]. Drop out is an effective tool to enhance generalizability. We employ dropout on the embedding layer and LSTM layer.

For all the experiments of the proposed model, the following hyperparameters are used as shown in Table 5.1.

Table 5. 1 Hyperparameters

Hyperparameters	**Value**
Learning rate	1.0
Batch size	10
num_epochs	100
dev_sample_percentage	0.1
embedding_dim	100
emb_dropout_keep_prob	0.7
hidden_size	100
rnn_dropout_keep_prob	0.7
dropout_keep_prob	0.5
l2_reg_lambda	1e-5
display_every	10
evaluate_every	100
num_checkpoints	5
decay_rate	0.9

5.5 Experimental Scenarios

To get a more comprehensive understanding of our model performance, we choose four variations of experiments. Att-BLSTM model and Softmax classifier are used for each variation. The Amharic-RE-Dataset is used for experimenting each variation. The influence of each variation is testify for the relation extraction by removing one type of variation from the proposed models each time.

The four experimental variations are described in the following:

(1) As a (2*9+1)-way classification, with directionality taken into account and with word embedding.

(2) As a (9+1)-way classification, with directionality ignored and with word embedding

(3) As a (9+1)-way classification, with directionality taken into account and without word embedding.

(4) As a (9+1)-way classification, with directionality taken into account and with word embedding.

5.6 Experimental Results

As stated above in section 5.2, the experiment results on these four separate steps of relation extraction are presented in the following subsections.

5.6.1 As (2*9+1)-Way Classification With Word Embedding and Directionality

(i) Confusion Matrix, Coverage, and Accuracy

. As stated in section 2.6, confusion matrics shows how the classification model is confused when it makes predictions. The number of correct and incorrect predictions are summarized with count values and broken down by each relation class as shown below in Figure 5.1, Figure 5.3, Figure 5.5, and Figure 5.7. These are tables with different combinations of predicted and actual values. It is extremely useful for measuring precision, and recall.

```
<<< (2*9+1)-WAY EVALUATION (USING DIRECTIONALITY)>>>:

Confusion matrix:
      C-E1 C-E2 C-W1 C-W2 C-C1 E-D1 E-D2 E-O1 E-O2 I-A1 I-A2 M-C1 M-C2 M-T1 M-T2 P-P1 P-P2  _O_ *CC2 <-- classified as
     +-------------------------------------------------------------------------------------------+ -SUM- skip ACTUAL
 C-E1 |  14    0    0    0    0    0    0    1    0    0    0    0    1    0    0    0    0    0    0 |   16    0   16
 C-E2 |   0    5    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0 |    5    0    5
 C-W1 |   0    0    4    0    1    0    0    0    0    0    0    0    1    0    0    0    0    0    0 |    6    0    6
 C-W2 |   0    0    0   20    0    0    0    2    0    0    0    1    1    0    0    0    0    0    0 |   24    0   24
 C-C1 |   0    0    0    0   19    0    0    1    0    0    0    0    0    0    0    0    0    0    1 |   21    0   21
 E-D1 |   0    0    0    0    1   25    0    0    0    0    0    0    1    0    0    0    0    0    0 |   27    0   27
 E-D2 |   0    0    0    0    0    0    2    0    0    0    0    0    0    0    0    0    0    0    0 |    2    0    2
 E-O1 |   0    0    0    0    0    0    0   17    0    0    1    0    0    0    0    0    0    0    0 |   18    0   18
 E-O2 |   0    0    0    0    0    0    0    0    6    0    0    1    0    0    0    0    0    0    1 |    8    0    8
 I-A1 |   0    0    0    0    0    0    0    0    0    8    0    0    0    0    0    0    0    0    0 |    8    0    8
 I-A2 |   0    0    0    0    0    0    1    0    0    0   10    1    0    0    0    0    0    0    0 |   12    0   12
 M-C1 |   0    0    0    0    1    0    0    0    0    0    0    8    0    0    0    0    0    0    0 |    9    0    9
 M-C2 |   0    0    0    0    0    0    0    0    0    0    0    2    6    0    0    0    0    0    0 |    8    0    8
 M-T1 |   0    0    0    0    0    0    0    0    0    0    0    0    0   19    0    0    0    1    0 |   20    0   20
 M-T2 |   0    0    0    0    1    0    0    0    0    0    0    0    0    0    7    0    0    0    0 |    8    0    8
 P-P1 |   0    0    1    0    0    0    0    1    0    0    0    0    1    0    0   10    0    1    0 |   14    0   14
 P-P2 |   0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    9    0    0 |    9    0    9
  _O_ |   0    0    0    0    0    3    0    0    0    0    1    0    1    0    0    0    0   30    0 |   35    0   35
     +-------------------------------------------------------------------------------------------+
 -SUM-   14    5    5   20   23   28    3   22    6    8   12   13   12   19    7   10    9   32    2   250    0  250

Coverage = 250/250 = 100.00%
Accuracy (calculated for the above confusion matrix) = 219/250 = 87.60%
Accuracy (considering all skipped examples as Wrong) = 219/250 = 87.60%
Accuracy (considering all skipped examples as Other) = 219/250 = 87.60%
```

*Figure 5. 1 Confusion matrix, coverage and accuracy with directionality and word embedding as a (2*9+1)-way classification*

In above Figure 5.1 where the abbreviations are represented as follows:

C-E1 = Cause-Effect(e1,e2)

C-E2 = Cause-Effect(e2,e1)

C-W1 = Component-Whole(e1,e2)

C-W2= Component-Whole(e2,e1)

C-C1 = Content-Container(e1,e2)

C-C2= Content-Container(e2,e1)

E-D1= Entity-Destination(e1,e2)

E-D2 = Entity-Destination(e2,e1)

E-O1 = Entity-Origin(e1,e2)

E-O2 = Entity-Origin(e2,e1)

