COMP 7607: NATURAL LANGUAGE PROCESSING ASSIGNMENT 2

Shihao Liang
The University of Hong Kong
Student ID: 3036196673
E-mail: shihaol@connect.hku.hk

ABSTRACT

In this assignments, we present a detailed study on the application and optimization of LSTM and Transformer models for Named Entity Recognition (NER). Utilizing the CoNLL-2003 dataset, we construct and fine-tune two different neural network architectures to tag entities in text. Our work includes a meticulous examination of the LSTM and Transformer models, detailing their structure, hyperparameters, and functionality. Furthermore, we adopt a grid search approach to optimize key parameters such as learning rate, batch size, and training epochs, examining their impact on model performance metrics like accuracy, precision, recall, and F1-score. The findings offer valuable insights into the behavior of these models under various configurations, highlighting the importance of careful hyperparameter tuning in achieving optimal NER performance. This work aims to contribute to the understanding of applying advanced neural network techniques to linguistically structured data, offering a pathway for future explorations in NER and related NLP tasks.

1 INTRODUCTION

In recent years, Named Entity Recognition (NER) has emerged as a critical task in the field of Natural Language Processing (NLP). It involves identifying and classifying key information (entities) in text into predefined categories. This study delves into the application of advanced neural network architectures, specifically Long Short-Term Memory (LSTM) and Transformer models, for NER tasks. We focus on experimenting with these models using the CoNLL-2003 dataset, which provides a rich collection of annotated sentences for entity tagging.

Our approach involves a comprehensive exploration of model architectures and hyperparameter tuning to understand their impacts on NER performance. We utilize LSTM models known for their effectiveness in handling sequential data, and Transformer models, renowned for their ability to capture long-range dependencies in text. Through a systematic grid search methodology, we investigate the influence of key hyperparameters, such as learning rate, batch size, and the number of epochs, on model performance.

The primary objective of this study is to not only implement these sophisticated models but also to provide a detailed analysis of how various settings affect their learning capabilities and overall performance in tagging entities accurately.

2 MODEL ARCHITECTURE

2.1 LSTM-BASED MODEL FOR NER

Our LSTM-based model leverages the strengths of recurrent neural networks, particularly LSTM (Long Short-Term Memory) units, for sequence tagging tasks like NER. The model is composed of the following layers:

Layer	Description
Embedding Layer	Transforms input tokens into dense vectors of a specified dimension.
LSTM Layer	A bidirectional LSTM layer with configurable hyperparameters: embedding dimension (embedding_dim), hidden state size (hidden_dim), and number of layers (num_layers). The layer is set to batch_first=True.
Linear Layer	Maps the LSTM output to the tag space. The input dimension is twice the hidden_dim, and the output dimension is the tagset size (tagset_size).

In the following experiments, the embedding_dim and hidden_dim is set to 128 as default. The LSTM layers is set to 4.

2.2 Transformer-Based Model for NER

The Transformer Model utilizes the Transformer architecture. Its key components are as follows:

Component	Description
Model Dimensions	Parameterized with a dimensionality (d_model) and the number of heads (nhead) for multi-head attention mechanisms.
Encoder Layers	Multiple encoder layers, each containing a multi-head self-attention mechanism and a feedforward neural network. The number of layers is configurable (num_encoder_layers).
Dropout	Applied with a rate (dropout) to prevent overfitting.
Output Projection	A linear layer mapping the Transformer encoder outputs to the tag space.

To ensure a fair comparison between the LSTM and Transformer models, we have carefully aligned the model parameters. Specifically, in our experiments, we set the Transformer model's parameters – d_model, nhead, and num_encoder_layers – to mirror those of the LSTM model. Thus, the default values for these parameters are: d_model = 128, nhead = 4, and num_encoder_layers = 4. This alignment facilitates an equitable assessment of both models under similar parameter configurations.

3 Experiments

3.1 Grid Search for Hyperparameter

We employed a grid search strategy to optimize the hyperparameters for both the LSTM and Transformer models on the valid set. This involved varying one hyperparameter at a time while keeping others fixed. The tuned hyperparameters included:

- Learning Rate (LR): Experimentation with different rates for optimal model convergence.
- Batch Size: Tested to understand impact on performance and training efficiency.
- Epochs: Varied to determine sufficient training amount without overfitting.

