Fine Tuning Pretrained BERT Model for Named Entity Recognition

Kanishka Parganiha
Northeastern University
parganiha.k@northeastern.edu

Abstract—**In this paper, we are extending the past research studies on transfer learning for named entity recognition. We will be fine-tuning Pretrained BERT model, a state-of-the-art model introduced by the Google team for performing different Natural Language Processing tasks. In particular we are interested in entity recognition problems where the class labels in the source and target are same.**

In this work we propose pretraining larger Transformer based Encode-Decoder models on massive text corpora of MIT movie dataset by adding a token-level classifier on the top of the BERT model. We will then evaluate our model using F-1 score as this problem is typically a multi-class problem

Keywords—BERT, LSTM, Transformers, Attention, Encoder-decoder, NER, Information Extraction

1. Introduction

Named Entity Recognition, also know as information extraction/chucking is the process in which the algorithm extract real world noun entity from the text data an classifies them into predefined categories. The main challenge of Named Entity Recognition is to work on unstructured text data and then classifies different entities into person, place, organization etc. Before Deep Learning researcher used rule-based and dictionary based methods such Hidden Markov Model machine learning method and Conditional Random Field.

But the classical Supervised machine Learning method is based on machine in isolation i.e a single prediction model for a task using a single dataset. This approach requires a large number of training examples and performs best for well defined and narrow task. To improve this, research have introduced the concept of transfer learning by leveraging data from additional domains or tasks to train a model with better generalization properties.

Transfer Learning method is a machine learning method where a model developed for a task for performing a similar domain task. Transfer Learning is mostly used in computer vision and Natural Language Processing like text summarization, Information Extraction, Sentiment Analysis and Text Classification. The main intuition behind transfer learning if that this model which is trained on large dataset and text corpora serves as a generic model for similar NLP Task.

Many Transfer Learning techniques have been developed for tasks such as Named Entity Recognition (Daumè III 2007) and sentiment classification(Blitzer et al. 2007). A broad taxonomy of transfer learning scenario in Machine Learning is given in (Zhang el at., 2017).

In the recent year deep learning and cognitive learning has been widely used in multiple NLP fields. **Huang et al** proposed a model that combines the Bi-Directional LSTM(Long Short Memory) with the CRF. It can use both forward and backward input features to improve the performance of the NER task.

The LSTM and Recurrent neural network approach has been firmly used for sequence modeling task in NLP until the Google introduced a research on 2018 "**Attention is all you need"** which introduced the Transformers, the first sequence transduction model based on attention mechanism, replacing the Recurrent layer most commonly used in encoder-decoder architecture with multi-headed self attention.

Inspired from the success of transfer learning with language models, the main purpose of this paper is to introduce the structure and pre-training tasks of a pretrained model. I have used a Pre-trained BERT(Bi-directional encoder decoder representation from Transformers) model to facilitate the task of Named Entity Recognition. I have used **MIT Movie,** a open source dataset which containing entities like actors,director, film, genre, as the source domain and MIT Movie test dataset as target domain. The model is then evaluated using F-1 score as this is multi-classification problem.

2. Background:

As mentioned above, the performance of deep learning methods depends on the quality of labeled training sets. Therefore, researcher have proposed many pretrained models.

A. **ERNIE**

ERNIE is a pretrained model. In addition to a basic-level masking strategy, unlike BERT, ERNIE uses entity level and phrase masking strategies to obtain the language representation enhanced by knowledge.

ERNIE has the same model structure as BERT BASE which used 12 Transformer encoder layers, 768 hidden units and 12 attention heads.

