Försättsblad till skriftlig tentamen vid Linköpings universitet

Datum för tentamen	2019-03-18
Sal (1)	KÅRA(33)
Tid	14-18
Utb. kod	TDDE09
Modul	TEN1
Utb. kodnamn/benämning **Modulnamn/benämning**	Språkteknologi Skriftlig tentamen
Institution	IDA
Antal uppgifter som ingår i tentamen	8
Jour/Kursansvarig Ange vem som besöker salen	Marco Kuhlmann
Telefon under skrivtiden	013-284644
Besöker salen ca klockan	15
Kursadministratör/kontaktperson (namn + tfnr + mailaddress)	Veronica Kindeland Gunnarsson, 013-285634, veronica.kindeland.gunnarsson@liu.se
Tillåtna hjälpmedel	inga
Övrigt	
Antal exemplar i påsen	

Försättsblad till skriftlig tentamen vid Linköpings universitet

Datum för tentamen	2019-03-18
Sal (1)	KÅRA(1)
Tid	14-18
Utb. kod	729A27
Modul	TEN1
Utb. kodnamn/benämning **Modulnamn/benämning**	Natural Language Processing Skriftlig tentamen
Institution	IDA
Antal uppgifter som ingår i tentamen	8
Jour/Kursansvarig Ange vem som besöker salen	Marco Kuhlmann
Telefon under skrivtiden	013-284644
Besöker salen ca klockan	15
Kursadministratör/kontaktperson (namn + tfnr + mailaddress)	Veronica Kindeland Gunnarsson, 013-285634, veronica.kindeland.gunnarsson@liu.se
Tillåtna hjälpmedel	inga
Övrigt	
Antal exemplar i påsen	

Exam 2019-03-18

Marco Kuhlmann

This exam consists of two parts:

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.

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.

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

Grade requirements TDDE09:

- Grade 3: at least 12 points in Part A
- Grade 4: at least 12 points in Part A and at least 7 points in Part B
- Grade 5: at least 12 points in Part A and at least 14 points in Part B

Grade requirements 729A27:

- Grade G: at least 12 points in Part A
- Grade VG: at least 12 points in Part A and at least 14 points in Part B

Wildcards: When grading the exam, we will credit you with the maximal number of points from up to three wildcards. Wildcards are only valid for Part A. The numbering of the questions corresponds to the numbering of the wildcards.

Good luck!

Part A

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

a) The evaluation of a text classifier produced the following confusion matrix. The marked cell gives the number of times the system classified a document as class A whereas the gold-standard class for the document was B.

	A	B	C
A	58	6	1
B	5	11	2
C	0	7	43

Set up fractions for the following values:

i. accuracy

ii. precision with respect to class B

b) Complete the learning algorithm for the (un-averaged) multi-class perceptron.

for each class c **do**
  $\boldsymbol{w}_c \leftarrow \mathbf{0}$
for each epoch **do**
  for each training example $(\boldsymbol{x}, y)$ **do**
    ...

c) A logistic regression classifier has been trained on a sentiment analysis data set. The data set consists of 8,544 annotated sentences extracted from movie reviews; each sentence has been annotated as either 'positive', 'negative', or 'neutral' towards the movie at hand. The classifier's vocabulary consists of 16,361 unique words, including a special 'word' for out-of-vocabulary items.

The trained classifier is used to predict the rating for an unseen text, which is translated into a bag-of-words vector $\boldsymbol{x}$ ('featurized') and fed into the logistic model at the core of the classifier, specified by the following equation.

$$\hat{\boldsymbol{y}} = \text{softmax}(\boldsymbol{x}\boldsymbol{W} + \boldsymbol{b})$$

State the dimensions (numbers of rows and columns) of the input vector $\boldsymbol{x}$, the weight matrix $\boldsymbol{W}$, the bias vector $\boldsymbol{b}$, and the output vector $\hat{\boldsymbol{y}}$. Answer with concrete numbers.

