**ARTIFICIAL INTELLIGENCE**

**ECE 569A**

**KNOWLEDGE GRAPH EXTRACTION**

**SUBMITTED BY**

**Hridya Divakaran - V00970167**

**Kaushik Jeyaraman - V00964926**

**Neeraja Neelakantan - V00966268**

**Table of Contents**

[**LIST OF FIGURES** 3](#_heading=h.gjdgxs)

[**LIST OF TABLES** 3](#_heading=h.30j0zll)

[**Abstract** 3](#_heading=h.1fob9te)

[**Introduction** 4](#_heading=h.3znysh7)

[**Problem Formulation** 5](#_heading=h.3dy6vkm)

[**Related Work** 6](#_heading=h.4d34og8)

[**Methodology** 6](#_heading=h.2s8eyo1)

[**Data Set** 6](#_heading=h.17dp8vu)

[**NER using CRF** 7](#_heading=h.26in1rg)

[**NER using LSTM** 7](#_heading=h.lnxbz9)

[**Relation Extractor** 8](#_heading=h.1ksv4uv)

[**Knowledge Graph Extractor** 9](#_heading=h.1y810tw)

[**User Interface** 10](#_heading=h.2xcytpi)

[**Results and Discussions** 10](#_heading=h.1ci93xb)

[**Conclusion** 12](#_heading=h.3as4poj)

[**Future Work** 13](#_heading=h.1pxezwc)

[**References** 13](#_heading=h.2p2csry)

# **LIST OF FIGURES**

[Figure 1: Steps in implementing knowledge graph extraction 5](#_heading=h.tyjcwt)

[Figure 2: Our implementation 5](#_heading=h.1t3h5sf)

[Figure 3: LSTM model 7](#_heading=h.3rdcrjn)

[Figure 4 : LSTM Architecture 8](#_heading=h.35nkun2)

[Figure 5 : Relation between Facebook and Mark Zuckerberg 8](#_heading=h.44sinio)

[Figure 6: Relation between Facebook and United States 8](#_heading=h.2jxsxqh)

[Figure 7 : Sample Training Sentences 9](#_heading=h.z337ya)

[Figure 8 : Relation extractor model 9](#_heading=h.3j2qqm3)

[Figure 9 : Knowledge graph extraction 10](#_heading=h.4i7ojhp)

[Figure 10 : Annotated Text 11](#_heading=h.3whwml4)

[Figure 11 : Knowledge graph depicting the relationship 11](#_heading=h.2bn6wsx)

[Figure 12 : Result showing the relation using CRF model (left) and Bi-LSTM model (right) 12](#_heading=h.qsh70q)

# **LIST OF TABLES**

|  |  |
| --- | --- |

Table 1: Accuracy of each model……………………………………..10

# 

# **Abstract**

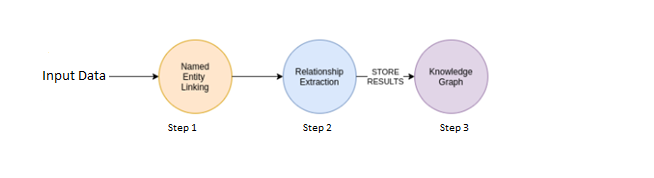
Knowledge graphs extraction identifies entities and relationships between them. It focuses on metadata describing authors, venues, organizations, geographic locations, topics, art etc. It's a new architecture that identifies entities and relationships using Natural Language Processing and Machine Learning techniques, subsequently aggregates them to generate a knowledge graph. The task of recognising key entities of interest (e.g., Organizations, People, Places, political parties, art etc.) from the text is known as the named entity recognition. The nodes in a knowledge graph are comprised of these entities. The task of relation extraction, also known as entity relation extraction, is to extract the relationships between two entities from the text. Relation extraction is sometimes used to extract attributes from a given entity. In our knowledge graph, the retrieved relations and properties will often become relations or node properties.

# **Introduction**

The unstructured text makes up the vast majority of data. As a result, it is critical to maximising its worth by extracting useful information from it. Knowledge graph extraction is a technique that identifies the entities and relations between them. This allows us to turn unstructured text into structured information, which AI may then use to create question-answering models. Knowledge graph extraction is divided into two parts. The first part of the project identifies the different named entities from text data and is known as named entity recognition and the second part of the project is entity relation extraction it finds the relationship between the entities and outputs a knowledge graph with the relations.

