F3

Faculty of Electrical Engineering
Department of Cybernetics

Semester Project

Named Entity Recognition

Stanislav Lamoš
Open Informatics

December 2022
Supervisor: Ing. Jan Pichl

Acknowledgement / Declaration

Thank you for all the help during the writing process of this report.

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 27. 12. 2022

.....

Abstrakt / Abstract

Tato práce se zabývá detekcí entit pomocí různých algoritmů.

Klíčová slova: Rozpoznávání entit, CRF, HMM

This report focuses on named entity recognition using various approaches and algorithms.

Keywords: NER, CRF, HMM

/ Contents

Tables / Figures

Chapter 1
Introduction

1.1 About

Machine learning is a subfield of artificial intelligence that involves training algorithms to learn patterns in data and make predictions or decisions based on that learning. In machine learning, an algorithm is trained on a dataset, which consists of input data and corresponding labels or outputs. The algorithm uses the input data to learn patterns and relationships in the data, and the labels or outputs provide guidance on how well the algorithm is learning.

Once the algorithm has been trained, it can be used to make predictions or decisions on new, unseen data. For example, a machine learning algorithm might be trained to recognize letters in images of handwritten text, and then be used to classify letters in new images.

There are many different types of machine learning algorithms, including supervised learning algorithms, unsupervised learning algorithms, and reinforcement learning algorithms. Supervised learning algorithms are trained on labeled data, where the correct output is provided for each input. Unsupervised learning algorithms are trained on unlabeled data, and must find patterns and relationships in the data without guidance. Reinforcement learning algorithms learn by interacting with an environment and receiving rewards or penalties for certain actions.

Machine learning has been applied to a wide range of problems, including image and speech recognition, natural language processing, and predictive modeling. It has also played a key role in the development of self-driving cars and other emerging technologies.

In the following subsections, we will discuss one particular field of machine learning - NLP. Also, we will introduce named entity recognition as one of NLP tasks.

1.2 Natural Language Processing Introduction

Natural language processing (NLP) is a field of artificial intelligence that focuses on developing algorithms and systems that can understand and process human language. NLP involves a wide range of tasks, including language translation, text classification, named entity recognition, part-of-speech tagging, and sentiment analysis.

NLP systems often rely on machine learning algorithms to learn patterns in large datasets of text or speech and make predictions or decisions based on that learning. NLP tasks also often require the development of feature functions that can extract relevant information from text and transform it into a numerical representation that can be used as input to a machine learning model.

On the whole, NLP has a wide range of applications, including machine translation, text summarization, information extraction, and chatbots. It has also played a key role in the development of virtual assistants and other emerging technologies.

1.3 Goal of the Thesis

Doplním ve finální bakalářské práci

1.4 Structure of the Report

Doplním ve finální bakalářské práci

Chapter 2
Problem description

Named entity recognition (NER) is a natural language processing (NLP) task that involves identifying and classifying named entities in text. Named entities are specific people, organizations, locations, or other entities that are mentioned by name in the text.

NER systems typically work by identifying words or phrases in the text that correspond to named entities, and then classifying them into predefined categories such as person, organization, location, etc. For example, a NER system might identify the words Tom Cruise and New York in a text and classify them as a person and a location, respectively.

There are several approaches to NER, including rule-based methods, which use a set of predefined rules to identify and classify named entities, and machine learning-based methods, which use supervised or unsupervised learning algorithms to learn patterns in the data and classify named entities.

NER has a wide range of applications, including information extraction, knowledge base construction, and text summarization. It is also an important component of many natural language processing systems, as named entities often provide key contextual information about the given text.

2.1 Use - cases

Doplním ve finální bakalářské práci

2.2 Related tasks

Doplním ve finální bakalářské práci

2.3 IOB tagging

IOB tagging, also known as Inside-Outside-Beginning tagging, is a scheme for annotating text data with named entity labels. In IOB tagging, each word in the text is assigned a label that indicates whether it is the beginning of a named entity (B), inside a named entity (I), or outside a named entity (O).

For example, consider the following text: Tom Cruise was born in New York. In IOB tagging, the words Tom and Cruise would be labeled as B-PER (beginning of a person entity) and I-PER (inside of a person entity), respectively. The word New would be labeled as B-LOC (beginning of a location entity). Lastly, the word York would be labeled as I-LOC (inside a location entity), as it is part of the same location entity as New. The remaining words in the text would be labeled as O (outside a named entity). Another example is shown in 2.1.

