Contents

Chapter 7

Results

After having created a gold standard (see Chapter 6) for evaluating the quality of the alignments, I compared the alignments computed by SimAlign with the alignments computed by a baseline system. I shall now proceed to present the results of the experiment.

7.1 Evaluation Metrics

To evaluate the quality of word alignment, four measures are used. The first three—percision, recall and F-mesaure—are traditional measures in information retrieval (Mihalcea and Pedersen 2003).

Precision is the percent of true positives out of the items marked by the system as positive. It answers the question, how many of the items marked as positive are true positives, and is normally defined as Precision = $\frac{\text{TP}}{\text{TP+FP}}$, where TP referes to true positives and FP to false positives.

Recall is the percent of true positives out of all positives retrieved. It answers the question, how many of all the true positives were found by the system. It is normally defined as Recall = $\frac{\text{TP}}{\text{TP+FN}}$, where TP refers to true positives and FN to false negatives.

F-measure is a score averaging precision and recall. The fourth measurement, Average Error Rate (AER), was introduced by Och and Ney 2000.

For the task of evaluating the word alignment, I used a script made available on GitHub[1] by the creators of SimAlign (Jalili Sabet et al. 2020).

The script uses a definition of precision, recall and AER which stems from Och and Ney 2000 and was later used by many others (Mihalcea and Pedersen 2003; Och and Ney 2003; Östling and Tiedemann 2016; Jalili Sabet et al. 2020).

Precision, recall, F-measure and AER are defined as follows:

$$\text{Recall} = \frac{|A \cap S|}{|S|}, \quad \text{Precision} = \frac{|A \cap P|}{|A|}, F_1 = 2\frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$\text{AER} = 1 - \frac{|A \cap S| + |A \cap P|}{|A| + |S|}$$

With A being the set of alignments generated by the model, S being the set of Sure alignments and P the set of Possible alignments.

[1] https://github.com/cisnlp/simalign/blob/master/scripts/calc_align_score.py

	Method	Dataset Size	Percision	Recall	F_1	AER
Baseline	fast_align	79,548	**0.622**	**0.782**	**0.693**	**0.307**
		25k	0.603	0.754	0.67	0.33
		600	0.515	0.644	0.572	0.427
	eflomal	79,548	0.827	0.877	0.851	0.148
		600	0.707	0.724	0.715	0.284

Table 7.1: Evaluation metrics for word alignments with the baseline model (fast_align) for different dataset sizes. "Dataset Size" refers to the number of sentence pairs.

For a short discussion on the problems of evaluation, see Section 7.4.1.

7.2 Baseline Systems

I chose two baseline systems: `fast_align` (Dyer, Chahuneau, and Smith 2013) and `eflomal` (Östling and Tiedemann 2016).

7.2.1 fast_align

`fast_align` is a re-parameterization of the IBM Model 2. It has become a popular seccessor to Giza++, serves as a baseline system in other works (Östling and Tiedemann 2016; Jalili Sabet et al. 2020), and is even recommended by WHO? as an alternative for Giza++ for computing the word alignments for Moses SMT. It outperforms Giza++ in many scenarios.

`fast_align` is extremely fast—computing the word alignments for the around 80,000 sentence pairs took around 50 seconds. It is well documented and is extremely easy to compile and to operate. All of this makes `fast_align` the most attractive system to use as a baseline system.

7.2.2 eflomal

`eflomal` is ...

7.2.3 Performance

To test the baseline systems (`fast_align`) I word-aligned all of the sentence pairs (79,548), then extracted the alignments for the 600 annotated sentences and again compared my alignments with those produced by `fast_align`. The results are shown in Table 7.1.

7.3 SimAlign

I tested SimAlign with different parameters to word align the 600 setence pairs of German-Romansh, for which I created a gold standard (see Chapter 6).

I tested the two multilingual embeddings that SimAlign works with out-of-the-box: mBERT[2] and XLM-R(Conneau et al. 2020). mBERT only works on a subword level (BPE), while XLM-R

[2]`https://github.com/google-research/bert/blob/master/multilingual.md`

	Embedding	Level	Method	Percision	Recall	F_1	AER
SimAlign	mBert	BPE	Argmax	**0.894**	0.622	0.734	0.266
			Itermax	0.832	0.731	0.778	0.222
			Match	0.795	**0.767**	**0.781**	**0.219**
	XLM-R	Word	Argmax	**0.848**	0.399	0.543	0.457
			Itermax	0.767	0.504	0.608	0.391
			Match	0.67	0.647	**0.658**	**0.342**
		BPE	Argmax	0.773	0.488	0.598	0.402
			Itermax	0.671	0.595	0.631	0.369
			Match	0.558	**0.719**	0.628	0.372

Table 7.2: Evaluation metrics for word alignments using SimAlign, with different embeddings and word/sub-word level. Best result per embedding type in bold.

works either on the word or the subword level.

For each embedding and word/subword-level combination, alignments are produced according to each of the three methods presented by Jalili Sabet et al. 2020 (see also Section 5.4.1).

7.3.1 Results

Table 7.2 shows the evaluation metrics for word alignments computed with SimAlign with the different methods.

To evaluate the alignments against the gold standard, I used a script provided by the creators of SimAlign[3].

For each embedding layer (mBERT and XLM-R), the best score for each column is marked in bold. Generally, the mBERT embeddings perform better. Argmax has the best percision (0.894), which means only 10.6% of the alignments are wrong. However, it has recall measure of only 0.622, which means 37.8% of the alignments are missing. Match has the lowest percision (0.795) but the highest recall (0.767), which makes it the best compromise between percision and recall and it thus has the lowest AER.

