CSCE 5290 - Natural Language Processing
Named Entity Recognition using BiLSTM
Project Report Group 8

Introduction

When we consider natural language processing we need a mechanism that basically works to recognize, observe and extract certain kinds or types of entities in the text that need to be processes, the entities can be as basic as possible and can be something like the *people* or the *names of places*, or the *addresses*, and all the other thing that come under these categories. To get this kind of classification Natural Language Processing provides a model name as Named Entity Recognition. This is an important feature in the searching mechanism on the web, let us say that the search engine does not understand what is being asked of in the start then the entire purpose of the search engine has failed. Now if there is a method like the Name Entity Recognition then the search engine would understand what an entity is and what is being searched by the user and then the results can be delivered to the user. It is to be noted that most of the search engines that are available generally resort to the *keyword search* method and then the mechanism that is followed is that the entire document is searched for the relevance of the keyword and then the score is kept in a record where the number of hits of a particular word would then be weighed by a factor called the TF-IDF. This method works fine but its efficiency can be increased when the feature of entity recognition is added to it, this will increase the relevance of the search and the search engine would also return key features about the entity after it has been recognised.

Now there is another feature that is needed is the extraction of the views in the spoken language and the written script language as well, this can be termed as Sentiment Analysis(SA). To better understand what sentiment analysis is you need to understand a little about what Text Mining is mainly because the former(sentiment analysis) is one of the types of the later(text mining). So, text analysis is nothing but the process of extracting information that is essential from basic language data that is in the form of text. This data can be anything from a text message to an email written in common language. It is mainly used in extracting patterns from such data. Now let us get back to Sentiment Analysis - it is basically a type of text mining that is used to find and extract information that is subjective from a source. This extraction helps businesses to better understand the social sentiment that is what people think about their brand, product or any services while checking the online conversations. Usually, this analysis confines to simple sentiment analysis and count-based indicators, which in turn leads to deep learning algorithms which enhance the capacity to analyze text data. In short, this method of analyzing and identifying the positive and

negative sentiments in text is known as sentiment analysis. Most businesses use this because it lets them better understand its consumers. SA uses natural language processing(NLP) songs with machine learning techniques for sentiment evaluations to themes and categories in a sentence or phrase.

LSTM(Long Short Term Memory) is a part of machine learning and deep learning, we are using LSTM for our sentiment analysis. So, let us first see what long short term memory does, LSTM mainly has the algorithms that can mirror or mimic the human brains and bring out important relationships from data. Many RNNs are capable of learning long-term dependencies. It is an advanced RNN and allows information retention. Now comes Bidirectional LSTM (biLSTM). It is basically a sequence processing model which consists of two LSTMs where one takes input forward while the other takes it in the backwards direction. Why do we need to use them? BiLSTMs are very effective to increase the amount of information available to the network which improves the content available to the algorithm making it work better. Both LSTM and biLSTM are used when the learning problem is sequential.

So coming to this project we are looking to incorporate the same two models or mechanisms into data that would be available to us and then classify the data. The classification of the data is also an important factor as both the NER and the BiLSTM use entities as their basic structure. We are looking to incorporate rule based entities as well to improve the relevance and the accuracy of the classification. With the use of the above models the macro average is the one that would be affected the most and we have tried to implement a method that can be used to improve the macro average of the model. Basically when we are trying to hyper tune the model that we have built we can actually make changes to macro average. Once the macro average has been improved to a certain level that is satisfactory, we would do BERT on the dataset to use transformers on the same task that was performed and achieved previously. With the help of the BERT and the transformers we will try to achieve the best entity recognition with the much improved macro average. Finally the rule based entity recognizer can be used to further enhance the features of the project. The addition of the rule based entity recognizer will enhance the project by increasing the ability to learn and classify the new entity by setting the custom rules.

Background

There were times when it was only the age of newspapers and the world population would sit and read the newspapers in the morning, but as the world moved towards technology then newspaper sales went down and the digital ages of the newspapers and articles began. It was during this time that the digital media took over the world and the thirst for digital content has increased. This thirst has not reduced but has grown exponentially and has been increasing even since. Now when we were learning the basics of any language we would be able to understand the context of the sentence and then the brain processes it in such a manner that we understand the meaning of the words and the connection between each word in the sentence and entirely the meaning of the the

entire passage that we are reading at that particular moment. Now when it comes to the digital media all the data in that particular context has to be sorted and then arranged by an editor before it is sent to the public so that they would be able to read and understand the entire meaning of the passage.

The digital content when written is often sorted and cleansed of all the improper formation of the sentences and the words but when previously it was all not possible. Identification of the words and the nouns and the adjectives were all placed in the mechanism for the recognition but it lacked overall and was not efficient, for this there were developments in the field of Artificial Intelligence and in particular the field of Natural Language Processing and there was a mechanism that was built in order to get the entities like the name s of the people, the addresses of the places, the traits of the human, and the different kinds of food that people around the world. Further the use of the BiLSTM also has increased the accuracy of the recognizing the entities in the generall text and the process of the information according to the need of the user.

