Computational Linguistics in the Netherlands Journal 9 (2019) 101–116 Submitted 09/2019; Published 12/2019

Predicting syntactic equivalence between source and target sentences

Bram Vanroy BRAM.VANROY@UGENT.BE
Arda Tezcan ARDA.TEZCAN@UGENT.BE
Lieve Macken LIEVE.MACKEN@UGENT.BE

LT³, Language and Translation Technology Team, Ghent University

Abstract

The translation difficulty of a text is influenced by many different factors. Some of these are specific to the source text and related to readability while others more directly involve translation and the relation between the source and the target text. One such factor is syntactic equivalence, which can be calculated on the basis of a source sentence and its translation. When the expected syntactic form of the target sentence is dissimilar to its source, translating said source sentence proves more difficult for a translator. The degree of syntactic equivalence between a word-aligned source and target sentence can be derived from the crossing alignment links, averaged by the number of alignments, either at word or at sequence level. However, when predicting the translatability of a source sentence, its translation is not available. Therefore, we train machine learning systems on a parallel English-Dutch corpus to predict the expected syntactic equivalence of an English source sentence without having access to its Dutch translation. We use traditional machine learning systems (Random Forest Regression and Support Vector Regression) combined with syntactic sentence-level features as well as recurrent neural networks that utilise word embeddings and accurate morpho-syntactic features.

1. Introduction

In translation studies, equivalence is a concept that indicates how a source text and its translation can be compared to each other. It works on low-level language features such as morphology, lexicon, and syntax, as well as on higher-level, general text properties such as semantic, pragmatic, and cultural planes (Baker 2011). In the current study, we focus on syntactic — that is, *structural* — equivalence, which we operationalise as the amount of reordering that is necessary to transform a source sentence into a target sentence. When the syntactic equivalence between a source and target sentence is high (i.e. they are structurally similar), no or few reordering steps are needed to transform the source text's syntactic form into the target structure. When syntactic equivalence is low, many reordering steps are required to create a good translation. We investigate two approaches of quantifying syntactic equivalence. The first one is based on word alignment whereas the second focuses on the alignment of sequences of words. How exactly we calculate these metrics will be discussed in Section 3.1.

In light of the PreDicT project[1] (Predicting Difficulty in Translation), this study aims to estimate an English source sentence's syntactic equivalence to an implied Dutch translation without the need for that translation. Our previous study shows that syntactic equivalence is correlated with cognitive effort, and thus the translation difficulty of a text (Vanroy et al. 2019). More related research will be discussed in Section 2. To predict syntactic equivalence, we use the wealth of data available in word-aligned source and target sentences of an English-Dutch parallel corpus to train machine learning models that can predict a source sentence's syntactic equivalence. The dataset used will be discussed in Section 3.2, followed by a detailed run-down of the machine learning systems that we have put to the test (Section 3.3). Then we show the results for both the approach based on a

1. `https://research.flw.ugent.be/en/projects/predict`

word-alignment corpus and the one utilising sequence alignment (Section 4), followed by a discussion in Section 5. Finally, we will provide some hints towards future research in Section 6.

2. Related research

Translatability is a topic in translation studies that is still open to much debate. Historically, there has been much discussion of whether or not texts can in fact be truly translated, feeding an existential sense of untranslatability. This absolute viewpoint is much less present today, even though 'it is assumed that the perfect translation, i.e. one which does not entail any losses from the original is unattainable' (de Pedro (1999), p. 556). In this paper we understand translatability as the difficulty of translation rather than the more philosophical possibility or attainability of translation. It has been suggested that the difficulty of translating a text can be measured by readability formulas (Jensen 2009), and that it can be modelled by using only source text features (Mishra et al. 2013). However, empirical research found that a text's translatability is only in part related with its readability (Sun and Shreve 2014). In addition to source text properties, the difficulties that translators are faced with can also be attributed to language-pair specific features. One such feature is equivalence, or the lack thereof, between a source text and its translation (Sun 2015). As mentioned before, we put our attention to syntactic equivalence between a source and target text.