I-A1= Instrument-Agency(e1,e2)

I-A2 = Instrument-Agency(e2,e1)

M-C1 = Member-Collection(e1,e2)

M-C2 = Member-Collection(e2,e1)

M-T1 = Message-Topic(e1,e2)

M-T2 = Message-Topic(e2,e1)

P-P1 = Product-Producer(e1,e2)

P-P2 = Product-Producer(e2,e1)

_O_= Other

As shown in Figure 5.1, Figure 5.3, Figure 5.5 and Figure 5.7 the scorer calculates and outputs the following statistics in confusion matrix, which shows:

- ✓ SUM-: the sums for each row/column
- ✓ skip: the number of skipped examples
- ✓ xDIRx: the number of examples with correct relation, but wrong directionality
- ✓ ACTUAL (= -SUM- + skip + xDIRx) the number of examples in the answer key file
- ✓ Finally the accuracy and coverage

(ii) Results for the Individual Relations

```
Results for the individual relations:
      Cause-Effect(e1,e2) :    P =   14/  14 = 100.00%     R =   14/  16 =  87.50%     F1 =  93.33%
      Cause-Effect(e2,e1) :    P =    5/   5 = 100.00%     R =    5/   5 = 100.00%     F1 = 100.00%
   Component-Whole(e1,e2) :    P =    4/   5 =  80.00%     R =    4/   6 =  66.67%     F1 =  72.73%
   Component-Whole(e2,e1) :    P =   20/  20 = 100.00%     R =   20/  24 =  83.33%     F1 =  90.91%
 Content-Container(e1,e2) :    P =   19/  23 =  82.61%     R =   19/  21 =  90.48%     F1 =  86.36%
Entity-Destination(e1,e2) :    P =   25/  28 =  89.29%     R =   25/  27 =  92.59%     F1 =  90.91%
Entity-Destination(e2,e1) :    P =    2/   3 =  66.67%     R =    2/   2 = 100.00%     F1 =  80.00%
     Entity-Origin(e1,e2) :    P =   17/  22 =  77.27%     R =   17/  18 =  94.44%     F1 =  85.00%
     Entity-Origin(e2,e1) :    P =    6/   6 = 100.00%     R =    6/   8 =  75.00%     F1 =  85.71%
 Instrument-Agency(e1,e2) :    P =    8/   8 = 100.00%     R =    8/   8 = 100.00%     F1 = 100.00%
 Instrument-Agency(e2,e1) :    P =   10/  12 =  83.33%     R =   10/  12 =  83.33%     F1 =  83.33%
 Member-Collection(e1,e2) :    P =    8/  13 =  61.54%     R =    8/   9 =  88.89%     F1 =  72.73%
 Member-Collection(e2,e1) :    P =    6/  12 =  50.00%     R =    6/   8 =  75.00%     F1 =  60.00%
     Message-Topic(e1,e2) :    P =   19/  19 = 100.00%     R =   19/  20 =  95.00%     F1 =  97.44%
     Message-Topic(e2,e1) :    P =    7/   7 = 100.00%     R =    7/   8 =  87.50%     F1 =  93.33%
  Product-Producer(e1,e2) :    P =   10/  10 = 100.00%     R =   10/  14 =  71.43%     F1 =  83.33%
  Product-Producer(e2,e1) :    P =    9/   9 = 100.00%     R =    9/   9 = 100.00%     F1 = 100.00%
                  _Other :    P =   30/  32 =  93.75%     R =   30/  35 =  85.71%     F1 =  89.55%

Micro-averaged result (excluding Other):
P =  189/ 216 =  87.50%     R =  189/ 215 =  87.91%     F1 =  87.70%

MACRO-averaged result (excluding Other):
P =  87.69%     R =  87.72%     F1 =  86.77%
```

*Figure 5. 2 Results for the individual relations with directionality and word embedding as a (2*9+1)-way classification*

As shown in Figure 5.2, Figure 5.4, Figure 5.6 and Figure 5.8 the scorer calculates and outputs the following statistics:

- ✓ Precision (P), recall (R), and F1-score for each relation
- ✓ Micro-averaged P, R, F1, where the calculations ignore the Other category.
- ✓ Macro-averaged P, R, F1, where the calculations ignore the Other category

The nine relations contain a sentence marked with two nominals e1 and e2, and the task is about predicting the relation between the two nominals and taking into account the

directionality. That means that the relation Cause-Effect(e1,e2) is different from the relation Cause-Effect(e2,e1), as shown in the examples below.

<e1>በመፈንቅለ መንግስቱ</e1> ምክንያት በተነሳው <e2>ግራ መጋባት</e2> መካከል አርቲስቶች እና ድርጅቶች አመጹን በመቃወም የነበራቸውን ተሳትፎ የሚያወጁበት መንገድ ነበር ። ⟶ Cause-Effect(e1,e2)

<e1>ህመሙ</e1> የተከሰተው በተበከለ ውሃ በሚታጠብበት ወይም በሚጠጣበት ጊዜ ወደ ቆዳ በሚገባ <e2>ጥገኛ</e2> ምክንያት ነው። ⟶Cause-Effect(e2,e1).

