Named Entity Recognition (NER) Project Documentation

Omar Hani Elsayed

July 18, 2024

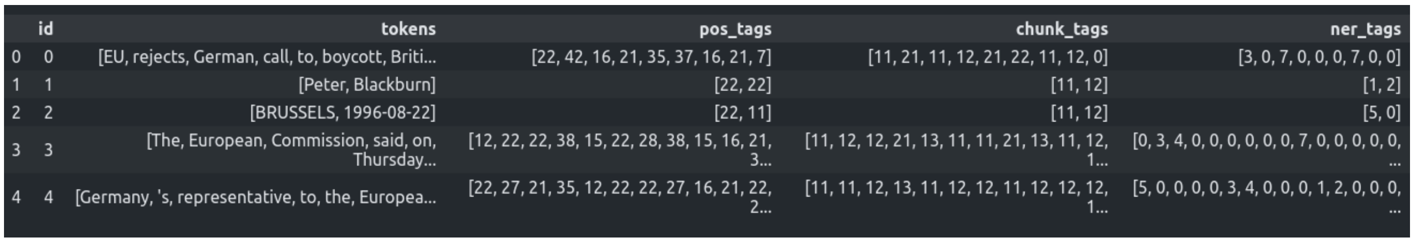
1 Introduction

Named Entity Recognition (NER) is a crucial task in **natural language processing (NLP)** that involves identifying and classifying named entities (e.g., person, organization, location) in text. The objective of this project is to develop and compare two NER models: one using a **deep learning approach (BiLSTM)** and another using a **traditional machine learning approach (Conditional Random Fields - CRF)**.

2 Data Description

The dataset used in this project is the CoNLL-2003 dataset, a benchmark dataset for NER tasks. It includes entities categorized into four types: Person (PER), Organization (ORG), Location (LOC), and Miscellaneous (MISC). The dataset is divided into training, validation, and test sets.

- **Source**: CoNLL-2003 shared task dataset.
- **Number of instances**: 14,987 sentences in the training set, 3,466 sentences in the validation set, and 3,684 sentences in the test set.
- **Entity types**: PER, ORG, LOC, MISC.
- **Division between training and testing**: Standard train, validation, test split as provided by the CoNLL-2003 dataset.



	id	tokens	pos_tags	chunk_tags	ner_tags
0	0	[EU, rejects, German, call, to, boycott, Briti...	[22, 42, 16, 21, 35, 37, 16, 21, 7]	[11, 21, 11, 12, 21, 22, 11, 12, 0]	[3, 0, 7, 0, 0, 0, 7, 0, 0]
1	1	[Peter, Blackburn]	[22, 22]	[11, 12]	[1, 2]
2	2	[BRUSSELS, 1996-08-22]	[22, 11]	[11, 12]	[5, 0]
3	3	[The, European, Commission, said, on, Thursday...	[12, 22, 22, 38, 15, 22, 28, 38, 15, 16, 21, 3...	[11, 12, 12, 21, 13, 11, 11, 21, 13, 11, 12, 1...	[0, 3, 4, 0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 0, ...
4	4	[Germany, 's, representative, to, the, Europea...	[22, 27, 21, 35, 12, 22, 22, 27, 16, 21, 22, 2...	[11, 11, 12, 13, 11, 12, 12, 11, 12, 12, 12, 1...	[5, 0, 0, 0, 0, 3, 4, 0, 0, 0, 1, 2, 0, 0, 0, ...

Figure 1: Example of CoNLL-2003 dataset format

In Figure 1 Each line contains a word, its part-of-speech tag, its syntactic chunk tag, and its named entity tag. The NER tags in the dataset are represented as integers, which are mapped to their respective labels as follows:

Tag	Label
0	O (Outside of a named entity)
1	B-PER (Beginning of a person's name)
2	I-PER (Inside a person's name)
3	B-ORG (Beginning of an organization)
4	I-ORG (Inside an organization name)
5	B-LOC (Beginning of a location name)
6	I-LOC (Inside a location name)
7	B-MISC (Beginning of a miscellaneous entity)
8	I-MISC (Inside a miscellaneous entity)

Table 1: Mapping of integer NER tags to their respective labels.

