SimAlign: High Quality Word Alignments Without Parallel Training Data Using Static and Contextualized Embeddings

Masoud Jalili Sabet[*1], **Philipp Dufter**[*1], **François Yvon**[2], **Hinrich Schütze**[1]
[1] Center for Information and Language Processing (CIS), LMU Munich, Germany
[2] Université Paris-Saclay, CNRS, LIMSI, France
{masoud,philipp}@cis.lmu.de, francois.yvon@limsi.fr

Abstract

Word alignments are useful for tasks like statistical and neural machine translation (NMT) and cross-lingual annotation projection. Statistical word aligners perform well, as do methods that extract alignments jointly with translations in NMT. However, most approaches require parallel training data, and quality decreases as less training data is available. We propose word alignment methods that require no parallel data. The key idea is to leverage multilingual word embeddings – both static and contextualized – for word alignment. Our multilingual embeddings are created from monolingual data only without relying on any parallel data or dictionaries. We find that alignments created from embeddings are superior for four and comparable for two language pairs compared to those produced by traditional statistical aligners – even with abundant parallel data; e.g., contextualized embeddings achieve a word alignment F_1 for English-German that is 5 percentage points higher than eflomal, a high-quality statistical aligner, trained on 100k parallel sentences.

1 Introduction

Word alignments are essential for statistical machine translation and useful in NMT, e.g., for imposing priors on attention matrices (Liu et al., 2016; Chen et al., 2016; Alkhouli and Ney, 2017; Alkhouli et al., 2018) or for decoding (Alkhouli et al., 2016; Press and Smith, 2018). Further, word alignments have been successfully used in a range of tasks such as typological analysis (Lewis and Xia, 2008; Östling, 2015b), annotation projection (Yarowsky et al., 2001; Padó and Lapata, 2009; Asgari and Schütze, 2017; Huck et al., 2019) and creating multilingual embeddings (Guo et al., 2016; Ammar et al., 2016; Dufter et al., 2018).

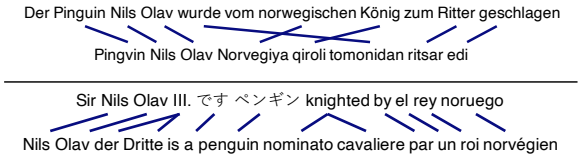

Figure 1: Our method does not rely on parallel training data and can align distant language pairs (German-Uzbek, top) and even mixed sentences (bottom). Example sentence is manually created. Algorithm: Itermax.

Statistical word aligners such as the IBM models (Brown et al., 1993) and their implementations Giza++ (Och and Ney, 2003), fast-align (Dyer et al., 2013), as well as newer models such as eflomal (Östling and Tiedemann, 2016) are widely used for alignment. With the rise of NMT (Bahdanau et al., 2014), attempts have been made to interpret attention matrices as soft word alignments (Cohn et al., 2016; Koehn and Knowles, 2017; Ghader and Monz, 2017). Several methods create alignments from attention matrices (Peter et al., 2017; Zenkel et al., 2019) or pursue a multitask approach for alignment and translation (Garg et al., 2019). However, most systems require parallel data (in sufficient amount to train high quality NMT systems) and their performance deteriorates when parallel text is scarce (Tables 1–2 in (Och and Ney, 2003)).

Recent unsupervised multilingual embedding algorithms that use only non-parallel data provide high quality static (Artetxe et al., 2018; Conneau et al., 2018) and contextualized embeddings (Devlin et al., 2019; Conneau et al., 2020). *Our key idea is to leverage these embeddings for word alignments – by extracting alignments from similarity matrices induced from embeddings – without relying on parallel data.* Requiring no or little parallel data is advantageous, e.g., in the low-resource case and in domain-specific settings without parallel data. A lack of parallel data cannot be easily

[*] Equal contribution - random order.

Findings of the Association for Computational Linguistics: EMNLP 2020, pages 1627–1643
November 16 - 20, 2020.

remedied: mining parallel sentences is possible (Schwenk et al., 2019) but assumes that comparable, monolingual corpora contain parallel sentences. Further, we find that large amounts of mined parallel data do not necessarily improve alignment quality.

Our main **contribution** is that we show that *word alignments obtained from multilingual pretrained language models are superior for four and comparable for two language pairs, compared to strong statistical word aligners like eflomal even in high resource scenarios.* Additionally, **(1)** we introduce three new alignment methods based on the matrix of embedding similarities and two extensions that handle null words and integrate positional information. They permit a flexible tradeoff of recall and precision. **(2)** We provide evidence that subword processing is beneficial for aligning rare words. **(3)** We bundle the source code of our methods in a tool called *SimAlign*, which is available.[1] An interactive online demo is available.[2]

2 Methods

2.1 Alignments from Similarity Matrices

We propose three methods to obtain alignments from similarity matrices. Argmax is a simple baseline, IterMax a novel iterative algorithm, and Match a graph-theoretical method based on identifying matchings in a bipartite graph.

Consider parallel sentences $s^{(e)}, s^{(f)}$, with lengths l_e, l_f in languages e, f. Assume we have access to some embedding function $\mathcal{E}$ that maps each word in a sentence to a d-dimensional vector, i.e., $\mathcal{E}(s^{(k)}) \in \mathbb{R}^{l_k \times d}$ for $k \in \{e, f\}$. Let $\mathcal{E}(s^{(k)})_i$ denote the vector of the i-th word in sentence $s^{(k)}$. For static embeddings $\mathcal{E}(s^{(k)})_i$ depends only on the word i in language k whereas for contextualized embeddings the vector depends on the full context $s^{(k)}$. We define the *similarity matrix* as the matrix $S \in [0,1]^{l_e \times l_f}$ induced by the embeddings where $S_{ij} := \text{sim}\left(\mathcal{E}(s^{(e)})_i, \mathcal{E}(s^{(f)})_j\right)$ is some normalized measure of similarity, e.g., cosine-similarity normalized to be between 0 and 1. We now describe our methods for extracting alignments from S, i.e., obtaining a binary matrix $A \in \{0,1\}^{l_e \times l_f}$.

Argmax. A simple baseline is to align i and j when $s_i^{(e)}$ is the most similar word to $s_j^{(f)}$ and vice-versa. That is, we set $A_{ij} = 1$ if

$$(i = \arg\max_l S_{l,j}) \wedge (j = \arg\max_l S_{i,l})$$

and $A_{ij} = 0$ otherwise. In case of ties, which are unlikely in similarity matrices, we choose the smaller index. If all entries in a row i or column j of S are 0 we set $A_{ij} = 0$ (this case can appear in Itermax). Similar methods have been applied to co-occurrences (Melamed, 2000) ("competitive linking"), Dice coefficients (Och and Ney, 2003) and attention matrices (Garg et al., 2019).

Algorithm 1 Itermax.

procedure ITERMAX(S, $n_{\max}$, $\alpha \in [0,1]$)
  $A, M = \text{zeros\_like}(S)$
  for $n \in [1, \ldots, n_{\max}]$ **do**
    $\forall i, j:$
    $$M_{ij} = \begin{cases} 1 \text{ if } \max\left(\sum_{l=0}^{l_e} A_{lj}, \sum_{l=0}^{l_f} A_{il}\right) = 0 \\ 0 \text{ if } \min\left(\sum_{l=0}^{l_e} A_{lj}, \sum_{l=0}^{l_f} A_{il}\right) > 0 \\ \alpha \text{ otherwise} \end{cases}$$
    $A_{\text{to\_add}} = \text{get\_argmax\_alignments}(S \odot M)$
    $A = A + A_{\text{to\_add}}$
  end for
  return A
end procedure

Figure 2: Description of the Itermax algorithm. *zeros_like* yields a matrix with zeros and with same shape as the input, *get_argmax_alignments* returns alignments obtained using the Argmax Method, $\odot$ is elementwise multiplication.

Itermax. There are many sentences for which Argmax only identifies few alignment edges because mutual argmaxes can be rare. As a remedy, we apply Argmax iteratively. Specifically, we modify the similarity matrix conditioned on the alignment edges found in a previous iteration: if two words i and j have *both* been aligned, we zero out the similarity. Similarly, if *neither* is aligned we leave the similarity unchanged. In case only one of them is aligned, we multiply the similarity with a discount factor $\alpha \in [0,1]$. Intuitively, this encourages the model to focus on unaligned word pairs. However, if the similarity with an already aligned word is exceptionally high, the model can add an additional edge. Note that this explicitly allows one token to be aligned to multiple other tokens. For details on the algorithm see Figure 2.

Match. Argmax finds a local, not a global optimum and Itermax is a greedy algorithm. To find global optima, we frame alignment as an assign-

[1] https://github.com/cisnlp/simalign
[2] https://simalign.cis.lmu.de/

ment problem: we search for a maximum-weight maximal matching (e.g., (Kuhn, 1955)) in the bipartite weighted graph which is induced by the similarity matrix. This optimization problem is defined by

$$A^* = \text{argmax}_{A \in \{0,1\}^{l_e \times l_f}} \sum_{i=1}^{l_e} \sum_{j=1}^{l_f} A_{ij} S_{ij}$$

subject to A being a matching (i.e., each node has at most one edge) that is maximal (i.e., no additional edge can be added). There are known algorithms to solve the above problem in polynomial time (e.g., (Galil, 1986)).

Note that alignments generated with the match method are inherently bidirectional. None of our methods require additional symmetrization as post-processing.

