# Contents

# Chapter 5

# Word Alignment

We now reach the core of thesis, computing word alignments using the novel method SimAlign suggested by Jalili Sabet et al. 2020 and evaluating it against a baseline method.

## 5.1 Introduction

Following the success statistical models had in the task of sentecne alignemnt, word alignment was seen as a natural extension of that work. This work had two main goals: offer a valuable resource in bilingual lexicography and develop a system for automatic translation (Brown et al. 1993). Word alignments are objects indicating for each word in a string in the target language $f$ which word in the source language $e$ it arose from(Brown et al. 1993). In other words, it is a mapping of words in a string of the source language $e$ to the words in a string of the target language $f$ (Koehn 2009, p. 84).

A simple example for an alignment for a pair of sentences from the corpus I compiled are the German sentence *Die Beratungen sind kostenlos* "The consultations are gratuitous" and its Romansh counterpart *Las cussegliaziuns èn gratuitas*.

1: Die 2: Beratungen 3: sind 4: kostenlos

1: Las 2: cussegliaziuns 3: èn 4: gratuitas

In this example, each word in German is aligned to exactly one word in Romansh and the words follow exactly the same order, such that the resulting alignment is the set of mappings $\{1 \rightarrow 1, 2 \rightarrow 2, 3 \rightarrow 3, 4 \rightarrow 4\}$. Such alignments, in which each word in the source sentence is aligned exactly one word in the target sentence in which the words follow the same order are considered simple (Koehn 2009, p. 85).

Things become more compliacted when word order differs between languages or when several words in one sentence are mapped to one or several words in the other sentence. A word in the target language may be aligned to several words in the source language (1-to-many), or several words in the target language may be aligned to one word in the source language (many-to-1). Sometimes words in the target have no realtion to the source (for instance in case of untranslatable words, but words might also generally be omitted in the translation). In that case, they will be aligned to a special `NULL` token (Koehn 2009, p. 85).

In order to deal with these challenges of different word order and alignments that are not 1-to-1 alignments, Brown et al. 1993 developed their pipeline of translation models, the IBM Models 1-5.

## 5.2 Overview of Methods

I shall now give a quick review of word alignment methods, that is of the IBM Models, of `fastalign`, which is an improved version of the IBM Model 2 and serves as a baseline model in my experiment, and of `SimAlign`, the model based on the novel method of word embeddings. Since I am not a mathematician, I will not go into the mathematics of these models. I will rather attempt to explain their principle of operation in a more intuitive way to allow the reader some basic understanding of the mechanics behind the scenes.

### 5.2.1 IBM Model 1

The IBM models are translation models. They were developed in order to compute the conditional probability of a sentence in the target language $f$ given a sentence in the source langauge $e$: $P(f|e)$ (Brown et al. 1993). In layman terms, they compute how likely a given sentence in the target language is a translation of a sentence in the source language. By modeling these probabilities, the models can generate a number of different translations for a sentence. However, there are infinitely many sentences in a language and most sentences occur even in large corpora only once. This makes modeling the probability distribution for full sentences hard and not promising. Instead, the problem is broken up into smaller steps: the model models the probability disributions for individual words—it computes how likely a word in one sentence is a translation of a word in that sentence's translation. The IBM Model 1 is therefore based solely on modeling the probability distributions of lexical translations, i.e., of individual words (Koehn 2009, p. 88).

#### Incomplete Data

There is, however, a problem. We can compute the probability distributions of lexical translations given their counts. That is, by counting how often a word $w_e$ in language $e$ was translated as a word $w_f$ in language $f$, we can compute the desired probability distributions. For example, by counting how many times the German word *das* was translated as *the*, how many times it was translated as *that*, etc., we can compute each word's translation probability distribution. With these individual probability distributions we can compute the likelihood of a sentence in language $f$ given a sentence in language $e$.

Unfortunately, while sentence alignment is a realtively easy task (at least for well-structured texts), and while sentence aligned parallel corpora are not hard to compile or come by, we do not know which words correspond to which words in the sentence pairs. This problem, dubbed as a *chicken and egg problem*, is basically the following: If we had word alignments, it wouldn't be a problem to estimate the lexical translation model and compute the probability distributions for words and sentences. And if we had a model, we could easily estimate the most likely correspondences between words in the source and the target languages. Unfortunately, we have none of the above (Koehn 2009, p. 88).

