Rapport TP4

NLP avec ‘Sequence Models’

Ghandouz Amina G2

1. About the dataset

The CoNLL-2003 dataset is a widely-used benchmark for Named Entity Recognition (NER) tasks, introduced as part of the CoNLL-2003 shared task on language-independent NER. It contains annotated text in English, where each word is labeled with its corresponding named entity tag. The main entity categories are persons (PER), locations (LOC), organizations (ORG), and miscellaneous names (MISC).

Each line contains four columns separated by spaces:

- Token (word or punctuation)
- POS tag (Part-of-Speech tag)
- Chunk tag (syntactic chunk tag)
- Named entity tag (the target label for NER)

and sentences are separated by blank lines where the "-DOCSTART-" line indicates the beginning of a new document in the dataset.

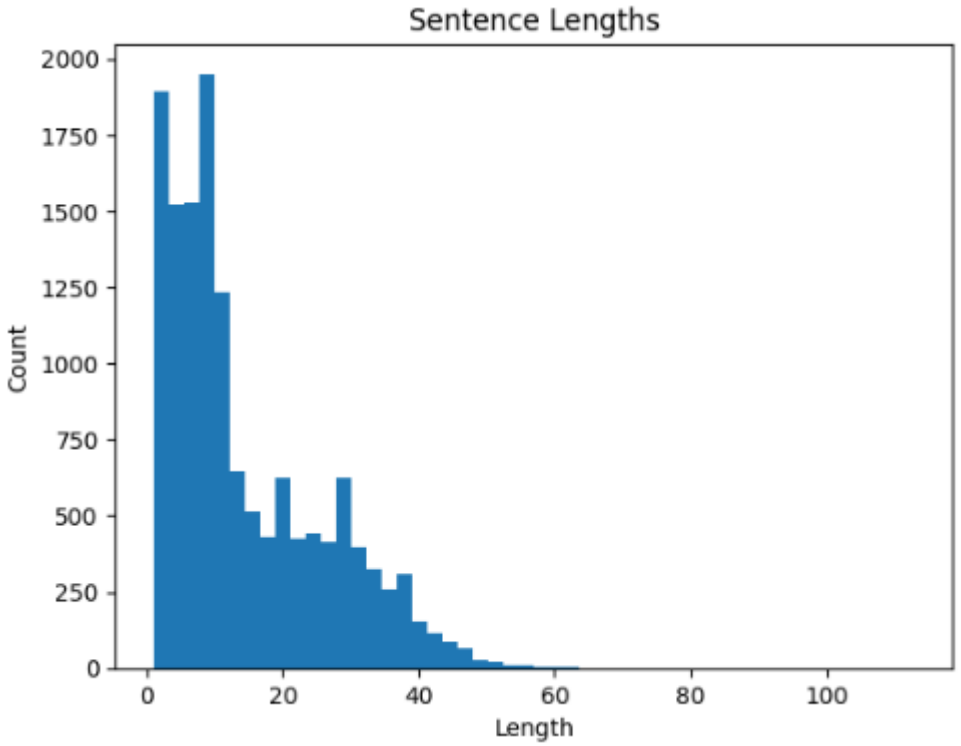
	train_sentences	train_labels
0	[eu, rejects, german, call, to, boycott, briti...	[B-ORG, O, B-MISC, O, O, O, B-MISC, O, O]
1	[peter, blackburn]	[B-PER, I-PER]
2	[brussels, 1996-08-22]	[B-LOC, O]
3	[the, european, commission, said, on, thursday...	[O, B-ORG, I-ORG, O, O, O, O, O, O, B-MISC, O,...
4	[germany, 's, representative, to, the, europea...	[B-LOC, O, O, O, O, B-ORG, I-ORG, O, O, O, B-P...
...	...	...
14036	[on, friday, :]	[O, O, O]
14037	[division, two]	[O, O]
14038	[plymouth, 2, preston, 1]	[B-ORG, O, B-ORG, O]
14039	[division, three]	[O, O]
14040	[swansea, 1, lincoln, 2]	[B-ORG, O, B-ORG, O]

