COMS4705 Midterm Type B

Jing Qian

TOTAL POINTS

80 / 116

QUESTION 1

Multiple choice 30 pts

1.1 naive bayes 3 / 3

- **- 1 pts** A
- ✓ **+ 3 pts** **Correct (B)**
- **- 1 pts** C
- **- 1 pts** D
- **+ 0 pts** We asked you to circle your answers! :(

1.2 probability of sentence 3 / 3

- ✓ **+ 3 pts** **Question dropped.**

1.3 dependency projective 3 / 3

- ✓ **+ 3 pts** **A (correct)**
- **- 1 pts** B
- **- 1 pts** C
- **- 1 pts** D
- **+ 0 pts** No answer

1.4 perplexity 3 / 3

- ✓ **+ 3 pts** **A, or both A and D (correct)**
- **- 1 pts** B
- **- 1 pts** C
- **+ 1.5 pts** D only
- **- 3 pts** No answer selected

1.5 distributional hypothesis 3 / 3

- ✓ **- 0 pts** **Correct (C)**
- **- 3 pts** Incorrect
- **- 3 pts** No answer

1.6 smoothing 0 / 3

- **- 0 pts** Correct (D)
- ✓ **- 3 pts** **Incorrect**
- **- 1 pts** Correct answer and one incorrect option selected
- **- 3 pts** No answer

1.7 discriminative classifier 3 / 3

- ✓ **- 0 pts** **Correct (B)**
- **- 1 pts** Correct answer and one incorrect option selected
- **- 3 pts** Incorrect
- **- 3 pts** No answer

1.8 cfg and subject predicate 1 / 3

- ✓ **- 1 pts** **A**
- ✓ **- 1 pts** **B**
- ✓ **+ 3 pts** **C (correct)**
- **- 1 pts** D
- **+ 0 pts** No answer selected

1.9 skip-gram 0 / 3

- **- 0 pts** Correct (A)
- ✓ **- 3 pts** **Incorrect**
- **- 1 pts** Correct answer selected in addition to one incorrect option
- **- 3 pts** No answer

1.10 markov assumption 2 / 3

- ✓ **- 1 pts** **A**
- **+ 2 pts** Correct answer (B only)
- **+ 3 pts** Correct answer (B and C)
- ✓ **+ 3 pts** **Correct answer (C only)**
- **- 1 pts** D
- **+ 0 pts** Empty set selected
- **+ 0 pts** No answer

QUESTION 2

Short answer 48 pts

2.1 LSTM diagram 8 / 8

✓ **- 0 pts Correct**

- 8 pts Did not attempt

- 1 pts Diagram missing gates/memory cell

- 0.5 pts Diagram shows gates/cell hooked up incorrectly

- 1 pts No input gate mentioned

- 1 pts No output gate mentioned

- 1 pts Memory cell mentioned but no forget gate

- 2 pts No memory cell mentioned and no forget gate

- 1.5 pts Incomplete justification

- 3 pts No justification

- 3 pts Incorrect justification

- 8 pts Not counted

2.2 backprop 8 / 8

✓ **- 0 pts Correct**

- 8 pts Did not attempt

- 0.5 pts Minor arithmetic errors

- 0.5 pts Update formulas correct except for learning rate

- 1 pts Update formulas for w and/or b correct but not followed through

- 1 pts Incorrect/missing update formula for w

- 1 pts Incorrect/missing update formula for b

- 1 pts Attempted gradient for w (dL/dw) but did not plug in correct formulas (independent of chain rule)

- 1 pts Attempted gradient for b (dL/db) but did not plug in correct formulas (independent of chain rule)

- 2 pts Attempted gradient for w (dL/dw) but misapplied chain rule

- 2 pts Attempted gradient for b (dL/db) but misapplied chain rule

- 3 pts Completely incorrect/missing gradient for w (dL/dw)

- 3 pts Completely incorrect/missing gradient for b (dL/db)

- 8 pts Not counted.

You seem to be treating g as f_2, which is fine for the purposes of this problem (as dg/df_2 = 1) but technically not accurate.

