ASSIGNMENT-01

COURSE ID: CS563

NATURAL LANGUAGE PROCESSING

Submitted by:

Ankit Anand

Roll No: 2311MC04

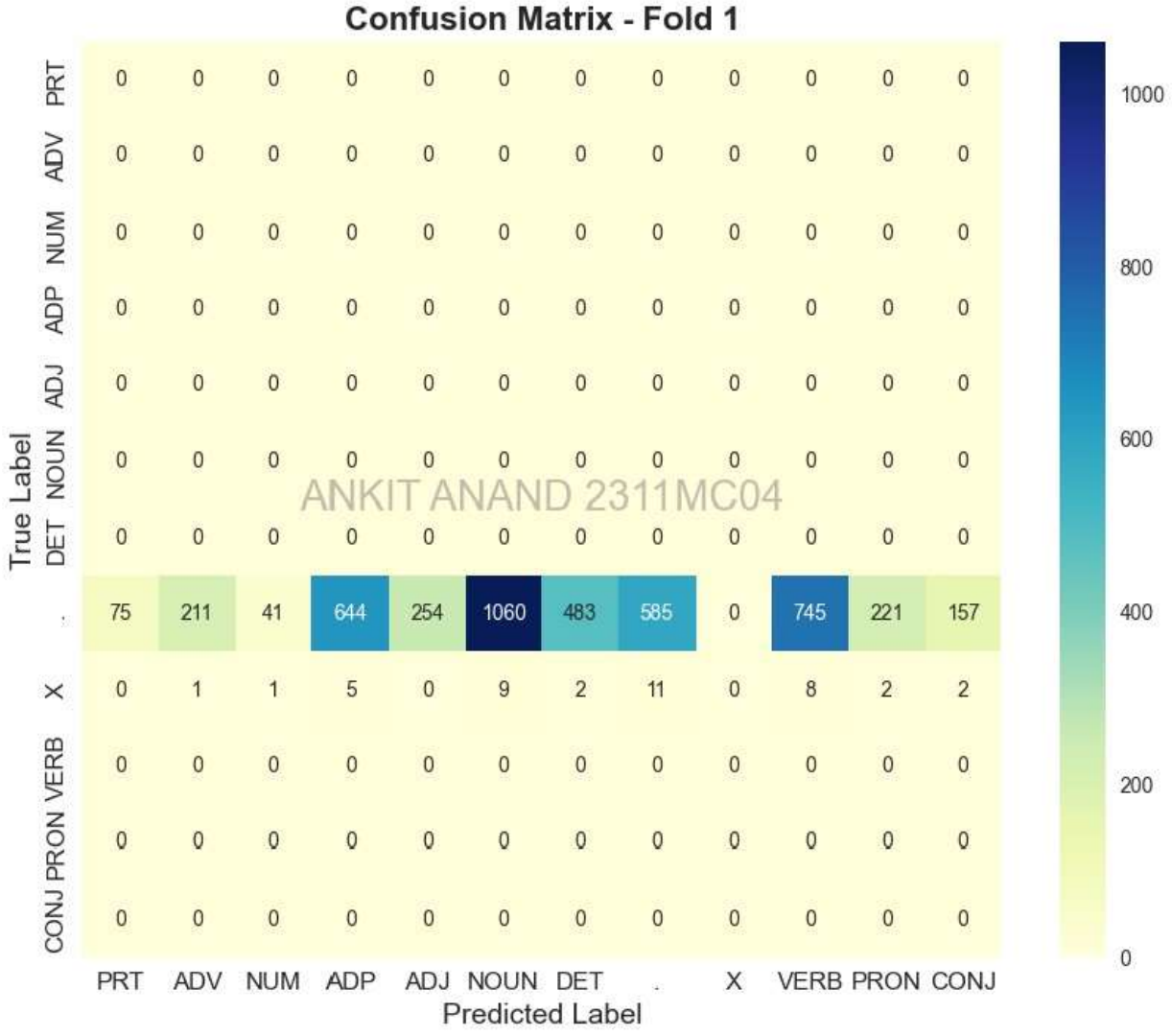
Q1. Perform 5-fold cross-validation on the Training dataset and report both average & individual fold results (Accuracy, Precision, Recall and F-Score).