Even though quantitative research on syntactic equivalence with respect to human translation is scarce, syntactic equivalence is a much more discussed topic in the field of machine translation (MT), where it can be seen as synonymous for word reordering. Birch et al. (2008) show that the amount of reordering necessary between a source text and its translation is a strong predictor of the performance of a statistical machine translation (SMT) system. In other words, language pairs that require more reordering are more difficult to translate by SMT systems. The reaction to this particular translation difficulty was addressed by incorporating syntax into SMT systems. The interest for this problem was large, at the time, and many different solutions were proposed. Most of them require preprocessing the source text (often called *pre-reordering*) to better match the expected target sentence's structure. The extent of this topic surpasses the scope of the current paper, but for more details and different approaches see for instance Barone and Attardi (2013), Collins et al. (2005), Xia and McCord (2004), and Yamada and Knight (2001).

In recent years, advances in deep learning gave rise to neural machine translation (NMT) systems, which outperform SMT in terms of translation quality and yield fewer errors across almost all error types, including word order errors (Castilho and O'Brien 2017, Bentivogli et al. 2016, Van Brussel et al. 2018). Hence, researchers have posed the question whether pre-reordering steps are actually still necessary (Du and Way 2017). Due to access to more context in NMT and the complexity and fine-grained feature analysis that neural networks are capable of, it is no surprise that NMT can implicitly learn word orders from training a translation model. In fact, Toral and Sánchez-Cartagena (2017) show that the reorderings that NMT introduces are closer to the reorderings in the reference translations than those by SMT. To further improve their performance, efforts have been made to teach NMT systems the linguistic nuances of natural language, particularly syntax and word (re)order(ing) (Huang et al. 2018, Zhang et al. 2017). Du and Way (2017) found that preprocessing a source sentence by reordering it actually lowers an NMT system's performance. Instead they suggest an alternative approach that improves translation quality by adding linguistic knowledge such as part-of-speech (PoS) tags and word class to Japanese-to-English and Chinese-to-English NMT systems. This addition was inspired by the work of Sennrich and Haddow (2016) on NMT for the language pair German and English where they added lemmas, PoS tags, syntactic dependency labels, and morphological features to the input of the neural network, leading to an increased performance of the system. Similarly, on the task of detecting grammatical errors in SMT output, Tezcan et al. (2017) showed that word-level morpho-syntactic features, consisting of PoS, dependency and morphology information, yield better results than using word embeddings as a word representation technique.

This approach is closely related to the recurrent neural network (RNN) architecture that we test in this study on the task of predicting syntactic equivalence (cf. Section 3.3.3). In a multi-task set-up, Niehues and Cho (2017) successfully used PoS-tagging as a secondary task next to neural machine translation. The idea being that the model learns the importance of part-of-speech tags, and that this information propagates to the MT task. Eriguchi et al. (2016) provided evidence that attentional NMT systems can be extended with a linguistic tree representation of the source text for English-to-Japanese translation. A similar idea was worked out by Bastings et al. (2017), who made use of graph convolutional networks (GCN) to encode the source text as syntax-aware word representations through syntactic dependency trees. This information contributes to improve over their baseline without syntactic information for English-German and English-Czech. Conversely, presenting the target text as a linguistic structure, Aharoni and Goldberg (2017) found that using a string-to-tree model can improve the performance of German-to-English NMT over a traditional string-to-string variant. Here, the target text is presented as a linearised and lexicalised constituency tree. See Currey and Heafield (2018) for a non-exhaustive yet comprehensive overview of efforts to incorporate syntax into RNN-based NMT systems. The authors also introduce their own approach of injecting syntactic information into an English-to-German NMT system by using both the source text as well as its linearised constituency parse as input. Especially the multi-task system performed well, improving over the baseline.

Generally speaking there seems to be an iterative process of linguistic features being added to existing translation systems to try to further improve their performance.

Even though the above only highlights the difficulties that MT systems experience as a result of the lack of linguistic knowledge, there is also some evidence that suggests that the syntax of a source and of its target text play a role for human translators. Bangalore et al. (2015) found that syntactic variation — that is, syntactic entropy — correlates with cognitive effort. In other words, the more possible variations in the target text structure, the more difficult the translation process is. In this paper, we are more interested in syntactic equivalence between a source and target text rather than the syntactic entropy of a given source sentence. Translation difficulty is often reduced to source text features such as its readability, but Sun and Shreve (2014) (reiterated in Sun (2015)) use the term *equivalence* to discuss translation-specific difficulty that is not solely related to the source text. The (syntactic) equivalence between a source text and its (possible) translation can cause difficulties for a translator when the source and target syntactic structure differ significantly. We refer the reader to a previous study (Vanroy et al. 2019), where we correlated translation process features with syntactic equivalence. We found that, indeed, a correlation exists between word reordering and a number of translation process features (taken from duration, revision and gaze categories) as a proxy for cognitive effort and, thus, translation difficulty. We take this to mean that the more transformations the source word order has to undergo during translation, the more cognitive effort is required by the translator. Considering that the goal of PreDicT is to build a translatability predicting system, we wish to model syntactic equivalence of a sentence off-line, that is, without the need of a target sentence which brings us to the current study.

