REPORT

FEED FORWARD NEURAL NETWORK(FFNN)

Hyper Parameters used to train the model

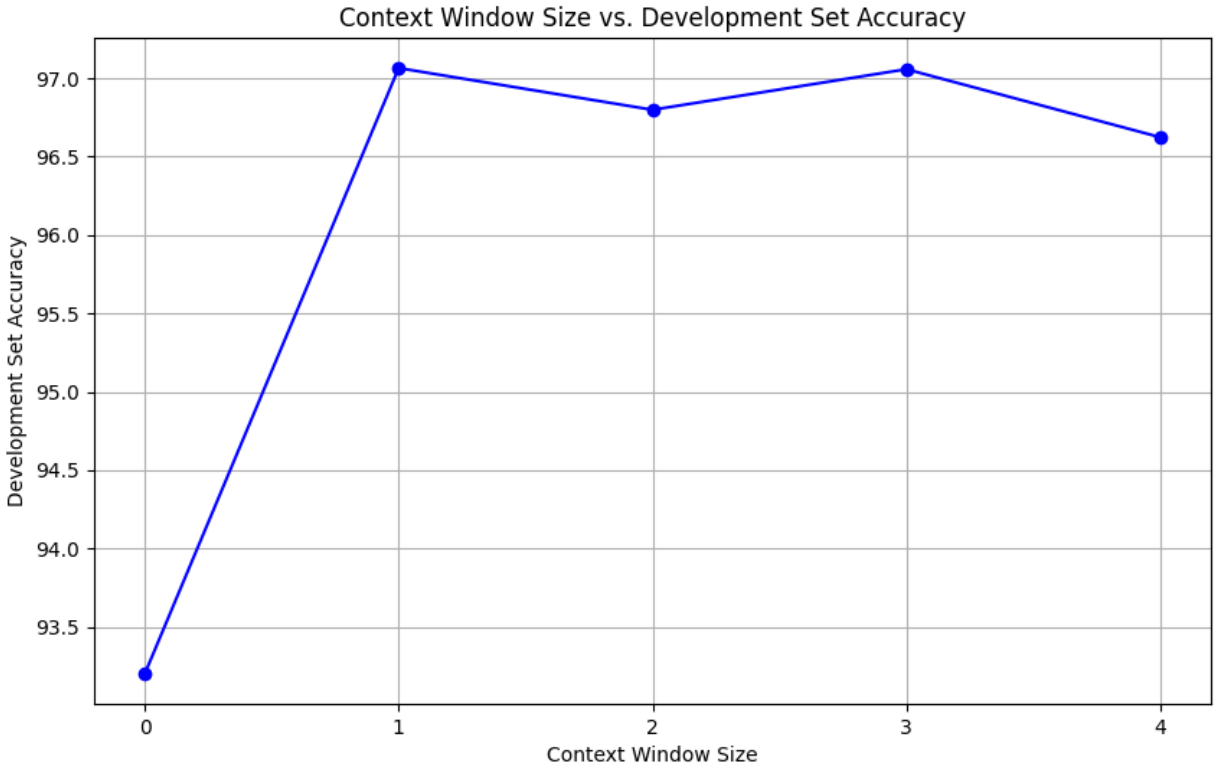
```
models = []
configs = [
    {'hidden_dims': [128], 'activation_fn': nn.ReLU},
    {'hidden_dims': [256, 128], 'activation_fn': nn.LeakyReLU},
    {'hidden_dims': [128, 128, 128], 'activation_fn': nn.Tanh},
]
```

- The model was tuned using the following configurations:
  - A single hidden layer with 128 neurons and a ReLU activation function.
  - Two hidden layers with 256 and 128 neurons, respectively, using a LeakyReLU activation function.
  - Three hidden layers, each with 128 neurons, using a Tanh activation function.
- The number of epochs was set to 5, as this was sufficient to achieve a decent accuracy of around 95% consistently.
- The default settings for context window size were:
  - Preceding window (p): 2
  - Succeeding window (s): 3

Best Accurate Model

- In the above image , i have displayed , results of each configuration and the results achieved.
- out of all , the best result was achieved when i used Tanh
- . The best dev set accuracy is around 96 percent.

Graph and evaluation metrics for FFNN



- The graph provides valuable insights into how the model interacts with the data.
- It suggests that there is a balance to be struck between providing enough context for the model to learn and providing so much context that it becomes counterproductive.