Some Natural Language Processing (NLP) activities require Named Entity Recognition (NER). It is important for some Natural Language Processing tasks such as Information Extraction (IE) and Question Answering (QA). On the one hand, deep learning, transfer learning, knowledge bases, and other strategies are widely have been used in NER systems. Its goal is to recognize and classify names of the person (PER), location (LOC), organization (ORG), and numeric expressions including date, currency and percentage [11]. The availability of labelled language resources is critical to NER's methods.

Entity relation extraction is a neural model for relation extraction (RE). It predicts relational facts from plain text. Entity relation extraction not only allows developers to train custom models to extract structured relational facts from the plain text but also supports quick model validation for researchers This model can extract facts in a variety of contexts and align them with Wikidata, which could assist a variety of downstream knowledge-driven applications (e.g., information retrieval and question answering).



*Figure 1: Steps in implementing knowledge graph extraction*

# **Problem Formulation**

A vast majority of data exists as unstructured text. Therefore, it is extremely important to harness its value by extracting meaningful information from it. Knowledge graph extraction is a technique that identifies the entities and relations between entities from raw unstructured text. This helps us convert unstructured text into structured information, and this information can be leveraged by AI to build models for Question-Answering.

We implemented this project in 2 major parts. The flowchart of our implementation is shown in Figure.

Diagram

Description automatically generated

*Figure 2: Our implementation*

* Named Entity Recognition (NER) – The first part of the project involves identifying the named entities from raw unstructured data of the text. NER is a type of information extraction technique that recognizes and classify the named entities in the text [2]. Named entities are categories of real-world objects such as person, money, organization etc. [3]. We have implemented 2 different NER models which are Bidirectional Long Short Term Memory Network trained using Keras and Conditional Random Fields trained using Sklearn Library. Both the models were trained on the NER dataset available in Kaggle [4]. 1 of the models is chosen for entity recognition by the user, which then outputs the annotated text with the identified named entities.

1. Bidirectional Long Short Term Memory Network (Bi-LSTM)

2. Conditional Random Fields (Sklearn’s CRF)

* Entity Relation Extraction – The second part of the project is to identify relations between the named entities. From the entities that were identified from the NER model, the relationship between the entities is identified in this part. The model outputs the final knowledge graph depicting the relations between the entities as a directed graph structure. This part is implemented using the BERT model and Softmax Neural Network. The dataset used for this part is the Wiki80 dataset [5].

# **Related Work**

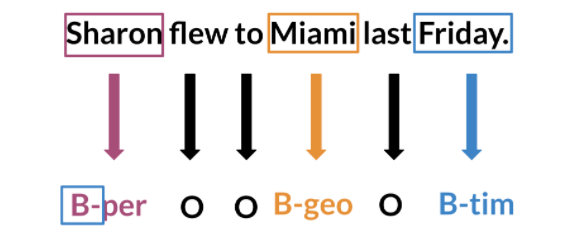
A knowledge graph is a large-scale semantic network that reflects knowledge engineering in the big data era. Knowledge graph extraction is based on two separate problems that is named entity recognition and entity relation extraction. Our neural network is inspired by the work of Collobert et al. (2011b), where lookup tables transform discrete features such as words and characters into continuous vector representations, which are then concatenated and fed into a neural network. Instead of a feed-forward network, we use the bi-directional long-short term memory (BLSTM) network. To induce character-level features, we use a convolutional neural network, which has been successfully applied to Spanish and Portuguese NER (Santos et al., 2015) and German POS-tagging (Labeau et al., 2015)[11].

At present, the in-depth integration of the knowledge graph with various fields and industries has become an important trend. In terms of domain knowledge graphs, representative studies abroad include GeoNames, which was an open global geographic knowledge graph, covering more than 250 countries and more than 10 million pieces of geographic location information[15].

# **Methodology**

## **Data Set**

The data set used for the NER model has one million data. These are divided into four columns namely ‘ Sentence', 'Word', 'POS', 'Tag’. Column word contains English dictionary words forming sentences, column POS is an abbreviation for Parts of speech and tag column represents standard named entity recognition tags for example organization, person, location etc.[IOB](https://en.wikipedia.org/wiki/Inside%E2%80%93outside%E2%80%93beginning_(tagging)) (inside, outside, beginning) tagging format for tagging tokens is also used where I is a prefix before a tag indicates that the tag is inside a chunk, B is a prefix before a tag indicates that the tag is the beginning of a chunk and O tag indicates that a token belongs to no chunk (outside).



