BRAC UNIVERSITY

Inspiring Excellence

Project Report

CSE440 : Natural Language Processing II

Project Title : Token-Level Multi-Class Classification

Dataset: *NER Dataset-1*

Group No : 05 , CSE440 Lab Section : 03	
ID	**Name**
23341100	Waseque Chowdhury
22141018	Fuad Ibne Rafi
24241363	Afif Jaber Atanu

I) Abstract:

This project explores the application of various recurrent neural network architectures for Named Entity Recognition (NER) tasks. We implemented and evaluated four distinct models: a simple RNN, LSTM, GRU, and Bidirectional LSTM, using a sequential labeling approach with BIO tagging scheme. Our comparative analysis demonstrates that bidirectional architectures achieve superior performance in capturing contextual dependencies crucial for entity recognition. Preprocessing techniques including tokenization, sequence padding, and categorical encoding were employed to optimize model training. The experimental results show significant performance variations among the architectures, with BiLSTM achieving the highest F1 score. This research contributes to the ongoing efforts to improve NER systems through advanced deep learning techniques, highlighting the importance of architectural choices in sequential labeling tasks.

II) Introduction:

Named Entity Recognition (NER) is a core natural language processing task that identifies and classifies entities like people, organizations, and locations in text. This capability is vital for information retrieval, question answering, summarization, and knowledge graph construction. While traditional NER approaches relied on domain-specific features, modern deep learning techniques, particularly recurrent neural networks have enabled more adaptable and robust systems.

This project addresses the need to systematically compare different neural architectures for NER by evaluating four distinct recurrent network models under consistent conditions. The project employs the BIO (Beginning, Inside, Outside) tagging scheme for precise entity boundary detection, framing NER as a token-level classification problem that leverages sequential learning to capture word-context relationships. The work provides insights into how architectural choices affect NER performance, revealing trade-offs between model complexity, computational efficiency, and accuracy.

III) Methodology:

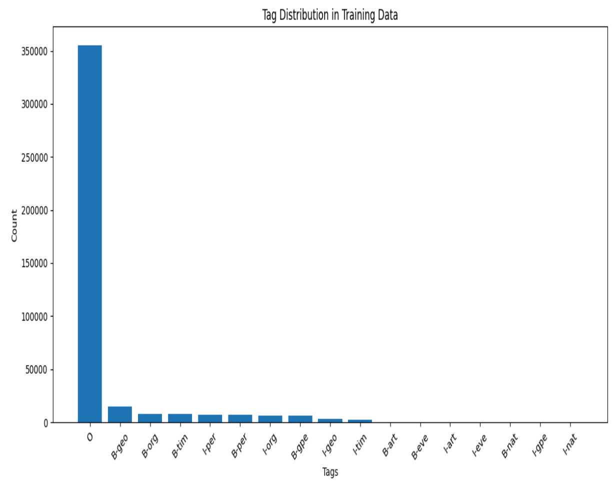
1)Data Exploration and Analysis:

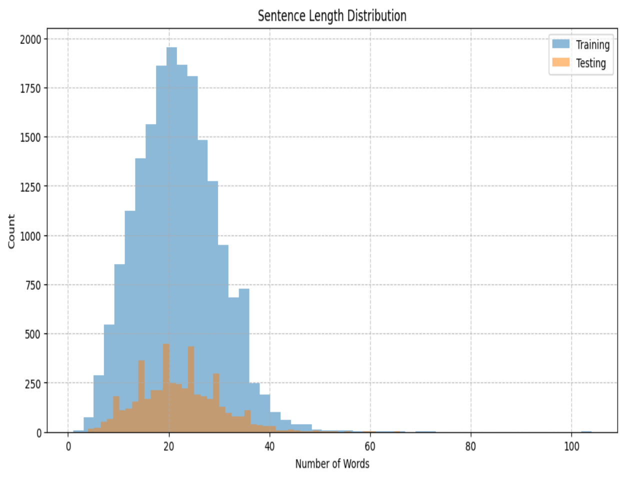
The project utilized NER-1 dataset divided into training and testing sets, consisting of sentences with corresponding entity tags. The exploratory data analysis revealed:

- The training set contained a significant number of sentences, each annotated with token-level tags

- The distribution of tags followed a typical pattern for NER tasks, with the majority of tokens labeled as 'O' (Outside) and entity tags accounting for a smaller proportion
- Several entity types were identified in the dataset, marked using the BIO tagging scheme
- Sentence lengths varied considerably, with an average of approximately 20-30 tokens per sentence
- The vocabulary size was substantial, with many low-frequency words appearing only once in the corpus

To better understand the dataset characteristics, we analyzed tag distribution and visualized sentence length distribution across both training and testing sets. This analysis informed decisions about model architecture and preprocessing techniques.