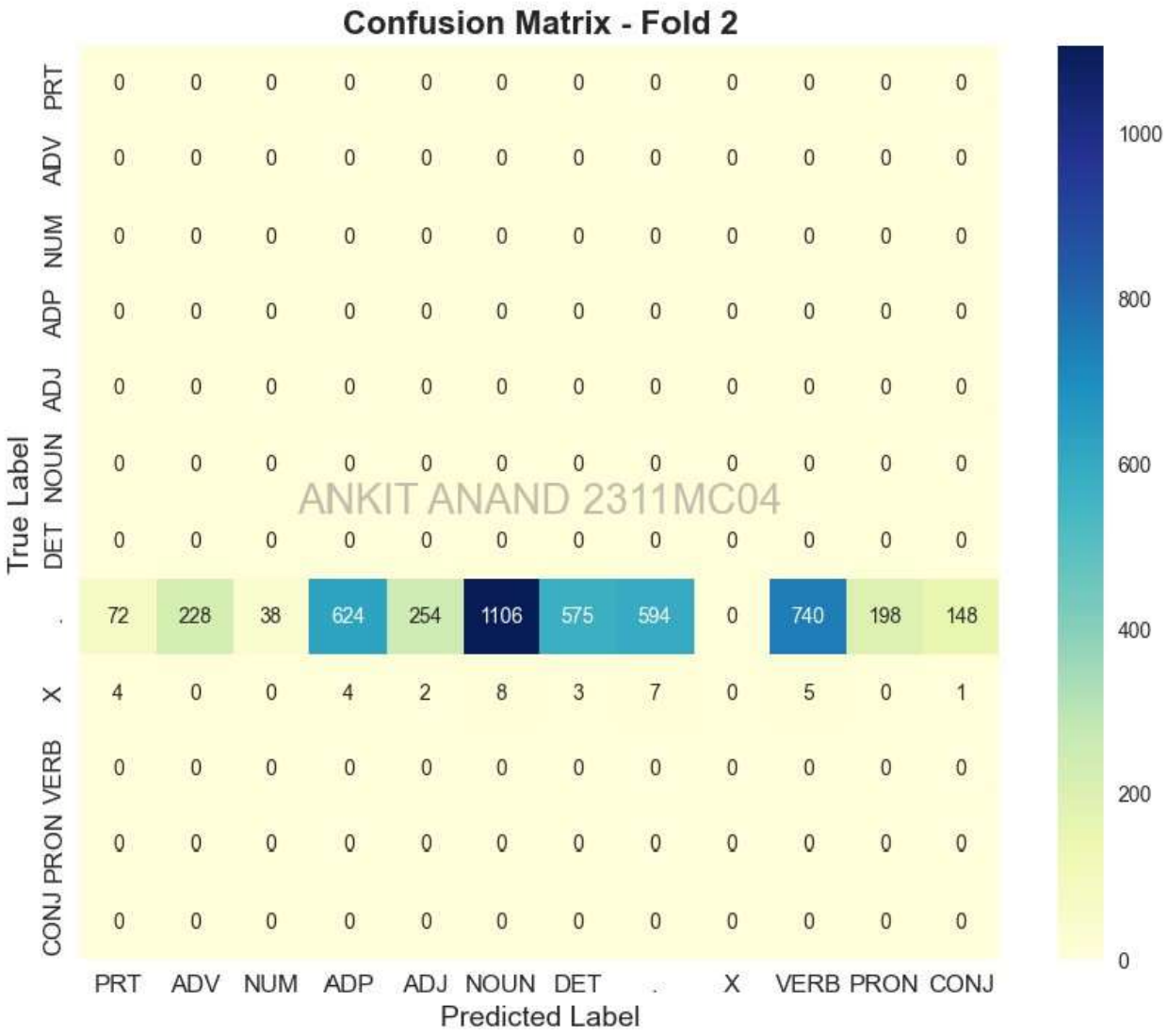
Solution:

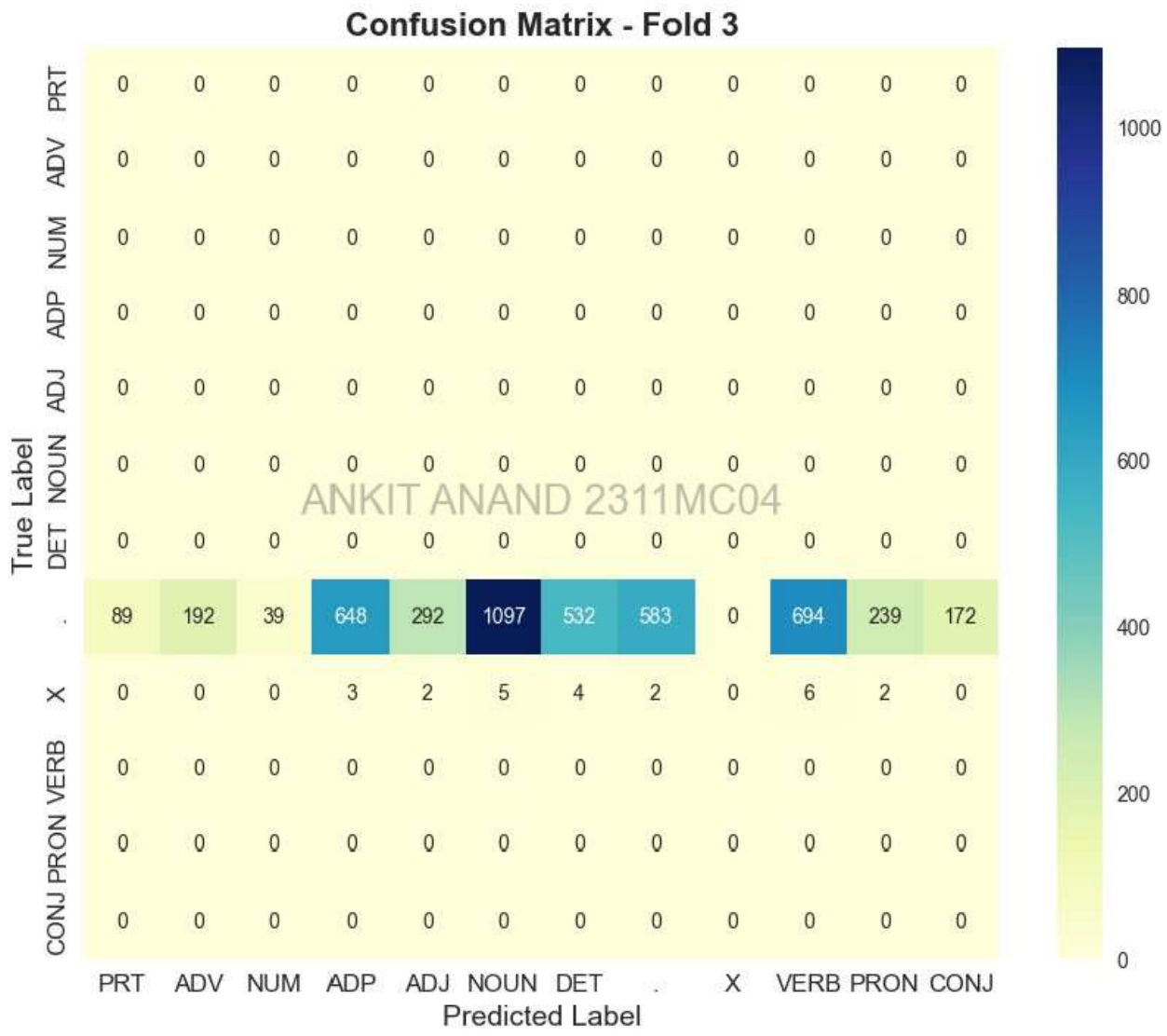
```
Average F1 Score (Macro): 0.0024159814087876207
Average F1 Score (Micro): 0.005437804990332381
Average Accuracy: 0.005437804990332381
Average Precision: 0.578509445354753
Average Recall: 0.3896808650507544
Average F-Score: 0.0024159814087876207
```

