Information Extraction from Amharic News Text by Integrating Reinforcement Learning

Natnael Girma Megersa

A Thesis Submitted to

The department of Computer science and Engineering

School of Electrical Engineering and Computing

Presented in Partial Fulfillment of the Requirement for the Degree of Master's in Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

September 2022

Adama, Ethiopia

Information Extraction from Amharic News Text by Integrating Reinforcement Learning

Natnael Girma Megersa

Advisor: Mohd Wazih Ahmed
(Assistant Professor)

A Thesis Submitted to
The department of Computer Science and Engineering
School of Electrical Engineering and Computing

Presented in Partial Fulfillment of the Requirement for the Degree of
Masters in Computer Science and Engineering

Office of Graduate Studies
Adama Science and Technology University

September 2022
Adama, Ethiopia

Approval Page of M.Sc. Thesis

I/we, the advisors of the thesis entitled “Information Extraction from Amharic news text by integrating Reinforcement Learning” and developed by Natnael Girma, hereby certify that the recommendation and suggestions made by the board of examiners are appropriately incorporated into the final version of the thesis.

Major Advisor	Signature	Date

We, the undersigned, members of the Board of Examiners of the thesis Natnael Girma have read and evaluated the thesis entitled “Information Extraction from Amharic news text by integrating Reinforcement Learning” and examined the candidate during open defense. This is, therefore, to certify that the thesis is accepted for partial fulfillment of the requirement of the degree of Master of Science in Computer Science and Engineering.

Chairperson	Signature	Date
Internal Examiner	Signature	Date
External Examiner	Signature	Date

Finally, approval and acceptance of the thesis is contingent upon submission of its final copy to the Office of Postgraduate Studies (OPGS) through the Department Graduate Council (DGC) and School Graduate Committee (SGC).

Department Head	Signature	Date
School Dean	Signature	Date
Office of Postgraduate Studies, Dean	Signature	Date

Declaration

I hereby declare that this Master Thesis entitled “Information Extraction from Amharic news text by integrating Reinforcement Learning” is my original work. That is, it has not been submitted for the award of any academic degree, diploma or certificate in any other university. All sources of materials that are used for this thesis have been duly acknowledged through citation.

_____	_____	_____
Name of the student	Signature	Date

Recommendation

I/we, the advisor(s) of this thesis, hereby certify that I/we have read the revised version of the thesis "Information Extraction from Amharic news text by integrating Reinforcement Learning" prepared under my/our guidance by Natnael Girma submitted in partial fulfillment of the requirements for the degree of Mater's of Science in Computer Science and Engineering. Therefore, I/we recommend the submission of revised version of the thesis to the department following the applicable procedures.

_____	_____	_____
Major Advisor	Signature	Date

ACKNOWLEDGEMENT

God deserves all the credit for giving me the strength to complete this research project from beginning to completion.

I want to extend my sincere gratitude to my adviser, Mohd Wazih Ahmed (Ph.D.), for his perseverance, zeal, cooperation, guidance, unwavering support, and efforts to see this task through to completion. I am particularly appreciative of the independence he granted me at this time. This thesis cannot be accomplished without his careful guidance, priceless trust, and helpful criticism.

Additionally, I want to express my sincere gratitude to my fellow MSc students and members of the Data Science Special Interest Group (SIG). I want to express my gratitude, in particular, to my friends Mr. Abraham Wolde, Mr. Abera Geda, Mr. Birhanu Gebisa, and Mr. Yohannes Gutema, who all played significant roles in the peer assessment of my thesis work.

Last but not least, I want to express my profound gratitude to my family and friends for their support.

Dedication

I wholeheartedly want to dedicate this thesis to my dear Mother, who served as my inspiration and the driving force behind all of my academic achievements but who I lost during the master's thesis journey. Mother, Rest in Peace.

CONTENTS

LIST OF TABLES

TABLE **PAGE**

LIST OF FIGURES

FIGURE **PAGE**

List of Acronyms and Abbreviations

IE	Information Extraction
RL	Reinforcement learning
NLP	Natural Language Processing
IR	Information Retrieval
MDP	Markov Decision Process
POS	Part Of Speech
NER	Named Entity Recognition
OIE	Open Information Extraction
AOIE	Amharic Open Information Extraction
HMM	Hidden Markov Model
SVM	Support vector machine
MT	Machine translation
ATIE	Amharic Text Information Extraction
RLIE-DQN	Reinforcement Learning Information Extraction - Deep Q Network
RLIE-A3C	Reinforcement Learning Information Extraction-Asynchronous Advantage Actor-Critic
EBC	Ethiopian Broadcasting Corporate
FBC	Fana broadcasting corporate
AI	Artificial intelligent
Amseg	Amharic segmenter
HTML	Hyper Text Markup language
MUC	Message Understanding Conferences
XML	Extensible Markup language
UML	Unified modeling language
API	Application programming interface
MSE	Mean Square Error
MAE	Mean Absolute Error
AIERL	Amharic information extraction with reinforcement learning
ReLU	Rectified linear unit

Abstract

Information extraction is a technology that enables relevant contents to be extracted from textual information available electronically. Information Extraction techniques, in the opinion of many researchers, require access to a large collection of documents. Preparing a large collection of documents and well-annotated datasets in low-resource languages like Amharic, however, has become a challenging task due to the scarcity of documents. Since many researchers working in this area didn't consider the fact that information is relative based on the source the developed IE systems are source-dependent. The objective of this research is to suggest a model that extracts information from Amharic news text where the amount of training data is an ounce and, in a case, where more related information is needed. The model is going to work by integrating Reinforcement learning techniques to gain additional documents with related information. The work in this research has two phases with different components and subcomponents inside them. Phase one is a rule-based extraction system and the other phase consists of a reinforcement learning agent that is responsible for making decisions based on the environmental state. The research work comprises extraction from a given news text and extraction from new sources or documents, which are recurring until satisfactory information is collected and this is addressed by the DQN algorithm. We use a deep Q network that has been tuned to maximize a reward function that measures extraction precision. Where our model learns to choose the best course of action based on the information in the document. The DQN agent is trained for 50,000 steps using 3 different policies to compare a better reward result. We performed individual experimental evaluations for each phase and component separately the experiment result performed on a total of 1590 news stories with 65810 sentences with 225269 words shows that our solution gives a promising result and has a combination of 93.5 % accuracy score.

Keywords: *information extraction, Reinforcement learning, environmental state, deep Q network (DQN).*

CHAPTER ONE

1. INTRODUCTION

1.1. Background of the study

Quick growths in Information Technology are making available an enormous amount of data and information. A considerable amount of this data is in electronic formats as unstructured and semi-structured forms. The number of Amharic documents on the web is also increasing from time to

time due to the current era of technology. What makes information extraction hard is the fact that data is initially unstructured and is labeled using human-understandable languages, which makes the data limited in the degree to which it is machine-understandable (Hirpassa, 2017) (Alansary & Magdy Nagi, 2015).

The automatic IE system process takes documents as input and generates the respective structured information as a result and this generated information can be utilized by other users in different applications for further analysis purposes (Hobbs, 2004) (Elsadig, Ali Ahmed, & Mubatak Himmat, 2015) (Aggarwal, 2012).

Amharic is an Ethiopian Semitic language, which is a subgroup within the Semitic branch of the Afro-Asiatic. It is spoken as a first language by the Amhara, and also serves as a lingua franca for other populations residing in major cities and towns of Ethiopia. The Amharic language serves as the current working language of Ethiopia alongside other languages and is also the working language of several of the Ethiopian regional and federal states (Ephrem Tadesse, Rosa Tsegaye, & Kuulaa Qaqqabaa, 2020).

Reinforcement learning (RL) is a part of machine learning concerned with how intelligent agents have to take action in an environment to get a reward. Reinforcement learning is one of three basic machine learning paradigms (Richard S. Sutton & Andrew G. Barto, 2014) (Takanobu, Tianyang Zhang, Jiexi Liu, & Minlie Huang, 2018).

Due to the diverse nature of data contained within digital data sources, searching and finding appropriate information is becoming extremely challenging and it remains an open research problem.

1.2. The motivation for the study

There is a wealth of Amharic content available online that covers every conceivable subject and is written and published every day. The necessity for automatic support for the extraction of pertinent information from these various unstructured texts has grown urgent due to the dramatically expanding size of these digital data sets of varied text types, such as news messages, articles, and web pages. The fact that information is relative based on the information provider source and availability of different news sources highly motivates us to do this study.

1.3. Statement of the problem

The information extraction task requires a vast number of documents and a large annotated dataset and also the text document in which the information extraction task is performed has to be well written and full of information needed to extract (Narasimhan, Adam Yala, & Regina Barzilay, 2016).

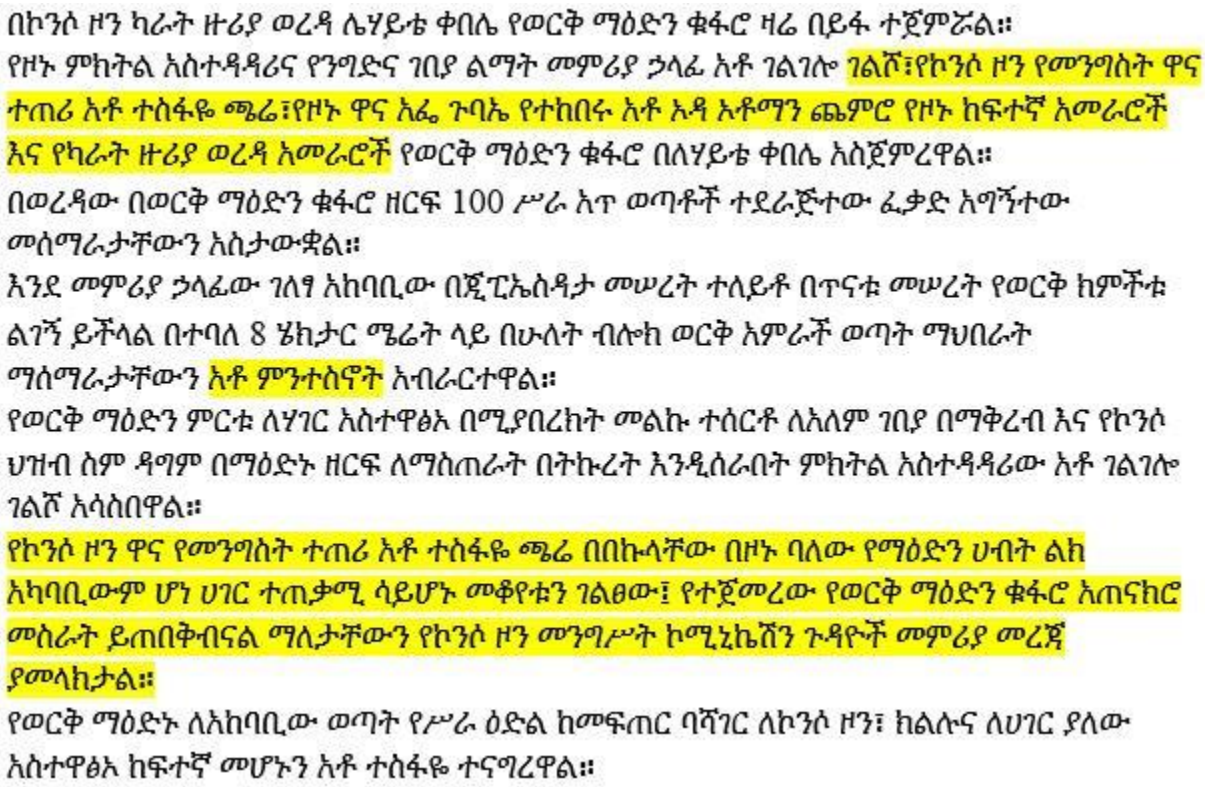
For instance, let's see an example:

በኮንሶ ዞን ካራት ዙሪያ ወረዳ ሌሃይቴ ቀበሌ የወርቅ ማዕድን ቁፋሮ ሥራ በዛሬው ዕለት በይፋ ተጀምሯል።
የቁፋሮ ሥራው የዞኑ ምክትል አስተዳዳሪና የንግድና ገበያ ልማት መምሪያ ኃላፊ አቶ ገልገሎ ገልሾን ጨምሮ የዞኑ የስራ ሀላፊዎች በተገኙበት ተጀምሯል።
በወረዳው በወርቅ ማዕድን ቁፋሮ ዘርፍ 100 ሥራ አጥ ወጣቶች በመደራጀት በተሰጣቸው ፈቃድ መሰረት ሥራ መጀመራቸው ተመላክቷል።
አካባቢው በጂፒኤስ ዳታ ተለይቶ በተጠናው መሰረት የወርቅ ክምችቱ ሊገኝ ይችላል በተባለበት 8 ሄክታር መሬት ላይ ሥራ መጀመሩ ነው የተገለጸው።
የወርቅ ማዕድኑ ለአካባቢው ወጣት የሥራ ዕድል ከመፍጠር ባሻገር ለዞኑ፣ ለክልሉ እና ለሀገር ያለው አስተዋፅኦ ከፍተኛ እንደሚሆን መገለፁን ከኮንሶ ዞን ኮሙኒኬሽን ያገኘነው መረጃ ያመላክታል።

FIGURE 1 - statement of the problem news one

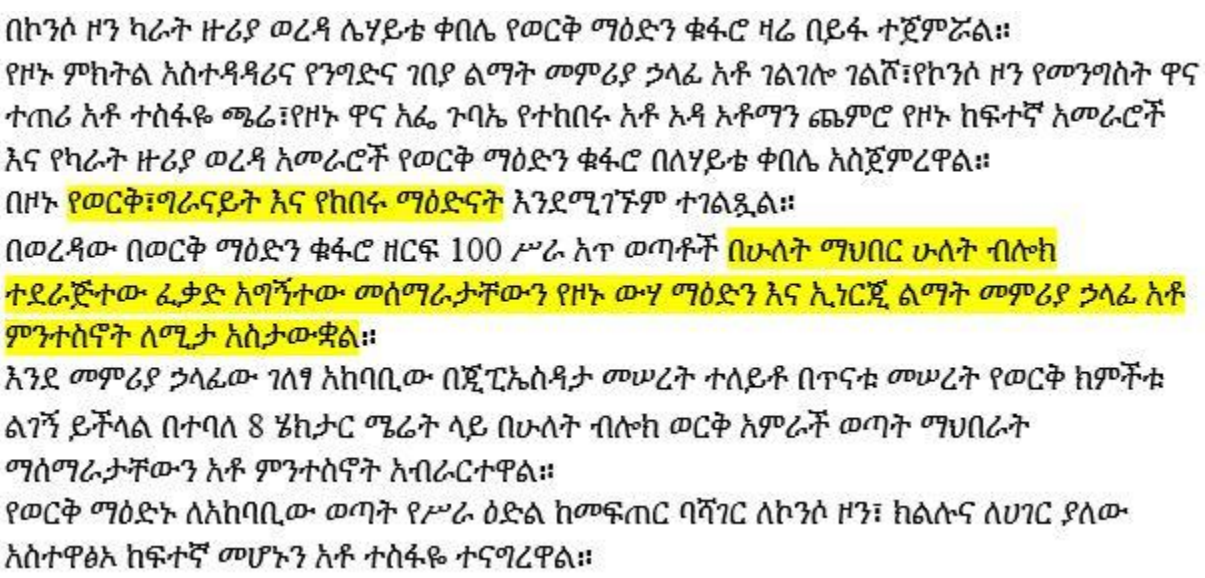
From the above figure of Amharic news text (Figure 1) an extraction system that is trained in the Amharic language can extract the specific type of information which is found only in the given text but this research work aims to find additional information that is not available in the given news text document by extracting from other sources but related with the given text news. The

below two figures of news text documents (figure 2 and figure 3) are from another source but they each contain additional information which is not available in the given news text.



በኮንሶ ዞን ካራት ዙሪያ ወረዳ ሌሃይቴ ቀበሌ የወርቅ ማዕድን ቁፋሮ ዛሬ በይፋ ተጀምሯል።
የዞኑ ምክትል አስተዳዳሪና የንግድና ገበያ ልማት መምሪያ ኃላፊ አቶ ገልገሎ ገልሾ፣የኮንሶ ዞን የመንግስት ዋና ተጠሪ አቶ ተስፋዬ ጫሬ፣የዞኑ ዋና አፈ ጉባኤ የተከበሩ አቶ አዳ አቶማን ጨምሮ የዞኑ ከፍተኛ አመራሮች እና የካራት ዙሪያ ወረዳ አመራሮች የወርቅ ማዕድን ቁፋሮ በለሃይቴ ቀበሌ አስጀምረዋል።
በወረዳው በወርቅ ማዕድን ቁፋሮ ዘርፍ 100 ሥራ አጥ ወጣቶች ተደራጅተው ፈቃድ አግኝተው መሰማራታቸውን አስታውቋል።
እንደ መምሪያ ኃላፊው ገለፃ አከባቢው በጂፒኤስዳታ መሠረት ተለይቶ በጥናቱ መሠረት የወርቅ ክምችቱ ልገኝ ይችላል በተባለ 8 ሄክታር ሜሬት ላይ በሁለት ብሎክ ወርቅ አምራች ወጣት ማህበራት ማሰማራታቸውን አቶ ምንተስኖት አብራርተዋል።
የወርቅ ማዕድን ምርቱ ለሃገር አስተዋፅኦ በሚያበረክት መልኩ ተሰርቶ ለአለም ገበያ በማቅረብ እና የኮንሶ ህዝብ ስም ዳግም በማዕድኑ ዘርፍ ለማስጠራት በትኩረት እንዲሰራበት ምክትል አስተዳዳሪው አቶ ገልገሎ ገልሾ አሳስበዋል።
የኮንሶ ዞን ዋና የመንግስት ተጠሪ አቶ ተስፋዬ ጫሬ በበኩላቸው በዞኑ ባለው የማዕድን ሀብት ልክ አካባቢውም ሆነ ሀገር ተጠቃሚ ሳይሆኑ መቆየቱን ገልፀው፤ የተጀመረው የወርቅ ማዕድን ቁፋሮ አጠናክሮ መስራት ይጠበቅብናል ማለታቸውን የኮንሶ ዞን መንግሥት ኮሚኒኬሽን ጉዳዮች መምሪያ መረጃ ያመላክታል።
የወርቅ ማዕድኑ ለአከባቢው ወጣት የሥራ ዕድል ከመፍጠር ባሻገር ለኮንሶ ዞን፣ ክልሉና ለሀገር ያለው አስተዋፅኦ ከፍተኛ መሆኑን አቶ ተስፋዬ ተናግረዋል።

FIGURE 2 - statement of the problem news two



በኮንሶ ዞን ካራት ዙሪያ ወረዳ ሌሃይቴ ቀበሌ የወርቅ ማዕድን ቁፋሮ ዛሬ በይፋ ተጀምሯል።
የዞኑ ምክትል አስተዳዳሪና የንግድና ገበያ ልማት መምሪያ ኃላፊ አቶ ገልገሎ ገልሾ፣የኮንሶ ዞን የመንግስት ዋና ተጠሪ አቶ ተስፋዬ ጫሬ፣የዞኑ ዋና አፈ ጉባኤ የተከበሩ አቶ አዳ አቶማን ጨምሮ የዞኑ ከፍተኛ አመራሮች እና የካራት ዙሪያ ወረዳ አመራሮች የወርቅ ማዕድን ቁፋሮ በለሃይቴ ቀበሌ አስጀምረዋል።
በዞኑ የወርቅ፣ግራናይት እና የከበሩ ማዕድናት እንደሚገኙም ተገልጿል።
በወረዳው በወርቅ ማዕድን ቁፋሮ ዘርፍ 100 ሥራ አጥ ወጣቶች በሁለት ማህበር ሁለት ብሎክ ተደራጅተው ፈቃድ አግኝተው መሰማራታቸውን የዞኑ ውሃ ማዕድን እና ኢነርጂ ልማት መምሪያ ኃላፊ አቶ ምንተስኖት ለሚታ አስታውቋል።
እንደ መምሪያ ኃላፊው ገለፃ አከባቢው በጂፒኤስዳታ መሠረት ተለይቶ በጥናቱ መሠረት የወርቅ ክምችቱ ልገኝ ይችላል በተባለ 8 ሄክታር ሜሬት ላይ በሁለት ብሎክ ወርቅ አምራች ወጣት ማህበራት ማሰማራታቸውን አቶ ምንተስኖት አብራርተዋል።
የወርቅ ማዕድኑ ለአከባቢው ወጣት የሥራ ዕድል ከመፍጠር ባሻገር ለኮንሶ ዞን፣ ክልሉና ለሀገር ያለው አስተዋፅኦ ከፍተኛ መሆኑን አቶ ተስፋዬ ተናግረዋል።

FIGURE 3 - statement of the problem news three

From the above two text news which has been found from two different news sources (Figure 2 and Figure 3), the highlighted text is additional information related to the first given news text

that is not available in news text one but is useful information that needs to be extracted as a piece of information but since the developed IE systems are source dependent, we can't get that information.

Also, an information extraction task is a language and domain-dependent. Available extraction tools models are not compatible with Amharic language news text documents and it needs linguistic knowledge of the Amharic language to propose an accurate extraction model (Gero, Getu Girma, & Shimels Hailu , 2021).

1.4. Research Questions

This research work addresses the problem by answering the following questions: -

- How to extract additional information that is not found in a given news document?
- Is it possible to integrate Reinforcement learning techniques to get extra documents for better information extraction?
- How accurately the proposed model extracts information?

1.5. Objectives of the Study

1.5.1. General objectives

The general objective of this research is to develop an automatic Information extractor from Amharic News by Integrating Reinforcement learning.

1.5.2. Specific Objective

To achieve the above-stated general objective the following specific objectives will be addressed:

- Reviewing formerly directed research and other literature on Information Extraction.
- Theoretical understanding of principles, theories, approaches, and algorithms for Information Extraction and Reinforcement Learning.
- Gathering News text data from Ethiopian News media.
- Constructing an Information extraction model from Amharic News by Integrating Reinforcement learning.
- Evaluating the performance of the proposed model by test datasets.
- Reporting the findings of the research and recommending future research direction.

1.6. Significance of the study

The end outcome of this study will be used in different beneficiaries such as different kinds of news agency organizations, personal bloggers, journalists, and anyone whose intention is getting information from Amharic news also, it's useful in some particular domain areas where information is scarce to extract.

The target beneficiaries of this study are mostly journalists and Amharic language reader information seekers. The outcome of this study also will be useful as input to different NLP tasks like information retrieval (IR), text summarization, question answering systems, text categorization, and text mining.

This study will also introduce a new approach for better information extraction from Amharic language text by integrating reinforcement learning for future researchers.

At the end of this research work, it's expected to provide an information extraction model which is integrated with reinforcement learning to extract information from a news document.

1.7. Thesis Organization

Following is how the rest of this thesis report is structured. The many IE problems are covered in Chapter 2 along with related topics. This Chapter describes what an IE system consists of, the methodologies employed, the various components required by an IE system, related works on IE systems in various languages and on various domains, and a brief explanation of the writing structure of the Amharic language. Chapter 3 discuss the methods used in this research work. The primary elements of our model, their functional interactions, and the particular sub-element of each element are briefly covered in Chapter 4. The main implementation concerns of our IE model are covered in Chapter 5 of the thesis documentation and discuss the algorithms, approaches, and procedures that were used to develop it successfully. The evaluation is presented in Chapter 6. The contribution of the research study is outlined in Chapter 7 and also this chapter illustrates some potential future study paths.

CHAPTER TWO

2. LITERATURE REVIEW

2.1. Overview

In this chapter, we provide a general overview of information extraction. The different subtasks and components of IE and also the methods used to develop an information extraction system are reviewed and presented. This chapter also reviews and presents the related NLP fields to information extraction to see their similarity and difference with IE. Different available approaches and evaluation standards used for the performance of IE are also provided in this chapter. The chapter provides a brief discussion of the Amharic language. Some related and relevant works to our research that are done so far are also reviewed and presented at the end of this chapter.

2.2. Information Extraction

Due to the information explosion of the last decade, there may be more text data in electronic form than ever before, but much of it is ignored. No human can read, understand, and synthesize megabytes of text on an everyday basis. Missed information and lost opportunities have encouraged researchers to explore various information management strategies to establish order in the text wilderness. As the size of textual information is exponentially growing, it is more than ever a crucial issue for knowledge management to develop intelligent tools and methods to give access to document content and extract relevant information. Information Extraction (IE) is one of the main research fields that attempt to fulfill this need. It aims at automatically extracting well-defined and domain-specific data from free or semi-structured textual documents.

(Elsadig, Ali Ahmed, & Mubatak Himmat, 2015) defines IE as "Information extraction (IE) is the process of scanning text for information relevant to some interest, including extracting entities, relations, and, most challenging, events–or who did what to whom when and where. It requires deeper analysis than keyword searches, but its aims fall short of the very hard and long-term problem of text understanding, where we seek to capture all the information in a text, along with the speaker's or writer's intention." Information Extraction (IE), identifying and pulling out a sub-sequence from a given sequence of instances that represents information we are interested in,

is an important task with many practical applications. Information extraction benefits many text/web applications.

(Herbrich & Thore Graepel, 2010) also defines IE as "information extraction (IE) is the task of automatically extracting structured information from unstructured and/or semi-structured machine-readable documents and other electronically represented sources. In most cases, this activity concerns processing human language texts using natural language processing (NLP)" Information extraction is also related to but distinct from another type of document processing, machine translation. Both technologies process texts in multiple languages. A machine translation system converts the entire text of a source document into a different target language; an information extraction system identifies and extracts relevant information within a document in a particular language. There is no conversion from one language to another.