3. Methodology

First, we elaborate on how we quantify syntactic equivalence by distinguishing two approaches in Section 3.1: word alignment and sequence alignment. Then we discuss the used dataset (Section 3.2), followed by the experimental set-up consisting of a baseline (3.3.1), a traditional machine-learning (ML) approach using sentence-level features (3.3.2), and finally a neural network that uses word-level features (3.3.3).

3.1 Alignment types

We present two similar approaches to quantify syntactic equivalence: the first quantifies how words have moved position during translation; the second takes the movement of word sequences into account rather than single words. The core idea is that we calculate syntactic equivalence as the number of times alignment links cross each other (hence its name *cross* value), averaged by the number of alignment links. Visual examples are given below.

Our first approach works on the word level: by looking at how each individual word has moved with respect to other words in the sentence, we calculate its cross value. Our measure of syntactic equivalence is bidirectional (or symmetrical) and applicable to either translation direction (English-to-Dutch or Dutch-to-English). This contrasts with Carl et al. (2016), who introduce a similar metric but which is asymmetrical, i.e. the resulting cross value differs depending on the translation direction. Typically, a word-aligned corpus is represented in the Pharaoh format. The format requires that every alignment link is shown as a pair of source and target word indices s_i-t_j. The indices indicate the position of that token in its sentence. An example is given below where 1a is the source sentence, 1b the target sentence, and 2 a representation of the word alignments in Pharaoh format.

(1) a. Sometimes she asks me why I used to call her father Harold .
    b. Soms vraagt ze waarom ik haar vader Harold noemde .

(2) 0-0 1-2 2-1 4-3 5-4 6-8 7-8 8-8 9-5 10-6 11-7 12-9

We can visualise this as follows in Figure 1. The arrows indicate the alignment links, i.e. where a given source word has moved to in the target sentence. The circles highlight where alignments cross one another; these are the *crosses*. In this example, we count ten crosses. This value is then averaged by the number of alignments (arrows) to get our final, average cross value of the whole sentence. In this case that is $^{10}/_{12} = 0.8333....$

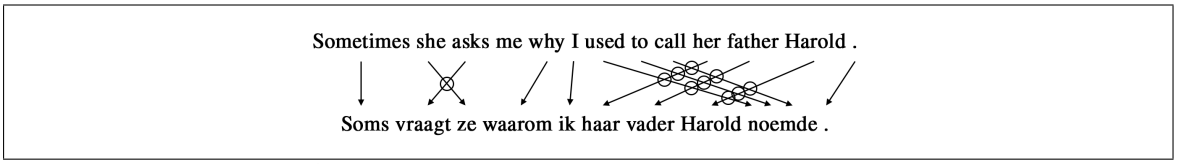

Figure 1: A visual representation of word alignment and cross values.

The second approach that we tested is based on sequential words that move together as a group. The intuition here is that words that move together are one unit and as such can count as one alignment link. To this end, we seek the longest possible word sequence alignments between the source and target sentences with the following criteria:

(i) each word in the source sequence is aligned to at least one word in the target sequence and vice versa;

(ii) each word in the source word sequence is only aligned to word(s) in the target word sequence and vice versa;

(iii) none of the alignments between the source and target word sequences cross each other.

The visualisation of Example 1 as sequence alignment in Figure 2 makes this clear. To illustrate: *why I* is seen as a sequence because they are sequential (there is no aligned word between them nor between their translations), and the order is the same (*why* is aligned with *waarom* which stands before the $I \rightarrow ik$ alignment link), so they do not cross each other. These alignments are also shown in Pharaoh format in Example 3.

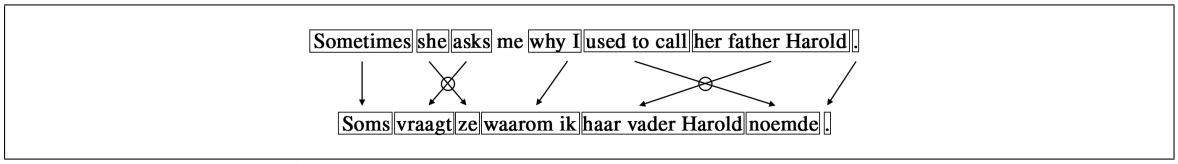

Figure 2: A visual representation of sequence alignment and cross values.