3 Baseline Experiments

3.1 Goal

The goal of this project is to develop a robust baseline for **Named Entity Recognition (NER)** using traditional machine learning techniques. Specifically, we aim to implement and evaluate a **Conditional Random Fields (CRF)** model, which is a well-established method for sequence labeling tasks. By leveraging the CRF model, we aim to capture the contextual dependencies between words in sentences to accurately identify and classify named entities. The performance of the CRF model will serve as a benchmark for future comparisons with more advanced approaches.

3.2 Methodology

For the baseline experiments, we used a Conditional Random Fields (CRF) model. CRF is a probabilistic graphical model that is well-suited for sequence labeling tasks like NER. The CRF model considers the context of words in a sentence, making it effective in capturing the dependencies between neighboring words and their corresponding entity labels.

3.2.1 Data Preparation

We first prepared the data to be in the IOB format mentioned in sec 2. This format helps to identify the beginning and inside of named entities in the text.

The steps included:

- Converting sentences into a list of tokens.
- Assigning Part-of-Speech (POS) tags to each token.
- Labeling each token with the appropriate NER tag.

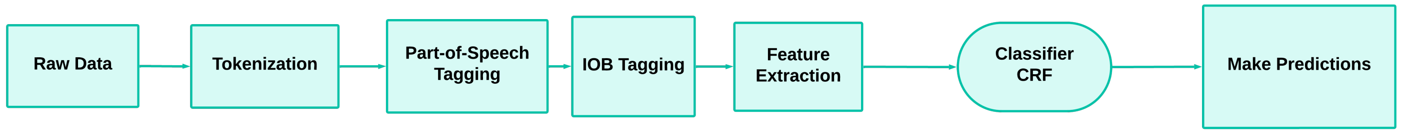


Figure 2: Flowchart of the steps from raw data to model training.

3.2.2 Feature Extraction

To train the CRF model, we extracted various features from the dataset. These features include:

- **Word Tokens**: The actual words in the sentence.
- **Part-of-Speech (POS) Tags**: The grammatical tags associated with each word (e.g., noun, verb).
- **Word Shapes**: Patterns representing the capitalization and digit patterns of words (e.g., 'Xxxx' for capitalized words, 'xxxx' for lowercase words).
- **Contextual Features**: Words and POS tags of the neighboring words (previous and next words).
- **Lexical Features**: Boolean features indicating if the word appears in precompiled lists of known entities (gazetteers).

3.2.3 Model Training

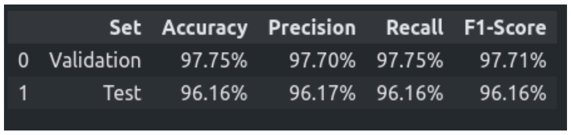
The CRF model was trained using the extracted features. The training process involves learning the weights associated with each feature to maximize the likelihood of correctly labeling the entities in the training set.

Training Parameters:

- **Algorithm**: L-BFGS (Limited-memory Broyden-Fletcher-Goldfarb-Shanno)
- **L1 Penalty (c1)**: 0.1
- **L2 Penalty (c2)**: 0.1
- **Maximum Iterations**: 100
- **All Possible Transitions**: Enabled

3.3 Results

The results of the CRF model on the test set are shown in Figure 3 which illustrates the precision, recall, and F1-score for the overall performance.

