Named Entity Recognition (NER)

Méthodes empiriques en traitement du langage (METL 2023)

Professor Tanja Samardzic

University of Geneva

Seyedvahid Mousavinezhad

June 10, 2023

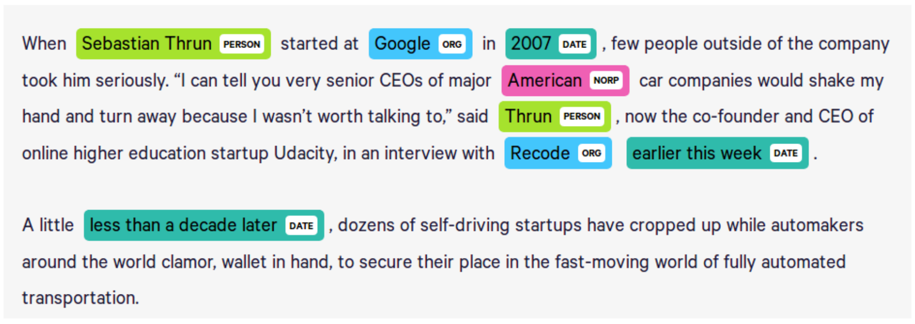
Named Entity Recognition (NER) is a task in Natural Language Processing (NLP) that identifies and classifies important things like names, places, and organizations in text, making searching and understanding text easier.

1 Introduction

An important NLP task, Named Entity Recognition (NER) identifies and classifies names, organizations, locations, and more in text. It plays a vital role in various applications like information retrieval, question answering, translation, sentiment analysis, and summarization. The main objective of NER is to precisely recognize and categorize named entities present in a text. This process entails two essential steps: detecting the boundaries of the entities, which involves identifying the starting and ending positions of the named entities and classifying the entities by assigning a predetermined label or category to each one. For example, in the sentence "Elon Musk has a plan to go to Mars." NER would classify "Elon Musk" as a person and "Mars" as a location.

NER is important because it enables machines to understand and extract meaningful information from unstructured text. By identifying named entities, NER helps in organizing and structuring textual data, which can be further used for various downstream tasks. For example, in information retrieval systems, NER can improve search results by allowing users to search for specific entities. In question-answering systems, NER helps in identifying the entities relevant to the question. In sentiment analysis, recognizing named entities can provide additional context to determine the sentiment expressed towards them.

Nevertheless, the NER task has several challenges. One major difficulty is that named entities can be ambiguous and vary in different contexts. For example, the word "Jaguar" can refer to an animal or a company. To solve these ambiguities, we need a deep understanding of the context and general knowledge. Additionally, named entities can have complex structures like compound names or nested entities, which makes their detection and classification more difficult. Another challenge in NER is the lack of labeled training data. Annotated datasets are often limited, especially for specialized domains or languages. Collecting and labeling large-scale datasets for NER can be time-consuming and expensive. This scarcity of labeled data can affect the performance of NER models, especially for less common or emerging entities.


When Sebastian Thrun PERSON started at Google ORG in 2007 DATE , few people outside of the company took him seriously. “I can tell you very senior CEOs of major American NORP car companies would shake my hand and turn away because I wasn’t worth talking to,” said Thrun PERSON , now the co-founder and CEO of online higher education startup Udacity, in an interview with Recode ORG earlier this week DATE .

A little less than a decade later DATE , dozens of self-driving startups have cropped up while automakers around the world clamor, wallet in hand, to secure their place in the fast-moving world of fully automated transportation.

Figure 1: In this example, the NER system found eight entities that have been classified into four different categories: person, organization, date and nationality.

In this project, I will explore two approaches for the NER problem: the CRF (Conditional Random Fields) as a machine learning model and using the Transformers as a deep learning model. I aim to enhance the performance of these models by addressing the challenges they encounter and finding effective solutions.

