

**CSE440: Natural Language Processing II**

**Project Report**

**Project Title: Comparative Analysis of RNN Variants for POS Tagging**

| **Group No: 08, CSE440 Section: 03, Spring 2025** | |
| --- | --- |
| **ID** | **Name** |
| 22201604 | Priyankan Biswas |
| 22241021 | K. M. Jubair Sami |
| 22299520 | Dipto Sumit |

**Abstract**

This study explores the task of Part-of-Speech (POS) tagging using different recurrent neural network (RNN) architectures. The models implemented include Simple RNN, LSTM, GRU, and BiLSTM. We performed exploratory data analysis on the dataset, preprocessed the input sequences, and designed consistent neural architectures for fair comparison.The data analysis was done in two phases: one without using word2vec and another with word2vec. Evaluation was based on macro F1-score. Results show that BiLSTM outperforms other variants in most evaluations, indicating the effectiveness of bidirectional context in sequence tagging tasks.

**1. Introduction**

POS tagging is a foundational problem in natural language processing (NLP), where each word in a sentence is labeled with its appropriate part of speech. In this project, we will assess how different RNN variants perform on the POS tagging task. The goal is to analyze which architecture provides the best trade-off between accuracy and model complexity.

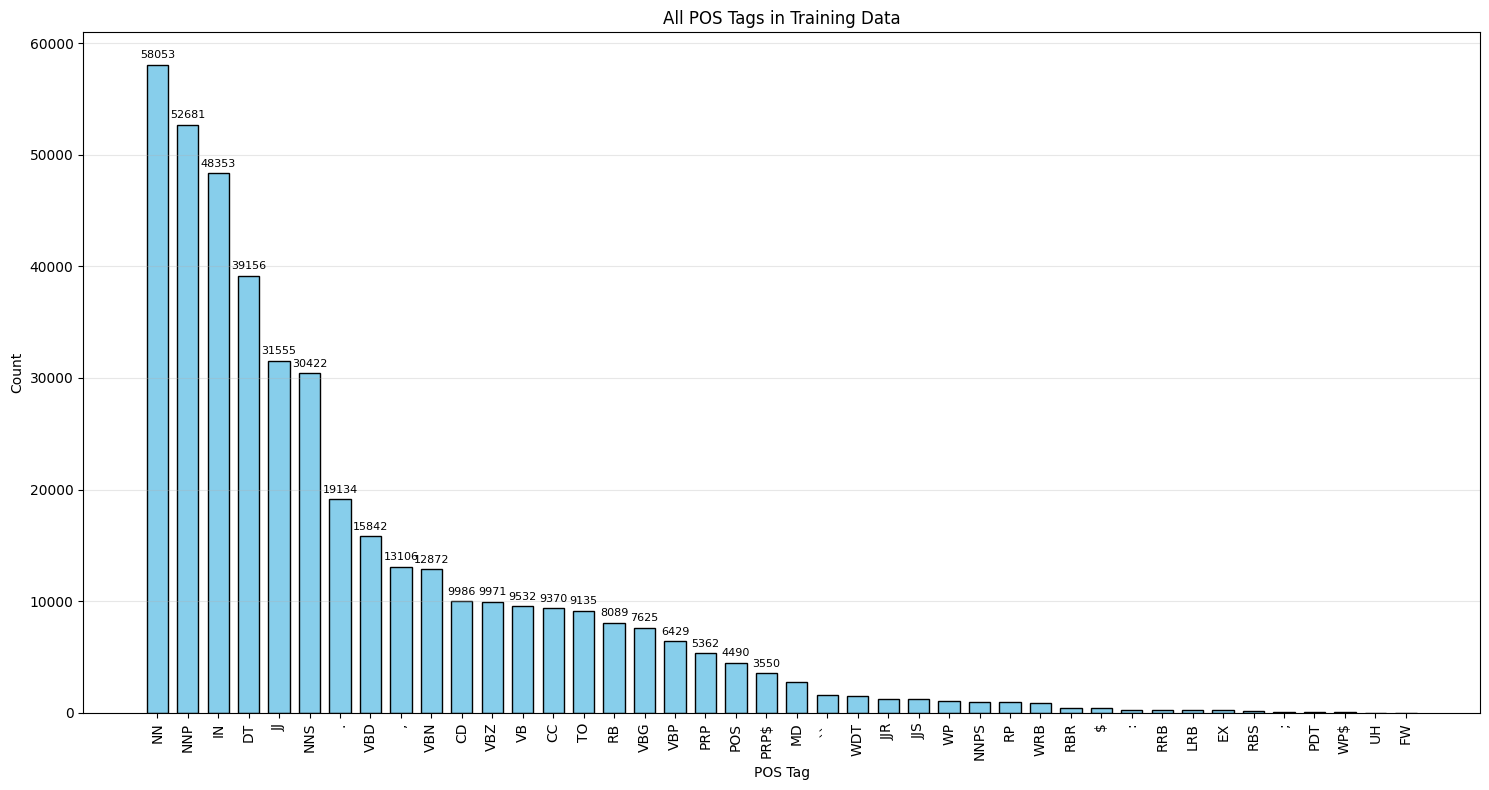
**2. Methodology**

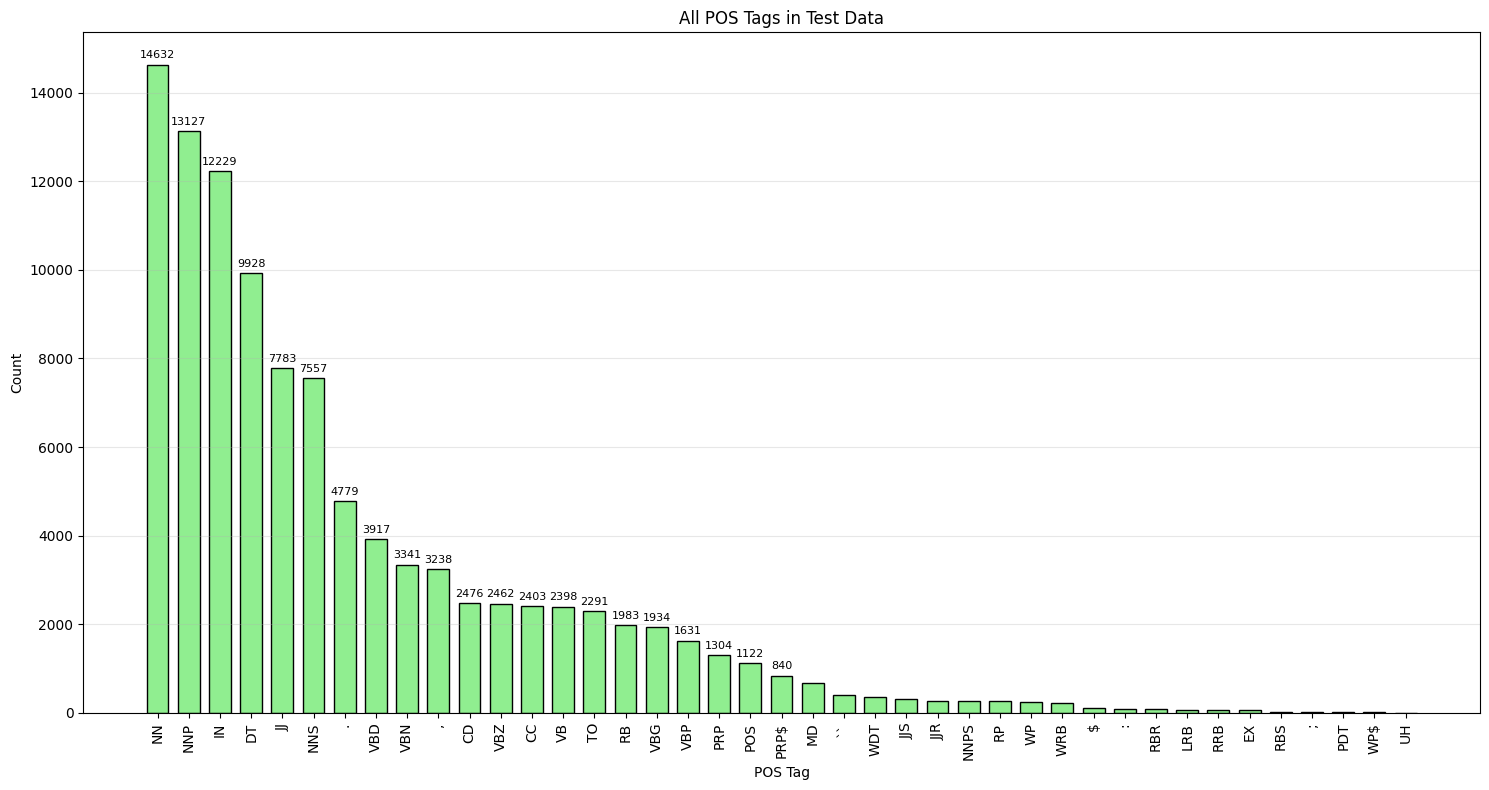
**2.1 Exploratory Data Analysis (EDA):**

