Midterm Preparation

Adam Meyers
New York University
Administrative Details

• Time: Tuesday March 20, 2018, normal class time
• Ask clarification questions during test
  – I especially want to fix errors such as typos
• Open Book, Open Notes
  – Bring paper materials
  – Search the web
  – Do simple calculations with a calculator
  – **DO NOT:**
    • communicate with others (texting, email, phone)
    • write/run actual programs
      – Don't run POS taggers or regular expressions, etc.
• Put your name and ID number on your test (and on any pages that get disconnected from your test)
• I will take attendance: please bring your ID
Gradescope

• I will be experimenting with new test grading software
  – This will involve scanning all the tests
  – I hope to improve speed and consistency in grading

• Write all answers in the specified boxes on the test.
  – This will make it easier for the system to find answers
  – I will include some duplicate boxes – if you make an error, cross it out clearly and write the new answer in the duplicate box

• Please do not remove (or add) any pages from the exam.
  – This will make it easier to keep tests together

• Please use pen or dark pencil.
  – Light pencil may not scan very well.

• Before this talk, I will have tried Gradescope once with the Python (Intro) class – these limitations may be clearer by then.
The Purpose of the Midterm?

• Pedagogical purposes
  – Track whether students learned parts of curriculum and what may need further clarification
  – Provide a motivating force for students to study the “important” parts of the curriculum
  – Clarify how to prepare students to do final projects

• Administrative purpose: determine 1/3 of grade

• Possible conflict
  – A difficult test makes pedagogical sense
  – An “acceptable” average grade may make administrative sense

• Strategy:
  – Motivate test based on pedagogical objectives
  – Make it as open book as possible
Outline

• Linguistic Resources & Descriptive Linguistics
  – Especially Corpus Annotation
• Rules used by Automated Procedures
  – Ones covered in Class
• Algorithms Discussed in Class
• How does Evaluation Work
• Sample Midterm
Annotation

• You should be able to write usable specifications
• You should be able to annotate based on specifications
• You should understand some of the mechanics
  – Character offsets
  – A Markup language
  – BIO tags
• You should understand the difference between training, development and test corpora
Descriptive Linguistics

• The basic parts of speech and phrasal categories.
  – The difference between a determiners and an adjective
  – Verbs, prepositions, coordinate conjunctions

• How to manually divide sentences into tokens

• You should know how to identify the head of a phrase

• You should be able to draw a phrase structure tree modeling the linguistic analysis of a sentence
Common Difficulties with Phrase Structure

• SBAR → IN S
  – …. that this is a sentence.

• PP → IN NP
  – in the house, for freedom, on the clock, of chocolate, ...

• Coordinate conjunctions (and, or, but) link 2 constituents of the same type together
  – [NP [NP …] and [NP …]]
  – [VP [VP …] or [VP …]]
  – [S [S …] but [S ]]

• to infinitives are VPs, e.g. [VP to [VP go to the movies]]
Rules: Regular Expressions

• You should know how to write a basic regular expression
  – Decent coverage, but not over-generate too much

• You should know how to write a phrase structure rule including at least:
  – Context free rules
  – Left (or right) regular rules

• For a regular expression, you should be able to identify a set of phrase structure rules that describe the same language (set of strings)
Algorithms: Deterministic Finite State Machine

- Given:
  - Finite State Machine (FSM)
  - Input String
- Would the FSM recognize the string?
- Which sequence of states would be entered before recognition was complete?
- How would the FSM on the next slide process:
  - AababAB
  - AABB
DFSA for Regexp: $A(ab)^*ABB$?
Algorithms: Context-Free Generator

• Show steps for randomly generating a sentence given a lexicon and context-free grammar with start symbol S

• The start symbol is inserted into a stack.