B. **RoBERTa**

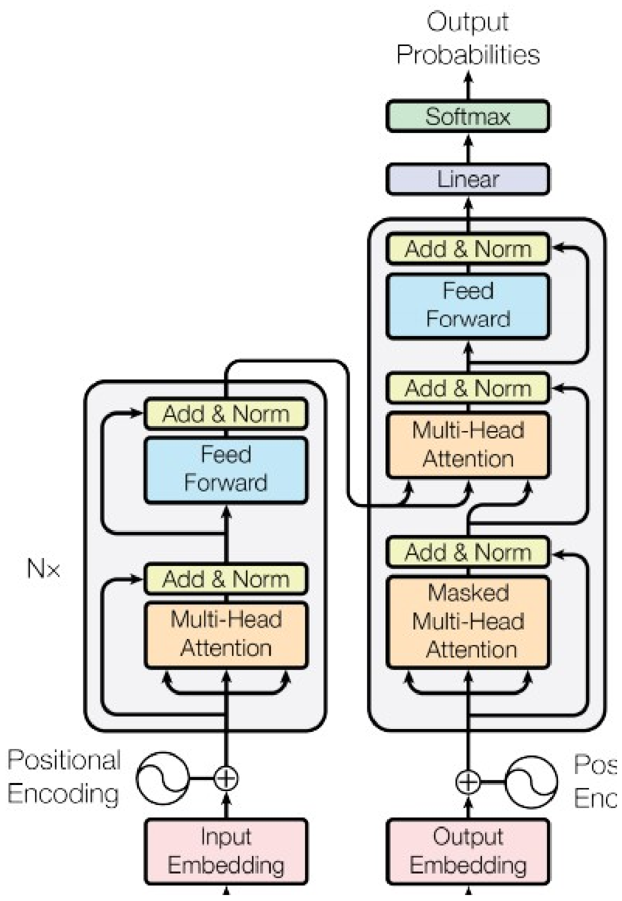
RoBERTa is similar to BERT except that it changes the masking strategy and removes the NSP task. Like ERNIE RoBERTa has the same model structure as BERT with 12 Transformer layers, 768 hidden layers and 12 self-Attention Layers.

RoBERTa removes the NSP task in BERT and changes the masking strategy for static to dynamic where RoBERTa changes masking position in every epoch.

There has been also work showing effective transfer from supervised task with larger datasets, such natural language inference(Conneau et. al., 2017) and machine translation (McCann et al.,2017). Computer vison has also demonstrated the importance of transfer learning from a large pretrained model.

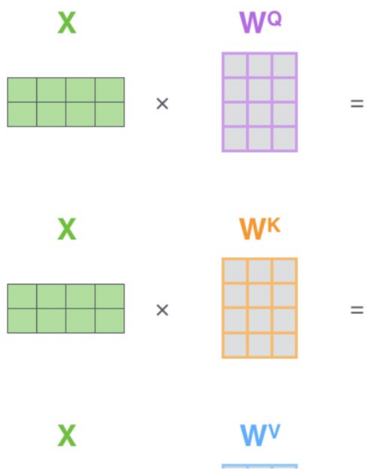
3. Approach

BERT is a stack of Transformers layers(Vaswani et al,. 2017). The variant of size "Base" has 12 layers of Transfromers, 12 self-Attention layers and a hidden state size of 768 per tokens. In total, BERT-Base has 110 million trainable parameters.



The Transformers implements some innovative concepts which are highly important to know about NER task.

A. MultiHead Attention: When a token is encoded as a weighted sum of its context. The weights are computed as a function of the context token(key) and the token to be encoded(query) and this function has trainable weights.



B. Stacking

Like RNN, multi attention layers are stacked. Interspersed with the self- attention layers are token-wise feed forward layers and all layers are connected via skip-connections.

C. Embedding layer.

Word embedding are the Feature vector representation of a word. The most BERT-based methods to generate sentence embeddings by simply averaging the word embedding of all words in a sentences.

