EDAN20
Final Examination

Pierre Nugues

October 30, 2014

The examination is worth 202 points. The distribution of points is indicated with the questions. You need 55% to have a mark of 4 and 70% to have a 5.

1 Closed Book Part: Questions

In this part, no document is allowed. It is worth 96 points.

Chapter 1. Cite three applications that use natural language processing to some extent. 3 points

Chapter 1. Annotate each word of the sentence *Pierre wrote notes* with its lemma and part of speech. You will use parts of speech you learned at school. 4 points

Chapter 1. Draw the graph of the sentence *Pierre wrote notes* using dependency relations. 4 points

Chapter 1. Represent the dependency graph of sentence *Pierre wrote notes* using the CoNLL format. You will use at least six columns: index, word, lemma, part of speech, head, and relation. 4 points

Chapter 1. Figure 1 shows some steps in semantic processing. Describe a possible simple technique to map the dependency graph to the logical form `wrote(Pierre, notes)` and what is entity linking. 8 points

Chapter 2. Describe what a concordance is and give all the case-sensitive concordances of the string *färg*: 3 points

> Tjuvar stal 1 200 liter färg
> MÅRTENS FÄLAD. Färg och puts till ett värde av 45 000 kronor stals vid ett nybygge på Mårtens Fälad. Sammanlagt rör det sig om 48 burkar innehållande 25 liter färg i varje.
>
> När målarna kom till sin arbetsplats på Sommarlovsvägen i fredags upptäckte de att två lastpallar med färg och puts var borta. De 48 burkarna, innehållande sammanlagt 1 200 liter vit färg och puts, hade placerats under en presenning utanför de nybyggda husen vars fasader skulle målas. – Eftersom självrisken

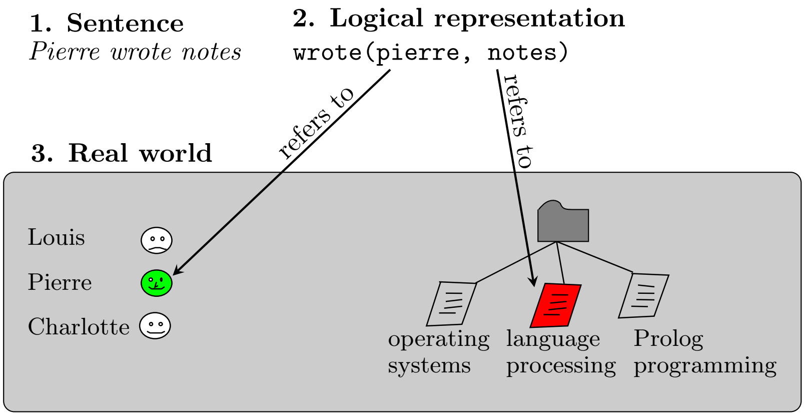

Figure 1: Steps in semantic processing

> uppgår till ett basbelopp, vilket är ungefär lika mycket som den stulna färgen, är det inte någon mening att anlita försäkringsbolaget. Istället får vi själv ta den här kostnaden. Sådana här stölder händer lite då och då. Men vi har aldrig fått reda på var den stulna färgen tagit vägen, säger Dan Karlsson, Karlssons Fasadrenovering AB.
>
> *Sydsvenskan.se*, Retrieved October 22, 2014. Author Tommy Lindstedt

Chapter 2. Identify what the regular expressions in the list below will match in the text above (identify all the matches or write no match if there is no match): 10 points

List of regular expressions. The symbol ␣ means a white space (visible white space):

1. `färg(en)*`
2. `färg(en)?`
3. `färg(en)+`
4. `\d{3}`
5. `\d+␣\d+`
6. `\w{15}`
7. `[A-Z]\.`
8. `[A-Z]\w+␣[A-Z]\w+`
9. `(.)\1\1`
10. `([ao])[a-z]\1`

Chapter 2. Name four edit operations used in approximate string matching and describe what they do. 2 points

Chapter 2. Use the edit operations to go from:

- *oepration* to *operation*,
- *opration* to *operation*, and
- *opesration* to *operation*.