• Repeat until stack is empty:
  – If the top most item is a terminal, pop it off and include it in the output.
  – Else, replace the top most item in the stack with the right hand of a rule of the form $X \rightarrow \text{Right Hand Side}$
Example of Generator

• Add S to top of empty stack
  – Stack is now: S

• Substitute NP VP for S
  – Stack is now: NP VP

• Substitute DT N PP for NP
  – Stack is now: DT N PP VP

• Pop off DT and add the to the output
  – Stack is now: N PP VP

• Etc.
The CKY parsing algorithm

• Fill in the triangular chart (next slide) given a (short) sentence and context free rules
• Rules must be in Chomsky Normal Form (binary)
• Rules assumed in next slide
  – $S \rightarrow NP \ VP$
  – $NP \rightarrow N$
  – $NP \rightarrow DT\ N$
  – $NP \rightarrow POSSP\ N$
  – $POSSP \rightarrow NP\ POSS$
  – $VP \rightarrow V$
  – $VP \rightarrow V\ NP$
# Filling in a CKY Chart

<table>
<thead>
<tr>
<th>The</th>
<th>clam</th>
<th>'s</th>
<th>group</th>
<th>had</th>
<th>knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>0</td>
<td>D [0,1] NP [0,2] POSSP [0,3] NP [0,4] S [0,5] S [0,6]</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>N, NP [1,2] POSSP [1,3] NP [1,4] S [1,5] S [1,6]</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td>V, VP [4,5] VP [4,6]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td>N,NP [5,6]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Language Models

- Calculate probability for words based on available information
- Bi-gram model (most info), backing off to unigram and then OOV
- Bigram and Unigram frequencies for 48 word corpus on next slide:
  - begin and end of sentence are treated as tokens
  - Words occurring only once are counted as *oov*
- Corpus = 48 word Edward Lear Limerick, where each line is assumed to be a separate sentence
  
  *There was an Old Man of the North,*
  
  *Who fell into a basin of broth;*
  
  *But a laudable cook,*
  
  *Fished him out with a hook,*
  
  *Which saved that Old Man of the North.*
Frequencies

• Unigram freqs:
  – 21: *oov*
  – 5: B_Sent, E_Sent
  – 3: of, a, *comma*
  – 2: old, man, the, north

• Bigram freqs:
  – 11: *oov* + *oov*
  – 5: B_Sent + *oov*
  – 3: *oov* + a, a + *oov*, *comma* + E_Sent, a + *oov*
  – 2: *oov* + old, *oov* + E_Sent, *oov* + *comma*, old + man, man + of, of + the, the + north
  – 1: *oov* + of, of + *oov*, north + *oov*, north + *comma*

• There are 48 tokens.

• What is the probability of the following sentence?
  – But a laudable cook, \( \rightarrow \) but + a + laudable + cook + ,
Probability of “but a laudable cook,”

- but = \( \frac{\text{freq}(\text{B-sent} + \text{*oov*})}{\text{freq}(\text{B-sent})} = \frac{5}{5} = 1 \)
- a = \( \frac{\text{freq}(\text{*oov*} + \text{a})}{\text{freq}(\text{*oov*})} = \frac{3}{21} \)
- laudible = \( \frac{\text{freq}(\text{a} + \text{*oov*})}{\text{freq}(\text{a})} = \frac{3}{3} = 1 \)
- cook = \( \frac{\text{freq}(\text{*oov*} + \text{*oov*})}{\text{freq}(\text{*oov*})} = \frac{11}{21} \)
- *comma* = \( \frac{\text{freq}(\text{*oov*} + \text{*comma*})}{\text{freq}(\text{*oov*})} = \frac{2}{21} \)
- E-sent = \( \frac{\text{freq}(\text{*comma*} + \text{E-sent})}{\text{freq}(\text{*comma*})} = \frac{3}{3} = 1 \)
- Probability of But a laudable cook, is:
  - \( 1 \times \frac{3}{21} \times \frac{3}{3} \times \frac{11}{21} \times \frac{2}{21} \times 1 = .00713 \)
Viterbi Decoding of HMM for *rose pickles*

- **Likelihood:**
  - *rose*: NNP .01, NN .02, VBD .05
  - *pickles*: NNP .001, NNS .03, VBZ .05

- **Transition Probabilities:**

![Diagram of HMM states and transition probabilities]
**Rose/NNP Pickles/VBZ**

- Each Cell = Max(Previous * Transition * Likelihood)
- P(Rose = NNP at position 1): 1 (Start) * .42 (start → NNP) * .01 (likelihood NNP → Rose)
- P(Pickles = VBZ at position 2, given NNP at position 1) = max prob at NNP, position 1 * .3 (NNP→ VBZ) * .05 (VBZ→ Pickles)
- P(End, given VBZ at position 2): (max position 2) * .45 (VBZ → End) = 2.85 * 10⁻⁵