2 State of the art

To train effective NER models, a high-quality dataset with annotated examples of named entities is essential. The dataset creation process consists of data collection from diverse sources such as news articles, books, and web pages to ensure broad coverage. Clear annotation guidelines are defined, specifying entity types, boundaries, and conventions. Annotators mark the named entities manually or with automated tools, followed by a quality assurance process to ensure accuracy and consistency.

After annotation, the dataset is typically divided into training, validation, and test sets. The training set is used to train the NER model, while the validation set aids in fine-tuning hyperparameters and evaluating model performance. The test set is then employed to assess the model’s generalization ability. Several notable NER datasets are widely used in research and development, for instance, CoNLL-2003[1], OntoNotes[2], and WikiNER[3]. These datasets serve as invaluable resources for training and evaluating NER models, advancing the field of NLP. Additionally, efforts are being made to create and share larger annotated datasets to train NER models. Crowdsourcing and active learning techniques can help in the efficient collection and labeling of data.

Machine learning and deep learning methods have been extensively explored for NER, each with its own strengths and characteristics. In the Machine Learning Approach, Conditional Random Fields (CRFs) are widely used in NER as probabilistic machine learning models. In this approach, relevant features such as word embeddings, part-of-speech tags, and contextual information are extracted from the input text. These features are then fed into a CRF model, which learns the dependencies between words and their corresponding entity labels. During training, the model is optimized to maximize the likelihood of the observed entity labels given the input features. Once trained, the model can make predictions by decoding the most likely label sequence based on the learned dependencies. The strength of this approach lies in its ability to capture dependencies between neighboring words, enhancing the accuracy of entity recognition.

According to this approach, significant contributions have been made by Lafferty, McCallum, and Pereira in their seminal work[4]. The authors address the limitations of Hidden Markov Models (HMMs) by introducing Conditional Random Fields (CRFs) as a powerful probabilistic modeling framework. They emphasize the ability of CRFs to capture rich dependencies between labels and input features, allowing for the modeling of long-range interactions. The work presents the mathematical formulation of CRFs as exponential models and discusses feature function design, parameter estimation, and regularization techniques. The effectiveness of CRFs is demonstrated through their application in various sequence labeling tasks, including named entity recognition.

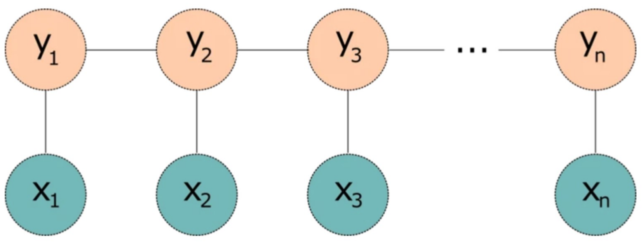


Figure 2: A simple linear-chain conditional random fields model. The model takes an input sequence x (words) and target sequence y (IOB tags).

In the Deep Learning Approach, Recurrent Neural Networks (RNNs) and Transformers have shown great promise in NER, particularly with sequential data. In this approach, recurrent neural networks (RNNs), often utilizing Long Short-Term Memory (LSTM) cells, are employed. The input text is tokenized, and each token is represented as a word embedding. These token embeddings are then processed by the LSTM-based RNN, which captures contextual dependencies. The output of the LSTM is fed into a softmax layer to predict the entity labels for each token. Another powerful deep learning architecture, the transformer model, has also gained popularity in NER. Transformers leverage attention mechanisms to capture global dependencies in the input sequence, eliminating the need for recurrent connections. They have achieved remarkable results in various NLP tasks, including NER.

This approach demonstrated in the Keras documentation example[5], involves tokenizing the input text, encoding labels, and training a model using TensorFlow's Keras API. The trained model can then predict entity labels for new text by assigning the most probable label to each token. By leveraging transformers, known for capturing contextual information and long-range dependencies, this deep learning approach showcases the effectiveness of NER in accurately identifying named entities.

In this project, my objective is to enhance the precision, recall, and F1 score of Named Entity Recognition (NER) on the CoNLL-2003 dataset. I will investigate the effects of modifying the dataset and tuning hyperparameters of the CRF model and also, I will explore the potential of different deep neural network architectures. As a result of studying and improving these approaches, I aim to advance NER technology and its uses in natural language processing.