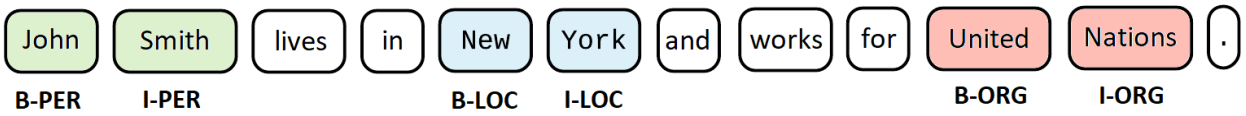

Figure 2.1. IOB tagging example [1].

IOB tagging is a common scheme for annotating named entities in text data and is used as a input format for many NER libraries or models.

Chapter 3
Background

To label entities in given dataset we can use multiple algorithms to do so. Most common sequence labelling algorithms divide into the following categories [2]:

- Rule-based methods using vocabularies or dictionaries.
- Feature-based supervised learning approaches such as: HMM or CRF.
- Deep learning techniques using machine learning models, for example: RNN or LSTM.

These three major categories are most commonly used to tackle NER related tasks. Particularly, Feature-based and deep learning approches represent state-of-the-art technology widely employed for extracting named entitities from text with high accuracy.

In the next subchapters we will discuss specific background methods in more detail.

3.1 Rule-based methods

Rule-based methods for named entity recognition (NER) involve using a set of predefined rules to identify and classify named entities in text. These rules might include regular expressions, dictionaries, and other types of linguistic patterns.

Rule-based NER systems typically work by applying a series of rules to the text, one at a time, to identify and classify named entities. For example, a rule-based NER system might use a regular expression to identify words that match a particular pattern, such as all capitalized words, and classify them as named entities. Alternatively, the system might use a dictionary of named entities to look up words in the text and classify them based on their definitions.

Advantages of such approaches are simplicity, customizability or efficiency. Rule-based systems are relatively simple to implement and can be easy to understand and customized to the specific needs of a particular application by modifying the rules or adding new ones. On the top of that, rule-based systems can be fast to execute, particularly for small or moderate-sized datasets.

On the contrary, rule-based systems may not generalize well to new or unseen data, as they are based on specific, predefined rules that may not apply in all cases. They are also limited in their ability to capture complex patterns in the data, as they rely on predefined rules rather than learning from the data itself. Moreover, rule-based systems are sensitive to errors in the rules, and may produce incorrect results if the rules are incomplete or incorrect.

On the whole, rule-based NER systems are a useful tool in certain situations, but they have limitations that may make them less effective than more advanced machine learning methods.

3.2 Markov Models

Hidden Markov Models is statistical model which augments Markov chains with hidden states. In the next subchapers we will introduce the concept of Markov Chains and then use it to describe its buildup in HMM.

3.2.1 Markov Chains

Standard Markov chains is the mathematical concept that describes states and transitions between them, which are weighted by probabilities. Formally, we define Markov chains in 3.1.

$Q = q_1 q_2 ... q_N$	set of N states
$A = a_{11} a_{12} ... a_{N1} ... a_{NN}$	transition probability matrix with $\sum_{j=1}^{N} a_{ij} = 1 \ \forall i$
$\pi = \pi_1, \pi_2 ... \pi_N$	initial probability distribution with $\sum_{i=1}^{N} \pi_i = 1$

Table 3.1. Formal definition of Markov chain. Table taken from [3].

The defining characteristic of Markov chains is established number of future states to which we can transition. It means that the very probability of transitioning to next state depends solely on the current state we are in. This attibute is called Markov assumption [4] which is mathematically written in (1).

$$P(q_i|q_1, ...q_{i-1}) = P(q_i|q_{i-1}) \tag{1}$$

To represent Markov chains, we use finite state machines or its variation in the form of transition matrix. Example of finite state machine is shown in 3.1.