We when we were in the initial stages of thinking about the ideal of the project were not able to wrap our heads around one particular thing that would actually help us in implementing the project and then make us understand the concepts behind the project. So after a lots of through process were finally narrowed down our thoughts to that we would be using the NER, BiLSTM, BERT and then Rule based entity recognition mechanism on the datasets that were available to us and then we would understand what it is like to actually have first hand look at the contents and the steps involved on the recognition process of the data. We were fortunate for the opportunity provided so that we were able to implement this project and increase the spectrum of our knowledge.

Model

For the implementation of the project we have used a combined 4 technologies or mechanisms to help in the recognition of the entities and to the improvement of the macro average of the model that was built. The mechanisms that were used are Named Entity Recognition(NER), Long Short Term Memory(LSTM), Bidirectional Encoder Representation using Transformers(BERT) and the Rule Based Entity Recognition. All of these each have their own specification but when all of them are applied on a data model that has been trained then we would get the results that are actually excellent when compared to the other models that are available in the market.

Coming to understand what the project does we will use a diagram and try to explain what is actually happening in the background. When we consider the data that we have on our hands we would probably classify the data in the order of the names, places, things, animals, etc. To perform all of these tasks NER would be helpful and the way that NER works also forms a good level of recognition and as we are combining the NER with LSTM and then doing the BERT with Transformers we are able to achieve results that would help in the entity classification efficiently.

So what is an entity is what needs to be understood in the first place. Let us see that with an example.

Figure 1 : Understanding the entities

In the above diagram there are a total of 5 blocks that have different words in them and each of the words together form a sentence. Now from the sentence we as a human and a natural language reader who can understand the different forms of the words and their type would be able to distinguish the difference of what is a noun and what is a connected word, but when we consider Artificial Intelligence it is not sure of what is what. As I have labeled the blocks in the above diagram i.e. **Peter** and **Dallas** the usage of NER and the LSTM and along with the others also would help the AI understand words like these are something that is very significant of a particular sentence and then need to be distinguished from the others. After the project has been completed and any set of sentences would be given as input then the system would be able to understand what an entity is and then would directly recognize it in the future.

Workflow

Let us now see the flow of the events in the project so as to understand what steps are being performed at what stage and then see what is actually being performed so that each stage is achieved and the results that are achieved are actually very excellent.

The diagram here shows the workflow that is associated with the project and involves all the major steps and keeps aside the finer details of all the other steps in the process of the custom recognition. In the first stage we would input the data in the form of a dataset but before that we have to understand what dataset needs to be input so that the correct recognition can be done. For this project we have searched for various datasets on Kaggle and we have stumbled upon multiple datasets that would work for us but we needed something concrete so that all the entities are correctly matched with their types. Care has also been taken so that the data set has a good collection of words so that the key entities can be actually classified.

Once the dataset was identified we put into a data cleaning process which would involve various

steps of processing out of which one key step is to remove the null values in the datasets, these null values are not good for any estimates and would create a bias in the results and hence we have used the method of ffill.

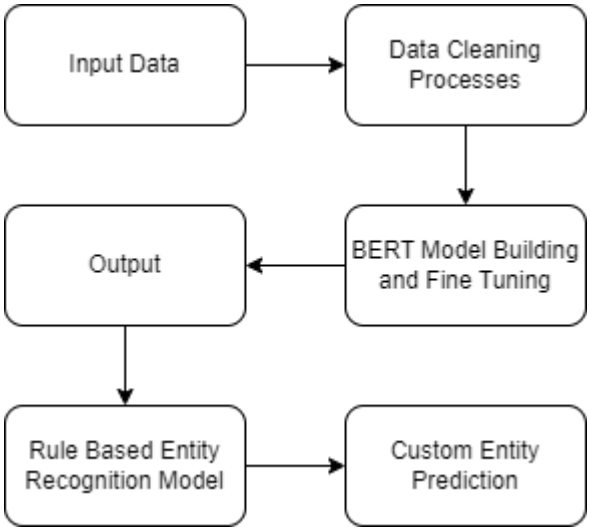

Figure 2 : Workflow Diagram

Once the Data Cleaning process has been completed we have trained and modeled the datasets and divided them into class. Next we have used the BERT method to build the model, in simple terms the BERT mechanism is used to model data in a semi-supervised training, i.e. compare the small data with reference to a much larger data. Once the BERT modeling has been completed we have performed other fine tuning methods so that the other problems that may come up in the data set training and modeling can be fixed.

The output achieved is then sent to the Rule Based Entity Recognition and then the output achieved will be according to the rules set by us. The end result would be that the prediction made would be according to the rules set in the final stages of the project and all of which would help in getting positive results for the dataset.