3.2 Model Results

In our comparative analysis, as shown in Table 3, **transformer model generally outperformed the LSTM model following an extensive grid search on the validation set**. It is noteworthy, however, that performance discrepancies were observed depending on the specific combination of parameters chosen. This variability occasionally led to instances where the LSTM model surpassed the Transformer model in effectiveness. Such observations underscore the impact of hyperparameter tuning and its role in the relative performance of these two architectures.

Employing the optimal set of hyperparameters identified through our rigorous tuning process, **the Transformer model exhibited a marginally superior performance compared to the LSTM model on the test dataset**. This comparative advantage is quantitatively illustrated in Table 4. The results highlight the nuanced differences in the efficacy of these models when calibrated with their respective best parameter configurations.

Model	LR	Epoch	Batch Size	Accuracy	Precision	Recall	F1-Score
LSTM	0.0001	3	32	0.6753	0.7023	0.6753	0.6857
	0.0005	3	32	0.7586	0.7094	0.7586	0.7327
	0.001	3	32	0.7684	0.7074	0.7684	0.7360
	0.005	3	32	0.7622	0.7083	0.7622	0.7338
	0.01	3	32	0.7655	0.7093	0.7655	0.7357
	0.001	3	16	0.7610	0.7086	0.7610	0.7333
	0.001	3	64	0.7700	0.7095	0.7700	0.7378
	0.001	**1**	**64**	**0.7849**	**0.7115**	**0.7849**	**0.7446**
	0.001	5	64	0.7669	0.7075	0.7669	0.7353
Transformer	0.0001	3	32	0.7971	0.7081	0.7971	0.7479
	0.0005	3	32	0.7710	0.7101	0.7710	0.7383
	0.001	3	32	0.7217	0.7095	0.7217	0.7143
	0.005	**3**	**32**	**0.8332**	**0.6942**	**0.8332**	**0.7574**
	0.01	3	32	0.8332	0.6942	0.8332	0.7574
	0.005	3	16	0.8332	0.6942	0.8332	0.7574
	0.005	3	64	0.8332	0.6942	0.8332	0.7574
	0.005	1	32	0.8332	0.6942	0.8332	0.7574
	0.005	5	32	0.8332	0.6942	0.8332	0.7574

Table 3: LSTM and transformer hyper-parameters grid search and performances on the valid set.

Model	LR	Epoch	Batch Size	Accuracy	Precision	Recall	F1-Score
LSTM	0.001	1	64	0.7761	0.7060	0.7761	0.7366
Transformer	0.005	3	32	**0.8262**	**0.6826**	**0.8262**	**0.7475**

Table 4: Results on the test set, using the best hyper-parameters for each model.

3.3 Detailed Analysis

3.3.1 LSTM Model

- **Learning Rate (LR):** The best LR for LSTM under our setting is 0.001. Larger LR like 0.005 would lead to performance decrease, indicating that the model might be over-fitting due to high LR.
- **Batch Size:** Larger sizes lead to more accurate gradient estimates but reduced generalization. Smaller sizes offer more frequent updates and greater generalization but may be less stable.
- **Epochs:** More epochs improve learning but can lead to over-fitting, affecting performance on validation data.

3.3.2 Transformer Model

We observed a trend that higher learning rates yielded more favorable outcomes. Specifically, the learning rate of 0.005 was identified as the most effective. This finding suggests that, in comparison to the LSTM model, the Transformer architecture benefits from larger weight updates to adequately fit the training data. This could be indicative of the Transformer's inherent characteristics and its responsiveness to more significant adjustments during the training phase. Our experimental results revealed a counterintuitive trend concerning the learning rate (LR) and model performance.

Besides, our experimental results revealed a counterintuitive trend concerning the learning rate (LR) and model performance. **As the LR was increased from** 0.0001 **to** 0.001**, we observed a consistent decrease in model performance. Intriguingly, a further increase in the LR to** 0.005 **led to the model achieving its best learning and performance metrics**. This phenomenon can be potentially explained through several points:

- **Learning Dynamics**: At lower LRs (from 0.0001 to 0.001), the model may be learning too slowly, potentially getting stuck in local minima or failing to converge within the given training epochs. This slow learning could account for the reduced performance.
- **Escaping Local Minima**: The improvement observed at a LR of 0.005 suggests that a higher LR helps the model escape local minima more effectively, allowing it to explore the parameter space more thoroughly and find a better global minimum.
- **Optimal Learning Window**: The specific architecture of the model under study might have an 'optimal learning window.' LRs lower than this window lead to underfitting, while those slightly higher facilitate rapid convergence to a more optimal solution.
- **Balance Between Exploration and Exploitation**: A higher LR might strike a better balance between exploration (searching new areas in the parameter space) and exploitation (refining the current areas), leading to improved model performance.

