REPORT

TITLE: Perform optical character recognition, usage
USAGE: ./ocr.py train-image-file.png train-text.txt test-image-file.png
Training file used during assignment: Tweets.train.txt

AUTHORS: Apurva Gupta, Surbhi Paithankar, Hasika Mahtta
(based on skeleton code by D. Crandall, Oct 2017)

This program aims at recognizing characters in a given image using the concept of HMM.
We have implemented this using three techniques viz.
a. Simple Naive Bayes Algorithm
b. Variable Elimination (Forward-Backward algorithm)
c. Viterbi Algorithm.

Probabilities:
We calculated following probabilities required for implementation.
1. Initial Probability:
P(Letter L1) = Number of occurences of Letter L1 in training file/ total Number of letters
Note : If No. of occurences of Letter L1 is zero then P(Letter L1) = 10e-14

2. Transition Probability:
P(L2|L1) : No. of occurences where L2 succeeds L1 in training file/ No. of occurences of L1
Note : If No. of occurences of Letter L1 is zero then P(L2|L1) = 10e-14

3. Emission Probability:
We consider every pixel to be an observation O1...n.
So as per Naive bayes:
P(O1...n|Letter) = P(O1|Letter)P(O2|Letter)....P(On|Letter)

In order to calculate P(Oi|Li) we used:
If,a given pixel matches at a particular position:
P(Oi|Li) = 1+1/((14*25)+ no. of black pixels in training letter Li)
else:
P(Oi|Li) = 0+1/((14*25)+ no. black pixels in training letter Li)

Note: Since we are doing calculations at pixel level, the joint probabilities lead to underflow. Therefore we would be using logs for the implementation of simple & viterbi algorithm.
However in variable elimination, since we need to perform summation to eliminate a variable, we cannot use logs. This is because summation of terms in logs implies multiplication and not addition. Hence we multiply a scaling factor in the emission probability to handle the underflow.
SCALING FACTOR USED:
We used a scaling factor of log(10e308)+log(10e308)+log(10e272). We arrived at this value using the method of trial and error.

METHOD 1:

Simple Naive:
In this we consider every observation to be independent of other. In short, we did not take transition probabilities into account.
Formulae with log:
Argmin P(Letter Li|Observed 14*25 pixels) = -log(P(Observed 14*25 pixels|Letter Li)) + -log(P(Letter Li))

METHOD 2:
Variable Elimination:
In Variable elimination we take all other states into account and eliminate them by performing summation over all possible values.
Lets say We need to find P(Letter Li| Pixel observation O1....n) where n is number of letters in the image
Step 1:
Compute tow_table(alpha) using forward algorithm from first letter upto the desired letter. i.e L1...Li

Step 2:
Compute tow_table(Beta) using backward algorithm from last letter upto the desired letter. i.e Ln....Li

Step 3:
Finally we multiply alpha(L1...Li)*emission(Joint observation O1...n|Letter Li) * beta(Li....n)

METHOD 3:
Viterbi Algorithm:

Step 1:
In this we created a viterbi matrix (I*J) where i = number of letters in test set and j = number of training letters
vit(i,j) stores viterbi values and a backpointer to the previous state.
Formula :
vit(0,j) = P(Observed 14*25 pixels|Letter Li) * P(Letter Li)
vit(Li,j) = max(vit(Li-1,j) * P(Li|Li-1)) * P(Observed 14*25 pixels|Letter Li)

Step 2:
After creating the viterbi matrix, we begin with the last letter and traverse along the path till the first letter using the backpointer maintained.

ASSUMPTION:
1. Every letter has same font with size 14*25.

OBSERVATION:

1. Simple gives a good performance in guessing any word.
2. We observe that VE and Viterbi performs sometimes better than simple.
3. VE and Viterbi performs very well when the provided training file is very exhaustive enough. For example: When we used bc.train we saw it performing not very well,infact simple looked to perform much better. However if the training file is replaced with tweets.train.txt, we saw a significant improvement in results.
This may be because of the training file better information for building the transition proabability table.

LIMITATION

1. Due to underflow issue, we add some value to scale the emission probabilities, But sometimes it also cause overflow,when very big numbers get multiplied. We observed that after multiplication,values in tou table came out to be +infinity in some cases. Thus, overflow and underflow cause some issue in this regard.Out of 20 test images provided,one of them doesnt work properly because of overflow.
We tried to calculate emssion probability in some other way,to avoid this problem,But we observed that this way of calculating emission probability gives the best result.