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1: Rose</th>
<th>2: Pickles</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NNP</td>
<td>.42 * .01</td>
<td><em>0</em>.001</td>
<td><em>0</em>.001</td>
<td>.23*.001 = .00023</td>
</tr>
<tr>
<td>NNS</td>
<td>.20 * .02</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NN</td>
<td>.20 * .02</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VBZ</td>
<td>.05 * .05</td>
<td><em>3</em>.05=6*10⁻⁵</td>
<td><em>1</em>.05</td>
<td><em>0</em>.05</td>
</tr>
<tr>
<td>VBD</td>
<td>.05 * .05</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>End</td>
<td>.2 =1.15*10⁻²</td>
<td>.15 = 9.0*10⁻⁵</td>
<td>.45 = 2.85*10⁻⁵</td>
<td></td>
</tr>
</tbody>
</table>
Common Evaluation Metrics

• If all instances are classified
  – Accuracy = Correct/All-Instances

• If only some instances are classified
  – Precision = Correct/Instances in System Output
  – Recall = Correct/Instances in Answer Key
  – F-measure = Mean of Precision and Recall
    • Harmonic Mean of Precision and Recall
    – \[ \frac{2}{\left(\frac{1}{\text{precision}} + \frac{1}{\text{recall}}\right)} \]
Sample Precision and Recall

- System for finding holiday names
- Exactly 10 correct holiday names in hand-coded corpus (the answer key)
- The system marks 12 holiday names, 8 of which match the ones in the answer key.
  - Precision = $\frac{8}{12} = .67$
  - Recall = $\frac{8}{10} = .80$
  - F-measure = $\frac{2}{\frac{1}{.80} + \frac{1}{.67}} = .73$
TFIDF

- TFIDF – Property of Term with respect to a document
  - keyword suitability, representativeness of a topic, etc.
  - Uses: Doc Retrieval, Term Extraction, etc.
- TF = frequency in a document
- IDF = number of documents in sample divided by number of documents containing word
- TFIDF = TF * log(IDF)
- Example: “rock” occurs 10 times in document X. It occurs in 100 out of 3000 documents in collection. TFIDF = 10*log(3000/100) = 34.01
- *Use natural logarithms – standard (and easier to grade)
  - Cosine similarity using TFIDF vectors is the same regardless of base used
Cosine Similarity Between Query and Document

\[
\text{Similarity}(A, B) = \frac{\sum_i a_i \times b_i}{\sqrt{\sum_i a_i^2 \times \sum_i b_i^2}}
\]

Example:

- the terms in the vectors include: animal, vegetable, mineral, monkey, golf enthusiast
- The vector for the query is: [0,0,0,34,.8]
- The vector for a given document is: [1,2,3,4,5]
- Similarity = \[\frac{(34 \times 4) + (8 \times .5)}{\sqrt{((34^2) + (8^2)) \times (1^2 + 2^2 + 3^2 + 4^2 + 5^2)}} = \frac{140}{\sqrt{1220 \times 113}} \approx \frac{140}{371.30} \approx .377\]

Note: Cosine similarity is always between 0 and 1
Practice Midterms

• Sample midterm & answers online
  – More time-consuming than actual midterm. 9 questions on the sample vs 7 on actual midterm.
  – Same format as midterm (geared for Gradescope)

• Last Term's Midterm & answers
  – https://cs.nyu.edu/courses/spring18/CSCI-UA.0480-009/last_terms_midterm.pdf
  – https://cs.nyu.edu/courses/spring18/CSCI-UA.0480-009/last_terms_midterm_with_answers.pdf
  – Difficulty comparable to this term's midterm
  – Old format – not geared for Gradescope

• Old Midterms from previous NLP classes: Fall 2015 to Present
  – See previous classes on my website

Midterm Preparation
2018
General Test-taking Advice

• Test is a game –
  – Not worth getting tense about
  – Staying calm makes it easier to think clearly

• Time may be an issue
  – Finish as many questions as possible
  – Budget time
    • 75 minutes/7 questions ≈ 10 minutes/question

• Show your work
  – It makes it easier justify partial credit

• Put answers in boxes for Gradescope compatibility