Midterm-F08upload-answers Madhav

Natural Language Processing (Pandit Deendayal Petroleum University)

COMS W4705x: Natural Language Processing
MIDTERM EXAM
October 21st, 2008

DIRECTIONS

This exam is closed book and closed notes. It consists of three parts. Each part is labeled with the amount of time you should expect to spend on it. If you are spending too much time, skip it and go on to the next section, coming back if you have time.

The first part is short answer. The second part is problem solving. The third part is essay.

Important: Answer Part I put answers on the test sheets and turn in the test itself. Answer Part II using a separate blue book for each problem. Answer Part III on the test itself. In other words, you should be handing in the test and at least four blue books.

Part I – Short Answer. 25 points total. 15 minutes.

Provide 2 or 3 sentences for *five out of the following seven* questions. Each question is worth 5 points.

NOTE: Choose FIVE.

1. State the difference between inflectional and derivational morphology, noting their relation with word class.

Inflectional morphology is when endings are added to indicate grammatical form and function. For example, they may be used to indicate a grammatical function like plural. The word class is the same both before and after the ending is added. Derivational morphology adds suffixes, affixes or infixes to a stem to create a new word with new meaning. In derivational morphology, the class changes after morphemes are added.

2. What is the difference between a FSA and a FST?

An FSA is a graph representing a regular expression that recognizes strings that are part of the language. An FST is a finite state transducer and it both recognizes and generates output. It can be used to parse the input or to transform it into another expression.

3. How would you build a dictionary for the most frequent POS tagging algorithm?

The most frequent POS tagger always tags a word with the most frequent tag that it is used with. To create a dictionary, therefore, for each word, we want to enter the frequency for each POS tag.

We will use a corpus that has been annotated by people with the correct POS for each word. For each word, we will count the number of times each POS is used. We will record the most frequent one in the dictionary.

4. Give the FOPC representation for the following two sentences, providing two representations if it is ambiguous, one representation if it is not and describing any problems that FOPC might have.

John asked every girl a question:

For-all x, there-exists y, there-esits e is-a(x, girl) and is-a (y, question) and is-a(e, asking) and e(John, x, y)

There-exists yfor-all X, there-exists e s.t. is-a(x, girl) and is-a(y, question) and is-a(e, asking(and e(john, x, y)

Someone in 4705 likes language:

There-exists x, there-esists e s.t. is-a(x person) and in (x,4705) and is-a (e.,liking) and e(x,language)

Most students are happy at the end of the semester:

This one is problematic in FOPC because there is no quantifier for "most". Should be:

Most x, s.t. there-exists s s.t. is-a(s, happiness) and s(x) and time(s, end of semester)

5. Give examples of three different types of structural ambiguities and say why they are ambiguous.

PP attachment
John ate the fish from a paper box. Did he eat from a paper box? Or, did he choose the fish from the paper box?

Coordination
John likes red cars and toys. Are the toys red?

NP bracketing
Spanish language teachers: Do the teachers teach Spanish or do they speak Spanish?

6. Assuming the grammar below, show the parse tree for the sentence *The big yellow dog sat under the house.*

S-> NP VP
VP -> VP PP
VP -> verb NP
VP -> verb

NP -> DET NOM
NOM -> ADJ NOM
NOM -> NOUN
PP -> PREP NP
DET -> the
ADJ -> big
ADJ -> yellow
NOUN -> dog
VERB -> sat
PREP -> under
NOUN -> house

(S ((NP (DET the) (NOM ((ADJ big) (NOM ((ADJ yellow) (NOM (NOUN dog))))))) (VP ((VERB sat) (PP ((PREP under)(NP ((DET the) (NOM (NOUN house)))))))

7. Show how you would have to modify the grammar above to handle the sentence *The dog in the white hat ran under the house.*

NOM -> NOM PP

Part II. Problem Solving. 45 points. 40 minutes.
There are three problems in this section worth 15 points each. Do all 3 problems. Use a separate blue book for each problem.

1. **Finite State Transducers**. Eliza was a program developed to answer questions as a psychologist would. For this question, you are asked to write an FST that will respond to questions.
   a. Write an FST that will respond as in the following dialog. Make sure that your FST will handle other similar dialogs but with different input statements and give an example of a similar dialog it can handle.

   Input: I am very unhappy.
   Eliza: Why are you very unhappy?
   Input: My girlfriend thinks I'm mean
   Eliza: And are you mean?

   b. Now create an FST to handle the input "*I am very unhappy*": as well as the two paraphrases: 1. ``*I am not very happy*'' 2. ``*I am just not happy*''

   c. Create a new FST to handle the input with ``I am sorry to hear that you… ''. Show an example of input and output.