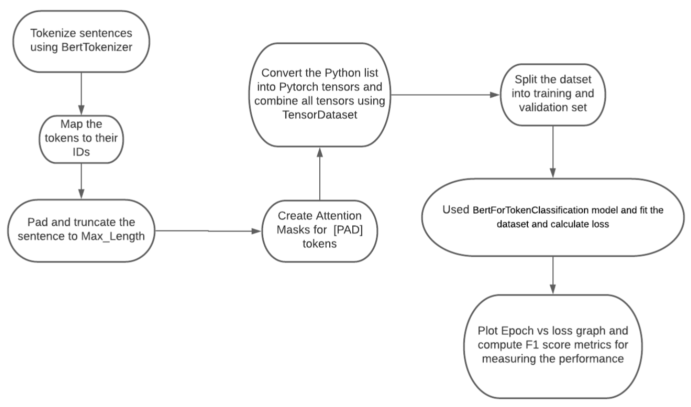
D. Encoder layer.

Encoder layer contains a multi-head Attention layer, a position-wise feed forward network and two "add and norm" connection block

E. Decoder layer.

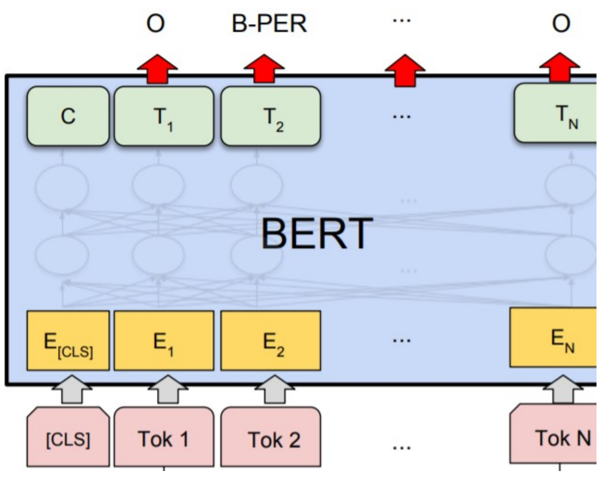
Decodes looks similar to the Encoder. However beside the two sub-layers(multi-head attention and positional encoding network), the decoder Transformer block contains third layer which applies multi-head attention on the output of the encoder stack

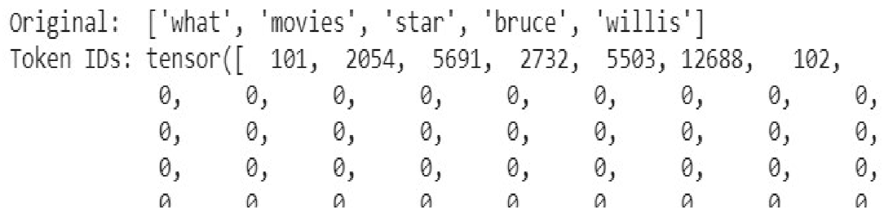
3.1 Main Process Flowchart


Tokenize sentences using BertTokenizer
Map the tokens to their IDs
Pad and truncate the sentence to Max_Length
Create Attention Masks for [PAD] tokens
Convert the Python list into Pytorch tensors and combine all tensors using TensorDataset
Split the datset into training and validation set
Used BertForTokenClassification model and fit the dataset and calculate loss
Plot Epoch vs loss graph and compute F1 score metrics for measuring the performance

3.1 Preprocessing:

Before fine-tuning the BERT it is required to prepare the dataset. We have to tokenize all the sentences using BERT tokenizer. It splits the tokens into sub-word tokens. These token are the special tokens which are prepend at the beginning and at the end of the input text sentence. These tokens are then convert into IDs using BertTokenizer. We then added Attention mask for padding and truncate the sentences to max length





```
Original:  ['what', 'movies', 'star', 'bruce', 'willis']
Token IDs: tensor([  101,  2054,  5691,  2732,  5503, 12688,   102,
              0,     0,     0,     0,     0,     0,     0,     0,     0,
              0,     0,     0,     0,     0,     0,     0,     0,     0,
              0,     0,     0,     0,     0,     0,     0,     0,     0,
              0      0      0      0      0      0      0      0      0
```

These input Ids which includes sentences ids, attention mask id and labels ids are then converted into Pytorch tensors

3.2 Splitting of datasets and Model Selections

After the conversion these tensors are then spilt into training and validation sets which is followed by Batch conversion of the set so that we can perform Batch Training.