5.6.2 As (9+1)-Way Classification, Without Directionality and with Word Embedding

(i) Confusion Matrix, Coverage, and Accuracy

```
<<< (9+1)-WAY EVALUATION IGNORING DIRECTIONALITY >>>:

Confusion matrix:
        C-E  C-W  C-C  E-D  E-O  I-A  M-C  M-T  P-P  _O_ <-- classified as
      +-------------------------------------------------+ -SUM- skip ACTUAL
  C-E |  19    0    0    0    1    0    1    0    0    0 |   21    0   21
  C-W |   0   24    1    0    2    0    3    0    0    0 |   30    0   30
  C-C |   0    0   20    0    1    0    0    0    0    0 |   21    0   21
  E-D |   0    0    1   27    0    0    1    0    0    0 |   29    0   29
  E-O |   0    0    1    0   23    1    1    0    0    0 |   26    0   26
  I-A |   0    0    0    1    0   18    1    0    0    0 |   20    0   20
  M-C |   0    0    1    0    0    0   16    0    0    0 |   17    0   17
  M-T |   0    0    1    0    0    0    0   26    0    1 |   28    0   28
  P-P |   0    1    0    0    1    0    1    0   19    1 |   23    0   23
  _O_ |   0    0    0    3    0    1    1    0    0   30 |   35    0   35
      +-------------------------------------------------+
 -SUM-   19   25   25   31   28   20   25   26   19   32    250    0  250

Coverage = 250/250 = 100.00%
Accuracy (calculated for the above confusion matrix) = 222/250 = 88.80%
Accuracy (considering all skipped examples as Wrong) = 222/250 = 88.80%
Accuracy (considering all skipped examples as Other) = 222/250 = 88.80%
```

Figure 5. 3 Confusion matrix, coverage and accuracy without directionality and with word embedding as a (9+1)-way classification

(ii) **Results for the Individual Relations**

```
Results for the individual relations:
             Cause-Effect :    P =   19/  19 = 100.00%     R =   19/  21 =  90.48%
          Component-Whole :    P =   24/  25 =  96.00%     R =   24/  30 =  80.00%
        Content-Container :    P =   20/  25 =  80.00%     R =   20/  21 =  95.24%
      Entity-Destination :    P =   27/  31 =  87.10%     R =   27/  29 =  93.10%
           Entity-Origin :    P =   23/  28 =  82.14%     R =   23/  26 =  88.46%
       Instrument-Agency :    P =   18/  20 =  90.00%     R =   18/  20 =  90.00%
       Member-Collection :    P =   16/  25 =  64.00%     R =   16/  17 =  94.12%
           Message-Topic :    P =   26/  26 = 100.00%     R =   26/  28 =  92.86%
        Product-Producer :    P =   19/  19 = 100.00%     R =   19/  23 =  82.61%
                  _Other :    P =   30/  32 =  93.75%     R =   30/  35 =  85.71%

Micro-averaged result (excluding Other):
P =  192/ 218 =  88.07%     R =  192/ 215 =  89.30%     F1 =  88.68%

MACRO-averaged result (excluding Other):
P =  88.80%     R =  89.65%     F1 =  88.60%
```

Figure 5. 4 Results for the individual relations without directionality and with word embedding as a (9+1)-way classification

5.6.3 As (9+1)-Way Classification, With Directionality and Without Word Embedding.

(i) Confusion Matrix, Coverage, and Accuracy

```
Confusion matrix:
        C-E  C-W  C-C  E-D  E-O  I-A  M-C  M-T  P-P  _O_ <-- classified as
      +-------------------------------------------------+ -SUM- xDIRx skip  ACTUAL
  C-E |  18    1    0    0    0    0    1    1    0    0 |   21     0     0     21
  C-W |   0   24    1    0    1    1    0    0    0    2 |   29     1     0     30
  C-C |   0    0   18    0    2    0    1    0    0    0 |   21     0     0     21
  E-D |   0    0    0   27    1    0    0    0    0    1 |   29     0     0     29
  E-O |   0    0    0    0   24    0    1    0    1    0 |   26     0     0     26
  I-A |   0    0    0    0    0   18    2    0    0    0 |   20     0     0     20
  M-C |   0    1    1    0    2    0   12    0    0    0 |   16     1     0     17
  M-T |   0    0    1    0    1    0    1   24    0    1 |   28     0     0     28
  P-P |   0    0    0    0    1    3    1    0   17    1 |   23     0     0     23
  _O_ |   0    1    1    1    0    0    1    0    0   31 |   35     0     0     35
      +-------------------------------------------------+
 -SUM-   18   27   22   28   32   22   20   25   18   36    248     2     0    250

Coverage = 250/250 = 100.00%
Accuracy (calculated for the above confusion matrix) = 213/250 = 85.20%
Accuracy (considering all skipped examples as Wrong) = 213/250 = 85.20%
Accuracy (considering all skipped examples as Other) = 213/250 = 85.20%
```

Figure 5. 5 Confusion matrix, coverage and accuracy with directionality and without word embedding as a (9+1)-way classification

(ii) **Results for the Individual Relations**

```
Results for the individual relations:
             Cause-Effect :    P =   18/(  18 +   0) = 100.00%     R =   18/  21 =  85.71%     F1 =  92.31%
          Component-Whole :    P =   24/(  27 +   1) =  85.71%     R =   24/  30 =  80.00%     F1 =  82.76%
        Content-Container :    P =   18/(  22 +   0) =  81.82%     R =   18/  21 =  85.71%     F1 =  83.72%
       Entity-Destination :    P =   27/(  28 +   0) =  96.43%     R =   27/  29 =  93.10%     F1 =  94.74%
            Entity-Origin :    P =   24/(  32 +   0) =  75.00%     R =   24/  26 =  92.31%     F1 =  82.76%
         Instrument-Agency :    P =   18/(  22 +   0) =  81.82%     R =   18/  20 =  90.00%     F1 =  85.71%
         Member-Collection :    P =   12/(  20 +   1) =  57.14%     R =   12/  17 =  70.59%     F1 =  63.16%
            Message-Topic :    P =   24/(  25 +   0) =  96.00%     R =   24/  28 =  85.71%     F1 =  90.57%
          Product-Producer :    P =   17/(  18 +   0) =  94.44%     R =   17/  23 =  73.91%     F1 =  82.93%
                    _Other :    P =   31/(  36 +   0) =  86.11%     R =   31/  35 =  88.57%     F1 =  87.32%

Micro-averaged result (excluding Other):
P =  182/ 214 =  85.05%     R =  182/ 215 =  84.65%     F1 =  84.85%

MACRO-averaged result (excluding Other):
P =  85.37%     R =  84.12%     F1 =  84.29%
```

Figure 5. 6 Results for the individual relations using directionality and without word embedding as a (9+1)-way classification

5.6.4 As (9+1)-Way Classification, With Directionality and Word Embedding

Most relation classification models rely on high-level lexical and syntactic features obtained by NLP tools such as WordNet, the dependency parser, part-of-speech (POS) tagger, and named entity recognizers (NER). Besides, state-of-the-art neural models based on attention mechanisms do not fully utilize information related to the entity, which may be the most crucial feature for relation extraction. To address these issues, in this study a relation has directionality.

