Report on Named Entity Recognition Using CNN and Hugging Face Transformer Models

Syed Zain Ali
MSc. Data Science
Matriculation number: 4373556
syed.ali@uni-bielefeld.de
15.08.2024

1. Introduction

Named entity recognition (NER), also called entity chunking or entity extraction, is a component of natural language processing (NLP) that identifies predefined categories of objects in a body of text. [1] Named entity recognition is an important task of information extraction systems. The CoNLL-2003 dataset uses the IOB2 tagging scheme and consists of four columns: word, part-of-speech (POS) tag, syntactic chunk tag, and named entity tag. This structured data format allows for the effective application of both CNN and transformer models. [2] The shared task of CoNLL-2003 concerns language-independent named entity recognition. [3] Named entities are phrases that contain the names of persons, organizations and locations. Example: [ORG U.N.] official [PER Ekeus] heads for [LOC Baghdad]. This sentence contains three named entities: Ekeus is a person, U.N. is an organization and Baghdad is a location. The data contains entities of four types: persons (PER), organizations (ORG), locations (LOC) and miscellaneous names (MISC).
This report investigates two different approaches to NER: a traditional deep-learning model based on Convolutional Neural Networks (CNNs) and a modern transformer-based model using the BERT (Bidirectional Encoder Representations from Transformers) architecture from Hugging Face. Both models are applied to the CoNLL-2003 dataset and their results are compared.

2. Related Work

Early work in NER systems in the 1990s was aimed primarily at extraction from journalistic articles. Attention then turned to processing of military dispatches and reports. Later stages of the automatic content extraction (ACE) evaluation also included several types of informal text styles, such as I-blogs and text transcripts from conversational telephone speech conversations. Since about 1998, there has been a great deal of interest in entity identification in the molecular biology, bioinformatics, and medical natural language processing communities. The most common entity of interest in that domain has been names of genes and gene products. [4] Historically, NER systems relied on rule-based methods and statistical models such as Conditional Random Fields (CRFs) and Hidden Markov Models (HMMs). These approaches, while effective, required extensive feature engineering and struggled with capturing context effectively. With the advent of deep learning, CNNs and Recurrent Neural Networks (RNNs) emerged as powerful tools, capable of automatic feature extraction and handling sequential data. More recently, transformers, particularly BERT, have

revolutionized NLP by providing deep contextual embeddings, significantly improving performance across various tasks, including NER. BERT's ability to consider the bidirectional context of words has set new benchmarks in the field, surpassing the performance of traditional models. An extra named entity category called MISC was added to denote all names which are not already in the other categories. This includes adjectives, like Italian, and events, like 1000 Lakes Rally, making it a very diverse category. There has been a lot of work on named entity recognition.

3. Data Challenges

I need to understand the encoding scheme and must encode and decode all data for correct prediction and label separation. Data pre-processing is challenging which involves proper tokenization. All data files contain one word per line with empty lines representing sentence boundaries. At the end of each line there is a tag which states whether the current word is inside a named entity or not. The tag also encodes the type of named entity. Each line contains four fields: the word, its part- of-speech tag, its chunk tag and its named entity tag. Words tagged with O are outside of named entities and the I-XXX tag is used for words inside a named entity of type XXX. Whenever two entities of type XXX are immediately next to each other, the first word of the second entity will be tagged B-XXX in order to show that it starts another entity.

3. Approach / Methods

3.1 Classic Deep-Learning Approach

In this approach, I developed a CNN-based model for the NER task. The model architecture includes an embedding layer that converts words into dense vectors, which are then processed by convolutional layers to capture local patterns. Finally, a dense layer with SoftMax activation classifies each word into one of the predefined entity categories.

Model Architecture:

- **Embedding Layer**: Converts words into dense vectors.
- **Convolutional Layers**: Extract local features from the input sequences.
- **Dense Layer with SoftMax Activation**: Produces the final classification of entities.

3.2 Pre-trained Transformer Approach

For the transformer-based approach, I utilized the BERT base model from Hugging Face, fine-tuned specifically for NER. BERT's ability to capture bidirectional context from the input text makes it particularly effective for NER tasks. To avoid complexity and computing time I've used Bert-base-cased model.

Model Components:

BERT Base-Cased Model: Provides deep contextual word representations with less complexity as compared to large case.

Classification Head for NER: Consists of a linear layer to classify each token based on BERT's embeddings.