2.2 Distortion and Null Extensions

Distortion Correction [Dist]. Distortion, as introduced in IBM Model 2, is essential for alignments based on non-contextualized embeddings since the similarity of two words is solely based on their surface form, independent of position. To penalize high distortions, we multiply the similarity matrix S componentwise with

$$P_{i,j} = 1 - \kappa \left(i/l_e - j/l_f\right)^2,$$

where κ is a hyperparameter to scale the distortion matrix P between $[(1-\kappa), 1]$. We use $\kappa = 0.5$. See supplementary for different values. We can interpret this as imposing a locality-preserving prior: given a choice, a word should be aligned to a word with a similar relative position ($\left(i/l_e - j/l_f\right)^2$ close to 0) rather than a more distant word (large $\left(i/l_e - j/l_f\right)^2$).

Null. Null words model untranslated words and are an important part of alignment models. We propose to model null words as follows: if a word is not particularly similar to any of the words in the target sentence, we do not align it. Specifically, given an alignment matrix A, we remove alignment edges when the normalized entropy of the similarity distribution is above a threshold τ, a hyperparameter. We use normalized entropy (i.e., entropy divided by the log of sentence length) to account for different sentence lengths; i.e., we set $A_{ij} = 0$ if

$$\min\left(-\frac{\sum_{k=1}^{l_f} S^h_{ik} \log S^h_{ik}}{\log l_f}, -\frac{\sum_{k=1}^{l_e} S^v_{kj} \log S^v_{kj}}{\log l_e}\right) > \tau,$$

where $S^h_{ik} := S_{ik} / \sum_{m=1}^{l_f} S_{im}$, and $S^v_{kj} := S_{kj} / \sum_{m=1}^{l_e} S_{mj}$. As the ideal value of τ depends on the actual similarity scores we set τ to a percentile of the entropy values of the similarity distribution across all aligned edges (we use the 95th percentile). Different percentiles are in the supplementary.

3 Experiments

3.1 Embedding Learning

Static. We train monolingual embeddings with fastText (Bojanowski et al., 2017) for each language on its Wikipedia. We then use VecMap (Artetxe et al., 2018) to map the embeddings into a common multilingual space. Note that this algorithm works without any crosslingual supervision (e.g., multilingual dictionaries). We use the same procedure for word and subword levels. We use the label **fastText** to refer to these embeddings as well as the alignments induced by them.

Contextualized. We use the multilingual BERT model (mBERT).[3] It is pretrained on the 104 largest Wikipedia languages. This model only provides embeddings at the subword level. To obtain a word embedding, we simply average the vectors of its subwords. We consider word representations from all 12 layers as well as the concatenation of all layers. Note that the model is not finetuned. We denote this method as mBERT[i] (when using embeddings from the i-th layer, where 0 means using the non-contextualized initial embedding layer) and mBERT[conc] (for concatenation).

In addition, we use XLM-RoBERTa base (Conneau et al., 2020), which is pretrained on 100 languages on cleaned CommonCrawl data (Wenzek et al., 2020). We denote alignments obtained using the embeddings from the i-th layer by XLM-R[i].

3.2 Word and Subword Alignments

We investigate both alignments between subwords such as wordpiece (Schuster and Nakajima, 2012) (which are widely used for contextualized language models) and words. We refer to computing alignment edges between words as *word level* and between subwords as *subword level*. Note that gold standards are all word-level. In order to evaluate alignments obtained at the subword level we convert subword to word alignments using the heuristic "two words are aligned if any of their subwords are

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

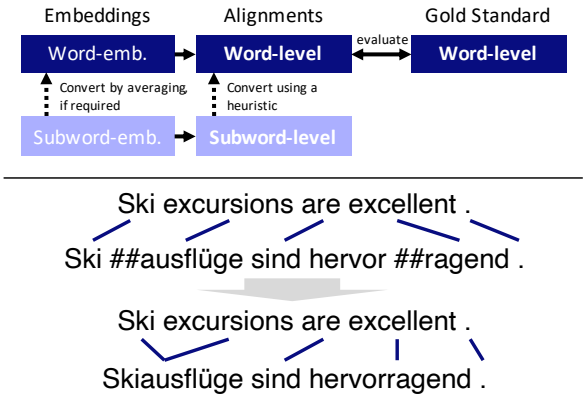

Figure 3: Subword alignments are always converted to word alignments for evaluation.

aligned" (see Figure 3). As a result a single word can be aligned with multiple other words.

For the *word* level, we use the NLTK tokenizer (Bird et al., 2009) (e.g., for tokenizing Wikipedia in order to train fastText). For the *subword* level, we generally use multilingual BERT's vocabulary[3] and BERT's wordpiece tokenizer. For XLM-R we use the XLM-R subword vocabulary. Since gold standards are already tokenized, they do not require additional tokenization.

3.3 Baselines

We compare to three popular statistical alignment models that all require parallel training data. **fast-align/IBM2** (Dyer et al., 2013) is an implementation of an alignment algorithm based on IBM Model 2. It is popular because of its speed and high quality. **eflomal**[4] (based on efmaral by Östling and Tiedemann (2016)), a Bayesian model with Markov Chain Monte Carlo inference, is claimed to outperform fast-align on speed and quality. Further we use the widely used software package **Giza++/IBM4** (Och and Ney, 2003), which implements IBM alignment models. We use its standard settings: 5 iterations each for the HMM model, IBM Models 1, 3 and 4 with $p_0 = 0.98$.

Symmetrization. Probabilistic word alignment models create forward and backward alignments and then symmetrize them (Och and Ney, 2003; Koehn et al., 2005). We compared the symmetrization methods grow-diag-final-and (GDFA) and intersection and found them to perform comparably; see supplementary. We use GDFA throughout the paper.

[4] github.com/robertostling/eflomal

3.4 Evaluation Measures

Given a set of predicted alignment edges A and a set of sure, possible gold standard edges S, P (where $S \subset P$), we use the following evaluation measures:

$$\text{prec} = \frac{|A \cap P|}{|A|}, \text{rec} = \frac{|A \cap S|}{|S|},$$
$$F_1 = \frac{2 \text{ prec rec}}{\text{prec} + \text{rec}},$$
$$\text{AER} = 1 - \frac{|A \cap S| + |A \cap P|}{|A| + |S|},$$

where $|\cdot|$ denotes the cardinality of a set. This is the standard evaluation (Och and Ney, 2003).

3.5 Data

Our **test data** are a diverse set of 6 language pairs: Czech, German, Persian, French, Hindi and Romanian, always paired with English. See Table 11 for corpora and supplementary for URLs.

For our baselines requiring parallel training data (i.e., eflomal, fast-align and Giza++) we select additional parallel **training data** that is consistent with the target domain where available. See Table 11 for the corpora. Unless indicated otherwise we use the whole parallel training data. Figure 5 shows the effect of using more or less training data.

Given the large amount of possible experiments when considering 6 language pairs we do not have space to present all numbers for all languages. If we show results for only one pair, we choose ENG-DEU as it is an established and well-known dataset (EuroParl). If we show results for more languages we fall back to DEU, CES and HIN, to show effects on a mid-resource morphologically rich language (CES) and a low-resource language written in a different script (HIN).

4 Results

4.1 Embedding Layer

Figure 4 shows a parabolic trend across layers of mBERT and XLM-R. We use layer 8 in this paper because it has best performance. This is consistent with other work (Hewitt and Manning, 2019; Tenney et al., 2019): in the first layers the contextualization is too weak for high-quality alignments while the last layers are too specialized on the pretraining task (masked language modeling).

Lang.	Gold Standard	Gold St. Size	$\|S\|$	$\|P \setminus S\|$	Parallel Data	Parallel Data Size	Wikipedia Size
ENG-CES	(Mareček, 2008)	2500	44292	23132	EuroParl (Koehn, 2005)	646k	8M
ENG-DEU	EuroParl-based[a]	508	9612	921	EuroParl (Koehn, 2005)	1920k	48M
ENG-FAS	(Tavakoli and Faili, 2014)	400	11606	0	TEP (Pilevar et al., 2011)	600k	5M
ENG-FRA	WPT2003, (Och and Ney, 2000),	447	4038	13400	Hansards (Germann, 2001)	1130k	32M
ENG-HIN	WPT2005[b]	90	1409	0	Emille (McEnery et al., 2000)	3k	1M
ENG-RON	WPT2005[b]	203	5033	0	Constitution, Newspaper[b]	50k	3M

[a] www-i6.informatik.rwth-aachen.de/goldAlignment/
[b] http://web.eecs.umich.edu/~mihalcea/wpt05/

Table 1: Overview of datasets. "Lang." uses ISO 639-3 language codes. "Size" refers to the number of sentences. "Parallel Data Size" refers to the number of parallel sentences in addition to the gold alignments that is used for training the baselines. Our sentence tokenized version of the English Wikipedia has 105M sentences.