(3) 0-0 1-2 2-1 3-3 4-5 5-4 6-6

In the example, the cross value based on sequences is $^2/_7 = 0.286$. We would argue that, intuitively, using crosses of sequence alignment better represents the syntactic shifts that a source sentence has to go through to become the target sentence because it indicates crossing groups of words rather than single entities. Therefore, our hypothesis is that cross values based on sequence alignment can be modelled better than those based on word alignment. To distinguish systems that use word alignment as input from those using sequence alignment, we will identify the former with `WORD` and the latter with `SEQ`.

Even though not similar, Birch et al. (2008) propose a word reordering metric that works by maximising aligned block pairs, too. The authors use this metric as a predictor for the performance of machine translation systems. A large difference with our approach is that they average their metric by the number of source tokens in a sentence whereas we average by the number of alignments. This is important, because in our approach, both for word alignment and sequence alignment, calculating cross is direction-agnostic, meaning that going from source text to target sentence or vice-versa will yield the same cross value. As such it is, indeed, a syntactic equivalence metric that compares two sentences irrespective of the translation direction. Do note that sequences are not linguistically motivated entities but only sequences that move as a single unit. Sequences of words have been used as (structurally useful) phrases before, perhaps most notably by the (H)TER evaluation metric which takes sequence shifts into account to calculate the edit distance between a translation and a reference ((Human-targeted) Translation Edit Rate; Snover et al. (2006)).

Because our aim is to predict syntactic equivalence automatically, which we measure with cross values in this study, we rely on automatic word alignment methods, which are error-prone. Therefore, we first analyse the quality of two widely-used automatic word alignment systems in order to (i) ensure that automatic methods are sufficiently accurate compared to human alignments; and (ii) to use the automatic method with the highest alignment quality in the remainder of this study. In the following test we compare the human, manual alignments (MA) of a small corpus, taken from the work of Macken (2010), to the output of GIZA++ (Och and Ney 2003) and `fast_align` (FA) (Dyer et al. 2013) to see whether the quality of automatic word alignment systems is acceptable, and which tool performs best. For both GIZA++ and FA, we use the `grow-diag-final-and` algorithm, which starts with the intersection of the forward and backward alignments and then adds additional alignment points (Koehn et al. 2003). The dataset contains 143 sentences after filtering out outlying sentences that consist of less than three or more than 50 tokens in either source or target text. As word alignment tools work better with larger corpora, we obtained word alignments with GIZA++ and FA after appending the manually aligned data to the Dutch Parallel Corpus (DPC; Macken et al. (2011)). We evaluate the alignments between MA and GIZA++ and MA and FA with Alignment Error Rate (AER), as proposed by Mihalcea and Pedersen (2003), which is based on earlier work by Och and Ney (2000). We use the AER implementation of NLTK in Python (Bird et al. 2009).

	MA-GIZA++ ↓	MA-FA ↓
Min.	0.0	0.0
Max.	0.5758	0.6471
Mean	0.0822	0.1127
Median	0.0189	0.0691
SD	0.1110	0.1329

Table 1: Statistics about AER calculated on MA-GIZA++ and MA-FA.

Table 1 shows that with mean AER 0.0822 and median AER 0.0189, GIZA++ is closer to manual alignments than FA (0.1127 and 0.0691, resp.). These results are in line with earlier findings by Peter et al. (2017) where GIZA++ outperformed `fast_align` in terms of alignment quality. We consider both AER scores to be sufficient for our task and that these automatic word alignment tools can be used as a proxy for manual alignment. Because of its better alignment quality, we will use GIZA++ as our word alignment tool of choice.

3.2 Data set

In our experiments we use the Dutch Parallel Corpus (Macken et al. 2011), which, as the name implies, is a parallel corpus that centres around Dutch as its core language. We make use of both English-to-Dutch and Dutch-to-English parts of the corpus, which in total contain 148,421 sentences after removing duplicates and sentences that are shorter than 3 tokens or longer than 70 tokens. The training set consists of 144,421 sentences, leaving 2,000 sentences for the validation set and 2,000 sentences for the test set. The data was tokenised and lower-cased by using the preprocessing scripts[2] provided by Moses (Koehn et al. 2007).

