

CAPSTONE PROPOSAL: NAMED ENTITY RECOGNITION MODEL ON CONLL 2003 DATASET WITH A BROWSER DEPLOYMENT

Domain Background

Named Entity Recognition (NER) is a typical and one of the most popular tasks from a Natural Language Processing (NLP) domain – a field in which a computer is programmed to analyze human language with an end goal of capturing its different nuances and learning its representation¹. As NER constantly serves as a basis for various NLP tasks including, among others, machine translation, question answering or relation extraction² it has been an object of a thorough research throughout recent years. Similar to other natural language related computational problems, techniques applied for NER historically range from simple rule-based approaches, through more sophisticated statistical or machine learning algorithms (with feature engineering included), to the most advanced and novel solutions incorporating deep learning³. As per my interest in both neural networks and NLP I would additionally like to present the results of this project by deploying a web application.

Problem Statement

In a nutshell, NER is considered a task of searching and tagging important objects (i.e. named entities) like locations, people or organizations within a chunk of chosen text⁴. From NLP and computational perspective NER is recognized as a sequence labelling problem⁵ – given a text input, an algorithm should be capable of assigning aforementioned entity types to particular elements (tokens) of that sequence. An example of NER in practice is shown in the Picture 1.

¹ Young, T., Hazarika, D., Poria, S., & Cambria, E. (2018). Recent Trends in Deep Learning Based Natural Language Processing [Review Article]. IEEE Computational Intelligence Magazine, 13, 55-75.

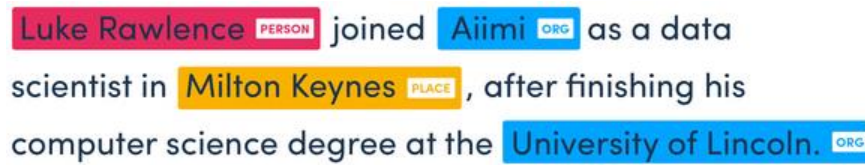
² Li, J., Sun, A., Han, J., & Li, C. (2018). A Survey on Deep Learning for Named Entity Recognition. ArXiv, abs/1812.09449.

³ Ibid.

⁴ Roy, A. (2021). Recent Trends in Named Entity Recognition (NER). ArXiv, abs/2101.11420.

⁵ Ibid.

Picture 1: Named Entity Recognition example



Luke Rawlence PERSON joined Aiimi ORG as a data scientist in Milton Keynes PLACE, after finishing his computer science degree at the University of Lincoln. ORG

Source: <https://www.aiimi.com/insights/aiimi-labs-on-named-entity-recognition>

Datasets and Inputs

For this project an English version of CoNLL-2003 dataset is used⁶. The dataset consists of Reuters news stories gathered between 1996 and 1997 and is widely exploited across many NER researchers to produce state of the art results. Data has been obtained from <https://deepai.org/dataset/conll-2003-english> and is included as a part of submission.

The dataset is divided into train, validation (development) and test files with a total of over 22k annotated sentences. Each file contains four single space separated columns with the first column being a word (token) of a given sentence (sentences are encoded in a row-wise manner) and the last one representing an annotated entity⁷, i.e. label to predict for a particular word. Second and third ones are part-of-speech and syntactic tags respectively and are of our less interest in the NER context.

Entities are split into four groups: person (B-PER, I-PER), location (B-LOC, I-LOC), organization (B-ORG, I-ORG) and miscellaneous (B-MISC, I-MISC). There is also a separate label O for no entity types. The I- prefix is applied in case of multi word entities of the same type to represent words following the first one. For example [European, Commission] would be tagged as [B-ORG, I-ORG]. Therefore we end up with multiclass prediction problem with a total of nine classes.

Solution Statement

Named Entity Recognition problem, due to its above-mentioned sequential nature, has a potential to be modeled by various deep learning algorithms with recurrent neural networks

⁶ Tjong Kim Sang, E. F., & De Meulder, F. (2003). Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In W. Daelemans, & M. Osborne (Eds.), Proceedings of CoNLL-2003, Edmonton, Canada (pp. 142-145). Morgan Kaufman Publishers.

⁷ <https://www.clips.uantwerpen.be/conll2003/ner/>

or RNNs (and long short-term memory networks or LSTMs in particular) making impression of the most appropriate ones⁸. Taking a numerically encoded sequential word input (whether those are one-hot vectors, pretrained word embeddings or embeddings trained from scratch) deep model returns a similar sequence with entities assigned at each position. Furthermore, the whole system might be enhanced by working on a character level (as opposed to word level) and incorporating convolutional neural networks (CNNs)⁹. Both LSTMs and CNNs provide means for automated feature engineering.

Benchmark Model

Two benchmarking approaches are utilized for the proposed solution. First off, a naive approach that serves as a baseline is introduced – entities from the training data are simply transferred and searched for in the testing dataset. F1 score for such approach is publicly available and equals 59.61¹⁰. Secondly, model is benchmarked against more advanced, deep learning based solutions available on <https://paperswithcode.com/sota/named-entity-recognition-ner-on-conll-2003>. An important thing to note is that the main idea of this project is not to push the state of the art results but rather to create a decently performing and generalizable algorithm that would be further deployed in a browser.

Evaluation Metrics

Problem solution is evaluated with F1 score, a typical metric for assessing NER systems, which is a weighted average of both recall and precision and can be calculated according to the following formula¹¹:

$$F1 = 2 * (precision * recall) / (precision + recall)$$

An exact match only is considered a correct prediction therefore any partial matches where algorithm correctly identifies an entity existence but predicts the wrong type are

⁸ There are also novel transformer based architectures but they are not considered in this project.

⁹ Chiu, J.P., & Nichols, E. (2016). Named Entity Recognition with Bidirectional LSTM-CNNs. Transactions of the Association for Computational Linguistics, 4, 357-370.

¹⁰ Tjong Kim Sang, E. F., & De Meulder, F. Ibid.

¹¹ https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html

discarded. Recall and precision metrics are computed globally by counting the total number of true positives, false negatives and false positives, therefore F1 metric is micro-averaged¹².

Additionally, during model training categorical crossentropy and categorical accuracy¹³ are monitored as the first one is also the model loss while the second might be a good indicator of system's general quality.

Project Design

Project is mainly conducted in Python, Tensorflow with Keras and Amazon SageMaker with an inclusion of JavaScript (required for web model deployment) and simple html script.

Theoretical steps are summarized by the following workflow:

1. Extract inputs and corresponding tags (labels) from provided CoNLL-2003 txt files.
2. Conduct any required text preprocessing if necessary (remove interpunction, lower text, etc.).
3. Tokenize and create vocabulary.
4. Text vectorization (representing texts in numerical form) of inputs and tags.
5. Design LSTM based model architecture with a potential CNN extension.
6. Train model (or potentially models to verify different architectures) on SageMaker, evaluate performance on test data to compare against benchmarks.
7. Retrieve keras model object for the final solution and convert with tensorflow.js.
8. Design simple html website powered by tensorflow.js converted keras model where an end user can type in an English text and get highlighted named entities. Model and website are publicly hosted.
9. Website tests and final deployment.

¹² Ibid.

¹³ https://www.tensorflow.org/api_docs/python/tf/keras/metrics