**COMS W4705x: Natural Language Processing**

**MIDTERM EXAM**

**October 27th, 2009**

# 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 multiple choice. The second part is short answer. The third part is problem solving.

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

**Part I – Multiple Choice. 20 points total. 15 Minutes.**

1. In an Early parser, a table is filled. Which of the following does *not* appear as a table entry?
	1. Completed constituents and their location
	2. Predicted constituents
	3. Word predictions
	4. In-progress constituents

Answer: c

1. In an HMM, tag transition probabilities measure
	1. The likelihood of a POS tag given a word
	2. The likelihood of a POS tag given the preceding tag
	3. The likelihood of a word given a POS tag
	4. The likelihood of a POS tag given all preceding tags

	Answer: b
2. In an HMM, observation likelihoods measure
	1. The likelihood of a POS tag given a word
	2. The likelihood of a POS tag given the preceding tag
	3. The likelihood of a word given a POS tag
	4. The likelihood of a POS tag given two preceding tags

Answer: c

1. To compute the likelihood of a sentence using a bigram model, you would:
	1. Calculate the conditional probability of each word given all preceding words in a sentence and multiply the resulting numbers
	2. Calculate the conditional probability of each word given all preceding words in a sentence and add the resulting numbers
	3. Calculate the conditional probability of each word in the sentence given the preceding word and multiply the resulting numbers
	4. Calculate the conditional probability of each word in the sentence given the preceding word and add the resulting numbers

Answer: c

1. When training a language model, if we use an overly narrow corpus, the probabilities
	1. Don’t reflect the task
	2. Reflect all possible wordings
	3. Reflect intuition
	4. Don’t generalize

	Answer: d
2. Morphotactics is a model of
	1. Spelling modifications that may occur during affixation
	2. How and which morphemes can be affixed to a stem
	3. All affixes in the English language
	4. Ngrams of affixes and stems

	Answer: b
3. N-fold cross-validation is a technique for evaluation that uses
	1. All data in the corpus for training
	2. 10% of the data in the corpus for training
	3. 90% of the data in the corpus for training
	4. 50% of the data in the corpus for training

	Answer: a
4. In the sentence *John sold Mary the book*, the theme of the sentence is:
	1. John
	2. Mary
	3. Book
	4. Sold

	answer: c
5. Indefinite determiners in language, such as “*a*”, are represented in first-order predicate logic by
	1. Connectives
	2. Variables
	3. Universal quantification
	4. Existential quantification

	Answer: d
6. In a dependency parse
	1. All nodes are labeled with words
	2. Only leaf nodes are labeled with words
	3. Only non-terminal nodes are labeled with words
	4. No words appear in the tree

Answer: a

**Part II** **– Short Answer. 28 points total. 20 minutes.**

**Provide 2 or 3 sentences for *four out of the following six* questions. Each question is worth 7 points.**

**NOTE: Choose FOUR. If you answer more than four, you will be graded on the first four answers you provide.**

**1.** You are an English teacher and you ask your class to write a play in the style of Shakespeare. You want to score their plays using a trigram language model you computed from a corpus of all Shakespeare plays but you find that the data is too sparse and most of your students’ sentences receive a score of zero. How would you use a back-off model to alleviate this problem? How about class-based smoothing?

**Answer:** In a back-off model, every time you had a zero count for an n-gram, you would back-off and use the counts for the bigrams instead. If the bigram was 0 also, you could back off to unigram counts. In class-based smoothing, you would re-label words or phrases with a class instead, such as NP or PERSON-NAME. So, you could count the number of times PERSON-NAMES appeared instead of individual names.

2 pt each for knowing what the back-off model is

1.5 pt each for saying how they would use it

**2.** Provide the four steps used in the Brill Transformation Based Learning algorithm to learn transformations.

**Answer:**

1. Tag all words using the most frequent tag
2. Select the transformation that gives the biggest drop in error against tagged corpus
3. Retag all words using new set of rules
4. Repeat steps two until error drops below some threshold

Note: Fine if they used either “drop in error” or “gain in accuracy”

2 pt for each step and 1 pt for the repeat.

**3.** For a compositional semantics model, show the semantics you would attach to the rule S-> NP VP, the semantic entry for the verb “sleep” and the semantic analysis that would result for the sentence “John slept” assuming that the semantics for the NP “John” is “JOHN1”.

Answer:

Sleep: λx Эe is-a(e, sleeping) and e(x)

Semantics for S-> NP VP: Sem(VP)(Sem(NP))

Semantic analysis of sentence: Эe is-a(e, sleeping) and e(JOHN1)

**2 pt each for first two and 3 pt for the third**

**4.** State the difference between homonymy and polysemy and give an example of each.

Homonymy refers to two words that either sound the same or are spelled the same but have very different meanings. Bank is an example of a homonym because it can mean either river bank or financial institution. Polysemy also refers to words that either sound the same or are spelled the same, but these words have different but closely related meanings. An example would be bass, where two closely related meanings would be the lowest part in choral music or a man who sings the lowest part of a choral music.