	Method	ENG-CES F_1	ENG-CES AER	ENG-DEU F_1	ENG-DEU AER	ENG-FAS F_1	ENG-FAS AER	ENG-FRA F_1	ENG-FRA AER	ENG-HIN F_1	ENG-HIN AER	ENG-RON F_1	ENG-RON AER
Prior Work	(Östling, 2015a) Bayesian							**.94**	**.06**	.57	.43	**.73**	**.27**
	(Östling, 2015a) Giza++							.92	.07	.51	.49	.72	.28
	(Legrand et al., 2016) Ensemble Method	.81	.16					.71	.10				
	(Östling and Tiedemann, 2016) efmaral							.93	.08	.53	.47	.72	.28
	(Östling and Tiedemann, 2016) fast-align							.86	.15	.33	.67	.68	.33
	(Zenkel et al., 2019) Giza++				.21				**.06**				.28
	(Garg et al., 2019) Multitask				.20				.08				
Baselines Word	fast-align/IBM2	.76	.25	.71	.29	.57	.43	.86	.15	.34	.66	.68	.33
	Giza++/IBM4	.75	.26	.77	.23	.51	.49	.92	.09	.45	.55	.69	.31
	eflomal	.85	.15	.77	.23	.61	.39	.93	.08	.51	.49	.71	.29
Baselines Subword	fast-align/IBM2	.78	.23	.71	.30	.58	.42	.85	.16	.38	.62	.68	.32
	Giza++/IBM4	.82	.18	.78	.22	.57	.43	.92	.09	.48	.52	.69	.32
	eflomal	.84	.17	.76	.24	.63	.37	.91	.09	.52	.48	.72	.28
This Work Word	fastText - Argmax	.70	.30	.60	.40	.50	.50	.77	.22	.49	.52	.47	.53
	mBERT[8] - Argmax	**.87**	**.13**	.79	.21	.67	.33	**.94**	**.06**	.54	.47	.64	.36
	XLM-R[8] - Argmax	**.87**	**.13**	.79	.21	.70	.30	.93	**.06**	.59	.41	.70	.30
This Work Subword	fastText - Argmax	.58	.42	.56	.44	.09	.91	.73	.26	.04	.96	.43	.58
	mBERT[8] - Argmax	.86	.14	**.81**	**.19**	.67	.33	**.94**	**.06**	.55	.45	.65	.35
	XLM-R[8] - Argmax	**.87**	**.13**	**.81**	**.19**	**.71**	**.29**	.93	.07	**.61**	**.39**	.71	.29

Table 2: Comparison of our methods, baselines and prior work in unsupervised word alignment. Best result per column in bold. A detailed version of the table with precision/recall and Itermax/Match results is in supplementary.

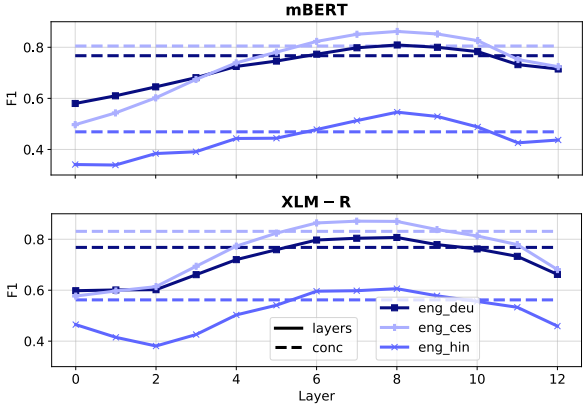

Figure 4: Word alignment performance across layers of mBERT (top) and XLM-R (bottom). Results are F_1 with Argmax at the subword level.

4.2 Comparison with Prior Work

Contextual Embeddings. Table 2 shows that mBERT and XLM-R consistently perform well with the Argmax method. XLM-R yields mostly higher values than mBERT. Our three baselines, eflomal, fast-align and Giza++, are always outperformed (except for RON). We outperform all prior work except for FRA where we match the performance and RON. This comparison is not entirely fair because methods relying on parallel data have access to the parallel sentences of the test data during training whereas our methods do not.

Romanian might be a special case as it exhibits a large amount of many to one links and further lacks determiners. How determiners are handled in the gold standard depends heavily on the annotation guidelines. Note that one of our settings, XLM-R[8] with Itermax at the subword level, has an F1 of .72 for ENG-RON, which comes very close to the performance by (Östling, 2015a) (see Table 3).

In summary, extracting alignments from similarity matrices is a very simple and efficient method that performs surprisingly strongly. It outperforms strong statistical baselines and most prior work in unsupervised word alignment for CES, DEU, FAS and HIN and is comparable for FRA and RON. We attribute this to the strong contextualization in mBERT and XLM-R.

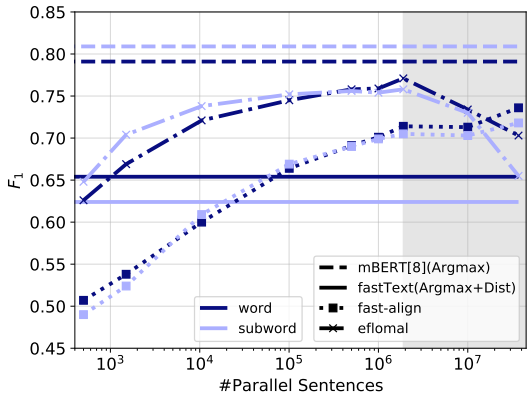

Figure 5: Learning curves of fast-align/eflomal vs. embedding-based alignments. Results shown are F_1 for ENG-DEU, contrasting subword and word representations. Up to 1.9M parallel sentences we use EuroParl. To demonstrate the effect with abundant parallel data we add up to 37M *additional* parallel sentences from ParaCrawl (Esplà et al., 2019) (see grey area).

Static Embeddings. fastText shows a solid performance on word level, which is worse but comes close to fast-align and outperforms it for HIN. We consider this surprising as fastText did not have access to parallel data or any multilingual signal. VecMap can also be used with crosslingual dictionaries. We expect this to boost performance and fastText could then become a viable alternative to fast-align.

Amount of Parallel Data. Figure 5 shows that fast-align and eflomal get better with more training data with eflomal outperforming fast-align, as expected. However, even with 1.9M parallel sentences mBERT outperforms both baselines. When adding up to 37M additional parallel sentences from ParaCrawl (Esplà et al., 2019) performance for fast-align increases slightly, however, eflomal decreases (grey area in plot). ParaCrawl contains mined parallel sentences whose lower quality probably harms eflomal. fastText (with distortion) is competitive with eflomal for fewer than 1000 parallel sentences and outperforms fast-align even with 10k sentences. Thus for very small parallel corpora (<10k sentences) using fastText embeddings is an alternative to fast-align.

The main takeaway from Figure 5 is that mBERT-based alignments, a method that does not need any parallel training data, outperforms state-of-the-art aligners like eflomal for ENG-DEU, even in the very high resource case.

Emb.	Method	ENG-CES	ENG-DEU	ENG-FAS	ENG-FRA	ENG-HIN	ENG-RON
mBERT[8]	Argmax	**.86**	**.81**	.67	**.94**	.55	.65
	Itermax	**.86**	**.81**	**.70**	.93	**.58**	**.69**
	Match	.82	.78	.67	.90	**.58**	.67
XLM-R[8]	Argmax	**.87**	**.81**	.71	**.93**	.61	.71
	Itermax	.86	.80	**.72**	.92	**.62**	**.72**
	Match	.81	.76	.68	.88	.60	.70

Table 3: Comparison of our three proposed methods across all languages for the best embeddings from Table 2: mBERT[8] and XLM-R[8]. We show F_1 at the subword level. Best result per embedding type in bold.

Emb.	$n_{\max}$	α	ENG-DEU Prec.	Rec.	F_1	AER	ENG-CES Prec.	Rec.	F_1	AER	ENG-HIN Prec.	Rec.	F_1	AER
mBERT[8]	1	-	**.92**	.69	.79	.21	**.95**	.80	**.87**	**.13**	**.84**	.39	.54	.47
	2	.90	.85	.77	**.81**	**.19**	.87	.87	**.87**	.14	.75	.47	.58	.42
		.95	.83	.80	**.81**	**.19**	.85	.89	**.87**	**.13**	.73	.48	.58	.42
		1	.77	.79	.78	.22	.80	.86	.83	.17	.63	.46	.53	.47
	3	.90	.81	.80	.80	.20	.83	.88	.85	.15	.70	.49	.57	.43
		.95	.78	**.83**	**.81**	.20	.81	**.91**	.86	.15	.68	**.52**	**.59**	**.41**
		1	.73	**.83**	.77	.23	.76	**.91**	.82	.18	.58	.51	.54	.46
fastText	1	-	**.81**	.48	.60	.40	**.86**	.59	.70	.30	**.75**	.36	.49	.52
	2	.90	.69	.56	**.62**	**.38**	.74	.69	**.72**	**.29**	.63	.42	**.51**	**.49**
		.95	.66	.56	.61	.39	.71	.69	.70	.30	.59	.41	.48	.52
		1	.59	.55	.57	.43	.62	.65	.63	.37	.53	.39	.45	.55
	3	.90	.63	**.59**	.61	.39	.67	.72	.70	.31	.57	.43	.49	.51
		.95	.59	**.59**	.59	.41	.63	**.73**	.68	.33	.53	**.44**	.48	.52
		1	.53	.58	.55	.45	.55	.70	.62	.39	.48	.43	.45	.55

Table 4: Itermax with different number of iterations ($n_{\max}$) and different α. Results are at the word level.

4.3 Additional Methods and Extensions

We already showed that Argmax yields alignments that are competitive with the state of the art. In this section we compare all our proposed methods and extensions more closely.

Itermax. Table 4 shows results for Argmax (i.e., 1 Iteration) as well as Itermax (i.e., 2 or more iterations of Argmax). As expected, with more iterations precision drops in favor of recall. Overall, Itermax achieves higher F_1 scores for the three language pairs (equal for ENG-CES) both for mBERT[8] and fastText embeddings. For Hindi the performance increase is the highest. We hypothesize that for more distant languages Itermax is more beneficial as similarity between wordpieces may be generally lower, thus exhibiting fewer mutual argmaxes. For the rest of the paper if we use Itermax we use 2 Iterations with $\alpha = 0.9$ as it exhibits best performance (5 out of 6 wins in Table 4).