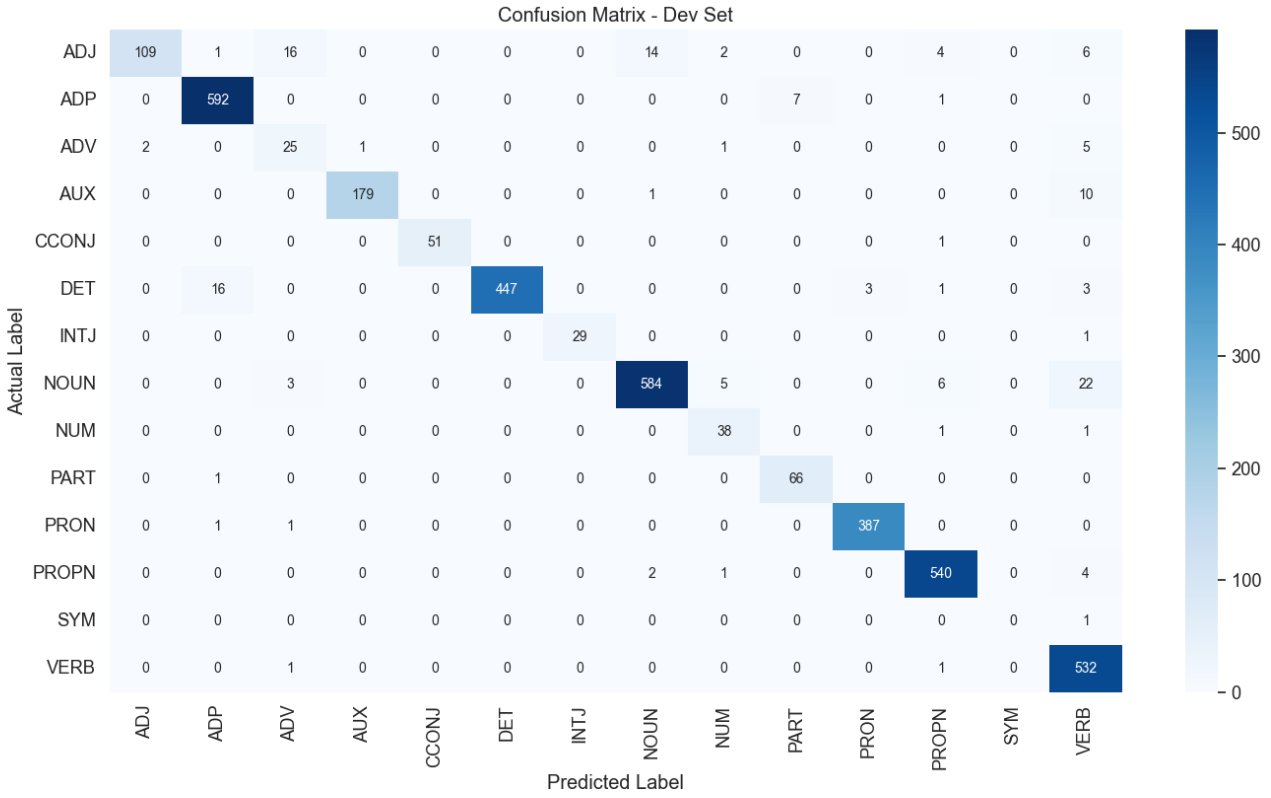
Initial Increase:

- The graph shows a sharp increase in development set accuracy as the context window size increases from 0 to 1.
- This indicates that including even a single word of context (either preceding or succeeding the target word) significantly improves the model's ability to predict the correct POS tag.

Peak Performance:

- The accuracy peaks when the context window size is 1.
- This suggests that providing one word of context on each side of the target word gives the model sufficient information to achieve the best performance on the development set.
- This statement also supports the fact that the context is mainly dependent on nearby words.