   Refer to the handwritten (scanned) part at and of document.

2. **Context Free Grammar:** You are given the grammar below. Show how it would be used to derive a parse tree for the sentence below. Show the order in which rules would be

applied if using a top down search strategy and the order in which the rules would be applied if using a bottom-up grammar. Identify when disambiguation would occur, all partial structures that would be built and the parse tree that would be returned.

The complex houses first-year students.

S -> NP VP
NP -> DET NOM
NOM -> ADJ NOM
NOM -> NOUN
VP -> VERB NP
VP -> VERB
DET -> the
ADJ -> complex
ADJ -> first-year
NOUN -> complex
NOUN -> first-year
NOUN -> houses
NOUN -> students
VERB -> houses

Refer to the handwritten (scanned) part at end of exam.

3. **Hidden Markov Models:** You are given the sentence below and the tables of probabilities show in Table 3a (this page) and Table 3b (next page).

I promise to back the bill.

a. Describe how the probabilities would be obtained using the Penn Treebank.

Two sets of probabilities would need to be determined. To get probabilities for finding a word being a specific POS (Table 3a), one would need to count the total number of times the word occurs with that specific POS divided by the total number of times the POS occurs in the corpus: Count (W with POS-tag)/ Count (POS-tag).

For the second table, one would need to compute the probability of each tag followed by another tag. To do this, one would count the number of times POS-I is seen followed by POS-J, normalized by the count of the first POS tag. So: Count (POS-I POS-J) / POS-I

b. A hidden markov model includes states, observations, transition probabilities, observation likelihoods. Describe what each one of these would correspond to when using an HMM for POS tagging.

States: The POS tags at specific points in the sentence.
Observations: The words that are observed as the sentence is read in.

Transition probabilities: the probability of finding POS tag N following POS tag N-1
Observation likelihoods: the probability of seeing a particular word

c. Given the sentence ``I *promise to back the bill*.'' show how you would compute the probability of ``*back*'' as a verb versus the probability of ``*back*'' as a noun using the probabilities in Tables 3a and 3b using the Viterbi algorithm. You are given the values for the third column of the Viterbi table which correspond to observation 3 or ``to''. They are VB: 0, TO: .00000018, NN: 0, PPSS: 0. Thus, you will show two computations both of which will use these values. You do not need to do the arithmetic; just show the formula that would be computed.

Back as a verb:

.00000018 * Prob(VB| TO) * Prob (VB| back) = .00000018* .00008*.83

Back as a noun:

.00000018* Prob (NN | TO) * Prob (NN | back) = .00000018* .00068 * .00047

	I	***promise***	***To***	***back***
VB	0	.0093	0	.00008
TO	0	0	.99	0
NN	0	.0085	0	.00068
PPSS	.37	0	0	0

Table 3a: Observation Likelihoods

	VB	**TO**	**NN**	**PPSS**
<s>	.019	.0043	.041	.067
VB	.0038	.035	.047	.0070
TO	.83	0	.00047	0
NN	.0040	.016	.087	.0045
PPSS	.23	.00079	.0012	.00014

Table 3b: Tag transition probabilities. The rows are labeled with the conditioning event. Thus, P(VB|<s>) = .019.

Part III. 30 points. 20 minutes.
Essay. Answer *2 out of the following 4 questions*, worth 15 points each. Use no more than 2 paragraphs for each question. Put your answers in one blue test book.

NOTE: CHOOSE 2

1. This question concerns the rule-based POS tagger, ENGTWOL, and Brill's transformational POS tagger, TBL. Give a brief description of how each tagger works and then state one similarity and one difference between the two.

ENGTWOL: ENGTWOL begins with a dictionary and assigns every POS associated with a word in the dictionary to words in the text to be tagged. For example, in the sentence "time flies like an arrow." "Time" would be tagged as both a noun and a verb. Then ENGTWOL uses a set of rules, written by hand, that remove tags that don't fit in the given context. For example, one rule eliminates VBN (past particple) if VBD is an option when the verb follows a pronoun, which in turn occurs at the beginning of the sentence.

TBL: the tagger begins by tagging every word with its most frequent tag (e.g., "time" is most commonly used as a noun). It then learns a set of rules that can be applied to change the tag to the correct one in context. The rules are learned over training data consisting of a corpus that has been manually labeled with POS tags. A set of rule templates are used to initiate the process using multiple iterations. At each iteration, rule transformations are generated, scored against the corpus and high-scoring rules are retained. The process is repeated until a threshold is reached. The learned rules are then used on unseen data to do the tagging.