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

The Corpus of Contemporary American English (COCA) is the largest freely-available corpus of English, containing approximately 560 million tokens. In this corpus we have the following counts of unigrams and bigrams:

snow	*white*	*white snow*	*purple*	*purple snow*
38,186	256,091	122	11,218	0

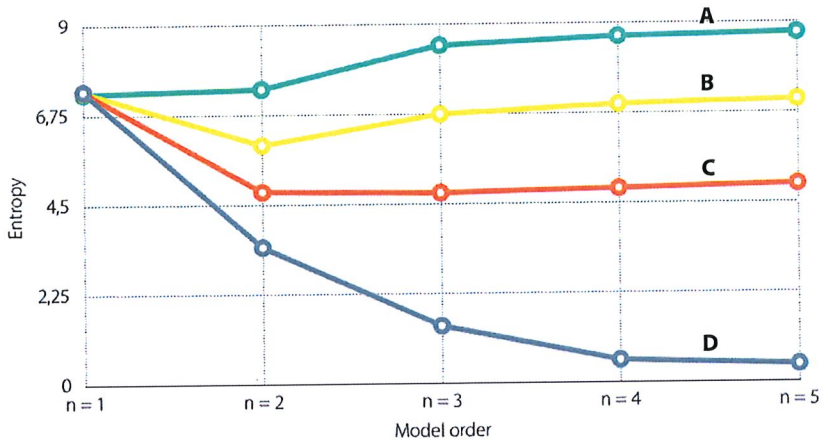
a) Estimate the following probabilities using maximum likelihood estimation without smoothing. Answer with fractions containing concrete numbers.

i. $P(snow)$ ii. $P(snow \mid white)$

b) Estimate the following probabilities using absolute discounting with $d = 0.2$. Assume that the vocabulary consists of 1,254,100 unique words, that 99% of these words are observed at least once, and that the number of unique words observed after the word *purple* is 2,462. Answer with fractions containing concrete numbers.

i. $P(snow)$ ii. $P(snow \mid purple)$

c) We use maximum likelihood estimation with add-k smoothing to train n-gram models on the COCA corpus, with $n \in \{1, \ldots, 5\}$ and $k \in \{0, 0.01, 0.1, 1\}$. The following graph shows the entropy of each trained model on the training data. Which series corresponds to which k-value, and why? Answer with a short text.



03 Part-of-speech tagging (3 points)

Here is a Hidden Markov model (HMM) specified 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

	they	freak	out
PL	17	17	4
PN	3	19	19
PP	19	19	3
VB	19	8	19

a) Compute the cost (negative log probability) of the following tagged sentence:

they	freak	out
PN	VB	PP

b) When using the Viterbi algorithm to calculate the least expensive (most probable) tag sequence for the same sentence, one gets the following matrix. Calculate the missing value (marked cell).

	they	freak	out
BOS	0		
PL	28	27	21
PN	5	28	35
PP	22	27	20
VB	23	14	36
EOS			

c) Consider a left-to-right part-of-speech tagger based on the neural architecture presented in class. The tagger has the following feature templates: current word, previous word, next word, and tag of the previous word. The size of the word vocabulary is 19,674; the size of the tag vocabulary is 18. The tagger uses a word embedding of width 100 and a tag embedding of width 10. How long is an input vector to the feedforward part of the network?

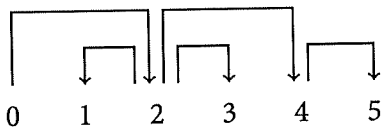
04 Syntactic analysis (3 points)

a) Draw all non-projective dependency trees for the sentence '1 2 3' in which the artificial root vertex 0 has exactly one child.

b) State the following asymptotic complexities for the Collins algorithm, measured in terms of the number of words in the sentence, n.

i. memory requirement ii. runtime requirement

c) State two different sequences of transitions that make the transition-based dependency parser produce the following dependency tree:



05 Semantic analysis (3 points)

a) Consider the following co-occurrence matrix. Rows correspond to target words, columns correspond to context words.

	butter	cake	school	cow	deer
cheese	12	2	0	1	0
bread	5	5	0	0	0
goat	0	0	0	6	1
sheep	0	0	0	7	5

Set up fractions for the following PPMI values:

i. PPMI(sheep, butter) ii. PPMI(goat, cow)

b) Explain why the cosine similarity of two word vectors read off from a PPMI matrix is never negative. Answer with a short text.

c) In a certain set of word embeddings, the two nearest neighbours of the word *fast* are *quick* and *slow*. Explain why this is not an unexpected result. Answer with a short text.

