Machine Learning and Deep learning

Project Report

Nguyen Xuan Tung (M22.ICT.006)

1 Introduction

Named entity recognition (NER) is an essential task in Natural Language Processing (NLP), which involves identifying and classifying entities into predefined categories such as location, organization, person names, dates, and quantities. NER plays a pivotal role in numerous (NLP) applications, such as question-answering, content summarization, information retrieval and analysis, and machine translation. NER encounters several challenges: ambiguity in entity meanings, quality of data annotation, domain-specific terminology, and loanwords. BIO annotation is a widely used tagging format for NER. Each token in a sentence is tagged with one of these three prefixes: 'B-' indicates the start of an entity, 'I-' indicates the oken is inside an entity, and 'O' indicates non-entity tokens.

2 Dataset

The dataset that we used is CoNLL-2003, a benchmark dataset widely used for NER training and evaluation. The dataset consists of news stories within one year from August 1996. There are four types of entities: person names, organization, location, and miscellaneous. In total, we have nine classes. The distribution of each class is presented in Table 1.

Table 1: Class distribution in the CoNLL-2003 dataset.

	Train	Validation
B-LOC	7140	1837
B-ORG	6320	1341
B-PER	6600	1842
B-MISC	3426	922
I-MISC	1154	346
I-PER	4528	1303
I-ORG	3692	751
I-LOC	1156	257
O	160260	42338

The dataset exhibits a significant imbalance, which is understandable given that, within any given sentence, only a few tokens are classified as entities. The number of entities that do not fall into any specific category surpasses those that do by a factor of 20. We need to adopt a robust training strategy that can handle these challenges.

3 Methodology

The dataset is in CSV format, with each row corresponding to one word. The dataset also contains several empty rows. We applied several techniques to preprocess the dataset. We removed empty rows and concatenated all the words into a sentence with "." denoting the end of that sentence. As a result, we ended up with 7374 train sentences and 1874 validation sentences of different lengths.

We used DeBERTa-V3, an improved version of DeBERTa (Decoding-enhanced BERT with disentangled attention). DeBERTa introduces a new way to handle word relationships in a sentence by using a disentangled attention and enhanced mask decoder. The third version of DeBERTa replaces the mask language modeling (MLM) with replaced token detection (RTD) and uses a new method of sharing information called gradient-disentangled embedding sharing. The architecture of our model is presented in Fig. 1

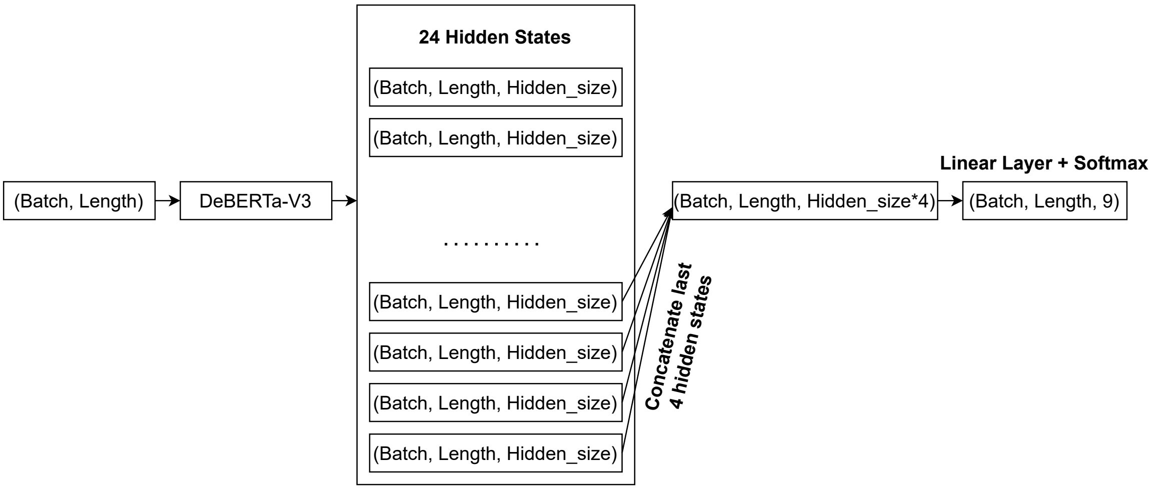
24 Hidden States
(Batch, Length, Hidden_size)
(Batch, Length, Hidden_size)
....
(Batch, Length, Hidden_size)
(Batch, Length, Hidden_size)
(Batch, Length, Hidden_size)
(Batch, Length, Hidden_size)
(Batch, Length)
DeBERTa-V3
Concatenate last 4 hidden states
(Batch, Length, Hidden_size*4)
Linear Layer + Softmax
(Batch, Length, 9)

Figure 1: The architecture of our model.

Given a sentence, we tokenize it and set `add_special_tokens`=False. We also add [CLS] at the beginning and [SEP] at the end of the sentence. We then pad the sentence to the length of 1024. We pass it through the DeBERTa-V3 model and get 24 hidden states of shape (`batch_size`, 1024, `hidden_size`). We concatenate the last four hidden states to get the hidden states of shape ($\texttt{batch\_size}, 1024, \texttt{hidden\_size} \times 4$). Then, we apply a fully connected layer with Softmax activation to get the output of shape (`batch_size`, 1024, 9).

4 Experiment and Results Analysis

4.1 Experiment

Hyperparameters. We employ AdamW optimizer with an initial learning rate of 9.41e-6 for DeBERTa-V3 large, 1.4e-5 for DeBERTa-V3 base and small, and apply Cosine Annealing learning rate scheduler to reduce the learning rate to 0 after four epochs. We set the batch size to 4 and seed to 42 for reproducibility. We use CrossEntropy as a loss function. We chose the model with the highest F1 score on the validation set for assessing the test set. Our experiments were performed on USTH's server with Intel(R) Core(TM) i9-10900K CPU @ 3.70GHz and a single RTX3090 GPU with 24GB VRAM.

4.2 Result Analysis

The best result of each model is presented in this table

Table 2: F1 score on CoNLL-2003 validation dataset and public leaderboard.

	Validation	Public Leaderboard
DeBERTa-V3 small	0.9450	0.9009
DeBERTa-V3 base	0.9534	0.9098
DeBERTa-V3 large	0.9528	0.9167

There is a strong correlation between the validation score and the public leaderboard score, indicating that improvements in the validation dataset consistently lead to higher scores on the public leaderboard. Our observations reveal that the performance across different models varies. The best score is achieved using DeBERTa-V3 large with a 0.9528 F1 score on the validation dataset and 0.9167 on the public leaderboard.

4.3 Conclusion & Future work

4.4 Conclusion

In this project, we created a simple NER training and evaluation baseline on the CoNLL-2003 dataset. We applied simple preprocessing techniques to clean the dataset and create sentences. We developed a NER deep learning model based on the backbone DeBERTa-V3. We successfully submitted our predictions on the leaderboard and got acceptable results ($>$ 0.9 in F1 score). The link to the repository is: https://github.com/ssjinkaido/NLP_NER_Final_Project and models is: https://drive.google.com/drive/folders/1RTfCe3-fEbIP1wH1lEIsVH6mxmG9IWNw?usp=sharing.

4.5 Future Work

There are several ideas that are subjectively evaluated to be new directions for further trials and experiments. We might apply different training strategies such as: Re-Initializing Transformer Layers, Layer-wise Learning Rate Decay, Dynamic Padding, and Adversarial Weight Perturbation. We assume that some small Large Language Models (LLM) may perform better than DeBERTa-V3.