Argmax/Itermax/Match. In Table 3 we compare our three proposed methods in terms of F_1 across all languages. We chose to show the two

Emb.	Method	ENG-DEU Prec.	Rec.	F_1	AER	ENG-CES Prec.	Rec.	F_1	AER	ENG-HIN Prec.	Rec.	F_1	AER
fastText	Argmax	.81	.48	.60	.40	.86	.59	.70	.30	**.75**	**.36**	**.49**	**.52**
	+Dist	**.84**	**.54**	**.65**	**.35**	**.89**	**.68**	**.77**	**.23**	.64	.30	.41	.59
	+Null	.81	.46	.59	.41	.86	.56	.68	.32	.74	.34	.46	.54
	Itermax	.69	.56	.62	.38	.74	.69	.72	.29	**.63**	**.42**	**.51**	**.49**
	+Dist	**.71**	**.62**	**.66**	**.34**	**.75**	**.76**	**.76**	**.25**	.54	.37	.44	.57
	+Null	.69	.53	.60	.40	.74	.66	.70	.30	**.63**	.40	.49	.51
	Match	.60	.58	.59	.41	.65	.71	.68	.32	.55	**.43**	**.48**	**.52**
	+Dist	**.67**	**.64**	**.65**	**.35**	**.72**	**.78**	**.75**	**.25**	.50	.39	.43	.57
	+Null	.61	.56	.58	.42	.66	.69	.67	.33	**.56**	.41	**.48**	**.52**
mBERT[8]	Argmax	.92	**.69**	**.79**	**.21**	**.95**	**.80**	**.87**	**.13**	.84	**.39**	**.54**	**.47**
	+Dist	.91	.67	.77	.23	.93	.79	.85	.15	.68	.29	.41	.59
	+Null	**.93**	.67	.78	.22	**.95**	.77	.85	.15	**.85**	.38	.53	**.47**
	Itermax	.85	**.77**	**.81**	**.19**	.87	**.87**	**.87**	**.14**	.75	**.47**	**.58**	**.43**
	+Dist	.82	.75	.79	.21	.84	.85	.85	.15	.56	.34	.43	.58
	+Null	**.86**	.75	.80	.20	**.88**	.84	.86	**.14**	**.76**	.45	.57	**.43**
	Match	.78	**.74**	**.76**	**.24**	.81	**.85**	**.83**	**.17**	.67	**.52**	**.59**	**.42**
	+Dist	.75	.71	.73	.27	.79	.83	.81	.20	.45	.35	.39	.61
	+Null	**.80**	.73	**.76**	**.24**	**.83**	.83	**.83**	**.17**	**.68**	.51	.58	**.42**

Table 5: Analysis of Null and Distortion Extensions. All alignments are obtained at word-level. Best result per embedding type and method in bold.

best performing settings from Table 2: mBERT[8] and XLM-R[8] at the subword level. Itermax performs slightly better than Argmax with 6 wins, 4 losses and 2 ties. Itermax seems to help more for more distant languages such as FAS, HIN and RON, but harms for FRA. Match has the lowest F_1, but generally exhibits a higher recall (see e.g., Table 5).

Null and Distortion Extensions. Table 5 shows that Argmax and Itermax generally have higher precision, whereas Match has higher recall. Adding Null almost always increases precision, but at the cost of recall, resulting mostly in a lower F_1 score. Adding a distortion prior boosts performance for static embeddings, e.g., from .70 to .77 for ENG-CES Argmax F_1 and similarly for ENG-DEU. For Hindi a distortion prior is harmful. Dist has little and sometimes harmful effects on mBERT indicating that mBERT's contextualized representations already match well across languages.

Summary. Argmax and Itermax exhibit the best and most stable performance. For most language pairs Itermax is recommended. If high recall alignments are required, Match is the recommended algorithm. Except for HIN, a distortion prior is beneficial for static embeddings. Null should be applied when one wants to push precision even higher (e.g., for annotation projection).

4.4 Words and Subwords

Table 2 shows that subword processing slightly outperforms word-level processing for most methods. Only fastText is harmed by subword processing.

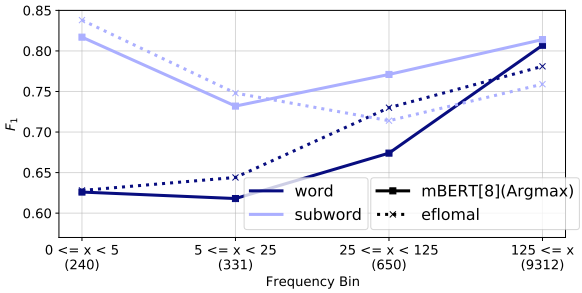

Figure 6: Results for different frequency bins on ENG-DEU. An edge in S, P, or A is attributed to exactly one bin based on the minimum frequency of the involved words (denoted by x). Number of gold edges in brackets. Eflomal is trained on all 1.9M parallel sentences. Frequencies are computed on the same corpus.

		ADJ	ADP	ADV	AUX	NOUN	PRON	VERB
eflomal	Word	**0.83**	0.69	**0.72**	0.63	0.85	0.79	0.63
	Subword	0.82	0.68	0.71	0.57	0.85	0.77	0.62
mBERT[8]	Word	0.79	0.74	0.71	0.71	0.81	**0.84**	**0.69**
	Subword	0.81	**0.75**	**0.72**	**0.72**	**0.87**	**0.84**	**0.69**

Table 6: Alignment performance (F_1) on ENG-DEU for POS. We use mBERT[8](Argmax) and Eflomal trained on 1.9M parallel sentences on the word level.

We use VecMap to match (sub)word distributions across languages. We hypothesize that it is harder to match subword than word distributions – this effect is strongest for Persian and Hindi, probably due to different scripts and thus different subword distributions. Initial experiments showed that adding supervision in form of a dictionary helps restore performance. We will investigate this in future work.

We hypothesize that subword processing is beneficial for aligning rare words. To show this, we compute our evaluation measures for different frequency bins. More specifically, we only consider gold standard alignment edges for the computation where at least one of the member words has a certain frequency in a reference corpus (in our case all 1.9M lines from the ENG-DEU EuroParl corpus). That is, we only consider the edge (i, j) in A, S or P if the minimum of the source and target word frequency is in $[\gamma_l, \gamma_u)$ where γ_l and γ_u are bin boundaries.

Figure 6 shows F_1 for different frequency bins. For rare words both eflomal and mBERT show a severely decreased performance at the word level, but not at the subword level. Thus, subword processing is indeed beneficial for rare words.

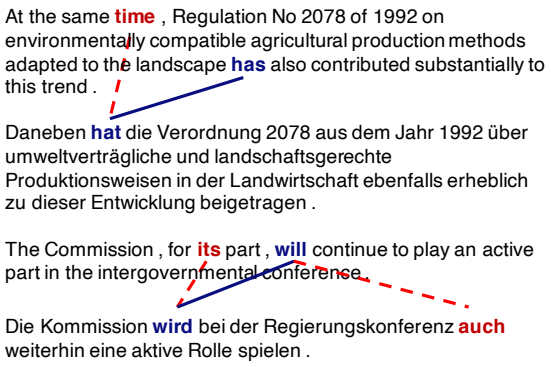

Figure 7: Example alignment of auxiliary verbs. Same setting as in Table 6. Solid lines: mBERT's alignment, identical to the gold standard. Dashed lines: eflomal's incorrect alignment.

4.5 Part-Of-Speech Analysis

To analyze the performance with respect to different part-of-speech (POS) tags, the ENG-DEU gold standard was tagged with the Stanza toolkit (Qi et al., 2020). We evaluate the alignment performance for each POS tag by only considering the alignment edges where at least one of their member words has this tag. Table 6 shows results for frequent POS tags. Compared to eflomal, mBERT aligns auxiliaries, pronouns and verbs better. The relative position of auxiliaries and verbs in German can diverge strongly from that in English because they occur at the end of the sentence (verb-end position) in many clause types. Positions of pronouns can also diverge due to a more flexible word order in German. It is difficult for an HMM-based aligner like eflomal to model such high-distortion alignments, a property that has been found by prior work as well (Ho and Yvon, 2019). In contrast, mBERT(Argmax) does not use distortion information, so high distortion is not a problem for it.

Figure 7 gives an example for auxiliaries. The gold alignment ("has" – "hat") is correctly identified by mBERT (solid line). Eflomal generates an incorrect alignment ("time" – "hat"): the two words have about the same relative position, indicating that distortion minimization is the main reason for this incorrect alignment. Analyzing all auxiliary alignment edges, the average absolute value of the distance between aligned words is 2.72 for eflomal and 3.22 for mBERT. This indicates that eflomal is more reluctant than mBERT to generate high-distortion alignments and thus loses accuracy.

5 Related Work

Brown et al. (1993) introduced the IBM models, the best known statistical word aligners. More recent aligners, often based on IBM models, include fast-align (Dyer et al., 2013), Giza++ (Och and Ney, 2003) and eflomal (Östling and Tiedemann, 2016). (Östling, 2015a) showed that Bayesian Alignment Models perform well. Neural network based extensions of these models have been considered (Ayan et al., 2005; Ho and Yvon, 2019). All of these models are trained on parallel text. Our method instead aligns based on embeddings that are induced from monolingual data only. We compare with prior methods and observe comparable performance.

Prior work on using learned representations for alignment includes (Smadja et al., 1996; Och and Ney, 2003) (Dice coefficient), (Jalili Sabet et al., 2016) (incorporation of embeddings into IBM models), (Legrand et al., 2016) (neural network alignment model) and (Pourdamghani et al., 2018) (embeddings are used to encourage words to align to similar words). Tamura et al. (2014) use recurrent neural networks to learn alignments. They use noise contrastive estimation to avoid supervision. Yang et al. (2013) train a neural network that uses pretrained word embeddings in the initial layer. All of this work requires parallel data. mBERT is used for word alignments in concurrent work: Libovický et al. (2019) use the high quality of mBERT alignments as evidence for the "language-neutrality" of mBERT. Nagata et al. (2020) phrase word alignment as crosslingual span prediction and finetune mBERT using gold alignments.