3 My approach

The primary objective of this research is to improve the performance of Named Entity Recognition (NER) through various approaches. The focus lies on enhancing the evaluation metrics on the testing data. To achieve this, several strategies will be employed.

One aspect of the research involves fine-tuning the hyperparameters of the CRF model to tackle the challenge of overfitting. The number of iterations will be adjusted, and the values of L1 and L2 regularization (c1 and c2) will be fine-tuned through a rigorous process. By exploring a range of hyperparameter values using random search and evaluating their performance using k-fold cross-validation, optimal settings will be identified to enhance the model's generalization capabilities.

Considering the imbalanced nature of the dataset, special attention will be given to tackling this issue. Class weighting techniques will be implemented to assign higher importance to the minority classes, enabling the model to better capture their patterns and improve their recognition. Moreover, undersampling of majority class instances will be employed to create a more balanced representation of the dataset, mitigating the bias towards the majority class and allowing the model to better learn from the minority class samples.

In addition to the CRF model, the research will explore the utilization of deep neural network architectures, with a particular focus on the powerful BERT model. By leveraging the pretraining and transfer learning capabilities of BERT, the model will be fine-tuned specifically for NER. This approach aims to harness the contextual information captured by BERT to improve the identification and classification of named entities.

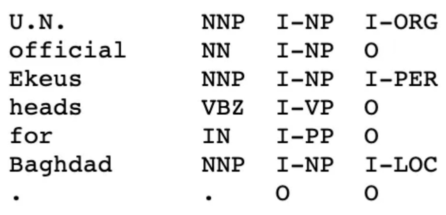
Finally, The outcomes of the introduced changes and modifications will be meticulously analyzed and reported. A comprehensive evaluation will be conducted, comparing the performance metrics of the proposed solutions against the baseline models. This analysis will provide valuable insights into the effectiveness of the implemented approaches and their impact on enhancing NER performance. Through this research, a deeper understanding of the impact of hyperparameter tuning, data balancing techniques, and the incorporation of deep neural network architectures will be gained.

4 Data and methods

4.1 CoNLL-2003

The dataset used in this project is the CoNLL-2003 dataset, a widely recognized benchmark dataset for Named Entity Recognition (NER) tasks. It was created for the Conference on Natural Language Learning (CoNLL) in 2003 and has been widely used in research and development. The data in the CoNLL-2003 dataset was collected and annotated from English and German news articles. The annotation process involves identifying and categorizing named entities in the text, such as person names, organization names, locations, and miscellaneous entities. The dataset follows the BIO (Beginning, Inside, Outside) format for tagging the named entities.

For this project, only the English language subset of the dataset is used, focusing solely on NER tasks in English. The annotations are provided at the token level, where each word or part of a word is assigned a specific named entity tag. The dataset consists of three main parts: the training set, the validation set, and the test set.



```
U.N.         NNP  I-NP  I-ORG
official     NN   I-NP  O
Ekeus        NNP  I-NP  I-PER
heads        VBZ  I-VP  O
for          IN   I-PP  O
Baghdad      NNP  I-NP  I-LOC
.            .    O     O
```

Figure 3: An example of CoNLL-2003 dataset, The first item on each line is a word, the second a part-of-speech (POS) tag, the third a syntactic chunk tag and the fourth the named entity tag..

To work with the CoNLL-2003 dataset in Python, I use the ”datasets” library. The datasets library is a powerful Python library for accessing and manipulating various datasets commonly used in natural language processing and machine learning. The dataset is typically divided into three parts: the training set, which is used to train NER model, the validation set, which is used to tune the hyperparameters of model, and the test set, which is used to evaluate the performance of trained model on unseen data.

