# EDAN20 Language Technology
# -Lab 5.

Nils Romanus

September 2022

# 1 Google's Neural Machine Translation System

Previously in the course, we have tokenized text using a rule-based technique using the space character as a delimiter between words. The tokens are generated using a probabilistic technique instead of a rule-based one with the aim of generating a corpus that is minimal and chosen in accordance with a maximum-likelihood language model. The tokens will be learned by firstly generating possible word-piece candidates from the existing text and then assigning each one a probability based on their occurrence in the corpus. This candidate to probability mapping is subsequently used to tokenize the text by segmenting it in a way that maximizes the likelihood computed using the aforementioned mapping. The main advantage of using such techniques is that some languages do not have a clear delimiter, e.g a space, between words. Using a maximum-likelihood language model is therefore advantageous in these languages. Furthermore, language is an evolving concept and a maximum-likelihood model handles future words or special characters better than a delimiter-based tokenizer.

# 2 Design of the BPE Algorithm

Bite-pair-encoding (BPE) works by firstly generating a sequence of all characters in a text and setting the desired vocabulary size. The adjacent characters that are most common are subsequently merged into one element of the sequence. This is repeated until the desired vocabulary size is reached. This is exactly what is done in our program the set of all characters in the corpus with an exception for the space symbol.

# 3 BPE Tokenizer

The generated BPE vocabulary can be used to tokenize a text. This is done by firstly sorting the vocabulary primarily on the token length and alphabetically secondarily. This vocabulary list can be transformed into a regex pattern where the elements in the list of tokens are joined together with a disjunction operator meta character. The regex find all function can then be called on the entire corpus using the regex pattern generated from the BPE.

# 4 Unigram Language Model

The unigram language model works by computing the probability of a sub-word sequence using a pre-determined vocabulary. It assumes the occurrence of all sub-words is independent and the probability of a sequence can therefore be generated by computing the product of the probabilities of all sub-words. The probability of each sub-word can be estimated by computing its relative occurrence in the corpus. The vocabulary of sub-words can be generated using multiple methods. One alternative is to create the vocabulary using BPE as

explained in the previous section. Another alternative is to use a maximum-likelihood approach. Firstly set the vocabulary size and start with all the sub-words occurring more than once in the corpus. Then iteratively remove the token for which the *reduction* in loss is maximal. Where loss in this context refers to the reduction in likelihood resulting from the token being removed from the vocabulary. After having computed the loss for each word in the vocabulary remove the ones that reduce the likelihood the most, i.e the ones with the highest loss. This can be done by keeping a fraction as in Kudo or removing it until we reach a set vocabulary size as in Bostrom and Durett. Using the generated vocabulary tokenization of a text can subsequently be performed by generating all possible sequences of sub-words in a text and then computing the sum of the log-likelihood, using the constructed vocabulary, for these potential segmentation sequences. The preferred segmentation sequence is the one that maximizes this log-likelihood.