Attention in NMT (Bahdanau et al., 2014) is related to a notion of soft alignment, but often deviates from conventional word alignments (Ghader and Monz, 2017; Koehn and Knowles, 2017). One difference is that standard attention does not have access to the target word. To address this, Peter et al. (2017) tailor attention matrices to obtain higher quality alignments. Li et al. (2018)'s and Zenkel et al. (2019)'s models perform similarly to and Zenkel et al. (2020) outperform Giza++. Ding et al. (2019) propose better decoding algorithms to deduce word alignments from NMT predictions. Chen et al. (2016), Mi et al. (2016) and Garg et al. (2019) obtain alignments and translations in a multitask setup. Garg et al. (2019) find that operating at the subword level can be beneficial for alignment models. Li et al. (2019) propose two methods to extract alignments from NMT

models, however they do not outperform fast-align. Stengel-Eskin et al. (2019) compute similarity matrices of encoder-decoder representations that are leveraged for word alignments, together with supervised learning, which requires manually annotated alignment. We find our proposed methods to be competitive with these approaches. In contrast to our work, they all require parallel data.

6 Conclusion

We presented word aligners based on contextualized embeddings that outperform in four and match the performance of state-of-the-art aligners in two language pairs; e.g., for ENG-DEU contextualized embeddings achieve an alignment F_1 that is 5 percentage points higher than eflomal trained on 100k parallel sentences. Further, we showed that alignments from static embeddings can be a viable alternative to statistical aligner when few parallel training data is available. In contrast to all prior work our methods do not require parallel data for training at all. With our proposed methods and extensions such as Match, Itermax and Null it is easy to obtain higher precision or recall depending on the use case.

Future work includes modeling fertility explicitly and investigating how to incorporate parallel data into the proposed methods.

Acknowledgments

We gratefully acknowledge funding through a Zentrum Digitalisierung.Bayern fellowship awarded to the second author. This work was supported by the European Research Council (# 740516). We thank Matthias Huck, Jindřich Libovický, Alex Fraser and the anonymous reviewers for interesting discussions and valuable comments. Thanks to Jindřich for pointing out that mBERT can align mixed-language sentences as shown in Figure 1.

References

Tamer Alkhouli, Gabriel Bretschner, and Hermann Ney. 2018. On the alignment problem in multi-head attention-based neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, Belgium, Brussels. Association for Computational Linguistics.

Tamer Alkhouli, Gabriel Bretschner, Jan-Thorsten Peter, Mohammed Hethnawi, Andreas Guta, and Hermann Ney. 2016. Alignment-based neural machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 1, Research Papers*, Berlin, Germany. Association for Computational Linguistics.

Tamer Alkhouli and Hermann Ney. 2017. Biasing attention-based recurrent neural networks using external alignment information. In *Proceedings of the Second Conference on Machine Translation*, Copenhagen, Denmark. Association for Computational Linguistics.

Waleed Ammar, George Mulcaire, Yulia Tsvetkov, Guillaume Lample, Chris Dyer, and Noah A Smith. 2016. Massively multilingual word embeddings. *arXiv preprint arXiv:1602.01925*.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia. Association for Computational Linguistics.

Ehsaneddin Asgari and Hinrich Schütze. 2017. Past, present, future: A computational investigation of the typology of tense in 1000 languages. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark. Association for Computational Linguistics.

Necip Fazil Ayan, Bonnie J. Dorr, and Christof Monz. 2005. NeurAlign: Combining word alignments using neural networks. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Representations*.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media, Inc.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2).

Wenhu Chen, Evgeny Matusov, Shahram Khadivi, and Jan-Thorsten Peter. 2016. Guided alignment training for topic-aware neural machine translation. *AMTA 2016*.

Trevor Cohn, Cong Duy Vu Hoang, Ekaterina Vymolova, Kaisheng Yao, Chris Dyer, and Gholamreza Haffari. 2016. Incorporating structural alignment biases into an attentional neural translation model. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 876–885, San Diego, California. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 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.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *Proceedings of the Sixth International Conference on Learning Representations*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota. Association for Computational Linguistics.

Shuoyang Ding, Hainan Xu, and Philipp Koehn. 2019. Saliency-driven word alignment interpretation for neural machine translation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*, Florence, Italy. Association for Computational Linguistics.

Philipp Dufter, Mengjie Zhao, Martin Schmitt, Alexander Fraser, and Hinrich Schütze. 2018. Embedding learning through multilingual concept induction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia. Association for Computational Linguistics.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 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.

Miquel Esplà, Mikel Forcada, Gema Ramírez-Sánchez, and Hieu Hoang. 2019. ParaCrawl: Web-scale parallel corpora for the languages of the EU. In *Proceedings of Machine Translation Summit XVII Volume 2: Translator, Project and User Tracks*, Dublin, Ireland. European Association for Machine Translation.

Zvi Galil. 1986. Efficient algorithms for finding maximum matching in graphs. *ACM Computing Surveys (CSUR)*, 18(1).

Sarthak Garg, Stephan Peitz, Udhyakumar Nallasamy, and Matthias Paulik. 2019. Jointly learning to align and translate with transformer models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China. Association for Computational Linguistics.

Ulrich Germann. 2001. Aligned Hansards of the 36th parliament of Canada.

Hamidreza Ghader and Christof Monz. 2017. What does attention in neural machine translation pay attention to? In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Jiang Guo, Wanxiang Che, David Yarowsky, Haifeng Wang, and Ting Liu. 2016. A representation learning framework for multi-source transfer parsing. In *Thirtieth AAAI Conference on Artificial Intelligence*.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota. Association for Computational Linguistics.

Anh Khoa Ngo Ho and François Yvon. 2019. Neural baselines for word alignment. In *Proceedings of the 16th International Workshop on Spoken Language Translation*.

Matthias Huck, Diana Dutka, and Alexander Fraser. 2019. Cross-lingual annotation projection is effective for neural part-of-speech tagging. In *Proceedings of the Sixth Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 223–233, Ann Arbor, Michigan. Association for Computational Linguistics.

Masoud Jalili Sabet, Heshaam Faili, and Gholamreza Haffari. 2016. Improving word alignment of rare words with word embeddings. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, Osaka, Japan. The COLING 2016 Organizing Committee.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Machine Translation Summit*, volume 5.

Philipp Koehn, Amittai Axelrod, Alexandra Birch Mayne, Chris Callison-Burch, Miles Osborne, and

David Talbot. 2005. Edinburgh system description for the 2005 IWSLT speech translation evaluation. In *International Workshop on Spoken Language Translation (IWSLT) 2005*.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, Vancouver. Association for Computational Linguistics.

Harold W Kuhn. 1955. The Hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2).

Joël Legrand, Michael Auli, and Ronan Collobert. 2016. Neural network-based word alignment through score aggregation. In *Proceedings of the First Conference on Machine Translation: Volume 1, Research Papers*, Berlin, Germany. Association for Computational Linguistics.

William D. Lewis and Fei Xia. 2008. Automatically identifying computationally relevant typological features. In *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-II*.

Xintong Li, Guanlin Li, Lemao Liu, Max Meng, and Shuming Shi. 2019. On the word alignment from neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy. Association for Computational Linguistics.

Xintong Li, Lemao Liu, Zhaopeng Tu, Shuming Shi, and Max Meng. 2018. Target foresight based attention for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, New Orleans, Louisiana. Association for Computational Linguistics.

Jindřich Libovický, Rudolf Rosa, and Alexander Fraser. 2019. How language-neutral is multilingual BERT? *arXiv preprint arXiv:1911.03310*.

Lemao Liu, Masao Utiyama, Andrew Finch, and Eiichiro Sumita. 2016. Neural machine translation with supervised attention. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, Osaka, Japan. The COLING 2016 Organizing Committee.

David Mareček. 2008. Automatic alignment of tectogrammatical trees from Czech-English parallel corpus. Master's thesis, Charles University, MFF UK.

Anthony McEnery, Paul Baker, Rob Gaizauskas, and Hamish Cunningham. 2000. Emille: Building a corpus of South Asian languages. *VIVEK-BOMBAY-*, 13(3).

I. Dan Melamed. 2000. Models of translation equivalence among words. *Computational Linguistics*, 26(2).

Haitao Mi, Zhiguo Wang, and Abe Ittycheriah. 2016. Supervised attentions for neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas. Association for Computational Linguistics.

Rada Mihalcea 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*.

Masaaki Nagata, Chousa Katsuki, and Masaaki Nishino. 2020. A supervised word alignment method based on cross-language span prediction using multilingual BERT. *arXiv preprint arXiv:2004.14516*.

Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, Hong Kong. Association for Computational Linguistics.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1).

Robert Östling. 2015a. *Bayesian models for multilingual word alignment*. Ph.D. thesis, Department of Linguistics, Stockholm University.

Robert Östling. 2015b. Word order typology through multilingual word alignment. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, Beijing, China. Association for Computational Linguistics.

Robert Östling and Jörg Tiedemann. 2016. Efficient word alignment with Markov Chain Monte Carlo. *The Prague Bulletin of Mathematical Linguistics*, 106(1).

Sebastian Padó and Mirella Lapata. 2009. Cross-lingual annotation projection for semantic roles. *Journal of Artificial Intelligence Research*, 36.

