COL 776 A1

Anupam Khandelwal (2013CS10212)

September 2016

1 Part 1 :

1.1 1(a) :

We do the following steps :

- Consider n nodes and max children value m. In this , first I generate number of children for each node using random number between 0 and m.
- Now , I sort nodes on basis of their no of children in descending order. I consider this order as the topological order of the graph.
- Let the no of children array be child[]. Now , for ith node in topological order , I chose child[i] nodes out of the left out nodes and make ith node their parent.
- Run using : ./run.sh <num-nodes> <max-chilren>

1.2 1(b) :

In the algorithm , we do following steps:

- First from Z , we mark all elements of Z and all their ancestors to be able to make a V-structure.
- We start BFS from X_i in outward direction from it. If we reach X_j from it, we say there exists an active trail between them and thus , they are not independent.
- For BFS , each element is in form of (Y, d) where Y is a node and d is direction (inward or outward edge). We push children to queue corresponding to the 4 cases of active trails (here , we use the above marked nodes to identify V-structure)
- To store active trail, I stored an extra array corresponding to the nodes which contains where the node was first visited from. I backtrack on this to find the active trail.
- Run using : ./run.sh <bn-filename> <query-filename>

1.3 1(c) :

Using my script, created output using sample-query.txt and sample-bn.txt and generated 5 images using display.py. The images are stored in the folder q1c.

2 Part 2 :

2.1 2(a) :

- Read file ocr.dat and stored ϕ_o in the form of 2D array of size images*all-chars($1000 * 256$).
- Read file trans.dat and stored ϕ_t in the form of 2D array of size all-chars*all-chars ($256 * 256$).
- Implementation allows all 3 modes.

2.2 2(b) :

- To calculate probability , I created a function getProb() which asks for given image , predicted word , mode of probability and want if bayesian network to be loaded.
- Probability is calculated by formula given in the question.
- Z is calculated by exhaustive summing over all combinations.

2.3 2(c) :

Max over all combinations is found using exhaustive checking using recursion. The accuracy parameters corresponding to all modes can be found in the table below :

Table 1: Accuracy Parameters vs Models

	Char Accuracy(%)	**Word Accuracy(%)**	**Avg Log Likelihood**
OCR Model	53.92	8.65	-7.487
Transition Model	66.27	25.96	-6.826
Combined Model	71.17	35.57	-6.043

We see that as we go from OCR to Transition to Combined Model, all the 3 accuracy parameters become better. Some examples of words getting corrected from OCR to Transition model are in following table :

Table 2: Correction from OCR to Transition Model

Original Word	OCR Model	Transition Model
ada	ads	ada
nesh	nhho	nesh
ortet	ohtet	ortet

Some examples of words getting corrected from OCR to Transition model and from Transition to Combined Model are in following table:

Table 3: Correction from OCR to Transition to Combined Model

Original Word	OCR Model	Transition Model	Combined Model
herne	hdrnd	herad	herne
noon	nssn	nson	noon
ratoon	raessn	rathon	ratoon

Here , we can see that from Transition to Combined model, words have replaced some letters with letters already present in the world. This is due to image value of both positions being same, so higher probability that both are same.

2.4 2(d) :

The tables for all 5 sets of images are as follows :

Table 4: allimages1 accuracy parameters

	Char Accuracy(%)	Word Accuracy(%)	Avg Log Likelihood
OCR Model	58.39	11.19	-7.553
Transition Model	68.04	24.04	-6.913
Combined Model	70.84	31.49	-6.043

Table 5: allimages2 accuracy parameters

	Char Accuracy(%)	Word Accuracy(%)	Avg Log Likelihood
OCR Model	57.25	10.01	-7.547
Transition Model	67.69	24.18	-6.903
Combined Model	70.72	31.81	-6.035

Table 6: allimages3 accuracy parameters

	Char Accuracy(%)	**Word Accuracy(%)**	**Avg Log Likelihood**
OCR Model	57.29	9.92	-7.548
Transition Model	67.87	24.68	-6.906
Combined Model	70.63	31.95	-6.039

Table 7: allimages4 accuracy parameters

	Char Accuracy(%)	**Word Accuracy(%)**	**Avg Log Likelihood**
OCR Model	57.58	11.47	-7.549
Transition Model	68.24	24.68	-6.912
Combined Model	70.77	31.86	-6.044

Table 8: allimages5 accuracy parameters

	Char Accuracy(%)	**Word Accuracy(%)**	**Avg Log Likelihood**
OCR Model	58.53	11.56	-7.541
Transition Model	68.46	26.69	-6.899
Combined Model	71.07	33.32	-6.033

The accuracies does not vary much as compared to small data file.

3 Extra Credit :

- We see that on increasing skip factor, avg log likelihood always increased , however , there was no change in char accuracy and word accuracy. This is because for all $img[i] = img[j]$, already $word[i] = word[j]$, even if skip factor was 2.
- Scaling up or down all ϕ_t or ϕ_o will have no effect as the factor will cancel out in Z. Neither will it change average log likelihood.
- Squaring ϕ_o increased confidence of false positives as transition and skip-factor models became weaker as compared to it. So, however average log likelihood increased, but accuracies became weaker.
- I changed skip factor to include a penalty and a boosting value :
  - if $img[i] = img[j]$ and $word[i]! = word[j]$ then $skip\_factor = 0.5$
  - if $img[i]! = img[j]$ and $word[i]! = word[j]$ then $skip\_factor = 2$

  This led to increase in all char accuracy , word accuracy and average log likelihood. Table for accuracy parameters over all datasets on new combined model :

Table 9: Accuracy parameters over new combined model

	Char Accuracy(%)	**Word Accuracy(%)**	**Avg Log Likelihood**
small-images	74.71	42.31	-5.462
allimages1	74.27	40.08	-5.436
allimages2	74.23	40.63	-5.425
allimages3	74.23	40.72	-5.426
allimages4	74.36	40.95	-5.434
allimages5	74.89	42.18	-5.422

Here , we see that accuracies have increased significantly and that too with better belief probabilities.