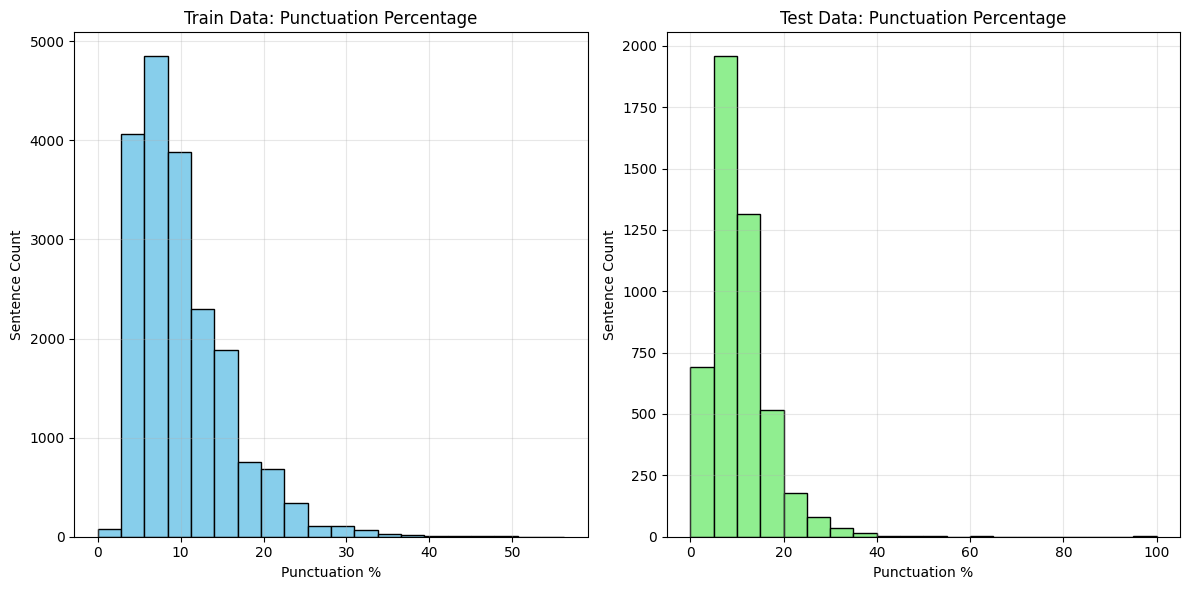
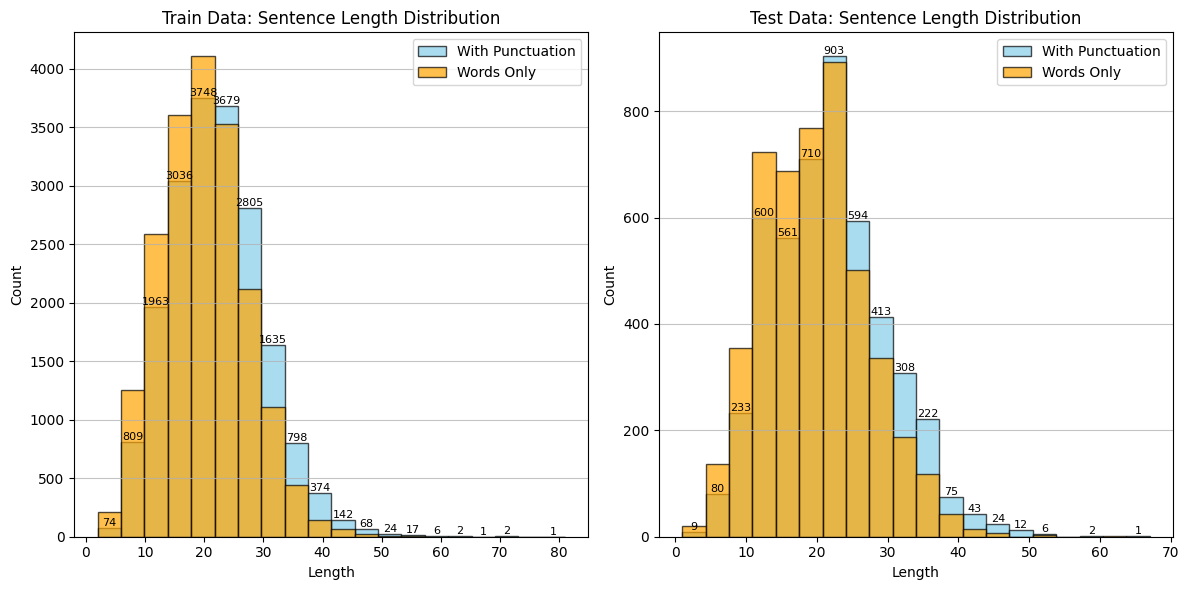
The dataset comprises sentences with tokens labeled by their corresponding POS tags. It was split 80:20 into training and testing sets. Key observations include:

* The train dataset contains 19184 sentences. On the other hand, the test dataset contains only 4796 sentences.
* Sentence lengths varied widely, typically ranging from 10–30 words, necessitating padding.
* The longest sentence in the train dataset consists of 81 tokens(72 words). For the test dataset, it was 67 tokens(62 words).
* The dataset contains 42 unique POS tags in train dataset and 41 in test dataset.
* For both train and test dataset, 'NN' (noun) and 'NNP' (proper noun, singular) were the most frequent among POS tags.
* Average punctuation percentage in both train and test dataset is more than 10%.

The distribution of sentences’ length, POS tags and punctuations are visualised through histograms.







**2.2 Preprocessing :**

* **Tokenization and vocabulary creation for both words and tags.**
* **Each word/tag is indexed; unknown words are mapped to UNK and sequences are padded to equal length.**
* **The training data was further divided for training (80%) and validation (20%).**
* **One-hot encoding is used for target sequences to train classification models.**
* **Word2vec was used to encode words into vector embeddings in the second phase.**

**2.3 Model Architectures**

We experimented using different model architectures and pipelines in different cases to find out the best performing pipelines. Some of the model’s architectures are:

* Embedding Layer (case-1): Using key value pairs of POS tags using pythons dictionary.
* Embedding Layer (case-2,3): Transforms words into dense vectors using Word2vec.
* RNN Variants (case-1,2): Simple RNN, LSTM, GRU, or Bidirectional LSTM (with 1 hidden layer).
* RNN Variants (case-3): Simple RNN, LSTM, GRU, or Bidirectional LSTM (with 2 hidden layers).
* Time Distributed Dense Layer: Applies classification across all time steps.
* Each model is compiled with **categorical Cross-entropy loss** and the **Adam** optimizer and for activation function **Softmax** is used.

**4. Training**

* **Hyperparameters were interchanged during every test to obtain the optimum macro F1 score.**
* **Optimizer: Adam, loss: categorical cross-entropy.**
* **Training conducted with a validation split to monitor overfitting.**

**Results**

Each model was evaluated on the test set using:

* **Accuracy**
* **Macro F1-score**
* **Confusion Matrix**

**Case-1:**

**After preprocessing without word2vec (one hidden layer):**

| RNN variants | Best hyperparameters (comparing macro avg of f1 score) | F1 score(Macro avg.) | Accuracy |
| --- | --- | --- | --- |
| Simple RNN | epoch:20 | .954 | .985 |
| Batch size:64 |
| units:64 |
| lr: .01 |
| Dropout: .2 |
| LSTM | epoch:20 | .946 | .984 |
| Batch size:64 |
| units:64 |
| lr: .01 |
| Dropout: .2 |
| GRU | Epoch: 20 | .975 | .984 |
| Batch size: 64 |
| Units: 64 |
| lr: .01 |
| Dropout: .2 |
| BILSTM | epoch:20 | .963 | .989 |
| Batch size: 64 |
| units: 64 |
| lr: .01 |
| Dropout: .2 |