Jan-Thorsten Peter, Arne Nix, and Hermann Ney. 2017. Generating alignments using target foresight in attention-based neural machine translation. *The Prague Bulletin of Mathematical Linguistics*, 108(1).

Mohammad Taher Pilevar, Heshaam Faili, and Abdol Hamid Pilevar. 2011. TEP: Tehran English-Persian parallel corpus. In *International Conference on Intelligent Text Processing and Computational Linguistics*. Springer.

Nima Pourdamghani, Marjan Ghazvininejad, and Kevin Knight. 2018. Using word vectors to improve word alignments for low resource machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, New Orleans, Louisiana. Association for Computational Linguistics.

Ofir Press and Noah A Smith. 2018. You may not need attention. *arXiv preprint arXiv:1810.13409*.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, Online. Association for Computational Linguistics.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE.

Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong, and Francisco Guzmán. 2019. Wikimatrix: Mining 135m parallel sentences in 1620 language pairs from wikipedia. *arXiv preprint arXiv:1907.05791*.

Frank Smadja, Kathleen R. McKeown, and Vasileios Hatzivassiloglou. 1996. Translating collocations for bilingual lexicons: A statistical approach. *Computational Linguistics*, 22(1).

Elias Stengel-Eskin, Tzu-ray Su, Matt Post, and Benjamin Van Durme. 2019. A discriminative neural model for cross-lingual word alignment. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China. Association for Computational Linguistics.

Akihiro Tamura, Taro Watanabe, and Eiichiro Sumita. 2014. Recurrent neural networks for word alignment model. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Baltimore, Maryland. Association for Computational Linguistics.

Leila Tavakoli and Heshaam Faili. 2014. Phrase alignments in parallel corpus using bootstrapping approach. *International Journal of Information & Communication Technology Research*, 6(3).

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy. Association for Computational Linguistics.

Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave. 2020. CCNet: Extracting high quality monolingual datasets from web crawl data. In *Proceedings of The 12th Language Resources and Evaluation Conference*, Marseille, France. European Language Resources Association.

Nan Yang, Shujie Liu, Mu Li, Ming Zhou, and Nenghai Yu. 2013. Word alignment modeling with context dependent deep neural network. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Sofia, Bulgaria. Association for Computational Linguistics.

David Yarowsky, Grace Ngai, and Richard Wicentowski. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *Proceedings of the First International Conference on Human Language Technology Research*.

Thomas Zenkel, Joern Wuebker, and John DeNero. 2019. Adding interpretable attention to neural translation models improves word alignment. *arXiv preprint arXiv:1901.11359*.

Thomas Zenkel, Joern Wuebker, and John DeNero. 2020. End-to-end neural word alignment outperforms GIZA++. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1605–1617, Online. Association for Computational Linguistics.

A Additional Non-central Results

A.1 Comparison with Prior Work

A more detailed version of Table 2 from the main paper that includes precision and recall and results on Itermax can be found in Table 7.

A.2 Rare Words

Figure 8 shows the same as Figure 6 from the main paper but now with a reference corpus of 100k/1000k instead of 1920k parallel sentences. The main takeaways are similar.

A.3 Symmetrization

For asymmetric alignments different symmetrization methods exist. Dyer et al. (2013) provide an overview and implementation (fast-align) for these methods, which we use. We compare intersection and grow-diag-final-and (GDFA) in Table 9. In terms of F1, GDFA performs better (Intersection wins four times, GDFA eleven times, three ties). As expected, Intersection yields higher precision while GDFA yields higher recall. Thus intersection is preferable for tasks like annotation projection,

		Method	ENG-CES Prec.	Rec.	F_1	AER	ENG-DEU Prec.	Rec.	F_1	AER	ENG-FAS Prec.	Rec.	F_1	AER	ENG-FRA Prec.	Rec.	F_1	AER	ENG-HIN Prec.	Rec.	F_1	AER	ENG-RON Prec.	Rec.	F_1	AER
Prior Work		(Östling, 2015a) Bayesian													.96	.92	**.94**	**.06**	.85	.43	.57	.43	.91	.61	**.73**	**.27**
		(Östling, 2015a) Giza++													**.98**	.87	.92	.07	.63	.44	.51	.49	.85	.63	.72	.28
		(Legrand et al., 2016) Ensemble Method	.79	.83	.81	.16									.59	.90	.71	.10								
		(Östling and Tiedemann, 2016) efmaral															.93	.08			.53	.47			.72	.28
		(Östling and Tiedemann, 2016) fast-align															.86	.15			.33	.67			.68	.33
		(Zenkel et al., 2019) Giza++								.21								**.06**								.28
		(Garg et al., 2019) Multitask								.20								.08								
Baselines	Word	fast-align/IBM2	.71	.81	.76	.25	.70	.73	.71	.29	.60	.54	.57	.43	.81	.93	.86	.15	.34	.33	.34	.66	.69	**.67**	.68	.33
		Giza++/IBM4	.71	.79	.75	.26	.79	.75	.77	.23	.55	.48	.51	.49	.90	.95	.92	.09	.47	.43	.45	.55	.74	.64	.69	.31
		eflomal	.84	.86	.85	.15	.80	.75	.77	.23	.68	.55	.61	.39	.91	.94	.93	.08	.61	.44	.51	.49	.81	.63	.71	.29
	Subword	fast-align/IBM2	.72	.84	.78	.23	.67	.74	.71	.30	.60	.56	.58	.42	.80	.92	.85	.16	.39	.37	.38	.62	.69	**.67**	.68	.32
		Giza++/IBM4	.79	.86	.82	.18	.78	.78	.78	.22	.58	.56	.57	.43	.89	.95	.92	.09	.52	.44	.48	.52	.74	.64	.69	.32
		eflomal	.80	.88	.84	.17	.74	.78	.76	.24	.66	.60	.63	.37	.88	.95	.91	.09	.58	.47	.52	.48	.78	**.67**	.72	.28
This Work	Word	fastText - Itermax	.74	.69	.72	.29	.69	.56	.62	.38	.63	.45	.53	.48	.74	.78	.76	.24	.63	.42	.51	.49	.64	.40	.50	.51
		mBERT[8] - Itermax	.87	.87	**.87**	.14	.85	.77	**.81**	**.19**	.80	.63	.70	.30	.91	.95	.93	.08	.75	.47	.58	.43	.82	.58	.68	.32
		XLM-R[8] - Itermax	.89	.85	**.87**	**.13**	.86	.73	.79	.21	.84	.63	**.72**	**.28**	.91	.93	.92	.08	.79	.49	.61	**.39**	.87	.61	.71	.29
		fastText - Argmax	.86	.59	.70	.30	.81	.48	.60	.40	.75	.38	.50	.50	.85	.71	.77	.22	.75	.36	.49	.52	.77	.34	.47	.53
		mBERT[8] - Argmax	.95	.80	**.87**	**.13**	.92	.69	.79	.21	.88	.54	.67	.33	.97	.91	**.94**	**.06**	.84	.39	.54	.47	.90	.50	.64	.36
		XLM-R[8] - Argmax	**.96**	.80	**.87**	**.13**	**.93**	.68	.79	.22	**.91**	.57	.70	.30	.96	.91	.93	**.06**	**.88**	.45	.59	.41	**.94**	.56	.70	.30
	Subword	fastText - Itermax	.61	.57	.59	.41	.63	.54	.58	.42	.20	.07	.11	.90	.70	.76	.73	.28	.14	.05	.07	.93	.56	.38	.45	.55
		mBERT[8] - Itermax	.84	**.89**	.86	.14	.83	**.80**	**.81**	**.19**	.76	.65	.70	.30	.91	**.96**	.93	.08	.71	.49	.58	.42	.79	.62	.69	.31
		XLM-R[8] - Itermax	.84	**.89**	.86	.14	.83	.78	.80	.20	.79	**.67**	**.72**	**.28**	.89	.94	.92	.09	.75	**.52**	**.62**	**.39**	.83	.64	.72	.28
		fastText - Argmax	.72	.48	.58	.42	.75	.45	.56	.44	.27	.06	.09	.91	.80	.67	.73	.26	.14	.02	.04	.96	.67	.31	.43	.58
		mBERT[8] - Argmax	.92	.81	.86	.14	.92	.72	**.81**	**.19**	.85	.56	.67	.33	.96	.92	**.94**	**.06**	.81	.41	.55	.45	.88	.51	.65	.35
		XLM-R[8] - Argmax	.92	.83	**.87**	**.13**	.92	.72	**.81**	**.19**	.87	.59	.71	.30	.95	.91	.93	.07	.86	.47	.61	**.39**	.91	.59	.71	.29

Table 7: Comparison of word and subword levels. Best overall result per column in bold.