You will represent the strings as two tapes that you will align with the ε symbol and you will use the edit operations to link the letters from the input tape to the output tape. 3 points

Chapter 3. Describe what UTF-8 is relatively to Unicode. 2 points

Chapter 3. Describe the effect of UTF-8 encoding of the string *Selma Lagerlöf* and a decoding of it using ASCII Latin1 codes. 2 points

Chapter 4. Give the mathematical definition of the entropy of a set of two classes, N and P whose probabilities are $P(N)$ and $P(P)$. When is this entropy maximal and when is it minimal? 2 points

Chapter 4. In machine–learning, describe what is a training set and what is a test set. 2 points

Chapter 4. In machine–learning, describe what a N-fold cross validation is. 3 points

Chapter 5. What does this Unix command do? 3 points

```
tr -cs 'A-Z' '\n' <file.txt
```

Chapter 5. What does this Unix command do? 2 points

```
uniq -c <file.txt
```

Chapter 5. Give the probabilistic model of the sentence *När målarna kom till sin arbetsplats* using no n-gram approximation (chain rule). You will ignore possible start and end of sentence symbols. 2 points

Chapter 5. Using a unigram approximation of the probability of the sentence *När målarna kom till sin arbetsplats.* You will ignore possible start and end of sentence symbols. 2 points

Chapter 5. Using a bigram approximation of the probability of the sentence *När målarna kom till sin arbetsplats.* You should have exactly the same number of terms as in the previous question. You will ignore possible start and end of sentence symbols. 2 points

Chapter 5. Describe what is the simple back off method to cope with unseen bigram. As example, you can use the sentence *När målarna kom till Concrez*, where the bigram *till Concrez*[1] has a count of 0 in Google. 2 points

Chapter 5. Give the definition of the mutual information association measure. 2 points

Words	POS	Groups
He	PRP	
reckons	VBZ	
the	DT	
current	JJ	
account	NN	
deficit	NN	
will	MD	
narrow	VB	
to	TO	
only	RB	
£	#	
1.8	CD	
billion	CD	
in	IN	
September	NNP	
.	.	

Table 1: An excerpt of the CoNLL 2000 dataset (Tjong Kim Sang and Buchholz, 2000)

Chapter 6. What are the lemmas, grammatical features, and parts of speech of the words in the sentence: *När målarna kom till sin arbetsplats.* Present your results in a CoNLL-like format.
If you cannot speak Swedish, use this sentence instead: *When the painters came back to their workplace.* 5 points

Chapters 9. and 10. Table 1 shows an excerpt of the CoNLL 2000 dataset (Tjong Kim Sang and Buchholz, 2000). Using the IOB-2 scheme, consisting of the begin, inside, and outside tags, annotate the sentence words with the noun groups and verb groups. 4 points

Chapters 9. and 10. Write DCG-like rules that identify all the noun groups and verb groups in Table 1. The left-hand side symbols will be either `np` or `vp` and the right-hand side symbols with the part-of-speech tags. The rules will have the form: 4 points

```
np --> POS1, POS2, ...
np --> POS3, POS4, ...
vp --> POS5, POS6, ...
...
```

Chapter 13. Nivre's parser uses four parsing actions: left-arc, right-arc, reduce, and shift. Define these four actions. 3 points

Chapter 13. Parse manually the sentence *Pierre write notes* using Nivre's parser. You will represent the stack and the queue at each parsing step and you should not need more than five actions. 6 points

[1] *Concrez* is a hamlet in France.

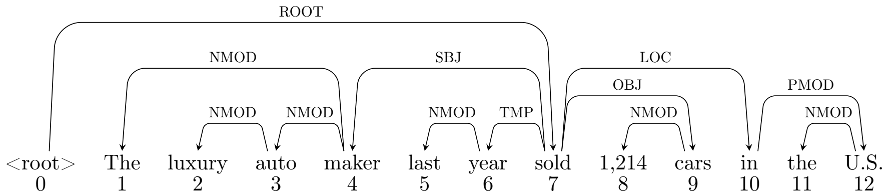

Figure 2: The syntactic dependencies

Chapter 14. Given a RDF triplestore containing these triples: 3 points

```
ilppp:Pierre rdf:type ilppp:person.
ilppp:Socrates rdf:type ilppp:person.

ilppp:table1 rdf:type ilppp:object.
ilppp:chair1 rdf:type ilppp:object.
ilppp:chair2 rdf:type ilppp:object.

ilppp:chair1 ilppp:in_front_of ilppp:table1.
ilppp:Socrates ilppp:in_front_of ilppp:table1.
ilppp:Pierre ilppp:on ilppp:table1.
```

what will be the result of this SPARQL query:

```
SELECT ?x ?y
WHERE
{
  ?x rdf:type ilppp:object.
  ?y rdf:type ilppp:object.
  ?x ilppp:in_front_of ?y
}
```