*Figure 3: LSTM model*

## **NER using CRF**

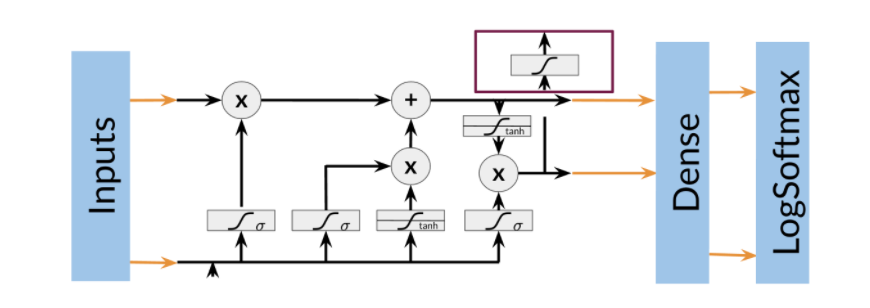
Conditional Random Forests (CRF) is a type of Machine learning algorithm and can be used for Named entity recognition, labelling sequential data etc. [7]. For developing the second model for identifying the entities, we have used the Sklearn library’s sklearn-crfsuite package. The Python code for CRF model creation is shown in Figure. The CRF model was trained for 100 iterations and the dataset used for training was the NER dataset from Kaggle which consists of 47,958 sentences. The NER dataset was split with 90% of the dataset for training and 10% of the whole dataset for testing. The training took approximately 5 minutes.

The trained CRF model was then saved as “ner\_model\_trained.pkl” in the pickle format using the library Joblib. This ensures that this saved pickle model can be loaded and reused for testing purposes for better efficiency. When the user selects the model as a CRF model from the dropdown menu and enter the input text, in the web application backend, the CRF model is loaded, and the entities are identified using this loaded model.

## **NER using LSTM**

Long Short Term Memory Networks (LSTMs) are a type of RNN that can learn long-term dependencies. Named Entity Recognition (NER) is implemented using lstm’s locates and extracts predefined entities from text. It allows you to find places, organizations, names, times and dates. Processing data is one of the most important tasks when training AI algorithms. For NER, we converted words and entity classes into arrays and then we padded with tokens to set sequence length to a certain number and use the <PAD> token to fill empty spaces then we created a data generator:

For training, a NER system we created a tensor for each input and its corresponding number was put in a batch and feed into an LSTM unit for running the output through a dense layer and it is predicted using a log softmax over K classes.



*Figure 4 : LSTM Architecture*

## **Relation Extractor**

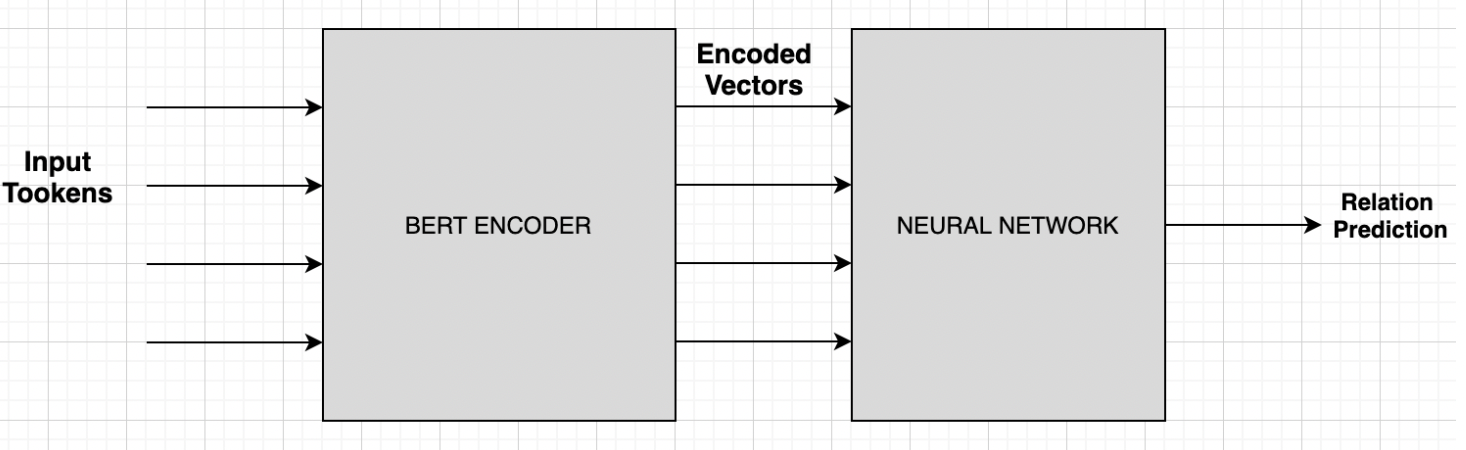
Relation extractor is the task of finding relationships between two given entities.