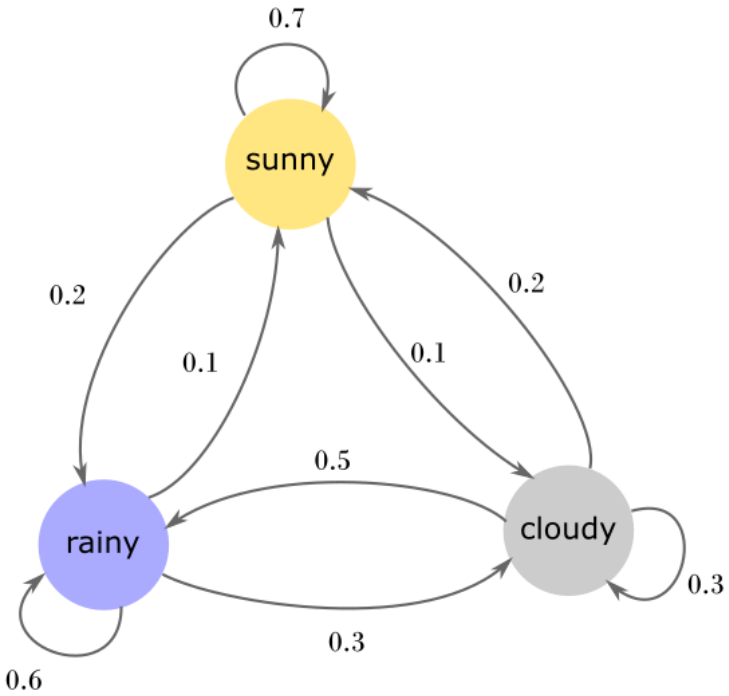

Figure 3.1. Representation of Markov chain using finite state machine [5].

3.2.2 The Hidden Markov Models

While Markov chains are used to determine the probability of sequence of observable states, in HMM these states are hidden. In Hidden Markov model we observe a sequence of emissions but we do not know the sequence of states which generated these emissions. The formalization is shown in 3.2.

$Q = q_1 q_2 ... q_N$	set of N states
$A = a_{11} a_{12} ... a_{N1} ... a_{NN}$	transition probability matrix with $\sum_{j=1}^{N} a_{ij} = 1 \; \forall i$
$\pi = \pi_1, \pi_2 ... \pi_N$	initial probability distribution with $\sum_{i=1}^{N} \pi_i = 1$
$B = b_i(o_t)$	emission probabilities
$O = o_1 o_2 ... o_T$	T observations

Table 3.2. Formal definition of Hidden Markov Model. Table taken from [3].

Using HMM we want to discover the suitable sequence of states by maximizing transition and emission probabilities [6]. In NER tasks, hidden states correspond to individual labels of named entities and observations are tokens from given dataset.

The transition probability is a probability distribution that specifies the probability of transitioning from one latent state to another given current state. In other words, the probability of appearing in the next hidden state depends on the hidden state we are in. Probability formula is shown in (2).

$$P(t_1 ... t_N) \approx \prod_{i=1}^{N} P(t_i | t_{i-1}) \tag{2}$$

Second probability distribution that defines HMM is emission probability. Emission probability establish the probability of observing a particular observation given a particular latent state. In HMM it is used to model the relationship between the latent states and the observations. Mathematical definition is described in (3).

$$P(w_1 ... w_N | t_1 ... t_N) \approx \prod_{i=1}^{N} P(w_i | t_i) \tag{3}$$

Given the formulas (2) and (3), we discover sequence of hidden states using (4). Generally, the process of assigning the sequence of observations to the seqeunce of hidden states is called decoding. The most common decoding algorithm is Viterbi algorithm which uses the principles of dynamic programming.

$$\hat{t} = \arg\max_{t_1 ... t_N} P(t_1 ... t_N | w_1 ... w_N) \approx \arg\max_{t_1 ... t_N} \prod_{i=1}^{N} P(w_i | t_i) P(t_i | t_{i-1}) \tag{4}$$

Firstly, Viterbi creates probability matrix with rows as hidden states and columns representing observations. Each index in the matrix indicates $v_t(j)$ which means being in the j state after t observations and visiting $t-1$ most probable hidden states. Since Viterbi is an instance of dynamic programming, it fills each cell of the matrix recursively and computes the most probable sequence of states that lead to the state we are in [3].

The formula for $v_t(j)$ is shown in (5). After the matrix is filled, the algorithm then backtracks through cells to find the most likely sequence of latent states.

$$v_t(j) = \max_{i=1}^{N} v_{t-1}(i) a_{ij} b_j(o_t) \tag{5}$$

Overall, while HMMs can be a useful tool for NER tasks, they have limitations that may make them less effective than more advanced machine learning models in some cases. For example, HMMs are limited in their ability to capture long-range dependencies: HMMs are based on the assumption that the current state depends only on the previous state, which can limit their ability to capture long-range dependencies in the data.