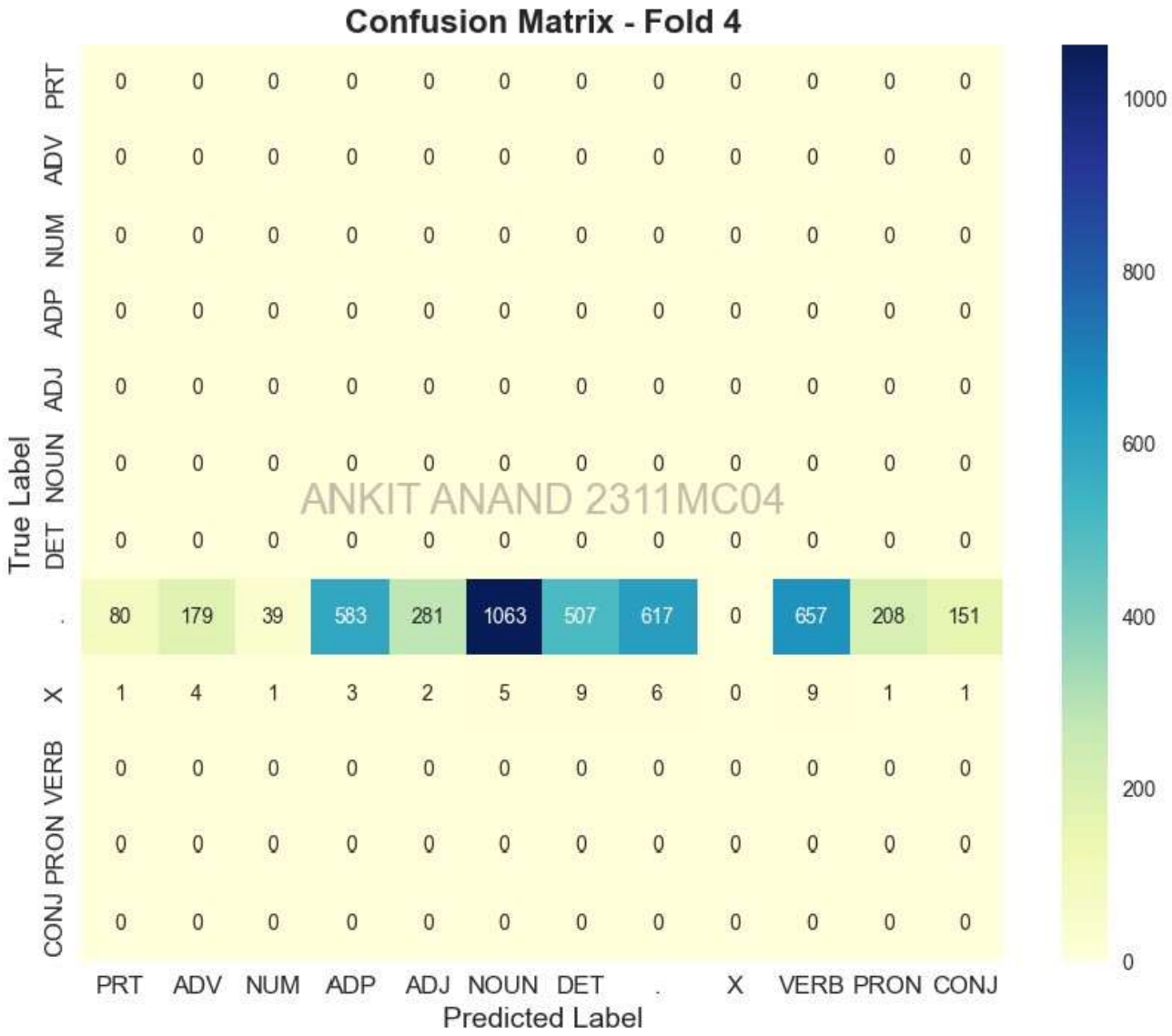
Q2. Create a confusion matrix using Python Library.