Confusion metrics for DEV_SET



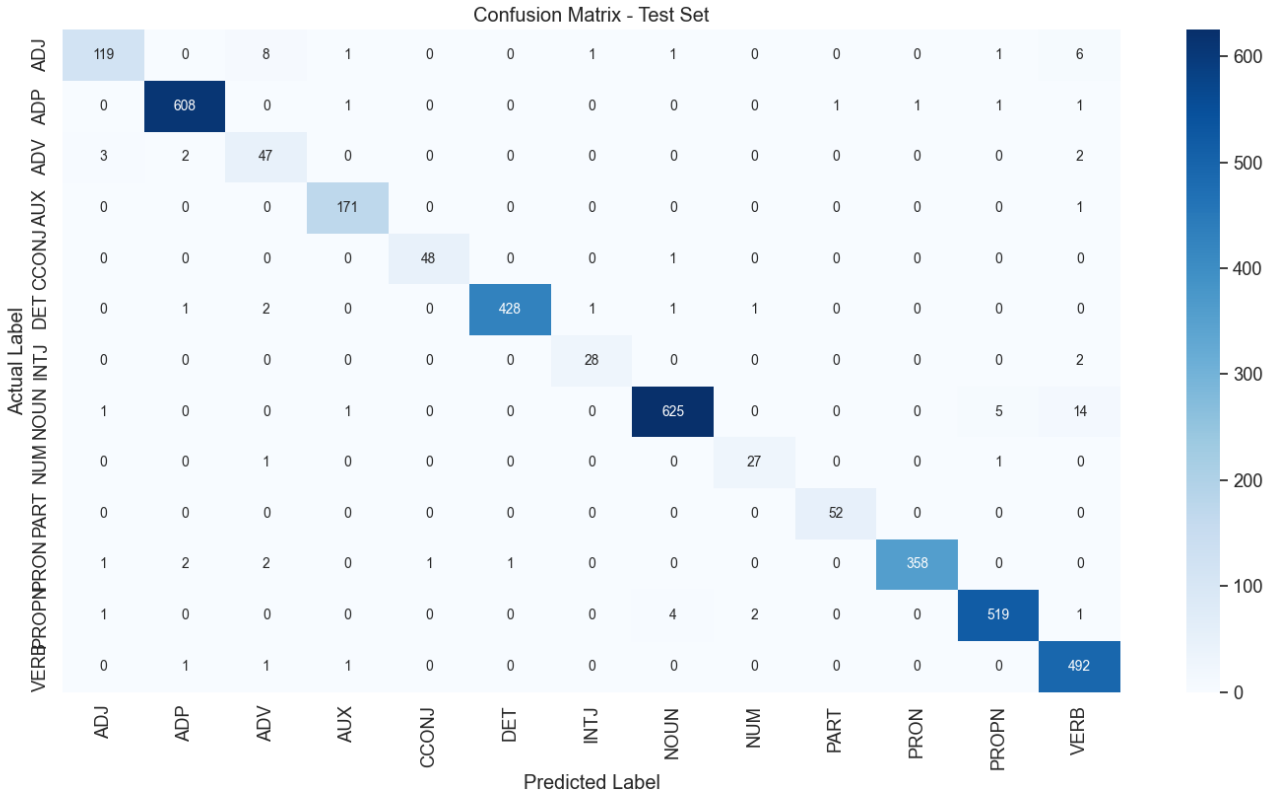
Diagonal Dominance:

- A well-performing model will have high numbers along the diagonal, representing correct classifications.
- In the matrix, high values on the diagonal indicate a large number of true positives for most POS tags, such as ADJ (Adjective), ADP (Ad position), NOUN, and VERB.

Off-Diagonal Elements:

- Elements off the diagonal represent misclassifications.
- For instance, there are instances where DET (Determiner) was misclassified as ADJ, and ADJ as NOUN.
- These misclassifications suggest areas where the model which i developed might be confused, potentially due to similarities in how these POS tags are used in context.

Confusion Metrics for Test_ set



Model Accuracy:

- The high concentration of values along the diagonal in the confusion matrices indicates that the model generally exhibits a high degree of accuracy across both datasets.
- This is evidenced by the substantial true positive rates for tags such as 'ADP' and 'NOUN'.

Misclassification Patterns:

- The matrices reveal certain patterns of misclassification, where some POS tags are consistently confused with others.
- This suggests that the model may benefit from further tuning to better distinguish between these tags.

Performance on Rare Tags:

- The performance on less frequent tags such as 'AUX' and 'SYM' appears to be less reliable, which likely results from a paucity of representative examples in the training phase.

TEST SET - Accuracy

```
(myenv) PS C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2> python postagger.py -f
Enter a sentence: an apple a day keeps the doctor away
an DET
apple ADJ
a DET
day NOUN
keeps VERB
the DET
doctor VERB
away ADP
Configuration: {'hidden_dims': [128], 'activation_fn': <class 'torch.nn.modules.activation.ReLU'>}, Dev Accuracy: 95.81320450885669
Configuration: {'hidden_dims': [256, 128], 'activation_fn': <class 'torch.nn.modules.activation.LeakyReLU'>}, Dev Accuracy: 95.92055823939883
Configuration: {'hidden_dims': [128, 128, 128], 'activation_fn': <class 'torch.nn.modules.activation.Tanh'>}, Dev Accuracy: 96.05475040257649
Best Configuration: {'hidden_dims': [128, 128, 128], 'activation_fn': <class 'torch.nn.modules.activation.Tanh'>}
Best Dev Set Metrics: {'accuracy': 96.05475040257649}
Test Accuracy: 97.75%
##################################################
 THE  TEST SET EVALUATION MATRICES
Accuracy: 97.751873438801%
Macro Recall: 96.01708479158066%
Macro F1: 95.65564161847622%
Micro Recall: 97.751873438801%
Micro F1: 97.751873438801%
##################################################
 THE  DEV SET EVALUATION MATRICES
Accuracy: 96.05475040257649%
Macro Recall: 86.67855262220087%
Macro F1: 86.03506933009048%
Micro Recall: 96.05475040257649%
Micro F1: 96.05475040257649%
##################################################
(myenv) PS C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2>
```

- I have added the above screenshots which displays the Test_Set accuracy of around 97 percent which is considerably good.