2.3 context free grammar 8 / 8

✓ **- 0 pts Correct part (a)**

- 0.5 pts (a) Minor mistakes in one parse tree

- 1 pts (a) Minor mistakes in both parse trees

- 2 pts (a) one parse tree incorrect

- 4 pts (b) both parse trees are incorrect

✓ **- 0 pts Correct part (b)**

- 2 pts No way to combine multiple adjectives

- 1 pts (b) Missing Adj -> large

- 1 pts (a) Missing Adj -> silver

- 8 pts Did not attempt

2.4 polysemy and metonymy 0 / 8

- 0 pts Correct

✓ **- 1 pts Don't describe similarities explicitly**

- 1 pts No example or incorrect example of polysemy

✓ **- 1 pts No example or incorrect example of metonymy**

- 1 pts Partially incorrect definition of polysemy

- 1 pts Partially incorrect definition of metonymy

- 2 pts No definition or incorrect definition of polysemy

✓ **- 2 pts No definition or incorrect definition of metonymy**

- 8 pts Did not attempt

✓ **- 8 pts Not counted**

2.5 NN dimensions 8 / 8

✓ **- 0 pts Correct dimensions for x, W, b.**

- 8 pts Didn't answer this question.

- 0 pts No points removed, but interpreted problem as Wx+b instead of xW+b.

- 2 pts Incorrect dimensions for x.

- 2 pts Partially incorrect dimensions for W.

- 4 pts Incorrect dimensions for W.

- 2 pts Incorrect dimensions for b.

- 8 pts Not counted

2.6 dropout 0 / 8

✓ **- 0 pts Correctly defines dropout and why it improves learning.**

- 8 pts Didn't answer this question.

- 4 pts Doesn't define or incorrectly defines dropout.

- 4 pts Doesn't explain or incorrectly explains why dropout improves learning.

- 2 pts Some details of dropout definition are incorrect.

- 2 pts Some details off dropout explanation are incorrect.

✓ **- 8 pts Not counted**

QUESTION 3

Problem solving 38 pts

3.1 dependency parsing 13 / 19

Part a

✓ **- 0 pts Correct dependency tree and original sentence**

- 2 pts Errors in dependency tree with correct sentence

- 3 pts Errors in dependency tree with incorrect sentence

- 3 pts No dependency tree or no sentence

- 6 pts No answer

- 1 pts Dependency tree labels are missing

- 1 pts Incorrect sentence

- 1 pts Dependency tree missing arcs

- 1 pts Small error

Part b

✓ **- 0 pts Correct stack, buffer, arc set**

- 2 pts Minor errors

- 4 pts Semi-moderate errors

- 5 pts Moderate errors

- 6 pts Major errors

- 7 pts No answer or completely incorrect

Part c

- 0 pts Correct and adequate answer

- 2 pts Correct idea but inadequate/unclear details

- 4 pts Very few details or unclear idea

✓ **- 6 pts Incorrect or no answer**

3.2 hidden markov models 14 / 19

- 4 pts Provided correct description of how to compute probabilities: counts of row and col POS tags divided by total count of col POS in corpus.

- 2 pts Showed a dynamic trellis with a node for every POS

- 2 pts Showed that they knew to use the formula in the algorithm loop to make the computation at each node

✓ **- 2 pts Correct calculation for “I”**

- 2 pts Correct calculation for "want"

✓ **- 2 pts Correct calculation for "a"**

- 2 pts Correct calculation for "ride"

- 3 pts Showed how "ride" is disambiguated to "NN" at the state for ride

- 0 pts Correct

- 1 Point adjustment

Calculation for ride should show multiplication of entire previous path.

Midterm Exam

COMS W4705: Natural Language Processing

October 21, 2019

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 itself. Answer Part II and Part III questions in the space on the test following the question. Turn in all test sheets at the end of the exam. If you need extra paper, write CONTINUED at the end of the space and put the problem number in bold on top of the extra piece of paper.

Name: Jing Qian UNI: jq2282.

UNI of the person to your left: None KP5238

UNI of the person to your right: None .

1 Multiple choice (30 points total)

Recommended time: 20 minutes.

1. (3 points) Which assumption(s) are made by a multinomial Naïve Bayes classifier?

   A. The classification problem is binary (i.e., the labels are $\in \{0, 1\}$). ?

   (B.) The words in a document are independent of each other given the class of the document.

   C. The probability of each word appearing in the document depends only on the word that appeared right before it. X

   D. Documents are represented as vectors of word counts. X ? bag of words

2. (3 points) Given the training data below, estimate the probability of the sequence "I like meatballs." Assume you are using a bigram language model and that punctuation DOES NOT count as a unigram.