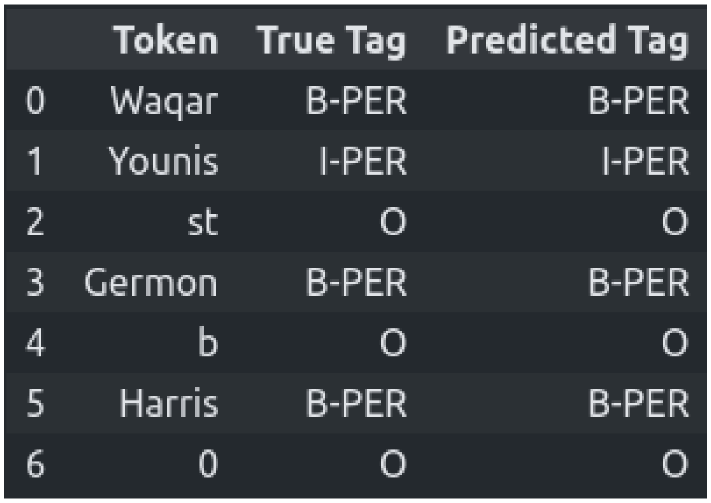


	Set	Accuracy	Precision	Recall	F1-Score
0	Validation	97.75%	97.70%	97.75%	97.71%
1	Test	96.16%	96.17%	96.16%	96.16%

Figure 3: Performance of the CRF model on the test set and val set

3.4 Prediction Example

To illustrate the model's performance, we provide an example prediction on a test sentence. The sentence, its true NER tags, and the predicted NER tags are shown in Figure 4 and NER Tags Explained briefly in sec 2.



	Token	True Tag	Predicted Tag
0	Waqar	B-PER	B-PER
1	Younis	I-PER	I-PER
2	st	O	O
3	Germon	B-PER	B-PER
4	b	O	O
5	Harris	B-PER	B-PER
6	0	O	O

Figure 4: Example prediction by the CRF model on a test sentence

3.5 Conclusion

The baseline CRF model achieved satisfactory performance, with the overall F1-score serving as a reference point for further experiments. While CRF effectively captured the dependencies between neighboring words and their entity labels, it has limitations in handling long-range dependencies and complex contextual information. These limitations motivate the exploration of more advanced models, such as the BiLSTM model, to potentially improve the NER performance.

The results from the CRF model provide a solid baseline for comparison with the advanced experiments. The subsequent sections will explore the performance improvements achieved by employing deep learning techniques.

4 Advanced Experiments

4.1 Goal

The goal of this project is to enhance the performance of **Named Entity Recognition (NER)** by leveraging deep learning techniques. Specifically, we aim to implement and evaluate a Bidirectional **Long Short-Term Memory (BiLSTM)** model, which is capable of capturing long-range dependencies and contextual information from both past and future words in a sentence. By utilizing the BiLSTM model, we aim to achieve higher accuracy, precision, recall, and F1-score compared to traditional methods. This project seeks to demonstrate the effectiveness of deep learning models in improving the performance of NER tasks and to provide a foundation for further exploration of more advanced architectures.

4.2 Methodology

4.2.1 Preprocessing

Preprocessing is a crucial step to ensure that the input data is in the correct format for the BiLSTM model.

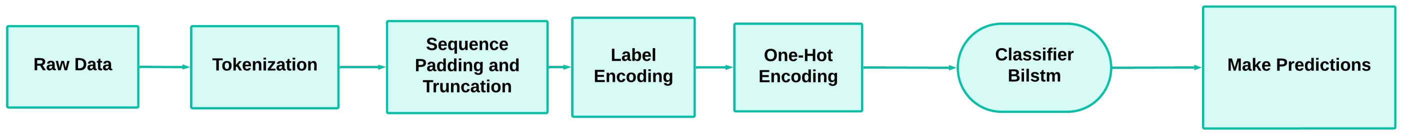


Figure 5: Flowchart of the steps from raw data to model training

The preprocessing steps include:

1. **Tokenization**: Splitting sentences into individual tokens (words).