2.3. Information Extraction Approaches

According to (Jerry & Ellen , 2010) There are mainly two approaches to designing an information extraction system namely: -

- Knowledge engineering approach
- Machine learning approach

2.3.1. Knowledge engineering approach

In knowledge engineering, rules are constructed by hand utilizing knowledge of application domains. Therefore, a person who creates such a type of system or is responsible for writing these rules must be an expert in the domain of the application. Knowledge engineering typically requires a lot of labor, a lengthy test-and-debug cycle, and linguistic resources, such as appropriate lexicons, in addition to skill and knowledge regarding the particular IE system (Herbrich & Thore Graepel, 2010).

Creating rules with the knowledge engineering approach requires a lot of time, and the system is also difficult to maintain. However, most of the best-performing systems are hand-crafted. With this approach, the computer system does not learn anything from the data. It only implements what human experts have learned (Johannes, February 2004).

With the information engineering method, an appropriate appearing gadget isn't difficult to construct if the information professional is on hand, and the appearance can be fine-tuned through hand-made rules. Among its disadvantages are that the development process is very

laborious, some changes in specification can be difficult to accommodate, and it requires expertise that may not be available. Due to the lack of experts who are knowledgeable about IE and the Amharic language, and the lack of NLP resources available in this language, developing an IE system will be very difficult (Appelt & David).

2.3.2. Machine learning approach

The process of handcrafting rules is very time-consuming and inefficient, and it also requires a high level of expertise, even though rule-based systems are simpler, easier to interpret and achieve higher results than rule-based systems. Machine learning techniques are widely used to create such rules automatically (Appelt & David) (Herbrich & Thore Graepel, 2010).

Following the Automatic Training Approach, it is not necessary to have anyone on hand with detailed knowledge of the IE system, or how to write rules for it. It's only necessary to have someone who understands the domain and the task well enough to take a corpus of texts and annotate the texts appropriately for the information extracted. The annotations describe a particular aspect of the system's processing (Goncalo, Helena , & Luisa). For instance, a name recognizer would be trained by annotating a corpus of texts with domain-relevant proper names. Following annotating a suitable training corpus, a training algorithm is run, and the results are information that the system can use to analyze novel texts. Another method is to interact with the user when processing a text.

As long as domain-related corpora are available, the same machine learning algorithm can be applied to different domains. Therefore, unlike knowledge engineering, machine learning is domain-independent. By analyzing the benefits and drawbacks of both approaches, one can determine what criteria should determine the choice between them. In the case of automatic training, the most crucial criterion for choosing the approach is the presence of suitable texts that can be used as training examples. In the case of knowledge engineering, the availability of a person with experience in writing extraction rules is the most important criterion for choosing the approach. In general, automatic learning systems can be categorized into three categories (Jim & Yorik) (Jerry & Ellen , 2010):

- supervised learning systems,
- semi-supervised learning systems
- Unsupervised learning systems.

2.3.2.1. Supervised learning

Training data is used in supervised learning algorithms to infer extraction rules. Consequently, no domain knowledge is required, but a large set of training data needs to be annotated according to the underlying structure of the information to be extracted. Preparation of the training data is the major bottleneck of supervised IE systems. In most systems, a large number of annotated documents is required for a particular extraction task, which limits portability. On the other hand, supervised learning relies heavily on a large amount of manually prepared training data. Even though supervised learning saves human experts' time, preparing the training data is still a time-consuming process. IE approaches supported by supervised machine learning techniques are divided into the following three categories (Jim & Yorik):

a) Rule learning

   This method employs a symbolic inductive learning process. In terms of properties and relationships between textual elements, the extraction patterns represent the training instances. Some IE systems, such as Auto Slog-TS and CRYSTAL, employ propositional learning (i.e., zero order logic), whereas others, such as WHISK and SRV, use relational learning (i.e., first order logic). This method has been used to extract information from structured, semi-structured, and unstructured documents (Jerry & Ellen , 2010)s.

b) Linear separators

   The classifiers are learned as sparse networks of linear functions in this method (i.e., linear separators of positive and negative examples). It's a popular method for extracting data from semi-structured documents. It's been used to solve problems like data extraction from job adverts and detecting e-mail address changes.

   In general, IE systems based on this technique propose an architecture based on the idea that learning the required extraction patterns is as simple as looking at the word combinations surrounding the important information (Jerry & Ellen , 2010).

c) Statistical learning

   Hidden Markov Models (HMMs) are learned as helpful knowledge to extract important pieces from documents in this approach.

   These Internet Explorer systems also differ in terms of the features they employ.

   Only basic characteristics like token string, capitalization, and token type are used by some (word, number, etc.).

Others, meanwhile, make advantage of linguistic aspects like part-of-speech, semantic information from gazetteer lists, and the outputs of other IE systems (most commonly termed entity recognizers). A few systems also make use of genre-specific data like document structure. In general, the more features a system has, the better its performance will be. Support Vector Machine (SVM), a broad supervised machine learning technique, is one of the most successful machine learning methods for IE. It has achieved state-of-the-art performance on many classification tasks, including named entity recognition (Jerry & Ellen , 2010).

2.3.2.2. Semi-supervised learning systems

A system learns from a mixture of labeled (annotated) and unlabeled data using semi-supervised learning. A small labeled data set coexists with a large unlabeled data set in many applications. It is not a good idea to train the system with only a tiny labeled data set because it is widely known that when the ratio of training samples to feature measurements is small, the training result is inaccurate. To boost performance, the system must blend labeled and unlabeled data during training. The unlabeled data can be utilized for density estimates or labeled data preprocessing, such as determining underlying domain structure. In other words, the system extracts patterns from annotated data and uses those patterns to automatically classify unannotated data. As a result, all data for the training is labeled. Semi-supervised learning saves time and effort while delivering results that are comparable to supervised learning (Jim & Yorik).

2.3.2.3. Unsupervised learning systems.

Unsupervised relation extraction aims to extract relationships from the Web without using labeled training data or specifying the sorts of relationships. For learning, the unsupervised approach does not rely on the presence of labeled corpora. Rather, it labels its training data autonomously. Self-supervised systems are those that automatically label their training data. It employs a set of generic patterns to generate relation-specific extraction rules automatically, then learns domain-specific extraction rules, and repeats the process recursively (Jim & Yorik). However, the procedure necessitates the processing of a large number of search requests, making it incredibly slow.

2.4. Components Of Information Extraction System

For different languages and domains, different scholars employ different steps for creating information extraction systems. The research work by (Goncalo, Helena , & Luisa) mainly categorizes IE into six different tasks.

- Part-of-Speech (POS) Tagging
- Named Entity Recognition (NER)
- Co-reference Resolution
- Template Element Construction
- Template Relation Construction
- Scenario Template Production

2.4.1. Part-of-Speech Tagging (POS)

POS Tagging (Parts of Speech Tagging) is a procedure to mark up the phrases in textual content layout for a specific part of a speech primarily based totally on its definition and context. It is answerable for textual content studying in a language and assigning a few particular tokens (Parts of Speech) to every word. It is likewise referred to as grammatical tagging. Define each word in a sentence statement that describes how that word is used in a sentence. This means that POS marking determines whether a word is used as a noun, adjective, verb, and so on. The POS tagger will search for possible tags or lexicon categories for any word, assuming the word is in a lexicon, and infer possible tags for unknown words. also selects possible tags for each word with an ambiguous part of speech. If multiple tags are associated with a particular word, it means that the word has a different meaning or may work in different contexts.

(Bjorn Gamback, Fredrik Olsson, Atelach Alemu Argaw, & Lars Asker, 2007) states that there are two approaches to automatic part-of-speech tagging. The rule-based approach uses linguistic knowledge to create a simple rule that uses contextual information to assign parts of speech to ambiguous words. The statistical approach (of which the hidden Markov model trained with the expected value maximization algorithm is the standard model) uses statistics collected from ambiguous or marked texts to describe sentences or texts. Estimate the probability of each possible interpretation of the section. That has been selected as the most likely ambiguity avoidance.

2.4.2. Named Entity Recognition (NER)

In natural language, nouns play a very important role in information extraction. The noun subcategory is a suitable noun. They reflect the names of people, places, and organizations, among other elements. NER is the task of recognizing and recognizing appropriate nouns in text and classifying them such as people, places, organizations, etc. (Herbrich & Thore Graepel, 2010).

Named entities are one of the most often extracted types of tokens during extracting information from documents. Named entity recognition is a classification of every word in a document as being a person-name, organization, location, date, time, monetary value, percentage, or none of the above. Some approaches use a simple lookup in predefined lists of geographic locations, company names, person names, and names of animals and other things from the gazetteers, while some others utilize trainable Hidden Markov Models to identify named entities and their type. The Message Understanding Conferences (MUC) is a series of conferences coordinated by the National Institute of Standards and Technology and the Advanced Research Projects Organization of the United States Department of Defense. MUC's seven IE conferences are thought to have had a major impact on the development of NLP science, especially in IE. the term named entity came to include seven categories; persons, organizations, locations (usually referred to as ENAMEX), temporal expressions, dates (TIMEX), percentages, and monetary expressions (NUMEX) (Chinchor, 2000).

Table 1 – MUC NER examples

Named Entity	Example
PERSON	አበበ ናትናኤል ብርሃኑ
ORGANIZATION	ዩኒቨርሲቲ ሆስፒታል
LOCATION	አዳማ ወንጂ አዲስአበባ
DATE	ነገ ህዳር 28/03/1990
PERSENTAGE	20%
MENETARY AMOUNT	10 ሚሊዮን መቶ ሺህ

2.4.3. Co-reference Resolution

Any entity in the text can be referenced multiple times and each time Differently. To identify all the methods used to name this entity throughout the document Co-reference resolution is

performed. Co-reference or anaphoric resolution, for example, Noun phrases are determined by whether they refer to the same entity. There are several types although co-references, the most common types are co-references of pronouns and appropriate nouns. In the first case, the noun is replaced by a pronoun, and in the second case, it is replaced by another noun or noun phrase.

Co-reference Resolution is the process of finding multiple references to the same object in the text. This refers to the task of identifying noun phrases that refer to the same non-linguistic unit in the text. This is especially important because the same thing is expressed for a single entity in a different sentence that contains a pronoun.

2.4.4. Template Element Construction

This is one of the IE tasks that assign descriptive information to the NER results. The various recognized name entities combined with cross-references are the inputs to the template element structure. Different named entities are recognized and have different attributes.

The task of building template elements is based on recognizing named entities and resolving cross-references. Its role is to associate descriptive information with an entity. It's about the properties that an entity has. The different named entities that are recognized have different attributes for building template elements. The structure of the template element is area dependent, as the associated information type depends on the entity type that is important to the application area.

2.4.5. Template Relation Construction

The task of determining probable relationships between template elements discovered during the template element construction activity is known as the template relation task. It looks for connections between the entities of template elements. It's all about the interrelationships between entities. The distinction between template entity and template connection is blurry because both indicate information about entities discovered by named entity recognition. The application's domain is what distinguishes them: Relationships between entities are required for template relations, and both the relation and entity types must be relevant to the application domain. The template element requires additional information about entities, which may include information about other entities; however, this data is primarily intended to improve the entity's description.

2.4.6. Scenario Template Production

Scenario templates are the prototype outputs of IE systems, and they were the term's first application. They use event descriptions to connect template element construction entities and template relation construction relations. The task of creating scenario templates is domain-dependent and, by definition, linked to the scenarios that consumers are interested in. The outcomes of the named entity, template relation, and template element, on the other hand, feed into the scenario template.

IE duties, on the other hand, are divided into five distinct and separate components, according to (Tsedalu G. , 2010) work. Because IE activity can be a fairly difficult undertaking, breaking it down into smaller tasks is beneficial. The fundamental benefit of decomposition is that it makes it easier to select the approaches and algorithms that are most suited to each task, as well as to troubleshoot an IE program quickly. The following are among the tasks considered in the survey:

- Segmentation,
- classification,
- association,
- normalization, and
- Co-reference resolution

Segmentation

The text is divided into atomic elements called segments or tokens by the Segmentation job. Even if the availability of whitespaces separating words makes this work easier in Western languages, there are rare circumstances where basic whitespace separation is insufficient. Typically, these scenarios are segmented using rules that outline how to handle each one. Oriental languages contain the most significant issues relating to this task. The Chinese, for example, do not use whitespace. As a result, with this language, fixing the challenges outlined above is insufficient. External resources are usually required in these situations. Syntactic or lexical analysis can also be combined with lexicons and grammar to complete the process of segmentation.

Classification

The type of each segment obtained in the segmentation task is determined by the Classification task. It determines the categorized output data structure where the inputs are segments, in other words. The classification of a set of segments as entities, which are items of a specific class

possibly important for the extraction domain, is the outcome of this activity. The categorization task's rule-based procedures are usually based on language resources like lexicons and grammar. Machine learning is one of the most prevalent ways of categorization. The majority of machine learning approaches utilized in this work are supervised, which necessitates the usage of an annotated corpus.

Association

The association job aims to determine the relationships between the various items found in the categorization task. The systems that conduct relationship extraction are less widespread than those that perform classification. This occurs as a result of the difficulties in producing satisfactory results in this task. Many of the approaches used in the association task are rules-based. Patterns are used in the simplest way to extract a small number of associations. Syntactic analysis is a more general rule-based method of association. Most of the time, the relationships we're looking for are grammatical ones. A verb, for example, could denote a relationship between two entities. Machine learning techniques can also be used in the association job.

Normalization and Co-reference resolution

Because they apply heuristics and rules that are specific to the data domain, normalization and co-reference resolution are the less generic aspects of the IE process. Because some information types do not follow a common format, the normalizing job is required. This is usually accomplished through the use of conversion rules that provide a pre-determined standard format. When the same real-world entity is alluded to in many ways in a text fragment, co-reference occurs.

This issue may develop as a result of the use of:

a) Different names describing the same entity can confuse
b) Expressions of classification
c) Pronoun

In most rule-based approaches to co-reference, semantic information about entities is taken into account. Clustering techniques for grouping related entities are used in a machine learning approach for co-reference resolution.

2.5. Related NLP Fields

2.5.1. Information Retrieval (IR)

Information Retrieval (IR) is a software program that organizes, stores, retrieves, and evaluates data from document repositories, particularly textual data. Information retrieval is the process of collecting content that is frequently unstructured in nature, i.e., text, and that satisfies an information demand from massive collections of data that are kept on computers. When a user enters a query into the system, for example, Information Retrieval occurs.

Returning relevant information for a certain query or subject of interest is what information retrieval is all about. It's worth noting that this information could also come in the form of broad publications; search engines are a good example of this. The initial set of documents/information and the query that specifies "what to look for" are the most significant entities recognizable for information retrieval. Information extraction, on the other hand, is mainly concerned with extracting (or inferring) general knowledge (or relations) from a collection of documents or data. Information extraction is the process of arranging unstructured data so that all of the (relevant) data can be processed easily.

The comparison of the goals of the IE and IR systems is that IR extracts relevant information from papers while IE retrieves relevant documents from collections. As a result, the two techniques are complementary, and when used together, they can provide a strong text processing tool.

2.5.2. Text Summarization

The term "summarization" refers to the process of condensing the major points of any source of information. It can be done with text, graphs, photos, videos, and other media. Written summarization is the process of condensing a long text content into a concise summary of its main ideas.

Text summarization is a process in which a computer generates an abstract or summary of one or more texts automatically. A summary, according to (Mehdi Allahyari, Seyedamin Pouriyeh, & Mehdi Assef, 2017) , is "a text created from one or more texts that convey crucial information in the original text(s) and is no longer than half of the original text(s) and frequently substantially less than that." Different approaches can be used to construct the summary from single or numerous

documents. This basic concept encapsulates three key characteristics of automatic summarization research which are:

- Summaries may be produced from a single document or multiple documents
- Summaries should preserve important information
- Summaries should be short

Even if text summarizing reduces the length of the text by extracting the most relevant sentence from bigger material, the user must still read the summary and extract the exact information she or he requires, and the data is not yet freely accessible to other computer applications.

2.5.3. Question and Answering

Question answering is a discipline found in computer science that combines information retrieval and natural language processing to create systems that automatically respond to queries presented by humans in natural language. Question answering systems are more typically able to extract answers from an unstructured collection of natural language texts. A question-and-answer system takes natural-language questions, searches a library of documents for answers, and extracts and formulates short responses.

In practically all question-answering systems, the three essential components as discussed (Ali Mohamed Nabil Allam & Mohamed Hassan Haggag, 2012) are Question processing, document retrieval, and answer extraction and formulation. Question answering is the process of extracting responses to natural language queries; however, IE systems extract the data of interest if the extraction domain is well defined. The information of interest in IE systems is in the form of slot fillers for some specified templates.

2.5.4. Text Categorization

The technique of assigning specified categories to free (unlabeled) text data is known as text categorization. Text categorization has become one of the most important strategies for processing and organizing text documents, thanks to the fast expansion of online data and information. Text classification techniques are used to classify new stories, identify intriguing information on the Internet, and lead a user's hypertext search. A text category is the result of text categorization.

2.5.5. Machine Translation

Machine translation is a branch of computational linguistics that studies the use of software to convert text or speech between languages. On a basic level, MT replaces words in one language with words in another, but this rarely results in a decent translation because recognition of entire phrases and their closest counterparts in the target language is required. Many words have several meanings, and not all terms in one language have comparable words in another. Solving this challenge with corpus statistical and neural techniques is a rapidly emerging topic that is leading to better translations, linguistic typology handling, idiom translation, and anomaly isolation.

In general, the NLP activities outlined above attempted to address the information overload difficulties that exist in today's environment. They were all used to solve problems with large amounts of text data and make the content more readable for the end user. Unlike text summarization, information retrieval, and question and answer systems, information extraction produces structured data that can be readily handled and accessed since it is stored in a database.

2.6. Related Works

Many scholars have attempted to construct Information Extraction systems utilizing a variety of methodologies and languages across many topics and languages. Among these, we'll look at some of the works that are most pertinent to this study.

2.6.1. Information Extraction from Urdu text

In the paper (Smruthi, Rohini, & Erik, 2010) The objective of their work is to develop an NLP infrastructure for Urdu that is customizable and capable of providing basic analysis on which more advanced information extraction tools can be built. Their suggested method gathers information from a variety of web sources to improve named entity tagging and Urdu-to-English transliteration. The annotated data needed to train the system's learning models are obtained by standardizing the currently limited Urdu resources available. To supplement the available annotated Urdu data, they use techniques like bootstrap learning and resource sharing from a syntactically comparable language, Hindi. Each of the new Urdu text processing modules is part of a larger text-mining platform. The tests were carried out to show that the accuracy levels met or exceeded the current state of the art.

(UmrinderPal Singh, Vishal Goyal, & Gurpreet Sin, 2012) In their work, they have developed an NLP system for Urdu that extracts basic language-related information. For POS, NER, and shallow parser modules are standardized and their results can now be used as a baseline for further improvements. The training data that they use is freely available for comparison. They have also shown that although Urdu has limited resources (required to develop statistical models), significant information and data can be successfully borrowed from a syntactically similar language like Hindi. Experiments performed in their work show that developing an Urdu-specific NLP system is a must and tools developed for Hindi cannot be directly used in Urdu despite the similarity between the two languages. Their framework can be used to preprocess text and perform further analyses such as emotion detection, agent-target identification, and question opinion mining. They also provide different transliteration methods based on pronunciation and IPA standards that can improve cross-lingual search and machine translation.

2.6.2. Information Extraction from the English text

(Ronen Feldman, Benjamin Rosenfeld, & Moshe Fresko, 2006) presents a hybrid approach to IE. To extract entities and relations at the sentence level, the authors suggest a hybrid knowledge-based and statistical machine learning approach. They demonstrate a hybrid entity and relation extraction method that incorporates the benefits of both knowledge-based and statistical machine learning approaches. The extraction criteria are constructed manually in this manner, and the probability of the extracted texts being part of the database slot is trained using an annotated corpus. This method allows the knowledge engineer to construct very simple and naive rules while preserving their potency, resulting in a significant reduction in the amount of effort required.

Furthermore, the training data is far smaller than the training data required for a pure machine learning system (to achieve comparable accuracy results). The authors use DIAL, which is built on a general-purpose rule language for creating extraction knowledge, and statistical HMM for machine learning. Three separate corpora, MUC7, ACE-2, and an industry corpus, are used to test the effectiveness of their technique. They conclude that combining modest handcrafted rules with machine learning methods will improve machine learning performance.

Another work (Jakub Piskorski & Roman Yangarber, 2012) provides a text categorization strategy for extracting information from business cards and processing address changes from email messages automatically. They scan business cards and perform certain cleaning operations afterward to

ensure that the text file is ready for the text classifier to extract information from. They sort the different lines of the scanned text into Address, Name, Title, Corporate name, Company Logo, Phone, Web address, Email address, Telex number, cable number, and other miscellaneous text using the Nave Bayes algorithm. They believe that the Naive Bayes method is reasonably accurate, but it misses significant structural limitations such as how the business card is laid out and which attributes come first. HMM handles such constraints in their job.

Because Information extraction (IE) systems need very large amounts of annotated data to provide high performance and unavailability of annotated data In (Narasimhan, Adam Yala, & Regina Barzilay, 2016) provides another method for increasing extraction correctness when a big training corpus is not available and to explore the task of obtaining and uniting other evidence to increase extraction accurateness in areas where the quantity of training data is rare the researchers proposed a model which is trained as a Deep Q Network performs 7.2% and 5% better than the traditional extractors on two different domains. They use the Markov decision process (MDP) used to design the Information Extraction task, the extraction system uses a maximum entropy classifier and uses a Reinforcement Learning approach trained to improve a reward function that returns extraction exactness.

Their study shows the task of acquiring and incorporating additional evidence to expand information extraction accurateness for domains with limited access to training data. This method includes allotting search queries, extraction from new sources, and settlement of extracted values, which is repeated until enough confirmation is gained.

2.6.3. Information Extraction from Amharic text

In a paper (Hirpassa, 2017) the researcher Sintayew Hirphassa developed an IE system that extracts information from Amharic vacancy announcement text due to problems such as: manually extracting information from such an often unstructured or semi-structured text is a very tedious and lingering task, getting accurate information for decision-making unstructured text is a big challenge, Unavailability of tools for extracting valuable information which is efficient enough to satisfy the users of Amharic language has also been a major problem. The researcher uses Part-of-Speech (POS) Tagging, Named Entity Recognition (NER), Syntax Analysis, Co-references and Discourse Analysis, Extraction Patterns and Bootstrapping methods in the research work and the results obtained from experiments show 79.56% precision and 66.6% recall on the 34 Amharic vacancy announcement texts test dataset. This work presented a rule-

based IE system for Amharic text and develops an IE system using the knowledge engineering approach.

Another work (Tsedalu G. , 2010) designs a generic model of an information extraction system from Amharic news text using a classification machine learning approach. This research work proposed a generic model for Amharic Text information extraction based on which they designed an IE system which they called an ATIE (Amharic Text Information Extraction) system. It has four main components and is developed using java and the open-source machine learning algorithms in Weka. IE component is developed using HMM (Hidden Markov Model) and Evaluation is done on different scenarios in each component of the proposed system and yields a higher accuracy rate of more than 95%

Another work (Fisseha, 2020) designs a model for Amharic OIE which can extract open-domain relations and their arguments automatically from Amharic text because of the problem that Traditional IE relied on a significant amount of human involvement in preparing handcrafted extraction patterns and hand-tagged training data, domain dependability, previous IE models are not scalable or portable across domains and not suitable for massive and heterogeneous corpora like Web. The model proposed an appropriate approach and architecture for the AOIE system. Identified that the rule-based approach operating on deep parsed sentences yields the most promising results for OIE systems, as they enable higher precision. The result of the evaluation showed that, out of 229 clauses, 74% of embedded clauses were correctly identified and 72% correctly paraphrased. The AOIE system has extracted 556 instances of relation from 215 sentences with a precision of 88%.

2.7. The Amharic Language

Amharic is a Semitic language that is spoken across Ethiopia. It is the official working language of Ethiopia's Federal Democratic Republic and hence has official status throughout the country. It is also the official or working language of several federal states and regions, including Amhara and the multi-ethnic Southern Nations, Nationalities, and Peoples area. Outside of Ethiopia, Amharic is spoken by millions of emigrants (most notably in Egypt, the United States, Israel, and Sweden), as well as in Eritrea.

2.7.1. The Amharic writing system

Amharic writing system as studied and published by (ያማኮ 1987) is written using the Fidel(ፊደል) writing system, which was borrowed from the now-extinct Ge'ez language. In Amharic, seven vowels are employed, each of which has seven alternative forms (orders) to mirror the seven vowel sounds. That is, each of the 33 Amharic symbols has seven variants, each of which represents both a consonant and a vowel, making the Amharic script syllabic. The basic form is the first order, with 33 basic forms totaling 231 characters.

