Name: Shivang Soni Reg. No: 915623718

Que 1.
The learning curve plots a performance measure evaluated on a fixed test set ( y - axis ) against the training data set size ( x - axis ). Generate a learning curve for the bigram HMM as we've provided it, using section 22 to evaluate . What are your thoughts about getting more POS - tagged data and how that would affect your system?

Solution 1.

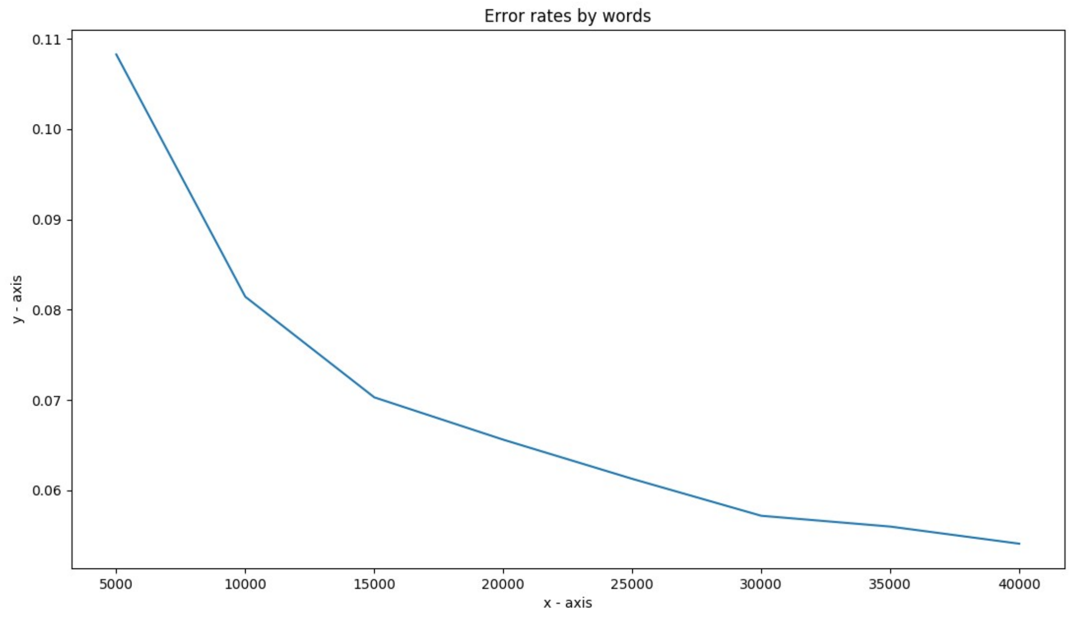

***Figure 1: For English Data Set***

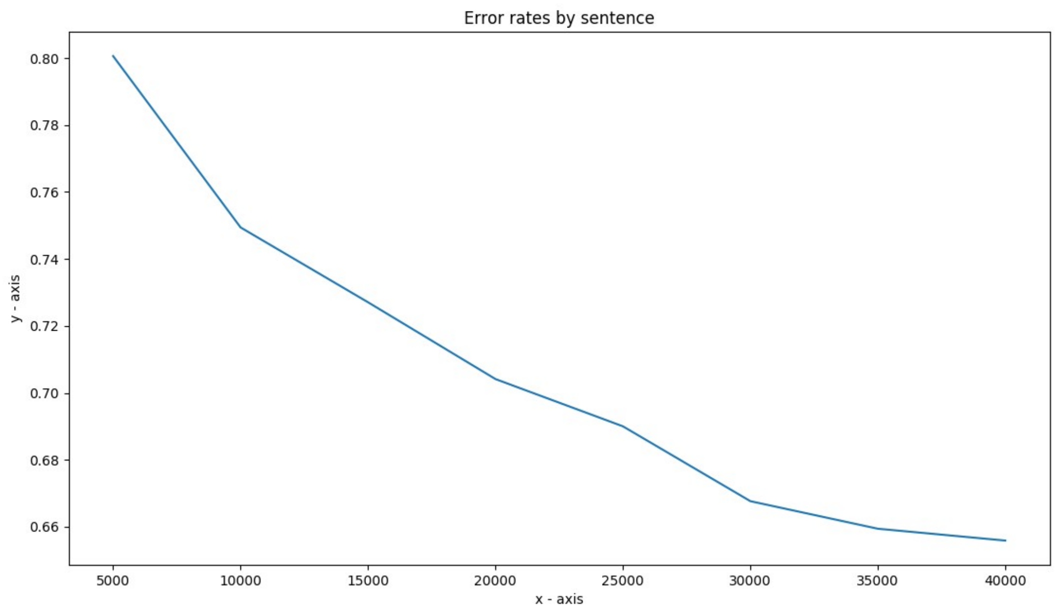

***Figure 2: Error Rate by Sentence for English DataSet***

As per Figure 1 and Figure 2 we can interpret that more will be the error rates means less tags matched. As if the training dataset is small that means many unseen words will be encountered when testing on test data, so more the number of unseen words more will be the error rate which infers the possibility of matching word to its correct tag is very less. Error can possibly occur because of two reasons:

- The dataset is trained on small quantity of data. In case of training on small dataset model will encounter more out of vocabulary words, leading to decrease in its performance and high error rate while evaluation.
- More error rate can also be caused due to bad model choice for the task.

Hence, based on the above points we can say that amount of data considered to train the model have an impact on model's accuracy. For example, more data (more words in vocabulary) used for analysis leads to decreases in the error of encountering "OOV_Words".

Name: Shivang Soni Reg. No: 915623718

Que 2. 1)Describe your approach clearly
2)Provide the performance of your model on ptb.22.*
3) Run your tagger on ptb.23.txt (the test data) and turn in the code and output.
Sol 2.
(1)I describe below the detail description of steps I followed

STEP 1:
I modified the **train_hmm.py** file to a file named **finaltrain.hmm** for trigram
Execution command: **python finaltrain.py ptb.2-21.tgs ptb.2-21.txt > my.hmm**