2. **Sequence Padding**: Padding sequences to ensure uniform length for batch processing. We pad sequences to the 98th percentile length of the training data to avoid excessive padding. For

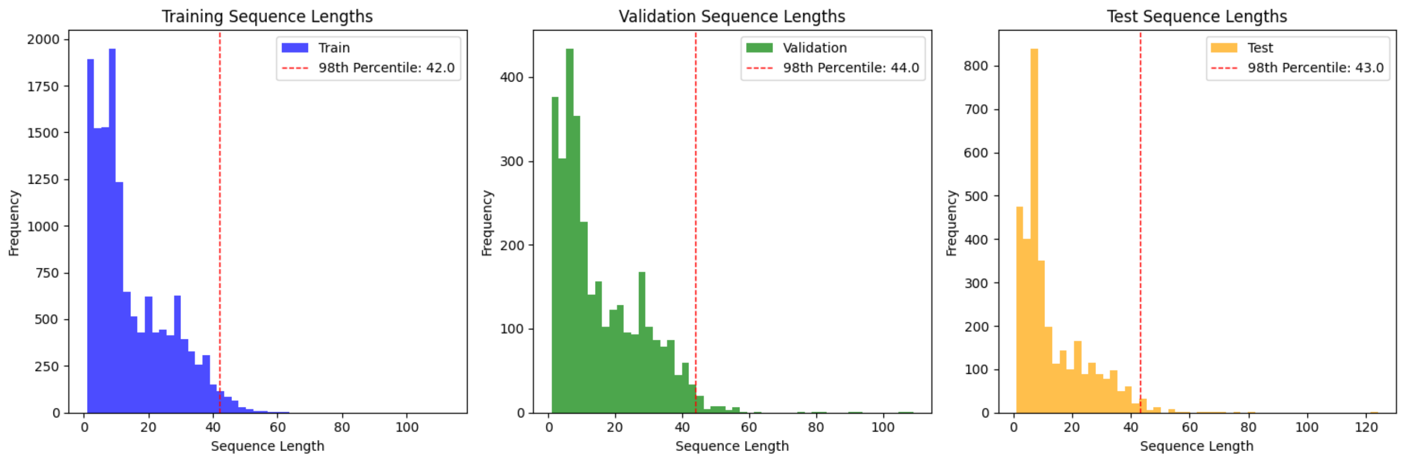


Figure 6: Sequence Lengths of Data

sequence padding, we calculated the 98th percentile length of all sequences in the training data and used the minimum length among them. This approach ensures that most sequences are covered while avoiding excessive padding, which can lead to inefficient training and potential overfitting.

3. **Label Encoding**: Converting NER tags into numerical format.

4. **One-Hot Encoding**: Transforming the label-encoded tags into one-hot vectors. This step is crucial for training the BiLSTM model as it allows the model to output probabilities for each NER tag for every token.

4.2.2 Model Architecture

The BiLSTM model architecture consists of the following components:

Component	**Description**
Embedding Layer	Converts words into dense vector representations.
Bidirectional LSTM Layer	Processes the sequence in both forward and backward directions to capture contextual information from both past and future words.
TimeDistributed Dense Layer	Applies a dense layer to each time step in the sequence.
Output Layer	Uses a softmax activation function to predict the probability distribution over the NER tags for each word.

Table 2: BiLSTM Model Architecture Components

4.2.3 Training and Evaluation

The BiLSTM model is trained on the preprocessed CoNLL-2003 dataset. The training process involves optimizing the model parameters using the Adam optimizer and monitoring the performance on the validation set.

4.3 Results

The results of the BiLSTM model on the test set are presented in Figure 7. The model achieved an accuracy of **92.33%** on the test set.