Similarity: Both use rules to do part-of-speech tagging.
Difference: Brill's TBL learns the rules automatically while in ENGTWOL the rules are written by hand.

2. In each of the following sentences, identify the semantic roles selecting from *agent, patient, theme, experiencer, stimulus, goal, recipient, benefactive, source, instrument, location, temporal*. Justify your choice.

The company wrote me a letter.
Agent: the company, recipient: me, patient: letter
Jack opened the lock with a paper clip.
Agent: Jack, patient: lock, instrument: paper clip
The river froze during the night
Theme: river, temporal: night
Kathy ran to class every day at Columbia.
Agent: Kathy, goal: class, temporal: every day, location: Columbia
I felt the warmth from the fire.

Experiencer: I, stimulus: warmth, source: the fire

Note: you should also have provided a justification. If the justification was good and the answer was different from what was here, you did not lose a point. If no justifications were provided, 1 point overall was deducted.

3. Consider the sentences *President George Bush has re-invigorated the economy by providing a bail-out program for failing Wall Street firms*. and *President George Bush has caused a disastrous economic situation by failing to provide regulations on Wall Street firms.* You'd like to compute the likelihood of these sentences given a corpus of NY Times, Wall Street Journal and the New York Post gathered over the last year. You develop a bi-gram language model. Describe how you would: 1. Build the language model, 2. Compute the likelihood of these sentences and 3. Evaluate your language model.

I would build the language model by computing the probability of each bigram (pair of words) in the corpus. To do this, I would count the frequency of each pair of words w1 and w2 and I would then normalize by the count of the first word in the bigram: Count(w1 w2)/ Count(w1). I would do this for each pair of words that appears consecutively in the corpus. Once I had the language model, I would compute the likelihood of each sentence by determining the probability of each bigram in the sentence (including the probability of start followed by word-1) and multiplying. So, for eaxmaple, Prob(start President) * (President George) * Prob (George Bush) * Prob (Bush has).. etc. This must be done for both sentences.

To evaluate the language model, a distinction between training and test set would need to be established, with construction of the model on the training set and evaluation on the test set. In addition, we need a metric which can measure how good the model is. Perplexity is the standard method to measure the goodness of a language model. Typically, one would compare one language model against another by measuring the perplexity of each one on the test data. The model with higher perplexity is the better one.

4. Describe how probabilities would be computed from a corpus for the rule VP -> verb NP PP in a PCFG grammar. How is the probability of a parse tree computed in a PCFG? Contrast this with the computation of probabilities from a corpus for rules capturing lexical dependencies that might be used to disambiguate sentences such as *The woman dropped the stamped letter into the mailbox*. and *The woman opened the letter from a friend.* How would disambiguation occur?

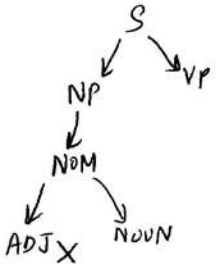
PCFG: probabilities would be counted using the Penn Treebank. We would count the total number of times we find a sub-tree headed by VP with verb and NP as the children and normalize that by the total number of times we find a sub-tree headed by VP with anything as children. The probability of a parse tree is computed by multiplying the probabilities of each rule in its derivation. In contrast, to compute lexical dependences, we would want the probabilities to distinguish whether the prepositional phrase was an argument to the verb or not. For example, for the verb “drop” we want a VP rule like VP -> V NP PP to apply, while for “open”, we want the rule VP -> V NP to apply. To do this, we condition the VP on its lexical head. Thus, we would have a rule VP (dropped) -> V NP PP. To get the counts for lexicalized rules like this, we would count how many times this rule used with (head) dump, divided by the number of VPs that dump appears (as head) in total. Disambiguation for the two examples involves determining PP attachment. This would occur probabilistic, but choosing the VP -> V NP PP because it was more likely than the rule VP -> V NP. Note that, if the latter rule is chosen, then the PP would be attached to the NP.

Scanned parts of the solution:

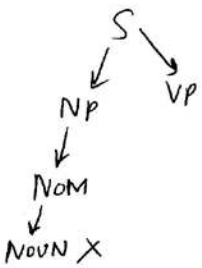
PART II 2) a) TOP-DOWN PARSING

(Drawing partial structures while assuming that whenever more than 1 rule is applicable, the wrong one is applied first)

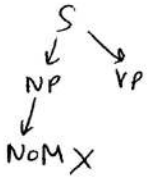
① S → NP VP
NP → NOM
NOM → ADJ NOUN
BACKTRACK!