	Base Sound	Orders						
		1st (ä)	2nd (u)	3rd (i)	4th (a)	5th (e)	6th (ə)	7th (o)
1	h	ሀ	ሁ	ሂ	ሃ	ሄ	ህ	ሆ
2	l	ለ	ሉ	ሊ	ላ	ሌ	ል	ሎ
3	h	ሐ	ሑ	ሒ	ሓ	ሔ	ሕ	ሖ
4	m	መ	ሙ	ሚ	ማ	ሜ	ም	ሞ
5	s	ሠ	ሡ	ሢ	ሣ	ሤ	ሥ	ሦ
6	r	ረ	ሩ	ሪ	ራ	ሬ	ር	ሮ
7	s	ሰ	ሱ	ሲ	ሳ	ሴ	ስ	ሶ
8	š	ሸ	ሹ	ሺ	ሻ	ሼ	ሽ	ሾ
9	q	ቀ	ቁ	ቂ	ቃ	ቄ	ቅ	ቆ
10	b	በ	ቡ	ቢ	ባ	ቤ	ብ	ቦ
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
31	p'	ጰ	ጱ	ጲ	ጳ	ጴ	ጵ	ጶ
32	f	ፈ	ፉ	ፊ	ፋ	ፌ	ፍ	ፎ
33	p	ፐ	ፑ	ፒ	ፓ	ፔ	ፕ	ፖ
34	v	ቨ	ቩ	ቪ	ቫ	ቬ	ቭ	ቮ

FIGURE 4 - Amharic alphabet

2.7.2. Amharic Punctuation Marks and Numerals

Punctuation marks are standard symbols that make reading and understanding a written text easier. There are roughly ten punctuation signals in Amharic. However, the AratNetib (":"), the question mark ("?"), the exclamation mark ("!"), and the word separator Hulet Neteb is the most commonly used marks (":").

The Arabic number system, the symbols of the Ethiopic number system, and employing words and symbols of the Arabic numeral system can all be used to indicate numbers in Amharic.

Table 2 - Amharic numerals

Arabic	Ethiopic	Alphanumeric	Arabic	Ethiopic	Alphanumeric
1	፩	አንድ	20	፳	ሃያ
2	፪	ሁለት	30	፴	ሰላሳ
3	፫	ሦስት	40	፵	አርባ
4	፬	አራት	50	፶	አምሳ /ሀምሳ

5	፭	አምስ ት	60	፷	ስልሳ /ስድሳ
6	፮	ስድስት	70	፸	ሰባ
7	፯	ሰባት	80	፹	ሰማኒያ
8	፰	ስምንት	90	፺	ዘጠና
9	፱	ዘጠኝ	100	፻	መቶ
10	፲	አስር	1000	፼	ሺ/ሽህ

2.7.3. Amharic Word Classes

The Amharic language has been classified into the following word categories: ስም(noun), ግስ(verb), ቅፅል (adjective), ተውሳከግስ(Adverb) and መስተዋድድ (preposition).

a) Noun: If a word can be pluralized by adding the suffix ኦች and used to nominate something like a person or an animal, it is classified as a noun. In a sentence, it serves as the subject. Pronouns, which were formerly considered an autonomous category by linguistics professionals, have been categorized as nouns after taking into account the distinctive features of the language, as prior linguists just adopted the English language structure for Amharic.

The following are some of the pronouns in Amharic ይህ, ያ, እሱ, እስዋ, እኔ, አንተ, አንች…; quantitative specifiers, which includes አንድ, አንዳንድ, ጥቂት, በጣም…; and possession specifiers such as የእኔ, የአንተ, የእሱ.

b) Verb: any word which can be placed at the end of a sentence and which can accept suffixes as /ህ/, /ሁ/, /ሽ/, etc. which is used to indicate masculine, feminine, and plurality is classified as a verb. For example, in "አበበ አንበሳ ገደለ" "ገደለ" is a verb since it appears at the end of the sentence.

c) Adjective: a word that comes before a noun and adds some kind of qualification to the noun. But every word that comes before a noun is not an adjective. For it to be an adjective it should also satisfy the condition when the word "በጣም" is added to it, it should be meaningful. For example, "ትልቅ በግ" in this example "ትልቅ" is an adjective to check it is an adjective adding the word "በጣም" before the adjective if it is meaningful, it is an adjective if not it isn't an adjective. In this case, it is meaningful, and "ትልቅ" is an adjective.

d) **Adverb:** a word that qualifies the verb by adding extra ideas from time, place, and situations point of view. The following are adverbs in Amharic ትናንት, ገና, ዛሬ, ቶሎ, ምንኛ, ከፉኛ, እንደገና, ጅልኛ and ግምኛ.

e) **Preposition:** a word that doesn't take any kind of suffix and prefix, that can't be used to create other words, and which doesn't have meaning by itself but can represent different adverbial roles when used with nouns. The different propositions include ከ፣ ለ ፣ ወደ፣ ስለ ፣ እንደ…፣ ወዘተ.

CHAPTER THREE

3. RESEARCH METHODOLOGY

3.1. Overview

The construction of the methodology for the study is the chapter's main objective. The methods and processes employed to accomplish the goal of the current study are described in this section. We've discussed the methodology and strategy used in the study, which began with problem identification, defined solution objectives, testing, and performance evaluation. The chapter also explains the software tools that were used to apply multiple algorithms for this research project and discusses the performance evaluation criteria that were used to assess the model.

3.2. Methodology

The four most frequently utilized research methodologies are mixed, design-oriented, quantitative, and qualitative, according to (Yong, 2014) . In quantitative research, numerical data are gathered and analyzed to characterize, explain, forecast, or control relevant occurrences. Numerical data processing is a challenging task that requires a systematic approach. Deductive reasoning is used in quantitative research. The goal of qualitative research is to better understand a particular phenomenon of interest by gathering, analyzing, and interpreting extensive narrative and visual evidence. The simultaneous examination of numerous facets of a phenomenon and an endeavor to understand things as they are in the natural world are two characteristics of qualitative research. The mixed research methodology is a way of combining quantitative and qualitative research in a single study. To comprehend a phenomenon, a researcher can map out a systematic process using research methods. Depending on the data at hand and how pertinent the data might be while attempting to analyze an issue or occurrence, various approaches must be applied (Vaismoradi, 2013).

The experimental method of quantitative research has been applied in this study. As a result, we have conducted numerous tests to examine the performance of the model. After determining the present research topic, we reviewed earlier studies that had been done in the field. However, during a literature review of publications especially linked to our case, we discovered several limitations. We have created study topics to get around these constraints. Then the steps

necessary for achieving the study's goal were taken. The created recommended remedy was then put into practice, and the findings were examined.

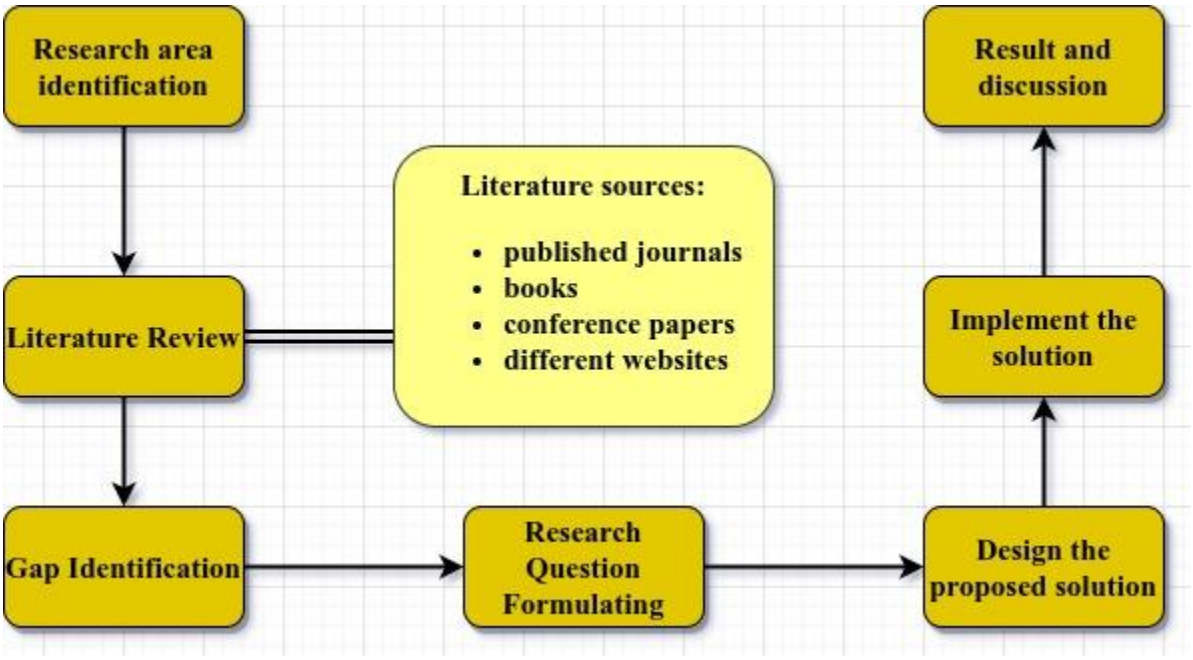


FIGURE 5 - Research methodology

3.3. Data Collection Methods

We used three data gathering sources because data is the central tenet of NLP research (Erik Cambria & Bebo White, 2014). We used secondary data sources as our sources. After taking into account the authenticity of the source and the importance of their contents concerning the acceptance of the source sites in society, the data collecting sources were selected. These data sources include Walta TV, Fana Broadcasting Corporation (FBC), and Ethiopian Broadcasting Corporation (EBC). In addition, we have made use of earlier thematic researchers.

The collected dataset used in this research work consists of 1590 news stories with 65810 sentences totaling 225269 words. The news was gathered at random from several domains, although the business category made up the majority of the news.

3.4. Tools

Python, a high-level programming language that is translated by a Python interpreter, is the primary programming language used in this study. Python is the finest language for AI and machine learning thanks to several advantages (AI). One of text analytics' key characteristics is

that it is open source and effective at addressing machine learning criteria like managing enormous data sets and statistical computation.

3.4.1. amseg

The Amharic segmenter (amseg) is a publicly available python library that is a part of semantic models for Amharic projects that can divide an Amharic document into sentences and tokens and can also be used to romanization and normalization of Amharic language texts (Introducing Various Semantic Models for Amharic: Experimentation and Evaluation with Multiple Tasks and Datasets, 2021).

3.4.2. am-flair-ner and am-flair-pos

From semantic models for the Amharic project (Introducing Various Semantic Models for Amharic: Experimentation and Evaluation with Multiple Tasks and Datasets, 2021) the two models am-flair-ner and am-flair-pos are used in this study. The am-flair-ner model is a publicly available flair-based named entity recognition classifier model which has 12 tags with good performance compared with other models currently as stated in their paper. the am-flair-pos is also a publicly available flair-based Part of Speech tagging model for the Amharic language with high performance compared with other POS taggers proven through experiments (Impacts of Homophone Normalization on Semantic Models for Amharic, 2021). The POS tagger has 66 tags.

3.4.3. gym

The gym is an open-source Python package that enables the development and comparison of reinforcement learning algorithms by offering a common API for coordinating learning algorithms with environments and a common collection of environments that adhere to that API. Since its release, Gym's API has emerged as the industry norm for carrying out this (OpenAI Gym, 2016).

The gym library enables the researchers to create a custom build environment and provide the DQN agent for this study since the OpenAI gym is a place where learning agents can be created and tested. It has a specific purpose and is ideal for reinforcement learning agents.

3.4.4. beautiful soup

A Python package called Beautiful Soup is used to extract data from markup languages like HTML, XML, and others. With the aid of Beautiful Soup, we can extract specific information from a webpage, strip away the HTML syntax, and save the data. It is a web scraping program that assists in cleaning up and parsing the documents we have downloaded from the internet. For processed pages, it generates a parse tree that can be used to extract HTML data for web scraping (Beautiful Soup Documentation, 2019).

Other tools used for this research work are listed in the below table (Table 3).

Table 3 - research tools

Tool	Type	Description
Python	Programming Language	Python is an interpreted, object-oriented, high-level, dynamically semantic programming language. It is frequently used in data science, artificial intelligence, and machine learning
Visual studio code (ubuntu)	software	Visual Studio Code is a quick yet effective source code editor that runs on Windows, macOS, and Linux.
Microsoft Word	software	Used to write and prepare documents
Draw.io	Online tool	An online tool for drawing diagrams, architectures, UML, and other types of drawings
Scikit-learn	Python Library	It is a Python package that offers several effective tools for statistical modeling and machine learning.
gensim	Python Library	Gensim is a free, open-source Python toolkit that aims to represent documents as semantic vectors as quickly and painlessly as possible for humans and computers.
NumPy	Python Library	The Python package NumPy is used to work with arrays.
TensorFlow	Python Library	a Python library for quick numerical computations. It is a foundation library that can be used to build Deep

		Learning models directly or through the usage of wrapper libraries created on top of TensorFlow that make the process easier.
Keras	Python Library	Neural network implementation is made simple with the help of Keras, a deep learning API that runs on top of the machine learning framework TensorFlow.
Matplotlib	Python Library	Python's Matplotlib toolkit provides a complete tool for building static, animated, and interactive visualizations.

3.5. Evaluation Matrices

To solve the difficulties at hand, IE systems may adopt very different approaches. It is frequently impossible to make a direct comparison of the outcomes. To determine which methodology works best in a specific situation, we need a comparative method. Additionally, the performance of individual IE components must be assessed.

The message understanding conferences prompted the need for evaluation measures for the information extraction problem. Although the event is referred to as a "conference," it might also be regarded as a "competition" between information extraction research groups or an "assessment" of their systems' performance. As a result, the main goal of these conferences was to assess the state-of-the-art in the field of information extraction, as well as to identify and promote innovative techniques for information extraction.

The concept of true and false positives, as well as true and false negatives, can be used to evaluate information extraction methods. True positives are entities that have been accurately extracted, whereas false positives are information that has been incorrectly extracted. False negatives are information that is important but not extracted and is left in the text; real negatives are information that is not extracted and is not relevant to the job.

Precision (P): The proportion of correctly extracted entities (Ncorrect) to the overall number of extracted entities (Nresponse) (the ratio between the number of needles in a hand and the number of needles and straws in the hand) is known as precision (P). It relates to the information that has been extracted trustworthiness.

$$P = \frac{Ncorrect}{Nresponse}$$

Recall(R): The proportion of correctly extracted entities (Ncorrect) to the overall number of manually extracted entities (Nkey) (the ratio between the number of needles in the hand and the total number of needles in the haystack) is referred to as recall. In a nutshell, it indicates how much of the data was accurately extracted.

$$R = \frac{Ncorrect}{Nkey}$$

Precision and recall must be considered while comparing the performance of different systems. However, because comparing the two parameters at the same time is difficult, numerous combination approaches have been proposed. In the discipline of information retrieval, (Monika Arora, Uma Kanjilal, & Dinesh Varshney, 2016) investigated the relationship between precision and recall. Their discoveries, however, can be applied to the field of information extraction as well. As they explained, recall is a measure of extraction efficacy, whereas accuracy is a measure of extraction purity. In a single measurement, the F measure combines precision P and recall R.

The average return: is the metric that is most frequently used to assess a policy. The total benefits earned while operating a policy in an environment for an episode make up the return. An average return results from running several episodes. Given the policy, environment, and number of episodes, the following function calculates the average return. For this research project, we shall employ the same evaluation.

loss function: A loss function, at its heart, measures how well your prediction model performs in terms of being able to forecast the anticipated result (or value). It gauges "how well" the output has been given a model output and the actual data. And it is used to change the model's parameters. The most commonly used loss functions are:

- **Mean Squared Error:** Mean Squared Error (MSE) is the workspace of fundamental loss functions because it is straightforward to comprehend, simple to use, and generally effective. You square the difference between your model's predictions and the actual data, average it across the entire dataset, and then use that result to determine MSE. Regardless of the sign of the predicted and actual numbers, the outcome is always positive, and a perfect value is 0.0**.**

- **Mean Absolute Error:** The concept of Mean Absolute Error (MAE), which differs from the MSE only slightly, is fascinating because it offers virtually exactly the opposite

features. The MAE is calculated by taking the difference between the predictions made by your model and the actual data, adding the absolute value to that difference, and then averaging the result across the entire dataset.

Reward: Given an agent (a model) and an environment, the environment assigns the agent a reward (or a penalty) after the agent completes an activity to determine "how good" the action has been. Simple rewards are either +1 or -1.

Therefore, Loss functions provide more than just a static illustration of how well the model is doing; they also serve as the foundation upon which the algorithms fit data. When choosing the optimum parameters (weights) for the data, the majority of machine learning algorithms employ some form of a loss function. Importantly, the activation function that is employed in the output layer of the neural network has a direct impact on the loss function. These two elements of design are interconnected. Due to these reasons, the MAE is selected for Reinforcement learning as a loss function alongside the Reward.

CHAPTER FOUR

4. RESEARCH DESIGN

4.1. Overview

In this chapter, we present our proposed model which is Information Extraction from Amharic news text by integrating Reinforcement Learning (AIERL). This section contains a detailed overview of the proposed AIERL model architecture. The model's primary components, as well as their subcomponents and the interaction between them, will be shown.

4.2. The AIERL

The proposed model Information Extraction from Amharic news text by integrating Reinforcement Learning (AIERL) have two major independent phases. The two phases are: -

1. The Extractor
2. The RL

Each phase of the proposed model has a separate task to perform and an output of one phase used as an input for the next phase. The Extractor phase has the following major components: The extractor preprocessor, language-based learning and the extractor postprocessor each component contain internal subcomponents. This phase accepts Amharic news text as an input and outputs formatted extracted information into predefined categories. The RL phase is the backbone phase of the proposed AIERL model. The RL phase is represented by a Markov decision process (MDP), in which the model learns to use external sources to improve extractions from a source article. This phase has an environment, state, action, reward and agent. In this phase two sets of the Extractor outputs will be represented in the environment and an agent that’s trained in making decisions by observing the environmental state is deployed. After taking outputs from the first phase a new set of extracted information will be generated by the agent as a final output.
The integration of the two phases works with the initially given input news text, first, the extractor extracts categorized information from the given text then using the extracted information’s query will be generated to search the web for news after scraping the news from the web only related and the most similar news will be selected and after selecting the most similar news the selected news will be passed to the extractor, the extractor will extract

categorized information from the selected news and pass it to the RL, finally using the two sets of extracted information the RL agent will decide what action to take.

The general structure of the AIERL is shown in the below figure (figure 6) and in the following subsections the two phases with their components will be discussed in details.

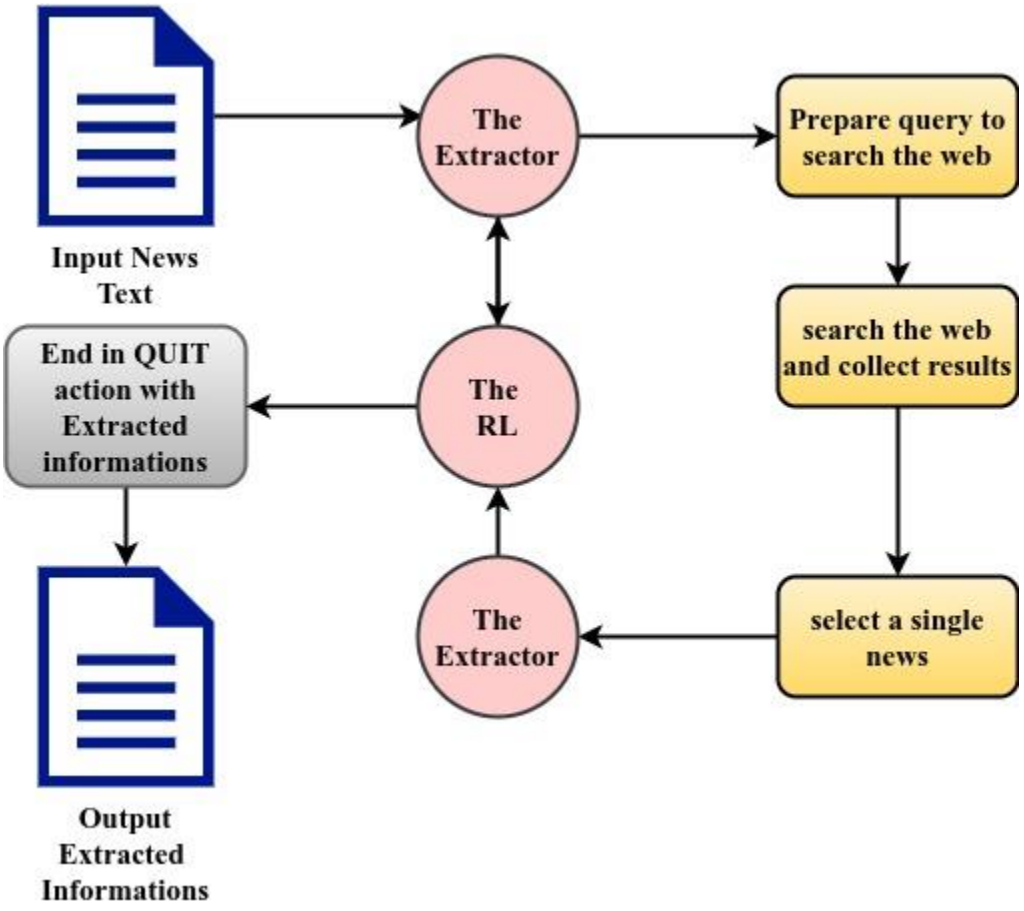


FIGURE 6 - the general structure of AIERL

4.3. Phases of AIERL

Regardless of the language or domain for which the IE system is designed, every IE system has three essential components: linguistic preprocessing, learning and extraction, and post-processing. Other subcomponents are contained in each of the main components in addition to these three.

The proposed AIERL has two phases called the Extractor and The RL as explained in the above section. The two phases interact with each other to make the AIERL by working interactively as input and output for each other.

4.3.1. The Extractor

This phase which is called the Extractor has the following components: The extractor preprocessor, language-based learning and the extractor postprocessor. The extractor preprocessor tokenizes the given input news text into sentences and tags each word in the text with appropriate POS and NER tags. The language-based learning takes the output of the first component and based on the Amharic language grammatical information it divides the sentences into subsentence (simple sentence) and identifies noun and verb phrases from the subsentence based on POS and morphological tags of words and also this component is responsible on finding every mention of individuals, locations and organizations in the subsentence based on the NER tags. The extractor postprocessor formats the output into predefined formatting categories. The relationship between these components is shown in the below figure [figure 7].

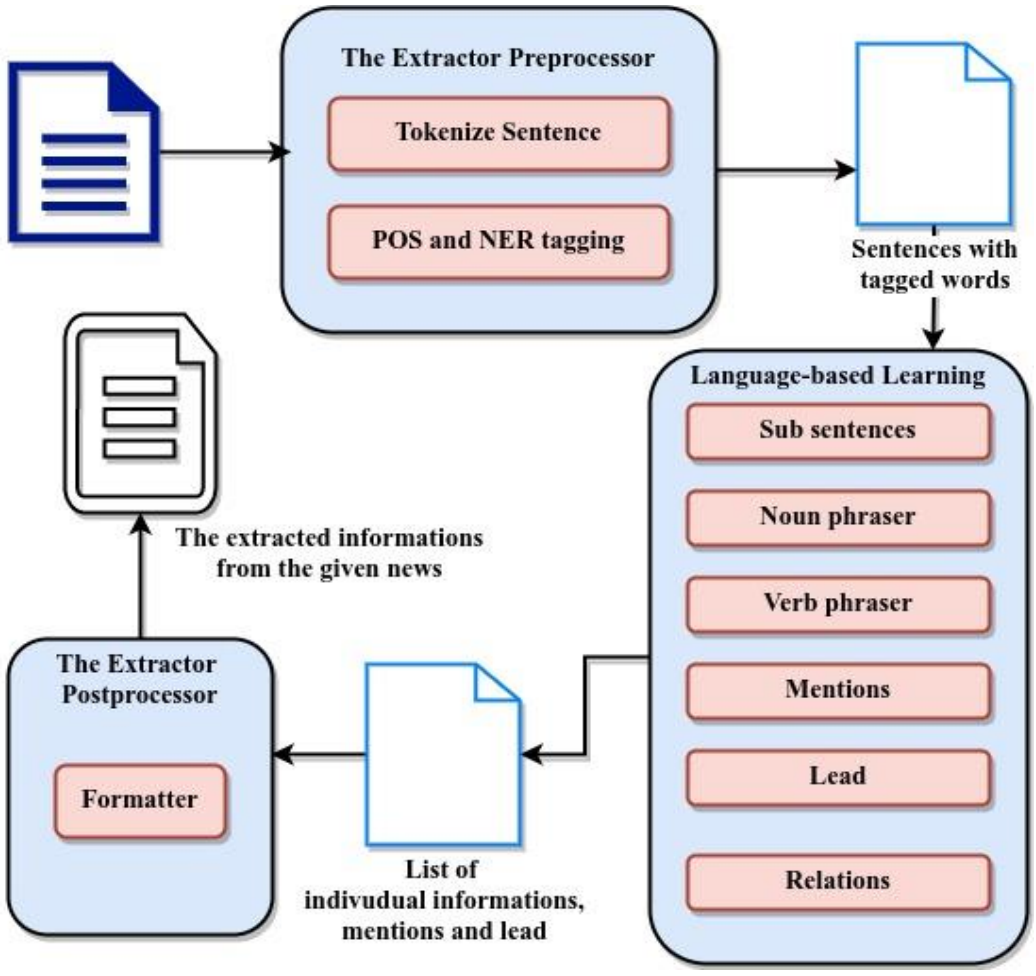


FIGURE 7 - Architecture of the Extractor phase

4.3.1.1. The Extractor preprocessor

The Extractor preprocessor is a component in the extractor phase which is responsible for taking input news text and preparing it for the subsequent component. In this component, two tasks are performed: Tokenize sentence POS and NER tagging. The output of this component is a list of sentences with the corresponding tag of POS and NER for every word in the sentence.

