Lab Class NLP: Words

By 11.06.2020 solutions for the following exercises have to be submitted: 1, 2

Exercise 1 : Part-of-Speech Tags (3+1 Points)

You are given the following three sentences:

Sentence 1	**UD**	**Penn**	**Sentence 2**	**UD**	**Penn**	**Sentence 3**	**UD**	**Penn**
It			Telling			Research		
is			good			on		
sunny			jokes			adult-learned		
throughout			is			second		
the			an			language		
year			art			(		
.			that			L2		
			comes			)		
			naturally			has		
			to			provided		
			some			considerable		
			people			insight		
			,			into		
			but			the		
			for			neurocognitive		
			others			mechanisms		
			it			underlying		
			takes			the		
			practice			learning		
			and			and		
			hard			processing		
			work			of		
			.			L2		
						grammar		
						.		

(a) Determine the Part-of-Speech tags for each word of all three sentences with both, the original Penn Treebank and the Universal Dependency (UD) tagsset [2].

(b) What is the major difference between the Penn and the UD tagset? Why were the changes form Penn to UD necessary for the mission of Universal Dependencies?

Exercise 2 : N-gram Language Models (3+1+1+1 Points)

A language model assigns probabilities to words based on the context they are observed in. Language models are used for many NLP and IR problems, such as text generation, document ranking, or grammar correction. Formally, a language model determines the probablility of the next word w_n from the preceeding sequence of words w_1^{n-1}:

$$P(w_n|w_1^{n-1}) = P(w_1)P(w_2|w_1)P(w_3|w_1^2)...P(w_n|w_1^{n-1}) = \prod_{k=1}^{n} P(w_k|w_1^{k-1}).$$

N-gram-based language models approximate the probability of w_n by only considering the history of a few words instead of the complete history. For example, a bigram model would estimate $P(w_n)$ only from the preceeding word $P(w_n|w_{n-1})$, a trigram model from $P(w_n|w_{n-2}^{n-1})$, in general $P(w_n|w_{n-N+1}^{n-1})$, where N is is length of the n-gram. The respective probabilities for each can be calculated from the counts $C(w)$ of the n-grams:

$$P(w_n|w_{n-N+1}^{n-1}) = \frac{C(w_{n-N+1}^{n})}{C(w_{n-N+1}^{n-1})}$$

For example, the probability of `is` following after observing `this` in a bigram model is calculated as:

$$P(\text{is}|\text{this}) = \frac{\text{count of } \texttt{this is}}{\text{count of } \texttt{this}}.$$

For a trigram model, it is:

$$P(\text{easy}|\text{this is}) = \frac{\text{count of } \texttt{this is easy}}{\text{count of } \texttt{this is}}.$$

You can find more information on language models here.

(a) Download the sample of the Corpus of Contemporary American English (COCA), concatenate all files and preprocess them by removing all `@`, `<p>`, and `\n` and lowercase the text. Tokenize the sample by splitting the preprocessed text on whitespaces. Now, build a trigram language model by counting all bi- and trigrams of the sample and then compute all $P(w_n|w_{n-2}^{n-1})$.

(b) Use the language model to compute the probability of the sentences below. For this, segment the text into trigrams, produce the probabilities for each trigram from your language model and then multiply all resulting probabilities to get the likelihood of the whole sentence being produced.

"he is from the east ."
"she is from the east ."
"he is from the west ."
"she is from the west . "

Note: Multiplying very small n-gram probabilities for longer sentences quickly becomes a numerical problem. In practice, we would convert the probabilities to e-base log-space for all computation:

$$P(w_1) \cdot P(w_2) \cdot P(w_3) = exp(logP(w_1) + logP(w_2) + logP(w_3)).$$

This conversion is also used frequently in machine learning and optimization since sums are easier to differentiate and the conversion to log-space conserves the optima of the function.

(c) Use your language model to complete the sentences listed below. For this, take the trailing two tokens from the sentence, extract the most likely next token from your language model, and add it to the sentence. Repeat this procedure until your language model predicts a ".".

"the adventure of"
"a student is"

(d) Try the operations from (b) and (c) with your own sentences. What limitations of this language model do you notice? Name at least two improvements you could do to circumvent these limitations.