   Begin training data:

   ```
   I like spaghetti and meatballs.  I like meatballs made of beef.  Yum!
   So hungry!
   ```

   End training data.

   $P(\text{I like}) = \frac{C(\text{I, like})}{C(\text{I})} = \frac{2}{2} = 1$

   $P(\text{like m}) = \frac{C(\text{like m})}{C(\text{like})} = \frac{1}{2}$

   $P(\text{I}) = \frac{1}{2}$

   A. 1/12

   B. 1/6

   C. 1/8

   (D.) 1/4

3. (3 points) A valid dependency tree is projective if

   (A.) None of the arrows in the tree cross any other arrow.

   B. No more than one arrow in the tree crosses other arrows.

   C. Every node has only one parent.

   D. None of the above.

4. (3 points) Which of the following are **true** about perplexity? (Select all that are true.)

   A. Perplexity is a measure of how well a grammar or language model models a corpus.

   B. A high perplexity means the model is better.

   C. A corpus of simple, repetitive sentences, like a children's book, would have a **higher** perplexity than a corpus of complex, varied sentences, like a complicated novel.

   D. Perplexities are always greater than 1.

5. (3 points) The Distributional Hypothesis states... (select only one)

   A. That we can distribute predictive power across all words in a sentence.

   B. That the frequency of words is distributed according to an inverse power law

   C. That words that occur in the same contexts tend to have similar meanings.

   D. That the probability of a word occurring can be accurately modeled with only the n previous words.

6. (3 points) Suppose that we have a vocabulary size of V, and a corpus C in which m unique bigrams are observed, where $m \leq V^2$. You create a bigram language model using the bigram counts from this corpus. However, you have some unobserved bigrams and choose to smooth your language model with add-one smoothing, where you add 1 to the count of every bigram (and renormalize your probabilities appropriately). How much probability mass is removed from observed bigrams to be given to unobserved bigrams (total)?

   A. $\sum_{(w_i,w_j)\in C} \frac{count(w_i,w_j)+1}{count(w_i)+V}$, for the m bigrams you originally observed

   B. $\sum_{(w_i,w_j)\notin C} \frac{count(w_i,w_j)}{count(w_i)}$, for the $V^2 - m$ bigrams you did not originally observe

   C. $\sum_{(w_i,w_j)\in C} \frac{count(w_i)+1}{count(w_i)+V}$, for the m bigrams you originally observed

   D. $\sum_{(w_i,w_j)\notin C} \frac{1}{count(w_i)+V}$, for the $V^2 - m$ bigrams you did not originally observe

7. (3 points) Which of the following statements is **not** true of a discriminative classifier?

   A. It learns boundaries between classes.

   B. It is probabilistic.

   C. It infers outputs based on inputs.

   D. It learns weights for features.

8. (3 points) In English, sentences generally follow subject-verb-object word order, where the verb and object form a constituent verb phrase (predicate). This is called "subject-predicate structure". Recall in an English CFG some examples of subject-predicate structure are expressed by the following rules:

```
S -> NP VP (John saw the dog)
S -> Aux NP VP (Did John see the dog?)
S -> Conj VP NP aux (And see the dog, John did!)
VP -> V NP. (see the dog)
```

In which of the following languages would it be impossible for a CFG to express subject-predicate structures?

A. A language with object-verb-subject order, just as Hixkaryana:
"toto yonoye kamara" (lit. "person ate jaguar" — "the jaguar ate the man").

B. A language with subject-object-verb order, such as Turkish:
"Mustafa eşekleri gördü" (lit. "Mustafa donkeys saw" — "Mustafa saw the donkeys").

C. A language with verb-subject-object order, such as Welsh:
"welodd Mair ddraig" (lit. "saw Mary dragon" — "Mary saw a dragon").

D. A language with verb-object-subject order, such as Malagasy:
"Namùnji àzi àhu" (lit. "helped out him I" — "I helped him out").

9. (3 points) In skip-gram with negative sampling, why do we use negative samples rather than simply trying to maximize $\sigma(w \cdot c)$ for (w, c) pairs in our corpus?

A. If we don't have any negative samples, we never learn which word and context vectors should be far apart.

B. If we use $\sigma(w \cdot c)$, we will always converge to the trivial solution where all word vectors are diametrically opposite to all context vectors.