CoNNL-2003 has class imbalances, meaning that certain entity types may appear much less frequently than others. For example, the number of person names mentioned in a text might be significantly lower than the number of common words. In such imbalanced scenarios, accuracy alone is not an adequate measure of a model's performance. This is because a model that simply predicts the majority class for every token will achieve a high accuracy but fail to detect the minority entities effectively. Therefore, accuracy can be misleading in NER tasks with imbalanced classes. The F1 score, on the other hand, combines precision and recall into a single value, providing a harmonic mean of the two. By taking both precision and recall into account, the F1 score provides a balanced evaluation of the model's ability to correctly identify entities while minimizing false positives and false negatives.

4.2 CRF

Conditional Random Fields (CRF) is a probabilistic modeling technique used in named entity recognition (NER) tasks. CRF addresses the challenge of capturing dependencies between neighboring tokens in a sequence, which is crucial for accurate entity recognition. By considering contextual information, CRF improves predictions by estimating the conditional probability of label sequences given input tokens. In my project, I utilized the scikit-learn library's implementation of the CRF model from the sklearn_crfsuite module. To improve the evaluation metrics compared to the baseline, I experimented the following adjustments:

- **I) Make the train set larger:** increasing the size of the training set in NER helps the model gain a broader understanding of named entities, improve generalization capabilities and reduce chance of overfitting. In order to do that, I decided to combine the train and validation sets into a single cohesive dataset. By merging these sets, I aimed to provide my model with a larger and more diverse collection of labeled examples.

- **II) Adjust the Hyperparameters:** The two key hyperparameters for CRF are c1 and c2, which control the L1 and L2 regularization terms respectively, and the max_iterations parameter, which specifies the maximum number of iterations for training.

  To find the optimal values for c1 and c2, I use random search combined with k-fold cross-validation. This approach helps explore a range of hyperparameter combinations and assess their impact on model performance(F1_Score) in a robust and systematic manner.

  Random search involves randomly sampling hyperparameter values within specified ranges. I define a range for c1 and c2 using probability exponential distributions. This distributions can control the likelihood of selecting different values within the range.

- **III) Balancing methods:** We're dealing with an imbalanced dataset, where the majority of the labels are "O" in the case of the CoNLL 2002 dataset, and I employ two strategies to address this issue.

  - **Class Weighting:** Adjust the class weights during training to give more importance to the minority class. I use the class_weight module in sklearn.utils library to find weights of the classes. afterward, to balance the classes, I duplicate each token according to the weight ratio of each class.

  - **Undersampling:** I Randomly remove instances of the majority class ("O" labels) to balance the dataset.

4.3 Transformers

Using self-attention mechanisms to capture information from both sides of a context, Transformer models have significantly improved Named Entity Recognition. The baseline that I consider, demonstrates how to train a transformer-based model for NER tasks. It walks through the process of preparing the data, tokenizing the input text, and creating the necessary input sequences and label encodings. It then builds and trains a deep neural network model using TensorFlow's Keras API. During training, the model learns to predict the entity labels for each token in the input text.

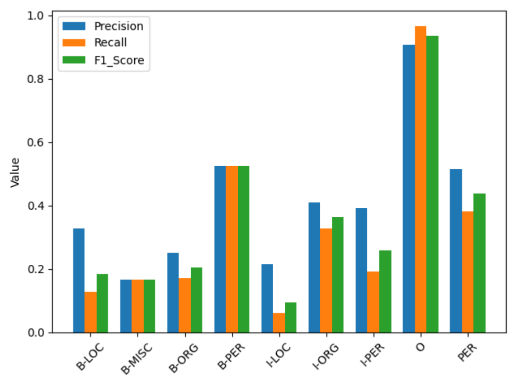
To make improvement, I employ a pre-trained BERT model for the fine-tuning process NER task. To start, I load the pre-trained BERT model using the Hugging Face's Transformers library. For the fine-tuning process, I prepare my NER dataset by organizing it into token-level annotations and tokenize the input text using the BERT tokenizer. I then proceed to train the modified BERT model on my NER dataset, allowing it to learn the predicted entity labels for each token. Throughout the training process, I adjust hyperparameters, including the learning rate, batch size, and regularization techniques, to optimize the performance of the fine-tuned model. After training, I evaluate the model's performance on a separate validation dataset, measuring precision, recall, and F1-score. This evaluation provides insights into the accuracy and effectiveness of the fine-tuned model in identifying named entities.