**Case-2:**

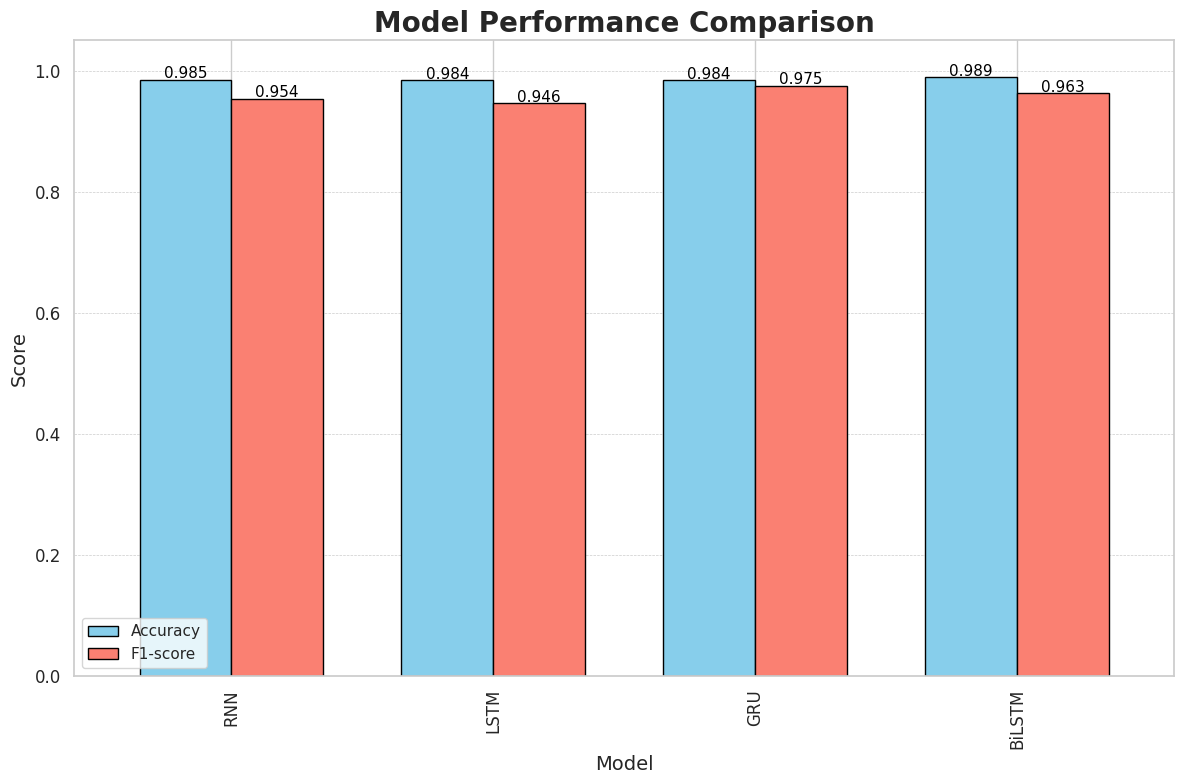
**After preprocessing using word2vec (one hidden layer):**

| RNN variants | Best Hyperparameters  (comparing macro avg of f1 score) | F1 score(Macro avg.) | Accuracy |
| --- | --- | --- | --- |
| Simple RNN | epoch:50 | .816 | .841 |
| Batch size:32 |
| units:64 |
| lr: .001 |
| Dropout: 0.2 |
| LSTM | epoch:50 | .787 | .841 |
| Batch size:32 |
| units:64 |
| lr: .001 |
| Dropout: 0.2 |
| GRU | epoch:50 | .788 | .840 |
| Batch size:32 |
| units:64 |
| lr: .001 |
| Dropout: 0.2 |
| BILSTM | epoch:50 | .844 | .877 |
| Batch size:32 |
| units:64 |
| lr: .001 |
| Dropout: 0.2 |