TEST SET AND DEV SET EVALUATION METRICS

```
(myenv) PS C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2> python postagger.py -f
Enter a sentence: an apple a day keeps the doctor away
an DET
apple ADJ
a DET
day NOUN
keeps VERB
the DET
doctor VERB
away ADP
Configuration: {'hidden_dims': [128], 'activation_fn': <class 'torch.nn.modules.activation.ReLU'>}, Dev Accuracy: 95.81320450885669
Configuration: {'hidden_dims': [256, 128], 'activation_fn': <class 'torch.nn.modules.activation.LeakyReLU'>}, Dev Accuracy: 95.92055823939883
Configuration: {'hidden_dims': [128, 128, 128], 'activation_fn': <class 'torch.nn.modules.activation.Tanh'>}, Dev Accuracy: 96.05475040257649
Best Configuration: {'hidden_dims': [128, 128, 128], 'activation_fn': <class 'torch.nn.modules.activation.Tanh'>}
Best Dev Set Metrics: {'accuracy': 96.05475040257649}
Test Accuracy: 97.75%
##################################################
 THE  TEST SET EVALUATION MATRICES
Accuracy: 97.751873438801%
Macro Recall: 96.01708479158066%
Macro F1: 95.6556416184762 2%
Micro Recall: 97.7518734388 01%
Micro F1: 97.751873438801%
##################################################
 THE  DEV SET EVALUATION MATRICES
Accuracy: 96.05475040257649%
Macro Recall: 86.67855262220087%
Macro F1: 86.03506933009048%
Micro Recall: 96.05475040257649%
Micro F1: 96.05475040257649%
##################################################
(myenv) PS C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2>
```

Test Set Evaluation Metrics:

- The test accuracy is extremely high at 97.75%. This indicates that the model is able to correctly predict the POS tags for unseen data with a high level of reliability.

- Macro Recall and F1 scores are both above 96%, which suggests that the model is consistent across different POS tags, not just the more frequently occurring ones. Macro scores are particularly important in datasets where there is class imbalance, as they give equal weight to each class.
- Micro Recall and F1 scores are identical to the accuracy in this case, which is typical for single-label classification tasks. These scores being above 97% further confirm the model's strong performance.

Development Set Evaluation Metrics:

- The accuracy on the development set is slightly lower than the test set at 96.05%. This is still a very strong performance and suggests that the model generalizes well, but it may indicate some overfitting to the training data since the test set performance is higher.
- The Macro Recall of approximately 86.68% and Macro F1 score of 86.04% are considerably lower than their test set counterparts. This discrepancy could be due to the model performing well on tags with a large number of examples but not as well on rarer tags, which would lower the macro scores more than the micro scores.
- Micro Recall and F1 scores are high, matching the accuracy, which indicates consistent performance on individual instances across the development set.

FINAL CONCLUSION ON FFNN

- The model is very effective in predicting POS tags, as evidenced by the high scores across both datasets.
- Macro scores are more sensitive to performance on infrequent classes. The lower macro scores on the development set indicate that there may be some POS tags that the model is not predicting as well(SYM).
- Given the high scores, the model is likely capturing the majority of the patterns in the data.

RECURRENT NEURAL NETWORKS(LSTM)

Hyper Parameters used to train the model

```
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
hyperparameters = [
    {'num_layers': 1, 'hidden_dim': 128, 'bidirectional': True, 'activation_fn': nn.ReLU()},
    {'num_layers': 2, 'hidden_dim': 256, 'bidirectional': False, 'activation_fn': nn.Tanh()},
    {'num_layers': 2, 'hidden_dim': 128, 'bidirectional': True, 'activation_fn': nn.LeakyReLU()}
]
```

- **Number of layers**: The depth of the neural network. Different configurations include both single-layer and multi-layer networks.
- **Hidden layer dimension**: The number of units within a hidden layer. Configurations have varied this dimension, testing with 128 and 256 units.
- **Bidirectionality**: A binary setting indicating whether the neural network processes data in both forward and backward directions, which is especially useful for understanding context in sequences.
- **Activation function**: The non-linear function applied to the output of a layer. The functions used in tuning include ReLU, LeakyReLU, and Tanh, each having distinct characteristics that affect the learning process.
- I have used the above things as my hyperparameters and tested in 3 different configurations.
- The number of epochs was set to 5, as this was sufficient to achieve a decent accuracy of around 95% consistently.