Dataset

We would use one publicly available dataset : CONLL 2003 (English) for our project. We will try to make some modification to the dataset for our task by doing some pre-processing. We intend to tag a given sequence by using our model and assign labels like Person (P), Location(L), Origination (O). We would also try to assign undecided words into a Miscellaneous (M) label and create a “Not sure” label for such complex words. We aim to implement a pre-trained model to assign proper or accurate labels to these “Not-sure” words for the sake of improving our proposed project accuracy. We would also employ new ideas

and algorithms upon recommendations and results.

The dataset that is being used in this project has the following attributes associated is feature engineered corpus annotated with the help of IOB and POS tags. There are a total of 47959 sentences in the dataset and out of which all the sentences are unique in nature and not a single sentence has been repeated in the entire dataset. Now coming to the words present in the dataset, there are a total of 1048575 words and coming to the unique words then they are scaled down to 35178. On further exploration of the dataset we were able to see that the dataset has 4 columns or we can call them features and a total of 1048575 rows. We have also noticed a lot of null values as well in the dataset. These null values need to be filled with some random values so that there would be no bias while generating the results. The accuracy of the result would be highly impacted by the presence of Null Values(NaN) values.

Analysis of Data

The dataset that we are using for this project has four main columns which act on the features of the dataset which in turn play a major part in the implementation of the project. The four major factors/features associated with the dataset are:

- Sentence : This feature is a combination of words and forms the crux of the dataset that is available for the modeling, training and testing the dataset. Each sentence in the dataset has a lot of NaN(Null) Values which need to be cleaned.

- Words : This is the main feature of the dataset as we would be using this feature in our dataset to tag this particular sentence to which the named entity is related.

- Parts of Speech(POS) : This feature is basically telling us the particular part of speech the word that was extracted is related to in the English Language. This feature is very useful when we need to do sentiment analysis.

- Tag : This is another key feature that would define the level of class in the dataset. In the current dataset there are 17 class levels and all of which are very useful in training and testing the model.

The dataset that we are currently using has a total of 1048575 rows and 4 columns, these 4 columns act as the entities or the features related to the dataset. The Features of the dataset are Sentence, Word, POS and Tag and the most important feature of the dataset is the words feature which would be basically used to build the Bi-LSTM model. More on the dataset is that each row in the dataset is a complete sentence and the other two columns are the POS and the tag columns which would be unique to each word of the sentence. There are a lot of null values in the column sentence which needs to be filled to avoid bias in the model result. Now when coming to the target data column, it would be the column tag.

Implementation

The dataset that we are currently using to implement the project contains a total of 17 tags that would be sufficient to get accurate results. In the initial steps of the implementation we have first displayed the first few lines of the dataset to see the contents of the dataset. Next upon the next steps we have classified the dataset to check the total count of the dataset uniqueness of the words that are present in the dataset. Once the total number of the words and the uniqueness of the dataset has been determined we have seen that there are a lot of NaN values that are actually present in the data set.

```
[ ]  print(data)

             Sentence #           Word  POS Tag
0        Sentence: 1.0      Thousands  NNS   O
1                 NaN             of   IN   O
2                 NaN  demonstrators  NNS   O
3                 NaN           have  VBP   O
4                 NaN        marched  VBN   O
...               ...            ...  ...  ..
1048570           NaN           they  PRP   O
1048571           NaN      responded  VBD   O
1048572           NaN             to   TO   O
1048573           NaN            the   DT   O
1048574           NaN         attack   NN   O

[1048575 rows x 4 columns]
```

Figure 3 : Code Snippet

There are many NaN values present in the column sentence and the first step that was supposed to be done is to fill the NaN values. The NaN values need to be changed as they may cause a bias in the results. We have checked and calculated the NaN(null) values in the dataframe.

```
# check how many rows have NaNs
data.isnull().sum()

Sentence #    1000616
Word                0
POS                 0
Tag                 0
dtype: int64
```

Figure 4 : Code Snippet

Now in order to get rid of the null values we have used the method of ffill, the name of the method is forward fill which basically propagates the value forward. The understanding of this is that the

null values will be filled or replaced with the last valid observation or entry in the dataframe.

```
[ ]  data = data.fillna(method='ffill')
     data.head(50)
```

	Sentence #	Word	POS	Tag
0	Sentence: 1	Thousands	NNS	O
1	Sentence: 1	of	IN	O
2	Sentence: 1	demonstrators	NNS	O
3	Sentence: 1	have	VBP	O
4	Sentence: 1	marched	VBN	O
5	Sentence: 1	through	IN	O
6	Sentence: 1	London	NNP	B-geo
7	Sentence: 1	to	TO	O
8	Sentence: 1	protest	VB	O
9	Sentence: 1	the	DT	O
10	Sentence: 1	war	NN	O