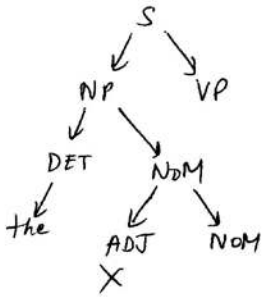
② S → NP VP
NP → NOM
NOM → NOUN
BACKTRACK!



③ S → NP VP
NP → NOM
BACKTRACK!



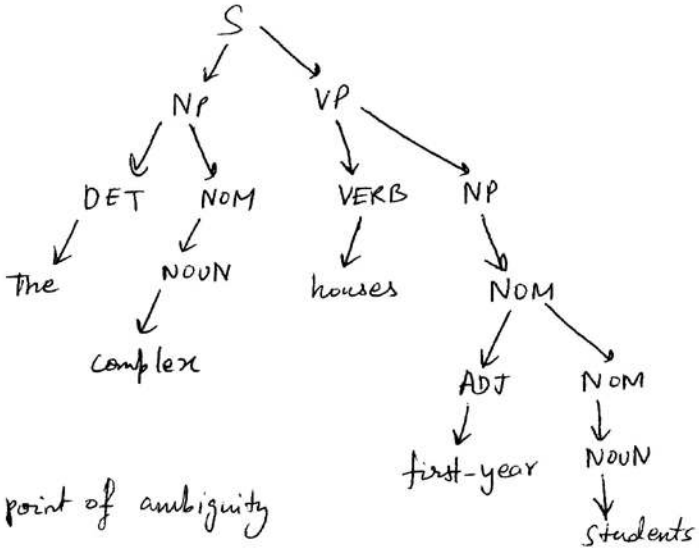
④ S → NP VP
NP → DET NOM
DET → the
NOM → ADJ NOM
BACKTRACK!



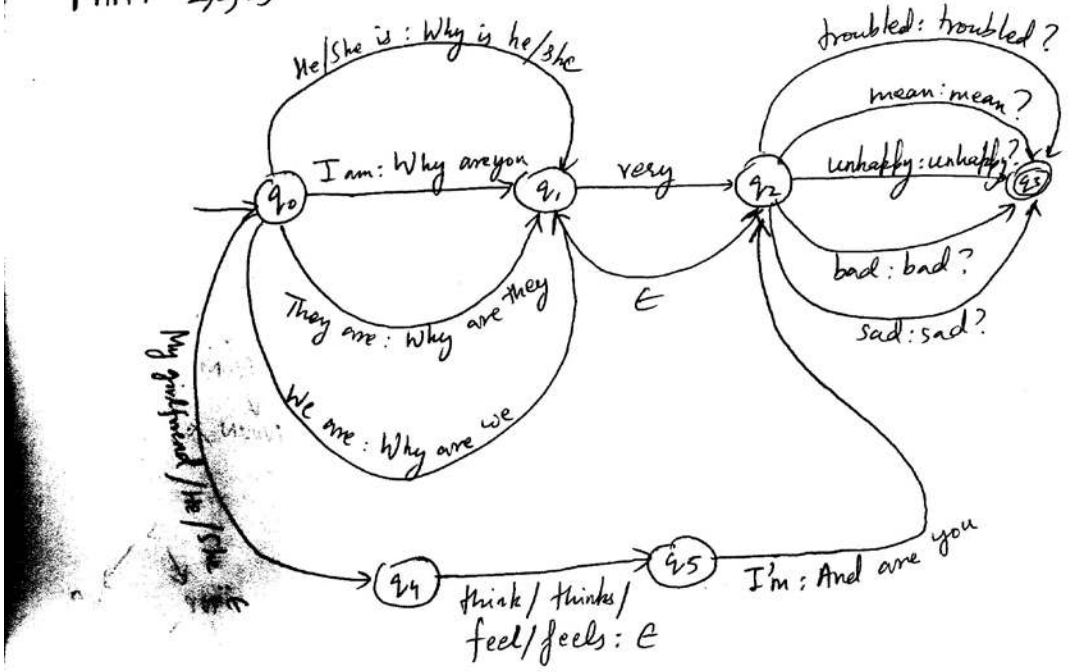
And so on.....