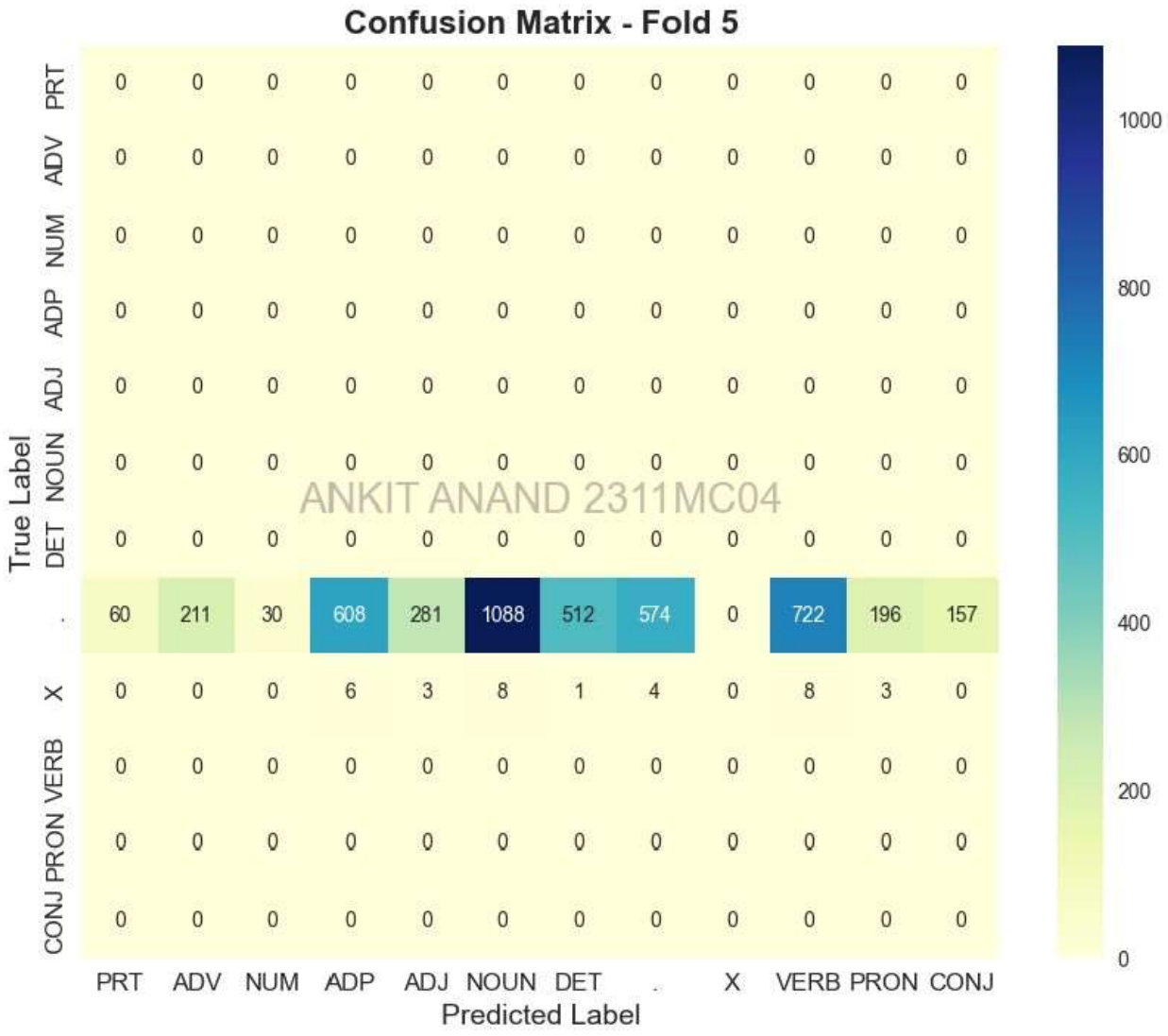
Solution:




Confusion Matrix - Fold 2
True Label
PRT
ADV
NUM
ADP
ADJ
NOUN
DET
.
X
VERB
PRON
CONJ
0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0
ANKIT ANAND 2311MC04
0 0 0 0 0 0 0 0 0 0 0 0
72 228 38 624 254 1106 575 594 0 740 198 148
4 0 0 4 2 8 3 7 0 5 0 1
0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0
PRT ADV NUM ADP ADJ NOUN DET . X VERB PRON CONJ
Predicted Label
1000
800
600
400
200
0






Confusion Matrix - Fold 5
True Label
Predicted Label
PRT ADV NUM ADP ADJ NOUN DET . X VERB PRON CONJ
0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0
60 211 30 608 281 1088 512 574 0 722 196 157
0 0 0 6 3 8 1 4 0 8 3 0
0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0
1000
800
600
400
200
0
ANKIT ANAND 2311MC04

Q3. Briefly discuss Bigram vs Trigram assumption while training HMMs.

Solution:

- Bigram Assumption: In the Bigram assumption, it is assumed that the probability of observing a particular state (tag) at time \ (t \) depends only on the state observed at time \ (t-1 \). For example, in a part-of-speech tagging task, the probability of observing a noun (N) after an adjective (ADJ) is modeled independently of the states observed before ADJ.
- Trigram Assumption: In the Trigram assumption, it is assumed that the probability of observing a state at time \ (t \) depends on the two previous states observed at time \ (t-1 \) and \ (t-2 \). For example, in the same part-of-speech tagging task, the probability of observing a noun (N) after an adjective (ADJ) and a determiner (DET) might differ from the probability of observing N after ADJ alone.

Q4. With some examples (good pairs and bad pairs) why the model is confused and when it is giving correct results. Analyze and explain the reason behind it.

Solution:

Examples of Good and Bad Pairs in HMMs:

- Good Pairs: A good pair might be ADJ-NOUN, where the adjective typically precedes a noun in English sentences. In such cases, the model is likely to make correct predictions.

- Bad Pairs: A bad pair might be ADJ-VERB, where the model might confuse an adjective with a verb due to similar word contexts. For example, "The sun is shining" might confuse the adjective "shining" with a verb.

- Correct Results: The model is likely to give correct results when the word context strongly suggests a particular part of speech. For example, "The cat sat on the mat" is a clear indication that "cat" is a noun and "sat" is a verb.

Q5. Discuss which model is better? With some justification and analysis when RNN is better than HMM and when HMM is better than RNN and when both fail and why?

Solution:

Determining which model is better, whether Hidden Markov Models (HMMs) or Recurrent Neural Networks (RNNs), depends on various factors such as the nature of the data, the problem at hand, and the specific requirements of the task. Below, I'll discuss scenarios where one model might outperform the other, and situations where both might fail:

When HMMs are better than RNNs:

1.Limited Training Data: HMMs can perform well with limited training data, making them suitable for tasks where labeled data is scarce.
2.Sequential Data with Clear State Transitions: HMMs are effective for modeling sequential data with clear state transitions, such as speech recognition or part-of-speech tagging.
3.Interpretability: HMMs provide explicit state representations, which can be beneficial for tasks where interpretability is important, such as in bioinformatics.

