Exam solutions IN4080, autumn 2023

You can find in blue the proposed solutions. It should be stressed that there may be more than one correct solution to some of the exam questions.

Q1 – Types and tokens

Imagine a newspaper article written in Norwegian*, which you run through a lemmatizer. You will thus have two versions of the article, the original and the lemmatized one.

Does the **type-token-ratio** differ significantly between the two versions? If so, which of the two is higher? Justify your answer by explaining how the number of tokens and the number of types differs.

(* You can also answer the same question for a different language, e.g. English. If so, specify the selected language and describe shortly how the choice of language can affect the results.)

- The number of tokens remains roughly identical after lemmatization. There may be rare cases where one token is split into two lemmas, but I cannot think of an example in Norwegian.
- The number of types (unique word forms) is diminished by lemmatization. A typical noun will be represented by up to four types in the original version (singular/plural, definite/non-definite), but only by one type in the lemmatized version. For verbs and adjectives, the reasoning is similar.
- The type-token-ratio is defined as nTypes / nTokens. Assuming that nTokens is constant, the original version will have a higher TTR and the lemmatized version a lower TTR.

Q2 – F-measure

Consider a classifier A whose precision is 85.1% and recall is 84.9%. Classifier B has a precision of 89.0% and a recall of 81.0%. Which classifier obtains the higher F1-score?

Classifier A: (2 * 0.851 * 0.849) / (0.851 + 0.849) = 1.445 / 1.7 = 0.85 = 85.00%

Classifier B: (2 * 0.89 * 0.81) / (0.89 + 0.81) = 1.4418 / 1.7 = 0.8481 = 84.81%

Classifier A obtains the higher score.

Q3 – Averaging

The following table represents the confusion matrix of a three-class sentiment classifier with labels Positive, Neutral and Negative:

		Gold		
		Positive	Neutral	Negative
Predicted	Positive	20	2	4
	Neutral	0	16	2
	Negative	0	1	14

This matrix can be separated into class-specific 2-by-2 matrices, as illustrated for the Positive class below:

		Positive Gold	
		True	False
Positive Predicted	True	20	6
	False	0	33

Answer the following questions:

1. Compute the 2-by-2 matrices for the Neutral and Negative classes.

		Neutral Gold	
		True	False
Neutral Predicted	True	16	2
	False	3	38

		Negative Gold	
		True	False
Negative Predicted	True	14	1
	False	6	38

2. What is the micro-average precision of the classifier?

Aggregate the three 2-by-2 matrices:

		All Gold	
		True	False

All Predicted	True	50	9
	False	9	109

Precision = 50 / (50+9) = 84.75% (50+9 is the sum of the first row)

3. What is the micro-average recall of the classifier?

Recall = 50 / (50+9) = 84.75% (50+9 is the sum of the first column)

4. What is the relationship between the micro-average precision and recall values? Under what conditions does this relationship hold?

The two values are identical.

This holds under all conditions examined here. This can be demonstrated by assigning the following variables to the values in the original 3-by-3 matrix:

		Gold		
		Positive	Neutral	Negative
Predicted	Positive	a	b	c
	Neutral	d	e	f
	Negative	g	h	i

The aggregated 2-by-2 matrix looks like this:

		All Gold	
All Predicted		True	False
	True	a+c+i	b+c+d+f+g+h
	False	d+g+b+h+c+f	…

This shows that the number of false negatives is identical to the number of false positives.

The relationship doesn't hold in multi-label classification tasks, i.e. where for a given sample more than one label is correct. But we haven't discussed multi-label classification in class, so this aspect is not required to obtain full points.

Q4 – Naïve Bayes

The following formula describes the prediction function of a Naïve Bayes classifier:

$$\hat{c} = \arg\max_{c \in C} \left(P(c) \cdot \prod_{i=1}^{n} P(f_i|c) \right)$$

Give a short description of the formula by focusing on the following parameters:

- What is $\hat{c}$, c and C?

$\hat{c}$ represents the predicted class label.

c represents any class label available for the current classification task.

C represents the set of all class labels available for the current classification task.

- What is f_i and n ?

f_i is the i-th feature in the list of features used for classification. In a bag-of-words text classifier, each feature typically corresponds to one word. n is the total number of features in the classifier. In a bag-of-words model, this corresponds to the number of distinct words (after stopword removal).