3.3 Experimental set-up

The objective of this paper is to predict the cross value of an English source sentence without having access to the Dutch target sentence. As mentioned before, we predict cross values based on word alignments as well as cross values based on sequence alignments. We train two types of systems with their own feature sets. The first type contains traditional ML systems combined with sentence-level features and the second one makes use of recurrent neural networks with word-level features. Additionally, we compare the estimation performance of the two ML approaches to a mean baseline, which we discuss first.

3.3.1 Mean baseline

As exemplified above, the cross values for the word alignment and sequence alignment of a sentence can differ. The mean cross value of the whole training set for `WORD` is 1.02 and for `SEQ` it is 0.91. We use these mean values as a baseline. More explicitly: for all 2,000 sentences in the test set, this baseline predicts 1.02 and 0.91 as cross values obtained from word and sequence alignments, respectively. The results for this approach will be referenced as `mean baseline` below.

3.3.2 Sentence-level features and RFR/SVR

In our traditional ML systems, we use sentence-level features as input (shown in 4), derived from the sentence level which contrasts with the word features in Section 3.3.3. These features have been chosen because — from a linguistic point-of-view — they provide information about the syntactic structure of a sentence. We used the Python package spaCy (Honnibal and Montani 2017) to extract the required information from the source sentences.

2. `https://github.com/moses-smt/mosesdecoder/tree/master/scripts/tokenizer`

(4)
- parse tree depth
- sentence length
- # coordinating conjunctions
- # subordinating conjunctions
- # punctuation marks
- # content words (adjectives, (proper) nouns, and verbs)
- # subjects
- # objects

We employ Random Forest Regression (`rfr`) and Support Vector Regression (`svr`) from the Python scikit-learn package (Pedregosa et al. 2011).[3] Both `rfr` and `svr` are optimised through grid search with mean squared error (MSE) as the criterion to minimise between predicted and actual values. For `rfr` we tuned the number of trees (`n_estimators`) and found the best results with 500 trees (`WORD`) and 1000 in (`SEQ`). In the case of `svr`, the best parameters were $C = 1$ (C is the penalty of the error term), $\epsilon = 0.01$ (ϵ is the penalty-free distance with respect to training loss), using a radial bias function kernel. This is the case for both `WORD` and `SEQ`.

3.3.3 Word-level features and RNN

In the previous section, we introduced the traditional machine learning systems that we use. These systems are relatively fast and rather intuitive in that the features are hand-crafted. This can also be a downside, however: feature extraction is a time-consuming process and the generated features are often incomplete. Neural networks, on the other hand, try to learn high-level features from data and eliminate the need of domain expertise and feature engineering. Combined with the success of word embeddings (Mikolov et al. 2013, Pennington et al. 2014), especially in the last decade, neural networks have been producing superior results compared to traditional machine learning algorithms on various natural language processing tasks, including named entity recognition (Turian et al. 2010), parsing (Socher et al. 2011a), sentiment analysis (Socher et al. 2011b), machine translation (Cho et al. 2014) and quality estimation of MT (Deng et al. 2018). Inspired by previous work on NLP and in addition to the traditional machine learning techniques described above, we use an RNN architecture for the task of predicting syntactic equivalence for a source sentence and an implied translation. RNNs can learn from a sequence of inputs rather than a single data point which makes them ideal for NLP tasks because sentences are sequences of words. Instead of representing a sentence as a single set of sentence-level features, which is the only option in the traditional ML systems, neural networks allow us to use a sentence as a set of words, which all have their own features. In other words, RNNs allow a sentence to be represented as a sequence of words, whereas traditional systems can only process a sentence as one unit. The RNN architecture that we have used will be discussed later in this section.

Word embeddings represent a word as a vector of size `n` based on its context and co-occurrences in a text. Each dimension in such a vector represents a latent feature of a given word, capturing useful syntactic and semantic properties (Turian et al. 2010). Despite its success and popularity on various NLP tasks, Tezcan et al. (2017) suggest that word embeddings, as a word representation technique, should not be considered as a one-size-fits-all approach. On the task of detecting grammatical errors in statistical machine translation (SMT) output, they report a marked improvement in performance over word embeddings by using accurate morpho-syntactic features. Moreover, in a more recent study, Tezcan et al. (2019) showed that the combination of morpho-syntactic features and word embeddings maximised the performance of an RNN system, in comparison to using either type of information alone, on the task of detecting all types of fluency errors in SMT output, consisting of both semantic and grammatical error types. Both studies suggest that such morpho-syntactic features provide

3. For more information about the fine-tuned parameters that we use in our experiments, see the package's documentation `https://scikit-learn.org/stable/documentation.html`.

complementary information to word embeddings when syntactic properties of texts are important in a given task. Considering the syntactic nature of the task at hand, namely predicting syntactic equivalence in translation, we use morpho-syntactic features, as an alternative word representation technique.