4.3.1.1.1. Tokenize Sentences

The Extractor preprocessor starts by tokenizing the given Amharic news text into sentences. Since Amharic writing is grammatically different as discussed in detail in chapter 2(The Amharic Language) it uses its unique sentence delimiters unlike other Latin-based languages such as English.

To split the text into sentences we have used a publicly available Amharic segmenter/tokenizer tool called amseg (Introducing Various Semantic Models for Amharic: Experimentation and Evaluation with Multiple Tasks and Datasets, 2021) which is part of the semantic models for Amharic language project (Impacts of Homophone Normalization on Semantic Models for Amharic, 2021).

4.3.1.1.2. POS and NER tagging

After the input news text is divided into sentences from the Tokenize Sentences sub-component the next task is to tag each sentence with the appropriate Part of speech (POS) and Named Entities recognition (NER) tag for every word in the sentence. Since there is a publicly available Amharic POS and NER model from the semantic models for the Amharic language project, we have used the FLAIR-based Amharic NER classifier model and the Flair-based Amharic POS tagger due to the reason that the provided models are of the highest in performance from other models built for Amharic language as stated in their paper based on experimental results (Impacts of Homophone Normalization on Semantic Models for Amharic, 2021).

The POS model contains a total of 66 tags and the NER model contains a total of 12 tags

The POS Dictionary with 66 tags: <unk>, O, N, NUMP, NUMCR, VP, VPS, VREL, NP, PUNC, NUM, NPC, NCS, VS, V, PREP, PRONPS, AUXC, ADJ, NPS, PRONS, VRELS, NS, ADV, NC, AUX, CONJ, PRON, ADJP, VRELCS, PRONC, PRONP, PRONCS, VN, VPCS, AUXS, VC, ADVP, VCS, INT, VPC, ADJS, ADJPCS, NPCS, VNS, AUXCS, ADJC, NUMOR, VRELC, PRONPC

The NER Dictionary with 12 tags: <unk>, O, B-ORG, I-ORG, B-LOC, B-PER, I-PER, B-TTL, I-LOC, B-TIME, I-TIME, I-TTL

4.3.1.2. Language-based Learning

The Amharic language follows a subject-object-verb (SOV) grammatical pattern as opposed to, for example, the English language which has a subject-verb-object (SVO) generally the verb goes at the end of the sentence (ይማም 1987).

Example:

- "አበበ በሶ በላ": in this sentence, the word "በላ" is the Verb and "አበበ" is the Subject of this sentence, from this sentence we can see the grammatical pattern of an Amharic sentence which is SOV.

This component of the Extraction phase is very central for the overall information extraction task a list of solo information and mentions in the news will be the output of this component. There are six subcomponents found in this Language-based Learning component specifically: Sub sentencer, noun-phrases, verb-phrases, mentions, lead, and Relations.

4.3.1.2.1. Sub sentencer

state-of-the-art IE systems recognize relationships between entities in a sentence by matching patterns over its POS tags or dependency tree. However, syntactically complex sentences, in which relevant relations are frequently found in multiple clauses, pose a challenge to current IE approaches, which are prone to incorrect and missing extractions. We have created a sub-sentence algorithm that uses a set of hand-crafted grammar rules to break down complex and compound sentences into simple sentences. The rules are based on the POS tags and the grammatical rules of Amharic sentences. Sentences with complex syntax are thus converted into several simple sentences that are easier to process.

Example:

- 50 ሺህ ካሬ ሜትር ቦታ ላይ ያረፈው ፋብሪካው በ5 ነጥብ 8 ቢሊየን ብር ወጪ መገንባቱ እንደተገለፀ እና ፋብሪካው ሙሉ ለሙሉ ወደ ስራ ሲገባ ለ700 ዜጎች የስራ ዕድል ይፈጥራል ተብሏል።

  This sentence is a perfect example of a compound or complex sentence. After this sentence undergo through sub sentencer sub component will be changed to a list of simple sentences:

  Sub 1: 50 ሺህ ካሬ ሜትር ቦታ ላይ ያረፈው

  Sub 2: ፋብሪካው በ5 ነጥብ 8 ቢሊየን ብር ወጪ መገንባቱ እንደተገለፀ

Sub 3: ፋብሪካው ሙሉ ለሙሉ ወደ ስራ ሲገባ ለ700 ዜጎች የስራ ዕድል ይፈጥራል ተብሏል።

To divide the sentences and generate new simple and independent sentences, a series of criteria is applied. The sub-sentencer checks a verb in the given sentence first. If the sentence contains only one verb, it is a simple sentence, and there is no need to sub-sentence it. If more than one verb is detected or the POS tag of the word is V (Verb), VP (Verb attached with a preposition), AUX (Auxiliary verb), and the word is not found at the end of the sentence, the sentence is complex, and it will be sub sentenced.

```
Input: dictionary of sentences with tagged words
Start
For sent in sentences with tagged words
  For s in sentence with tagged words[sent]
     If the word tag starts with ('V','PUNC','AUX')
       Initialize the next word
       If next word tag not starts with ('V','PUNC','AUX')
          Then this marks the end of the sub sentence
       Else see the third word
          If third word tag not starts with ('V','PUNC','AUX')
          Then this marks the end of the sub sentence
End
Output: dictionary of sub sentences with tagged words
```

FIGURE 8 - Sub sentence algorithm

The output of this sub-component is a list of simple sentences with tagged words and will be input for the next subsequent sub-component.

4.3.1.2.2. Noun Phrases

The general task of this sub-component is to identify and tag noun phrases found in one simple sentence. According to (ይማም, 1987) from Amharic grammatical pattern SOV, a simple sentence contains at least one noun phrase the subject, and the object. Each phrase may be a composite of Nouns, pronouns, adjectives, prepositions, and conjunctions. This sub-component is responsible to detect a noun phrase and tag with the ‘PHRASE_N’ tag.

Example:

- From the above example the first two sub sentences which are:
  Sub 1: 50 ሺህ ካሬ ሜትር ቦታ ላይ ያረፈው

Sub 2: ፋብሪካው በ5 ነጥብ 8 ቢሊየን ብር ወጪ መገንባቱ እንደተገለፀ

After going through this sub-component will have:

Sub 1: 50 ሺህ ካሬ ሜትር ቦታ ላይ ያረፈው

one noun phrase:

1. 50 ሺህ ካሬ ሜትር ቦታ

Sub 2: ፋብሪካው በ5 ነጥብ 8 ቢሊየን ብር ወጪ መገንባቱ እንደተገለፀ

Two noun phrases:

1. ፋብሪካው
2. በ5 ነጥብ 8 ቢሊየን ብር ወጪ

To perform this task each POS tag of words in one sub-sentence is checked for Nouns, pronouns, adjectives, prepositions, and conjunctions if they are found consecutively, they will be treated as one noun phrase and a preposition tag NP (Noun attached with a preposition) will mark the second noun phrase if exists.

```
Input: dictionary of sub sentences with tagged words
Start
Initialize tracker to false
For words in sub sentences with tagged words
     Initialize phrase word as empty
     If tracker is false and the word tag starts
     with('N','CONJ','PRE','ADJ','PRON')
        This word index marks the beginning of the phrase
        Set the tracker to true
        If next word tag is not ('N','CONJ','PRE','ADJ','PRON')
          This word index marks the end of the phrase
          Add phrase word from the begging up to this word
          Set tracker to false
End
Output: dictionary of sub sentences with phrased and tagged words
```

FIGURE 9 - Noun phrase algorithm

4.3.1.2.3. Verb Phrases

Relation in one simple sentence is described by a verb that will explain the relationship between the noun phrases (Subject and Objects) and sometimes in Amharic language text a simple

sentence may have more than one verb explaining a single relation (ይማም, 1987) in this research work we combine those verbs and create a verb phrase explaining one relation.

Example:

- From the above example of sub sentence:

  Sub 1: 50 ሺህ ካሬ ሜትር ቦታ ላይ ያረፈው

  Sub 2: ፋብሪካው በ5 ነጥብ 8 ቢሊየን ብር ወጪ መገንባቱ እንደተገለፀ

  Sub 3: ፋብሪካው ሙሉ ለሙሉ ወደ ስራ ሲገባ ለ700 ዜጎች የስራ ዕድል ይፈጥራል ተብሏል።

  Of the 3 sub-sentences Sub 1 contains only one verb explaining the relation of the noun phrase. The verb is 'ያረፈው'. But the sub-2 and sub-3 both contain 2 verbs:

  Sub 2: Verbs 'መገንባቱ' and 'እንደተገለፀ' together form a verb phrase "መገንባቱ እንደተገለፀ" explaining the relation

  Sub 3: Verbs 'ይፈጥራል' and 'ተብሏል' together form a verb phrase "ይፈጥራል ተብሏል" explaining the relation.

```
Input: dictionary of sub sentences with phrased and tagged words
Start
Initialize tracker to false
For words in sub sentences with phrased and tagged words
If tracker is false and word tag starts with('V','PUNC','AUX')
  This word index marks the beginning of the verb phrase
  Set the tracker to true
  If next word tag is not('V','PUNC','AUX')
     This word index marks the end of the phrase
     Add phrase word from the begging up to this word
     Set tracker to false
End
Output: dictionary of sub sentences with verb phrase, noun phrase
and POS/NER tagged words
```

FIGURE 10 - Verb phrase algorithm

4.3.1.2.4. Mentions

The NER tagger used in this research from the semantic models for the Amharic language project (Introducing Various Semantic Models for Amharic: Experimentation and Evaluation with Multiple Tasks and Datasets, 2021) which is the FLAIR-based

Amharic NER classifier model tags each word with their corresponding NER tag at The Extractor preprocessor component. Using the NER tag at this stage this sub-component will store all mentions of location, person, and organizations.

Example:

- በደቡብ ክልል ባለፉት 11 ወራት ከ11 ነጥብ 6 ቢሊየን ብር በላይ ገቢ መሰብሰቡን የክልሉ ገቢዎች ቢሮ አስታወቀ፡፡ የቢሮው ኃላፊ ወይዘሮ ዘይቱና ኢብራሂም እንደገለጹት በበጀት ዓመቱ 13 ነጥብ 1 ቢሊየን ብር ለመሰብሰብ ታቅዶ ወደ ስራ የተገባ ሲሆን÷ ባለፉት 11 ወራት ከ11 ነጥብ 6 ቢሊየን በላይ ብር ገቢ መሰብሰብ ተችሏል።

  All the mentions from the above-given news will be:

  በደቡብ ክልል, የክልሉ ገቢዎች ቢሮ, ወይዘሮ ዘይቱና ኢብራሂም

The final output of this stage is a list of all the mentioned locations, persons, and organizations from the given Amharic news.

4.3.1.2.5. Lead

According to (Mckane, 2014) every news has a lead role for which the news narrates about in this research work we call that ‘lead’ and will be identified in this sub-component. A lead will always be found in the title or headline of the news according to the general guide and structure of the news.

To find the most probable lead from the given news the headline or title of the news will be examined. Using the POS and NER tags of the news title or headline if the headline starts with the PREP (preposition) tag and the next word is in the list of mentions that mention word will be classified as lead or if the headline starts with the N(noun) tag the whole noun phrase starting from the first word of the headline will be the lead else the lead will be labeled as ‘unknown’.

Example:

- በደቡብ ክልል ባለፉት 11 ወራት ከ11 ነጥብ 6 ቢሊየን ብር በላይ ገቢ ተሰበሰበ

  From this title ‘በደቡብ ክልል’ is the lead since the sentence starts with PREP ‘በ’ and the next word is “RECOGNIZED LOCATION”

- ሐምዛ አብዲ ባሬ የሶማሊያ ጠቅላይ ሚኒስትር ሆነው ተሾሙ

  From this title, ‘ሐምዛ አብዲ ባሬ’ is the lead since the sentence starts with a word that has a “RECOGNIZED PERSON” tag

```
Input: Head line or title of the news
Start
initialize lead to empty
pass the head line through tokenize sentence, POS and NER tagger
then
If word tag starts with('RECOGNIZED')
  Add the word lead
Else if word tag starts with('NC')
  Add the next word to lead
Else if word tag starts with('N')
   See the next word
   If next word tag start with("NS","NP")
   Add the two words to lead
End
Output: lead
```

FIGURE 11 - Lead algorithm

4.3.1.2.6. Relations

Relations is the final sub-component of Language-based Learning. In this stage from every simple sub-sentence, single information will be drowned if it meets a certain rule and add to the list of final relations. The input of this stage from the above sub-components will have not only the POS and the NER tags but also the 'RELATION',' PHRASE_N', and 'RECOGNIZED' tags. The Relations sub-component will apply rules to find individual relation-based news from each sub-sentence by examining their word tags. The detailed implementation will be discussed in the next chapter.

Example :

- የቢሮው ኃላፊ ወይዘሮ ዘይቱና ኢብራሂም እንደገለጹት በበጀት ዓመቱ 13 ነጥብ 1 ቢሊየን ብር ለመሰብሰብ ታቅዶ ወደ ስራ የተገባ ሲሆን ባለፉት 11 ወራት ከመደበኛ እና ከማዘጋጃ ቤት ከ11 ነጥብ 6 ቢሊየን በላይ ብር ገቢ መሰብሰብ ተችሏል

  From the above sub sentence the following are an individual relations or facts:

1. የቢሮው ኃላፊ ወይዘሮ ዘይቱና ኢብራሂም
2. በበጀት ዓመቱ 13 ነጥብ 1 ቢሊየን ብር ለመሰብሰብ እንደታቀደ
3. በ 11 ወራት ከመደበኛ እና ከማዘጋጃ ቤት ከ11 ነጥብ 6 ቢሊየን በላይ ብር ገቢ መሰብሰብ እንደተቻለ

By combining the final output of this sub-component with other sub-components output a list of mentions, a list of individual relation-based information, and a lead will be passed as input for the next component.

4.3.1.3. The Extractor Postprocessor

It's the last part of the Extractor phase, and it's where the different attributes retrieved from the news text are formatted according to a specified format. The Extractor Postprocessor component's main job is to organize the extracted data's format so that it may be used in the next phase.

4.3.1.3.1. Formatter

In both reporting and editing, a news report follows a specific pattern. Every news organization may have its writing style, but all news at the basic level follows a standard pattern that is separated into numerous sections (Mckane, 2014). According to (Tereszkiewicz, 2012)a news story or report is divided into 5 parts namely; Headline, Byline, Lead, Body, and Ending.

In this, by using the outputs of the previous component as an input we are going into 5 categories:

1. Headline – explains the narrative and the title under which the news is published
2. Byline – Identifies who wrote the story and where it was published.
3. Lead – tells what or who is the subject of the news
4. Facts – contains a list of individual relational information found on the news
5. Mentions – show all people, places, and organizations mentioned in the provided news

After all, the process completes the output of the Extractor phase from the proposed AIERL model is a categorized extracted information in the form that is explained in the formatter sub-component.

Example:

- የአሉቶ የከርሰምድር እንፋሎት ፍለጋ ቁፋሮ ከተጠናቀቀባቸው አራት ጉድጓዶች መካከል አንደኛው የምርት ሙከራ በይፋ መጀመሩ ተገለጸ።የፕሮጀክቱ ሥራ አስኪያጅ መሳይ ፍቃዱ እንደገለፁት የአምስተኛው ጉድጓድ ቁፋሮ ሰባት መቶ ሜትር ላይ ደርሷል።ግንቦት 21 ቀን 2013 ዓ.ም በይፋ የተጀመረው የአሉቶ ላንጋኖ የከርሰምድር እንፋሎት ማስፋፊያ ፕሮጀክት ስድስተኛውን የእንፋሎት ፍለጋ ጉድጓድ ቁፋሮ ለማስጀመር የመቆፈሪያ ማሽን ወደቦታው በማንቀሳቀስ የመትከል ሥራ እየተከናወነ መሆኑን ጠቁመዋል።አንድ ጉድጓድ ለመቆፈር በአማካይ እስከ 75 ቀናት ይፈጃል ያሉት ሥራ አስኪያጁ ቁፋሯቸው ከተጠናቀቁት አራት ጉድጓዶች መካከል የሁለተኛውን ጉድጓድ የምርት

ሙከራ ሥራ ከሁለት ሳምንታት በኋላ ይጀመራል ብለዋል፡፡የእንፋሎት ፍለጋ የጉድጓድ ቁፋሮው ወደ ከርሰምድር እስከ 3 ሺሕ ሜትር ጥልቀት የሚኖረው ሲሆን ፕሮጀክቱ እስከ 70 ሜጋ ዋት ኤሌክትሪክ ለማመንጨት ታስቦ እየተሰራ መሆኑን ከኢትዮጵያ ኤሌክትሪክ ኃይል ያገኘነው መረጃ ያመለክታል፡፡

This is a news copied from Walta tv business news page. If this specific news goes to the first phase of AIERL which is The Extractor the output will be:

'Head_Line': 'unknown',

'By_line': 'unknown',

'Lead': 'unknown',

'Facts': {

1: 'የአሉቶ የከርሰምድር እንፋሎት ፍለጋ ቁፋሮ ከተጠናቀቀባቸው አራት ጉድጓዶች መካከል አንደኛው የምርት ሙከራ በይፋ መጀመሩ ተገለጸ',

2: 'የፕሮጀክቱ ሥራ አስኪያጅ መሳይ ፍቃዱ',

3: 'የአምስተኛው ጉድጓድ ቁፋሮ ሰባት መቶ ሜትር ላይ ደርሷል',

4: 'የአሉቶ ላንጋኖ የከርሰምድር እንፋሎት ማስፋፊያ ፕሮጀክት ስድስተኛውን የእንፋሎት ፍለጋ ጉድጓድ ቁፋሮ ለማስጀመር የመቆፈሪያ ማሽን ወደቦታው በማንቀሳቀስ የመትከል ሥራ እየተከናወነ መሆኑን ጠቁመዋል',

5: 'አንድ ጉድጓድ ለመቆፈር በአማካይ እስከ 75 ቀናት ይፈጃል',

6: 'አራት ጉድጓዶች መካከል የሁለተኛውን ጉድጓድ የምርት ሙከራ ሥራ ከሁለት ሳምንታት በኋላ ይጀመራል ብለዋል',

7: 'የእንፋሎት ፍለጋ የጉድጓድ ቁፋሮው ወደ ከርሰምድር እስከ 3 ሺሕ ሜትር ጥልቀት የሚኖረው'

8: 'ፕሮጀክቱ እስከ 70 ሜጋ ዋት ኤሌክትሪክ ለማመንጨት ታስቦ እየተሰራ መሆኑን'},

'Mentions': ['መሳይ ፍቃዱ', 'የአሉቶ ላንጋኖ', 'ከኢትዮጵያ ኤሌክትሪክ ኃይል']

Because the news from the site only contains the body part it's impossible for the extractor to extract the headline of the news and since the lead is extracted from headline both categories have the value of 'unknown' and due to the fact that the extractor component doesn't know where the news is given from the byline is also 'unknown' at this phase.

4.3.2. The RL

The Markov decision process is an outcome prediction model. The model attempts to predict an outcome based on information provided by the state. The Markov decision process takes into account the characteristics of actions and motivations. The decision maker may choose to take an action available in the state at any time during the process (Martijn van Otterlo & Marco Wiering ,

2012) The MDP framework allows us to dynamically incorporate entity predictions while also allowing us to select the type of articles from which to extract. Based on the general Reinforcement learning architecture we construct the architecture for this phase and it's shown in the below figure [figure 12]

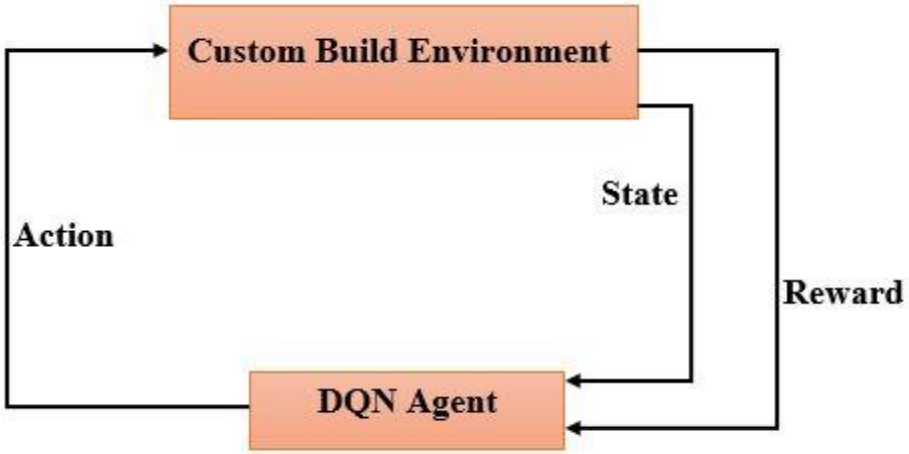


FIGURE 12 - General Architecture of Reinforcement learning

4.3.2.1. Custom Build Environment

We created a custom environment in which the agent can observe and learn. The environment was created with the OpenAI Gym library. Since The custom environment we build inherits from the OpenAI Gym environment it is a class that has a list of arranged methods inside.

In the class constructor, we define our observation and action space where the action space is a representation of all possible actions the agent will take (the actions will be discussed in detail below in the Action section). To define the action space from the gym.space class we use the Discrete space. For the observation space which also means state representation in other papers. We use the box, dict, and tuple spaces from gym.spaces to represent the states in the environment. The environment observation space is a direct representation of all possible states our environment could have (the states will be discussed in detail below)

We have a compulsory method to define called 'step' in this environment class where it takes action and state as an input argument. In this method, we define what every action would change in the state of the environment and the reward for that action then returns the current state and reward. Another mandatory function in this class is a 'reset' method which clears the environment and returns an initial observation.

4.3.2.2. State

Each state in our environment is created based on two sets of output from The Extractor phase. The two sets are the current set and the new set. The current set (we use the word set to describe a categorized extracted information in the form of Headline, by line, Lead, Facts, and Mentions) is extracted from the provided news using phase one of our AIERL model. The second set is also extracted information which is extracted using the extractor but the news is from selected authorized news sites scraped using a query formed from the given news. A state in our environment consists of the following entities:

- The current set
- The new set
- Comparison of facts
- Match

Both the current and the new set have a category called facts, the comparison of facts entity of a state is a comparison of those facts. The match is the similarity measure of each category in the two sets, we will compare the current set headline with the new set headline if they are similar or not, and likewise for all the 5 categories.