Chapter 15. Figure 2 shows the dependency graph of the sentence *The luxury auto maker last year sold 1,214 cars in the U.S.* and Fig. 3 shows the extraction of predicate–argument structures from this sentence. Imagine simple rules that will take the dependency graph as input and that will extract the arguments of the predicate *sell* and the argument labels. 4 points

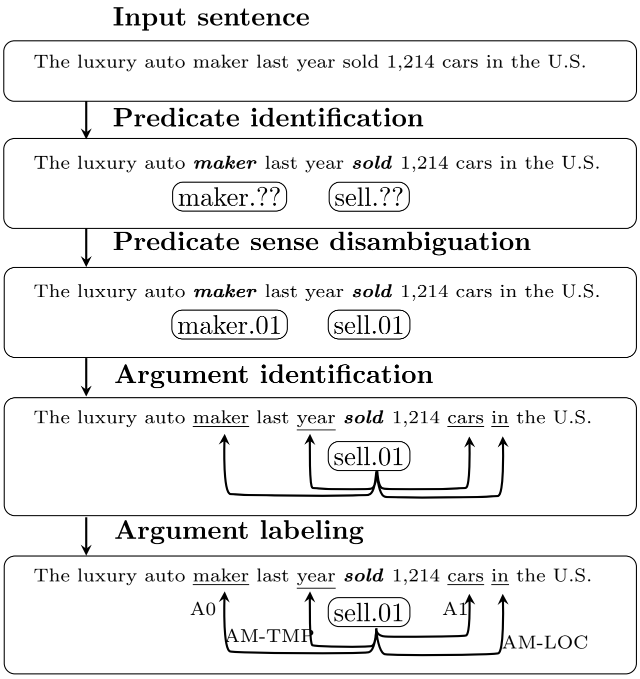
Input sentence
The luxury auto maker last year sold 1,214 cars in the U.S.
Predicate identification
The luxury auto maker last year sold 1,214 cars in the U.S.
maker.??
sell.??
Predicate sense disambiguation
The luxury auto maker last year sold 1,214 cars in the U.S.
maker.01
sell.01
Argument identification
The luxury auto maker last year sold 1,214 cars in the U.S.
sell.01
Argument labeling
The luxury auto maker last year sold 1,214 cars in the U.S.
A0
AM-TMP
sell.01
A1
AM-LOC

Figure 3: The sequence of classifiers: predicate identification, predicate sense disambiguation, argument identification, and argument labeling

2 Problem

In this part, documents are allowed. It is worth 106 points.

In this part, you will write a program to cluster words using a technique described in Brown et al. (1992). This technique automatically groups words according to their semantic similarity and is used in many NLP components such as part-of-speech tagger or parsers.

As programming language, you will use Java. You will focus on the program structure and not on the syntactic details. You can ignore the Java packages or imports for instance.

2.1 The Task

1. Read the introduction (Sect. 1) of the article by Brown et al. (1992) and Sect. 3. Summarize them in approximately 10 lines. If you are not sure to understand the complete mathematical development, do not spend too much time on it as we will review the key points in the next questions. 8 points

2. Rewrite the sentence probabilities of the two sentences below using a unigram and a bigram models: 2 points

   $$\begin{aligned} P(\text{Meeting on Monday 27th}) &= \\ P(\text{Meeting on Tuesday 28th}) &= \end{aligned}$$

   You will ignore the start-of-sentence symbol.

3. An n-gram class model uses the equality: 2 points

   $$P(w_k|w_1^{k-1}) = P(w_k|c_k)P(c_k|c_1^{k-1}),$$

   where w_1^k denotes a k word sequence: $w_1, w_2, ..., w_k$. Rewrite this equality for a unigram and a bigram class models.