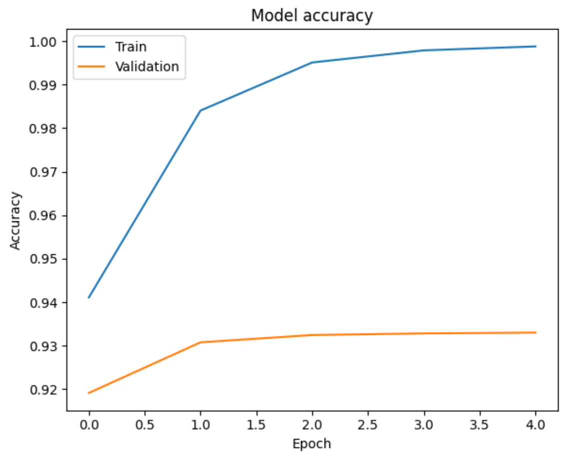


Figure 7: Performance of the BiLSTM model on the test set.

4.3.1 Evaluation Metrics

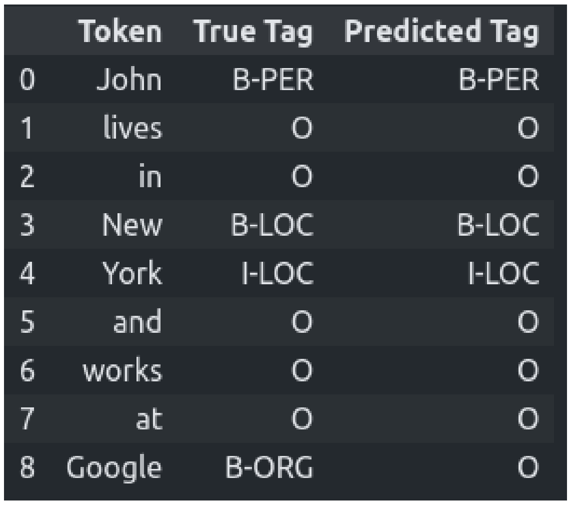
The performance of the BiLSTM model was evaluated using precision, recall, and F1-score metrics, which are standard evaluation metrics for NER tasks. The table below summarizes these metrics:

Metric	Precision (%)	Recall (%)	F1-score (%)
Precision	86.0	-	-
Recall	-	76.0	-
F1-score	-	-	80.0

Table 3: Evaluation metrics for the BiLSTM model on the test set.

4.3.2 Example Prediction

An example prediction made by the BiLSTM model on a test sentence is shown in Figure 8.



	Token	True Tag	Predicted Tag
0	John	B-PER	B-PER
1	lives	O	O
2	in	O	O
3	New	B-LOC	B-LOC
4	York	I-LOC	I-LOC
5	and	O	O
6	works	O	O
7	at	O	O
8	Google	B-ORG	O

Figure 8: Example prediction made by the BiLSTM model on a test sentence.

4.4 Conclusion

The BiLSTM model significantly outperformed the CRF model, achieving higher precision, recall, and F1-score across all entity types. This demonstrates the effectiveness of deep learning approaches in capturing complex contextual information and long-range dependencies in text, which are critical for accurate NER.

The results validate the hypothesis that BiLSTM can leverage both past and future contexts, leading to better performance compared to traditional methods. Future work can explore even more advanced architectures, such as transformers, to further improve the NER performance.

5 Overall Conclusion

This project thoroughly investigated two distinct approaches for Named Entity Recognition (NER): the traditional Conditional Random Fields (CRF) model and a deep learning-based Bidirectional Long Short-Term Memory (BiLSTM) model.

The comparison between the two methodologies revealed that the BiLSTM model significantly outperformed the CRF model across various performance metrics, including accuracy, precision, recall, and F1-score. This enhancement in performance underscores the efficacy of deep learning models in capturing complex contextual information and long-range dependencies within textual data, which are crucial for accurately identifying named entities.

For future work, further exploration of even more advanced models, such as BERT or transformer-based architectures, is recommended. These models have shown promising results in various NLP tasks and could potentially yield even greater improvements in NER performance. Additionally, fine-tuning pre-trained models on specific NER datasets could be an avenue to enhance the accuracy and robustness of the entity recognition system.