5 Evaluation

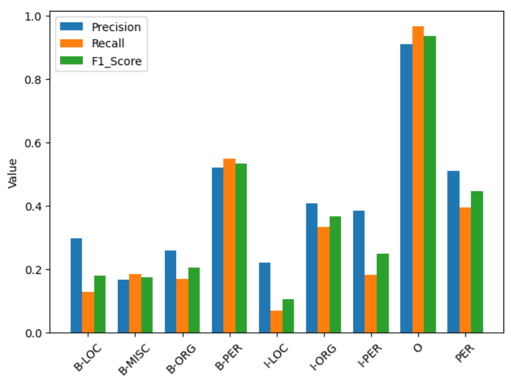
5.1 CRF

I have implemented all the methods mentioned in the previous section, and I will explain each of them in detail.

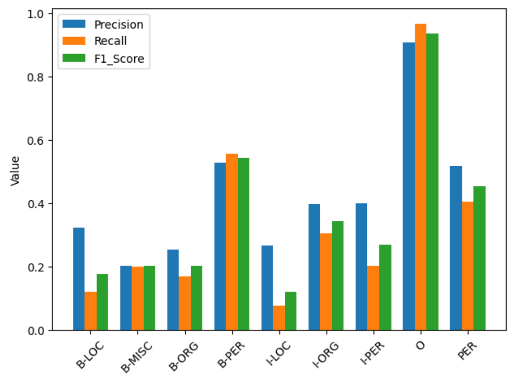
- **I) Make the train set larger:** By combining the training set and validation set to create a larger training set, the model's performance improved, resulting in a 0.1% increase in the F1 score.



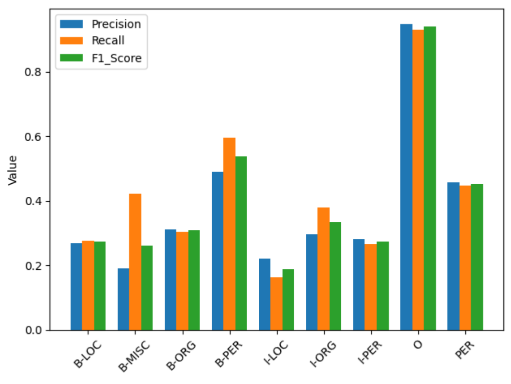
(a) Baseline



(b) Increasing trainset size



(c) c1 & c2 adjustment



(d) Undersampling

Figure 4: Diagrams of Precision, Recall and F1_Score for different methods used in this project.

- **II) Adjust the Hyperparameters:** I attempted to find the optimal values for c1 and c2 through multiple runs using random search and k-fold cross-validation. However, despite the various attempts, the values obtained for L1 and L2 regularization did not lead to any noticeable changes in the F1 score.

- **III) Balancing methods:**
  - **Class Weighting:** The class weighting method proved ineffective. I assigned weights to words in each sequence of the training set based on the weight ratio of each class. However, during the evaluation process, the model failed to predict properly. This was primarily due to the duplication of entities, such as a person entity being replicated four times (based on the weight scale), which posed significant challenges for the model's learning process.
  - **Undersampling:** By randomly removing some samples from the majority class 'O,' I observed a remarkable improvement in the F1 score, as illustrated in Table 1. This method has proven to be highly effective and successful.