(i) Confusion Matrix, Coverage, and Accuracy

```
<<< (9+1)-WAY EVALUATION TAKING DIRECTIONALITY INTO ACCOUNT -- OFFICIAL >>>:

Confusion matrix:
         C-E  C-W  C-C  E-D  E-O  I-A  M-C  M-T  P-P  _O_ <-- classified as
       +-------------------------------------------------+ -SUM- xDIRx skip  ACTUAL
  C-E  |  19    0    0    0    1    0    1    0    0    0 |   21     0    0     21
  C-W  |   0   24    1    0    2    0    3    0    0    0 |   30     0    0     30
  C-C  |   0    0   19    0    1    0    0    0    0    0 |   20     1    0     21
  E-D  |   0    0    1   27    0    0    1    0    0    0 |   29     0    0     29
  E-O  |   0    0    1    0   23    1    1    0    0    0 |   26     0    0     26
  I-A  |   0    0    0    1    0   18    1    0    0    0 |   20     0    0     20
  M-C  |   0    0    1    0    0    0   14    0    0    0 |   15     2    0     17
  M-T  |   0    0    1    0    0    0    0   26    0    1 |   28     0    0     28
  P-P  |   0    1    0    0    1    0    1    0   19    1 |   23     0    0     23
  _O_  |   0    0    0    3    0    1    1    0    0   30 |   35     0    0     35
       +-------------------------------------------------+
 -SUM-   19   25   24   31   28   20   23   26   19   32    247     3    0    250

Coverage = 250/250 = 100.00%
Accuracy (calculated for the above confusion matrix) = 219/250 = 87.60%
Accuracy (considering all skipped examples as Wrong) = 219/250 = 87.60%
Accuracy (considering all skipped examples as Other) = 219/250 = 87.60%
```

Figure 5. 7 Confusion matrix, coverage and accuracy with directionality and word embedding as a (9+1)-way classification

(iii)Results for the Individual Relations

```
Coverage = 250/250 = 100.00%
Accuracy (calculated for the above confusion matrix) = 219/250 = 87.60%
Accuracy (considering all skipped examples as Wrong) = 219/250 = 87.60%
Accuracy (considering all skipped examples as Other) = 219/250 = 87.60%

Results for the individual relations:
             Cause-Effect :    P =  19/( 19 +  0) = 100.00%     R =  19/ 21 =  90.48%     F1 =  95.00%
          Component-Whole :    P =  24/( 25 +  0) =  96.00%     R =  24/ 30 =  80.00%     F1 =  87.27%
        Content-Container :    P =  19/( 24 +  1) =  76.00%     R =  19/ 21 =  90.48%     F1 =  82.61%
      Entity-Destination :    P =  27/( 31 +  0) =  87.10%     R =  27/ 29 =  93.10%     F1 =  90.00%
           Entity-Origin :    P =  23/( 28 +  0) =  82.14%     R =  23/ 26 =  88.46%     F1 =  85.19%
       Instrument-Agency :    P =  18/( 20 +  0) =  90.00%     R =  18/ 20 =  90.00%     F1 =  90.00%
       Member-Collection :    P =  14/( 23 +  2) =  56.00%     R =  14/ 17 =  82.35%     F1 =  66.67%
           Message-Topic :    P =  26/( 26 +  0) = 100.00%     R =  26/ 28 =  92.86%     F1 =  96.30%
        Product-Producer :    P =  19/( 19 +  0) = 100.00%     R =  19/ 23 =  82.61%     F1 =  90.48%
                  _Other :    P =  30/( 32 +  0) =  93.75%     R =  30/ 35 =  85.71%     F1 =  89.55%

Micro-averaged result (excluding Other):
P =  189/ 218 =  86.70%     R =  189/ 215 =  87.91%     F1 =  87.30%

MACRO-averaged result (excluding Other):
P =  87.47%     R =  87.82%     F1 =  87.06%
```

Figure 5. 8 Results for the individual relations with directionality and word embedding as a (9+1)-way classification

Figure 5.8 shows the final proposed model, which includes attention mechanism, Bidirectional LSTM, and word embedding as a feature set. It presents precision (P), recall (R), and F1-Score for each relation type, micro-averaged P, R, F1, and macro-averaged P, R, F1. For micro-averaged and macro-averaged, the calculations ignored the OTHER relation. Our official scoring metric is macro-averaged F1-Score for (9+1)-way classification, considering directionality and archives F1-score 87.06%.