This file generates the updated value of transitions and emit matrix. The transitions matrix now contains 2 pervious tags and 1 current state tag. The emit matrix nonetheless remains the same as generated by the bigram model.

STEP 2:
I created a new python script file for implementing viterbi algorithm using Trigram.
The file is named as **modifiedviterbi.py** and can be executed by command:
Execution command: **python modifiedviterbi.py my.hmm < ptb.22.txt > my.out**
This file reads the data generated for the transition and emit matrix. After applying filtering on the data by using regular expressions, this file stores the data in trans and emit dictionary.

STEP 3:
**Implementation of trigram Viterbi:**
A text file is read from the Command Line which is splitted into lines and then to words. This functionthen attaches the appropriate POS tags to the word based on the previous two words. This takes transition matrix, emit matrix, vocabulary words(These words are on which model is trained) ,word List(The list on which the model is to be tested), taglist(set of all possible states) and in turn returns the corresponding POS tags for word List words.

STEP 4:
**Improvement using Deleted Interpolation:**
I have implemented a function Deleted Interpolation in order to normalize the probablity values so as to fetch the proper improvement.

STEP 5:

Name: Shivang Soni Reg. No: 915623718

**EVALUATION:**
Execution command: **./tag_acc.pl ptb.22.tgs my.out**

(2) Performance on ptb.22.*:

```
      error rate by word:  0.238726724331331 (9577 errors out of 40117)
      error rate by sentence:  0.977058823529412 (1661 errors out of 1700)
```

The reason for the low performance is due to not proper handling of the unknown words, can also be due to choosing among the set of tags with same probability or might be due to over-fitting.

(3) The output tags for **ptb.23.txt** file is there in file **Generated-ptb.23.tgs** .

Que 3:
1.) What factors lead to the differences among performance on English, Japanese, and Bulgarian?
2.) What about the base line and your model makes them relatively better or worse
on the data in these other two languages?