3.3 Modifications

3.3.1 CNN Modifications:

Character-Level Embeddings: Added to enhance feature extraction by capturing sub-word information.
Increased Filter Size and Dropout: Adjusted the model by increasing the number of filters and adding dropout layers to prevent overfitting.
Batch Normalization: for stabilize training
Two activation functions: Softmax and Relu

3.3.2 BERT Modifications:

Conditional Random Field (CRF) Layer: Added a CRF layer on top of BERT to improve sequence labeling by considering the dependencies betIen output labels.
Hyperparameter Tuning: Adjusted learning rate and batch size and implemented gradient accumulation to improve model performance.
Efficiency: By saving storage and time
Batch Size: Decreased from 32 to 16

4. Experimental Settings

Dataset

The CoNLL-2003 dataset was used, consisting of labeled sentences split into training, validation, and test sets. Preprocessing included tokenization, padding, and converting the words and labels into numerical representations using vocabularies built from the training data.

Training

CNN Model:

Original	**Modified**
1 Convolutional layer with Filters=256	2 Convolutional layers with Filters=512 & 256 respectively
Learning Rate: 0.001 (Optimized by testing different combinations)	Learning Rate: 0.001 (Optimized by testing different combinations)
Dense Layer	Dense Layer
activation function softmax	2 activation functions, softmax and Relu
----	Batch Normalization
Epochs: 5	Epochs: 5
Embedding layer with output dimension 64	Embedding layer with output dimension 128

----	2 Dropout layers with 0.5 to prevent overfitting (Optimized by testing deifferent values)
Optimizer: Adam	Optimizer: Adam

BERT Model:

Original	**Modified**
Batch Size 32	Batch Size 16
Learning Rate: 2e-5	Learning Rate: 3e-5
---	50 Logging steps
Epochs: 3	Epochs: 3
----	Save limit to save diskspace
Weight decay	Weight decay
Took More time	Took less time

Evaluation Metrics

Both models are evaluated using precision, recall, and F1-score, which are standard metrics for NER tasks. These metrics provide a balanced view of the models' ability to correctly identify and categorize entities.

5. Results

CNN Model:

Original	**With Modifications**
Accuracy: 94%	Accuracy: 95%
F1-Score: 93%	F1-Score: 94%
Precision: 93%	Precision: 94%
Recall: 94%	Recall: 95%

BERT Model:

Original	**With Modifications**
Accuracy: 97.0%	Accuracy: 98.2%
F1-Score: 89%	F1-Score: 90%
Precision: 88%	Precision: 90%
Recall: 90%	Recall: 91%

The BERT model consistently outperformed the CNN model in accuracy, with the modified versions showing further improvements. The inclusion of character-level embeddings and dropout in the CNN model provided modest gains, while hyperparameter tuning in the BERT model led to significant improvements.

6. Discussion

The CNN model, despite its efficiency in capturing local dependencies, lacked the ability to understand the broader context as compared to Bert, which is critical in NER tasks. The inclusion of character-level embeddings improved the model's ability to capture sub-word information, resulting in better performance. However, the BERT model's deep contextual understanding, demonstrated a superior ability to correctly label entities, particularly in complex cases where context plays a crucial role. But it takes more computation time and memory.

The results indicate that both models, CNNs and BERT can be effective for NER. However, transformers like BERT, especially when fine-tuned and enhanced with sequence modeling layer, are more suitable for tasks requiring comprehensive context comprehension.

7. Conclusion

This study highlights the strengths and Lack-ness of CNN and transformer-based models for NER. CNNs, though efficient, are limited in their ability to capture global context, making them less effective for complex NLP tasks compared to transformers. The BERT model, with its ability to understand bidirectional context, outperforms CNNs, particularly when further optimized. Future work could explore hybrid models that combine the efficiency of CNNs with the contextual understanding of transformers to achieve even better performance in NER tasks. However, BERT models require more computation time and memory further research can be done to minimize BERT computation time and complexity.

References:

[1]https://www.ibm.com/topics/namedentityrecognition#:~:text=Named%20entity%20recognition%20(NER),in%20a%20body%20of%20text.
[2] https://huggingface.co/datasets/eriktks/conll2003
[3] https://aclanthology.org/W03-0419.pdf
[4] https://en.wikipedia.org/wiki/Named-entity_recognition