- What is meant by *arg max*?

Arg max returns the class associated with the maximum value. So, for each class c, the probability is computed, and then the maximum of all probabilities is selected. Max returns the probability value, while arg max returns the class to which the maximum probability corresponds.

- What does $P(c)$ represent?

$P(c)$ corresponds to the prior. It represents the overall probability of the class according to the training data, without looking at the features in detail.

Q5 – Logistic regression

You are given a logistic regression model for three classes: A, B and C. Your current model parameters are w = {wA, wB, wC} where wi is the weight vector for class i:

wA = [1, 1.2, -2, 1.5, 1]

wB = [-2, 3, 1, 0, -2]

wC = [0, -3, 0, -2, 5]

You are additionally given an example whose feature vector is x = [0, 1, 0, 1, 1]. Compute p(i|x; w) for each of the classes i.

First compute dot products:

- wA . x = 1.2 + 1.5 + 1 = 3.7
- wB . x = 3 + 0 + (-2) = 1
- wC . x = (-3) + (-2) + 5 = 0

Then exponentiate:

- A: $e^{3.7} = 40.44$
- B: $e^{1} = 2.72$
- C: $e^{0} = 1$

Take the sum:

- 40.44 + 2.72 + 1 = 44.16

Finally normalize:

- P(A|x) = 40.44/44.16 = 0.9158 = 91.58%
- P(B|x) = 2.72/44.16 = 0.0616 = 6.16%

- P(C|x) = 1/44.16 = 0.0226 = 2.26%

Q6 – Sequence labeling

Consider the two sentences

1. February made me shiver.

2. February gave me shiver.

and the two tag sequences

a) NOUN VERB PRON VERB

b) NOUN VERB PRON NOUN

It can be argued that the best tag sequence for (1) is (a) and for sentence (2) it is (b).

- What conditions does a **HMM tagger** need to fulfill to assign the two correct tag sequences?

The only way for HMM taggers to include context is through transition probabilities, i.e. the previously predicted labels. A **bigram** HMM tagger uses one previous label, PRON in both cases, and this is unable to distinguish the two sentences. A **trigram** HMM tagger uses two previous labels, VERB+PRON in both cases, and is also unable to distinguish the two sentences. HMM taggers cannot include information about previous **words**. It could work by using a larger tagset that distinguishes the two types of verbs.

- What conditions does a **greedy perceptron tagger** need to fulfill to assign the two correct tag sequences?

A greedy perceptron tagger can include context word features at positions n-1 and n-2, which allow it to perform correctly.

Q7 – Sequence labeling

Consider the following training corpus for a simplified part-of-speech tagging problem (N stands for noun and V for verb):

dogs/N eat/V fish/N
cats/N eat/V mice/N
cats/N like/V fish/N
dogs/N fish/V

Consider a **bigram HMM tagger** trained on this corpus:

- What are the emission probabilities (with add-one smoothing)?
- What are the transition probabilities (with add-one smoothing)? Also include the start symbol * and the end symbol †.
- What is the probability of predicting the label sequence N V N for the sentence *cats fish fish*?

Emission probabilities (without :: with smoothing):

P(dogs|N) = 2/7 :: 3/13 P(dogs|V) = 0/4 :: 1/10

P(cats|N) = 2/7 :: 3/13	P(cats|V) = 0/4 :: 1/10
P(fish|N) = 2/7 :: 3/13	P(fish|V) = 1/4 :: 2/10
P(mice|N) = 1/7 :: 2/13	P(mice|V) = 0/4 :: 1/10
P(eat|N) = 0/7 :: 1/13	P(eat|V) = 2/4 :: 3/10
P(like|N) = 0/7 :: 1/13	P(like|V) = 1/4 :: 2/10

Transition probabilities (without :: with smoothing):

P(N|*) = 4/4 :: 5/7 or 5/6	P(N|N) = 0/7 :: 1/10	P(N|V) = 3/4 :: 4/7
P(V|*) = 0/4 :: 1/7 or 1/6	P(V|N) = 4/7 :: 5/10	P(V|V) = 0/4 :: 1/7
P(†|*) = 0/4 :: 1/7 or 0	P(†|N) = 3/7 :: 4/10	P(†|V) = 1/4 :: 2/7

P(†|*) can be left at 0.