| *Figure 5 : Relation between Facebook and Mark Zuckerberg* | *Figure 6: Relation between Facebook and United States* |
| --- | --- |

This model is developed on the wiki80 dataset, It contains 56,000 manually sentences from Wikipedia articles which are annotated with NER and then manually labelled with a relationship. There is 80 such unique relationship, thus the name wiki80. These labelled relations are the target labels that need to be predicted by our machine learning model.

| *Figure 7 : Sample Training Sentences* |
| --- |

In any machine learning algorithm, we need to pass a set of features to be able to make a prediction. These features are usually represented in rows and columns. But in the case of language models, we just have raw text. So these texts are converted into features with a vector representation. For our module, we use the most popular language model encoder, i.e the BERT encoder. This outputs numerically encoded vectors. These vectors are used to train a neural network module to predict a relationship. For the simplicity of the project, we used a pre-trained BERT model.



*Figure 8 : Relation extractor model*

## **Knowledge Graph Extractor**

First, we preprocess text to remove special characters, then we pass it to Named Entity resolution (NER) which finds out the entities and gives the indices of the entities in the text. These indices are passed on to the relationship extractor and this tries to find the relationship between two given entities and if it can predict a relationship with a confidence above the threshold then we store this relation. We combine all the entities and relations to form a connected network know as a knowledge graph. We output the final graph on UI developed with the python-streamlit library.



*Figure 9 : Knowledge graph extraction*

## **User Interface**

As our final implementation of the project, we have developed a Web Application using the Streamlit Library in Python. Streamlit is an open-source framework for creating web applications for Machine Learning and Data Science related works. The user can enter the text/ sentence for extracting the named entities and relations between the entities in the text area of the web application (Figure 1). There is a dropdown menu for the user to select either of the 2 Named Entity recognition models, the CRF or the Bi-LSTM Keras model (Figure 2). Once the dropdown is selected and a text is entered, pressing the “Submit” button will result in 2 outputs being displayed. First, the annotated text with the identified named entities will be displayed and then the knowledge graph will be displayed. For displaying annotated text with highlighted named entities with different colours, we have used the Annotated Text component for Streamlit [8]. The knowledge graph is displayed as a directed graph using the “Graphviz” package.

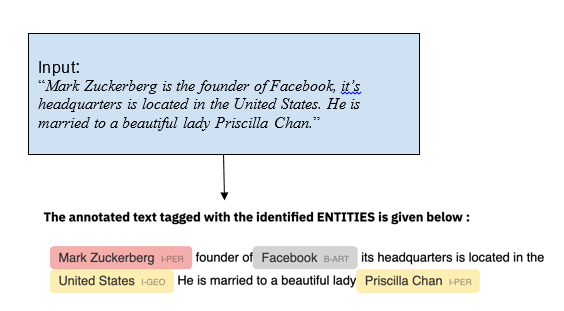
All the codes including the NER component, Relation extraction code, the web application etc. were implemented using the Python Programming Language. We have used Sklearn, Keras, TensorFlow, Streamlit etc. to build these models and applications.

# **Results and Discussions**

*Table 1: Accuracy of each model*

| Models | Training Accuracy | Test Accuracy |
| --- | --- | --- |
| NER – CRF | 82% | 70% |
| NER – Bi LSTM | 98% | 95% |
| Relation Extractor | 98.57% | 88.57% |

As we can see individually accuracy of each of the models was good. We were unable to save and load the model for NER-Bi-LSTM, it was giving strange results. Due to time constraints, we couldn't solve it. In other models, we were able to successfully save and load. Despite getting good accuracy for individual models, the combined NER and Relation Extractor performs poorly due to the incompatibility of the datasets on which these were trained. Our NER is capable of identifying only 8 different entities and the Relation extractor is capable of identifying 80 relations. Only the cases where the entities and relations are similar to the ones in the models. In other cases, it performs poorly.