C. Using negative samples increases numerical stability.

D. If we use $\sigma(w \cdot c)$, probabilities will not sum to 1.

10. (3 points) The nth-order version of the Markov assumption states that the probability of occurrence of a word at any given point in time is only dependent on the n words that occurred immediately before it. This is a very strong assumption for low n, but grows more realistic as our history length n increases. Why, then, do we typically choose low values of n, in the 1-3 range? (Select all that apply.)

A. In practice, most in-sentence dependencies do not stretch further than three words back in time.

B. It is easier to model longer sequences of words with more powerful models, such as recurrent neural networks.

C. For efficiency's sake - computation is impractical for large n.

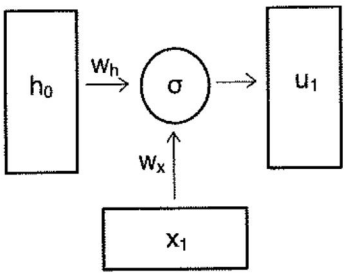
D. Increasing n reduces the number of n-grams we can observe in a sentence, as each n-gram is longer; therefore we are actually losing information when we use higher n.

2 Short answer (32 points total)

Provide at most 2 or 3 sentences for four out of the following six questions. Each question is worth 8 points.

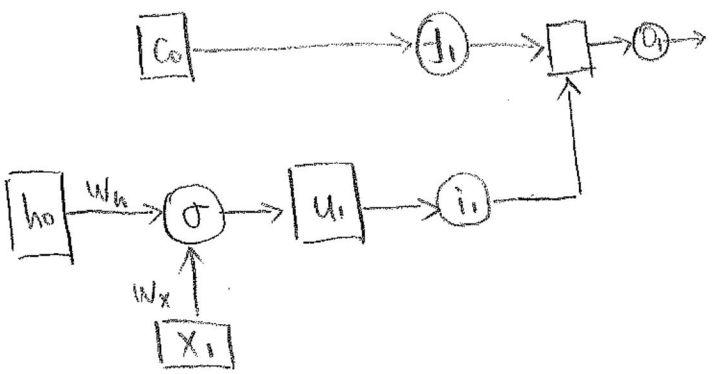
NOTE: Choose FOUR. If you answer more than four, you will be graded on the first four answers you provide.

Recommended time: 25 minutes.



1. (8 points) A diagram of a basic RNN is shown above. To augment it to an LSTM, what would you add and why? Show the augmented diagram of the basic LSTM.

I would add the memory term. LSTM is long short-term memory, which means we keep memories and combine it with the weighted input



i, a, f, are the gates to control input, output and forget

2. (8 points) Consider the following basic neural network.

$f_1 : \mathbb{R}^3 \to \mathbb{R}$ $\quad f_1(x) = w \cdot x_1 + b \cdot x_3$

$f_2 : \mathbb{R} \to \mathbb{R}$ $\quad f_2(x) = e^{2x}$

$g : \mathbb{R}^3 \to \mathbb{R}$ $\quad g(x) = f_2(f_1(x))$

$$L(D) = -\sum_{x^{(i)} \in D} \log g(x^{(i)}).$$

Suppose that we have data $D = \{x^{(1)} = [x_1, x_2, x_3]\} = \{x^{(1)} = [1, 2, -1]\}$ and $w = 2, b = 1$. Assume also that the learning rate $\eta = 0.1$. After one pass through D of minibatch stochastic gradient descent with batch size of 1, what are w and b?

∵ Batch size = 1

∴ $L(D) = -\log g(x^{(1)})$

$f_1(x) = w \cdot x_1 + b \cdot x_3$
$= 2 \times 1 + 1 \times (-1)$
$= 1$

$g(x) = f_2(f_1(x)) = e^{2f_1(x)} = e^2$

$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial g} \frac{\partial g}{\partial f_1} \frac{\partial f_1}{\partial w} = (-\frac{1}{g})(g \cdot 2) \cdot x_1$
$= -2x_1$
$= -2.$

$\frac{\partial g}{\partial f_1} = \frac{\partial (e^{2f_1})}{\partial f_1} = 2e^{2f_1}$
$= 2g.$

$\frac{\partial f_1}{\partial w} = x_1$

$\frac{\partial f_1}{\partial b} = x_3$

$\frac{\partial L}{\partial g} = -\frac{1}{g}$

$\frac{\partial L}{\partial b} = \frac{\partial L}{\partial g} \frac{\partial g}{\partial f_1} \frac{\partial f_1}{\partial b} = (-\frac{1}{g})(g \cdot 2) \cdot x_3$
$= -2x_3$
$= 2.$