14041 rows × 2 columns

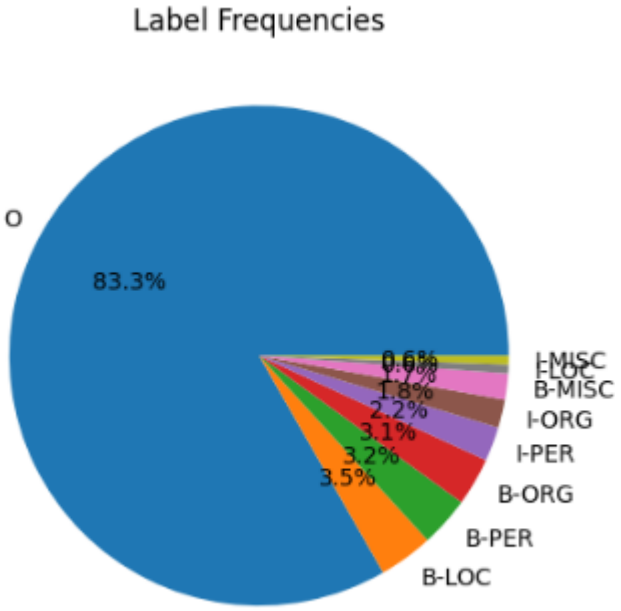
2. Preprocessing step:

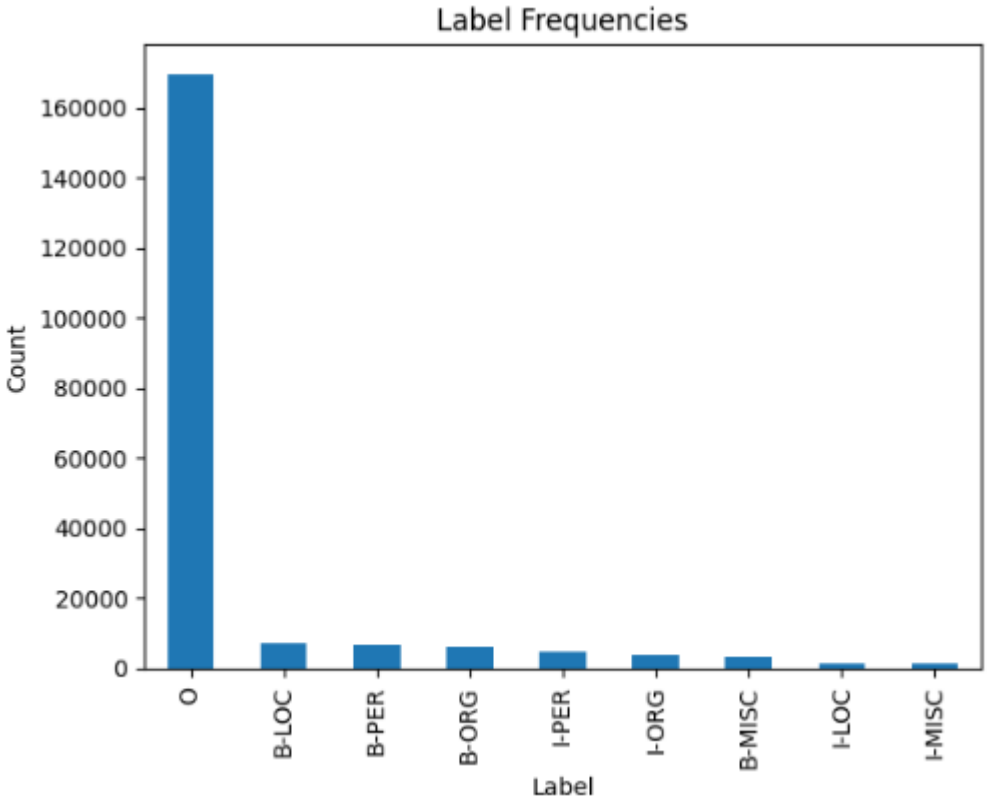
In this step, we read the file line by line. Each sentence is reconstructed by grouping words together until an empty line is encountered (which marks the end of a sentence). Tokenization is performed by separating the words and their corresponding labels. Additionally, case normalization is applied by converting all words to lowercase to ensure consistency in token representation.