Example:

- በደቡብ ክልል ባለፉት 11 ወራት ከ11 ቢሊየን ብር በላይ ገቢ መሰብሰቡን የክልሉ ገቢዎች ቢሮ አስታወቀ፡፡ በበጀት ዓመቱ 13 ነጥብ 1 ቢሊየን ብር ለመሰብሰብ ታቅዶ ወደ ስራ የተገባ ሲሆን÷ ባለፉት 11 ወራት ከመደበኛ እና ከማዘጋጃ ቤት ከ11 ነጥብ 6 ቢሊየን በላይ ብር ገቢ መሰብሰብ ተችሏል። እንደ ኃላፊዋ ገለጻ በኡዲት ፣ የግብር ከፋዩን ማህደር በመመርመር የተገኙ የአሰራር ግድፈቶችን ለማረም ጥረት የተደረገ ሲሆን የህግ ጥሰት በፈጸሙ ሰራተኞች ላይ ህጋዊ እርምጃ መወሰዱን ጠቁመዋል። የገቢ አሰባሰቡ ከእቅዱ አንጻር የተሻለ አፈጻጸም ቢያሳይም ካለው የገቢ አሰባሰብ አቅም አንጻር የሚቀሩ ጉዳዮች መኖራቸውን ተገንዝበናል ብለዋል። ከህግ ማስከበር ጋር በተያያዘ በፈጻሚው እና በግብር ከፋዩ ዘንድ የሚስተዋሉ የስነ ምግባር ጥሰቶችን በመለየት የእርምት እርምጃ ተወስዷል ያሉት ኃላፊዋ በተለይ ሀሰተኛ ደረሰኝ ማቅረብ እና ግብር መሰወር ላይ ያሉ ችግሮችን የማረም ስራ እየተሰራ መሆኑን ጠቁመዋል። በክልሉ ከ1 ሺህ በላይ ሀሰተኛ ደረሰኝ የተገኘ ሲሆን÷ ከ187 ሚሊየን ብር በላይ በህገ ወጥ መንገድ ግብይት እንደተፈጸመ መናገራቸውን ከክልሉ መንግስት ኮሙኒኬሽን ያገኘነው መረጃ ያመላክታል፡፡

  From this news the current set will be:

  'Head_Line': 'unknown',

  'By_line': 'unknown',

'Lead': 'unknown',

'Facts': {1: 'ባለፉት 11 ወራት ከ 11 ቢሊየን ብር በላይ ገቢ መሰብሰቡን',

2: 'የክልሉ ገቢዎች ቢሮ አስታወቀ',

3: 'በበጀት ዓመቱ 13 ነጥብ 1 ቢሊየን ብር ለመሰብሰብ ታቅዶ',

4: '11 ወራት ከመደበኛ እና ከማዘጋጃ ቤት ከ 11 ነጥብ 6 ቢሊየን በላይ ብር ገቢ መሰብሰብ ተችሏል',

5: 'የህግ ጥሰት በፈጸሙ ሰራተኞች ላይ ህጋዊ እርምጃ መወሰዱን ጠቁመዋል',

6: 'የሚቀሩ ጉዳዮች መኖራቸውን ተገንዝበናል ብለዋል',

7: 'በፈጻሚው እና በግብር ከፋዩ ዘንድ የሚስተዋሉ የስነ ምግባር ጥሰቶችን በመለየት የእርምት እርምጃ ተወስዷል',

8: 'ሀሰተኛ ደረሰኝ ማቅረብ እና ግብር መሰወር',

9: 'ላይ ያሉ ችግሮችን የማረም ስራ እየተሰራ መሆኑን ጠቁመዋል',

10: 'በክልሉ ከ 1 ሺህ በላይ ሀሰተኛ ደረሰኝ የተገኘ ሲሆን ከ 187 ሚሊየን ብር በላይ በህገ ወጥ መንገድ ግብይት እንደተፈጸመ መናገራቸውን',

11: 'ከክልሉ መንግስት ኮሙኒኬሽን ያገኘነው መረጃ ያመላክታል'},

'Mentions': ['በደቡብ ክልል']}

The new set after scraping (we will see in detail about the scraping process) from the web will be:

'Head_Line': 'በደቡብ ክልል ባለፉት 11 ወራት ከ11 ነጥብ 6 ቢሊየን ብር በላይ ገቢ ተሰበሰበ',

'By_line': 'fana_tv',

'Lead': 'በደቡብ ክልል',

'Facts': {1: 'ባለፉት 11 ወራት ከ 11 ነጥብ 6 ቢሊየን ብር በላይ ገቢ መሰብሰቡን',

2: 'የክልሉ ገቢዎች ቢሮ አስታወቀ',

3: 'የቢሮው ኃላፊ ወይዘሮ ዘይቱና ኢብራሂም',

4: 'በበጀት ዓመቱ 13 ነጥብ 1 ቢሊየን ብር ለመሰብሰብ ታቅዶ',

5: '11 ወራት ከመደበኛ እና ከማዘጋጃ ቤት ከ 11 ነጥብ 6 ቢሊየን በላይ ብር ገቢ መሰብሰብ ተችሏል',

6: 'የህግ ጥሰት በፈጸሙ ሰራተኞች ላይ ህጋዊ እርምጃ መወሰዱን ጠቁመዋል',

7: 'የሚቀሩ ጉዳዮች መኖራቸውን ተገንዝበናል ብለዋል',

8: 'በፈጻሚው እና በግብር ከፋዩ ዘንድ የሚስተዋሉ የስነ ምግባር ጥሰቶችን በመለየት የእርምት እርምጃ ተወስዷል',

9: 'ሀሰተኛ ደረሰኝ ማቅረብ እና ግብር መሰወር',

10: 'ላይ ያሉ ችግሮችን የማረም ስራ እየተሰራ መሆኑን ጠቁመዋል',

11: 'በክልሉ ከ 1 ሺህ በላይ ሀሰተኛ ደረሰኝ የተገኘ ሲሆን ከ 187 ሚሊየን ብር በላይ በህገ ወጥ መንገድ ግብይት እንደተፈጸመ መናገራቸውን',

12: 'ከክልሉ መንግስት ኮሙኒኬሽን ያገኘነው መረጃ ያመላክታል',

13: 'በቀጣዩ 2015 በጀት ዓመት 15 ነጥብ 665 ቢሊየን ብር ገቢ ለመሰብሰብ ማቀዱ ታውቋል'},

'Mentions': ['አዲስ አበባ', 'በደቡብ ክልል', 'ወይዘሮ ዘይቱና ኢብራሂም']}

Using this two sets the current and the new the formed state will be as follows:

{'the current set':

{'Head_Line': 'unknown', 'By_line': 'unknown', 'Lead': 'unknown', 'Facts': {1: 'ባለፉት 11 ወራት ከ 11 ቢሊየን ብር በላይ ገቢ መሰብሰቡን', 2: 'የክልሉ ገቢዎች ቢሮ አስታወቀ', 3: 'በበጀት ዓመቱ 13 ነጥብ 1 ቢሊየን ብር ለመሰብሰብ ታቅዶ', 4: '11 ወራት ከመደበኛ እና ከማዘጋጃ ቤት ከ 11 ነጥብ 6 ቢሊየን በላይ ብር ገቢ መሰብሰብ ተችሏል', 5: 'የህግ ጥሰት በፈጸሙ ሰራተኞች ላይ ህጋዊ እርምጃ መወሰዱን ጠቁመዋል', 6: 'የሚቀሩ ጉዳዮች መኖራቸውን ተገንዝበናል ብለዋል', 7: 'በፈጻሚው እና በግብር ከፋዩ ዘንድ የሚስተዋሉ የስነ ምግባር ጥሰቶችን በመለየት የእርምት እርምጃ ተወስዷል', 8: 'ሀሰተኛ ደረሰኝ ማቅረብ', 9: 'እና ግብር መሰወር', 10: 'ላይ ያሉ ችግሮችን የማረም ስራ እየተሰራ መሆኑን ጠቁመዋል', 11: 'በክልሉ ከ 1 ሺህ በላይ ሀሰተኛ ደረሰኝ የተገኘ ሲሆን ከ 187 ሚሊየን ብር በላይ በህገ ወጥ መንገድ ግብይት እንደተፈጸመ መናገራቸውን', 12: 'ከክልሉ መንግስት ኮሙኒኬሽን ያገኘነው መረጃ ያመላክታል'}, 'Mentions': ['በደቡብ ክልል']},

'the new set':

` {'Head_Line': 'በደቡብ ክልል ባለፉት 11 ወራት ከ11 ነጥብ 6 ቢሊየን ብር በላይ ገቢ ተሰበሰበ', 'By_line': 'fana_tv', 'Lead': 'በደቡብ ክልል', 'Facts': {1: 'ባለፉት 11 ወራት ከ 11 ነጥብ 6 ቢሊየን ብር በላይ ገቢ መሰብሰቡን', 2: 'የክልሉ ገቢዎች ቢሮ አስታወቀ', 3: 'የቢሮው ኃላፊ ወይዘሮ ዘይቱና ኢብራሂም', 4: 'በበጀት ዓመቱ 13 ነጥብ 1 ቢሊየን ብር ለመሰብሰብ ታቅዶ', 5: '11 ወራት ከመደበኛ እና ከማዘጋጃ ቤት ከ 11 ነጥብ 6 ቢሊየን በላይ ብር ገቢ መሰብሰብ ተችሏል', 6: 'የህግ ጥሰት በፈጸሙ ሰራተኞች ላይ ህጋዊ እርምጃ መወሰዱን ጠቁመዋል', 7: 'የሚቀሩ ጉዳዮች መኖራቸውን ተገንዝበናል ብለዋል', 8: 'በፈጻሚው እና በግብር ከፋዩ ዘንድ የሚስተዋሉ የስነ ምግባር ጥሰቶችን በመለየት የእርምት እርምጃ ተወስዷል', 9: 'ሀሰተኛ ደረሰኝ ማቅረብ', 10: 'እና ግብር መሰወር', 11: 'ላይ ያሉ ችግሮችን የማረም ስራ እየተሰራ መሆኑን ጠቁመዋል', 12: 'በክልሉ ከ 1 ሺህ በላይ ሀሰተኛ ደረሰኝ የተገኘ ሲሆን ከ 187 ሚሊየን ብር በላይ በህገ ወጥ መንገድ ግብይት እንደተፈጸመ መናገራቸውን', 13: 'ከክልሉ መንግስት ኮሙኒኬሽን ያገኘነው መረጃ ያመላክታል', 14: 'በቀጣዩ 2015 በጀት ዓመት 15 ነጥብ 665 ቢሊየን ብር ገቢ ለመሰብሰብ ማቀዱ ታውቋል'}, 'Mentions': ['አዲስ አበባ', 'በደቡብ ክልል', 'ወይዘሮ ዘይቱና ኢብራሂም']},

'match':

[0.26, 0.6, 0.4, 0.92, 0.59],

'comparisions_of_facts':

[0.93, 1.0, 0.22, 0.55, 0.34, 0.27, 0.26, 0.14, 0.34, 0.26, 0.25, 0.27, 0.0, 0.0]}

In the above example, the new set which is scraped from the web using a query formed from the current set has different contents of information but both are related news.

4.3.2.3. Action

At each stage, the agent must take a compromise decision by using and observing the state. The decision on the newly extracted values can be of one of four types:

1. accept the value of a specific entity (take some from the new)
2. accept all entity values (take all from the new),
3. reject all values
4. stop

In cases 1-3, the agent inspects more news by preparing a query to scrape from the web, while the episode ends if a stop action (4) is selected.

The agent observes the match and comparison of facts entities to decide on which action to take; if all match values do not equal to 1.0 the agent will take action 2, if the match values of the first 3 are equal to 1.0 and the 4^{th} is not equaled to 1.0 the agent will take action 1, and if all match entity values equal to 1.0 which means both sets are similar the agent takes action 3 or 4 based on the comparison_of_facts values. If all comparison_of_facts value equal to 1.0 the agent will take action 4 else action 3.

4.3.2.4. The Agent

To determine which action (A) to perform in the state (S), the agent typically employs a state action value function Q (S, A). We use a deep Q network because our problem has a continuous state space (DQN). The DQN, which uses a deep neural network to approximate the Q-function, has been shown to learn better value functions than linear approximators (Narasimhan, Adam Yala, & Regina Barzilay, 2016) and can capture non-linear interactions between the various pieces of information in our state. The agent for this research work is a DQN agent from OpenAi Gym library.

The agent uses a DQN consisting of a sequential neural network model with one rectified linear unit (ReLU) (24 hidden units) of the input layer followed by another rectified linear unit (24 hidden units), along with one separate linear output layer.

The overall implementation will be discussed in the next chapter.

4.4. Other Components

The AIERL uses additional components apart from the two phases which are responsible for providing related news from other sources.

4.4.1. Query preparation and searching the web

From the extracted information of the current set by using the fact category each fact will act as a query to retrieve the most related news for the given news text. From one news we may have a list of queries.

Using the python beautiful soup library and the list of queries it's going to search a list of links that have news for the specific query from google. To be in the list of links the found news link must be from fana tv, walta tv, or Ebc or the link will not be selected because those 3 Ethiopian broadcasting sites are selected as trusted sites for this research work and they are authorized news sites of Ethiopia. Using each selected link, a set of news title, posted site and posted by if available and a text news body will be scraped and stored for the next news selection process. The below figure [figure 13] shows the algorithm for this process.

```
Input: a list of queries
Start
initialize set of scraped news to empty
initialize links to empty
for q in queries
    response = search in google
    if response is from fana, walta or ebc
       add response to links
for l in links
    based on their website structure of the site
    scrapped = scrape title, posted by and text news
    add scrapped to set of scraped news
End
Output: set of scraped news
```

FIGURE 13 - Algorithm for scraping process

4.4.2. Selecting News

This process is to select one news that is the most related and similar from the scraped related set of news, to do that this process uses cosine similarity by calculating the cosine similarity measure of each scraped news with the first given input news text and will select the one with maximum cosine similarity measure to pass it to the next process.

CHAPTER FIVE

5. IMPLEMENTATION OF AIERL

5.1. Overview

We have explained the implementation of the suggested study in this section. The section discusses how each component of the two AIERL phases and other components has been implemented.

5.2. Implementation of the Extraction phase

As mentioned in the previous chapter (RESEARCH DESIGN) The Extractor and RL are the two phases of the AIERL. The implementation of the Extractor phase will be briefly discussed in this section.

5.2.1. Importing libraries

The first step in implementing the current study is to import Python library packages. Importing the necessary library is a must for using that specific library throughout our program. As a result, we have imported the libraries listed below, which will be used for the model's Extractor phase.

```
#importing the required libraries for the Extractor phase
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
from thefuzz import fuzz
import itertools
import re
import gensim
import pandas as pd
from flair.data import Sentence
from flair.models import SequenceTagger
from amseg.amharicSegmenter import AmharicSegmenter
```

FIGURE 14 - libraries used in the Extractor phase

5.2.2. Implementation of the Extractor preprocessor

The Extractor preprocessor contains two sub-components which are tokenizer and POS/NER tagging in which the tokenizer is done using amseg free Amharic segmenter and for POS/NER tagging we have used publicly available flair-based models.

The Amharic Segmenter(amseg) can be installed as pip install amseg. As per the amseg guideline, we load the necessary classes for the tokenizer.

```
#necessary classes for amseg tokenizer
sent_punct = []
word_punct = []
segmenter = AmharicSegmenter(sent_punct,word_punct)
```

FIGURE 15 - classes for the amseg

After downloading the free available POS and NER models we then load them into the python program.

```
#importing existing free models for pos and ner
NERmodel = SequenceTagger.load('NERfinal-model.pt')
POSmodel = SequenceTagger.load('POSfinal-model.pt')
```

FIGURE 16 - loading the models

The below figure (figure 17) shows how the models are used to tag Amharic news with the appropriate POS and NER tag after dividing the news into sentences.

```
#a function that returns a sentences with tagged pos and ner values
def load_sentences(input):
    sentencewithtagedwords={}
    counter=1
    sentencesB=segmenter.tokenize_sentence(input)
    for sents in sentencesB:
        sentforb=re.sub(r'[~\,\.\-\=\+\:\=\!\;\÷\+\÷\⁝\※\⁜\⁝\÷"\"\"\(]',' ',sents)
        sentforb=Sentence(sentforb)
        NERmodel.predict(sentforb)
        POSmodel.predict(sentforb)
        strw=sentforb.to_tagged_string()
        words=strw.split('>')
        tagedWords=[]
        for w in words:
            stringword=str(w).strip()
            stringword=re.sub(r'<','',stringword)
            if not stringword == '':
                tagedWords.append(stringword)
        sentencewithtagedwords[counter]=tagedWords
        counter+=1
    return sentencewithtagedwords
```

FIGURE 17 - POS and NER tagger

5.2.3. Implementation of the language-based learning

The language-based learning contains six sub-components namely Sub sentencer, noun-phrases, verb-phrases, mentions, lead, and Relations.

5.2.3.1. Implementation of the sub sentencer

The Sub sentencer converts difficult and convoluted sentences into simple sentences using a set of hand-crafted grammatical rules. The guidelines are based on Amharic sentence structure and POS tags as discussed in detail in chapter 4(Sub sentencer).

```
#function that returns a list of subsentences from a list of sentences
def sub_sentencer(sentence_with_tagged_words):
    list_of_words=[]
    for sents in sentence_with_tagged_words:
      for s in sentence_with_tagged_words[sents]:
         list_of_words.append(s.strip())
    sub_sents=[]
    lengh=len(list_of_words)
    sub_sentences_dict={}
    counter=1
    for word_from_sub in list_of_words:
        word_index=list_of_words.index(word_from_sub)
        word_splited=word_from_sub.split(' ')
        sub_sents.append(word_from_sub)
        if (word_splited[1].startswith(('V','PUNC','AUX'))) and not word_splited[1].startswith(('VRE','VP')):
            if word_index+1 < lengh:
                next_word=list_of_words[word_index+1]
                nws=next_word.split(' ')
                if not (nws[1].startswith(('VP','PUNC','AUX','CON','VN')) or (nws[1] == 'V' or nws[1] == 'VS')):
                    sub_sentences_dict[counter]=sub_sents
                    counter+=1
                    sub_sents=[]
            else:
                sub_sentences_dict[counter]=sub_sents
                counter+=1
                break
        elif word_index+1 == lengh:
            sub_sentences_dict[counter]=sub_sents
            counter+=1
            sub_sents=[]
    return sub_sentences_dict
```

FIGURE 18 - implementation of sub sentencer

5.2.3.2. Implementation of noun-phrases

A simple sentence consists of at least one general noun phrase, the subject, and the object, each of which may be made up of conjunctions, prepositions, pronouns, adjectives, and nouns. This noun phrase is in charge of identifying noun phrases and marking them with the tag "PHRASE N."

```
#this function will catagorize nouns together to create a nounphrase based on some rules
def noun_phraser(sub_sentences_dict):
    phrased_sub_Sentences={}
    new_sub_sentence=[]
    #for each subsentence
    for i in sub_sentences_dict:
        new_sub_sentence=[]
        phrased_word=''
        #for each words in each subsentences
        for words in sub_sentences_dict[i]:
            word_splited=words.split(" ")
            word_index=sub_sentences_dict[i].index(words)
            #if word tag is of the following
            if word_splited[1].startswith(('N','CONJ','PRE','ADJ','PRON')):
                phrased_word=phrased_word+' '+word_splited[0]
                next_word=sub_sentences_dict[i][word_index+1]
                nws=next_word.split(" ")
                if not nws[1].startswith(('N','CONJ','PRE','ADJ','PRON')):
                    new_sub_sentence.append("PHRASE_N "+(phrased_word).strip())
                    phrased_word=''
            else:
                new_sub_sentence.append(words)
        phrased_sub_Sentences[i]=new_sub_sentence
    return phrased_sub_Sentences
```

FIGURE 19 - implementation of noun-phrase

5.2.3.3. Implementation of verb-phrases

The relationship between the noun phrases (Subject and Objects) in a simple sentence is described by the verb, and occasionally in an Amharic text, a simple sentence may include more than one verb defining a single relation. The verb phrase is in charge of identifying these verbs, creating a verb phrase that explains the connection, and tagging it as "V PHRASE WORD."

The implementation of the verb-phrases sub-component is shown in the figure below (figure 20).

```
#a function that will find a verb_based phrases from the list of sub sentences
def verb_based_relationer(phrased_sub_Sentences):
    NV_phrased_sub_Sentences={}
    NV_phrased=[]
    for i in phrased_sub_Sentences:
        length=len(phrased_sub_Sentences[i])
        V_PHRASE_WORD=''
        NV_phrased=[]
        for words in phrased_sub_Sentences[i]:
            word_index=phrased_sub_Sentences[i].index(words)
            word_splited=words.split(' ')
            if word_splited[0].startswith('RECOGNIZED_TT'):
                continue
            if word_splited[1].startswith(('V','PUNC','AUX')):
                V_PHRASE_WORD+=' '+word_splited[0]
                if not word_index+1==length:
                    n_W=phrased_sub_Sentences[i][word_index+1]
                    nw_sp=n_W.split(" ")
                    if not nw_sp[1].startswith(('V','PUNC','AUX')):
                        if word_splited[1].startswith(('VREL','VP')):
                            NV_phrased.append("N_RELATION "+(V_PHRASE_WORD).strip())
                            V_PHRASE_WORD=''
                        else:
                            NV_phrased.append("RELATION "+(V_PHRASE_WORD).strip())
                            V_PHRASE_WORD=''
                else:
                    NV_phrased.append("RELATION "+(V_PHRASE_WORD).strip())
                    V_PHRASE_WORD=''
            else:
                NV_phrased.append(words)
        NV_phrased_sub_Sentences[i]=NV_phrased
    return NV_phrased_sub_Sentences
```

FIGURE 20 - an implementation for verb phrase

5.2.3.4. Implementation of mentions

As described in the previous chapter (Mentions) this sub-component will archive all references to places, people, and organizations with the NER tag. To perform the specified task, the mentions use two separate functions namely entity recognizer and mentions.

The entity recognizer uses the NER tag of each word to tag the word as “RECOGNIZED”. The entity recognizer handles if the recognized word has a consecutive word that will go with the previous one. It will find these kinds of words and use them together as one recognized word. The output of This function will act as an input for the next function which is mentions that selects all words with the “RECOGNIZED” tag and by replacing some unnecessary characters it stores all mentions in a list.

Implementations of the two functions are shown in the below figures (figure 21 and figure 22).

```
#this function will return a dictionary containing recognized location,person and organiztions
def entity_recognizer(phrased_sub_Sentences):
    recognized_sub_Sentences={}
    recognized=[]
    V_PHRASE_WORD=''
    for i in phrased_sub_Sentences:
        V_PHRASE_WORD=''
        type=''
        recognized=[]
        for words in phrased_sub_Sentences[i]:
            word_index=phrased_sub_Sentences[i].index(words)
            word_splited=words.split(' ')
            if word_splited[1].startswith(('B-','I-')):
                type=str(word_splited[1])[2:4]
                if word_splited[1].startswith('B-'):
                    V_PHRASE_WORD+=' '+word_splited[0]
                    next_word=phrased_sub_Sentences[i][word_index+1]
                    nws=next_word.split(' ')
                    if not nws[1].startswith('I-'):
                        recognized.insert(word_index,"RECOGNIZED_"+type+V_PHRASE_WORD)
                elif word_splited[1].startswith('I-'):
                    V_PHRASE_WORD+=' '+word_splited[0]
                    next_word=phrased_sub_Sentences[i][word_index+1]
                    nws=next_word.split(' ')
                    if not nws[1].startswith(('I-')):
                        recognized.insert(word_index,"RECOGNIZED_"+type+V_PHRASE_WORD)
                        V_PHRASE_WORD=''
            else:
                recognized.append(words)
        recognized_sub_Sentences[i]=recognized
    return recognized_sub_Sentences
```

FIGURE 21 - entity recognizer function implementation

```
#this function returns a list of mentions that are recognized in the entity_recognizer
def mentions(NV_phrased_sub_Sentences):
    all_mentions=[]
    for i in NV_phrased_sub_Sentences:
        for words in NV_phrased_sub_Sentences[i]:
            word_splited=words.split(' ')
            length=len(word_splited)
            if word_splited[0].startswith(('RECOGNIZED_P','RECOGNIZED_O','RECOGNIZED_L')):
                string=str(word_splited[1:length])
                fstring=string.replace(',','').replace("'",'').replace('[','').replace(']','')
                if not fstring in all_mentions:
                    all_mentions.append(fstring)
    return all_mentions
```

FIGURE 22 - an implementation of a mention function

5.2.3.5. Implementation of lead

If the headline begins with the PREP (preposition) tag and the subsequent word is listed among mentions, that mention word will be classified as the lead; alternatively, if the headline begins with the N (noun) tag, the entire noun phrase beginning with the first word of the headline will be the lead; otherwise, the lead will be labeled as "unknown."

```
# this function finds the lead roll for the news from the news article title
def lead(head_line):
    lead_text=''
    ls=load_sentences(head_line)
    er=entity_recognizer(ls)
    for i in er:
        for word in er[i]:
            word_splited=word.split(" ",1)
            if word_splited[0].startswith(('RECOGNIZED_OR','RECOGNIZED_LO','RECOGNIZED_PE')):
                lead_text=str(word_splited[1])
                break
            elif word_splited[1].startswith("NC"):
                word_index=er[i].index(word)
                next_word=er[i][word_index+1]
                nws=next_word.split(" ",1)
                lead_text=str(word_splited[0]+" "+nws[0])
                break
            elif word_splited[1].startswith("NP"):
                word_index=er[i].index(word)
                next_word=er[i][word_index+1]
                nws=next_word.split(" ",1)
                lead_text=str(word_splited[0]+" "+nws[0])
                break
            elif word_splited[1]==("N"):
                word_index=er[i].index(word)
                if not word_index > len(er[i]):
                    next_word=er[i][word_index+1]
                    nws=next_word.split(" ",1)
                    if nws[1].startswith(("NS","NP")):
                        lead_text=str(word_splited[0]+" "+nws[0])
                        break
                    elif nws[1].startswith("V"):
                        lead_text=str(word_splited[0])
                        break
                    elif nws[1]== ("N"):
                        lead_text=str(word_splited[0]+" "+nws[0])
                        break
            else:
                lead_text='unknown'
    return lead_text
```

FIGURE 23 - Implementation of the lead subcomponent

5.2.3.6. Implementation of relations

The last component of language-based learning is Relations. At this point, if a simple subordinate clause satisfies a predetermined rule, it will be drowned and included in the list of final relations. To perform this task this component uses two functions. Since words in one sub-sentence have different kinds of tags at this stage The first function (fact_finder) will work with all kinds of tags such as NER tag, Noun phrase tag, mention tag, and verb phrase tag to make a relation tag with ‘s_FACT” and “FACT’ tag. The second function (final_fact) uses the output of the first function to classify as a relation or not.