Cats/N fish/V fish/N:

P(N|*) * P(cats|N) * P(V|N) * P(fish|V) * P(N|V) * P(fish|N) * P(†|N)

= 5/7 * 3/13 * 5/10 * 2/10 * 4/7 * 3/13 * 4/10

= 7200 / 8281000 = 0.00086946

Q8 – Lexical semantics

Which semantic relations hold between the words in each of the following pairs?

- apple – fruit -- → Hyper/hyponymy (apple is hyponym, fruit is hypernym)
- apple – cider → Relatedness
- apple – orange → Similarity
- enormous – tiny → Antonymy
- enormous – colossal → Synonymy

Q9 – Cosine similarity

Assume the following simplified word vectors.

elephant - (1, 3)

monkey - (2, 1)

horse - (4, 5)

Report the cosine similarities between *elephant* and *monkey*, and between *elephant* and *horse*. Which of the two words *monkey* or *horse* is more similar to *elephant*?

Elephant-monkey: $\frac{E\cdot M}{\|E\|\cdot\|M\|} = \frac{5}{\sqrt{10}\cdot\sqrt{5}} = \frac{5}{7.07} = 0.707$

Elephant-horse: $\frac{E \cdot H}{\|E\| \cdot \|H\|} = \frac{19}{\sqrt{10} \cdot \sqrt{41}} = \frac{19}{20.246} = 0.938$

Horse has higher similarity than monkey and is more similar to elephant.

Q10 – Word vectors

In the course, we have seen two ways of representing words as vectors based on their distribution in a corpus:

- Word-context matrices based on co-occurrence counts.
- Word embeddings obtained with the skip-gram with negative sampling approach.

Describe shortly the main ideas of the two approaches. In particular, compare the two approaches with respect to the form of the vectors and how the vectors are derived.

Aspects that can be mentioned:

- Word-context matrices are derived directly and deterministically from a corpus, possibly with normalization, essentially co-occurrence counts
- SGNS representations are based on a corpus, but not directly because of sampling; use a neural network in a binary classification task
- Sparse (=mostly zeros) vs dense vectors
- Interpretable vs non-interpretable vectors
- Dimensionality given by data vs fixed ad-hoc
- Sensitivity to similarities between context words

Q11- Transformer

How does self-attention in a Transformer encoder differ from self-attention in a Transformer decoder?

Encoder self-attention covers the entire sequence, i.e. both the words to the left of the current position and the words to the right of the current position.

Decoder self-attention only covers the words to the left of the current position (because the words to the right are not yet known at test time).

Q12- Transformer

The Transformer uses so-called **multi-head attention.**

1. What does a head represent in this context?

A head refers to one “way” of attending the tokens of the sentence, i.e. a particular pair of key and query vectors.

2. What are, in your opinion, the advantages of using multiple heads compared to a single head?

Different heads can focus on different (linguistic) aspects. For example, one head can represent syntactic relations, another one can track co-reference, a third one can focus on more semantic relations. With a single head, all aspects would need to be dealt with by the same attention mechanism.

Q13 – Dialogue foundations

Explain what a deictic is and provide at least 3 examples of deictics.