∴ $w \leftarrow w - \eta \frac{\partial L}{\partial w} = 2 - 0.1 \times (-2) = 2.2$

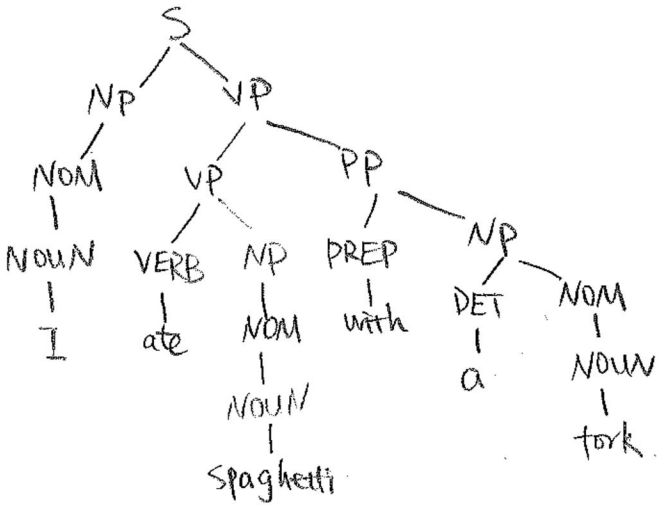
$b \leftarrow b - \eta \frac{\partial L}{\partial b} = 1 - 0.1 \times 2 = 0.8.$

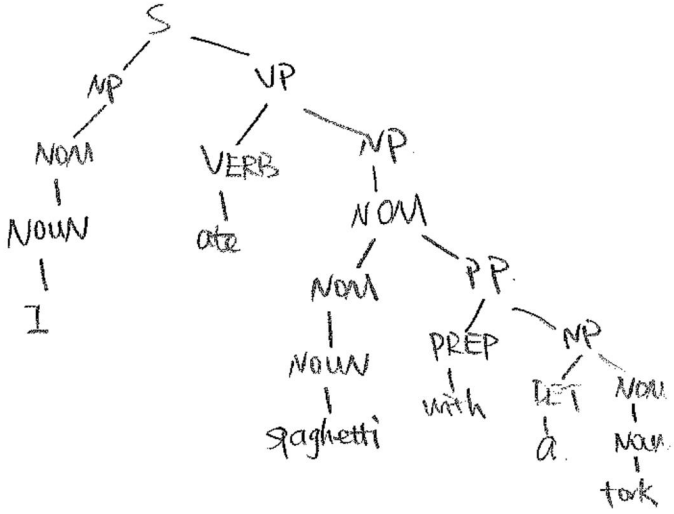
3. (8 points) Consider the context free grammar below.

```
S-> NP VP
VP -> VP PP
VP -> VERB NP
VP -> VERB
NP -> DET NOM
NP -> NOM
NOM -> NOM PP
NOM -> NOUN
PP -> PREP NP

NOUN -> I
NOUN -> spaghetti
NOUN -> fork
NOUN -> night
VERB -> ate
PREP -> with
PREP -> at
DET -> a
```

(a) Show two different parses that would be derived for "I ate spaghetti with a fork."





(b) Augment the grammar to include one or more adjectives so that it can parse the sentence "I ate spaghetti with a large silver fork."

JJ → large

JJ → silver

NOM → JJ NOM

?Adj?

4. (8 points) How are polysemy and metonymy similar and/or different? Give an example of each.

Polysemy means a word having different meanings.
Eg: "bat" could be an animal or a sports instrument.

?

5. (8 points) Suppose you are given a fully connected layer $l(x) = xW + b$ with input dimensionality d_{in} and output dimensionality d_{out}. Give the dimensionality of x, W and b in terms of d_{in} and d_{out}. (x is the input).

The dimensionality of X is: batch size × d in. (If no batch, then 1 × d in)

W is: $d_{in} \times d_{out}$

b is: $1 \times d_{out}$

6. (8 points) What is dropout? How does it work? Why does it improve learning in a neural net?

Dropout means we randomly turn off some neurons in the neural network.
With the dropout rate setting to a probability, the neural network discard the output of the randomly selected neurons.

Dropout would prevent overfitting in the neural net, especially when there are noises in the training set. So dropout improves learning.

3 Problem solving (38 points)

There are two problems in this section. Do both problems. Put your answers on the blank page after each question.