Figure 5 : Code Snippet

Next after the NaN values have been filled we would be grouped by the tags/labels, after that we extract the words and create a list of words from all the sentences and corresponding tags for the sentences.

```
[ ]  word_list = list(set(data['Word'].values))
     number_of_words = len(word_list)

[ ]  list_of_tags = list(set(data['Tag'].values))#iob2 tags
     number_of_tags = len(list_of_tags)

[ ]  number_of_words, number_of_tags

     (35178, 17)
```

Figure 6 : Code Snippet

Next steps would be to assign a unique id for each word so that it would be useful for mapping for the training and testing of the model dataset.

```
[ ]  word_id = {w: i for i, w in enumerate(word_list)}
     print (word_id)

     {'neighbours': 0, 'one': 1, 'Simkins': 2, 'Hermosa': 3, 'provides': 4

[ ]  tag_id = {t: i for i, t in enumerate(list_of_tags)}
     print(tag_id)

     {'O': 0, 'B-art': 1, 'I-per': 2, 'B-eve': 3, 'I-tim': 4, 'I-nat': 5,
```

Figure 7 : Code Snippet

Next we would plot the lengths of the sentences in our dataset so as to understand the volume of the sentences in the datasets present.

```
[ ]  import matplotlib.pyplot as plt
     plt.hist([len(s) for s in sentences], bins=50)
     plt.show()
```

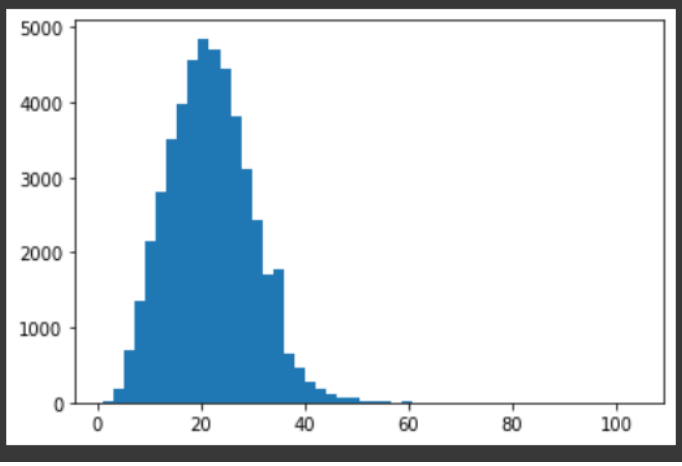

Figure 8 : Code with Graph Snippet

Next we built and then trained the model by splitting the dataset into 3 classes.

```
[ ]  import tensorflow as tf
     #https://www.tensorflow.org/api_docs/python/tf/keras/utils/to_categorical
     from tensorflow.keras.preprocessing.sequence import pad_sequences#add numbers to the sequences to make them all be the same length.
     from tensorflow.keras.utils import to_categorical

     max_len = 50
     X = [[word_id[w[0]] for w in s] for s in sentences]
     X = pad_sequences(maxlen = max_len, sequences = X, padding='post', value=number_of_words-1)
     y = [[tag_id[w[2]] for w in s] for s in sentences]
     y = pad_sequences(maxlen = max_len, sequences = y, padding = 'post', value = tag_id['O'])
     y = [to_categorical(i, num_classes=number_of_tags) for i in y]
```

Figure 9 : Code Snippet

To implement the next part we have used the BERT model which is basically a transformer and for this we have used the 2.6 version of pytorch as it was a stable version and had no dependencies. We have also used the GPU to run the code and the cuda as well. cuda was used for fine tuning.

```
[ ]  bert.cuda();
```

```
[ ]  #taken from BERT-finetuning documentation

    FULL_FINETUNING = True
    if FULL_FINETUNING:
        param_optimizer = list(bert.named_parameters())
        no_decay = ['bias', 'gamma', 'beta']
        optimizer_grouped_parameters = [
            {'params': [p for n, p in param_optimizer if not any(nd in n for nd in no_decay)],
             'weight_decay_rate': 0.01},
            {'params': [p for n, p in param_optimizer if any(nd in n for nd in no_decay)],
             'weight_decay_rate': 0.0}
        ]
    else:
        param_optimizer = list(model.classifier.named_parameters())
        optimizer_grouped_parameters = [{"params": [p for n, p in param_optimizer]}]

    optimizer = AdamW(
        optimizer_grouped_parameters,
        lr=3e-5,
        eps=1e-8
    )
```

Figure 10 : Code Snippet

Next we have set a padding value to 75 so that all the sentences that would be used in the model will have a fixed length. Next we used a total of 3 epochs and stored the data in each epoch and plotted the graph after each epoch.

```
[ ]  import matplotlib.pyplot as plt
     %matplotlib inline

     import seaborn as sns
     sns.set(style='darkgrid')
     sns.set(font_scale=1.5)
     plt.rcParams["figure.figsize"] = (8,4)
     plt.plot(loss_values, 'b-o', label="training loss")
     plt.plot(validation_loss_values, 'r-o', label="validation loss")
     # Label the plot.
     plt.title("Learning curve")
     plt.xlabel("Epoch")
     plt.ylabel("Loss")
     plt.legend()

     plt.show()
```