Table 1: Comparison of the evaluation metrics of different methods

Method	Accuracy	Precision	Recall	F1_Score
Baseline	84.8%	81.9%	84.8%	83.0%
Increasing trainset size	84.8%	82.0%	84.8%	83.1%
c1 & c2 adjustment	84.9%	82.1%	84.9%	83.1%
Undersampling	83.4%	84.3%	83.4%	83.8%

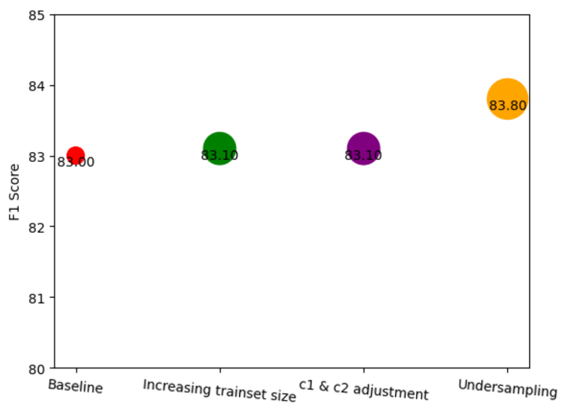


Figure 5: This diagram illustrates the F1 scores obtained for each method implemented in this project. It is evident from the diagram that the undersampling technique, specifically removing instances belonging to the majority class, yielded the best solution.

5.2 Transformers

As shown in Table 2, the NER model that was fine-tuned on the pre-trained BERT model achieved a significantly higher F1 score (almost 95%) on the CoNLL-2003 dataset. This improvement can be attributed to the model's inherent knowledge of words acquired during the pretraining process, as well as the utilization of subword tokenization. In Figure 6, we can observe a significant improvement in the F1 score of the model from the very first epoch of fine-tuning, surpassing the F1 score obtained by the baseline model. This indicates that the fine-tuning process has already started to enhance the model's performance right from the initial epoch.

Table 2: Comparison of the evaluation metrics of models

Model	Accuracy	Precision	Recall	F1_Score
Baseline	93.56%	72.38%	67.92%	70.08%
Fine-Tuned	98.69%	94.48%	95.17%	94.82%

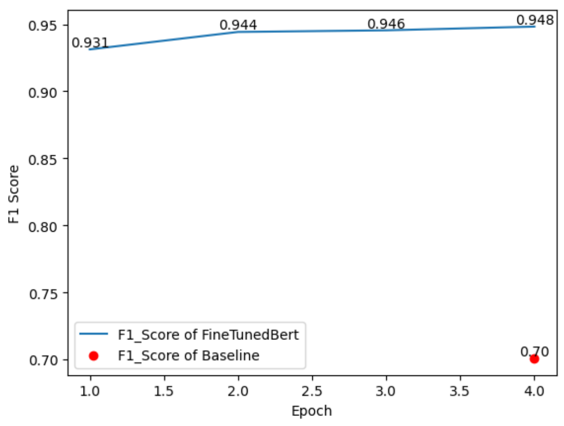


Figure 6: This diagram represents the F1 scores of each epoch during the fine-tuning process of the pretrained model method. The red dot on the diagram represents the F1 score of the baseline model.

6 Interpretation

In the case of CRF, the implemented solution has proven to be effective and successful. Various methods were attempted, including combining the training set and validation set, class weighting, adjusting the Hyperparameters, and undersampling. Among these approaches, undersampling the majority class resulted in the most notable improvement in the F1 score, indicating its effectiveness in addressing the problem.

Regarding the Transformers, fine-tuning a pretrained model, specifically utilizing BERT, demonstrated remarkable performance, achieving a significantly higher F1 score compared to the baseline model. This success can be attributed to the model's inherent knowledge of words acquired during pretraining and the utilization of subword tokenization, which improved the model's ability to capture nuanced information.

The problem has been effectively resolved through the implemented solutions. As a result of using these methods for the CRF approach and the fine-tuning of a pretrained Transformer model, such as BERT, have showcased their efficacy in enhancing the model's performance and achieving superior F1 scores. These outcomes validate the selected approaches and their effectiveness in addressing the problem at hand.

7 Discussion

7.1 CRF

One unexpected observation was that the class weighting method, which duplicates words based on the weight ratio of each class, did not improve the model's performance as expected. The duplication of entities caused by this method led to challenges in the model's learning process and resulted in improper predictions.