Recommended time: 30 minutes.

1. (19 points) **Dependency parsing.**

   (a) Given the following dependency parse output, provide a diagram of the dependency tree and give the original sentence that was input (the sentence has two instances of "the").

   ```
   A = {
       (<root>, PRED, are; right-arc), (are, NSUBJ, dogs; left-arc),
       (dogs, DET, the; left-arc), (dogs, PREP, on; right-arc),
       (on, POBJ, couch; right-arc), (couch, DET, the; left-arc),
       (are, ACOMP, cute; right-arc), (cute, CC, and; right-arc),
       (cute, CONJ, fuzzy; right-arc), (are, PUNCT, .; right-arc)
   }
   ```

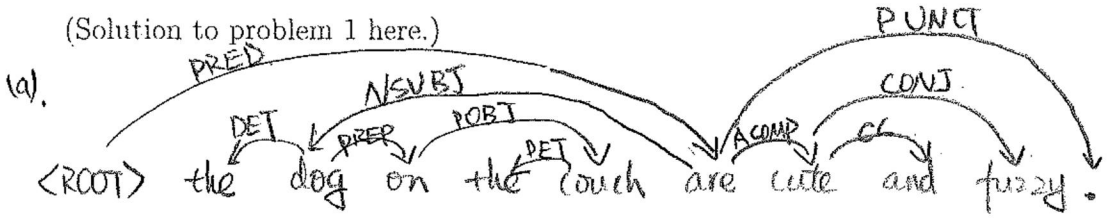
   (b) Show the transitions that would occur in parsing the following sentence using an arc standard parser and assuming an oracle that can tell you which transition to take next. Show the state of the stack, the buffer and the arc set (you may omit the label) as the sentence is parsed.

   "Japan woke to flooded rivers."

   (c) When there is a choice of actions to apply, how does the parser know which action to choose?

(Solution to problem 1 here.)

(a).



The original sentence is:

The dog on the couch are cute and fuzzy.

(b)

Transitions	stack	parser	arc set
	[ROOT]	Japan woke to flooded rivers.	
SHIFT			
	[ROOT] Japan	woke to flooded rivers.	
SHIFT			
	[ROOT] Japan woke	to flooded rivers.	
LEFT ARC			(woke, Japan; left-arc)
	[ROOT] woke	to flooded rivers.	
SHIFT			
	[ROOT] woke to	flooded rivers.	
SHIFT			
	[ROOT] woke to flooded	rivers.	
SHIFT			
	[ROOT] woke to flooded rivers	.	
LEFT ARC			(rivers, flooded; left-arc)
	[ROOT] woke to rivers	.	
RIGHT ARC			(to, rivers; right-arc)
	[ROOT] woke to	.	
RIGHT ARC			(woke, to; right-arc)
	[ROOT] woke	.	
SHIFT			
	[ROOT] woke .		
RIGHT ARC			(woke, . ; right-arc)
	[ROOT] woke		
RIGHT ARC FINISH	[ROOT]		([ROOT], woke; right-arc)

(c). The parser chooses the word that all its children have been parsed.

2. (19 points) **Hidden Markov Models.** You are given the following sentence and the tables of probabilities shown in Table 1 and Table 2 below:

   I want a ride on the merry-go-round.

   (a) Describe how you would compute the tag transition probabilities in Table 2.

   (b) Given the sentence "I want a ride on the merry-go-round." show how you would compute the probability of "ride" as a verb versus the probability of "ride" as a noun using the probabilities in Tables 1 and 2 and the Viterbi algorithm. Note that there may be other POS tags not shown here, and thus the columns and rows don't always add up to 1.

   The Viterbi algorithm is shown on the next page. For this question you should:

   i) Show the dynamic programming trellis at each state up to the point where "ride" is disambiguated.

   ii) Show the formula with the values that would be used to compute the probability of "ride" as either verb or noun. You do not need to do the arithmetic. Just show the formula that would be computed (e.g., .05*.03*0).

	I	*want*	*ride*	*the*	*on*	*a*	*merry-go-round*
VB	0	.50	.40	0	0	0	0
TO	0	0	0	0	.99	0	0
NN	0	.40	.30	0	0	0	.99
PPSS	.50	0	0	0	0	0	0
DT	0	0	0	.99	0	.90	0

Table 1: Observation likelihoods.