Part B

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

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) Compute the Levenshtein distance between the two words *leda* and *deal* using the Wagner–Fischer algorithm. Your answer should contain both the distance itself and the complete matrix.

c) It is much more likely for a user to mistype the word *deal* as *seal* than as *beal*; this is because the keys for the letters *d* and *s* are much closer to each other on the keyboard than the keys for the letters *d* and *b*. Explain how the Wagner–Fischer algorithm could be adapted to take this information into account.

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

Here is incomplete pseudocode for the Eisner algorithm:

for i **in** $[0, \ldots, n]$:
    // two lines missing
for k **in** $[1, \ldots, n]$:
    for i **in** $[k-1, \ldots, 0]$:
        $T_4[i][k] = \max_{i \leq j < k} (T_2[i][j] + T_1[j+1][k] + A[i][k])$
        // 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) Provide pseudocode for the structured perceptron algorithm for learning a dependency parser under an arc-factored model. Explain your notation. Make it clear how the Eisner algorithm is used in structured perceptron training.

c) Computing the score for an item of type 4 involves two operations: The *outer operation* is max, which maximises over all possible choices of 'split points'. The *inner operation* is +, which adds the scores of two 'simpler' subproblems, as well as the score of an arc.

Suppose now that instead of finding the highest-scoring dependency tree, you want to find the total number of possible trees for a sentence. Describe how you could solve this problem using a variant of the Eisner algorithm that employs different inner and outer operations.

08 **The arc-hybrid system** **(6 points)**

Note: For this item, consider dependency trees without an artificial root vertex, that is, trees where the root vertex may take any position between 1 and n.

The *arc-hybrid* system has the same configurations and the same initialisation and termination conditions as the arc-standard system that you know from class, but makes use of a different LA transition. Let us denote a configuration as $c = (\sigma, \beta, A)$, where σ is the stack, β is the buffer, and A is the set of already constructed dependency arcs. Then the three transitions of the arc-hybrid system can be defined as follows:

$$(\sigma, b|\beta, A) \rightarrow (\sigma|b, \beta, A) \qquad \text{SH}$$
$$(\sigma|s_1|s_0, \beta, A) \rightarrow (\sigma|s_1, \beta, A \cup \{s_1 \rightarrow s_0\}) \qquad \text{RA}$$
$$(\sigma|s, b|\beta, A) \rightarrow (\sigma, b|\beta, A \cup \{b \rightarrow s\}) \qquad \text{LA}$$

a) Demonstrate your understanding of the arc-hybrid system:

   i. State a sequence of transitions that make an arc-hybrid parser produce the dependency tree from item 04 c).

   ii. How many transitions does an arc-hybrid parser make when processing a sentence with n words? State your answer as a function of n.

b) In the arc-standard system, each valid transition sequence X has one of the following three forms:

   SH X X RA X X LA (where X is another valid sequence)

   Provide a similar characterisation for the arc-hybrid system. What does your characterisation tell you about the relation between the number of valid transition sequences in the two systems?

c) Arc-hybrid parsers are trained using oracles that in each configuration c pick a transition t with *lowest cost*. Here, the cost is defined as the smallest number of gold-standard arcs that will become impossible for the parser to construct after taking the transition, no matter what it does later on. For example: Doing SH in a configuration $c = (\sigma|s_0, b|\beta, A)$ means that b will no longer be able to acquire heads from $H = \sigma$, nor dependents from $D = \{s_0\} \cup \sigma$. Therefore, the cost in this case is the number of gold-standard arcs of the form $b \rightarrow d$ and $h \rightarrow b$, for $h \in H$ and $d \in D$. Provide a similar reasoning for

   i. the cost of LA out of a configuration $c = (\sigma|s_0, b|\beta, A)$

   ii. the cost of RA out of a configuration $c = (\sigma|s_1|s_0, \beta, A)$