These explanations underscore the complex interplay between learning rate and model performance, emphasizing the necessity of empirical tuning for optimal model training.

In summary, the impact of hyperparameters on model performance is nuanced and varies based on architecture and dataset. Identifying the "sweet spot" for each hyperparameter is key, often achieved through grid search and performance analysis.

4 Conclusion

In this study, we have conducted a comprehensive investigation into the effectiveness of LSTM and Transformer models in the context of Named Entity Recognition (NER) tasks, using the CoNLL-2003 dataset. Our research was grounded in a detailed analysis of model architectures, hyperparameter tuning, and performance evaluation, leading to significant insights into the nuances of NER model performance.

Our findings indicate that while both LSTM and Transformer models have their unique strengths, the Transformer model generally exhibited superior performance in our experiments. This was particularly evident after an exhaustive grid search process on the validation set, where the Transformer model consistently outperformed the LSTM model under most parameter configurations. The detailed analysis of hyperparameters such as learning rate, batch size, and epochs provided us with a deeper understanding of how these factors influence model performance.

A key takeaway from our study is the importance of hyperparameter tuning in optimizing NER models. The Transformer model, with its optimal learning rate of 0.005, demonstrated how a higher learning rate can significantly impact model effectiveness, possibly by facilitating a better balance between exploration and exploitation in the learning process. In contrast, the LSTM model showed optimal performance at a lower learning rate of 0.001, highlighting the distinct learning dynamics and requirements of different architectures.

The results from our study have practical implications for the deployment of NER systems in real-world applications. The performance metrics obtained from both LSTM and Transformer models, particularly their precision, recall, and F1-scores, are indicative of their potential in accurately tagging entities in various NLP tasks. Our research underscores the necessity of careful model selection and hyperparameter optimization to achieve the best possible performance in NER tasks.

In conclusion, we have conduct a thorough comparative analysis of LSTM and Transformer models for NER. The insights gained from this research not only enhance our understanding of these models' capabilities but also offer valuable guidelines for practitioners in the field of NLP. Future research could explore the integration of these findings into more complex NER systems, potentially incorporating additional linguistic features or experimenting with different datasets to further validate and extend our conclusions.

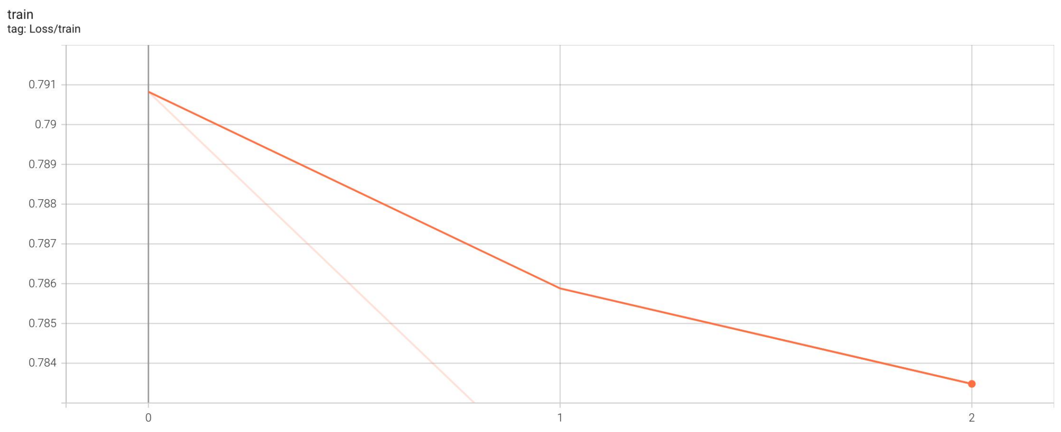

Figure 1: Loss-Epoch curve of Transformer model during training.