Solution 3:

Name: Shivang Soni Reg. No: 915623718

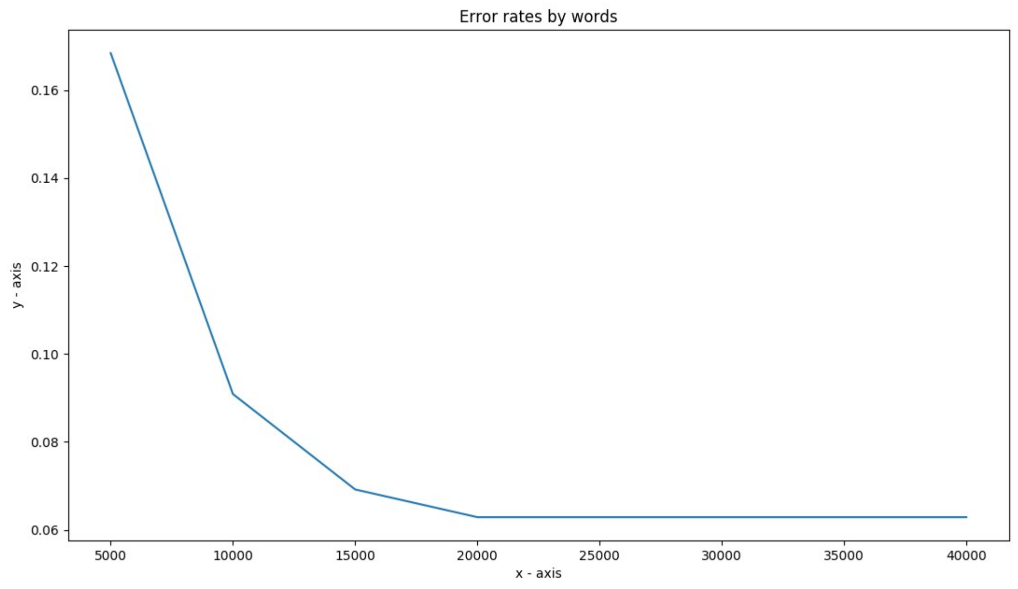

***Figure 3: For JV Data Set***

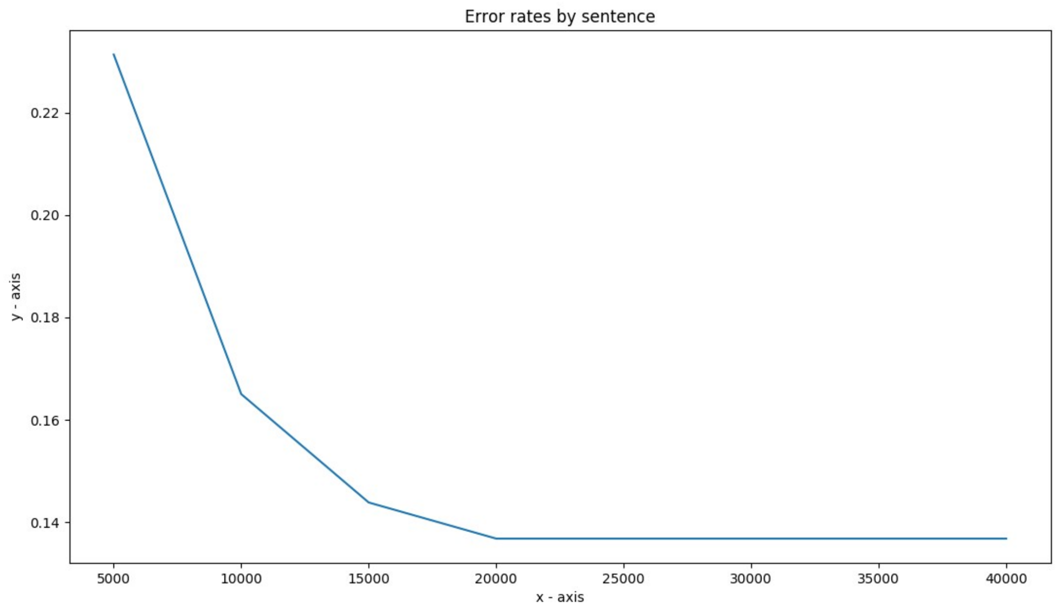

*Figure 4: For JV DataSet Error By Sentence for English*

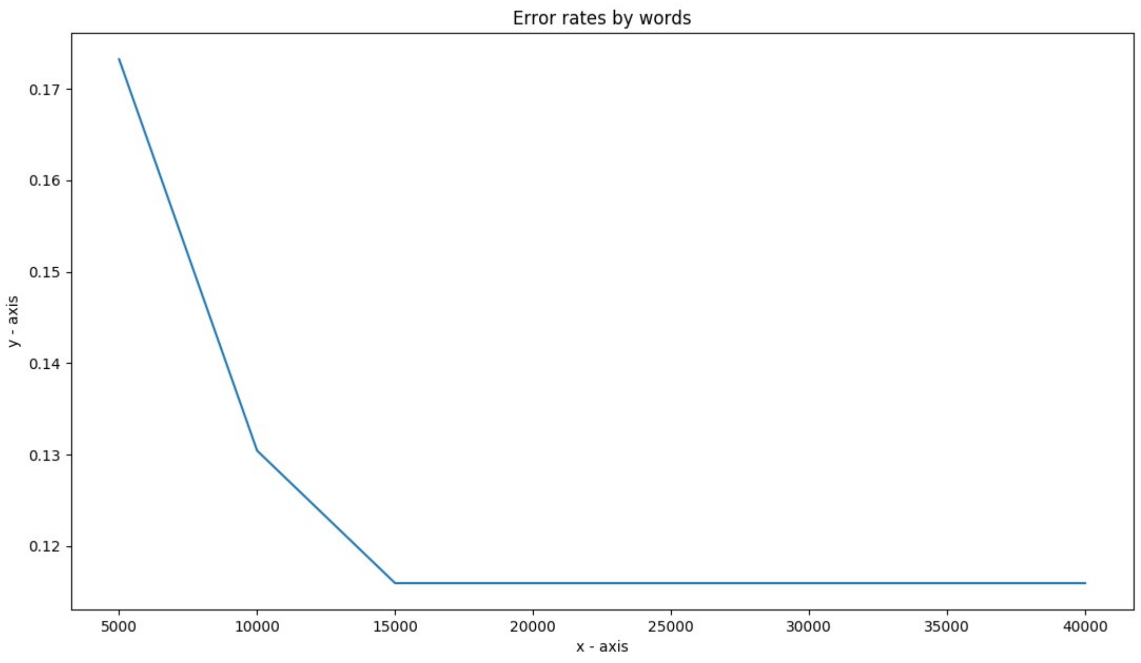

***Figure 5: BTB Data Set***

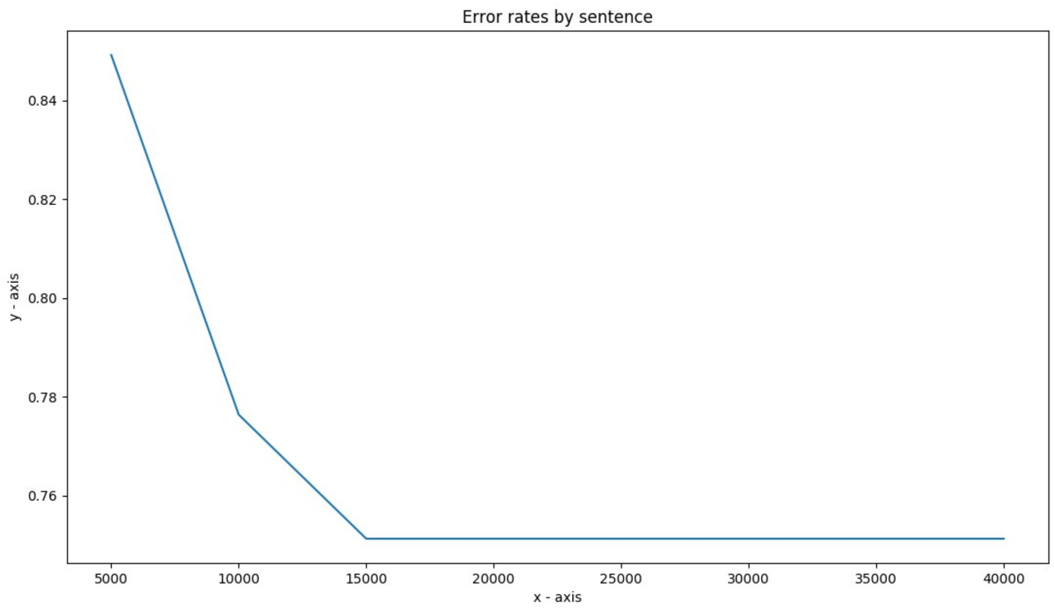

***Figure 6: Error by Sentence for btb Data Set***

Name: Shivang Soni Reg. No: 915623718

1.)
As Illustrated in Figure: 1,3,5 above:
For the English Data Set as the number of words taken into account while training is more as compared to Japanese and Bulgarian so the error by word for English dataset is low as compared to the other two datasets due to the presence of less number of Out Of Vocabulary words.
Similarly the above mention conclusion also holds true for Error by sentence as been Illustrated in Figure: 2,4,6 above.

Among the other two datasets the performance of Japanese dataset is better due to presence of more words while training as compared to Bulgarian dataset.

2.)
Error rates by word for my implemented model:

Name: Shivang Soni Reg. No: 915623718

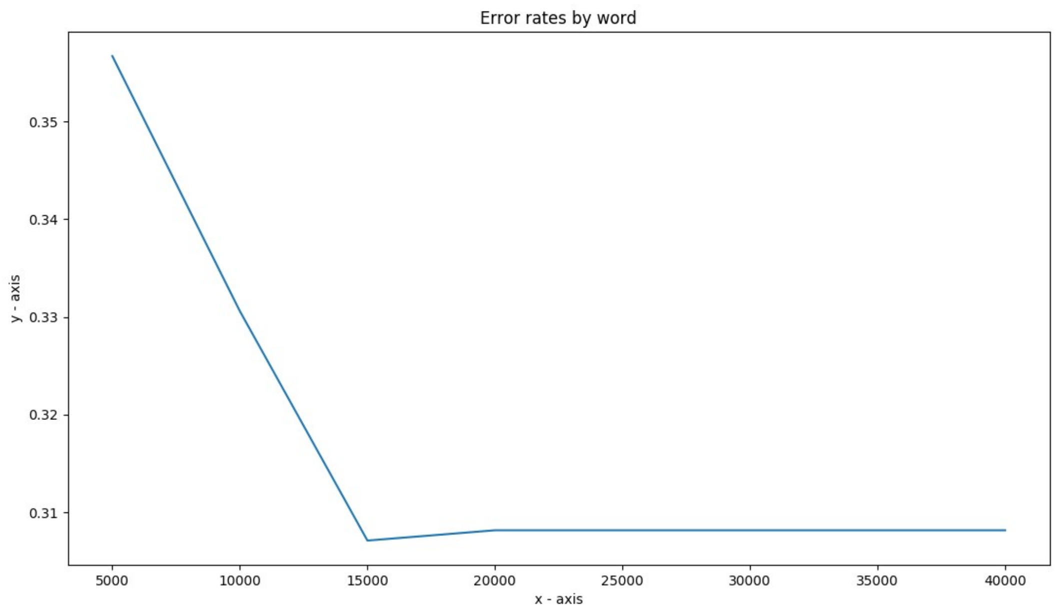

***Figure 7:Japanese Error Rate By Word using TrigramHmm***

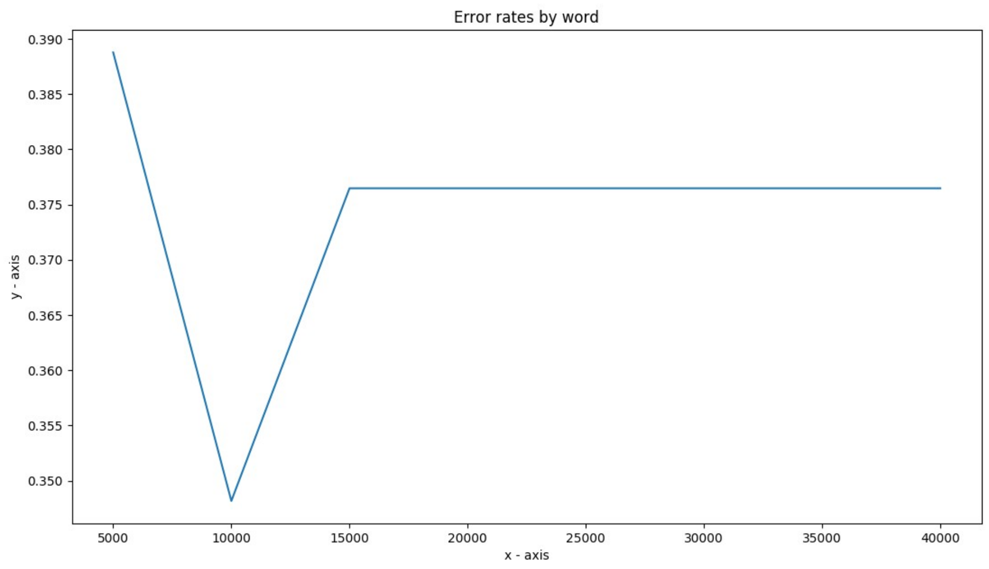

***Figure 8: Bulgarian Error Rate By Word using TrigramHmm***

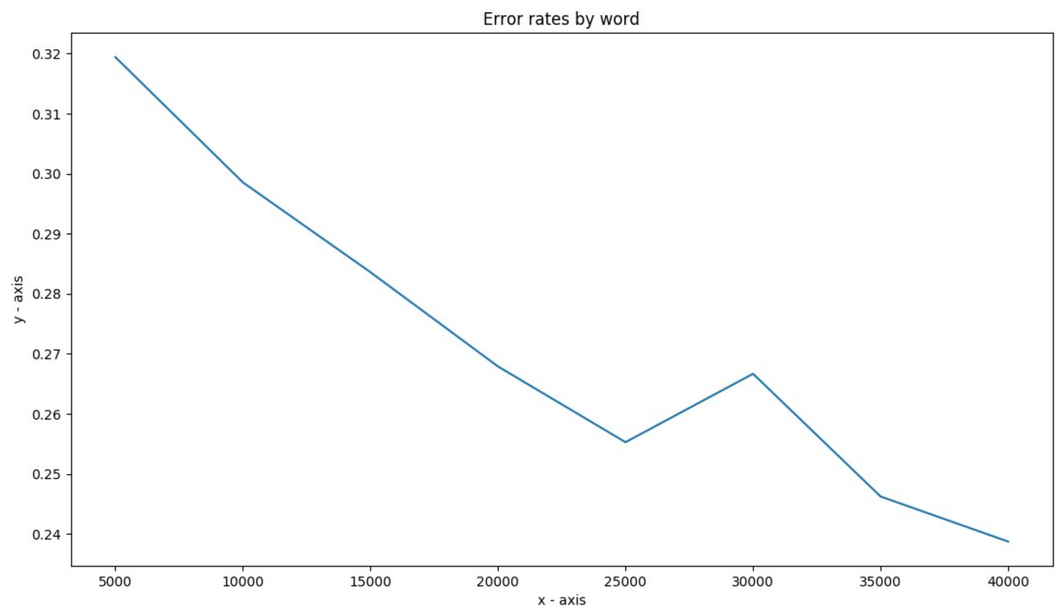

***Figure 9: Error by word for English Data Set using TrigramHMM***

From the above Figures: 7,8,9 error rates above shows for English, Japanese and Bulgarian the error rates drops to a minimum value and then starts increasing again which is different when taken into consideration Figures: 1,3,5 for which error rate always decreases. So in conclusion, the baseline model is good for the error rates but the my model as it shows increasing error rates after a particular point so is not considered to be good. The reason for this can be over fitting of the tags.

References::-
1.) https://stathwang.github.io/part-of-speech-tagging-with-trigram-hidden-markov-models-and-the-viterbi-algorithm.html