3.



Visualization step:


Label Frequencies
O
83.3%
I-MISC
I-LOC
B-MISC
I-ORG
I-PER
B-ORG
B-PER
B-LOC
3.5%
3.2%
3.1%
2.2%
1.8%


Label Frequencies
Count
160000
140000
120000
100000
80000
60000
40000
20000
0
O
B-LOC
B-PER
B-ORG
I-PER
I-ORG
B-MISC
I-LOC
I-MISC
Label

4. Vectorisation step:

In this step, we performed the vectorization of the textual data. First, we used a tokenizer to convert each word in the sentences into a unique numerical index, then we transformed all sentences into sequences of these indices. Since the sentences have different lengths, we applied padding to ensure that all sequences have the same length (50 tokens). We followed the same process for the labels: we tokenized them into numerical indices and padded them as well. This way, both the input sentences and their corresponding labels are represented as fixed-size numerical vectors, ready to be used for model training.

5. Model training step:

In this step, we trained a Named Entity Recognition (NER) model using the CoNLL-2003 dataset.
We first defined the vocabulary size to get the length of word tokenizer + 1, the dimension of the embedding vectors to 100, the number of hidden units in the LSTM layer to 64, the number of classes corresponding to the different named entity tags (9), the batch size to 32 and the number of epochs to 10.

We built a sequential deep learning model composed of three main layers:

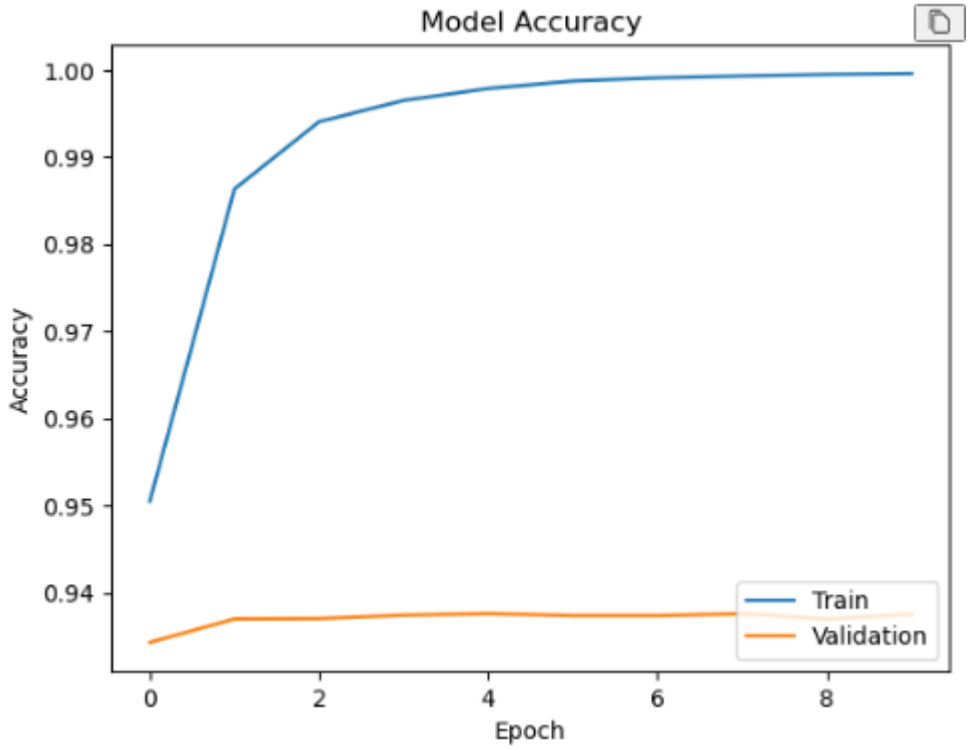
- **Embedding Layer**: this layer transforms each word index into a dense 100-dimensional vector representation, allowing the model to capture semantic information about words.