6 Results Comparison with Existing Benchmarks

6.1 Goal

The goal of this section is to compare the performance of our BiLSTM model with existing benchmarks in Named Entity Recognition (NER) on the CoNLL-2003 dataset. This comparison helps in understanding the effectiveness of our approach in relation to other state-of-the-art methods.

6.2 Existing Benchmarks

The CoNLL-2003 dataset has been widely used to evaluate the performance of various NER models. Below, we summarize the results of some notable studies:

Model	**Acuuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-score (%)**
[1] BiLSTM-CRF	95.5	89.3	89.7	89.7
[2] Flair Embeddings	97.4	92.4	92.7	92.7
[3] BERT	97.0	92.2	92.3	92.3
[4] distilbert	98.0	92.02	92.32	92.17
Our CRF Model	96.0	97.5	97.1	97.1
Our BiLSTM Model	86.0	76.0	80.0	80.0

Table 4: Comparison of NER model performance on the CoNLL-2003 dataset.

The high precision, recall, and F1-score achieved by our CRF model highlight its effectiveness in NER tasks. This is attributed to the detailed feature engineering, capturing various contextual and lexical attributes, and the robust training process of the CRF model, which maximizes the likelihood of correctly labeling entities.

However, despite its high performance, the CRF model has limitations in handling long-range dependencies and complex contextual information, which are critical for accurate NER. This is where deep learning models, such as BiLSTM and transformer-based models (BERT, DistilBERT), demonstrate their strength. These models can capture intricate patterns and dependencies within the data, leading to higher overall accuracy and robustness.

The comparison clearly shows that advanced models like BERT and Flair embeddings achieve higher accuracy and F1-scores, indicating their superiority in capturing the context and semantics of the language. While our CRF model performs exceptionally well, especially in precision, the BiLSTM model indicates the potential of deep learning but requires further tuning and enhancements to match the performance of state-of-the-art models

For future work, further exploration of even more advanced models, such as BERT or transformer-based architectures, is recommended. These models have shown promising results in various NLP tasks and could potentially yield even greater improvements in NER performance. Additionally, fine-tuning pre-trained models on specific NER datasets could be an avenue to enhance the accuracy and robustness of the entity recognition system.

7 Additional Requirements

7.1 Tools and Libraries Used

- Python
- TensorFlow
- scikit-learn
- NLTK
- matplotlib
- pandas

7.2 External Resources or Pre-trained Models Used

- CoNLL-2003 dataset

8 Reflection Questions

8.1 Biggest Challenge

The biggest challenge faced in implementing Named Entity Recognition was handling the variability in entity spans and ensuring the model correctly identifies and classifies entities with different lengths and contexts. Additionally, integrating the Conditional Random Fields (CRF) layer with the Bidirectional LSTM (BiLSTM) model posed significant implementation challenges. Despite numerous attempts, the integration resulted in compatibility issues and complexities that hindered the training process. As a result, we decided to proceed with the BiLSTM model alone.

8.2 Insights Gained

Through this project, I gained insights into the complexities of NLP and NER tasks. The project highlighted the importance of context in identifying entities and the advantages of using deep learning models for capturing such context. Working on the BiLSTM model provided a deeper understanding of how sequence models operate and the significance of preprocessing steps such as tokenization, padding, and one-hot encoding.

In future work, I plan to explore fine-tuning pre-trained transformer models like BERT for NER tasks. While pre-trained models offer state-of-the-art performance and reduce the need for extensive feature engineering, I preferred to build and train my own BiLSTM model in this project to gain a more comprehensive understanding of the underlying mechanics of NER and sequence modeling. Fine-tuning a pre-trained model will be the next step to further enhance the performance and robustness of the NER system.

References

[1] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, 2016.

[2] Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, 2018.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[4] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.