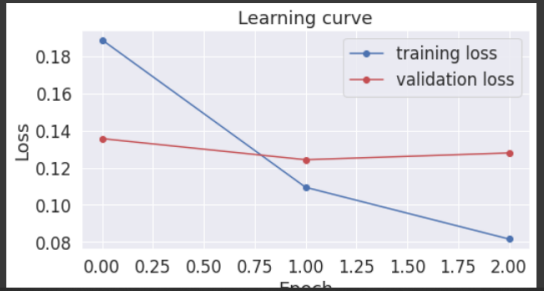

Figure 11 : Code Snippet and Graph Snippet

Next each tokenized word was used and the list was appended. The CPU was used and the preprocessing was done to eliminate bias. There was a minor flaw where one tag was incorrectly recognized then custom rules were set so that further classification could be done with the help of two features namely soccer and cricket.

```
[ ]  #applying new rules
     new_rule = nlp.add_pipe("entity_ruler", before='ner',name="cricket")
     new_rule.add_patterns(pattern_cric)
     new_rule = nlp.add_pipe("entity_ruler", before='ner',name="soccer")
     new_rule.add_patterns(pattern_socc)

[ ]  #check if the new rules were updated
     nlp.pipe_names

     ['tok2vec',
      'tagger',
      'parser',
      'attribute_ruler',
      'lemmatizer',
      'cricket',
      'soccer',
      'ner']
```

Figure 12 : Code Snippet

Results

The primary goal of this project is to form a hybrid model that uses the Name Entity Recognition Model and the same using the Bi-LSTM Model. Now when coming to the dataset we have selected a dataset that needed to be huge in volume so that the model would be better trained. Before we train the model we have to first understand the data that is present on our hands. By the meaning of understanding it means that the dataset needs to be first sorted so as to avoid the null the values to as to avoid the bias. The steps that were followed for this step is the use of forward fill or in short “ffill” method to fill the null values with the preceding acceptable value of the dataset. Next the volume of the sentences i.e. the length of each sentence is then checked just to have a deeper understanding of the dataset.

Once all the pre-steps have been completed the dataset would then be needed to be trained and then tested. Once the training and the testing of the dataset has been completed we would build the Bidirectional LSTM Model. Over here we would notice that the bidirectional model built would have two parameters namely the “Trainable Parameters” and the other being the “Non-Trainable Parameters”, the total number of parameter would be the sum of the Trainable and the Non-Trainable Parameters.

```
 Layer (type)                 Output Shape              Param #
=================================================================
 input_1 (InputLayer)         [(None, 50)]              0

 embedding (Embedding)        (None, 50, 50)            1758900

 spatial_dropout1d (SpatialD  (None, 50, 50)            0
 ropout1D)

 bidirectional (Bidirectiona  (None, 50, 200)           120800
 l)

 bidirectional_1 (Bidirectio  (None, 50, 200)           240800
 nal)

 time_distributed (TimeDistr  (None, 50, 17)            3417
 ibuted)

=================================================================
Total params: 2,123,917
Trainable params: 2,123,917
Non-trainable params: 0
```

Figure 13 : Accuracy Snippet

Once the model is built we would then need to train the model as well. After the training process has been completed we would evaluate the model that was built and trained.

The following graphs we obtained after the model built was evaluated:

```
[ ]  def plot_graphs(history, string):
         plt.plot(history.history[string])
         plt.plot(history.history['val_'+string])
         plt.xlabel("Epochs")
         plt.ylabel(string)
         plt.legend([string, 'val_'+string])
         plt.show()

     plot_graphs(history, "accuracy")
     plot_graphs(history, "loss");
```