7.4 Discussion

Comparing the best performance of SimAlign against the best performance of the baseline systems, SimAlign outperforms `fast_align`, but is outperformed by `eflomal`.

Nonetheless, I believe the results are still surprising and promising. SimAlign uses embeddings from language models which have never seen Romansh, a scenario which is also referred to as zero-shot. Despite this fact, the performance is excellent. SimAlign's recall is on par with fast_align and its precision is 27% higher than that of fast_align.

Also, in the hypothetical case that we only had the 600 annotated sentences to compute word alignment, SimAlign would have outperformed eflomal as well with an AER of 0.284 (SimAlign) against an AER of 0.219 (eflomal).

[3] `https://github.com/cisnlp/simalign/blob/master/scripts/calc_align_score.py`

Method	Precision	Recall	F_1	AER
fast_align	0.622	0.782	0.693	0.307
eflomal	**0.827**	**0.877**	**0.851**	**0.148**
SimAlign: mBERT-BPE	0.795	0.767	0.781	0.219

Table 7.3: Comparison of the best performance of each of the three methods. The best value in each column is in bold.

Further, the SimAlign's performance on the language pair German-Romansh (AER 0.219) doesn't fall from the performance of SimAlign on English-German (AER 0.21 (Table 2 in Jalili Sabet et al. 2020)), which means performance in a zero-shot setting with mBERT embeddings for German-Romansh is just as good.

7.4.1 General Problems with Evaluation

It should also be mentioned that each word alignment gold standard has different annotation guidelines and might be more preferable or biased towards one model or the other. For instance a gold standard which prefers 1-to-1 alignments will reward a model which generates little or no 1-to-many alignments. At the same time, it will penlaize the precision performance of a model that generates 1-to-many alignments, although they might be correct.

Handling Sure and Possible alignments in a different way in each gold standard, might also affect the performance evaluation. Not using Possible alignments will lead to a lower value for $|A \cap P|$, which will negatively affect precision and will penalize a model that performs better than expected. Labeling many of the alignments as Possible alignments instead of Sure will keep S small and thus lead to favorable recall.

Problems with the Gold Standard for German-Romansh

As already explained in Section ??, the gold standard I created is not perfect (no second annotator, no Possible alignments). In my annotation guidelines, I preferred 1-to-1 alignments (see Section ??) and used no Possible label for labeling alignments that might still be correct. Theoratically, not using Possible alignments may explain fast_align's low percision. In theory, it is possible that fast_align generates *correct* 1-to-many alignments which I ignored in my annotations. In that case, we should solely concentrate on recall, which is not affected by Possible alignments. If we were indeed to ignore the other measurements, fast_align would beat SimAlign by 2%, which are insignifcant.

All that being said, I believe the excellent performance of eflomal proves that the gold standard is of good quality and is sensible for measuring the performance of word alignment models.

7.5 Summary

I evaluated the performance of the two baseline models (fast_align and eflomal) and SimAlign, a similarity based word alignment model, using a gold standard of 600 annotated sentence pairs in German-Romansh, which I created myself. I compared the performance of two baseline statistical models with the performance of SimAlign using multilingual embeddings in a zero-shot setting. SimAlign outperformed fast_align, but not eflomal (see Table 7.3).

SimAlign's performance, altough worse the eflomal's performance, is promising. It shows that mBERT's embeddings may be used in a zero-shot setting for the task of word alignment and may give others the impulse to testing the performance of mBERT (or other multilingual models) on Romansh in other tasks, such as information extraction, question answering, sentiment analysis etc.

Glossary

Graubünden The Canton of Grisons. 7

recall Percent of missing positives. Calculated as true positives divided by all positives (true positives plus false negatives) $\frac{\text{TP}}{\text{TP+FP}}$. 45

Acronyms

AER Average Error Rate. 35, 43, 45, 46

List of Tables

List of Figures

Bibliography

Conneau, Alexis et al. (July 2020). “Unsupervised Cross-lingual Representation Learning at Scale”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, pp. 8440–8451. DOI: 10.18653/v1/2020.acl-main.747. URL: https://aclanthology.org/2020.acl-main.747.

Dyer, Chris, Victor Chahuneau, and Noah A. Smith (June 2013). “A Simple, Fast, and Effective Reparameterization of IBM Model 2”. In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Atlanta, Georgia: Association for Computational Linguistics, pp. 644–648. URL: https://aclanthology.org/N13-1073.

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.

Mihalcea, Rada and Ted Pedersen (2003). “An Evaluation Exercise for Word Alignment”. In: *Proceedings of the HLT-NAACL 2003 Workshop on Building and Using Parallel Texts: Data Driven Machine Translation and Beyond*, pp. 1–10. URL: https://aclanthology.org/W03-0301.

Och, Franz Josef and Hermann Ney (Oct. 2000). “Improved Statistical Alignment Models”. In: *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*. Hong Kong: Association for Computational Linguistics, pp. 440–447. DOI: 10.3115/1075218.1075274. URL: https://aclanthology.org/P00-1056.

— (2003). “A Systematic Comparison of Various Statistical Alignment Models”. In: *Computational Linguistics* 29.1, pp. 19–51. DOI: 10.1162/089120103321337421. URL: https://aclanthology.org/J03-1002.

Östling, Robert and Jörg Tiedemann (Oct. 2016). “Efficient word alignment with Markov Chain Monte Carlo”. In: *Prague Bulletin of Mathematical Linguistics* 106, pp. 125–146. URL: http://ufal.mff.cuni.cz/pbml/106/art-ostling-tiedemann.pdf.