Implementations of the two functions are shown in the below figures (figure 24 and figure 25).

```
#this function find relations and facts from a given news to be inputed for the next function
def fact_finder(NV_phrased_sub_Sentences):
    facts={}
    fact=[]
    indexs=[]
    for i in NV_phrased_sub_Sentences:
        fact=[]
        indexs=[]
        for words in NV_phrased_sub_Sentences[i]:
            word_index=NV_phrased_sub_Sentences[i].index(words)
            word_splited=words.split(' ',1)
            if not word_index in indexs:
                if word_splited[0].startswith('PHRASE_N'):
                    n_W=NV_phrased_sub_Sentences[i][word_index+1]
                    nw_sp=n_W.split(" ",1)
                    if nw_sp[0].startswith('RELAT'):
                        fact.append("FACT "+word_splited[1]+" "+nw_sp[1])
                        indexs.append(word_index+1)
                    elif nw_sp[0].startswith('RECOGNIZED_PE'):
                        fact.append("FACT "+word_splited[1]+" "+nw_sp[1])
                        indexs.append(word_index+1)
                    elif nw_sp[0].startswith('N_RE'):
                        fact.append("s_FACT "+word_splited[1]+" "+nw_sp[1])
                        indexs.append(word_index+1)
                    else:
                        fact.append(words)
                else:
                    fact.append(words)
        facts[i]=fact
    return facts
```

FIGURE 24 - implementation of fact_finder function

```python
#this function takes facts as input and returns a list of relation found in the news
def final_fact(facts):
    list_of_facts={}
    f_facts={}
    f_fact=[]
    indexs=[]
    counter=1
    for i in facts:
        f_fact=[]
        indexs=[]
        for words in facts[i]:
            word_index=facts[i].index(words)
            word_splited=words.split(' ',1)
            if not word_index in indexs:
                if word_splited[0].startswith('RECOGNIZED'):
                    n_W=facts[i][word_index+1]
                    nw_sp=n_W.split(" ",1)
                    if nw_sp[0].startswith('FACT'):
                        f_fact.append("FACT "+word_splited[1]+" "+nw_sp[1])
                        indexs.append(word_index+1)
                    else:
                        f_fact.append(words)
                elif word_splited[0].startswith('s_FACT'):
                    n_W=facts[i][word_index+1]
                    nw_sp=n_W.split(" ",1)
                    if nw_sp[0].startswith('FACT'):
                        f_fact.append("FACT "+word_splited[1]+" "+nw_sp[1])
                        indexs.append(word_index+1)
                    else:
                        f_fact.append(words)
                else:
                    f_fact.append(words)
        f_facts[i]=f_fact
    for i in f_facts:
        for f in f_facts[i]:
            fs=f.split(' ',1)
            if fs[0].startswith("FACT"):
                list_of_facts[counter]=re.sub(r'[:\"\"\(\)]','',fs[1])
                counter+=1
    return list_of_facts
```

FIGURE 25 - implementation of the final_fact function

5.3. Implementation of the scraping process

After the Extractor phase, the next work of the AIERL is to search related news from google and scrape a specific set of information from a selected site as described in chapter 4 (Query preparation and searching the web). To perform this duty, we first need to prepare a query to

search from the web from the given news text. The implementation of query generation is shown below figure (figure 26).

```
#query generation from the given news text
import re
from amseg.amharicSegmenter import AmharicSegmenter
sent_punct = []
word_punct = []
segmenter = AmharicSegmenter(sent_punct,word_punct)
def query_generator(text):
    list_of_querys=[]
    querys=segmenter.tokenize_sentence(text)
    for query in querys:
        list_of_querys.append(re.sub(r'[:\፡\።]',' ',query).strip())
    return list_of_querys
```

FIGURE 26 - implementation of query generator

After a query is formed the next step is to search the web and select related news from a specific selected site using the formed queries. For this task, we have created different functions using the below libraries (figure 27)

```
from bs4 import  BeautifulSoup
import requests
import urllib
from requests_html import HTMLSession
import cos_similarity as cs
```

FIGURE 27 - libraries for the scrapper

From the imported libraries the “cos_similarity” is a library class we created to measure the similarity of two given news. This class is created using ‘TfidfVectorizer’ method from the ‘sklearn.feature_extraction.text’ library. After transforming the text into vectorized numbers using the vectorizer for the transformer we then measure the cosine similarity of the documents using the ‘cosine_similarity’ method from the ‘sklearn.metrics.pairwise’ library.

The implementation is shown below figure (figure 28).

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
def similarity_cheker(doc_1,doc_2):
    vectorizer=TfidfVectorizer()
    doc1_tfidf=vectorizer.fit_transform([doc_1])
    doc2_tfidf=vectorizer.transform([doc_2])
    similarity_measure=cosine_similarity(doc1_tfidf,doc2_tfidf)
    return similarity_measure[0][0]
```

FIGURE 28 - cosine similarity implementation

Three methods are created for the scrapping process. The first (scrape_google) googles the web using the queries and stores the results link in a list.

```
def scrape_google(querys):
    listLinks=[]
    for query in querys:
        query = urllib.parse.quote_plus(query)
        response = get_source("https://www.google.com/search?q=" + query)
        if response is not None:
            links = list(response.html.absolute_links)
            google_domains = ('https://www.google.',
                              'https://google.',
                              'https://webcache.googleusercontent.',
                              'http://webcache.googleusercontent.',
                              'https://policies.google.',
                              'https://support.google.',
                              'https://maps.google.')
            selected_sites=('https://www.fanabc','https://waltainfo')
            for url in links[:]:
                if url.startswith(google_domains):
                    links.remove(url)
            for url in links:
                if url.startswith(selected_sites):
                    if url not in listLinks:
                        listLinks.append(url)
        else:
            pass
    return listLinks
```

FIGURE 29 - implementation of search from google

The second method(text_getter), sees through each link and performs the scraping process if and only if the links are from the 3 selected sites. The implementation of the method is site dependent its implemented based on the structure of each site.

The text_getter method only scrapes three pieces of information from the site. It scrapes the title of the news, publisher name if available, or the site's name and the news text by ignoring some unwanted text for the extractor from the news text. After scraping these three pieces of information the function store them in a dictionary with a unique key to be used for the next method.

```
def text_getter(links):
    founded_news={}
    counter=1
    for link in links:
        if requests.get(link):
            html_text= requests.get(link).text
            soup=BeautifulSoup(html_text, 'html.parser')
            #if the link is from walta tv
            if link.startswith('https://waltainfo'):
                text=''
                if soup.find('h1', class_='entry-title') is not None:
                    title=soup.find('h1', class_='entry-title').text.strip()
                    publisher=soup.find('div' ,class_='by-author vcard author')
                    publisher_name=publisher.a.text.strip()
                    contents=soup.find('div',class_='entry-content')
                    if contents is not None:
                        with_div=contents.find_all('div', dir='auto')
                        with_p=contents.find_all('p')
                        if len(with_p)>0:
                            for p in with_p:
                                if p.text.startswith("ለፊጣን መረጃዎች፦"):
                                    break
                                text+=" "+str(p.text)
                        else:
                            for d in with_div:
                                text+=" "+str(d.text)
                        founded_news['w_title'+str(counter)]=title
                        founded_news['w_by_line'+str(counter)]='walta_tv '+publisher_name
                        founded_news['w_news'+str(counter)]=text.strip()
                        counter+=1
                    else:
                        continue
                else:
                    continue
```

FIGURE 30 - scrapper from walta tv

```
#if the link is from fana tv
elif link.startswith('https://www.fanabc'):
    text=''
    if soup.find('span', class_='post-title') is not None:
        title=soup.find('span', class_='post-title').text.strip()
        content=soup.find('div', class_='entry-content clearfix single-post-content')
        if content is not None:
            divs=content.find_all(dir='auto')
            ps=content.find_all('p')
            if len(ps) > 0:
                for p in ps:
                    if p.text.startswith(("ኢትዮጵያን እናልማ፣ እንገንባ፣ እንዘጋጅ","ወቅታዊ፣ትኩስ","ከፈስቡክ ገፃችን")):
                        break
                    text+=" "+str(p.text)
            else:
                for d in divs:
                    if not d.has_attr("class"):
                        text+=str(d.text.strip())

            founded_news['f_title'+str(counter)]=title
            founded_news['f_by_line'+str(counter)]='fana_tv'
            founded_news['f_news'+str(counter)]=text.strip()
            counter+=1

        else:
            continue
    else:
        continue
```

FIGURE 31 - continued part of the text_getter method

The last function(selected_news) selects only one of the most similar news from the scraped news's using their cosine similarity measure by comparing it with the initially given news.

5.4. Implementation of the RL phase

A Markov decision process serves as a representation of the RL phase (MDP). MDP is an outcome prediction model. Based on data provided by the state, the model tries to forecast a result. The Markov decision-making process considers the traits of motives and acts. At any point during the process, the decision-maker may opt to take legal action in the state (The RL). The implementation of this phase consists of building a custom environment by defining action and observation spaces. The implementation of this phase will be discussed briefly in this section(Implementation of the RL phase).

5.4.1. Importing libraries

Since The term "Python library" refers to a group of interconnected modules that include collections of code that can be used repeatedly in various projects. For the programmer, it simplifies and makes Python programming more practical. Since we don't have to write the same code for many projects repeatedly the first task is to import libraries. The libraries imported for this phase are shown in the below figure (figure 32).

```
#importing library for the RL phase
import numpy as np
import stater as st
import numpy as np
import scraper as scr
from gym import Env,spaces
from gym.spaces import Discrete, Box
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Flatten
from tensorflow.keras.optimizers import Adam
from rl.agents import DQNAgent
from rl.policy import EpsGreedyQPolicy
from rl.memory import SequentialMemory
```

FIGURE 32 - importing libraries for the RL phase

5.4.2. Implementation of the custom build Environment

The OpenAI Gym library was used to build the Environment. As per the OpenAI Gym guideline, an environment must inherit from the env class. The class "__init__" and "step" functions are the most mandatory functions to define in the class.

In the "__init__" function we define our action and observation spaces where the action space shows all sets of actions the agent will take in this environment and the observation space defines the state of the environment in which the agent observes to make an action. The gym makes available two different categories of data structures called "Spaces": Box and Discrete. The subset of integers represented by The Discrete class is finite. A Box represents the Cartesian product of n closed intervals. In addition to the box and discrete class, we have used a dict class spaces. The components of this space are represented by (ordered) dictionaries of their respective constituents if we wish to make complex observations or actions more readable for humans, it can be useful to use the class: 'Dict' spaces.

```
class MyEnv(Env):
    ACTION_ELEMENTS=4
    ACTION_TAKE_SOME=0
    ACTION_TAKE_ALL=1
    ACTION_TAKENONE=2
    ACTION_QUIT=3

    CURRENT_STATE={}
    OLD_STATE={}
    state_space=[]

    def __init__(self,size):

        #ACTION SPACE THAT SHOW ALL SETS OF ACTIONS
        self.action_space = Discrete(MyEnv.ACTION_ELEMENTS)

        #THE STATE REPSESENTATION OF THE ENVIRONMENT
        env_space= {
            'match':spaces.Box(0.0,1.0, shape=(1,5)),
            'facts':spaces.Box(0.0,1.0, shape=(1,size))
            }
        self.observation_space = spaces.Dict(env_space)
```

FIGURE 33 - the init function of the environment

In the step function, we define what each action means and what changes it makes in the state. For The step function, we create additional useful methods.

```
def query2(new_facts):
    new_list_of_querys=[]
    for n in new_facts:
        new_list_of_querys.append(new_facts[n])
    return new_list_of_querys
def evaluate_action(action,stt):
    do_the_action=False
    if action == 1:
        if not stt['match'][3] == 1:
            do_the_action=True
    if action == 0:
        if not stt['match'][3] == 1:
            do_the_action=True
    if action == 2:
        if stt['match'] == [1.0, 1.0, 1.0, 1.0, 1.0]:
            do_the_action=True
    return do_the_action
```

FIGURE 34 - additional useful methods

For putting in place the step function, which will have the code to change our environment's state in response to an action. This area, in many respects, makes up the bulk of our environment, and it is here that most planning takes place.

We must first make a list of the conditions that must be met during each environmental transition step. This can be divided into two parts:

1. Applying actions to our agent.
2. Every other occurrence in the surroundings, such as non-RL actors' actions

We define what change action will make in the environment and observation state therefore we individually define rules to follow for all sets of actions.

The step function implementation is shown in the below figures of code (figures 35,36, and 37).

```
def step(self,ACTION,stt):
    MyEnv.OLD_STATE=stt['the new set']
    QUERY=[]
    done=False
    reward=0
    self.state_space=stt['match']
    #ACTION_TAKE NONE_QUIT
    if ACTION == 2:
        do_the_action=MyEnv.evaluate_action(ACTION,stt)
        if do_the_action == True:
            MyEnv.CURRENT_STATE['Head_Line']=stt['the new set']['Head_Line']
            MyEnv.CURRENT_STATE['By_line']=stt['the new set']['By_line']
            MyEnv.CURRENT_STATE['Lead']=stt['the new set']['Lead']
            MyEnv.CURRENT_STATE['Facts']=stt['the new set']['Facts']
            MyEnv.CURRENT_STATE['Mentions']=stt['the new set']['Mentions']
            if stt['match'] == [1.0, 1.0, 1.0, 1.0, 1.0]:
                done=True
                reward=10
        else:
            MyEnv.CURRENT_STATE['Head_Line']=stt['the new set']['Head_Line']
            MyEnv.CURRENT_STATE['By_line']=stt['the new set']['By_line']
            MyEnv.CURRENT_STATE['Lead']=stt['the new set']['Lead']
            MyEnv.CURRENT_STATE['Facts']=stt['the new set']['Facts']
            MyEnv.CURRENT_STATE['Mentions']=stt['the new set']['Mentions']
            QUERY=MyEnv.query2(MyEnv.CURRENT_STATE['Facts'])
            reward=-1
```

FIGURE 35 - implementation of the step function

```
#ACTION_TAKE_SOME
elif ACTION == 0:
    do_the_action=MyEnv.evaluate_action(ACTION,stt)
    if do_the_action == True:
        factsize_current=len(stt['the current set']['Facts'])
        factsize_new=len(stt['the new set']['Facts'])
        if factsize_new > factsize_current:
            MyEnv.CURRENT_STATE['Head_Line']=stt['the current set']['Head_Line']
            MyEnv.CURRENT_STATE['By_line']=stt['the current set']['By_line']
            MyEnv.CURRENT_STATE['Lead']=stt['the current set']['Lead']
            MyEnv.CURRENT_STATE['Facts']=stt['the new set']['Facts']
            MyEnv.CURRENT_STATE['Mentions']=stt['the current set']['Mentions']
        elif factsize_new < factsize_current:
            MyEnv.CURRENT_STATE['Head_Line']=stt['the new set']['Head_Line']
            MyEnv.CURRENT_STATE['By_line']=stt['the new set']['By_line']
            MyEnv.CURRENT_STATE['Lead']=stt['the new set']['Lead']
            MyEnv.CURRENT_STATE['Facts']=stt['the current set']['Facts']
            MyEnv.CURRENT_STATE['Mentions']=stt['the new set']['Mentions']
        else:
            MyEnv.CURRENT_STATE['Head_Line']=stt['the new set']['Head_Line']
            MyEnv.CURRENT_STATE['By_line']=stt['the new set']['By_line']
            MyEnv.CURRENT_STATE['Lead']=stt['the new set']['Lead']
            MyEnv.CURRENT_STATE['Facts']=stt['the new set']['Facts']
            MyEnv.CURRENT_STATE['Mentions']=stt['the new set']['Mentions']
        QUERY=MyEnv.query2(MyEnv.CURRENT_STATE['Facts'])
        reward=1
    else:
        MyEnv.CURRENT_STATE['Head_Line']=stt['the current set']['Head_Line']
        MyEnv.CURRENT_STATE['By_line']=stt['the current set']['By_line']
        MyEnv.CURRENT_STATE['Lead']=stt['the current set']['Lead']
        MyEnv.CURRENT_STATE['Facts']=stt['the current set']['Facts']
        MyEnv.CURRENT_STATE['Mentions']=stt['the current set']['Mentions']
        reward=-1
        QUERY=MyEnv.query2(MyEnv.CURRENT_STATE['Facts'])
```

FIGURE 36 - implementation of the step function continued

```
#ACTION_TAKE_ALL
elif ACTION == 1:
    do_the_action=MyEnv.evaluate_action(ACTION,stt)
    if do_the_action == True:
        MyEnv.CURRENT_STATE['Head_Line']=stt['the new set']['Head_Line']
        MyEnv.CURRENT_STATE['By_line']=stt['the new set']['By_line']
        MyEnv.CURRENT_STATE['Lead']=stt['the new set']['Lead']
        MyEnv.CURRENT_STATE['Facts']=stt['the new set']['Facts']
        MyEnv.CURRENT_STATE['Mentions']=stt['the new set']['Mentions']

        QUERY=MyEnv.query2(MyEnv.CURRENT_STATE['Facts'])
        reward=1
    else:
        MyEnv.CURRENT_STATE['Head_Line']=stt['the current set']['Head_Line']
        MyEnv.CURRENT_STATE['By_line']=stt['the current set']['By_line']
        MyEnv.CURRENT_STATE['Lead']=stt['the current set']['Lead']
        MyEnv.CURRENT_STATE['Facts']=stt['the current set']['Facts']
        MyEnv.CURRENT_STATE['Mentions']=stt['the current set']['Mentions']
        reward=-1
        QUERY=MyEnv.query2(MyEnv.CURRENT_STATE['Facts'])
```

FIGURE 37 - implementation of the step function continued 2

5.4.3. Implementation of the Agent

The DQN agent used in this research comes from the OpenAi Gym library. The agent utilizes a DQN that has a rectified linear unit (ReLU) (24 hidden units) of the input layer, another rectified linear unit (24 hidden units), and one separate liner output layer as described in (The Agent).

```
#inputs for the neural network from the environment
states = env.observation_space.shape
actions = env.action_space.n
#function that build a neural network for the DQN agent
def build_model(states, actions):
    model = Sequential()
    model.add(Dense(24, activation='relu', input_shape=states))
    model.add(Dense(24, activation='relu'))
    model.add(Dense(actions, activation='linear'))
    model.add(Flatten())
    return model
#using the build_model method to build the model
model = build_model(states, actions)
model.summary()
```

FIGURE 38 - the neural network for the DQN agent

The model will be used as an input for the DQN agent as shown in the below figure (figure 39)

```
#a function that bulds the agent using the neural model
def build_agent(model, actions):
    policy = EpsGreedyQPolicy(eps=0.001)
    memory = SequentialMemory(limit=50000, window_length=1)
    dqn = DQNAgent(model=model, memory=memory, policy=policy,
                   nb_actions=actions, nb_steps_warmup=1000, target_model_update=1000)
    return dqn
#USING THE FUNCTION TO BULD THE DQN AGENT
dqn = build_agent(model, actions)
dqn.compile(Adam(learning_rate=1e-3), metrics=['mae'])
dqn.fit(env, nb_steps=50000, visualize=False, verbose=1)
```

FIGURE 39 - the DQN agent

CHAPTER SIX

6. RESULTS AND DISCUSSION

6.1. Overview

The specifics of the AIERL experiment are covered in this chapter. This section outlines the experiment that was done to gauge how well the suggested method and algorithms worked. It discusses the process used to construct the test datasets and annotate the sentences. Additionally, a discussion of the evaluation's findings is included. We outline the set of steps we took to experiment, as well as how we evaluated it and what the results mean.

6.2. Experimental Setup

The process of measuring one or more characteristics of an algorithm or a system is called evaluation. It is crucial for both technology consumers and system engineers in natural language processing (Herbrich & Thore Graepel, 2010).

In this thesis, we will primarily do the black-box evaluation. We evaluate various information extraction components as standalone systems (intrinsic). We performed black-box evaluation within the isolated systems since we will only contrast the outputs for certain inputs with the gold standard.

This research work experiment is broken up into two halves. Since the extractor and the RL are the two phases that make up the entire AIERL (Components of AIERL), we examine each phase independently.

To evaluate the extractor phase, we use Manually annotated documents and each component of the extractor must be evaluated separately according to the annotated gold standards. The most popular evaluation metrics in information extraction precision and recall are employed to measure this phase.

The average reward per episode/epoch or the average reward over the past N timesteps for the continuing activity will be used to evaluate the RL phase. We use the loss function metrics 'MAE' to examine the loss and reward of the RL agent.

6.3. The Dataset

The dataset used in the extractor phase experiment consists of 1590 news stories with 6581 sentences totaling 225269 words. The news was gathered from several online Amharic news sources, including Fana tv, Walta tv, EBC, VOA Amharic, the Reporter Ethiopia, and other online media. The news was gathered at random from several domains, although the business category made up the majority of the news. From the news dataset, a total of 300 news is annotated with the number of sentences, sub sentences, lead, and mentions for the experiment purpose to evaluate individual components of the extractor phase.

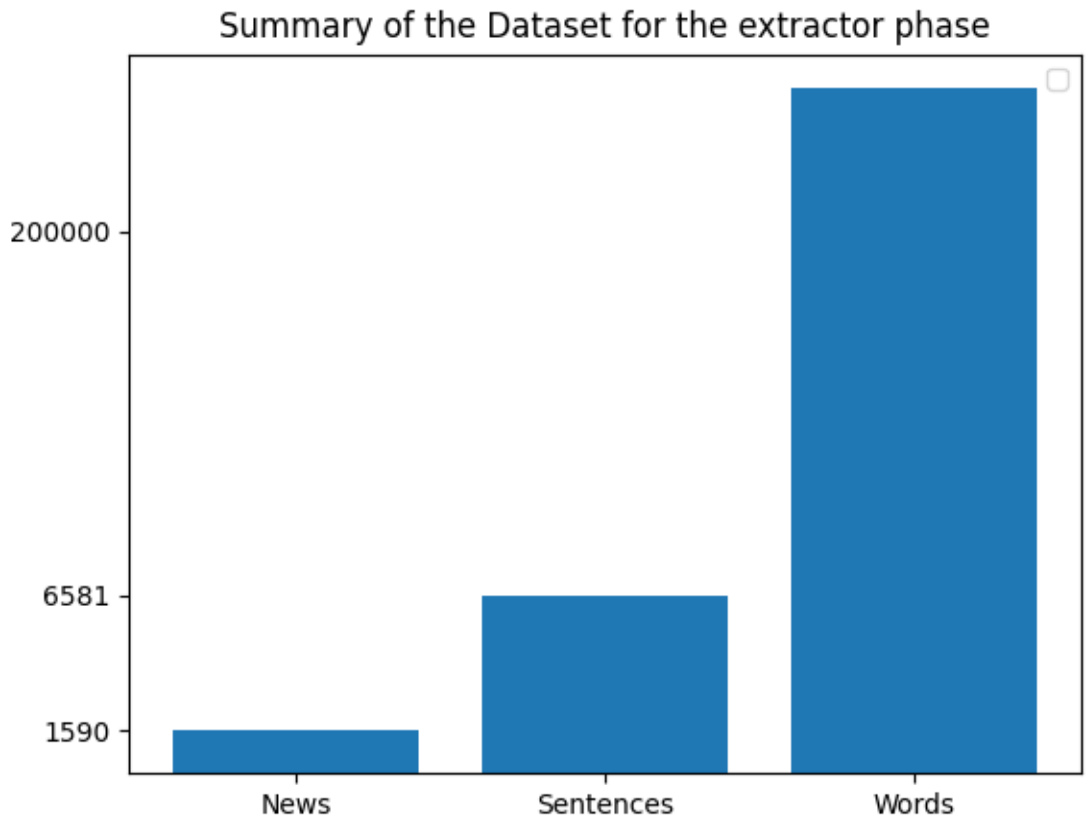


FIGURE 40 - summary of the data set

6.4. Training the RL Agent

The RL agent learns about 50000 steps. Since an agent follows a policy to make a decision, we train the agent using 3 different policies namely: EpsGreedy, Boltzmann, and LinearAnnealed.