Figure 14 : Code Snippet

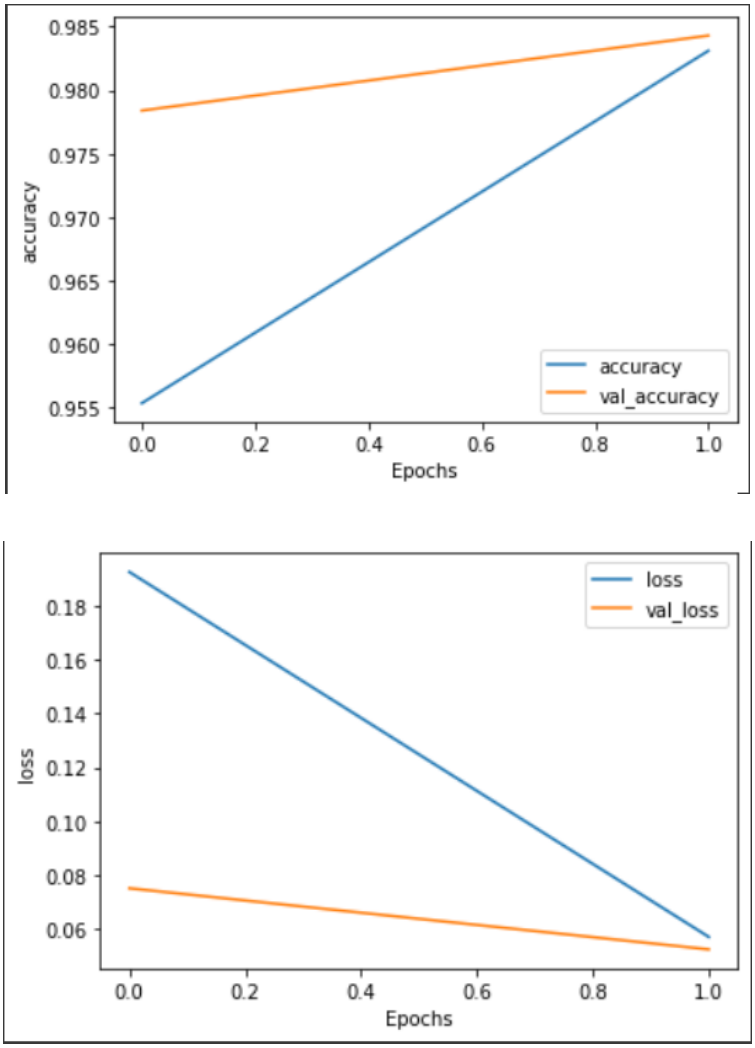

Figure 15 : Epoch Loss Graphs

Now we have to test the evaluated model to see the accuracy of the model. Once the test set has been tested we have achieved an accuracy value of 0.97 which is a very good result but on the other hand there is another factor that plays an important role as well and that is the macro average. This macro average value has been generated to be 0.39 and which is not recommended for the results and a higher macro average would give us an even more accurate result.

	precision	recall	f1-score	support
0	0.00	0.00	0.00	17
1	0.99	1.00	0.99	671658
2	0.83	0.76	0.80	11090
3	0.00	0.00	0.00	62
4	0.92	0.84	0.88	4755
5	0.68	0.78	0.73	5093
6	0.56	0.57	0.56	4917
7	0.73	0.62	0.67	5052
8	0.00	0.00	0.00	140
9	0.00	0.00	0.00	128
10	0.90	0.68	0.78	6118
11	0.00	0.00	0.00	101
12	0.00	0.00	0.00	60
13	0.88	0.43	0.58	2194
14	0.67	0.03	0.06	1950
15	0.00	0.00	0.00	113
16	0.50	0.58	0.53	5952
accuracy			0.97	719400
macro avg	0.45	0.37	0.39	719400
weighted avg	0.97	0.97	0.97	719400

Figure 16 : Macro Average Results

The next phase of the project involved the BERT transformer, the data frame was used to load the dataset and then some preprocessing steps were performed on the dataset so that no bias would be created on the same. Basically the BERT was used for tokenizing and pre trained BERT tokenizers were used.

```
[ ]  def tok_with_label(sentence, text_labels):
         tokenized_sent = []
         labels = []

         for word, label in zip(sentence, text_labels):

             # Tokenize the word and count # of subwords the word is broken into
             tokenized_word = bert_tok.tokenize(word)
             n_subwords = len(tokenized_word)

             # Add the tokenized word to the final tokenized word list
             tokenized_sent.extend(tokenized_word)

             # Add the same label to the new list of labels `n_subwords` times
             labels.extend([label] * n_subwords)

         return tokenized_sent, labels
```

Figure 17 : Code Snippet

Then there were a few steps before plotting the loss, to plot the average loss we have used epochs up until the number 5 but after the number 3 there was not much of a distance and hence we have only used 3 as the epoch for our case. The average loss that was attained was plotted as below.

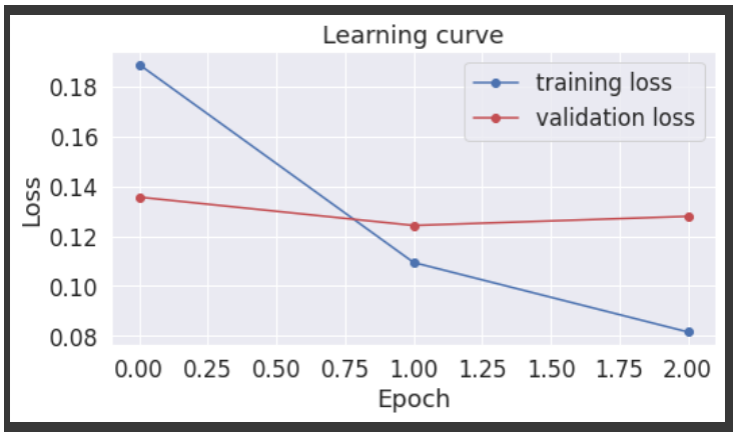

Figure 18 : Learning Curve

Then the data was loaded and the model was run and the results looked into and then we visualized the results to better understand the outcome.