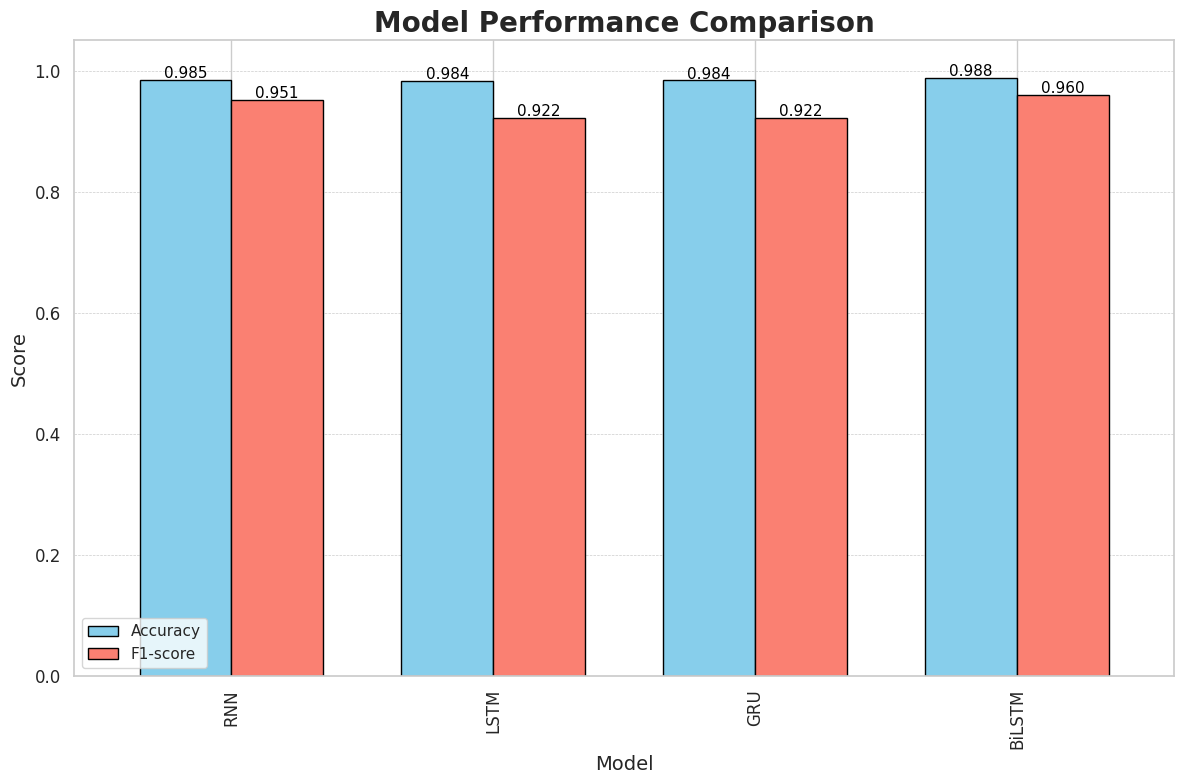
**However, we also ran a model with 10 epochs using two hidden layers(Case-3) of all the RNN variants, but the performance decreased compared to the single hidden layer trained with the same number of epochs.Thats why we disregarded the results.**