Emb.	Method	ENG-DEU Prec.	Rec.	F_1	AER	ENG-CES Prec.	Rec.	F_1	AER	ENG-HIN Prec.	Rec.	F_1	AER
fastText	Argmax	.75	.45	.56	.44	.72	.48	.58	.42	.14	.02	.04	.96
	+Dist	**.79**	**.51**	**.62**	**.38**	**.77**	**.58**	**.66**	**.34**	**.16**	**.04**	**.06**	**.94**
	+Null	.76	.43	.55	.45	.74	.47	.57	.42	.14	.02	.04	.96
	Itermax	.63	.54	.58	.42	.61	.57	.59	.41	.14	.05	.07	.93
	+Dist	**.67**	**.60**	**.64**	**.36**	**.63**	**.66**	**.65**	**.36**	**.15**	**.07**	**.09**	**.91**
	+Null	.64	.52	.57	.43	.62	.56	.59	.41	.14	.04	.07	.93
	Match	.51	.58	.54	.46	.44	.61	.52	.49	**.10**	.08	**.09**	**.91**
	+Dist	**.59**	**.66**	**.62**	**.38**	**.54**	**.71**	**.61**	**.39**	**.10**	**.09**	**.09**	**.91**
	+Null	.52	.57	.54	.46	.46	.60	.52	.48	**.10**	.08	**.09**	**.91**
mBERT[8]	Argmax	.92	**.72**	**.81**	**.19**	**.92**	**.81**	**.86**	**.14**	.81	**.41**	**.55**	**.45**
	+Dist	.90	.70	.79	.21	.91	.80	.85	.15	.65	.30	.41	.59
	+Null	**.93**	.70	.80	.20	**.92**	.78	.85	.15	**.82**	.40	.54	.47
	Itermax	.83	**.80**	**.81**	**.19**	.84	**.89**	**.86**	**.14**	.71	**.49**	**.58**	**.42**
	+Dist	.81	.77	.79	.21	.82	.87	.84	.16	.53	.35	.42	.58
	+Null	**.85**	.77	**.81**	.20	**.84**	.86	.85	.15	**.72**	.47	.57	.43
	Match	.75	**.80**	**.78**	**.23**	.76	**.90**	**.82**	**.18**	.64	**.52**	**.58**	**.43**
	+Dist	.72	.77	.75	.26	.74	.88	.80	.20	.45	.37	.40	.60
	+Null	**.77**	.78	**.78**	**.23**	**.77**	.88	**.82**	.19	**.65**	.51	.57	**.43**

Table 8: Comparison of methods for inducing alignments from similarity matrices. All results are subword-level. Best result per embedding type across columns in bold.

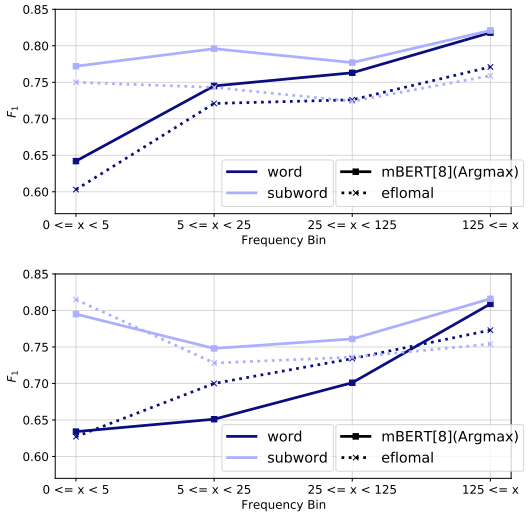

Figure 8: Results for different frequency bins. An edge in S, P, or A is attributed to exactly one bin based on the minimum frequency of the involved words (denoted by x). Top: Eflomal trained and frequencies computed on 100k parallel sentences. Bottom: 1000k parallel sentences.

whereas GDFA is typically used in statistical machine translation.

A.4 Alignment Examples for Different Methods

We show examples in Figure 10, Figure 11, Figure 12, and Figure 13. They provide an overview how the methods actually affect results.

B Hyperparameters

B.1 Overview

We provide a list of customized hyperparameters used in our computations in Table 10. There are three options how we came up with the hyperparameters: a) We simply used default values of 3rd party software. b) We chose an arbitrary value.

Method	Symm.	ENG-CES Prec.	Rec.	F_1	AER	ENG-DEU Prec.	Rec.	F_1	AER	ENG-FAS Prec.	Rec.	F_1	AER	ENG-FRA Prec.	Rec.	F_1	AER	ENG-HIN Prec.	Rec.	F_1	AER	ENG-RON Prec.	Rec.	F_1	AER
eflomal	Inters.	**.95**	.79	**.86**	**.14**	**.91**	.66	.76	.24	**.88**	.43	.58	.42	**.96**	.90	**.93**	**.07**	**.81**	.37	**.51**	**.49**	**.91**	.56	.70	.31
	GDFA	.84	**.86**	.85	.15	.80	**.75**	**.77**	**.23**	.68	**.55**	**.61**	**.39**	.91	**.94**	**.93**	.08	.61	**.44**	**.51**	**.49**	.81	**.63**	**.71**	**.29**
fast-align	Inters.	**.89**	.69	**.78**	**.22**	**.87**	.60	**.71**	**.29**	**.78**	.43	.55	.45	**.93**	.84	**.88**	**.11**	**.55**	.22	.31	.69	**.89**	.50	.64	.36
	GDFA	.71	**.81**	.76	.25	.70	**.73**	**.71**	**.29**	.60	**.54**	**.57**	**.43**	.81	**.93**	.86	.15	.34	**.33**	**.34**	**.66**	.69	**.67**	**.68**	**.33**
GIZA++	Inters.	**.95**	.60	.74	**.26**	**.92**	.62	.74	.26	**.89**	.26	.40	.60	**.97**	.89	**.93**	**.06**	**.82**	.25	.38	.62	**.95**	.47	.63	.37
	GDFA	.71	**.79**	**.75**	**.26**	.79	**.75**	**.77**	**.23**	.55	**.48**	**.51**	**.49**	.90	**.95**	.92	.09	.47	**.43**	**.45**	**.55**	.74	**.64**	**.69**	**.31**

Table 9: Comparison of symmetrization methods at the word level. Best result across rows per method in bold.

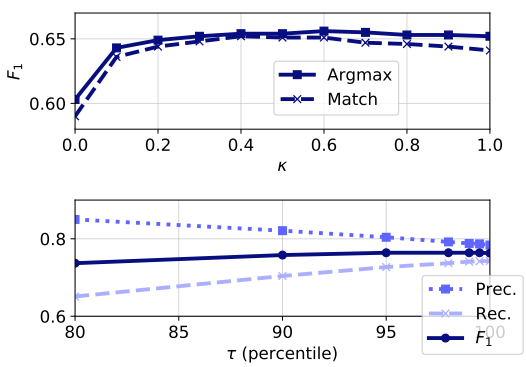

Figure 9: Top: F1 for ENG-DEU with fastText at word-level for different values of κ. Bottom: Performance for ENG-DEU with mBERT[8] (Match) at word-level when setting the value of τ to different percentiles. τ can be used for trading precision against recall. F_1 remains stable although it decreases slightly when assigning τ the value of a smaller percentile (e.g., 80).

Usually we fell back to well-established and rather conventional values (e.g., embedding dimension 300 for static embeddings). c) We defined a reasonable but arbitrary range, out of which we selected the best value using grid search. Table 10 lists the final values we used as well as how we came up with the specific value. For option c) the corresponding analyses are in Figure 4 and Table 3 in the main paper as well as in §B.2 in this supplementary material.

B.2 Null and Distortion Extensions

In Figure 9 we plot the performance for different values of κ. We observe that introducing distortion indeed helps (i.e., $\kappa > 0$) but the actual value is not decisive for performance. This is rather intuitive, as a small adjustment to the similarities is sufficient while larger adjustments do not necessarily change the argmax or the optimal point in the matching algorithm. We choose $\kappa = 0.5$.

For τ in null-word extension, we plot precision, recall and F_1 in Figure 9 when assigning τ different percentile values. Note that values for τ depend on the similarity distribution of all aligned edges. As expected, when using the 100th percentile no edges are removed and thus the performance is not changed compared to not having a null-word extension. When decreasing the value of τ the precision increases and recall goes down, while F_1 remains stable. We use the 95th percentile for τ .

C Reproducibility Information

C.1 Computing Infrastructures, Runtimes, Number of Parameters

We did all computations on up to 48 cores of Intel(R) Xeon(R) CPU E7-8857 v2 with 1TB memory and a single GeForce GTX 1080 GPU with 8GB memory.

Runtimes for aligning 500 parallel sentences on ENG-DEU are reported in Table 12. mBERT and XLM-R computations are done on the GPU. Note that fast-align, GIZA++ and eflomal usually need to be trained on much more parallel data to achieve better performance: this increases their runtime.

All our proposed methods are **parameter-free**. If we consider the parameters of the pretrained language models and pretrained embeddings then fastText has around 1 billion parameters (up to 500k words per language, 7 languages and embedding dimension 300), mBERT has 172 million, XLM-R 270 million parameters.

Method	Runtime[s]
fast-align	4
GIZA++	18
eflomal	5
mBERT[8] - Argmax	15
XLM-R[8] - Argmax	22

Table 12: Runtime (average across 5 runs) in seconds for each method to align 500 parallel sentences.

C.2 Data

Table 11 provides download links to all data used.

System	Parameter	Value
fastText	Version	0.9.1
	Code URL	https://github.com/facebookresearch/fastText/archive/v0.9.1.zip
	Downloaded on	11.11.2019
	Embedding Dimension	300
mBERT,XLM-R	Code: Huggingface Transformer	Version 2.3.1
	Maximum Sequence Length	128
fastalign	Code URL	https://github.com/clab/fast_align
	Git Hash	7c2bbca3d5d61ba4b0f634f098c4fcf63c1373e1
	Flags	-d -o -v
eflomal	Code URL	https://github.com/robertostling/eflomal
	Git Hash	9ef1ace1929c7687a4817ec6f75f47ee684f9aff
	Flags	–model 3
GIZA++	Code URL	http://web.archive.org/web/20100221051856/http://code.google.com/p/giza-pp
	Version	1.0.3
	Iterations	5 iter. HMM, 5 iter. Model 1, 5 iter. Model3, 5 iter. Model 4 (DEFAULT)
	p0	0.98
Vecmap	Code URL	https://github.com/artetxem/vecmap.git
	Git Hash	b82246f6c249633039f67fa6156e51d852bd73a3
	Manual Vocabulary Cutoff	500000
Distortion Ext.	κ	0.5 (chosen ouf of $[0.0, 0.1, \ldots, 1.0]$ by grid search, criterion: F_1)
Null Extension	τ	95th percentile of similarity distribution of aligned edges (chosen out of [80, 90, 95, 98, 99, 99.5] by grid search, criterion: F_1)
Argmax	Layer	8 (for mBERT and XLM-R, chosen out of $[0, 1, \ldots, 12]$ by grid search, criterion: F_1)
Vecmap	α	0.9 (chosen out of [0.9, 0.95, 1] by grid search, criterion: F_1)
	Iterations $n_{\max}$	2 (chosen out of [1,2,3] by grid search, criterion: F_1)

Table 10: Overview on hyperparameters. We only list parameters where we do **not** use default values. Shown are the values which we use unless specifically indicated otherwise.