The results of relation extraction using Att-BLSTM with word embedding and without word embedding, and with directionality and without directionality are evaluated using evaluation metrics precision (P), recall (R), and F1 in percentage (%) as provided below in Figure 5.9

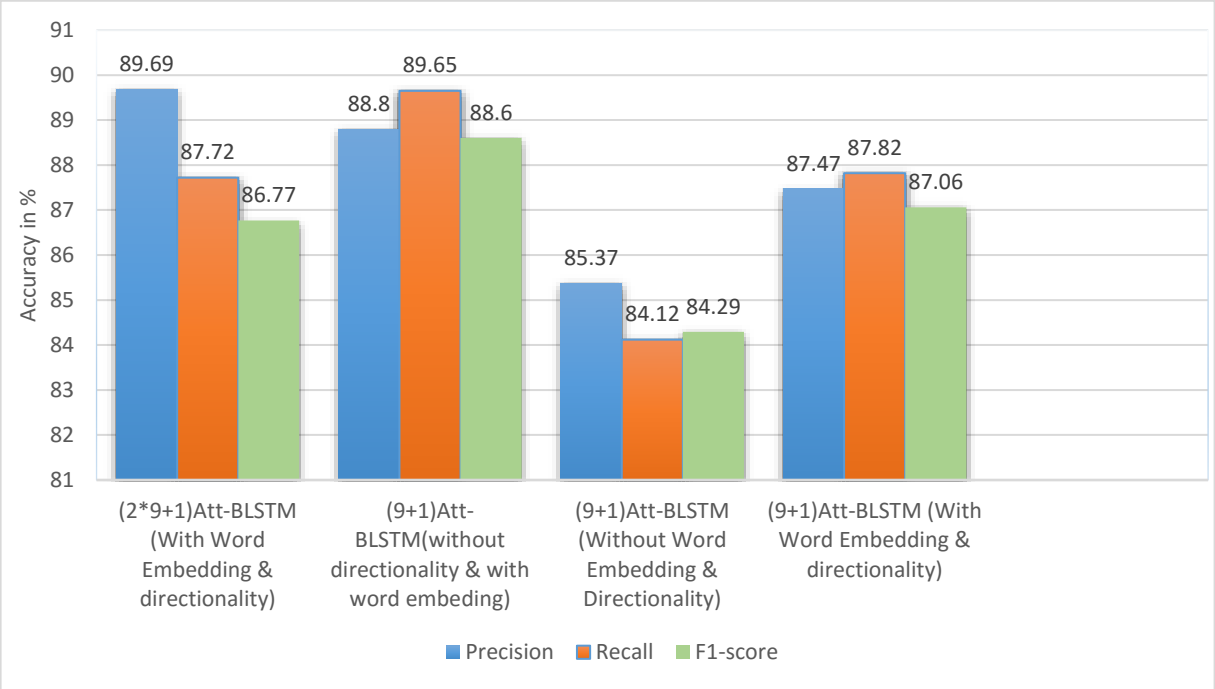

Figure 5. 9 Evaluation result difference in all variations

As shown in above Figure 5.9, the main difference in the first experiment and the last experiment is the way macro-average calculates the result. When the number of class varies the result also varies in macro-average score.

5.7 Discussion

In this study, we target an independent RE system for Amharic text that both avoids complicated feature engineering and minimizes the reliance on the supervised NLP modules and lexical resources for feature generation. It shows that BLSTM has a better representation

of sentence level relationships. Due to this, the proposed study potentially alleviates the error propagation and advances the performance only with word embedding as a feature set. To be concrete, the proposed Amharic semantic relation extraction is provided only with raw sentences marked with the positions of the two entities of interest. To fully analyze the effectiveness of the proposed model, we conduct several experiments based on different variations.

Figure 5.9 summarized various experimental results as follows. One of the experiment variations is relation extraction without directionality and with directionality. As Results indicate, relation extraction without directionality is increased by 1.54% compared to relation extraction with directionality. This shows that identifying the direction of the relation in a sentence is difficult after the model predicts the type of relation. Directionality matters the order of context words. We observe that the order of context words is significant in assigning the relation label to each sentence. For example, Cause-Effect(e1,e2) means that e1 is the Cause and e2 is the Effect, whereas Cause-Effect(e2,e1) means that e2 is the Cause and e1 is the Effect.

The other experiment variation is relation extraction without word embedding and with word embedding. As shown in Figure 5.9, the proposed approach with word embedding improves F-score by 2.77% as compared to without word embedding. It shows that word embedding can substitute manually designed features for relation extraction. In other words, understanding the semantics of the word is very critical for relation extraction.

From the results, we observe that using the position indicators can help the model pinpoint the location of the two marked nominals.

Example: <e1>ህመሙ</e1> የተከሰተው በተበከለ ውሃ በሚታጠብበት ወይም በሚጠጣበት ጊዜ ወደ ቆዳ በሚገባ <e2>ጥገኛ</e2> ምክንያት ነው።

In this example where two nominals <e1>ህመሙ</e1> and <e2>ጥገኛ</e2> are relatively far apart from each other, the model can still assign a relatively high salience score for these two nominals. By doing so, the model improves its prediction with the special position indicators <e1>, </e1>, <e2> and </e2> as compared to a model that does not know the exact nominal locations. Besides, we can see that the model can automatically learn which words are more important under different relation types by applying attention mechanisms. This Amharic

sentence belongs to relation type Cause-Effect(e2,e1), and the word ምክንያት is with the highest salience scores apart from target nominals. This word indicates the relationship between the two nominals is Cause-Effect(e2,e1) even without the help of other words. Generally, all these demonstrate that the proposed position indicator and attention mechanisms are critical for relation extraction.

During this study, we have tried to analyze the performance of individual relations. We observe from individual relations, that there are stable patterns across all experiments. The best relation (presumably the easiest to classify) is Cause-Effect, which archives above 91% for almost all experiments with comparatively small differences between the experiments. On the other hand, the hardest relation is generally Member-Collection, which archives a maximum of 76% and a minimum of 63% in all experiments. Some sentences of these relations are misclassified, most frequently as Component-Whole and Other. We observed that the two types of relations (Member-Collection and Component-Whole) are strongly related in sentence structure, which confuses the model to predict.