**Sample of F1 score and Accuracy comparison between each variant of RNN(without Word2Vec and with macro average):**

**20 Epochs:**

****

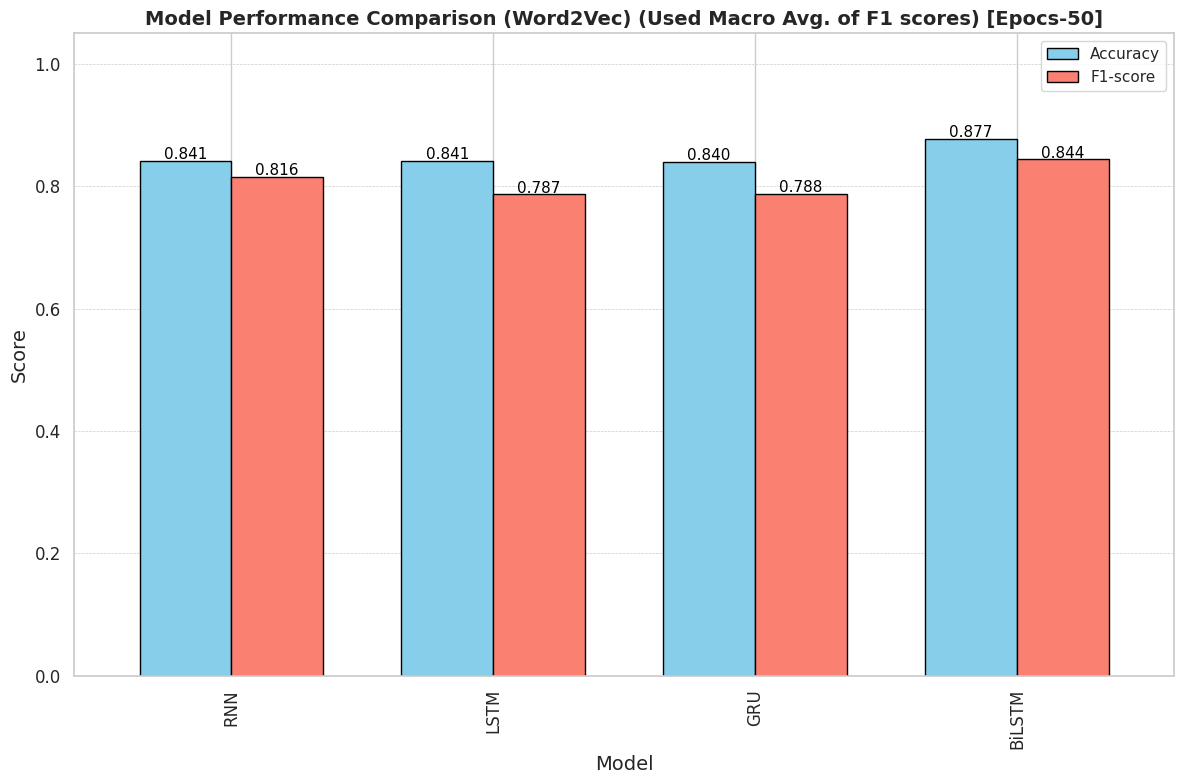
**30 Epochs:**

****

**Analysis of the model trained without Word2Vec embeddings:**

This model(without using Word2Vec for embeddings) was trained and tested four times for all four variants of RNN using different hyperparameters. A steady growth was observed up to 20 epochs. But, the macro F1 score declined when it was evaluated after training for 30 epochs. Therefore, it can be stated that overfitting occurred in the last case. Moreover, BiLSTM did not achieve the best macro F1 score for every test. In some tests, GRU outperformed BiLSTM.

**Sample of F1 score and Accuracy comparison between each variant of RNN(with Word2Vec and macro avg.):**

****

**Analysis of the model trained with Word2Vec embeddings:**

* **BiLSTM** achieved the highest macro F1 score.
* **LSTM and GRU** performed similarly.
* **RNN** performed inconsistently during every evaluation.

Visualization of confusion matrices further illustrated the superiority of BiLSTM. When examining using Word2vec, almost all the RNN variants performed better while increasing the number of epochs.

**Conclusion**

The analysis revealed that BiLSTM consistently outperformed other models in POS tagging across all phases and modifications. When word2vec was not used, we got the best F1 score for each variant of RNN using 20 epochs. After that, a case of overfitting was observed for 30 epochs. On the other hand, using Word2Vec embeddings gave an upward trajectory of F1 score with the increasing number of epochs. 50 epochs gave us the best outcome for every variant of RNN when word2vec is used. Moreover, the macro F1 score was used for comparisons as it showed better handling of class imbalance than weighted averages. The main limitation we faced during this analysis was the imbalanced POS tag distribution and the availability of time to run with even more epochs. If sufficient time was available, the best possible outcome for the phase of training with word2vec embedding vectors could have been achieved by increasing the number of epochs even more. If we had been allowed to incorporate pre-trained embeddings like BERT, the best possible outcome could have been attained easily.

**References**  
 [1] F. Chollet, *Keras Documentation*, [https://keras.io](https://keras.io/) [2] Scikit-learn Developers, *scikit-learn: Machine Learning in Python*, [https://scikit-learn.org](https://scikit-learn.org/) [3] Matplotlib, Seaborn – Python visualization libraries used for plots.