SAHIL JINDAL

2013CS10253

COL776 Assignment1

1.

(a) To create a random Bayesian network, total number of nodes (n) and maximum number of children a node can have are taken as system arguments. Bayesian network is stores as a list of nodes where each node contains its node_id, a list of its children node_ids, and a list of its parents node_ids. To generate the network, iterate over each node, select random number u between 1 and k and select u numbers randomly between 1 and n excluding current node. After generation of the network, it is written to the output file in the specified format.

(b) To find out whether x and y are d-separated, following algorithm is implemented:

- Ancestors of all the observed variables are added along with the observed variables to a set, say observed_ancestors. This is used for the notion that a trail is active at a V-structure node iff the node or any of its descendants are observed.
- Then BFS is implemented starting with nodes of BFS being a pair of node of Bayesian network and the direction to which that node need to be visited. Initially (x,↑) is inserted in the queue as we start finding the trail from x and see out-neighbors of x, i.e. neighbours x2 such that x→x2 in the bayes net.
- For a node which is to be visited by going out of the node, it should not be observed for the trail through it to be active, and its parents should also be visited by traversing out as it is the only way of reaching the current node from its parents directly, similarily its children should be visited by traversing down the child node. Hence the parents and children which are not already visited are added to the queue with appropriate directions.
- For a node to be visited by traversing down at that node, if the node is in observed_ancestors the trail will be active through its parents (V-Structure) with direction of parent nodes being out. If the node is not in observed ancestor, and not observed, the trail will be active through its children nodes by direction of the children being down.
- During this algorithm, an array prev_path is stored where prev_path[i] stores the node_id from which the current node was visited for the first time.

2.

(a) Model:

- Logarithm of OCR factors are stored in 2D array of size 10x100
- Logarithm of Transition factors are stored in 10x10 array
- A string variable used to get character from its index
- A map stored to convert a character to its index
- To have flexibility in changing modes, include_transition and include_skip variables are kept and taken as input from user which specify whether to include transition factor and skip factors respectively.

(b) To obtain the probability of an assignment of characters and image variables, the function subtracts log(normalizer) from the sum of log(factors) required of the assignment provided and then removing the log.

(c) For a given set of image variables, to obtain the best character assignment, the function iterates over all possible assignments of the characters possible and compare the sum of log(factors) for each assignment and retrieve the assignment with maximum sum, i.e. maximum probability as normalizer is same for every assignment. Normalizer calculation is also during this iteration by summing the product of the factors over each assignment possible and stored in a file so that we don't have to iterate over each iteration to obtain it again while calculating log-likelihood.

For small dataset, accuracy vs models table:

	OCR	OCR + Transition	OCR + Transition + Skip
Character Accuracy	53.92%	66.27%	71.17%
Word Accuracy	8.65%	25.96%	35.57%
Avg. Log Likelihood	-7.808	-7.097	-6.279

Some Words that were incorrect by OCR, and correctly fixed by transition model:

Correct word	OCR prediction	Transition prediction
arad	arae	arad
hent	ohnt	hent

Some Words that were incorrect in OCR, partially correct in transition and correct in combined model:

Correct word	OCR prediction	OCR + Transition	OCR+Transition+Skip
herne	hdrnd	herad	herne
torrid	tshhid	ishrid	torrid

(d) Tables for all 5 set of iamges:

Accuracy vs model for allimage1:

	OCR	OCR+Transition	OCR+Transition+Skip
Character Accuracy	58.39%	68.04%	70.83%
Word Accuracy	11.19%	24.04%	31.49%
Avg. Log-likelihood	-7.876	-7.175	-6.27

Accuracy vs model for allimage2:

	OCR	OCR+Transition	OCR+Transition+Skip
Character Accuracy	57.26%	67.69%	70.72%
Word Accuracy	10.01%	24.17%	31.81%
Avg. Log-likelihood	-7.874	-7.174	-6.271

Accuracy vs model for allimage3:

	OCR	OCR+Transition	OCR+Transition+Skip
Character Accuracy	57.25%	67.87%	70.63%
Word Accuracy	9.91%	24.68%	31.94%
Avg. Log-likelihood	-7.865	-7.167	-6.265

Accuracy vs model for allimage4:

	OCR	OCR+Transition	OCR+Transition+Skip
Character Accuracy	57.58%	68.24%	70.77%
Word Accuracy	11.47%	24.68%	31.85%
Avg. Log-likelihood	-7.869	-7.170	-6.267

Accuracy vs model for allimage5:

	OCR	OCR+Transition	OCR+Transition+Skip
Character Accuracy	58.53%	68.45%	71.06%
Word Accuracy	11.56%	26.69%	33.31%
Avg. Log-likelihood	-7.857	-7.158	-6.257