In all variation experiments as shown in all confusion matrices figures, the number of skipped values is zero. This shows that the proposed model can reduce the influence of data noise on the model training, and enhance entity discrimination feature with position attention mechanism so that the entity information can be combined effectively in the relation extraction. Finally, with all these, the official score is (9+1)-way evaluation with directionality and word embedding and macro-averaged F1-score 87.06%.

Chapter 6: Conclusion, Recommendation, and Future Work

6.1 Conclusion

Relation extraction is very important in NLP and can be beneficial for semantic search, machine translation, question answering, knowledge harvesting, paraphrasing, building thesauri, etc. In this paper, we propose a deep learning model, named Att-BLSTM, for Amharic semantic relation extraction. This model does not rely on NLP tools or lexical resources to get high features; it uses raw text with position indicators as input. This model contains different components such as the pre-processing, word embedding, feature extractor, model builder, and relation extractor. BLSTM is applied for load forecasting which cannot only consider the past information, but also consider the future information, and make the prediction result accurate. BLSTM networks are easier to learn long-term dependencies than the simple loop architecture. Attention-based Bi-directional LSTM networks (Att-BLSTM) are intended to capture the important semantic information at any position in the sentence. This study introduces an attention mechanism, which can automatically focus on the words that have a decisive effect on relation extraction, to capture the most important semantic information in a sentence.

The effectiveness of Att-BLSTM is demonstrated by evaluating the model on the Amharic-RE-Dataset (excluding "Other") with many variations such as with directionality, without directionality, without word embedding, and with word embedding. Relation extraction with word embedding achieves better performance at learning features along with the attention mechanism, compared with without word embedding. The result also investigated that without directionality achieves a better performance than with directionality. The dropout regularization strategy is applied to avoid overfitting in the process of model construction. Finally, our features do not require expensive linguistic analysis for sentence pre-processing. It only requires cleaning, stop-word removal, and tokenization. The features we devised mainly focus on how to represent the existence of entities and relations in sentences, most of them being easy to compute. This makes our model relatively easy and efficient. The proposed model (Att-BLSTM) archives an F1-score of 87.06% for Amharic semantic relation extraction on Amharic-RE-Dataset.

6.2 Contribution of this Work

The main contributions of this work are summarized as follows:

- Amharic semantic relation extraction dataset is prepared with the help of language experts.
- We generate word embedding for Amharic text to use as an input in the model.
- We introduce the feasibility of using BLSTM with an attention mechanism for Amharic semantic relation extraction.
- We study the influence of adding word embedding on the final extraction performance.
- This study shows the influence of adding directionality to the final extraction performance.
- We explore the feasibility of performing relation extraction in Amharic text without using extra knowledge and NLP systems.

6.3 Recommendation and Future Work

We recommend proposing question answering, machine translation, knowledge base population, and other Amharic NLP applications based on relations between entities extracted from the presented model. Because one of the reasons for developing relation extraction is to be used as an input for other NLP applications. Since deep learning methods require a huge amount of training data, the performance of the system can still be improved by increasing the training data size. A small amount of resources like training data for RE highly affects the accuracy of the system. Extending the scope of relation extraction between a pair of entities across sentences with a pre-defined relation set with a large number of relations can be done in future work. Testing other models may be a supplement of our work and can be done in future work.

References

[1] D. Jurafsky and J. H. Martin, *Speech and Language Processing:An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, Third Edit. 2018.

[2] X. Yao and B. Van Durme, "Information Extraction over Structured Data : Question Answering with Freebase," in *Proceedings of the 52nd Annual Meeting ofthe Association for Computational Linguistics*, 2014, pp. 956–966.

[3] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. May, 2015, doi: 10.1038/nature14539.

[4] M. M. Lopez and J. Kalita, "Deep Learning applied to NLP," *cs.CL*, vol. 1, 2017.

[5] C. A. Deepa, P. C. Reghuraj, and A. Ramanujan, "RELATION EXTRACTION USING DEEP LEARNING METHODS - A SURVEY," *ICTACT*, vol. 09, no. SOFT COMPUTING, pp. 1893–1902, 2019, doi: 10.21917/ijsc.2019.0263.

[6] Z. Alom *et al.*, "A State-of-the-Art Survey on Deep Learning Theory and Architectures," *electronics*, pp. 1–67, 2019, doi: 10.3390/electronics8030292.

[7] S. Zhang, D. Zheng, X. Hu, and M. Yang, "Bidirectional Long Short-Term Memory Networks for Relation Classification," in *29th Pacific Asia Conference on Language, Information and Computation pages*, 2015, pp. 73–78.

[8] M. Wu, L. Liu, W. Yao, C. Yin, and J. Wang, "Semantic Relation Classification by Bi-directional LSTM Architecture," *Adv. Sci. Technol. Lett. ASTL*, vol. 143, no. Ast, pp. 205–210, 2017.

[9] P. Zhou, W. Shi, J. Tian, Z. Qi, B. Li, and H. Hao, "Attention-based bidirectional long short-term memory networks for relation classification," in *Proceedings of the 54th Annual Meeting ofthe Association for Computational Linguistics*, 2016, no. September, pp. 207–212, doi: 10.18653/v1/p16-2034.

[10] J. Lee, S. Seo., and Y. S. Choi, "Semantic Relation Classification via Bidirectional LSTM Networks with Entity-Aware Attention Using Latent Entity Typing," *Symmetry (Basel).*, pp. 1–12, 2019, doi: 10.3390/sym11060785.

[11] T. H. Nguyen and R. Grishman, "Relation Extraction : Perspective from Convolutional Neural Networks," *Proc. NAACL-HLT 2015*, pp. 39–48, 2015.

[12] X. Guo, H. U. I. Zhang, H. Yang, L. Xu, and Z. Ye, "A Single Attention-Based

Combination of CNN and RNN for Relation Classification," *IEEE Access*, vol. 7, pp. 12467–12475, 2019, doi: 10.1109/ACCESS.2019.2891770.