2 pt for each example

3 pt for stating the difference

**5.** What is the difference between a top-down, recursive descent parsing strategy and a bottom-up, parallel parsing strategy? After which word in the sentence *The old dog the footsteps of the young* would disambiguation occur in each case?

**Answer:** A top-down parsing strategy will start by expanding roots at the top level and will keep expanding the left-most node in the tree until it reaches the leafs. If it gets to the leaf and no word matches, then it must back up. A bottom-up parallel parsing strategy is data-driven. It expands all rules whose right hand sides match the words in the words in the sentence. It then expands all rules whose right hand sides match the constituents that were added to the chart. It pursues many parses in parallel. The top-down strategy will disambiguate the sentence after reading the word “the” after “dog”. It is looking for a verb, but none can be found. The bottom-up strategy will disambiguate only after it reaches the end of the sentence.

4 pt for difference and 3 pt for saying where disambiguation would occur

**6.** Draw a FSA that will accept the names of all people who have a first name beginning with K and a last name beginning with M, any number of middle initials, and the titles Professor, Prof. Dr. and Ms.

Titles: 2 pt

First name: 1.5 pt

Last name 1.5 pt

Initial: 2 pt

**Part III. Problem Solving. 52 points. 40 minutes.**

**There are two problems in this section. Do both problems. Use a separate blue book for each problem.**

**2. [22 points total] Syntax.** Consider the following grammar:

|  |  |  |
| --- | --- | --- |
| **S ->NP VP****VP -> Verb NP****VP -> Verb PP****VP ->VP PP****NP ->NP PP****NP -> NP and NP****PP-> P NP** | **NP->Kathy****NP->Connecticut****NP->Massachusetts****NP->November** | **Verb->drove****P->to****P->in****CONJ -> and** |

1. [12 points] Show three parse trees that would be derived for the sentence *Kathy drove to Connecticut and Massachusetts in November* as well as what the first chart entry made by an Early parser would look like after “Kathy” was read.

3 pt for each parse tree, 3 pt for first chart entry (should just be chart-0. If they did chart-1 also no problem)

Tree 1:

S-> NP VP

NP -> Kathy

VP -> Verb PP

Verb -> drove

PP -> P NP

P -> to

NP -> NP and NP

NP -> Connecticutt

NP -> NP PP

NP -> Massachusetts

PP -> P NP

P .-> in

NP -> November

Tree 2:

NP -> NP PP

NP -> NP and NP

NP -> Connecticut

NP -> Massachusetts

PP -> P NP

P -> in

NP -> November

Tree 3:

VP -> VP PP

PP -> P NP

P -> in

NP -> November

VP -> verb PP

verb -> drove

PP -> P NP

P -> to

NP -> NP and NP

NP -> Connecticutt

NP -> Massachusetts

Early parser first chart entry:

Chart0:

S-> .NP VP 0,0 Predictor

Chart 1:

NP -> Kathy. 0, 1 Scanner

S -> NP.VP

B. [5 points] Given a treebank how would you determine probabilities for the grammar rules given in Question 1 (for use with a basic PCFG parser)?

Let’s take the VP rule. There are three VP rules. I would count the total number of VP rules in the Treebank. Then, for each rule, I would count the number of times that rule occurs and divide by the total number of VP rules. That would yield the probability for each rule. I would follow a similar procedure for each rule where the same non-terminal appeared on the left-hand side.

C. [5 points] To improve performance on sentences like the example in Question 1, advanced probabilistic parsers make use of probability estimates other than those based on grammar rules alone, taking words into account. Describe one such probability estimate that might have helped with determining the best parse of part a.

For the rule VP -> VP PP I would add lexical dependencies, so I would get rules that looked like this:

VP -> VP (drove) PP (in),

VP -> VP (drove) PP (to),

I would do the same for the NP-> NP PP getting rules like

NP-> NP (Massachusetts) PP (in)

NP -> NP (Massachusetts) PP (in)

And so forth. Then I would calculate the probabilities associated with such rules by counting the number of times, for example, that drove occurs as a VP with the modifier PP headed by “to” following it and divide that by the total number of times that the drove occurs as a VP at all. Similarly, for the NP rules, I would count the number of times that the NP headed by Massachusetts occurs with a PP headed by in as a modifier and I would divide that by the number of times that the NP Massachusetts occurs at all.

1. **[30 points]** **Word Sense Disambiguation**.

Consider the word “jam”. Three different senses drawn from WordNet, along with their glosses and examples are shown in Fig. 1 below. Fig. 2 (next page) shows three excerpts from the Brown corpus illustrating the different meanings of “jam.”

