Exam 2017-06-08

Marco Kuhlmann

This exam consists of three parts:

1. **Part A** consists of 5 items, each worth 3 points. These items test your understanding of the basic algorithms that are covered in the course. They require only compact answers, such as a short text, calculation, or diagram.

   Collected wildcards are valid for this part of the exam. The numbering of the questions corresponds to the numbering of the wildcards.

2. **Part B** consists of 3 items, each worth 6 points. These items test your understanding of the more advanced algorithms that are covered in the course. They require detailed and coherent answers with correct terminology.

3. **Part C** consists of 1 item worth 12 points. This item tests your understanding of algorithms that have not been explicitly covered in the course. This item requires a detailed and coherent answer with correct terminology.

Grade requirements TDDE09: For grade 3, you need at least 12 points in Part A. For grade 4, you additionally need at least 12 points in Part B. For grade 5, you additionally need at least 6 points in Part C.

Grade requirements 729A27: For grade G (Pass), you need at least 12 points in Part A. For grade VG (Pass with distinction), you additionally need at least 12 points in Part B, and at least 6 points in Part C.

Note that surplus points in one part do not raise your score in another part.

Good luck!

Part A

01 **Text classification** **(3 points)**

A Naive Bayes classifier has to decide whether the document 'London Paris' is news about the United Kingdom (class `U`) or news about Spain (klass `S`).

a) State the formula for the Naive Bayes classification rule and explain its parts.

b) Estimate the probabilities that are relevant for the classification of the specific document above from the following document collection using Maximum Likelihood estimation (without smoothing). Answer with fractions.

	document	class
1	London Paris	`U`
2	Madrid London	`S`
3	London Madrid	`U`
4	Madrid Paris	`S`

c) Based on the estimated probabilities, which class does the classifier predict? Explain. Show that you have understood the Naive Bayes classification rule.

d) Practical implementations of a Naive Bayes classifier often use log probabilities. Draw a graph for the function $f(p) = \log p$ in the relevant interval.

02 **Language modelling** **(3 points)**

For the 520 million word Corpus of Contemporary American English, we have the following counts of unigrams and bigrams: *your*, 883,614; *rights*, 80,891; *doorposts*, 21; *your rights*, 378; *your doorposts*, 0.

a) Estimate the probabilities $P(\textit{rights})$ and $P(\textit{rights} \mid \textit{your})$ using Maximum Likelihood estimation (no smoothing). Answer with fractions.

b) Estimate the bigram probability $P(\textit{doorposts} \mid \textit{your})$ using Maximum Likelihood estimation and add-k smoothing with $k = 0.01$. Assume that the vocabulary consists of 1,254,193 unique words. Answer with a fraction.

c) Suppose that we train three n-gram models on 38 million words of newspaper text: a unigram model, a bigram model, and a trigram model. Suppose further that we evaluate the trained models on 1.5 million words of text with the same vocabulary and obtain the following entropy scores: 7.4090, 6.768, and 9.910. Which entropy belongs to which model? Provide a short explanation.

03 **Part-of-speech tagging** **(3 points)**

The following matrices specify a Hidden Markov model in terms of costs (negative log probabilities). The marked cell gives the transition cost from BOS to PL.

	PL	PN	PP	VB	EOS
BOS	11	2	3	4	19
PL	17	3	2	5	7
PN	5	4	3	1	8
PP	12	4	6	7	9
VB	3	2	3	3	7

	hen	vilar	ut
PL	17	17	4
PN	3	19	19
PP	19	19	3
VB	19	8	19

When using the Viterbi algorithm to calculate the least expensive (most probable) tag sequence for the sentence 'hen vilar ut' according to this model, one gets the following matrix. Note that the matrix is missing three values (marked cells).

		hen	vilar	ut
BOS	0			
PL		A	27	21
PN		5	28	35
PP		22	27	20
VB		23	B	36
EOS				C

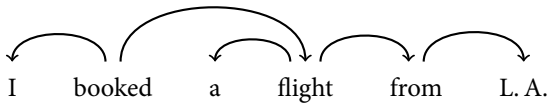
a) Calculate the value for the cell A. Explain your calculation.

b) Calculate the values for the cells B and C. Explain your calculations.

c) Starting in cell C, draw the backpointers that identify the most probable tag sequence for the sentence. State that tag sequence.

04 **Syntactic analysis** **(3 points)**

A transition-based dependency parser analyses the sentence *I booked a flight from L. A.* Here is the gold-standard tree for this sentence.



a) Suppose that the parser starts in the initial configuration for the sentence and takes the transitions `SH`, `SH`, `LA`. State the new configuration. Represent the partial dependency tree by listing its arcs.

b) State a complete sequence of transitions that takes the parser all the way from the initial configuration to a terminal configuration, and that recreates all arcs of the gold-standard tree.

c) Provide a modified transition sequence where the parser mistakingly predicts the arc *booked* → *from* instead of *flight* → *from*, but gets the other dependencies right.

05 **Semantic analysis** **(3 points)**

The Lesk algorithm is a simple method for word sense disambiguation that relies on the use of dictionaries. Here are glosses and examples from Wiktionary for three different senses of the word *course*:

1 A normal or customary sequence. **2** A learning program, as in university. *I need to take a French course.* **3** The direction of movement of a vessel at any given moment. *The ship changed its course 15 degrees towards south.*

a) Which of the three senses of the word *course* does the Lesk algorithm predict in the following sentence, based on the given glosses and examples?

In the United States, the normal length of a course is one academic term.