Lang.	Name	Description	Link
ENG-CES	(Mareček, 2008)	Gold Alignment	http://ufal.mff.cuni.cz/czech-english-manual-word-alignment
ENG-DEU	EuroParl-based	Gold Alignment	www-i6.informatik.rwth-aachen.de/goldAlignment/
ENG-FAS	(Tavakoli and Faili, 2014)	Gold Alignment	http://eceold.ut.ac.ir/en/node/940
ENG-FRA	WPT2003, (Och and Ney, 2000),	Gold Alignment	http://web.eecs.umich.edu/ mihalcea/wpt/
ENG-HIN	WPT2005	Gold Alignment	http://web.eecs.umich.edu/ mihalcea/wpt05/
ENG-RON	WPT2005 (Mihalcea and Pedersen, 2003)	Gold Alignment	http://web.eecs.umich.edu/ mihalcea/wpt05/
ENG-CES	EuroParl (Koehn, 2005)	Parallel Data	https://www.statmt.org/europarl/
ENG-DEU	EuroParl (Koehn, 2005)	Parallel Data	https://www.statmt.org/europarl/
ENG-DEU	ParaCrawl	Parallel Data	https://paracrawl.eu/
ENG-FAS	TEP (Pilevar et al., 2011)	Parallel Data	http://opus.nlpl.eu/TEP.php
ENG-FRA	Hansards (Germann, 2001)	Parallel Data	https://www.isi.edu/natural-language/download/hansard/index.html
ENG-HIN	Emille (McEnery et al., 2000)	Parallel Data	http://web.eecs.umich.edu/mihalcea/wpt05/
ENG-RON	Constitution, Newspaper	Parallel Data	http://web.eecs.umich.edu/ mihalcea/wpt05/
All langs.	Wikipedia (downloaded October 2019)	Monolingual Text	download.wikimedia.org/[X]wiki/latest/[X]wiki-latest-pages-articles.xml.bz2

Table 11: Overview of datasets. “Lang.” uses ISO 639-3 language codes.

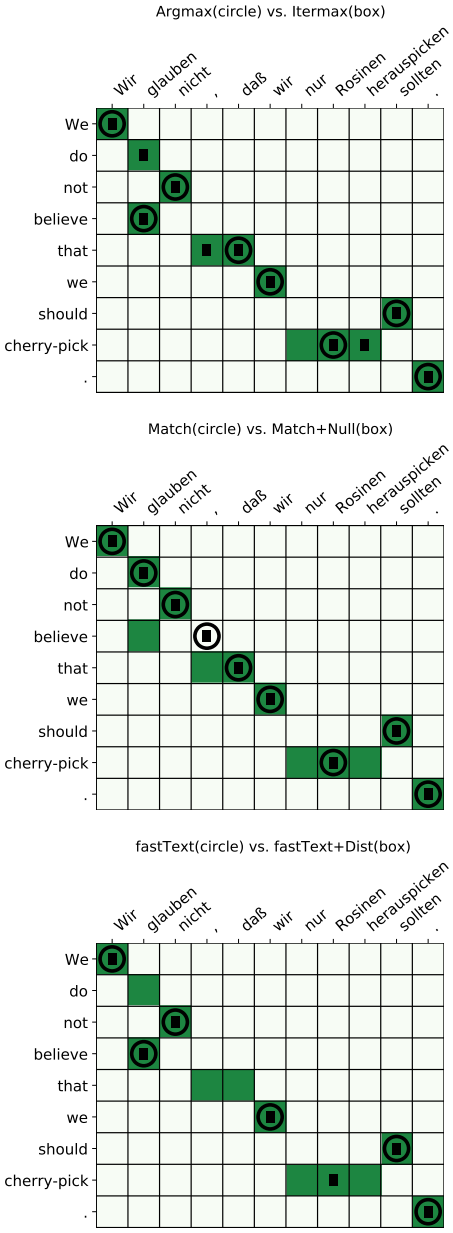

Figure 10: Comparison of alignment methods. Dark/light green: sure/possible edges in the gold standard. Circles are alignments from the first mentioned method in the subfigure title, boxes alignments from the second method.

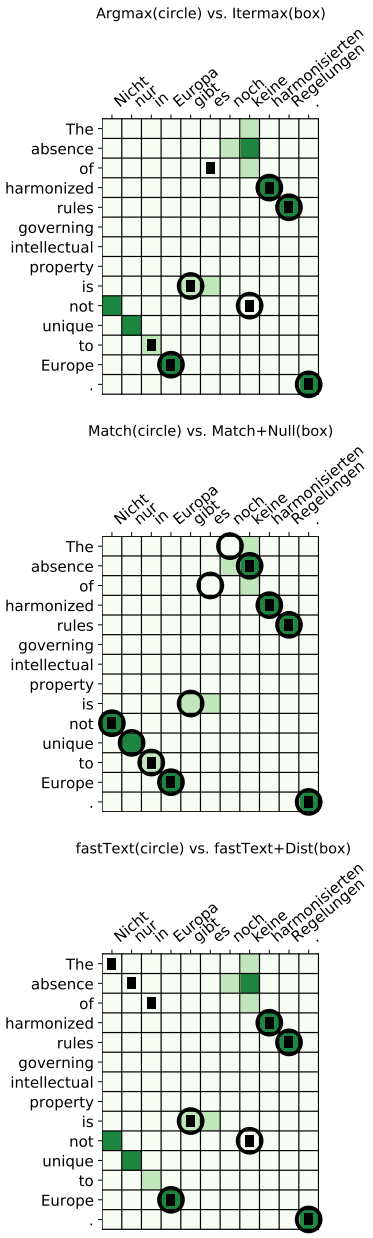

Figure 11: More examples.

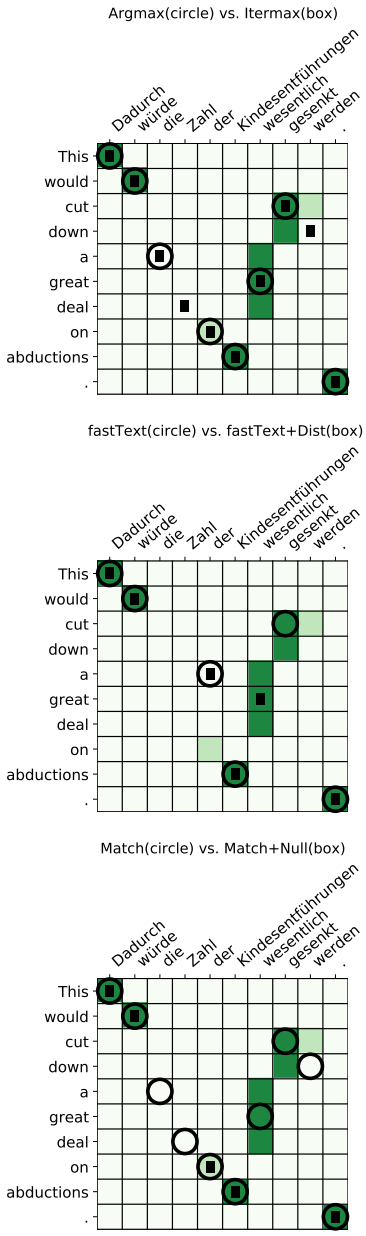

Figure 12: More examples.

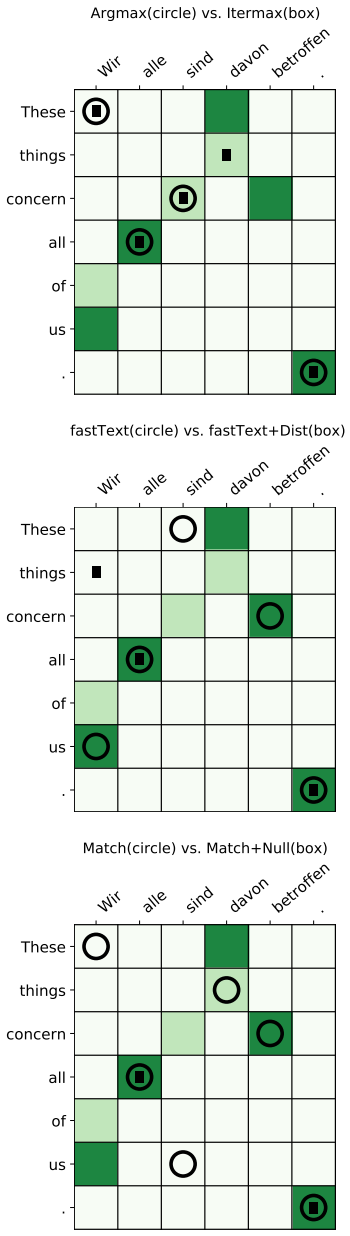

Figure 13: More examples.