In class we saw a statistical method using Naïve Bayes for word sense disambiguation and an algorithm that is called “Simple Lesk.”

1. [10 points] For disambiguating “jam” show the feature vector for Excerpt #1 that would be used if you were using collocational features with a window size of + or - 2. Show the feature vector for Excerpt #1 that would be used if you were using a bag of words approach with a window size of + or – 10.

Collocational feature vector:

Word-2 POS-2 Word-1 POS-1 Word+1 POS+1 Word+2 POS+2

incipient JJ traffic N ahead ADV . Punctuation

Bag of words feature vector:

A 1

 of 1

a 1

four 1

freeway 1

spies 1

 an 1

incipient 1

traffic 2

ahead 1

in 1

the 1

next 1

lane 2

appears 1

 to 1

be 1

moving 1

more 1

smoothly 1

1. [10 points] Show how you would calculate the likelihood of “jam” as jam-1, jam-2, or jam-3 in the following sentence following a Naïve Bayes approach with collocational features and a window size of +-2, using the excerpts in Fig. 2 as your corpus. Show the formula you would use.

*Every morning for breakfast I have an egg and a slice of toast with a tablespoon of jam*

Formula: argmax P(s) \* P(F|s) (for each F)

 Over all s

Jam-1: P(jam-1) \* P(tablespoon-2|jam1) \* P(noun-2|jam-1). P(jam-1) is 2/5 but no point in working this out any further since P(tablespoon|jam1) = 0 and therefore, the probability of this sense is 0.

Jam-2 P(jam-2)= 1/5. Again P(tabspoon-2|jam2) is 0 and thus, probability of jam-2 is 0.

For Jam-3:

P(jam3) = 2/5

2/5\*P(tablespoon-2|jam3)\*P(noun|jam3)\*P(of-1|jam3)\*P(Prep-1|jam3)\*P(.|jam3)

Probability of words before “jam” is .5 in each position. Probability of POS is 1 in each position, but probability of having a period after “jam” is 0 in the corpus, so again we have a total of 0.

2.5\*.5\*1\*.5\*1\*.0

No sense is selected unless some sort of smoothing is used.

1. [10 points] Show how you would disambiguate the same sentence using the Simple Lesk approach. Describe the algorithm and show how it would apply in this instance.

For Simple Lesk, we would count the number of words in the sentence that overlap with the words in the gloss or example. For sense-1, there are only overlapping words with the gloss or example are “a” and “for” for a count of 2. For sense-2, the overlapping words are “an” for a count of 1. For sense-3, the overlapping words are “a” and “toast” and “and” for a count of 3. Sense-3 is selected.

*Every morning for breakfast I have an egg and a slice of toast with a tablespoon of jam*

**Fig. 1:**

**Sense jam-1**

**Gloss:** a crowded mass that impedes or blocks <a traffic jam>

Example: Trucks sat in a jam for ten hours waiting to cross the bridge.

**Sense jam-2**

Gloss: an often impromptu performance by a group especially of jazz musicians that is characterized by improvisation

Example: The saxophone players took part in a free-form jazz jam

**Sense jam-3**

Gloss: a food made by boiling fruit and sugar to a thick consistency

Example: He spread home-made jam on his toast.

**Fig. 2:**

**Sense jam-1**

**Excerpt #1**

Since moving from a Chicago suburb to Southern California a few months ago, I've been introduced to a new game called Lanesmanship. Played mostly on the freeways around Los Angeles, it goes like this: A driver cruising easily at 70 m.p.h. in Lane ~A of a four-lane freeway spies an incipient traffic ***jam*** ahead. Traffic in the next lane appears to be moving more smoothly so he pokes a tentative fender into Lane ~B, which is heavily populated by cars also moving at 70 m.p.h..

**Excerpt #2**

You don't need concentration. If you cut down these horrible buildings you'll have no more traffic ***jams***. You'll have trees again.

**Sense jam-2**

**Excerpt #3.**

The angriest young man in Newport last night was at the Playhouse, where "Epitaph for George Dillon" opened as the jazz festival closed. For the hero of this work by John Osborne and Anthony Creighton is a chap embittered by more than the lack of beer during a ***jam*** session. He's mad at a world he did not make.

**Sense jam-3**

**Excerpt #4**

Peel, core, and slice across enough apples to make a dome in the pie tin, and set aside. In a saucepan put sufficient water to cover them, an equal amount of sugar, a sliced lemon, a tablespoon of ***jam*** or apricot preserve or, a pinch each of clove and nutmeg, and a large bay leaf. Let this boil gently for twenty minutes, then strain.

**Excerpt #5**

He poured the water off the sourdough and off the flour, salvaging the chunky, watery messes for biscuits of a sort. Their ***jams*** and jellies had not suffered. He found a jar of preserved tomatoes and one of eggs that they had meant to save.