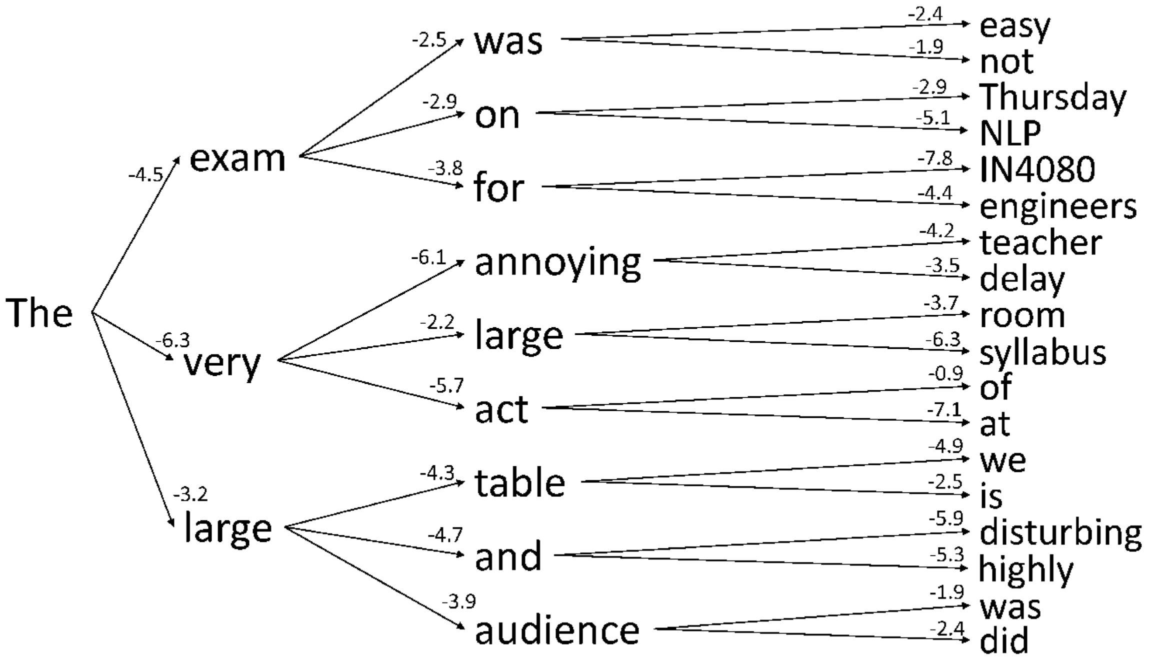
A deictic is a linguistic expression whose meaning depends on the time and/or place of occurrence. Examples of deictics are temporal expressions such as “now”, spatial expressions such as “there, or first and second-person pronouns.

Assume you are developing a talking robot and wish to use a large language model (LLM) to generate the robot responses to the user. Can the presence of deictics in the user utterances create problems for your LLM? Explain in 2-3 sentences.

Yes, an LLM has no direct access to contextual information about the time and place of occurrence of a given input, expect if this information is explicitly provided in the prompt. The LLM will therefore be unable to understand what a word like “there” or “yesterday” actually refers to, except if one explicitly includes sentences that describes the current context the robot finds itself in.

Q14 – Decoding

Assume we have a large language model (LLM) used to generate continuations after the word "The". The possible continuations are shown in the attached image. Each branch in this tree of possible continuations is associated with a given log-probability.



Based on this tree, search for the most likely continuation according to the two following strategies:

1. **Greedy search**

Greedy search will simply select the most likely continuation token at each step, which correspond to the token will highest log-probability.

Step 1: The *large* → log-prob: -3.2

Step 2: The large *audience* → log-prob: -7.1 (for the two tokens)

Step 3: The large audience *was* → log-prob: -9.0 (for the three tokens)

2. **Beam search with a beam of size 2**

With beam search, we keep at each time a set of k hypotheses. At each step, we consider all possible continuations for each hypothesis, and retain only the k most likely (hypothesis + continuation).

Step 1:

- Hypothesis 1: The *large* → log-prob -3.2
- Hypothesis 2: The *exam* → log-prob -4.5

Step 2:

We have a set of 6 possible continuations:

- The exam *was* → log prob: -7 (for the two tokens)
- The exam *on* → log-prob: -7.4
- The exam *for* → log-prob -8.3
- The large *table* → log-prob -7.5
- The large and → log-prob: -7.9
- The large audience → log-prob: -7.1

We retain the 2 best hypotheses in the beam, namely:

- Hypothesis 1: The exam *was* → log prob: -7
- Hypothesis 2: The large audience → log-prob: -7.1

Step 3:

We have a set of 4 possible continuations:

- The exam was *easy* → log prob: -9.4
- The exam was *not* → log prob: -8.9
- The large audience *was* → log-prob: -9.0
- The large audience *did* → log-prob: -9.5

Again, only the two best hypotheses are retained:

- Hypothesis 1: The exam was *not* → log prob: -8.9
- Hypothesis 2: The large audience *was* → log-prob: -9.0

The best continuation would then be “The exam was not”, with a total log-probability of -8.9.

Q15 – Reinforcement learning

Explain in a few sentences the difference between the reward function R(s,a) and the Q-value Q(s,a) in reinforcement learning.

The reward function specifies the *immediate* reward received by the system when executing action a in a given state s, while the Q-value corresponds to the *expected cumulative reward* over a (potentially infinite) time horizon, as stipulated by the Bellman equation:

$$Q(s,a) = R(s,a) + \gamma \sum_{s'} P(s'|s,a) \max_{a'} Q(s',a')$$

In other words, the while the R value specifies the reward the agent can obtain here and now by executing action a in state s, the Q-value describes the accumulation of rewards we can expect now *and in the future* if we execute action a.