Best Accurate Model

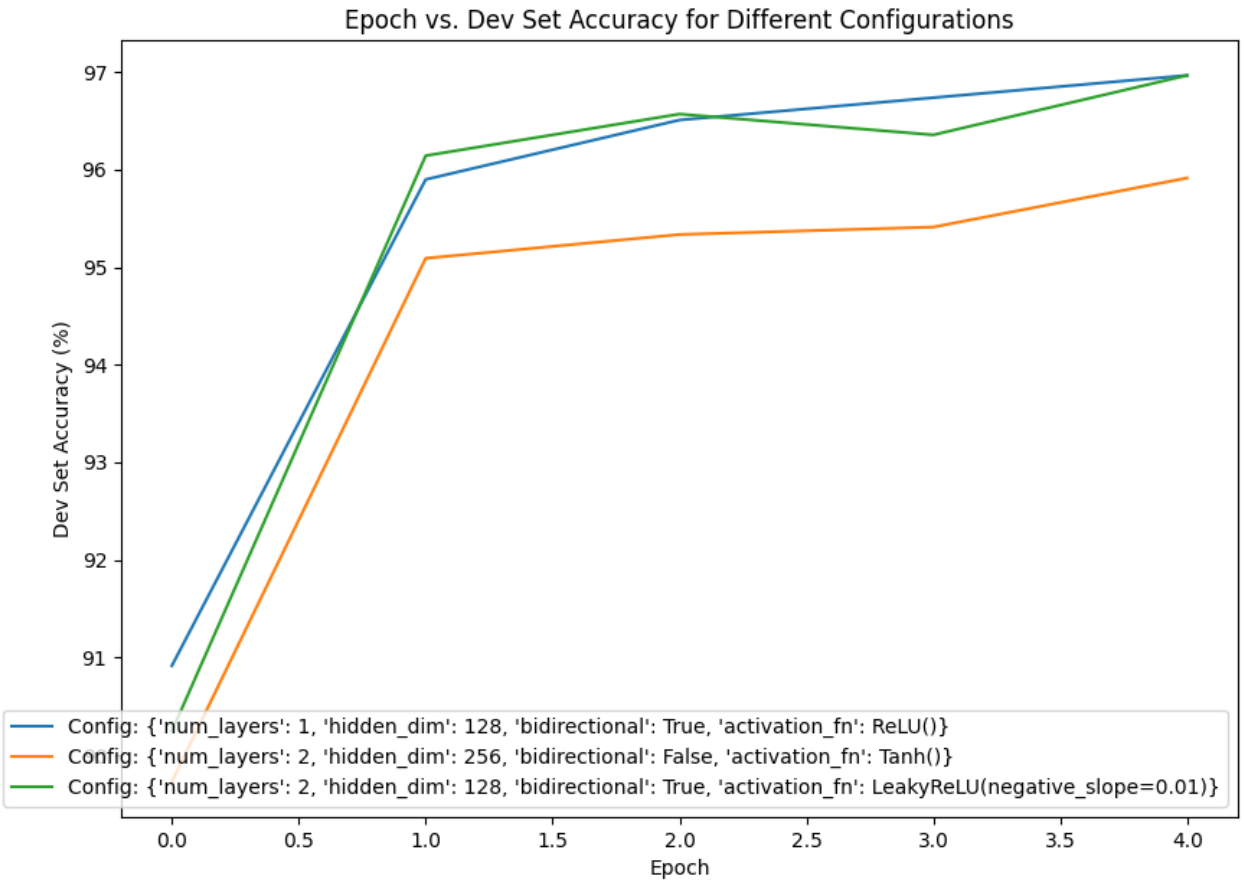
```
(myenv) PS C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2> python postagger.py -r
C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2\lstm.py:96: UserWarning: Creating a tensor from a list of numpy.ndarrays is extremely
slow. Please consider converting the list to a single numpy.ndarray with numpy.array() before converting to a tensor. (Triggered internally at ..\torch\csr
c\utils\tensor_new.cpp:278.)
  sentence_tensor = torch.FloatTensor(sentence)
Enter a sentence: an apple a day keeps the doctor away
[('an', 'DET'), ('apple', 'ADJ'), ('a', 'DET'), ('day', 'NOUN'), ('keeps', 'VERB'), ('the', 'DET'), ('doctor', 'NOUN'), ('away', 'ADP')]
Config: {'num_layers': 1, 'hidden_dim': 128, 'bidirectional': True, 'activation_fn': ReLU()}, Dev Accuracy: 96.79926840420667%
Config: {'num_layers': 2, 'hidden_dim': 256, 'bidirectional': False, 'activation_fn': Tanh()}, Dev Accuracy: 95.8085657674135%
Config: {'num_layers': 2, 'hidden_dim': 128, 'bidirectional': True, 'activation_fn': LeakyReLU(negative_slope=0.01)}, Dev Accuracy: 96.98216735253772%
Best Config: {'num_layers': 2, 'hidden_dim': 128, 'bidirectional': True, 'activation_fn': LeakyReLU(negative_slope=0.01)}, Best Dev Accuracy: 96.98216735253
3772%
 Test Accuracy: 97.07602339181285%
C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2\myenv\Lib\site-packages\sklearn\metrics\_classification.py:1509: UndefinedMetricWarnin
g: Recall is ill-defined and being set to 0.0 in labels with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2\myenv\Lib\site-packages\sklearn\metrics\_classification.py:1509: UndefinedMetricWarnin
g: F-score is ill-defined and being set to 0.0 in labels with no true nor predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
##################################################################
Dev set accuracy parameters Set Evaluation:
Accuracy: 96.98%
Recall (Micro): 96.98%
Recall (Macro): 80.62%
F1 Score (Micro): 96.98%
F1 Score (Macro): 80.66%
Confusion Matrix:


C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2\myenv\Lib\site-packages\sklearn\metrics\_classification.py:1509: UndefinedMetricWarnin
g: Recall is ill-defined and being set to 0.0 in labels with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2\myenv\Lib\site-packages\sklearn\metrics\_classification.py:1509: UndefinedMetricWarnin
g: F-score is ill-defined and being set to 0.0 in labels with no true nor predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
##################################################################
Test set accuracy parameters Set Evaluation:
Accuracy: 97.06%
Recall (Micro): 97.06%
Recall (Macro): 80.85%
F1 Score (Micro): 97.06%
F1 Score (Macro): 81.11%
Confusion Matrix:
```