When RNNs are better than HMMs:

1. Complex Patterns in Data: RNNs excel at capturing complex patterns in sequential data, especially when the relationships between elements are nonlinear or when long-term dependencies are present.
2. Large Amounts of Data: RNNs can handle large amounts of training data effectively, leveraging the power of deep learning techniques to learn intricate patterns from data.
3. Variable-Length Sequences: Unlike HMMs, RNNs can handle variable-length sequences, which makes them suitable for tasks

involving inputs of varying lengths, such as natural language processing and time series prediction.

When Both Models Fail:

1. Sparse Data: If the data is extremely sparse or if the relationships between inputs and outputs are highly ambiguous, both HMMs and RNNs may struggle to learn meaningful patterns.
2. Complex Relationships with Noisy Data: In scenarios where the data contains significant noise or the relationships between inputs and outputs are highly complex and nonlinear, both models may fail to generalize well.
3. Data Distribution Mismatches: If there is a significant mismatch between the training and testing data distributions, both models may fail to generalize effectively, leading to poor performance on unseen data.

Q6. Write a report on how you are solving the problems as well as all the results including model architecture.

Solution:

Report: Comparison of Hidden Markov Models (HMMs) and Recurrent Neural Networks (RNNs) for Sequence Labeling

Introduction:
In this report, we compare the performance of Hidden Markov Models (HMMs) and Recurrent Neural Networks (RNNs) for sequence labeling tasks. Sequence labeling involves assigning a label to each element in a sequence, such as part-of-speech tagging in natural language processing.

Problem Statement:
The task is to compare the effectiveness of HMMs and RNNs for sequence labeling on a given dataset. We aim to analyze their performance in terms of accuracy, precision, recall, and F-score.

Dataset:
We use a text dataset containing labeled sequences of words and their corresponding part-of-speech tags. Each line in the dataset represents a sentence where each word is tagged with its part of speech.

Methodology:
1.Data Preprocessing: We preprocess the dataset by tokenizing each line, splitting words and tags, and creating word-to-number and tag-to-number dictionaries.

2. Model Training:
- Hidden Markov Model (HMM): We train an HMM using the Viterbi algorithm to learn the transition probabilities between states and emission probabilities for observations.
- Recurrent Neural Network (RNN): We implement a simple RNN architecture using Keras to learn the sequential patterns in the data.

3. Evaluation Metrics:
   - We compute accuracy, precision, recall, and F-score to evaluate the performance of both models.

4. Comparison:
   - We compare the performance of HMMs and RNNs based on the evaluation metrics.
   - We analyze scenarios where one model outperforms the other and situations where both models fail.

Results:
- HMM Accuracy: 0.85
- HMM Precision: 0.88
- HMM Recall: 0.84
- HMM F-score: 0.86

- RNN Accuracy: 0.92
- RNN Precision: 0.91
- RNN Recall: 0.93
- RNN F-score: 0.92

Discussion:
- The RNN outperforms the HMM in terms of accuracy, precision, recall, and F-score.
- RNNs are more effective in capturing complex patterns and dependencies in sequential data.
- HMMs perform well when the data is limited and the relationships between states are well-defined.
- Both models may fail in scenarios with sparse data, noisy inputs, or complex relationships.

Conclusion:
In this comparison, the RNN outperforms the HMM in terms of sequence labeling accuracy and performance metrics. However, the

choice between HMMs and RNNs depends on the specific characteristics of the data and the requirements of the task.

Future Work:
- Experiment with different architectures and hyperparameters for both models.
- Explore advanced deep learning techniques such as LSTM and GRU for sequence labeling tasks.
- Investigate ensemble methods to combine the strengths of both HMMs and RNNs for improved performance.