3.3 Conditional Random Fields

Even though HMM is a solid approach to label named entities, it has its limitations as it only takes relationship of two subsequent entities into account. To enrich our classifier with feature functions for better accuracy, we can use one of the most prominent algorithm for such tasks - Conditional Random Fields. In addition to probability distributions, CRF classifiers incorporate feature functions to reflect word capitalization or length in labeling named entities.

Feature functions are functions that map input data to a set of features that are used as input to a model. Feature functions are used to extract relevant information from raw data and transform it into a form that is more suitable for the model to learn from. For the purpose of labeling named entities, we can use large variety of feature functions [7], such as:

- word length
- word suffix or prefix
- is number
- word embeddings
- gazetteer features

CRFs use a set of feature functions that capture the relationship between the current label and the previous and subsequent labels, as well as the relationship between the current observation and the previous and subsequent observations. These feature functions are used to define the probability distribution over the possible label sequences.

In contrast to HMMs, CRF directly computes conditional probability to determine most possible sequence of named entitities in given text [8]. The formula is shown in (6).

$$\hat{Y} = \arg\max_{Y \in \mathscr{Y}} \sum_{i=1}^{N} \sum_{k=1}^{K} w_k f_k(y_{i-1}, y_i, X, i) \tag{6}$$

To solve formula from (6), CRF uses, its linear chain variant in particular, Viterbi algorithm. The same algorithm used in the HMMs. Viterbi recursively visits each cell of the $N \times T$ matrix and computes probability shown in (7). As soon as the matrix is filled, Viterbi algorithm then backtracks through cells to find the most likely sequence of entity labels [3].

$$v_t(j) = \max_{i=1}^{N} v_{t-1}(i) \sum_{k=1}^{K} w_k f_k(y_{t-1}, y_t, X, t) \quad 1 \leq j \leq N, 1 < t \leq T \tag{7}$$

CRFs can capture long-range dependencies. CRFs are able to capture dependencies between elements that are far apart in the sequence, which is useful for tasks such as named entity recognition where the label of a word may depend on the labels of words that are far away in the sentence. On the other hand, CRFs can be slow to train and predict. CRFs can be computationally intensive to train and predict, particularly for large datasets with many elements. Overall, CRFs are a powerful tool for sequence labeling tasks, but they have some limitations that may make them less effective than more advanced machine learning models.

Chapter 4
Experiment

In this chapter, we will discuss the implementation part of this project. In particular, we describe used technologies and datasets in our implementation as well as results we achieved.

4.1 Solution outline

We used spaCy[1] Python library to train our own named entity recognition model on two provided datasets. In addition, we used pretrained model[2] provided by spaCy library We also coded baseline methods using string maching and fasttext[3] word embeddings with cosine similarity. Everything was tested and programmed in Python 3.9.

4.2 Evaluation metrics

There are several evaluation metrics that can be used to assess the performance of a named entity recognition (NER) system. Some common metrics include: recall, precision and F1 score.

Recall is the fraction of true named entities that were correctly predicted. It is calculated as the number of true positives divided by the total number of true named entities. The formula is shown in (1).

$$\frac{tp}{tp + fn} \tag{1}$$

Precision is the fraction of predicted named entities that are correct. It is calculated as the number of true positives (correctly predicted named entities) divided by the total number of predicted named entities as shown in (2).

$$\frac{tp}{tp + fp} \tag{2}$$

The F1 score is a measure of the balance between precision and recall. It is calculated as the harmonic mean of precision and recall depicted in (3).

$$\frac{2 \cdot precision \cdot recall}{precision + recall} \tag{3}$$

To put mentioned metrics into perspective, we can use micro average. Micro average is a method of calculating the overall performance of a classification system on a dataset by aggregating the performance of the system on individual samples. We calculate micro average for every NER evaluation metric:

[1] https://spacy.io/
[2] https://spacy.io/models/en##en_core_web_sm/
[3] https://fasttext.cc/

- Precision micro average

$$\frac{\sum_{i=1}^{N} tp_i}{\sum_{i=1}^{N} tp_i + fp_i} \tag{4}$$

- Recall micro average

$$\frac{\sum_{i=1}^{N} tp_i}{\sum_{i=1}^{N} tp_i + fn_i} \tag{5}$$

- F1 score micro average

$$\frac{\sum_{i=1}^{N} tp_i}{\sum_{i=0}^{N} tp_i + \frac{1}{2} \cdot (fp_i + fn_i)} \tag{6}$$

4.3 Datasets

We tested our models on two datasets - conll2003[4] and OntoNotes 5.0[5].