We can see from the output in the below image that the data that was sent as input has been classified into different types of entities. The different kinds of entities that were used in the data that were corresponding to it were the names of the player and the name of the country a player belonged to and there was also an entity that mentioned the Continent to which a player would be playing under with respect to their country. It has to be noted that the dataset that was used here was personally created so that most of the text in the data would have more number of the entities so that the entity recognition can be performed efficiently.

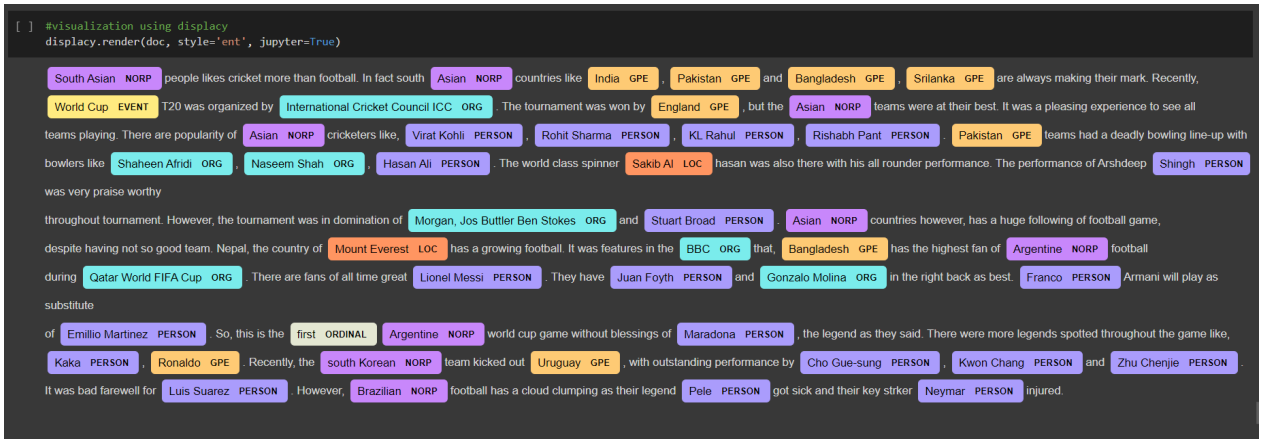

Figure 19 : Entity Recognition

Here the Purple represents the continent, turquoise represents organization, dark purple blue represents person and yellow represents country. But here there were few improper classifications and hence rules were set up and the classification was subclassified and then gave better results.

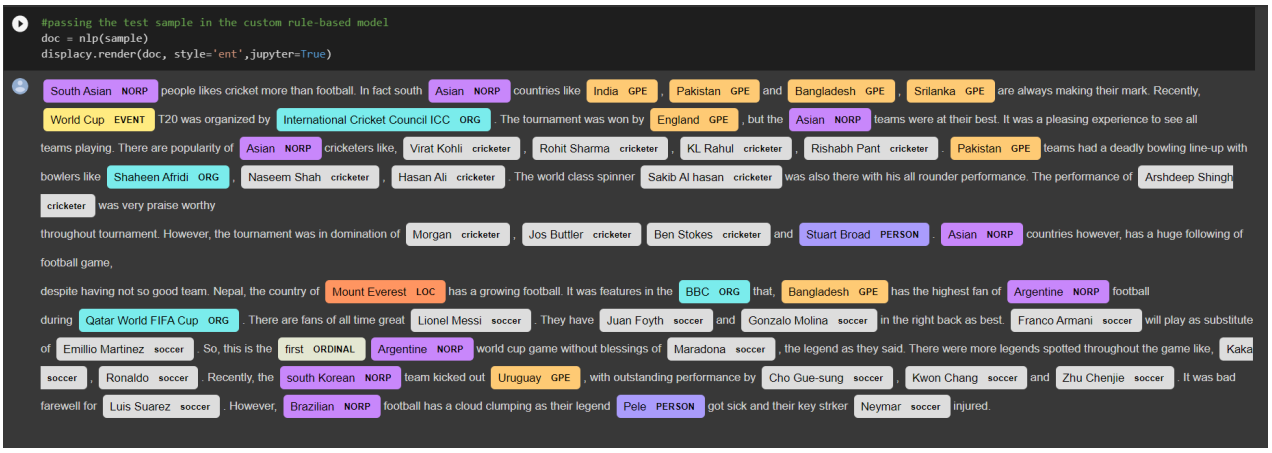

Figure 20 : Rule Based entity recognition

Here each entity person is either assigned to cricket or soccer with respect to the sport they play.