Is it possible for the reward function R(s,a) to have a negative value while the Q-value Q(s,a) for the same state s and action a is positive? Use a short example to illustrate your answer.

Yes, it is possible to have an action with a negative reward in the short-term but which leads us to a state that will yield higher rewards in future timesteps.

For instance, in a dialogue system, a clarification request may have a small negative reward (we may annoy the user), but a positive Q-value, as the clarification request will lead us to a state where we have a higher confidence in the user intent.

Q16 – Dialogue system evaluation

How can we evaluate task-based dialogue systems? Describe possible approaches and highlight their advantage and limitations. (2-3 paragraphs).

Here one may answer in multiple ways, but here are some salient points:

1. There is no single “standard” way to evaluate a dialogue system, although multiple evaluation metrics have been proposed. For a task-based system, measuring how often the dialogue system manages to successfully complete its task (i.e. book a flight ticket) is obviously an important factor.

2. To get a more detailed picture of the system performance, we should rely on *user satisfaction studies* where users are tasked to interact with the system to accomplish a given task. After each interaction, the users are then asked to fill a user survey about the extent to which they were satisfied with the system. One can for instance ask whether the system understood the user requests, whether the system questions were relevant and easy to understand, how the interaction pace was, etc.
3. The main advantage of user evaluations is that they assess the system performance in the context of actual interactions with real human users. However, when developing a new dialogue system, it also means that a new evaluation would need to be conducted for each system iteration. User ratings are also subjective and may vary from person to person.
4. Instead of relying on subjective user ratings, one can also take advantage of performance heuristics that can be automatically extracted from dialogues, such as the dialogue length, number of clarification requests, or task completion. Those measures are indeed often correlated with user ratings – which means that if we improve the performance on those measures, we likely also improve the user satisfaction.
5. Those performance heuristics can be grouped into 3 groups, namely task completion success (for instance completion ratio), efficiency costs (e.g. nb of turns, elapsed time)., and quality costs (e.g. number of ASR errors, number of clarification requests, etc.).
6. If one has a corpus of existing dialogues, one can also in principle compare the response produced by the system with the response found in the dialogue corpus, using metrics such as BLEU. This is something that has been done in many papers. However, it turns out to be a bad idea, as experimental studies have shown very weak correlation between such metrics and actual user satisfaction.
7. There are, however, alternative metrics that do correlate better, such as ADEM (Lowe et al, 2017). One can also use observer evaluations where the system output are rated by a third party annotator.

Q17 – Social biases

During the lecture on ethics, we reviewed the approach proposed by Bolukbasi (2016) to "debias" word embeddings. One of the proposed steps focuses on "neutralizing" words that are not definitional. Explain in a few sentences the motivation behind this neutralisation step, and how it is performed.

The problem tackled by Bolukbasi (2016) is that word embeddings incorporate many social stereotypes and biases. For instance, a word like “scientist” will often be more closely associated with male than female attributes, while the opposite is true for a word like “childcare”. Those words should ideally not contain any gender information (in other words, gender is not part of the definition of the word, in contrast to “grandmother” or “pregnant”). The goal of Bolukbasi is therefore to neutralize the stereotypical part of those word embeddings.

This neutralization is done by:

1) identifying the bias direction. This can be achieved by taking a set of words that are definitional for the demographic division of interest, for instance boy-girl, mother-father for the “gender” bias. We can then identify the bias direction by taking the average of the difference between the word embeddings for those pairs.
2) Setting the non-definitional words like “scientists” or “childcare” to zero in the bias direction.

Q18 – data privacy

Explain the difference between direct identifiers and quasi-identifiers in the field of data privacy and give a few examples of both direct identifiers and quasi-identifiers.

A direct identifier is an information that can be used to univocally identify a specific individual, for instance a full person name, mobile phone number or home address.

A quasi-identifier, on the other hand, is an information that is not sufficient to single out a specific individual when seen in isolation but may lead to re-identification when combined with other quasi-identifiers and background knowledge. For instance, while gender, date of birth and municipality are not sufficient to single out a person when considered each on its own, they will often do so when they are provided together.