FINAL TREE:

S → NP VP
NP → DET NOM
DET → the
NOM → NOUN
NOUN → complex
VP → VERB NP
VERB → houses
NP → NOM
NOM → ADJ NOM
ADJ → first-year → point of ambiguity
NOM → NOUN
NOUN → students



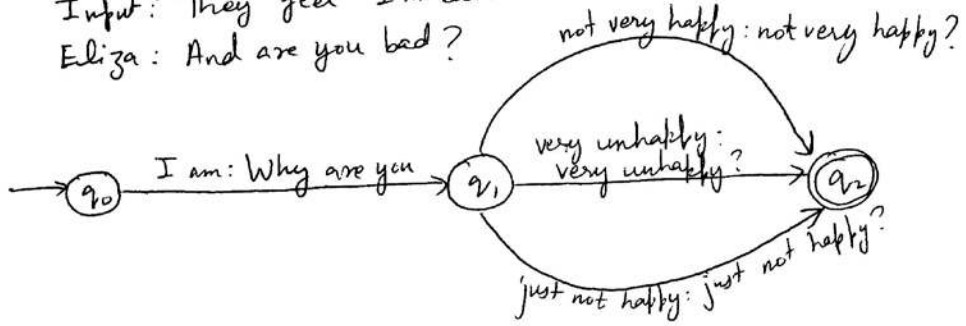
PART 2)1)a)



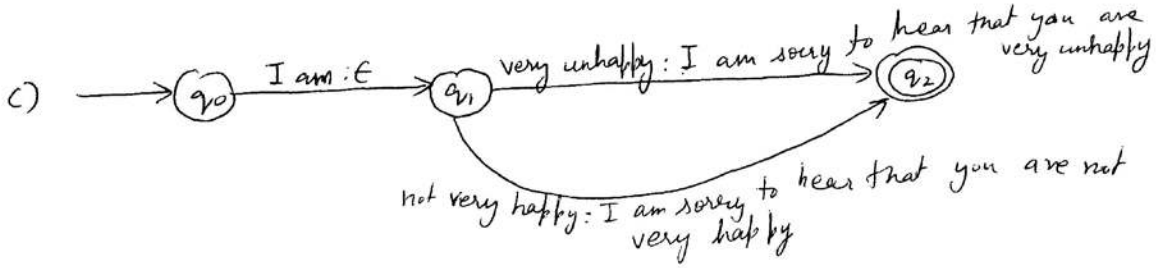
Examples:

Input: He is very sad
Eliza: Why is he very sad?
Input: They feel I'm bad.
Eliza: And are you bad?

b)

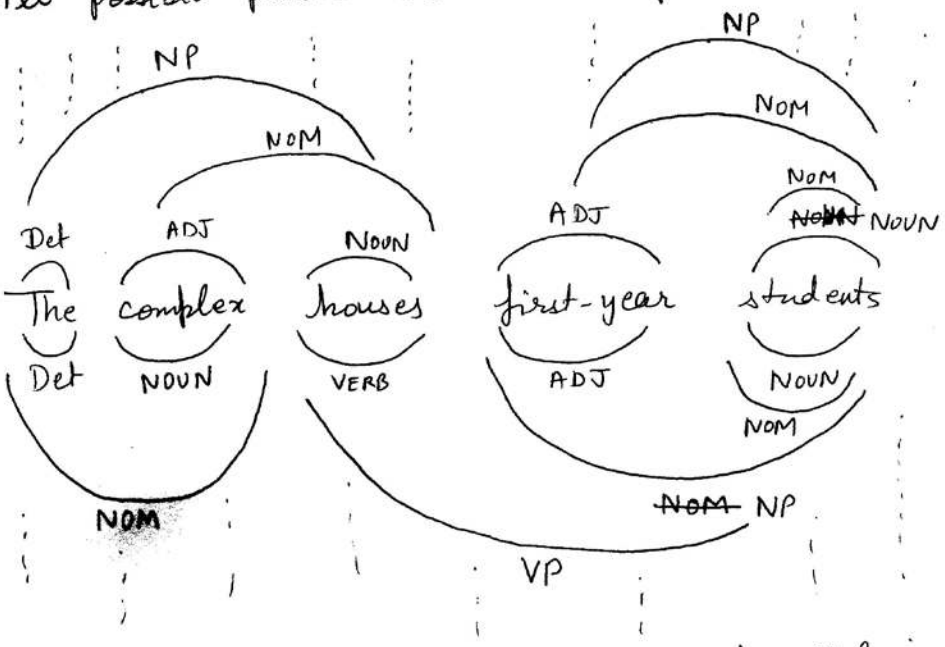


c)



PART II 2) b) BOTTOM-UP PARSING

All possible parses are built in parallel



One of all the parse trees built in parallel is the correct one.