- Evaluated three neural network configurations with varying hidden layer sizes and activation functions for hyperparameter tuning.
- Configuration 1: Single hidden layer with 128 units, ReLU activation function, yielded a development set accuracy of 96.799%.
- Configuration 2: Two hidden layers with 256 and 128 units respectively, Tanh activation function, resulted in a development set accuracy of 95.808%.
- Configuration 3: Two hidden layers with 128 units each, LeakyReLU activation function, achieved the highest development set accuracy of 96.982%.

- The third configuration with LeakyReLU was determined to be the best model due to its highest accuracy and the ability to mitigate the vanishing gradient issue.

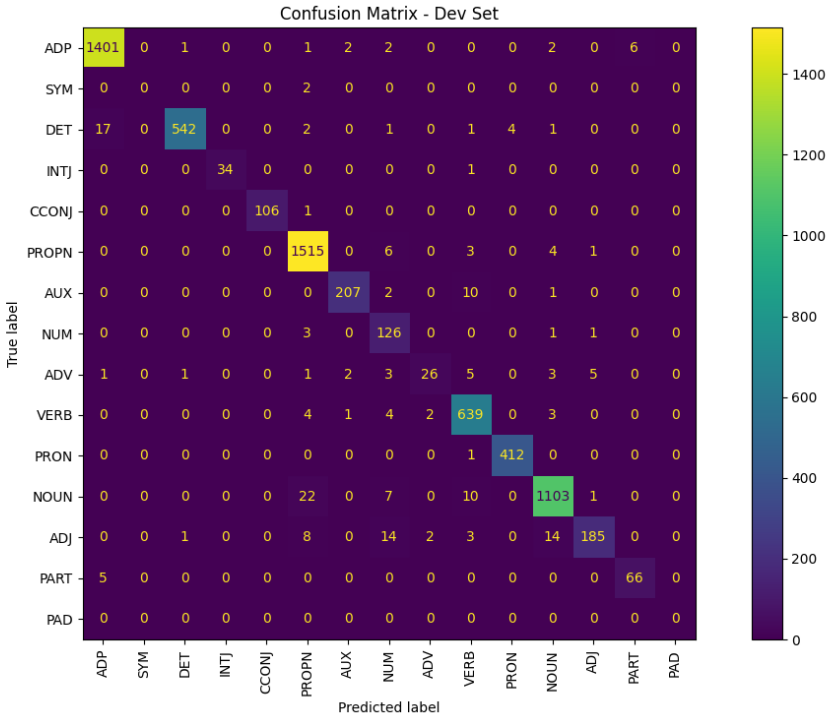
Graph and evaluation metrics for RNN(LSTM)



- The graph titled "Epoch vs. Dev Set Accuracy for Different Configurations" compares the development set accuracy over epochs for three different neural network configurations.
- Configuration 1, depicted in blue, represents a model with one hidden layer of 128 units and utilizes the ReLU activation function. It shows rapid improvement in accuracy, stabilizing around 96%.
- Configuration 2, shown in orange, has two hidden layers, with 256 and 128 units, and employs the Tanh activation function. This configuration exhibits a steady increase in accuracy but plateaus below the performance of the other two configurations at approximately 95%.
- Configuration 3, illustrated in green, also has two hidden layers of 128 units each but uses the LeakyReLU activation function. It achieves comparable accuracy to Configuration 1, indicating robust performance.
- The initial sharp increase in accuracy during the first epoch across all configurations suggests that the majority of learning occurs early in the training process.
- The plateau in accuracy for Configuration 2 may indicate that Tanh activation function is less effective for this particular task when compared to ReLU and LeakyReLU.

- Configuration 3's use of LeakyReLU, which prevents dying ReLU issues by allowing a small gradient when the unit is inactive, seems to be as effective as standard ReLU in this case.

