LING 570: Hw7
Due at 11pm on Nov 16
Total points: 100

All the example files are under dropbox/17-18/570/hw7/examples/.

Q1 (45 points): Write a script, **viterbi.sh**, that implements the Viterbi algorithm. You can reuse some functions from your check_hmm.sh in Hw6.

- The format is: viterbi.sh input_hmm test_file output_file
- The input_hmm is a state-emission hmm, which has the same format as the ones specified in Hw6. The output is produced by the to-states.
  - You can assume that the input_hmm does not contain any emission probability for empty string (i.e., a state cannot generate an empty string).
  - The output symbols are produced by the to-states.
  - Use input_hmm as it is. Do NOT try to smooth it. If there is no transition probability line from state s_i to s_j, that means that it is impossible to go from s_i to s_j. If there is no emission line for state s_j and output symbol w_k, that means that s_j cannot generate w_k.
  - For hw7, you don't need to check whether the three probability distributions (initial, transition, and emission ones) in the input_hmm satisfy the constraints that are checked by check_hmm.sh in Hw6. If a line contains a probability that is not in the [0, 1] range, your code just prints out a warning message to stderr ("warning: the prob is not in [0,1] range: $line", where $line is the line), and ignore those lines.
- The format of the test_file: each line is an observation (i.e., a sequence of output symbols). For POS tagging, an observation will be a sentence (cf. **test.word**):
  - The sentence in test_file may or may not include special symbol ($< /s >$) for EOS. If it does not include the symbol, do NOT insert that symbol because we want to make viterbi.sh work for any HMM, not just the one for n-gram taggers. In that case, $P(EOS \mid t_{n-1}t_n)$ will not be used when finding the best state sequence, but that is fine.
  - In other words, do NOT do anything special for BOS and EOS. Your code will work for any HMM, not just the one for ngram taggers.
- The format of the output_file (cf. **sys**): "observ => state_seq lgprob":
  - state_seq is the best state sequence for the observation.
  - lgprob is $lg\ P(observ, state\_seq)$; lg(x) is base-10 log.
- Your code should be able to handle unknown "word" in the observation: let the observation be "$o_1\ o_2\ ...\ o_n$". For each o_i, if o_i does not appear in the input_hmm at all, o_i is *unknown* and it (aka $< unk >$) can be generated by any state s_j with the probability $P(< unk > | s_j)$ larger than 0. If for some s_j, $P(< unk > | s_j)$ is zero or does not appear in the input_hmm, that means s_j cannot generate any unknown word.

Q2 (30 points): Run viterbi.sh on test data and convert the output to the correct format.

- You are given five hmm models named hmm[1-5] under the hw7/examples/ on patas. For the transition and emission probability lines, please ignore anything that is after ##.
- Write your own script, **conv_format.sh**, to convert the format of the output file of Step 2.
  - The command line is "cat file1 | conv_format.sh > file2".
  - file1 is the file created by Step 2, and file2 has the format "w1/t1 w2/t2 ... wn/tn".
  - Note that t_i is the second tag of the state that generates w_i.
- For each hmm file, do the following:
  1. Run viterbi.sh on test.word to produce an output file. The output file has the format "observ => state_seq lgprob", as explained in Q1.
  2. Run conv_format.sh to convert the format.
  3. Run calc_tagging_accuracy.pl (which is given to you) to calculate the tagging accuracy.
     - The format is: calc_tagging_accuracy.pl gold_standard sys_res > sys_res.acc
     - gold_standard and sys_res have the format "w1/t1 w2/t2 ... wn/tn" (e.g., **test.word_pos**).
     - The gold standard for the file *test.word* is *test.word_pos*, and the sys_res is the file created by conv_format.sh.
- Fill out Tablel 1.

For instance, to get the accuracy for the first row in Table 1, you should run the following commands (here, "1" in the file names refers to the number in the hmm file for that column):

- viterbi.sh hmm1 test.word sys1
- cat sys1 | conv_format.sh > sys1_res
- calc_tagging_accuracy.pl test.word_pos sys1_res > sys1_res.acc 2>&1

Table 1: Tagging accuracy

HMM model	tagging accuracy
hmm1	
hmm2	
hmm3	
hmm4	
hmm5	

The submission should include:

- readme.[txt|pdf] that includes Table 1.
- hw.tar.gz that includes the files specified in submit-file-list.