[13] J. Wen, "Recurrent Convolutional Neural Network for Relation Classification," *Assoc. Adv. Artif. Intell.*, 2017.

[14] X. Han, Y. Zhang, W. Zhang, and T. Huang, "An Attention-Based Model Using Character Composition of Entities in Chinese Relation Extraction," *Information*, pp. 1–17, 2020, doi: 10.3390/info11020079.

[15] L. Ouyang, H. Tang, and G. Xiao, "Chinese Text Relation Extraction with Multi-instance Multi-label BLSTM Neural Networks," 2019. doi: 10.18293/SEKE2019-106.

[16] A. A. Pareed and S. M. Idicula, "A Relation Extraction System for Indian Languages," *ASTES*, vol. 4, no. Recent Advances in Engineering Systems, pp. 65–69, 2019.

[17] D. Thenmozhi, B. S. Kumar, and C. Aravindan, "Deep Learning Approach to Named Entity Recognition and Relation Extraction for Conversational Systems in Indian Languages," no. 2266, pp. 1–15, 2018.

[18] M. Abate and Y. Assabie, "Development of Amharic Morphological Analyzer Using Memory-Based Learning," *Proc. 9th Int. Conf. Nat. Lang. Process.*, pp. 1–13, 2014.

[19] B. Worku, "Information Extraction from Amharic language Text : Knowledge-poor Approach," ADDIS ABABA UNIVERSITY, 2015.

[20] R. C. Bunescu and R. J. Mooney, "A Shortest Path Dependency Kernel for Relation Extraction," in *Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, 2005, no. October.

[21] M. Mintz, S. Bills, R. Snow, and D. Jurafsky, "Distant supervision for relation extraction without labeled data," in *Proceedings of the 47th Annual Meeting ofthe ACL and the 4th IJCNLP ofthe AFNLP*, 2009, pp. 1003–1011.

[22] B. Rink and S. Harabagiu, "UTD : Classifying Semantic Relations by Combining Lexical and Semantic Resources," in *Proceedings of the 5th International Workshop on Semantic Evaluation, ACL 2010*, 2010, no. July, pp. 256–259.

[23] D. Zeng, K. Liu, S. Lai, G. Zhou, and J. Zhao, "Relation Classification via Convolutional Deep Neural Network," in *the 25th International Conference on Computational Linguistics:*, 2014, no. 2011, pp. 2335–2344.

[24] I. Hendrickx, S. N. Kim, Z. Kozareva, P. Nakov, L. Romano, and S. Szpakowicz,

"SemEval-2010 Task 8 : Multi-Way Classification of Semantic Relations Between Pairs of Nominals," *Proc. 5th Int. Work. Semant. Eval.*, no. July, pp. 33–38, 2010.

[25] F. Wu and D. S. Weld, "Open Information Extraction using Wikipedia," *118 Proc. ofthe 48th Annu. Meet. ofthe Assoc. Comput. Linguist.*, no. July, pp. 118–127, 2010.

[26] C. Projection, E. Riloff, C. Schafer, and D. Yarowsky, "Inducing Information Extraction Systems for New Languages via Cross-Language Projection."

[27] M.-F. Moens, *Information Extraction: Algorithms and Prospects in a Retrieval Context*. Springer, 2006.

[28] N. Chinchor and E. Marsh, "MUC-7 Information Extraction Task Definition," no. July, pp. 359–367, 1998.

[29] N. Konstantinova, "Review of Relation Extraction Methods : What is New Out There ?," *Univ. Wolverhampton,UK*.

[30] D. Zelenko and A. Richardella, "Kernel Methods for Relation Extraction," *J. ofMachine Learn. Res.*, vol. 3, pp. 1083–1106, 2003.

[31] M. Miwa and M. Bansal, "End-to-End Relation Extraction using LSTMs on Sequences and Tree Structures," 2015.

[32] S. K. Tomas Mikolov, Martin Karafiat, Lukas Burget1, Jan Honz Cernocky, "ˇRecurrent neural network based language model," *INTERSPEECH*, no. September, pp. 1045–1048, 2010.

[33] Y. Liu, F. Wei, S. Li, H. Ji, M. Zhou, and H. Wang, "A Dependency-Based Neural Network for Relation Classification," *cs.CL*, 2015.

[34] A. Graves, M. Liwicki, S. Fern, R. Bertolami, and H. Bunke, "A Novel Connectionist System for Unconstrained Handwriting Recognition," pp. 1–14, 2008.

[35] A. Graves, A. Mohamed, and G. Hinton, "SPEECH RECOGNITION WITH DEEP RECURRENT NEURAL NETWORKS," *cs.NE*, no. 3, 2013.

[36] Y. Xu, L. Mou, G. Li, and Y. Chen, "Classifying Relations via Long Short Term Memory Networks along Shortest Dependency Paths," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, no. September, pp. 1785–1794.

[37] S. Hochreiter and J. Schmidhuber, "LONG SHORT-TERM MEMORY," *Neural Comput.*, vol. 9, no. 8, pp. 1–32, 1997.

[38] A. Graves, “Generating Sequences With Recurrent Neural Networks,” *cs.CL*, pp. 1–43, 2914.

[39] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Distributed Representations of Words and Phrases and their Compositionality,” *cs.CL*, vol. 1, pp. 1–9, 2013.

[40] J. Pennington, R. Socher, and C. D. Manning, “GloVe : Global Vectors for Word Representation,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543.

[41] T. Mikolov, G. Corrado, K. Chen, and J. Dean, “Efficient Estimation of Word Representations in Vector Space,” *cs.CL*, pp. 1–12, 2013.

[42] V. Van Asch, “Macro-and micro-averaged evaluation measures [[BASIC DRAFT]].,” *Belgium: CLiPS*, pp. 1–27, 2013, [Online]. Available: https://pdfs.semanticscholar.org/1d10/6a2730801b6210a67f7622e4d192bb309303.pdf.