4. Supposing that your corpus consists of the two sentences, *Meeting on Monday 27th* and *Meeting on Tuesday 28th*, map the six different words to four different classes that you think a word clustering algorithm would produce. Denote the mapping π as in the paper and the classes: c_α, c_β, c_γ, and c_δ. Note that α, β, γ, and δ are not index numbers. 3 points

   $$\begin{aligned} \pi(\text{Meeting}) &= \\ \pi(\text{on}) &= \\ \pi(\text{Monday}) &= \\ \pi(\text{27th}) &= \\ \pi(\text{Tuesday}) &= \\ \pi(\text{28th}) &= \end{aligned}$$

5. Rewrite the sentence probabilities of the two sentences below using a unigram and a bigram models and word classes: 2 points

   $$\begin{aligned} P(\text{Meeting on Monday 27th}) &= \\ P(\text{Meeting on Tuesday 28th}) &= \end{aligned}$$

   You will use the four different classes of the previous question.

6. Tell why a language model would be improved by a bigram class model for these sentences. You will discuss how the size of the training corpus may influence the results. 3 points

7. Brown et al. (1992) use a greedy algorithm to create the clusters. Say why and describe this algorithm in a few sentence. 5 points

2.2 Mathematical Development and Hand-Calculations

In this section, we will use a bigram language model and we will review the mathematical function that guides the clustering. We will experiment it on a short symbol sequence.

Brown et al. (1992) defined the quality of a class-based language model as the opposite of the entropy rate computed over a long sequence, here the corpus:

$$H = \frac{1}{n} \log_2 P(w_1, ..., w_n).$$

1. Rewrite H using a bigram class model. 2 points

2. Brown et al. (1992) showed in their Eq. 9 that:

$$H = \sum_{c_i, c_{i+1}} P(c_i, c_{i+1}) \log_2 \frac{P(c_{i+1}|c_i)}{P(c_{i+1})} - \sum_{w} P(w) \log_2 P(w),$$

   where w is a word of the corpus and c_i, c_{i+1} a class bigram in the corpus. Show that 5 points

$$\log_2 \frac{P(c_{i+1}|c_i)}{P(c_{i+1})}$$

   is equivalent to

$$\log_2 \frac{P(c_i, c_{i+1})}{P(c_i)P(c_{i+1})}$$

   and hence to mutual information.

3. In this exercise, we will experiment how clustering influences the average mutual information. Let us suppose that we have a (tiny) corpus consisting of 11 tokens and three words: a, b, and c:

   a b a b a c a b a b a

   - Give the unigram and bigram counts: $C(a)$, $C(b)$, $C(c)$, $C(a, b)$, $C(a, c)$, $C(b, a)$, and $C(c, a)$. Give their probabilities. You will not use start-of-sentence symbols. 2 points
   - To cluster the words in two classes, there are three possible mappings: $\{\{a, c\}, b\}$, $\{a, \{b, c\}\}$, and $\{\{a, b\}, c\}$. Which one seems the most natural? Justify your choice. As the entropy term 2 points

$$\sum_{w} P(w) \log_2 P(w),$$

   is equal for the three mappings. We will ignore it in the next question.

- Compute the average mutual information 8 points

$$\sum_{c_i, c_{i+1}} P(c_i, c_{i+1}) \log_2 \frac{P(c_i, c_{i+1})}{P(c_i)P(c_{i+1})}$$

for the three mapping. You will use the exact counts of unigrams and bigrams, not the textbook approximation, where $N \approx N - 1$ for large values of N. You will write out your calculations and you will simplify the logarithms[2]. Do these numbers reflect the intuitive quality of the clustering? Justify your answer.

2.3 Programming

In this section, you will write a Java implementation of the Brown word clustering. You can suppose that you have a corpus of 1 million word and you want to cluster the words into 1,000 classes. The tokens of the corpus, including the punctuation, are separated by a white space.

1. Write a `tokenize` method: 3 points

   ```
   List<String> tokenize(File input)
   ```

   that tokenizes the corpus and returns the list of tokens.

2. Write a `countUnigrams` method: 3 points

   ```
   Map<String, Integer> countUnigrams(List<String> text)
   ```

   that counts the unigrams and stores them in a Java `Map`.