	VB	TO	NN	PPSS	DT
<s>	.05	.01	.10	.40	.30
VB	.01	.20	.50	.01	0
TO	.90	0	.01	0	.01
NN	.20	.10	.30	.10	.10
PPSS	.50	.02	.10	.05	.10
DT	0	0	.80	0	0

Table 2: Tag transition probabilities. The rows are labeled with the conditioning event. Thus, P(VB|<s>) = .01.

function VITERBI(*observations* of len T, *state-graph*) **returns** *best-path*

num-states ← NUM-OF-STATES(*state-graph*)
Create a path probability matrix *viterbi[num-states+2,T+2]*
viterbi[0,0] ← 1.0
for each time step t **from** 1 **to** T **do**
  for each state s **from** 1 **to** *num-states* **do**
    $viterbi[s,t] \leftarrow \max_{1 \leq s' \leq num\text{-}states} viterbi[s',t-1] * a_{s',s} * b_s(o_t)$
    $backpointer[s,t] \leftarrow \text{argmax}_{1 \leq s' \leq num\text{-}states} viterbi[s',t-1] * a_{s',s}$
Backtrace from highest probability state in final column of *viterbi[]* and return path

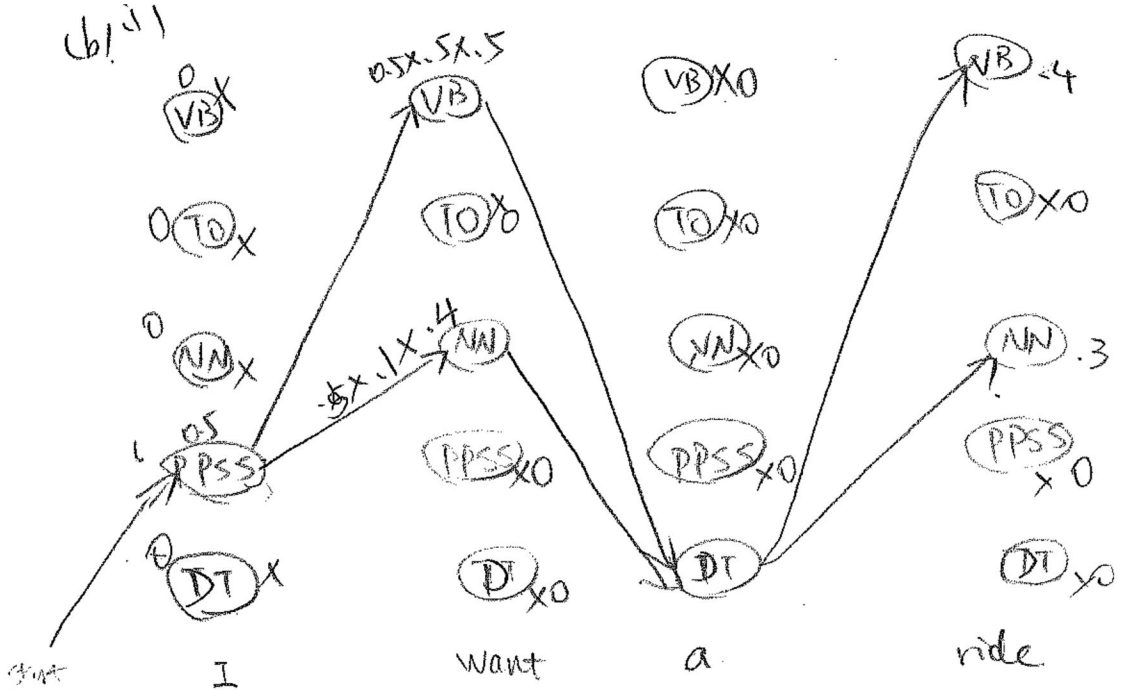
Figure 1: The Viterbi algorithm.

(Solution to problem 2 here.)

(a) Use the division between counts $P(\text{Tag2}|\text{Tag1}) = \frac{\text{count}(\text{Tag1}, \text{Tag2})}{\text{count}(\text{Tag1})}$.

Eg: $P(VB|TO) = \frac{P(TO\ VB)}{P(TO)}$.

(b)(i)



(ii):

$P(\text{ride}|VB)\ P(VB|DT)\ P(TO|VB) = .4 \times 0 \times .9 = 0$

$P(\text{ride}|NN)\ P(NN|DT)\ P(TO|NN) = .3 \times .8 \times .1 \to 0$

$\therefore$ ride is a NN here.