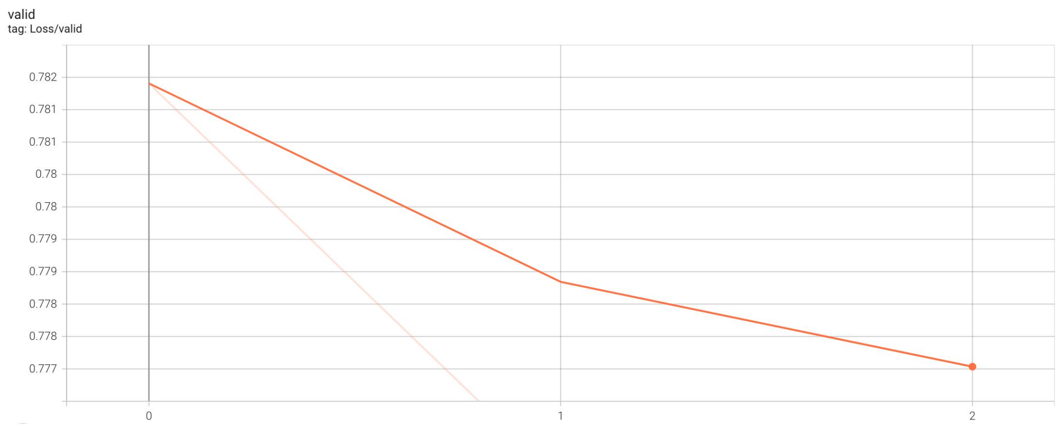

Figure 2: Loss-Epoch curve of Transformer model during validation.

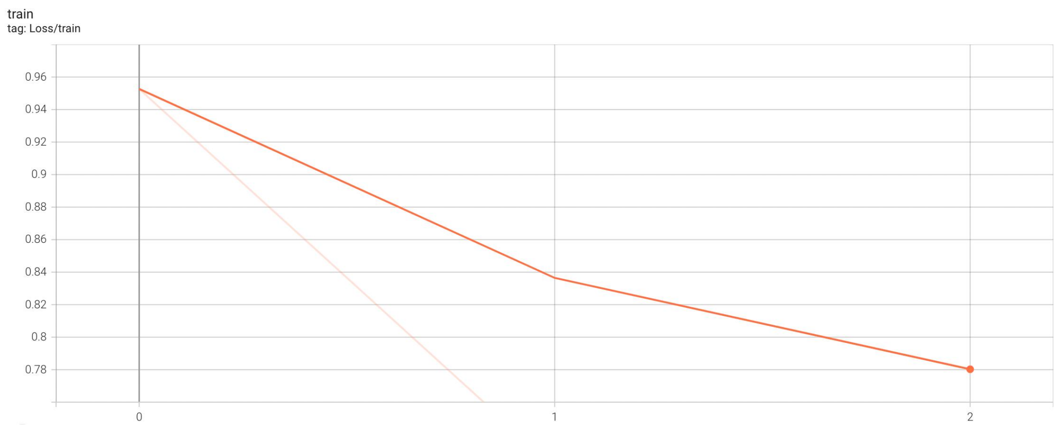

Figure 3: Loss-Epoch curve of LSTM model during training.

APPENDIX

A LOSS CURVE

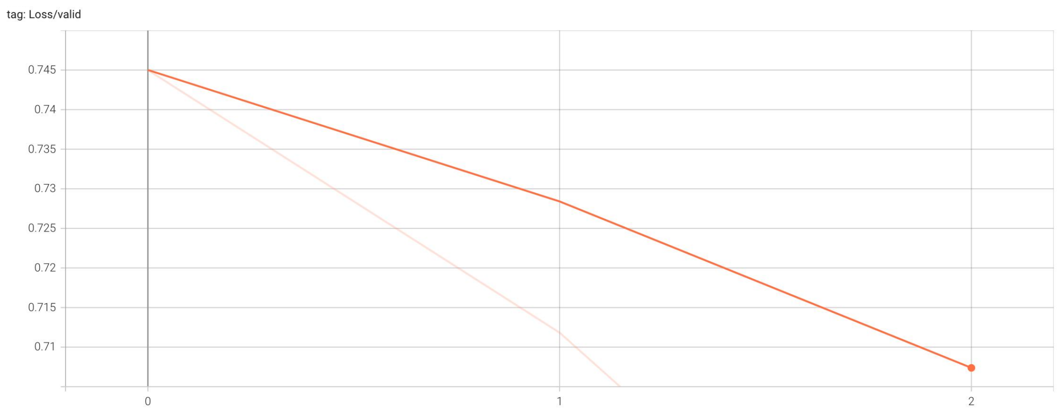

Figure 4: Loss-Epoch curve of LSTM model during validation.