[43] J. Opitz and S. Burst, “Macro F1 and Macro F1,” *cs.LG*, no. 2, pp. 1–12, 2019, [Online]. Available: https://pdfs.semanticscholar.org/1d10/6a2730801b6210a67f7622e4d192bb309303.pdf.

[44] A. K. Santra and C. J. Christy, “Genetic Algorithm and Confusion Matrix for Document Clustering 1,” *IJCSI Int. J. Comput. Sci. Issues*, vol. 9, no. 1, pp. 322–328, 2012.

[45] “About Amharic Language,” p. 2020, 2020.

[46] Z. Leyew, “The Ethiopian Language Policy: A Historical and Typological Overview 1 Zelealem Leyew,” *Ethiop. J. Lang. Lit.*, vol. XII, no. 2, pp. 1–59, 2012.

[47] B. Yimam, *የአማርኛ ሰዋሰው (Amharic Grammar)*, 2nd ed. Addis Aba: Birhanina Selam Printing Press., 2000.

[48] T. Bloor, “The Ethiopic Writing System:,” *J. Simpl. Spell. Soc.*, pp. 30–36, 1995.

[49] A. Temesgen, “Development of Amharic Grammar Checker Using Morphological Features of Words and N-Gram Based Probabilistic Methods,” *13th Int. Conf. Parsing Technol.*, pp. 106–112, 2013.

[50] S. S. Zhang, Kaifang, Huayou SuYong Dou, “Evaluation of the Influences of Hyper-Parameters and L 2 -norm Regularization on ANN Model for MNIST Recognition,” *Int. Conf. Intell. Comput. Autom. Syst.*, pp. 379–386, 2019, doi:

10.1109/ICICAS48597.2019.00086.

[51] E. Phaisangittisagul, “An Analysis of the Regularization between L 2 and Dropout in Single Hidden Layer Neural Network,” *2016 7th Int. Conf. Intell. Syst. Model. Simul.*, 2016, doi: 10.1109/ISMS.2016.14.

[52] A. Aghaebrahimian and M. Cieliebak, “Hyperparameter Tuning for Deep Learning in Natural Language Processing,” 2019.

[53] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, “Improving neural networks by preventing co-adaptation of feature detectors,” *cs . NE*, vol. 1, pp. 1–18, 2012.

Annex A: The Sample Amharic-RE-Dataset

```
1   "<e1>በፖርኩ</e1> ውስጥ አንበሳ ፣ ነብር ፣ ቆርኬ ፣ ደኩላ ፣ ዝሆን ፣ ጉሬዛ ፣ ቀይ ቶጣና ሌሎችም <e2>እንስሳት</e2> እንደሚገኙ ከቢሰሙያው ገለጻ ለመረዳት ተችሏል።"
Content-Container(e2,e1)
Comment:

2   "ከእገልግሎት ውጪ የነበረ <e1>ሰቪየት ሰራሽ</e1> ጸረ-ኦውሮፕላን <e2>ሚሰይል</e2> በሰማለያ ሞቃጺኝ ከተጣ በማረገዳቱ 14 ሰዎች መሞታቸውና በርካታዎቹ መቁሰላቸው ተገለጸ።"
Product-Producer(e2,e1)
Comment:

3   "<e1>ዮተማረ</e1> <e2>ማህበሩ</e2> በነው ዮርክ ስቱት ዩኒቨርሲቲ በበሩሉ ውስጥ ያለ የቅድመ ምረቃ ተማረ ድምጽ ነው።"
Member-Collection(e1,e2)
Comment:

4   "<e1>ሰዎች</e1> ወደ <e2>መሄ ከተማ</e2> በመሄድ ላይ ናቸው።"
Entity-Destination(e1,e2)
Comment:

5   "የሽፉማት ኮሚቴው የሽፉማቱን <e1>ምክንያቶች</e1> የሚያብራራ ሀሣዊ <e2>ሰነድ</e2> ይሰጣል።"
Message-Topic(e2,e1)
Comment:

6   "<e1>ፍንዳታ</e1> የተከሰተው በውኃ መጶሻ <e2>ሣሪት</e2> ምክንያት ነው።"
Cause-Effect(e2,e1)
Comment:

7   "የንጣይ <e1>ኔትወርኮችም</e1> እንኳ ወደ ከፍተኛ ጥራት ወዳለው <e2>አሰፋይ</e2> ተዛውረዋል።"
Instrument-Agency(e2,e1)
```

Annex B: Sample Relation Prediction Output

```
0	Message-Topic(e1,e2)
1	Product-Producer(e2,e1)
2	Instrument-Agency(e2,e1)
3	Entity-Destination(e1,e2)
4	Cause-Effect(e2,e1)
5	Component-Whole(e2,e1)
6	Product-Producer(e2,e1)
7	Member-Collection(e1,e2)
8	Component-Whole(e2,e1)
9	Message-Topic(e1,e2)
10	Entity-Destination(e1,e2)
11	Other
12	Entity-Destination(e1,e2)
13	Product-Producer(e1,e2)
14	Entity-Origin(e1,e2)
15	Entity-Origin(e1,e2)
16	Member-Collection(e2,e1)
17	Member-Collection(e1,e2)
18	Entity-Origin(e1,e2)
19	Message-Topic(e1,e2)
20	Content-Container(e1,e2)
21	Product-Producer(e1,e2)
22	Other
23	Entity-Origin(e2,e1)
24	Product-Producer(e1,e2)
25	Cause-Effect(e2,e1)
```

Declaration

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

Declared by:

Name: Aschenaki Abi Abera _____

Signature: _____

Date: _____

Confirmed by advisor:

Name: Yaregal Assabie (PhD_____

Signature: _____

Date: _____