Responsibility and Contribution
Project Management:

Implementation Status Report

- **Work Completed:**
  - **Description**

We have begun our project by selecting the dataset that was correct and related to the project that we were trying to implement. The dataset that was selected was in the format of the .csv format. Next the selected dataset was put into an essential process of cleaning. The cleaning steps here were to get rid of the NaN(null) values and fill them with some valuable arbitrary values. For this we have used the method of forward fill. Next we have built the bidirectional LSTM Model and trained and evaluated the model to predict the accuracy of the model built.
Then in the next increment we wanted to have a wider spectrum of analysis and then we looked into various ways of how we can achieve that. Our initial target was to get the macro average to classify the data correctly. For this we have finally decided that we would be using the BERT transformer to model our data and then set a few rules using the rule based custom entity model as there may be few unforeseen deviations that may arise in the model. The final setting that was done is that once a data was provided the data was classified according to the entities and when the rules were set they were classified as per the classification.

  - **Responsibility(Task, Person)**

<u>Venkata Saran Praneeth Koduru</u>

- Researched the appropriate dataset that would be essential for the project.
- Split the dataset so the steps of training and testing the model can be performed.
- Performed the tasks of understanding the dataset to assign labels/tags to each word so as to count the number of words.
- Changed the network so that the macro average can be changed and would see a possible positive increase in the macro average.
- Looked into methods to better improve the classification that was achieved.
- Custom Rule Based Entity mechanism was performed to further improve

the classification.
- Took the task of receiving feedback from the correct people towards the betterment of the project.
- Completed all the documentation related to the project with all the necessary materials and references.

Farhad Md Mokter

- Analyzed the dataset to check if there are any cleaning steps that need to be completed on the dataset before training and testing the model.
- Removed the NaN values from the dataset and filled them with the appropriate values using the method called forward fill.
- Built the bidirectional LSTM model and evaluated the model to generate the accuracy of the model.
- Created the list of words from all the sentences and corresponding tags for the sentences.
- Performed the task of giving each word and tag an index that is unique, so basically defined the mapping between the sentences and tags.
- Implemented the BERT modeling on the dataset and performed the fine tuning mechanisms.
- Checked and implemented epochs that were needed to get an accurate measurement of the data and validation loss.
- Spearheaded the team and was the primary motivator to push towards the results.

○ Contributions(Members/Percentage)

Farhad Md Mokter	**50%**
Venkata Saran Praneeth Koduru	**50%**

■ Issues/Concerns

- There are a lot of samples in one particular class when compared to the other classes. In our case, the class is the Class 2 where there are considerably more samples when compared to the other classes.
- There is a class imbalance due to the unequal samples from each section of classes through the observations vary from one class to another.
- The class-wise performance is very effective but when all the classes are clubbed the performance is comparatively lower than individual classes.

- The classification that was achieved was not a complete perfect success as there were few or rare entities that were improperly classified.
- Correct rules needed to be designed so that subclassification would have been done correctly.

References

1. Mayhew, Stephen, Gupta Nitish, and Dan Roth. "Robust named entity recognition with truecasing pretraining." *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. No. 05. 2020.
2. Thanh Hai Dang, Hoang-Quynh Le, Trang M Nguyen, Sinh T Vu, D3NER: biomedical named entity recognition using CRF-biLSTM improved with fine-tuned embeddings of various linguistic information, *Bioinformatics*, Volume 34, Issue 20, 15 October 2018, Pages 3539– 3546, https://doi.org/10.1093/bioinformatics/bty356
3. G. Yang and H. Xu, "Named Entity Recognition via Interlayer Attention Residual LSTM," 2022 International Joint Conference on Neural Networks (IJCNN), 2022, pp. 01-08, doi: 10.1109/IJCNN55064.2022.9892493.
4. Abdo Ababor Abafogi, "Boosting Afaan Oromo Named Entity Recognition with Multiple Methods", 8th October 2021 International Journal of Information Engineering and Electronic Business,https://www.semanticscholar.org/paper/Boosting-Afaan-Oromo-Named-Entity-Recognition-with-Abafogi/69d40fed3cb71bf377a457b32c99b7f78d0ad338.
5. Tome Eftimov,Barbara Koroušić Seljak and Peter Korošec, "A rule-based named-entity recognition method for knowledge extraction of evidence-based dietary recommendations", 23rd June 2017 Eftimov et al.
6. https://towardsdatascience.com/named-entity-recognition-and-classification-with-scikit-learn-f05372f07ba2
7. https://www.kaggle.com/code/abhinavwalia95/how-to-loading-and-fitting-dataset-to-scikit/data

Git Hub Link : https://github.com/praneethk6795/NER-Using-BiLSTM