2)Methodology:

Step	Description
Data Cleaning	Extracted sentences and corresponding tags from dataset Verified alignment between tokens and tags- Identified and handled rows with mismatched lengths
Tokenization and Vocabulary Creation	Built vocabulary of top 10,000 most frequent words- Converted words to lowercase- Introduced <OOV> token for out-of-vocabulary words

Sequence Processing	Converted sentences to sequences of numerical indices- Created mapping between tags and numerical indices Determined maximum sequence length based on longest sentence
Padding and Masking	Padded sequences to uniform length for batch processing Created mask to distinguish actual tokens from padding during loss calculation Ensured padding tokens did not contribute to model training
Label Encoding	Encoded tags as one-hot vectors Prepared target variables for multi-class classification training

<u>3)Model Architecture:</u>

Four recurrent neural network architectures were implemented for the NER task, each following a similar structure: an embedding layer (100 dimensions), dropout layers (rate 0.2), a recurrent layer with 100 units (returning sequences), and a time-distributed dense layer with softmax activation. The models differed in the type of recurrent layer used: Simple RNN, LSTM, GRU, and Bidirectional LSTM.

<u>4)Training Process:</u>

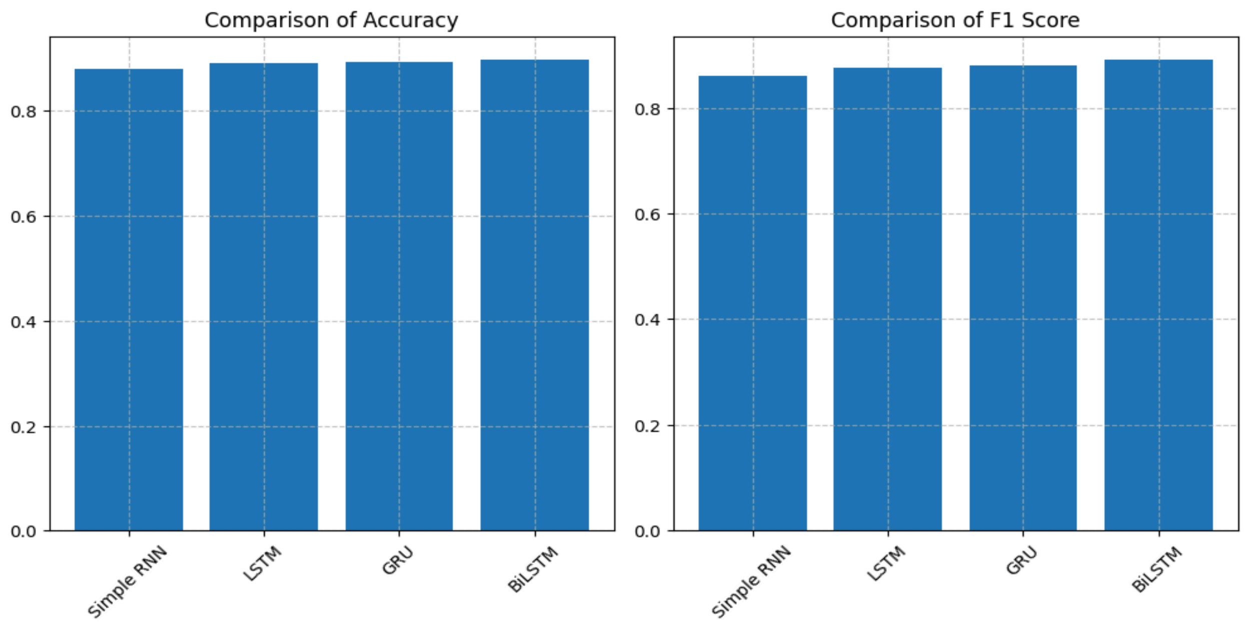
All models shared key design elements, including word embeddings, dropout for regularization, a time-distributed dense layer for token-level classification, categorical cross-entropy loss, the Adam optimizer, and callbacks like early stopping and model checkpointing. Training was conducted with a batch size of 32, up to 10 epochs (with early stopping), a 10% validation split, and monitoring of validation loss to save the best model.

