Monsoon 2020 - NLP
ReadMe File

NOTE:-

- The Output.pdf file contains the sample outputs along with screenshots wherever required.
- **Theory answers are written in the ReadMe.pdf file.**
- Both .py and .ipynb are attached in the zipped folder.

Q1)

- Importing the required libraries.
- Reading the given file.
    - Extracting sentences based on the next line character.
    - Extract word-tag pairs based on space as delimiter.
    - Extract words and tags separately based on underscore as delimiter.
- Calculating the statistics of sentences.
    - Minimum length of sentence
    - Maximum length of sentence
- Creating a list of tuples for the entire file.Each tuple contains (*word,tag)*.
- 'Vocab' is a set of unique words.
- 'Tag_set' is the set of unique tags.
- 'Tag_count' is the dictionary of tag and corresponding count of occurrences of tag in the given corpus.
- Emission Probabilities:-
    - For calculating the emission probabilities, initially calculating word-to-tag count which says about the count of a word given the tag.
    - Using these word-to-tag counts,emission probabilities are calculated.
- Transition Probabilities:-
    - Calculating Tag2-to-Tag1 count which says the count of tag2 given tag1.
    - With these Tag2-to-Tag1 counts, transition probabilities are calculated.
    - While calculating these probabilities, Laplace smoothing is used to handle the values of zeros.
- Viterbi Algorithm:
    - Initially, creating a predict_proba list with size of input word file.
    - For every word,
        - Creating a list for storing the probabilities that a particular tag can set to this word.
        - This probability is calculated using the product of emission probability and transition probability.

        - Extract the max probability from this list, indicating that the tag with maximum probability is correct for the current word.
        - Append the tag with max probability to predict_proba list.
    - Now, this predict_proba list contains the corresponding tags for the list of words.
- A function is written to calculate the confusion-matrix which takes the predicted tags and ground-truth tags and returns the matrix.
    - The same function is used to calculate the confusion matrix in all cases.
- Function for accuracy:-
    - Takes predicted tags and true tags as arguments and returns the accuracy for this prediction.
- Function for precision,recall and F1-score:-
    - Takes the Confusion matrix as an argument for this function.
    - For every tag,
        - Calculating True Positives,True Negatives,False Positives and False Negatives.
        - Using these values, precision,recall and f1-score is calculated for that particular tag.
    - Using the values for individual tags, the overall precision,recall and f1-score is calculated.

3 - fold Cross Validation:-

- Dividing the given set of sentences into three parts.
- For each fold,we consider one part as a validation set and the other two as training data.
- For every fold:-
    - Obtaining predicted tags for the validation data using Viterbi Algorithm.
    - Obtain the wrongly predicted words in particular fold.
    - Calculating Confusion Matrix with Predicted tags and True tags.
    - Calculating Accuracy.
    - Calculating Precision,Recall and F1-score for every tag in the validation data and for the overall validation data.

TRIGRAMS

- Emission Probabilities
    - These are the same for any n-grams as these are just the probability of a word given tag.
- Transition Probabilities
    - These are the probabilities of a tag3 given that the previous two tags are tag2 and tag1.

    - For computational purposes, I am combining the tag2 and tag1 with '!' (exclamation) in the middle.
    - For eg:- if tag2='NN' and tag1='AT', I will store them as 'NN!AT'.
    - Whenever I need these tags separately, I will split them based on '!' as a delimiter.
        - While calculating these probabilities, we will be using the probability of bigrams, i.e., probability of tag2 given tag1.
    - Laplace smoothing is used to handle all the cases.
- All the steps followed for each fold in 3-fold cross validation for HMM of length 1, are also followed here.

Observations:-

- There is not much difference in the accuracy values of bigrams and Trigrams.
- As the data is imbalanced, the evaluation metric cannot be the accuracy in this case.
- So, if we compare the precision,recall and F1-score values of bigrams language model and trigrams language model,
    - We can conclude that, for the given corpus '**BIGRAM LANGUAGE MODEL'** will work better comparatively.

Q2) HMM for Markov Assumption length 2

HMM:-

Goal:- Maximize $P(S|O)$ tag sequence by choosing best sequene S.

O → Observation ; S → States.

$$\boxed{S^* = \text{argmax}_S \; P(S|O)}$$

$$P(S|O) = P(\{S_1, S_2 \ldots, S_n\} | \{O_1, O_2, \ldots O_n\})$$

→ Chain Rule

$$= P(S_1|O) \cdot P(S_2|S_1, O), \ldots \ldots$$

→ Apply Markov Assumption

$$P(S|O) = \text{argmax}_S \; P(O|S) \cdot P(S)$$

Prior :-

$$P(S) = P(S_1) \cdot P(S_2|S_1) \cdot P(S_3|S_2 S_1) \cdot P(S_4|S_3 S_2) \ldots \ldots$$

Likelihood:

$$P(O|S) = P(O_1|S) \cdot P(O_2|O_1, S) \ldots \ldots$$

⇒ For probability of tag3 given that previous two tags are tag2 and tag1, we will be using the probability of tag2 given tag1.

Q3)

- For every set of validation, I printed the wrongly predicted words as a list of lists which contain [word,true-tag,predicted-tag].
- If we observe true-tags in this list, some of those tags are AT,NN,IN,VBD.
- In the given corpus, these tags have more frequency and wrongly predicted words are also mapped to different types of tags.
- Due to this there will be more probability of assigning wrong labels to the word

References used:-

➔ https://github.com/shikhinmehrotra/NLP-POS-tagging-using-HMMs-and-Viterbi-heuristic/blob/master/NLP-POS%20tagging%20using%20HMMs%20and%20Viterbi%20heuristic.ipynb
➔ https://youtu.be/68hmUltbPnw
➔ Textbook posted in google classroom for Viterbi Algo.
➔ https://stackoverflow.com/questions/2148543/how-to-write-a-confusion-matrix-in-python
➔ https://stackoverflow.com/questions/31324218/scikit-learn-how-to-obtain-true-positive-true-negative-false-positive-and-fal
➔ https://machinelearningmastery.com/precision-recall-and-f-measure-for-imbalanced-classification/
➔ Slides for solving Q2.

****