One limitation of the CRF approach was that adjusting the hyperparameters, specifically c1 and c2, did not lead to noticeable improvements in the F1 score. This suggests that the impact of L1 and L2 regularization on the model's performance may be limited in this context.

In the future, it may be beneficial to explore other techniques for addressing class imbalances, as the class weighting method did not yield satisfactory results. Moreover, it may be worthwhile to investigate other regularization techniques or optimization algorithms that could potentially improve the CRF approach. Additionally, ensemble methods, where multiple models are combined(LSTM-CRF), could be explored to further boost performance.

7.2 Transformers

For the Transformers, although fine-tuning a pretrained BERT model proved to be highly effective and achieved a significantly higher F1 score, it is essential to consider the computational and time requirements associated with this process. Fine-tuning a large-scale model like BERT involves adjusting numerous parameters and updating the model's weights based on the specific task at hand. This process requires substantial computational resources, including powerful GPUs or TPUs, and can be time-consuming, especially when dealing with extensive datasets.

Moreover, fine-tuning is an iterative process, and the optimal hyperparameters for a given task may require careful tuning and experimentation. This can add an additional layer of complexity and time requirements during the fine-tuning process.

In the future, experimenting with different pretraining models, such as GPT or RoBERTa, could provide insights into their effectiveness for the specific task of Named Entity Recognition (NER). Additionally, exploring different strategies for tokenization could further enhance the model's performance.

8 Conclusion

In conclusion, this project made significant advancements in improving the performance of Named Entity Recognition (NER) models. NER is a critical task in Natural Language Processing (NLP) that involves identifying and classifying named entities in text. The project specifically focused on enhancing the precision, recall, and F1 score of NER models on the CoNLL-2003 dataset, which is widely used for benchmarking NER algorithms.

To achieve this goal, the project employed a comprehensive research approach. Firstly, the hyperparameters of the Conditional Random Fields (CRF) model were fine-tuned to optimize its performance. Additionally, the project tackled the challenge of imbalanced data by implementing class weighting and undersampling techniques, ensuring that the models are trained on a well-balanced dataset. Furthermore, the project explored the potential of deep neural network architectures, specifically leveraging Transformers and finetuning pretrained models, to capture more complex patterns and contextual information in the text.

The evaluation phase of the project involved comparing the performance metrics of the modified models against baseline models. By conducting a thorough analysis, the project provides valuable insights into the effectiveness of the implemented approaches and their impact on enhancing NER performance. Overall, this project sheds light on the significance of NER, the challenges it poses, and the diverse range of approaches, including both traditional machine learning methods like CRF and advanced deep learning models like Transformers, that can be employed to tackle these challenges and improve NER performance.

Table 3: Project Timeline

Task	**Date**
Implement Baseline of Neural model	18.05.2023
Implement Baseline of CRF model	23.05.2023
Adjusting c1 & c2 values	30.05.2023
Create Git repository for project	01.06.2023
Implement class weight method	02.06.2023
Start Writing the Report	03.06.2023
Implement Undersampling method	03.06.2023
Implement Finetuning method	05.06.2023
Complete the Report	10.06.2023

Git Repository Link:

`https://gitlab.unige.ch/Seyedvahid.Mousavinezhad/metl_2023`

References

[1] T. K. Sang, F. Erik, and D. M. Fien, "Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition," *In Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL, pages 142–147.*, 2003.

[2] R. Weischedel, M. Palmer, M. Marcus, E. Hovy, and S. Pradhan, "Ontonotes release 5.0 ldc2013t19," *Web Download. Philadelphia: Linguistic Data Consortium*, 2013.

[3] A. Arun, C. Dyer, and P. Koehn, "Monte carlo inference and maximization for phrase-based translation.," *In Proceedings of the Thirteenth Conference on Computational Natural Language Learning, pages 102–110*, 2009.

[4] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data.," *In Proceedings of the 18th International Conference on Machine Learning (ICML), pp. 282-289)*, 2001.

[5] V. Singh, "Named entity recognition using transformers and data from conll 2003 shared task.," *Keras Documentation*, 2021.

Feedback