We also used hyper parameter tuning where it defines a parameter grid exploring variations in batch size (fixed at 64), learning rate (0.005 and 0.0001), RNN units (fixed at 128), embedding dimensions (fixed at 128), and dropout rate (fixed at 0.3). For each parameter combination, a model is constructed, trained for up to 15 epochs with early stopping and learning rate reduction callbacks, and evaluated on test data. As due to a lack of full GPU runtime we could not test all possible variations in tuning with the dataset.

<u>IV) Results:</u>

The experimental evaluation revealed distinct performance patterns across the four implemented architectures. The BiLSTM model demonstrated superior performance, achieving the highest accuracy (89.68%) and F1 score (89.04%) among all models. This was followed by the GRU model (89.28% accuracy, 88.04% F1), then the LSTM model (89.13% accuracy, 87.52% F1), with the Simple RNN model showing the lowest performance (87.90% accuracy, 85.99% F1).

A notable observation was the significant difference in prediction time, with the BiLSTM model being the most efficient (3.33 seconds) despite its architectural complexity, while the Simple RNN model was substantially slower (17.71 seconds) despite its simpler structure.



Examining entity-specific metrics revealed important insights:

1. All models performed well on the 'O' (Outside) tag with precision and recall exceeding 90%.
2. Common entity types like persons (PER), locations (GEO), and organizations (ORG) showed moderate performance with F1 scores ranging from 0.30 to 0.68.
3. Rare entity types (ART, NAT, EVE) showed poor recognition across all models, with most achieving 0% F1 scores.
4. The BiLSTM architecture consistently demonstrated better recall for entity types, suggesting superior contextual understanding.

The performance disparity between common and rare entities highlights the challenge of class imbalance in NER tasks. Despite overall accuracy exceeding 87% across models, the macro-average F1 scores remained below 40%, indicating substantial room for improvement in minority class recognition.

V) Conclusion:

This project demonstrates the effectiveness of bidirectional architectures for Named Entity Recognition tasks, with the BiLSTM model showing superior performance in both accuracy and computational efficiency. The experimental results align with theoretical expectations, as bidirectional processing allows the model to leverage both past and future contextual information crucial for entity boundary detection.

Key findings and insights include:

1. Architectural complexity does not necessarily correlate with inference time, as demonstrated by the BiLSTM's superior speed despite its more complex structure.
2. The severe class imbalance in NER datasets poses a significant challenge, with rare entity types being particularly difficult to recognize.
3. While overall accuracy appears high, entity-specific metrics reveal substantial room for improvement in entity recognition capabilities.

Limitations of this project include:

- Limited model training duration (10/15 epochs maximum)
- Basic embedding approach without pre-trained embeddings
- Uniform architecture parameters across models (100 units)
- Limited GPU access

Future improvements could include:

- Incorporating pre-trained word embeddings (e.g., GloVe, BERT)
- Implementing attention mechanisms to enhance contextual understanding
- Exploring data augmentation techniques for rare entity types
- Experimenting with class weighting to address imbalance issues
- Implementing CRF (Conditional Random Fields) as the output layer

This comparative analysis provides valuable insights for practitioners selecting neural architectures for NER tasks, highlighting the trade-offs between different RNN variants and emphasizing the importance of considering both performance metrics and computational efficiency in model selection.

VI) References:

[1] NumPy Contributors, "NumPy," NumPy, 2025. [Online]. Available: https://numpy.org/

[2] Pandas Contributors, "pandas - Python Data Analysis Library," 2025. [Online]. Available: https://pandas.pydata.org/

[3] The Matplotlib Development Team, "Matplotlib: Visualization with Python," 2025. [Online]. Available: https://matplotlib.org/

[4] The Seaborn Development Team, "Seaborn: statistical data visualization," 2025. [Online]. Available: https://seaborn.pydata.org/

[5] The scikit-learn developers, "scikit-learn: Machine Learning in Python," 2025. [Online]. Available: https://scikit-learn.org/

[6] Martín Abadi et al., "TensorFlow: Large-scale machine learning on heterogeneous systems," 2025. [Online]. Available: https://www.tensorflow.org/

[7] The Keras Team, "Keras," 2025. [Online]. Available: https://keras.io/