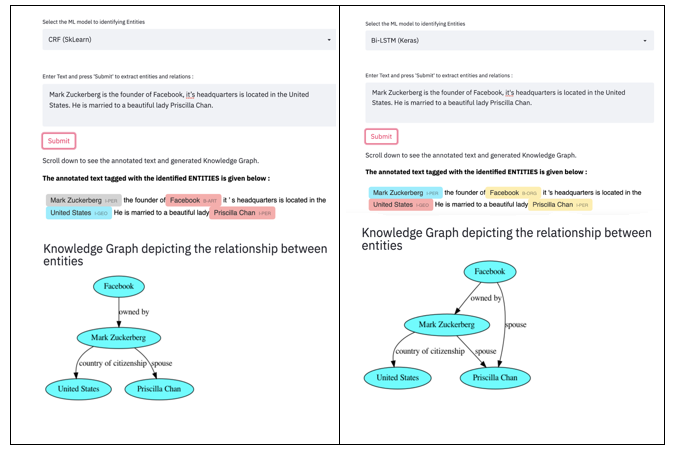


*Figure 10 : Annotated Text*

Diagram

Description automatically generated

*Figure 11 : Knowledge graph depicting the relationship*



*Figure 12 : Result showing the relation using CRF model (left) and Bi-LSTM model (right)*

# **Conclusion**

This report describes the development of the Knowledge Graph Extraction technique for identifying the named entities and relations between entities from raw unstructured text. The first part of the project involves the named entity recognition task which was implemented using 2 different approaches. Both the Bi-LSTM and CRF model has pros and cons and would require further optimization, ensemble techniques or other methods for improving the performance and accuracy. Overall, both models correctly identified most of the entities from the input sentences. The second part of the project is to extract the relationships between the identified entities and output the associated knowledge graph. The relation extractor model managed to identify some of the relations accurately while some were incorrect. Overall, the performance of the relation extractor is very good than expected since the NER and relation extractor algorithms were trained on 2 different datasets. We believe that the performance of our knowledge graph extractor would greatly improve when trained on the same dataset and also a much bigger and diverse dataset.

# 

# **Future Work**

There are a few limitations and drawbacks of our project which could be improved in future.

2 different train datasets - 1 major drawback of our project is that we have built and trained the NER models and Relation extraction on 2 extremely different input datasets. This implies that our final implementation may not work accurately with many varieties of input sentences.

Time and Speed – The Bi-directional LSTM model developed using Keras and Tensorflow is much slower (~ 20 minutes) while training and testing compared to the CRF model (~ 5 minutes) for the Named entity recognition task.

Accuracy – The CRF model developed using the Sklearn-crfsuite library have much lesser accuracy (70%) compared to the Bi-LSTM model (99%) on the test dataset.

Reproducibility of the results – When the CRF model was saved and loaded separately for testing, we were able to achieve the same accuracy and reproduce the exact prediction results on the web application. But, for the Bi-directional LSTM model, when the model was saved and loaded for testing separately, we were unable to reproduce the same results and the accuracy was bad. The model gave random predictions, and the prediction was not good.

# 

# **References**

[1] https://docs.streamlit.io/en/stable/

[2] https://en.wikipedia.org/wiki/Named-entity\_recognition

[3] https://en.wikipedia.org/wiki/Named\_entity

[4] https://www.kaggle.com/namanj27/ner-dataset

[5] https://thunlp.oss-cn-qingdao.aliyuncs.com/opennre/benchmark/wiki80/wiki80\_val.txt

[6] https://sklearn-crfsuite.readthedocs.io/en/latest/tutorial.html

[7] https://en.wikipedia.org/wiki/Conditional\_random\_field

[8] <https://github.com/tvst/st-annotated-text>

[9]https://towardsdatascience.com/from-text-to-knowledge-the-information-extraction-pipeline-b65e7e30273e

[10] OpenNRE research paper: <https://aclanthology.org/D19-3029/>

[11] <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>

[12] Peng Sun, Xuezhen Yang, Xiaobing Zhao, Zhijuan Wang: “An Overview of Named Entity Recognition”, 2018

[13] James Hammerton, Alfa-Informatica : “Named Entity Recognition with Long Short-Term Memory”, 2018

[14]https://web.stanford.edu/class/cs520/2020/notes/How\_To\_Create\_A\_Knowledge\_Graph\_From\_Text.html