Confusion metrics for DEV_SET



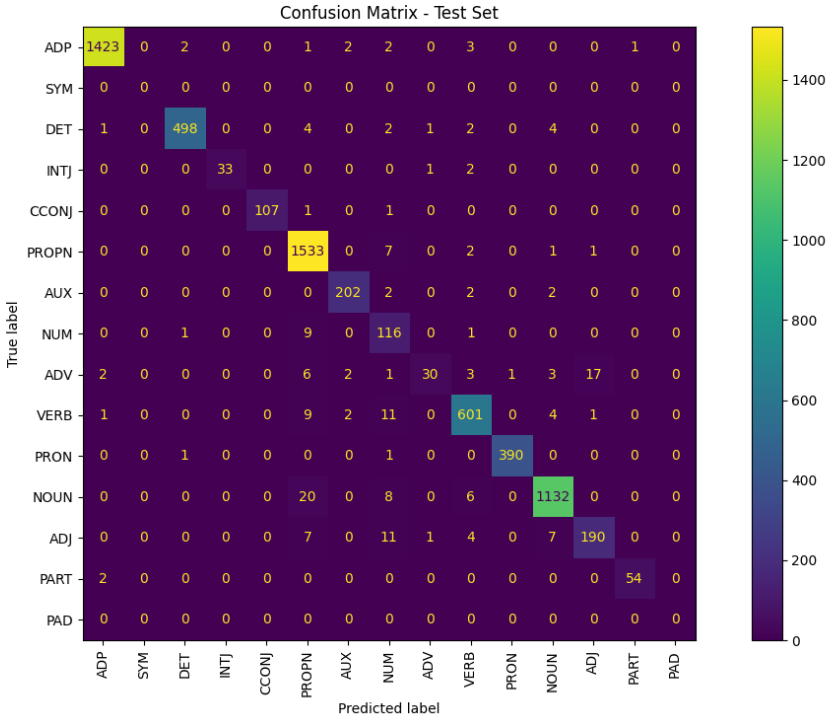
Diagonal Dominance:

- A well-performing model will have high numbers along the diagonal, representing correct classifications.
- In the matrix, high values on the diagonal indicate a large number of true positives for most POS tags, such as ADP (Ad position), PRPRN, and DET.

Off-Diagonal Elements:

- Elements off the diagonal represent misclassifications.
- For instance, there are instances where NOUN was misclassified as PROPRN .
- These misclassifications suggest areas where the model which i developed might be confused, potentially due to similarities in how these POS tags are used in context.

Confusion Metrics for Test_set



Model Accuracy:

- The high concentration of values along the diagonal in the confusion matrices indicates that the model generally exhibits a high degree of accuracy across both datasets.
- This is evidenced by the substantial true positive rates for tags such as 'ADP' and 'PRPN' and ‘NOUN’

Misclassification Patterns:

- The matrices reveal certain patterns of misclassification, where some POS tags are consistently confused with others.
- This suggests that the model may benefit from further tuning to better distinguish between these tags.

TEST SET - Accuracy

```
(myenv) PS C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2> python postagger.py -r
C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2\lstm.py:96: UserWarning: Creating a tensor from a list of numpy.ndarrays is extremely
slow. Please consider converting the list to a single numpy.ndarray with numpy.array() before converting to a tensor. (Triggered internally at ..\torch\csr
c\utils\tensor_new.cpp:278.)
  sentence_tensor = torch.FloatTensor(sentence)
Enter a sentence: an apple a day keeps the doctor away
[('an', 'DET'), ('apple', 'ADJ'), ('a', 'DET'), ('day', 'NOUN'), ('keeps', 'VERB'), ('the', 'DET'), ('doctor', 'NOUN'), ('away', 'ADP')]
Config: {'num_layers': 1, 'hidden_dim': 128, 'bidirectional': True, 'activation_fn': ReLU()}, Dev Accuracy: 96.79926840420667%
Config: {'num_layers': 2, 'hidden_dim': 256, 'bidirectional': False, 'activation_fn': Tanh()}, Dev Accuracy: 95.8085657674135%
Config: {'num_layers': 2, 'hidden_dim': 128, 'bidirectional': True, 'activation_fn': LeakyReLU(negative_slope=0.01)}, Dev Accuracy: 96.98216735253772%
Best Config: {'num_layers': 2, 'hidden_dim': 128, 'bidirectional': True, 'activation_fn': LeakyReLU(negative_slope=0.01)}, Best Dev Accuracy: 96.9821673525
3772%
 Test Accuracy: 97.07602339181285%
C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2\myenv\Lib\site-packages\sklearn\metrics\_classification.py:1509: UndefinedMetricWarnin
g: Recall is ill-defined and being set to 0.0 in labels with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2\myenv\Lib\site-packages\sklearn\metrics\_classification.py:1509: UndefinedMetricWarnin
g: F-score is ill-defined and being set to 0.0 in labels with no true nor predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
######################################################################
Dev set accuracy parameters Set Evaluation:
Accuracy: 96.98%
Recall (Micro): 96.98%
Recall (Macro): 80.62%
F1 Score (Micro): 96.98%
F1 Score (Macro): 80.66%
Confusion Matrix:


C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2\myenv\Lib\site-packages\sklearn\metrics\_classification.py:1509: UndefinedMetricWarnin
g: Recall is ill-defined and being set to 0.0 in labels with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2\myenv\Lib\site-packages\sklearn\metrics\_classification.py:1509: UndefinedMetricWarnin
g: F-score is ill-defined and being set to 0.0 in labels with no true nor predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
######################################################################
Test set accuracy parameters Set Evaluation:
Accuracy: 97.06%
Recall (Micro): 97.06%
Recall (Macro): 80.85%
F1 Score (Micro): 97.06%
F1 Score (Macro): 81.11%
Confusion Matrix:
```

- I have added the above screenshots which displays the Test_Set accuracy of around 97 percent which is considerably good.