4.4 Results

On conll2003 dataset we achieved results shown in the table 4.1. Complete results can be found in the table (cela tabulka v appendixu), where we present evaluation metrics on individual entitities.

Micro average of\Evaluation metric	Precision	Recall	F1 score
fasttext method	0.32	0.77	0.45
dictionary method	0.55	0.51	0.53
trained model using spaCy	0.84	0.82	0.83
pretrained model from spaCy	0.23	0.32	0.27

Table 4.1. Overall results achieved on conll2003 dataset.

Futhermore, the table 4.2 shows the results obtained by running mentioned algorithms on OntoNotes 5.0 dataset. Individual evaluation of entity labels is presented in Appendix A.

Micro average of\Evaluation metric	Precision	Recall	F1 score
fasttext method	0.15	0.65	0.24
dictionary method	0.10	0.70	0.18
trained model using spaCy	0.74	0.77	0.75
pretrained model from spaCy	0.65	0.82	0.73

Table 4.2. Overall results achieved on OntoNotes 5.0 dataset.

[4] `https://huggingface.co/datasets/conll2003`
[5] `https://paperswithcode.com/dataset/ontonotes-5-0`

4.5 Conclusion

In conclusion, on conll2003 we have achieved highest scores in terms of evaluation metrics when we used newly-trained spaCy model on the dataset. It outperformed all the other approches and surpassed 80 % level in all three evaluation categories. However, pretrained model from spaCy library showed really bad performance. The reason behind bad scores of this approach is the fact that such pretrained model was originally trained on datasets with different entity labels compared to entity categories in conll2003 dataset. This resulted into overall bad results and the worst scores in evaluation categories. Lastly, we have used dictionary and fasttext methods on the conll2003 dataset. This methods achieved similar results that are nowhere near the performance of newly-trained spaCy model.

A the same time, we used indentical algorithms on OntoNotes dataset. As on conll2003 dataset, highest scores were achieved using newly-trained spaCy model. But in this case, performance of the pretrained model from spaCy library was similarly good. It was due to the fact that such pretrained model was trained on many datasets including OntoNotes so it includes alike entity labels. Fasttext and dictionary methods showed very bad scores in all evaluation categories. This classifies them as baseline approches.

References

[1] Steve Zheng. *Sequence Labeling.* https://stevezheng23.github.io/sequence_labeling_tf/.

[2] Sichang Tu. *Experiments on Approaches to Named Entity Recognition in IsiZulu.* Georgetown University, 2021. http://hdl.handle.net/10822/1062365.

[3] D. Jurafsky, and J.H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition.* Pearson Prentice Hall, 2009. ISBN 9780131873216. https://books.google.cz/books?id=fZmj5UNK8AQC.

[4] Brilliant.org. *Markov Chains.* https://brilliant.org/wiki/markov-chains/.

[5] deparkes. *Markov Chains.*

[6] MathWorks. *Hidden Markov Models (HMM).* https://www.mathworks.com/help/stats/hidden-markov-models-hmm.html.

[7] Pavel Král. *Features for named entity recognition in Czech language.* 2011. https://www.researchgate.net/publication/256605620_Features_for_named_entity_recognition_in_Czech_language.

[8] Nita Patil, Ajay Patil, and B.V. Pawar. Named Entity Recognition using Conditional Random Fields. *Procedia Computer Science.* 2020, 167 1181-1188. DOI https://doi.org/10.1016/j.procs.2020.03.431. International Conference on Computational Intelligence and Data Science.

Appendix A
Complete results

In this section, we specify complete results achieved on conll2003 and OntoNotes 5 datasets. For every used method and dataset, we created individual table. These detailed results are presented in tables A.3, A.4, A.5, A.6, A.7, A.8, A.9 and A.10.

Entity\Evaluation metric	Precision	Recall	F1 score
PERSON	0.32	0.79	0.46
TIME	0.12	0.73	0.2
CARDINAL	0.3	0.51	0.38
ORG	0.09	0.51	0.16
GPE	0.62	0.69	0.65
DATE	0.47	0.63	0.54
WORK_OF_ART	0.01	0.49	0.02
LAW	0.01	0.5	0.01
MONEY	0.43	0.74	0.55
LOC	0.06	0.67	0.11
ORDINAL	0.3	0.96	0.46
PERCENT	0.14	0.24	0.18
NORP	0.41	0.78	0.54
EVENT	0.02	0.28	0.03
FAC	0.03	0.73	0.06
QUANTITY	0.04	0.67	0.08
LANGUAGE	0.05	0.5	0.09
PRODUCT	0.01	0.43	0.03

Table A.3. Complete results achieved on OntoNotes dataset using fasttext method.