#### EM Algorithm

In order to solve the problem of incomplete data, an iterative learning algorithm, the expectation-maximization algorithm (EM algorithm) comes into play. The EM algorithm is mathematically intricate. I shall try to explain in simple words the idea behind it.

In the very first iteration, the values of the model parameters are unkown and are initialized with a uniform distribution. This means all words are equally likely to be translations of each other. Then, in the estimation step, the model is applied to the data to compute the most likely alignments. In the maximization step, the model is learned from the data based on counts collected from it. The algorithm counts co-occurences of words in the source and the target languages, which are then weighted with the probabilities that were computed in the estimation step. These weighted counts are used to compute again the probabilities in the estimation step. These two steps, estimation and maximization, are then repeated until convergence—until a global minimum is reached and the probabilities computed stop changing.

In simple words, the model does not know in the beginning which words in the source language correspond to which words in the target language. In the very first iteration, all alignments are equally likely—any word in a sentence in the target language is equally likely a translation of any word in the source language. In order to find the most probable correspondences (or alignments), the model counts how often words are aligned with each other, that is, how often they co-occur in parallel sentences (maximization step). These counts are weighted with the probabilities computed in the previous estimation step to refine the values in the next estimation step. Likely links between words are strengthened, while less likely links are weekened. This goes on until the model converges and the most likely word alignments have been learned by the model (Koehn 2009, pp. 88–92; Brown et al. 1993).

### 5.2.2 Higher IBM Models

Without going too much into details, I will shortly mention the other IBM models, Models 2-5.

Model 1 makes the unrealistic assumption that all connections for each position are equally likely. This means, word order is not modeled by Model 1. In other words, the word order does not influence the likelihood of word alignments. Therefore, for Model 2 does depend on word order. It adds an explicit model for alignment based on the positions of the source and the target words (Brown et al. 1993; Koehn 2009, p. 99).

Model 3 adds a probability distribution of the number of words a source word is usually translated to (dubbed *fertility*). It is able to model alignments of types other than 1-to-1 (Koehn 2009, p. 100).

Models 4 and 5 add more complexity and take into account for instance the positions of any other target words that are connected with the same source word (Brown et al. 1993), since words that are next to each other in the source sentence tend to be next to each other in the target sentence (large phrases tend to move together as units) (Koehn 2009, p. 107).

Models 1-4 serve as stepping stones towards the training of Model 5. Model 1 has a simple mathematical form and a one unique local minimum, which means the parameters learned by it do not depend on the starting point[1]. The estimates learned by Model 1 are used to intialize the training of Model 2, those of Model 2 are used to initialize Model 3, and so on and so forth—each model is initiailized from the parameters of the model before it. This way, the estimates arrived at by the end of training of Model 5 do not depend on the initial estimates of the parameters for Model 1 (Brown et al. 1993).

## 5.3 SimAlign

### 5.3.1 Word Embeddings

Word embeddings are simply put vector representations of words. Embeddings are learned as a sort of byproduct of a neural language model. A language model is a model that assigns probabilities to sentences of a given language—it can say how probable a given sentence is in a language. A language model can also be used to generate sentences by concatenating words based on their probability distributions.

Without going too much into details, when a simple neural language models learns the probabilty distributions for words given all the previous words, the weights in the hidden layer are adapted. We can then use the language model not for generating language, but extract the weights for a specific word from the inner layer, which are an $n$-dimensional vector. We call this vector word embedding.

It has been shown that words that occur in similar contexts have similar word embeddings. This vector similarity can be measured with the cosine similarity.

### 5.3.2 SimAlign

## 5.4 Computing the Word Alignments

[1]The other models have several minima; this means according to the starting parameters, different minima can be arrived at.

# Chapter 7

# Evaluation

## Chapter 8

# Contributions

# Chapter 9

# Summary

# Bibliography

Brown, Peter F. et al. (1993). "The Mathematics of Statistical Machine Translation: Parameter Estimation". In: *Computational Linguistics* 19.2, pp. 263–311. URL: https://aclanthology.org/J93-2003.

Jalili Sabet, Masoud et al. (Nov. 2020). "SimAlign: High Quality Word Alignments Without Parallel Training Data Using Static and Contextualized Embeddings". In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Online: Association for Computational Linguistics, pp. 1627–1643. DOI: 10.18653/v1/2020.findings-emnlp.147. URL: https://aclanthology.org/2020.findings-emnlp.147.

Koehn, Philipp (2009). *Statistical Machine Translation*. Cambridge University Press.