TEST SET AND DEV SET EVALUATION METRICS

```
(myenv) PS C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2> python postagger.py -r
C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2\lstm.py:96: UserWarning: Creating a tensor from a list of numpy.ndarrays is extremely
slow. Please consider converting the list to a single numpy.ndarray with numpy.array() before converting to a tensor. (Triggered internally at ..\torch\csr
c\utils\tensor_new.cpp:278.)
  sentence_tensor = torch.FloatTensor(sentence)
Enter a sentence: an apple a day keeps the doctor away
[('an', 'DET'), ('apple', 'ADJ'), ('a', 'DET'), ('day', 'NOUN'), ('keeps', 'VERB'), ('the', 'DET'), ('doctor', 'NOUN'), ('away', 'ADP')]
Config: {'num_layers': 1, 'hidden_dim': 128, 'bidirectional': True, 'activation_fn': ReLU()}, Dev Accuracy: 96.79926840420667%
Config: {'num_layers': 2, 'hidden_dim': 256, 'bidirectional': False, 'activation_fn': Tanh()}, Dev Accuracy: 95.8085657674135%
Config: {'num_layers': 2, 'hidden_dim': 128, 'bidirectional': True, 'activation_fn': LeakyReLU(negative_slope=0.01)}, Dev Accuracy: 96.98216735253772%
Best Config: {'num_layers': 2, 'hidden_dim': 128, 'bidirectional': True, 'activation_fn': LeakyReLU(negative_slope=0.01)}, Best Dev Accuracy: 96.9821673525
3772%
 Test Accuracy: 97.07602339181285%
C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2\myenv\Lib\site-packages\sklearn\metrics\_classification.py:1509: UndefinedMetricWarnin
g: Recall is ill-defined and being set to 0.0 in labels with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2\myenv\Lib\site-packages\sklearn\metrics\_classification.py:1509: UndefinedMetricWarnin
g: F-score is ill-defined and being set to 0.0 in labels with no true nor predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
######################################################################
Dev set accuracy parameters Set Evaluation:
Accuracy: 96.98%
Recall (Micro): 96.98%
Recall (Macro): 80.62%
F1 Score (Micro): 96.98%
F1 Score (Macro): 80.66%
Confusion Matrix:


C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2\myenv\Lib\site-packages\sklearn\metrics\_classification.py:1509: UndefinedMetricWarnin
g: Recall is ill-defined and being set to 0.0 in labels with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\nithi\OneDrive\Desktop\IIIT HYDERABAD\sem2\NLP\ASSIGNMENT 2\myenv\Lib\site-packages\sklearn\metrics\_classification.py:1509: UndefinedMetricWarnin
g: F-score is ill-defined and being set to 0.0 in labels with no true nor predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
######################################################################
Test set accuracy parameters Set Evaluation:
Accuracy: 97.06%
Recall (Micro): 97.06%
Recall (Macro): 80.85%
F1 Score (Micro): 97.06%
F1 Score (Macro): 81.11%
Confusion Matrix:
```

Observations:

- **Accuracy:** The model achieves high accuracy on both the dev set (96.98%) and the test set (97.06%). This indicates that the model is successfully classifying a large proportion of tokens correctly.
- **Recall:**
  - **Micro-averaging:** This metric shows similar performance for both dev and test sets, with a slight improvement on the test set (97.06% vs. 96.98%). This suggests that the model is effectively identifying positive cases overall.
  - **Macro-averaging:** Both dev and test sets show lower recall values compared to micro-averaging (80.62% and 80.85%, respectively). This indicates that the model might have difficulty classifying some of the less frequent classes.
- **F1 Score:** Similar to recall, the F1 score follows the same pattern. Micro-averaging shows similar performance between dev and test sets (96.98% and 97.06%, respectively), while macro-averaging shows lower values for both sets (80.66% and 81.11%, respectively).

FINAL CONCLUSION ON LSTM

- **Test Set Accuracy:** 97.06%, indicating successful classification of a large portion of tokens.
- **Micro-averaged Recall and F1 Score:** High values on both dev and test sets, suggesting overall effectiveness in identifying positive cases.

- **Macro-averaged Recall and F1 Score:** Lower values compared to micro-averaging, suggesting potential difficulties with less frequent classes.

************************END OF REPORT***********************************