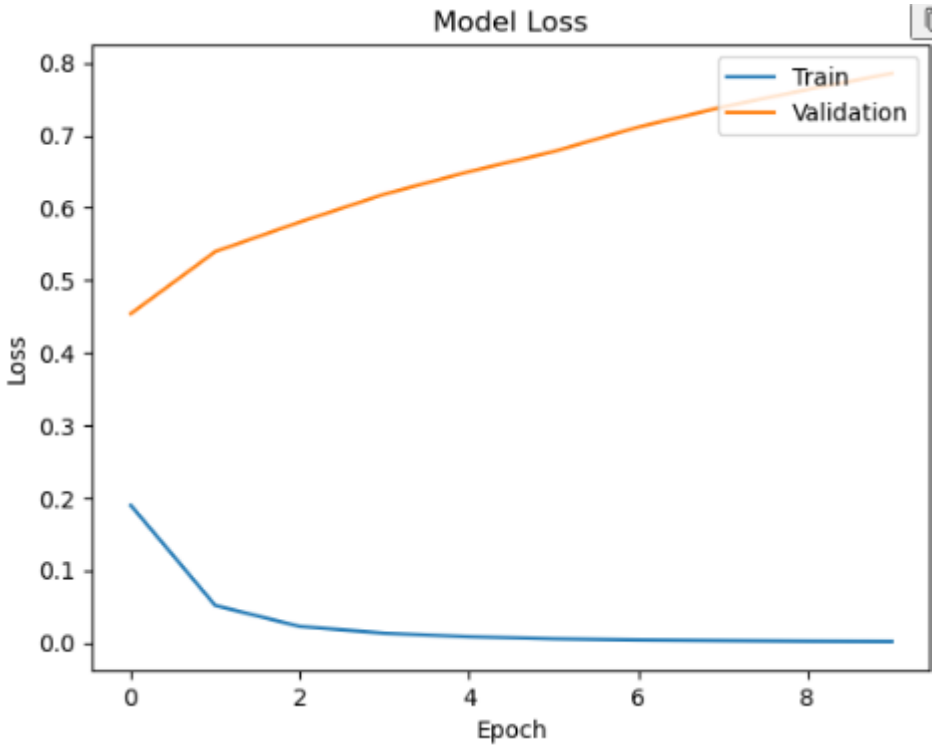
- **Bidirectional LSTM Layer**: we used a Bidirectional LSTM to process the sequences from both directions (forward and backward). This helps the model understand context better, which is crucial for recognizing named entities that depend on both previous and following words.

- **Dense Layer**: we added a dense layer with a softmax activation function to predict the entity class for each token in the sequence.

When compiling the model, we used the **adam optimizer**, we used only two sets, the training set and the test set which was used to validate the model.

During training, we observed that the accuracy improved significantly over the epochs, reaching a very high accuracy on both the training and validation sets.





During training, we noticed that the model achieved a very high accuracy (>99%) on the training set, and the training loss decreased almost to 0. However, on the validation set, the accuracy remained around 94% and the validation loss started to increase, reaching around 0.8, se we have overfitting, it happens when the model learns the training data too well, including small details and noise, but fails to generalize properly to new, unseen data (like the validation set).

- The very low training loss and high training accuracy show that the model is performing perfectly on the training data.
- The higher validation loss and the fact that it keeps increasing (while validation accuracy stays stable) indicate that the model struggles a bit more with generalizing.

6. Evaluation step:

- **Accuracy:** 0.28
- **Precision:** 0.32
- **Recall:** 0.28
- **F1-score:** 0.29

high precision (0.32) compared to recall (0.28) means our model makes positive predictions, we have a low accuracy (0.28) means the model is not doing well, maybe because of the imbalanced dataset,

overfitting problem or because of the tokenization method that we used.

7. Amelioration step:

We used tf-idf to help the model to understand the meanings of words based on their contexts, and we used the GradientBoostingClassifier model which is an ensemble of decision trees, and we got high accuracy (83%).