3. Write a `countBigrams` method: 5 points

   ```
   Map<String, Integer> countBigrams(List<String> text)
   ```

   that counts the bigrams and stores them in a `Map`. You will represent the bigrams as a string, where the words will be separated by a space as for instance, `"the table"`.

4. Write an `entropy` function: 5 points

   ```
   double entropy(Map<String, Integer> map)
   ```

   where `map` is the unigram map and that computes

   $$\sum_{w} P(w) \log_2 P(w),$$

   As the entropy is constant for any partition, we will ignore this term in the rest.

5. Let us first suppose that each word defines its own class, that is, there one class per word. Write an `ami` method 8 points

[2] To help you with possible miscalculations, you should find these numbers: $2\log_2 11 + \frac{14}{5}\log_2 5 - \frac{2}{5}\log_2 2 = 0.0175$, $2\log_2 11 - \frac{6}{5}\log_2 7 - \log_2 5 - \frac{4}{5}\log_2 2 = 0.4281$, and $2\log_2 11 - \log_2 5 - \log_2 3 - 2\log_2 2 = 1.0120$.

```
double ami(Map<String, Integer> map, Map<String, Integer> bigramMap)
```

that computes the average mutual information:

$$\sum_{c_i, c_{i+1}} P(c_i, c_{i+1}) \log_2 \frac{P(c_i, c_{i+1})}{P(c_i)P(c_{i+1})}.$$

6. You will now implement one iteration of the Brown et al. (1992) clustering method, where you will compute the average mutual information resulting from the merging of a pair of words. Write a method 12 points

   ```
   Set<String> bestPair(List<String> text)
   ```

   that generates pairs of words and that for each pair computes the average mutual information resulting from the merging of the pair in one class. As a result, you will create a cluster from the pair having the highest average mutual information. To implement the method, you can use a dummy symbol like `#temp`, replace the two words of your pair in the text with this dummy symbol, and compute the average mutual information on the new text.

7. What is the complexity of the method above? Is it a realistic implementation? 2 points

8. Brown et al. (1992) lowered the complexity using a table, $L(c_1, c_2)$, to store the loss of average mutual information resulting from the merging of c_1 and c_2. The computation of the loss is based on the average mutual information of a cluster pair 12 points

   $$Q(c_1, c_2) = P(c_1, c_2) \log_2 \frac{P(c_1, c_2)}{P(c_1)P(c_2)} + P(c_2, c_1) \log_2 \frac{P(c_2, c_1)}{P(c_2)P(c_1)},$$

   where c_1 and c_2 are two clusters.

   Implement a `twoClassMI` method that computes this quantity (Q):

   ```
   double twoClassMI(Set<String> c1, Set<String> c2,
     Map<String, Integer> map, Map<String, Integer> bigramMap)
   ```

   where the clusters are stored in Java sets, `Set`, and the word unigram and bigram counts in `map` and `bigramMap`.

9. At a given iteration of the merging process, denoting V, the vocabulary size, $V - k$, the iteration step, $c_k^1, c_k^2, ..., c_k^k$, the resulting classes, the algorithm selects the pair of classes that minimizes the loss of average mutual information

   $$L_k(c_k^i, c_k^j) = After(i, j, k) - Before(i, j, k)$$

   where

   $$After(i, j, k) = \sum_{m=1}^{k-1} Q(c_k^i \cup c_k^j, c_{k-1}^m)$$

and

$$Before(i, j, k) = \sum_{m=1}^{k} Q(c_k^i, c_k^m) + Q(c_k^j, c_k^m)$$

Outline how to program this function to reduce the complexity. 6 points

10. Finally, Brown et al. (1992, p. 474, 2nd paragraph) provide a way to handle vocabularies of more than 5,000 words. Outline how to program a method implementing this. 6 points

Postscript: Liang (2005) implemented a C++ version of the Brown algorithm. It is available from this site: `http://cs.stanford.edu/~pliang/software/`

References

Brown, P. F., Della Pietra, V. J., deSouza, P. V., Lai, J. C., and Mercer, R. L. (1992). Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–489.

Liang, P. (2005). Semi-supervised learning for natural language. Master's thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts.

Tjong Kim Sang, E. F. and Buchholz, S. (2000). Introduction to the CoNLL-2000 shared task: Chunking. In *Proceedings of CoNLL-2000 and LLL-2000*, pages 127–132, Lisbon.