```
Training for 50000 steps ...
Interval 1 (0 steps performed)
10000/10000 [==============================] - 37s 4ms/step - reward: 0.7712
Interval 2 (10000 steps performed)
10000/10000 [==============================] - 42s 4ms/step - reward: 0.7582
Interval 3 (20000 steps performed)
10000/10000 [==============================] - 43s 4ms/step - reward: 0.7658
Interval 4 (30000 steps performed)
10000/10000 [==============================] - 42s 4ms/step - reward: 0.7596
Interval 5 (40000 steps performed)
10000/10000 [==============================] - 42s 4ms/step - reward: 0.7602
done, took 206.609 seconds

<keras.callbacks.History at 0x7ff27070b1c0>
```

FIGURE 41 - training using Boltzmann policy

```
Training for 50000 steps ...
Interval 1 (0 steps performed)
10000/10000 [==============================] - 37s 4ms/step - reward: 0.9996
Interval 2 (10000 steps performed)
10000/10000 [==============================] - 41s 4ms/step - reward: 0.9914
Interval 3 (20000 steps performed)
10000/10000 [==============================] - 40s 4ms/step - reward: 0.9990
Interval 4 (30000 steps performed)
10000/10000 [==============================] - 41s 4ms/step - reward: 0.9940
Interval 5 (40000 steps performed)
10000/10000 [==============================] - 42s 4ms/step - reward: 0.9988
done, took 201.643 seconds

<keras.callbacks.History at 0x7ff2604b0be0>
```

FIGURE 42 - training using EpsGreedy policy

```
Training for 50000 steps ...
Interval 1 (0 steps performed)
10000/10000 [==============================] - 38s 4ms/step - reward: 0.8598
Interval 2 (10000 steps performed)
10000/10000 [==============================] - 43s 4ms/step - reward: 0.9032
Interval 3 (20000 steps performed)
10000/10000 [==============================] - 43s 4ms/step - reward: 0.9032
Interval 4 (30000 steps performed)
10000/10000 [==============================] - 46s 5ms/step - reward: 0.8980
Interval 5 (40000 steps performed)
10000/10000 [==============================] - 45s 4ms/step - reward: 0.9042
done, took 214.958 seconds

<keras.callbacks.History at 0x7fad0868e610>
```

FIGURE 43 - training using LinearAnnealed Policy

A summary of the RL agent training using the three different policies is shown in the below figure (figure 44).

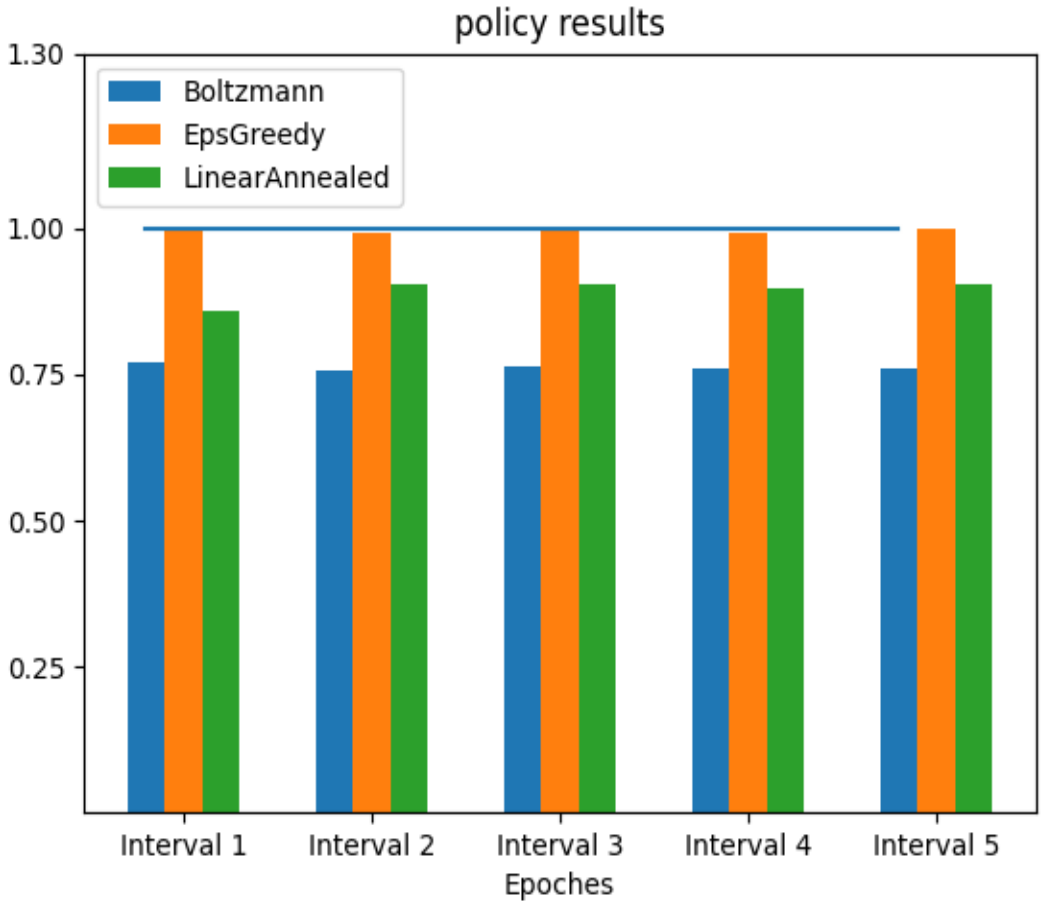


FIGURE 44 - summary of the training

The neural network for the DQN agent is a sequential model with:

- input layer - states from the environment, activation ReLU, Dense (24 units)
- hidden layer- activation ReLU, Dense (24 units)
- output layer – actions from the environment, activation linear, Dense (24 units)

Using the neural network, the DQN agent trains using the following parameters:

Table 4 - DQN agent Training parameters

Policy	EpsGreedyQpolicy(eps=0.001)
memory	Sequential memory(limit=50000)
nb_actions	actions
nb_step_warmup	1000
target_model_update	1000
optimizer	Adam
tearning_rate	1e-3
metrics	mae
nb_steps	50000

6.5. Evaluation Result

As described in the preceding section (Experimental Setup), we examined the two phases separately, as well as the components of the extractor phase, because the performance of each algorithm in the components has a substantial impact on the performance of the extractor phase.

For the Evaluation of the extractor phase, we have used a total of 300 different randomly selected news and manually annotated each news with the corresponding sentences, sub sentences, lead, mentions, and a verb phrase for each sub sentence.

After annotating selected 300 news, we then provide the news for the extractor phase to see how the developed algorithms of each component of the extractor phase correctly recognize the annotated gold standards.

Table 5 - Evaluation Table

	News	Sentences	Sub Sentences	lead	mentions	Verb phrases (for each sub-sentence)
Total Number(annotated)	300	1308	1996	300	547	2112

Using the annotated data set we evaluate four sub-components of the extractor phase namely: sub sentence, lead, mention, and verb phrases, and the experimental results are listed in the below table (table 6)

Table 6 - Experiment result of components

	Total Annotated values	Total Predicted values	Total Correctly predicted values against the annotated	Total incorrectly predicted values against the annotated	Precision (0-1)	Recall (0-1)
Sentence	1308	1308	1308	0	1	1
Sub Sentence	1996	1991	1935	56	0.971	0.969
Lead	300	295	256	39	0.867	0.853
mentions	547	547	546	1	0.998	0.998
Verb Phrases	2112	2197	2031	92	0.924	0.979

As shown in the above table (table 6) the sentence algorithm got a perfect score of precision and recall in the experiment, this happens because to annotate each news to its number of sentences the annotator uses the Amharic sentence delimiter አራት ነጥብ '።' to detect the end of a sentence, similarly, the algorithm to detect a sentence that uses Amharic segmenter and tokenizer(amseg) uses the same sentence delimiter that is the reason for the perfect score.

On the other hand, the mentions sub-component got 0.998 with only 1 incorrectly predicted value this is because both the annotator and the algorithm use the same NER model, the 1 incorrect happens because of misspelling. The reason why we don't annotate manually without using the NER model and testing the algorithm using the NER model is we will end up testing the NER model which is not in our interest to do.

Finally, to evaluate how the RL agent is performing we select 25 random news and search related news for each news from different Ethiopian websites. After searching the news, we then annotate each news with its headline, byline, mentions, and lead from the 3 selected news sites

(Fana, Walta, and EBC) so that each news will have a headline, byline, mentions, and lead from the 3 sites separately. The sample set is shown in the below table (table 7).

Table 7 - Sample testing dataset

Random news	Contents		
የኢትዮጵያ አየር መንገድ ከአዲስ አበባ በቶጎ ሎሜ ወደ ዋሽንግተን ዲሲ ከዛሬ ጀምሮ በሳምንት ሶስት ቀን ቀጥታ በረራ መጀመሩን አስታወቀ።ከአዲስ አበባ ካይሮ በሳምንት አንድ ጊዜ የበረራ አገልግሎት በመስጠት ስራውን የጀመረው የኢትዮጵያ አየር መንገድ በአሁኑ ወቅት 116 ዓለም ዓቀፍ መዳረሻዎች እንዲሁም 23 …	Fana	News	የኢትዮጵያ አየር መንገድ በቶጎ ሎሜ በኩል ወደ …
		Headline	የኢትዮጵያ አየር መንገድ በቶጎ ሎሜ በኩል ወደ ዋሽንግተን…
		By line	ዓለምሰገድ አሳዬ
		lead	የኢትዮጵያ አየር መንገድ
		Mention	የኢትዮጵያ አየር መንገድ, መስፍን ጣሠው, …
	Walta	News	ከአዲስ አበባ ካይሮ በሳምንት አንድ ጊዜ የበረራ…
		Headline	የኢትዮጵያ አየር መንገድ ከአዲስ አበባ በቶጎ ሎሜ ወደ …
		By line	Walta
		lead	የኢትዮጵያ አየር መንገድ
		Mention	የኢትዮጵያ አየር መንገድ
	EBC	News	ግንቦት 24 ቀን 2014 (ኢቢሲ) የኢትዮጵያ አየር መንገድ…
		Headline	የኢትዮጵያ አየር መንገድ ከአዲስ አበባ በቶጎ ሎሜ…
		By line	EBC
		lead	የኢትዮጵያ አየር መንገድ
		Mention	የኢትዮጵያ አየር መንገድ, ከአዲስ አበባ

Then by preparing a simple interface and matching algorithm we feed the random news and run it through the trained RL Agent and see the output results if it matches any of the pre-annotated contents of the news.

The result shows that from the 25 news 21 match all set of a specific website and 4 of them matches partially which indicates the RL Agent is trained and is now performing with more than 85% of accuracy.

CHAPTER SEVEN

7. CONCLUSION AND RECOMMENDATION

7.1. Conclusion

In this thesis, by using the reinforcement learning framework, we proposed a different approach and architecture for the information extraction task and we called it AIERL (Information Extraction from Amharic news text by integrating Reinforcement Learning). To achieve this, we surveyed and compared existing methodologies, algorithms, and strategies for information extraction.

The AIERL has two major phases which work autonomously and are integrated. The first phase (the extractor) is an independent rule-based extractor that extracts headlines, by line, lead, mentions, and facts from given text news by using formulated rule-based algorithms. This phase has 3 components: The extractor preprocessor, language-based learning, and the extractor postprocessor with each component containing internal subcomponents. The extractor can work without the second phase (the RL) and still provides an output however it only focuses on the given news text. The RL, the second stage of the AIERL, is a reinforcement learning model that uses a DQN agent to make decisions about what to do in the environment based on its current state. To establish a state for the DQN agent to act on during this phase, an output from the first phase will be used as input. The DQN agent will continue to query for new relevant news, extract information from the new news, and then formulate a state utilizing the two sets of news until a quit action decision is delivered.

7.2. Contribution

This research work contributes in several ways to the information extraction task for the Amharic language. The following list summarizes the study's primary contributions:

- The overall design of the information extractor from the text in Amharic.
- An algorithm is developed for language-specific sentence simplification that divides a complex sentence into sub-sentences.
- A rule is developed to extract individual standalone information from a news text.
- We present a way to extract more information for given news by preparing queries and extracting from additional sources.

- Designed an algorithm that will query google and select a set of related news from a specific website.
- A way to integrate the Amharic text IE task and reinforcement learning is designed.
- We Represent news in the reinforcement learning environment.
- We Train a DQN agent that will decide what to do from the set of actions in a given state of news.

Generally, the research work contributes to additional tasks such as:

- Validating a News text with a pre-given website to check if it's fake or not by considering the source of the news where and who.
- Retrieving additional information for a given news text from a selected authorized website.
- Correction of misinformation when extracting from the given news text.

7.3. Recommendation

The IE task is very complex for such under-resourced languages, and there has been a lot of research done by different researchers, and for further improvement, we can integrate with different methods and technology such as Reinforcement learning, which we used in this research work to improve different aspects of the IE task. The established Amharic information extraction has aspects that need to be improved further, and we would like to recommend these as future tasks. We have identified the following work directions for the future.

- The size of the training and test collection used in this study is insufficient. However, one can improve performance by increasing data collection.
- This research work uses POS and NER but, in the future, using the Amharic Text summarizer model will help extract the lead role of the news from the overall news text since in this research work, we only extract the lead role from the headline of the news. We also recommend using Amharic stemmer to get the root form of the verb phrase in the verb phrase sub component
- Since the extractor is an independent phase, we recommend using a developed or developing machine learning or deep learning Amharic information extractor model rather than a rule-based information extractor system.
- For this research work, we train only DQN agent but for future work, we recommend Training different Agents and comparing their results.

***

This research was sponsored by Adama science and Technology University with a grant number

ASTUSM-R/504/22

Adama, Ethiopia

References

Ephrem Tadesse, Rosa Tsegaye, & Kuulaa Qaqqabaa. (2020). Event extraction from unstructured amharic text. *Proceedings of the 12th Language Resources and Evaluation Conference*.

Jerry , H., & Ellen , R. (2010). *Handbook of Natural Language Processing: InformationExtraction", Machine Learning and Pattern Recognition Series, 2nd Edition.*

Aggarwal, C. C. (2012). *Mining Text Data.* Yorktown Heights, NY, USA: Springer.

Alansary, S., & Magdy Nagi. (2015). A knowledge extaction system (KEYS) based on UNL knowledge infrastructure. *TENCON 2015 - 2015 IEEE Region 10 Conference*, 8.

Ali Mohamed Nabil Allam, & Mohamed Hassan Haggag. (2012). The Question Answering Systems: A Survey. *International Journal of Research and Reviews in Information Sciences (IJRRIS).*

Appelt, D., & David, I. (n.d.). Introduction to Information Extraction Technology. *a Tutorial Prepared for IJCAI-99, Artificial Intelligence Center, Menlo Park, CA.*

Belay, T. D. (2021). Impacts of Homophone Normalization on Semantic Models for Amharic. *IEEE*.

Bjorn Gamback, Fredrik Olsson, Atelach Alemu Argaw, & Lars Asker. (2007). Methods for Amharic Part-of-Speech Tagging.

Chinchor, N. (2000). *MUC-7 Named Entity Task Definition.*

Elsadig, M. A., Ali Ahmed, & Mubatak Himmat. (2015). INFORMATION EXTRACTION METHODS AND EXTRACTION. *ARPN*, 7.

Erik Cambria, & Bebo White. (2014). Jumping NLP Curves: A Review of Natural Language Processing Research. *IEEE*.

Feng, Y., Hongjun, Z., Wenning , H., & Gang, C. (2017). Joint Extraction of Entities and Relations using Reinforcemrnt Learning and Deep Learning. *Hindawi*.

Fisseha, S. G. (2020). *Amharic Open Information Extraction.* ADDIS ABABA UNIVERSITY.

Gero, S., Getu Girma, & Shimels Hailu . (2021). Using Classification Model for Information Extraction. *GSJ*, 12.

Girma, S. (2020). Amharic Open Information Extraction. *Springer*, 457-469.

Goncalo, S., Helena , G., & Luisa , C. (n.d.). *Information Extraction tasks: a survey.*

Herbrich, R., & Thore Graepel. (2010). *HANDBOOK OF NATURAL LANGUAGE PROCESSING.* New York: Springer.

Hirpassa, S. (2017). Information Extraction System for Amharic Text. *International Journal of Computer Science Trends and Technology (IJCST)*, 15.

Hobbs, J. R. (2004). Information extraction from biomedical text. *Journal of Biomedical Informatics*, 260-264.

Jakub Piskorski, & Roman Yangarber. (2012). Information Extraction: Past, Present and Future. *Springer*.

Jim, C., & Yorik, W. (n.d.). *"Information Extraction", Hand Book of Natural Language.* Robert Dale, Hermann Moisl and Harold Somers,.

Johannes, P. (February 2004). *Multilingual Information Extraction.* University of Helsinki 15th: Department of Computer Science.

Martijn van Otterlo , & Marco Wiering . (2012). Reinforcement Learning and Markov Decision Processes. *Springer*.

Mckane, A. (2014). *News Writing.*

Mehdi Allahyari, Seyedamin Pouriyeh, & Mehdi Assef. (2017). Text Summarization Techniques: A Brief Survey. *arXiv*.

Monika Arora, Uma Kanjilal, & Dinesh Varshney. (2016). Evaluation of information retrieval: precision and. *Research Gate*.

Narasimhan, K., Adam Yala, & Regina Barzilay. (2016). Improving Information Extraction by Acquiring External Evidence with. *GSJ*, 12.

Richard S. Sutton, & Andrew G. Barto. (2014). *Reinforcement Learning:*. London, England: The MIT Press.

Richardson, L. (2019). *Beautiful Soup Documentation.*

Ronen Feldman, Benjamin Rosenfeld, & Moshe Fresko. (2006). TEG—a hybrid approach to information extraction. *Springer*.

Sharma, A., Parekh, Z, & Talukdar, P. (2017). Speeding up reinforcement learning-based information extraction training using asynchronous methods. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing* , 2658-2663.

Smruthi, M., Rohini, S., & Erik, P. (2010). An Information-Extraction System for Urdu---A Resource-Poor Language. *ACM Transactions on Asian Language Information Processing*.

Tadesse, E., Rosa Tsegaye Aga, & Kuulaa Qaqqabaa. (2020). Event Extraction from Unstructured Amharic Text. *LREC 2020*, 7.

Takanobu, R., Tianyang Zhang, Jiexi Liu, & Minlie Huang. (2018). A Hierarchical Framework for Relation Extraction. *THUAI*, 9.

Tereszkiewicz, A. (2012). Lead, headline, news abstract? – genre conventions of news sections on newspaper websites.

Tsedalu, G. (2010). *INFORMATION EXTRACTION MODEL FROM AMHARIC* . ADDIS ABABA UNIVERSITY .

Tsedalu, G. (2010). Information extraction model from Amharic news texts. *Addis Ababa University*.

UmrinderPal Singh, Vishal Goyal, & Gurpreet Sin. (2012). Named Entity Recognition System for Urdu. *Proceedings of COLING 2012: Technical Papers, ,* 2507–2518.

Vaismoradi, M. (2013). Content analysis and thematic analysis: Implications for conducting a qualitative descriptive study.

Yimam, S. M. (2021). Introducing Various Semantic Models for Amharic: Experimentation and Evaluation with Multiple Tasks and Datasets. *Future Internet*.

Yong, H. (2014). An Empirical Review of Research Methodologies and Methods in Creativity Studies. *Creative research journal*.

Zaremba, G. B. (2016). *OpenAI Gym.*

ይማም, ባ. (1987). *የ አማር ኛ ሰዋሰው.* ት.መ.ማ.ማ.ድ. .

APPENDICES

Appendix A: sample code for the RL phase

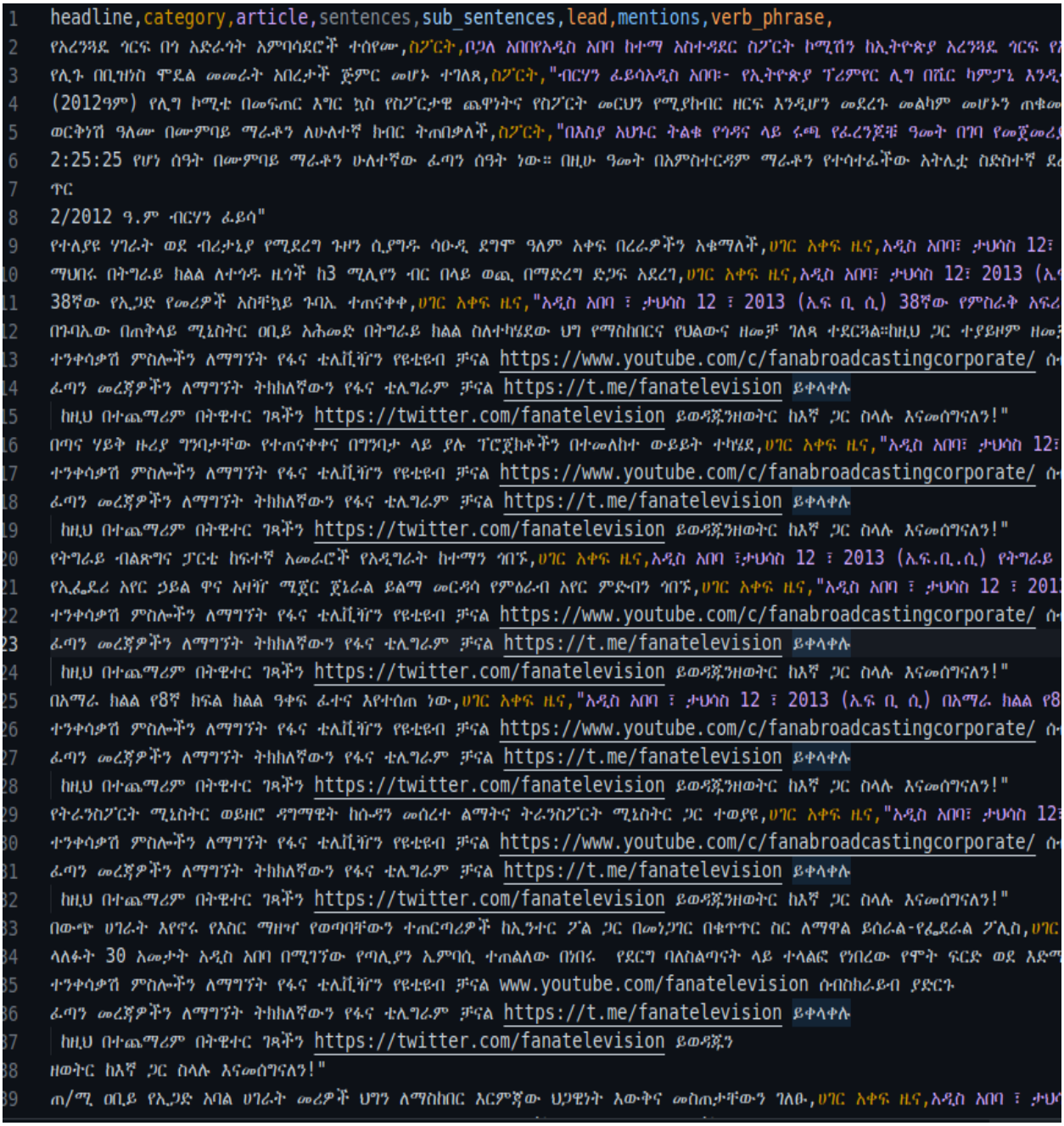
```
#importing library for the RL phase
import numpy as np
import stater as st
import numpy as np
from gym import Env,spaces
from gym.spaces import Discrete, Box
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Flatten
from tensorflow.keras.optimizers import Adam
from rl.agents import DQNAgent
from rl.policy import EpsGreedyQPolicy,BoltzmannQPolicy,SoftmaxPolicy,LinearAnnealedPolicy
from rl.memory import SequentialMemory
MyEnv(Env):
    ACTION_ELEMENTS=3
    ACTION_TAKE_SOME=0
    ACTION_TAKE_ALL=1
    ACTION_TAKENONE_QUIT=2
    CURRENT_STATE={}
    OLD_STATE={}
    state_space=[]
    def __init__(self):
        self.action_space = Discrete(MyEnv.ACTION_ELEMENTS)
        self.observation_space = Box(0.0,1.0, shape=(1,5))
    def query2(new_facts):
        new_list_of_querys=[]
        for n in new_facts:
            new_list_of_querys.append(new_facts[n])
        return new_list_of_querys
    def evaluate_action(action,stt):
        do_the_action=False
        if action == 1:
            if not stt['match'][3] == 1:
                do_the_action=True
        if action == 0:
            if not stt['match'][3] == 1:
                do_the_action=True
        if action == 2:
            if stt['match'] == [1.0, 1.0, 1.0, 1.0, 1.0]:
                do_the_action=True
        return do_the_action
    def step(self,ACTION,stt):
        MyEnv.OLD_STATE=stt['the new set']
        QUERY=[]
        done=False
        reward=0
```

```
self.state_space=stt['match']
#ACTION_TAKE NONE_QUIT
if ACTION == 2:
    do_the_action=MyEnv.evaluate_action(ACTION,stt)
    if do_the_action == True:
        MyEnv.CURRENT_STATE['Head_Line']=stt['the new set']['Head_Line']
        MyEnv.CURRENT_STATE['By_line']=stt['the new set']['By_line']
        MyEnv.CURRENT_STATE['Lead']=stt['the new set']['Lead']
        MyEnv.CURRENT_STATE['Facts']=stt['the new set']['Facts']
        MyEnv.CURRENT_STATE['Mentions']=stt['the new set']['Mentions']
        if stt['match'] == [1.0, 1.0, 1.0, 1.0, 1.0]:
            done=True
            reward=10
    else:
        MyEnv.CURRENT_STATE['Head_Line']=stt['the new set']['Head_Line']
        MyEnv.CURRENT_STATE['By_line']=stt['the new set']['By_line']
        MyEnv.CURRENT_STATE['Lead']=stt['the new set']['Lead']
        MyEnv.CURRENT_STATE['Facts']=stt['the new set']['Facts']
        MyEnv.CURRENT_STATE['Mentions']=stt['the new set']['Mentions']
        QUERY=MyEnv.query2(MyEnv.CURRENT_STATE['Facts'])
        reward=-1
#ACTION_TAKE_SOME
elif ACTION == 0:
    do_the_action=MyEnv.evaluate_action(ACTION,stt)
    if do_the_action == True:
        factsize_current=len(stt['the current set']['Facts'])
        factsize_new=len(stt['the new set']['Facts'])
        if factsize_new > factsize_current:
            MyEnv.CURRENT_STATE['Head_Line']=stt['the current set']['Head_Line']
            MyEnv.CURRENT_STATE['By_line']=stt['the current set']['By_line']
            MyEnv.CURRENT_STATE['Lead']=stt['the current set']['Lead']
            MyEnv.CURRENT_STATE['Facts']=stt['the new set']['Facts']
            MyEnv.CURRENT_STATE['Mentions']=stt['the current set']['Mentions']
        elif factsize_new < factsize_current:
            MyEnv.CURRENT_STATE['Head_Line']=stt['the new set']['Head_Line']
            MyEnv.CURRENT_STATE['By_line']=stt['the new set']['By_line']
            MyEnv.CURRENT_STATE['Lead']=stt['the new set']['Lead']
            MyEnv.CURRENT_STATE['Facts']=stt['the current set']['Facts']
            MyEnv.CURRENT_STATE['Mentions']=stt['the new set']['Mentions']
        else:
            MyEnv.CURRENT_STATE['Head_Line']=stt['the new set']['Head_Line']
            MyEnv.CURRENT_STATE['By_line']=stt['the new set']['By_line']
            MyEnv.CURRENT_STATE['Lead']=stt['the new set']['Lead']
            MyEnv.CURRENT_STATE['Facts']=stt['the new set']['Facts']
            MyEnv.CURRENT_STATE['Mentions']=stt['the new set']['Mentions']
        QUERY=MyEnv.query2(MyEnv.CURRENT_STATE['Facts'])
        reward=1
    else:
        MyEnv.CURRENT_STATE['Head_Line']=stt['the current set']['Head_Line']
        MyEnv.CURRENT_STATE['By_line']=stt['the current set']['By_line']
        MyEnv.CURRENT_STATE['Lead']=stt['the current set']['Lead']
```

```
            MyEnv.CURRENT_STATE['Facts']=stt['the current set']['Facts']
            MyEnv.CURRENT_STATE['Mentions']=stt['the current set']['Mentions']
            reward=-1
            QUERY=MyEnv.query2(MyEnv.CURRENT_STATE['Facts'])
        #ACTION_TAKE_ALL
        elif ACTION == 1:
            do_the_action=MyEnv.evaluate_action(ACTION,stt)
            if do_the_action == True:
                MyEnv.CURRENT_STATE['Head_Line']=stt['the new set']['Head_Line']
                MyEnv.CURRENT_STATE['By_line']=stt['the new set']['By_line']
                MyEnv.CURRENT_STATE['Lead']=stt['the new set']['Lead']
                MyEnv.CURRENT_STATE['Facts']=stt['the new set']['Facts']
                MyEnv.CURRENT_STATE['Mentions']=stt['the new set']['Mentions']