For training I have used the model from **huggingface library named "BertForTokenClassification"** in pytorch for performing token-level predictions. This model is a fine tuning model that wraps BERT Model and adds token-level classifier on the top of the BERT model. The top-level classifier is a linear layer that takes as input the last hidden state of the sequence. We will then load the pre-Trained '**bert-based-cased**' model and provide the number of possible labels.

3.3 Fine Tuning the BERT Model:

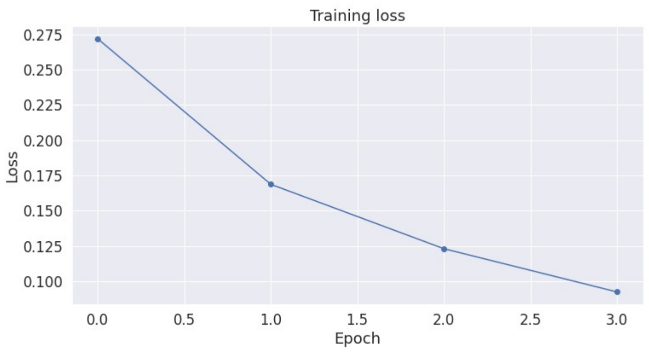
For fine tuning the model I have used AdaW optimizer with a small learning rate of **5e-5** epoch of **4.**

To update the learning rate I have also introduced a 'learning rate scheduler' which linearly increases the learning rate from 0.

As we are training in batches so I have to calculate average loss over the training data.

4.Result:

After training the dataset in Tesla T4 GPU for several minutes, we have plotted the plot between **Number of epochs and Loss**

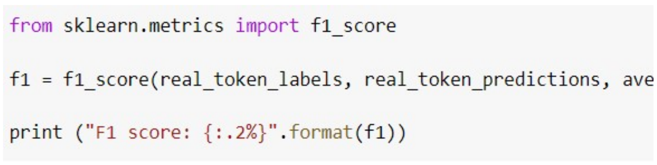


A we can notice that the loss value decreases as the number of epochs increases. At the end of the epoch 4, the loss value come to less than 1 which is a phenomenal result

After plotting the result, we then predicted the test of MIT movie dataset to check the performance of the model. As this model is a Multi-Class problem so we computed the F-1 score.

Before performing this, we have to follow the same steps of pre-processing of test data i.e Tokenization, mapping to IDs and then feeding it to the model.

The model got the F-1 score of **94.4 %** against the testing dataset.



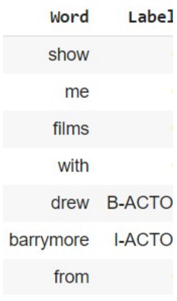
```
from sklearn.metrics import f1_score

f1 = f1_score(real_token_labels, real_token_predictions, ave

print ("F1 score: {:.2%}".format(f1))
```

4.1 Dataset format

The MIT movie dataset are annotated with BIO-schema and the label looks like this:



Word	Label
show	
me	
films	
with	
drew	B-ACTO
barrymore	I-ACTO
from	

5.Conclusion

In this paper we introduced the application BERT model in Named Entity Recognition. We evaluated the model using the F-1 score which comes to be **94.4%** which is better than **ERNIE model(93.37%)** and **BASELINE model(90.32%)**

6.References

[1] Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" last revised 24 May 2019

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, Illia Polosukhin "Attention is All you Need" last revised 6 Dec 2017

[3] Jingqing Zhang Yao Zhao Mohammad Saleh Peter J. Liu ”PEGASUS: Pre-training with Extracted Gap-sentences forAbstractive Summarization".

[4] Juan Diego Rodriguez, Adam Caldwell and Alexander Liu "Transfer Learning for Entity Recognition of Novel Classes" 2018

[5] Mengdi Zhu Zheye Deng Wenhan Xiong "NEURAL CORRECTION MODEL FOR OPEN-DOMAIN NAMED ENTITY RECOGNITION"