Entity\Evaluation metric	Precision	Recall	F1 score
LOC	0.39	0.72	0.5
PER	0.34	0.85	0.48
ORG	0.29	0.79	0.43
MISC	0.23	0.62	0.33

Table A.7. Complete results achieved on conll2003 dataset using fasttext method.

Entity\Evaluation metric	Precision	Recall	F1 score
PERSON	0.04	0.52	0.08
TIME	0.05	0.59	0.10
CARDINAL	0.13	0.83	0.23
ORG	0.10	0.57	0.17
GPE	0.17	0.88	0.28
DATE	0.15	0.81	0.25
WORK_OF_ART	0.06	0.25	0.10
LAW	0.16	0.30	0.21
MONEY	0.46	0.37	0.41
LOC	0.27	0.60	0.37
ORDINAL	0.21	0.97	0.34
PERCENT	0.96	0.75	0.84
NORP	0.10	0.92	0.18
EVENT	0.35	0.46	0.40
FAC	0.09	0.11	0.10
QUANTITY	0.21	0.20	0.21
LANGUAGE	0.00	0.41	0.00
PRODUCT	0.06	0.24	0.10

Table A.4. Complete results achieved on OntoNotes dataset using dictionary method.

Entity\Evaluation metric	Precision	Recall	F1 score
PERSON	0.52	0.86	0.65
TIME	0.53	0.65	0.59
CARDINAL	0.62	0.81	0.70
ORG	0.65	0.79	0.71
GPE	0.75	0.89	0.81
DATE	0.71	0.87	0.78
WORK_OF_ART	0.33	0.24	0.28
LAW	0.44	0.36	0.40
MONEY	0.84	0.87	0.85
LOC	0.50	0.71	0.59
ORDINAL	0.53	0.86	0.66
PERCENT	0.84	0.89	0.86
NORP	0.74	0.88	0.80
EVENT	0.55	0.49	0.52
FAC	0.36	0.23	0.28
QUANTITY	0.69	0.59	0.64
LANGUAGE	0.59	0.45	0.51
PRODUCT	0.44	0.43	0.44

Table A.5. Complete results achieved on OntoNotes dataset using spaCy pretrained model.

Entity\Evaluation metric	Precision	Recall	F1 score
PERSON	0.76	0.84	0.80
TIME	0.55	0.38	0.45
CARDINAL	0.59	0.80	0.68
ORG	0.77	0.70	0.73
GPE	0.79	0.89	0.84
DATE	0.73	0.77	0.75
WORK_OF_ART	0.35	0.11	0.16
LAW	0.47	0.16	0.24
MONEY	0.83	0.81	0.82
LOC	0.50	0.61	0.55
ORDINAL	0.54	0.81	0.64
PERCENT	0.82	0.88	0.85
NORP	0.80	0.85	0.82
EVENT	0.68	0.22	0.34
FAC	0.37	0.10	0.16
QUANTITY	0.71	0.39	0.50
LANGUAGE	0.75	0.14	0.23
PRODUCT	0.79	0.26	0.39

Table A.6. Complete results achieved on OntoNotes dataset using new spaCy trained model.

Entity\Evaluation metric	Precision	Recall	F1 score
LOC	0.84	0.81	0.83
PER	0.28	0.15	0.19
ORG	0.77	0.49	0.6
MISC	0.27	0.67	0.38

Table A.8. Complete results achieved on conll2003 dataset using dictionary method.

Entity\Evaluation metric	Precision	Recall	F1 score
LOC	0.38	0.02	0.03
PER	0.73	0.57	0.64
ORG	0.46	0.29	0.36
MISC	0.07	0.58	0.13

Table A.9. Complete results achieved on conll2003 dataset using spaCy pretrained model.

Entity\Evaluation metric	Precision	Recall	F1 score
LOC	0.89	0.87	0.88
PER	0.85	0.87	0.86
ORG	0.81	0.74	0.77
MISC	0.76	0.76	0.76

Table A.10. Complete results achieved on conll2003 dataset using new spaCy trained model.