                QUERY=MyEnv.query2(MyEnv.CURRENT_STATE['Facts'])
                reward=1
            else:
                MyEnv.CURRENT_STATE['Head_Line']=stt['the current set']['Head_Line']
                MyEnv.CURRENT_STATE['By_line']=stt['the current set']['By_line']
                MyEnv.CURRENT_STATE['Lead']=stt['the current set']['Lead']
                MyEnv.CURRENT_STATE['Facts']=stt['the current set']['Facts']
                MyEnv.CURRENT_STATE['Mentions']=stt['the current set']['Mentions']
                reward=-1
                QUERY=MyEnv.query2(MyEnv.CURRENT_STATE['Facts'])

        return MyEnv.OLD_STATE,MyEnv.CURRENT_STATE,reward,done,QUERY
    def reset(self):
        self.state_space=[0.0,0.0,0.0,0.0,0.0]
        return self.state_space
    def render(self):
        pass
env=MyEnv()
states = env.observation_space.shape
actions = env.action_space.n
def build_model(states, actions):
    model = Sequential()
    model.add(Dense(48, activation='relu', input_shape=states))
    model.add(Dense(48, activation='relu'))
    model.add(Flatten())
    model.add(Dense(actions, activation='linear'))
    return model

model = build_model(states, actions)
model.summary()

#a function that bulds the agent using the neural model
def build_agent(model, actions):
    policy = EpsGreedyQPolicy(eps=0.001)
    #policy = BoltzmannQPolicy()
    #policy = LinearAnnealedPolicy(EpsGreedyQPolicy(), attr='eps', value_max=1.,
    # value_min=.1, value_test=.05,nb_steps=1000)
```

```
    memory = SequentialMemory(limit=50000, window_length=1)
    dqn = DQNAgent(model=model, memory=memory, policy=policy,
                   nb_actions=actions, nb_steps_warmup=1000, target_model_update=1000)
    return dqn
#USING THE FUNCTION TO BULD THE DQN AGENT
dqn = build_agent(model, actions)
dqn.compile(Adam(learning_rate=1e-3), metrics=['mae'])
dqn.fit(env, nb_steps=50000, visualize=False, verbose=1)

scores = dqn.test(env, nb_episodes=10, visualize=False)
print(np.mean(scores.history['episode_reward'])))
#SAVING THE MODEL
Self.dqn.save(self.dqn)
Self.dqn=load_model(self.dqn)
```

Appendix B: sample Dataset



headline,category,article,sentences,sub_sentences,lead,mentions,verb_phrase,

የአረንጓዴ ጎርፍ በ9 አድራጎት አምባሳደሮች ተሰየሙ,ስፖርት,ቦጋለ አበበየአዲስ አበባ ከተማ አስተዳደር ስፖርት ኮሚሽን ከኢትዮጵያ አረንጓዴ ጎርፍ የ

የሊጉ በቢዝነስ ሞዴል መመራት አበረታች ጅምር መሆኑ ተገለጸ,ስፖርት,"ብርሃን ፊይሳአዲስ አበባ፡- የኢትዮጵያ ፕሪምየር ሊግ በሸር ካምፓኒ እንዲ

(2012ዓም) የሊግ ኮሚቴ በመፍጠር እግር ኳስ የስፖርታዊ ጨዋነትና የስፖርት መርህን የሚያከብር ዘርፍ እንዲሆን መደረጉ መልካም መሆኑን ጠቁመ

ወርቅነሽ ዓለሙ በሙምባይ ማራቶን ለሁለተኛ ከብር ትጠበቃለች,ስፖርት,"በእስያ አህጉር ትልቁ የጎዳና ላይ ሩጫ የፊረንጀቹ ዓመት በገባ የመጀመሪ

2:25:25 የሆነ ሰዓት በሙምባይ ማራቶን ሁለተኛው ፈጣን ሰዓት ነው። በዚሁ ዓመት በአምስተርዳም ማራቶን የተሳተፈችው አትሌቷ ስድስተኛ ደ

ጥር

2/2012 ዓ.ም ብርሃን ፊይሳ"

የተለያዩ ሃገራት ወደ ብሪታኒያ የሚደረግ ጉዞን ሲያግዱ ሳቡዲ ደግሞ ዓለም አቀፍ በረራዎችን አቁማለች,ሀገር አቀፍ ዜና,አዲስ አበባ፣ ታህሳስ 12፣

ማህበሩ በትግራይ ክልል ለተጎዱ ዜጎች ከ3 ሚሊየን ብር በላይ ወጪ በማድረግ ድጋፍ አደረገ,ሀገር አቀፍ ዜና,አዲስ አበባ፣ ታህሳስ 12፣ 2013 (አ

38ኛው የኢጋድ የመሪዎች አስቸኳይ ጉባኤ ተጠናቀቀ,ሀገር አቀፍ ዜና,"አዲስ አበባ ፣ ታህሳስ 12 ፣ 2013 (ኤፍ ቢ ሲ) 38ኛው የምስራቅ አፍሪ

በጉባኤው በጠቅላይ ሚኒስትር ዐቢይ አሕመድ በትግራይ ክልል ስለተካሄደው ህግ የማስከበርና የህልውና ዘመቻ ገለጻ ተደርጓል።ከዚህ ጋር ተያይዞም ዘመ

ተንቀሳቃሽ ምስሎችን ለማግኘት የፋና ቴሌቪዥን የዩቲዩብ ቻናል https://www.youtube.com/c/fanabroadcastingcorporate/ ሱ

ፈጣን መረጃዎችን ለማግኘት ትክክለኛውን የፋና ቴሌግራም ቻናል https://t.me/fanatelevision ይቀላቀሉ

ከዚህ በተጨማሪም በትዊተር ገጻችን https://twitter.com/fanatelevision ይወዳጁንዝውትር ከእኛ ጋር ስላሉ እናመሰግናለን!"

በጣና ሃይቅ ዙሪያ ግንባታቸው የተጠናቀቀና በግንባታ ላይ ያሉ ፕሮጀክቶችን በተመለከተ ውይይት ተካሄደ,ሀገር አቀፍ ዜና,"አዲስ አበባ፣ ታህሳስ 12፣

ተንቀሳቃሽ ምስሎችን ለማግኘት የፋና ቴሌቪዥን የዩቲዩብ ቻናል https://www.youtube.com/c/fanabroadcastingcorporate/ ሱ

ፈጣን መረጃዎችን ለማግኘት ትክክለኛውን የፋና ቴሌግራም ቻናል https://t.me/fanatelevision ይቀላቀሉ

ከዚህ በተጨማሪም በትዊተር ገጻችን https://twitter.com/fanatelevision ይወዳጁንዝውትር ከእኛ ጋር ስላሉ እናመሰግናለን!"

የትግራይ ብልጽግና ፓርቲ ከፍተኛ አመራሮች የአዲግራት ከተማን ጎበኙ,ሀገር አቀፍ ዜና,አዲስ አበባ ፣ታህሳስ 12 ፣ 2013 (ኤፍ.ቢ.ሲ) የትግራይ

የኢፌዴሪ አየር ኃይል ዋና አዛዥ ሜጀር ጀኔራል ይልማ መርዳሳ የምዕራብ አየር ምድብን ጎበኙ,ሀገር አቀፍ ዜና,"አዲስ አበባ ፣ ታህሳስ 12 ፣ 201

ተንቀሳቃሽ ምስሎችን ለማግኘት የፋና ቴሌቪዥን የዩቲዩብ ቻናል https://www.youtube.com/c/fanabroadcastingcorporate/ ሱ

ፈጣን መረጃዎችን ለማግኘት ትክክለኛውን የፋና ቴሌግራም ቻናል https://t.me/fanatelevision ይቀላቀሉ

ከዚህ በተጨማሪም በትዊተር ገጻችን https://twitter.com/fanatelevision ይወዳጁንዝውትር ከእኛ ጋር ስላሉ እናመሰግናለን!"

በአማራ ክልል የ8ኛ ክፍል ክልል ዓቀፍ ፈተና እየተሰጠ ነው,ሀገር አቀፍ ዜና,"አዲስ አበባ ፣ ታህሳስ 12 ፣ 2013 (ኤፍ ቢ ሲ) በአማራ ክልል የ8

ተንቀሳቃሽ ምስሎችን ለማግኘት የፋና ቴሌቪዥን የዩቲዩብ ቻናል https://www.youtube.com/c/fanabroadcastingcorporate/ ሱ

ፈጣን መረጃዎችን ለማግኘት ትክክለኛውን የፋና ቴሌግራም ቻናል https://t.me/fanatelevision ይቀላቀሉ

ከዚህ በተጨማሪም በትዊተር ገጻችን https://twitter.com/fanatelevision ይወዳጁንዝውትር ከእኛ ጋር ስላሉ እናመሰግናለን!"

የትራንስፖርት ሚኒስትር ወይዘሮ ዳግማዊት ከሱዳን መሰረተ ልማትና ትራንስፖርት ሚኒስትር ጋር ተወያዩ,ሀገር አቀፍ ዜና,"አዲስ አበባ፣ ታህሳስ 12፣

ተንቀሳቃሽ ምስሎችን ለማግኘት የፋና ቴሌቪዥን የዩቲዩብ ቻናል https://www.youtube.com/c/fanabroadcastingcorporate/ ሱ

ፈጣን መረጃዎችን ለማግኘት ትክክለኛውን የፋና ቴሌግራም ቻናል https://t.me/fanatelevision ይቀላቀሉ

ከዚህ በተጨማሪም በትዊተር ገጻችን https://twitter.com/fanatelevision ይወዳጁንዝውትር ከእኛ ጋር ስላሉ እናመሰግናለን!"

በውጭ ሀገራት እየኖሩ የአስር ማዘጋጃ የወጣባቸውን ተጠርጣሪዎች ከኢንተር ፖል ጋር በመነጋገር በቁጥጥር ስር ለማዋል ይሰራል-የፌደራል ፖሊስ,ሀገር

ላለፉት 30 አመታት አዲስ አበባ በሚገኘው የጣሊያን ኢምባሲ ተጠልለው በነበሩ የደርግ ባለስልጣናት ላይ ተላልፎ የነበረው የሞት ፍርድ ወደ እድ

ተንቀሳቃሽ ምስሎችን ለማግኘት የፋና ቴሌቪዥን የዩቲዩብ ቻናል www.youtube.com/fanatelevision ሱብስክራይብ ያድርጉ

ፈጣን መረጃዎችን ለማግኘት ትክክለኛውን የፋና ቴሌግራም ቻናል https://t.me/fanatelevision ይቀላቀሉ

ከዚህ በተጨማሪም በትዊተር ገጻችን https://twitter.com/fanatelevision ይወዳጁን

ዘወትር ከእኛ ጋር ስላሉ እናመሰግናለን!"

ጠ/ሚ ዐቢይ የኢጋድ አባል ሀገራት መሪዎች ህግን ለማስከበር እርምጃው ህጋዊነት እውቅና መስጠታቸውን ገለፁ,ሀገር አቀፍ ዜና,አዲስ አበባ ፣ ታህ

Appendix C: sample running outputs

text='በደቡብ ክልል ባለፉት 11 ወራት ከ11 ቢሊየን ብር በላይ ገቢ መሰብሰቡን የክልሉ ገቢዎች ቢሮ አስታወቀ፡፡ በበጀት ዓመቱ 13 ነጥብ 1 ቢሊየን ብር ለመሰብሰብ ታቅዶ ወደ ስራ የተገባ ሲሆን÷ ባለፉት 11 ወራት ከመደበኛ እና ከማዘጋጃ ቤት ከ11 ነጥብ 6 ቢሊየን በላይ ብር ገቢ መሰብሰብ ተችሏል። እንደ ኃላፊዋ ገለጻ በኡዲት ፣ የግብር ከፋዩን ማህደር በመመርመር የተገኙ የአሰራር ግድፈቶችን ለማረም ጥረት የተደረገ ሲሆን የህግ ጥሰት በፈጸሙ ሰራተኞች ላይ ህጋዊ እርምጃ መወሰዱን ጠቁመዋል። የገቢ አሰባሰቡ ከእቅዱ አንጻር የተሻለ አፈጻጸም ቢያሳይም ካለው የገቢ አሰባሰብ አቅም አንጻር የሚቀሩ ጉዳዮች መኖራቸውን ተገንዝበናል ብለዋል። ከህግ ማስከበር ጋር በተያያዘ በፈጻሚው እና በግብር ከፋዩ ዘንድ የሚስተዋሉ የስነ ምግባር ጥሰቶችን በመለየት የእርምት እርምጃ ተወስዷል ያሉት ኃላፊዋ በተለይ ሀሰተኛ ደረሰኝ ማቅረብ እና ግብር መሰወር ላይ ያሉ ችግሮችን የማረም ስራ እየተሰራ መሆኑን ጠቁመዋል። በክልሉ ከ1 ሺህ በላይ ሀሰተኛ ደረሰኝ የተገኘ ሲሆን÷ ከ187 ሚሊየን ብር በላይ በህገ ወጥ መንገድ ግብይት እንደተፈጸመ መናገራቸውን ከክልሉ መንግስት ኮሙኒኬሽን ያገኘነው መረጃ ያመላክታል፡፡'

2022-09-01 10:42:17.424404: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'libcudart.so.11.0'; dlerror: libcudart.so.11.0: cannot open shared object file: No such file or directory
2022-09-01 10:42:17.424492: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore above cudart dlerror if you do not have a GPU set up on your machine.
2022-09-01 10:42:38,997 loading file pro/NERfinal-model.pt
2022-09-01 10:42:41,891 loading file pro/POSfinal-model.pt

0 -1 False
{'the current set': {'Head_Line': 'unknown', 'By_line': 'unknown', 'Lead': 'unknown', 'Facts': {1: 'ባለፉት 11 ወራት ከ 11 ቢሊየን ብር በላይ ገቢ መሰብሰቡን', 2: 'የክልሉ ገቢዎች ቢሮ አስታወቀ', 3: 'በበጀት ዓመቱ 13 ነጥብ 1 ቢሊየን ብር ለመሰብሰብ ታቅዶ', 4: '11 ወራት ከመደበኛ እና ከማዘጋጃ ቤት ከ 11 ነጥብ 6 ቢሊየን በላይ ብር ገቢ መሰብሰብ ተችሏል', 5: 'የህግ ጥሰት በፈጸሙ ሰራተኞች ላይ ህጋዊ እርምጃ መወሰዱን ጠቁመዋል', 6: 'የሚቀሩ ጉዳዮች መኖራቸውን ተገንዝበናል ብለዋል', 7: 'በፈጻሚው እና በግብር ከፋዩ ዘንድ የሚስተዋሉ የስነ ምግባር ጥሰቶችን በመለየት የእርምት እርምጃ ተወስዷል', 8: 'ሀሰተኛ ደረሰኝ ማቅረብ', 9: 'እና ግብር መሰወር', 10: 'ላይ ያሉ ችግሮችን የማረም ስራ እየተሰራ መሆኑን ጠቁመዋል', 11: 'በክልሉ ከ 1 ሺህ በላይ ሀሰተኛ ደረሰኝ የተገኘ ሲሆን ከ 187 ሚሊየን ብር በላይ በህገ ወጥ መንገድ ግብይት እንደተፈጸመ መናገራቸውን', 12: 'ከክልሉ መንግስት ኮሙኒኬሽን ያገኘነው መረጃ ያመላክታል'}, 'Mentions': ['በደቡብ ክልል']}, 'the new set': {'Head_Line': 'በደቡብ ክልል ባለፉት 11 ወራት ከ11 ነጥብ 6 ቢሊየን ብር በላይ ገቢ ተሰበሰበ', 'By_line': 'fana_tv', 'Lead': 'በደቡብ ክልል', 'Facts': {1: 'ባለፉት 11 ወራት ከ 11 ነጥብ 6 ቢሊየን ብር በላይ ገቢ መሰብሰቡን', 2: 'የክልሉ ገቢዎች ቢሮ አስታወቀ', 3: 'የቢሮው ኃላፊ ወይዘሮ ዘይቱና ኢብራሂም', 4: 'በበጀት ዓመቱ 13 ነጥብ 1 ቢሊየን ብር ለመሰብሰብ ታቅዶ', 5: '11 ወራት ከመደበኛ እና ከማዘጋጃ ቤት ከ 11 ነጥብ 6 ቢሊየን በላይ ብር ገቢ መሰብሰብ ተችሏል', 6: 'የህግ ጥሰት በፈጸሙ ሰራተኞች ላይ ህጋዊ እርምጃ መወሰዱን ጠቁመዋል', 7: 'የሚቀሩ ጉዳዮች መኖራቸውን ተገንዝበናል ብለዋል', 8: 'በፈጻሚው እና በግብር ከፋዩ ዘንድ የሚስተዋሉ የስነ ምግባር ጥሰቶችን በመለየት የእርምት እርምጃ ተወስዷል', 9: 'ሀሰተኛ ደረሰኝ ማቅረብ', 10: 'እና ግብር መሰወር', 11: 'ላይ ያሉ ችግሮችን የማረም ስራ እየተሰራ መሆኑን ጠቁመዋል', 12: 'በክልሉ ከ 1 ሺህ በላይ ሀሰተኛ ደረሰኝ የተገኘ ሲሆን ከ 187 ሚሊየን ብር በላይ በህገ ወጥ መንገድ ግብይት እንደተፈጸመ መናገራቸውን', 13: 'ከክልሉ መንግስት ኮሙኒኬሽን ያገኘነው መረጃ ያመላክታል', 14: 'በቀጣዩ 2015 በጀት ዓመት 15 ነጥብ 665 ቢሊየን ብር ገቢ ለመሰብሰብ ማቀዱ ታውቋል'}, 'Mentions': ['አዲስ አበባ', 'በደቡብ ክልል', 'ወይዘሮ ዘይቱና ኢብራሂም']}, 'match': [0.26, 0.6, 0.4, 0.92, 0.59], 'comparitions_of_facts': [0.93, 1.0, 0.22, 0.55, 0.34, 0.27, 0.26, 0.14, 0.34, 0.26, 0.25, 0.27, 0.0, 0.0]}

2 -1 False
{'the current set': {'Head_Line': 'unknown', 'By_line': 'unknown', 'Lead': 'unknown', 'Facts': {1: 'ባለፉት 11 ወራት ከ 11 ነጥብ 6 ቢሊየን ብር በላይ ገቢ መሰብሰቡን', 2: 'የክልሉ ገቢዎች ቢሮ አስታወቀ', 3: 'የቢሮው ኃላፊ ወይዘሮ ዘይቱና ኢብራሂም', 4: 'በበጀት ዓመቱ 13 ነጥብ 1 ቢሊየን ብር ለመሰብሰብ ታቅዶ', 5: '11 ወራት ከመደበኛ እና ከማዘጋጃ ቤት ከ 11 ነጥብ 6 ቢሊየን በላይ ብር ገቢ መሰብሰብ ተችሏል', 6: 'የህግ ጥሰት በፈጸሙ ሰራተኞች ላይ ህጋዊ እርምጃ መወሰዱን ጠቁመዋል', 7: 'የሚቀሩ ጉዳዮች መኖራቸውን ተገንዝበናል ብለዋል', 8: 'በፈጻሚው እና በግብር ከፋዩ ዘንድ የሚስተዋሉ የስነ ምግባር ጥሰቶችን በመለየት የእርምት እርምጃ ተወስዷል', 9: 'ሀሰተኛ ደረሰኝ ማቅረብ', 10: 'እና ግብር መሰወር', 11: 'ላይ ያሉ ችግሮችን የማረም ስራ እየተሰራ መሆኑን ጠቁመዋል', 12: 'በክልሉ ከ 1 ሺህ በላይ ሀሰተኛ ደረሰኝ የተገኘ ሲሆን ከ 187 ሚሊየን ብር በላይ በህገ ወጥ መንገድ ግብይት እንደተፈጸመ መናገራቸውን', 13: 'ከክልሉ መንግስት ኮሙኒኬሽን ያገኘነው መረጃ ያመላክታል', 14: 'በቀጣዩ 2015 በጀት ዓመት 15

ነጥብ 665 ቢሊየን ብር ገቢ ለመሰብሰብ ማቀዱ ታውቋል'}, 'Mentions': ['በደቡብ ክልል']}, 'the new set': {'Head_Line': 'በደቡብ ክልል ባለፉት 11 ወራት ከ11 ነጥብ 6 ቢሊየን ብር በላይ ገቢ ተሰበሰበ', 'By_line': 'fana_tv', 'Lead': 'በደቡብ ክልል', 'Facts': {1: 'ባለፉት 11 ወራት ከ 11 ነጥብ 6 ቢሊየን ብር በላይ ገቢ መሰብሰቡን', 2: 'የክልሉ ገቢዎች ቢሮ አስታወቀ', 3: 'የቢሮው ኃላፊ ወይዘሮ ዘይቱና ኢብራሂም', 4: 'በበጀት ዓመቱ 13 ነጥብ 1 ቢሊየን ብር ለመሰብሰብ ታቅዶ', 5: '11 ወራት ከመደበኛ እና ከማዘጋጃ ቤት ከ 11 ነጥብ 6 ቢሊየን በላይ ብር ገቢ መሰብሰብ ተችሏል', 6: 'የህግ ጥሰት በፈጸሙ ሰራተኞች ላይ ህጋዊ እርምጃ መወሰዱን ጠቁመዋል', 7: 'የሚቀሩ ጉዳዮች መኖራቸውን ተገንዝበናል ብለዋል', 8: 'በፈጻሚው እና በግብር ከፋዩ ዘንድ የሚስተዋሉ የስነ ምግባር ጥሰቶችን በመለየት የእርምት እርምጃ ተወስዷል', 9: 'ሀሰተኛ ደረሰኝ ማቅረብ', 10: 'እና ግብር መሰወር', 11: 'ላይ ያሉ ችግሮችን የማረም ስራ እየተሰራ መሆኑን ጠቁመዋል', 12: 'በክልሉ ከ 1 ሺህ በላይ ሀሰተኛ ደረሰኝ የተገኘ ሲሆን ከ 187 ሚሊየን ብር በላይ በህገ ወጥ መንገድ ግብይት እንደተፈጸመ መናገራቸውን', 13: 'ከክልሉ መንግስት ኮሙኒኬሽን ያገኘነው መረጃ ያመላክታል', 14: 'በቀጣዩ 2015 በጀት ዓመት 15 ነጥብ 665 ቢሊየን ብር ገቢ ለመሰብሰብ ማቀዱ ታውቋል'}, 'Mentions': ['አዲስ አበባ', 'በደቡብ ክልል', 'ወይዘሮ ዘይቱና ኢብራሂም']}, 'match': [0.26, 0.6, 0.4, 1.0, 0.59], 'comparitions_of_facts': [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]}

2 10 True
final _State {'Head_Line': 'በደቡብ ክልል ባለፉት 11 ወራት ከ11 ነጥብ 6 ቢሊየን ብር በላይ ገቢ ተሰበሰበ', 'By_line': 'fana_tv', 'Lead': 'በደቡብ ክልል', 'Facts': {1: 'ባለፉት 11 ወራት ከ 11 ነጥብ 6 ቢሊየን ብር በላይ ገቢ መሰብሰቡን', 2: 'የክልሉ ገቢዎች ቢሮ አስታወቀ', 3: 'የቢሮው ኃላፊ ወይዘሮ ዘይቱና ኢብራሂም', 4: 'በበጀት ዓመቱ 13 ነጥብ 1 ቢሊየን ብር ለመሰብሰብ ታቅዶ', 5: '11 ወራት ከመደበኛ እና ከማዘጋጃ ቤት ከ 11 ነጥብ 6 ቢሊየን በላይ ብር ገቢ መሰብሰብ ተችሏል', 6: 'የህግ ጥሰት በፈጸሙ ሰራተኞች ላይ ህጋዊ እርምጃ መወሰዱን ጠቁመዋል', 7: 'የሚቀሩ ጉዳዮች መኖራቸውን ተገንዝበናል ብለዋል', 8: 'በፈጻሚው እና በግብር ከፋዩ ዘንድ የሚስተዋሉ የስነ ምግባር ጥሰቶችን በመለየት የእርምት እርምጃ ተወስዷል', 9: 'ሀሰተኛ ደረሰኝ ማቅረብ', 10: 'እና ግብር መሰወር', 11: 'ላይ ያሉ ችግሮችን የማረም ስራ እየተሰራ መሆኑን ጠቁመዋል', 12: 'በክልሉ ከ 1 ሺህ በላይ ሀሰተኛ ደረሰኝ የተገኘ ሲሆን ከ 187 ሚሊየን ብር በላይ በህገ ወጥ መንገድ ግብይት እንደተፈጸመ መናገራቸውን', 13: 'ከክልሉ መንግስት ኮሙኒኬሽን ያገኘነው መረጃ ያመላክታል', 14: 'በቀጣዩ 2015 በጀት ዓመት 15 ነጥብ 665 ቢሊየን ብር ገቢ ለመሰብሰብ ማቀዱ ታውቋል'}, 'Mentions': ['አዲስ አበባ', 'በደቡብ ክልል', 'ወይዘሮ ዘይቱና ኢብራሂም']}