Ignore the word *course*, punctuation, and stop words. Explain your answer.

b) Change the sentence such that the word *course* maintains its intended sense, but the Lesk algorithm now predicts a different sense than in the previous item.

c) In the extension of the Lesk algorithm known as Corpus Lesk, instead of just counting overlapping words, each word w is weighted as follows:

$$\text{weight}(w) = \log \frac{\text{number of glosses and examples}}{\text{number of glosses and examples which contain } w}$$

Under this scheme, what is the weight of a stop word such as *and*, *the*, or *you*?

Part B

06 **Levenshtein distance** **(6 points)**

The following matrix shows the values computed by the Wagner–Fisher algorithm for finding the Levenshtein distance between the two words *intention* and *execution*. Note that the matrix is missing some values (marked cells).

n	A	8	8	8	8	8	8	7	6	5
o	A	7	7	7	7	7	7	6	5	6
i	A	6	6	6	6	6	6	5	6	7
t	A	5	5	5	5	5	5	6	7	8
n	A	4	4	4	4	5	6	7	7	7
e	A	3	4	B	4	5	6	6	7	8
t	A	3	3	3	4	5	6	6	7	8
n	A	2	2	3	4	5	6	7	7	7
i	A	1	2	3	4	5	6	6	7	8
#	A	A	A	A	A	A	A	A	A	A
	#	e	x	e	c	u	t	i	o	n

a) Define the concept of the Levenshtein distance between two words. The definition should be understandable even to readers who have not taken this course.

b) Provide the values for the cells marked A. Explain.

c) Calculate the value for the cell marked B. Explain. Show that you have understood the Wagner–Fisher algorithm.

07 **Eisner algorithm** **(6 points)**

Here is incomplete pseudocode for the Eisner algorithm.

Please note that in this code, spans of words are indexed by the leftmost word and the rightmost word in the span, and words are indexed by their positions from 1 to n, the total number of words in the sentence. (This is the same indexing scheme as in the extra material, but different from the scheme in the lecture.)

for i **in** $[1, \dots, n]$:
    // two lines missing
for k **in** $[2, \dots, n]$:
    for i **in** $[k-1, \dots, 1]$:
        $T_3[i][k] = \max_{i \leq j < k} \left(T_2[i][j] + T_1[j+1][k] + A[i][k]\right)$
        // three lines missing

The table A holds the arc-specific scores. The tables T_t hold values from the set $\mathbb{R} \cup \{-\infty\}$ and correspond to the four different types of subproblems:

type 1 type 2 type 3 type 4

These tables are initialised with the value $-\infty$.

a) Complete the missing lines.

b) State upper bounds for the memory requirement and the runtime of the Eisner algorithm relative to the sentence length n. Explain your answer.

c) Explain why parsing with the algorithm for transition-based dependency parsing may yield better results than parsing with the Eisner algorithm, despite the fact that the former only performs a greedy search, while the latter solves the relevant optimisation problem exactly.

08 **Named entity tagging** **(6 points)**

Named entity tagging is the task of identifying entities such as persons, organisations, and locations in running text. One idea to approach this task is to use the same techniques as in part-of-speech tagging. However, a complicating factor is that named entities can span more than one word. Consider the following sentence:

Thomas Edison was an inventor from the United States.

In this example we would like to identify the bigram 'Thomas Edison' as a mention of *one* named entity of type 'person' (PER), not two words; similarly, we would like to identify 'United States' as one named entity of type 'location' (LOC).

To solve this problem, we can use the so-called BIO tagging. In this scheme we introduce a special 'part-of-speech' tag for the beginning (B) and inside (I) of each entity type, as well as one tag for words outside (O) any entity. Here is the example sentence represented with BIO tags:

$\text{Thomas}_{\text{B-PER}}$ $\text{Edison}_{\text{I-PER}}$ was_{O} an_{O} $\text{inventor}_{\text{O}}$ from_{O} the_{O} $\text{United}_{\text{B-LOC}}$ $\text{States}_{\text{I-LOC}}$

a) Explain what type of data would be required for training named entity taggers that use the BIO tagging scheme using supervised machine learning.

b) Discuss which type of gold-standard data it is easier to obtain more of: data for part-of-speech taggers or data for named entity taggers.

c) Named entity taggers using the BIO tagging scheme can be evaluated at the level of words ('How many words were tagged with the correct BIO tag?') or at the level of entities ('How many *n*-grams were tagged with their correct entity type?'). Provide a concrete example, similar to the one above, which shows that a system can have a high word-level tagging accuracy but a low entity-level tagging accuracy. Your example should contain at least one named entity of type person (PER). Explain.

Part C

09 **Variations on the Eisner algorithm** **(12 points)**

Explain your solutions in detail!

a) The Eisner algorithm is restricted to parse into projective dependency trees. How could you construct the arc matrix A to let the Eisner algorithm decide whether an input dependency tree is projective? (You can construct the arc matrix differently for each input tree.)

b) To compute the score for an item of type 3, the Eisner algorithm solves the following optimisation problem (see item 07):

$$T_3[i][k] = \max_{i \leq j < k} \left(T_2[i][j] + T_1[j+1][k] + A[i][k] \right)$$

This line contains two operations: The *outer operation* is max, which maximises a term over all possible choices of a word position j. The *inner operation* is +, which adds the scores of two 'simpler' subproblems and the score of an arc.

Suppose that you want to find the total number of possible trees for an input sentence x. How could you solve this problem by using different mathematical operations for the inner and the outer operation of the rules, as well as appropriately constructing the arc matrix A?

c) The Eisner algorithm can be simplified by replacing subproblems of type 3 and subproblems of type 4 with one new type of subproblem, which we may refer to as type 5. To compute the score of subproblems of this type, we combine two 'triangles' without charging the score for an arc:

$$T_5[i][k] = \max_{i \leq j < k} \left(T_2[i][j] + T_1[j+1][k] \right)$$

How do you have to modify the other rules of the Eisner algorithm such that the modified algorithm becomes equivalent to the original one?