As described in Tezcan et al. (2017), we transform each token of a given sentence into its morpho-syntactic representation, in the form of a multi-hot encoded vector that provides accurate information about its part-of-speech tag, dependency label, and morphology. For every word, the vector values are set to 0 except for the morpho-syntactic features that apply to it, which are set to 1. Figure 3 shows the morpho-syntactic features that are extracted for the word *is*, in a given English sentence. In this example, the morpho-syntactic feature vector of the word *is* consists of all zeros except for the fields that represent its PoS tag (*VBZ*), dependency label (*Root*), and morphology information (*finite*, *present tense*, *singular*, and *third person*).

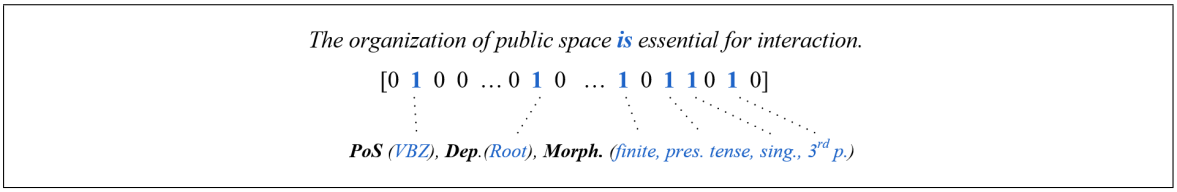

Figure 3: A visual representation of how tokens are represented as morpho-syntactic, multi-hot encoded features.

We used spaCy (Honnibal and Montani 2017) to extract the aforementioned morpho-syntactic features on the English part of our dataset, as discussed in Section 3.2. We set the length of morpho-syntactic feature vectors to 147, namely the total number of possible features obtained by spaCy. The word embedding model was trained using word2vec (Mikolov et al. 2013) on a merged data set consisting of the English part of our dataset and the English news crawl dataset from the WMT shared task of 2017[4]. To keep the amount of information provided to the RNN system by the two types of input vectors balanced, we trained word embedding models with 147 dimensions.

To go into more detail about the architecture: we built an RNN architecture such that each input vector, word embedding or morpho-syntactic feature vector, is fed into a dedicated Bidirectional Gated Recurrent Unit (BiGRU). GRUs are a specialised variant of RNNs, which are well suited for capturing long range dependencies on multiple time scales (Cho et al. 2014). Bi-directional GRUs consist of two recurrent layers, each processing the given input sequence in opposite direction, as a means to overcome the generalisation limitations of RNNs in general, which have the tendency to represent recent input nodes better (Bahdanau et al. 2014). As the final output of a GRU, we take the concatenation of the last state of each layer. When both morpho-syntactic features and word embeddings are provided as input, the outputs of both BiGRU layers are concatenated before they are connected to the output layer, which predicts the cross value for a given input sentence using a linear activation function. To help prevent overfitting, we apply dropout in the BiGRU layers (for the input gates and the recurrent connections) and between the BiGRU layers and the output layer (Srivastava et al. 2014).

We test three RNN architectures on this task, with different word representation combinations as input:

- `rnn_ms`: Only morpho-syntactic features
- `rnn_w2v` Only word embeddings
- `rnn_ms+w2v`: Both morpho-syntactic features and word embeddings

All systems were built with Keras (Chollet 2015) on top of a TensorFlow backend (Abadi et al. 2015), using the Adam optimiser with a learning rate of 1e−3, hidden layer of size 200 and a

4. Available at `http://data.statmt.org/wmt17/translation-task/`.

batch size of 200. We used *tanh* as activation function between input and hidden layers, and *linear* activation between hidden and output layers. As loss function, we used MSE. All systems were trained for 100 epochs. We trained each system with three different dropout values, namely 0, 0.2 and 0.4, and kept the model that performed best on the development set as the best model. Figure 4 illustrates the proposed RNN architecture, which takes both morpho-syntactic features and word embeddings as input.

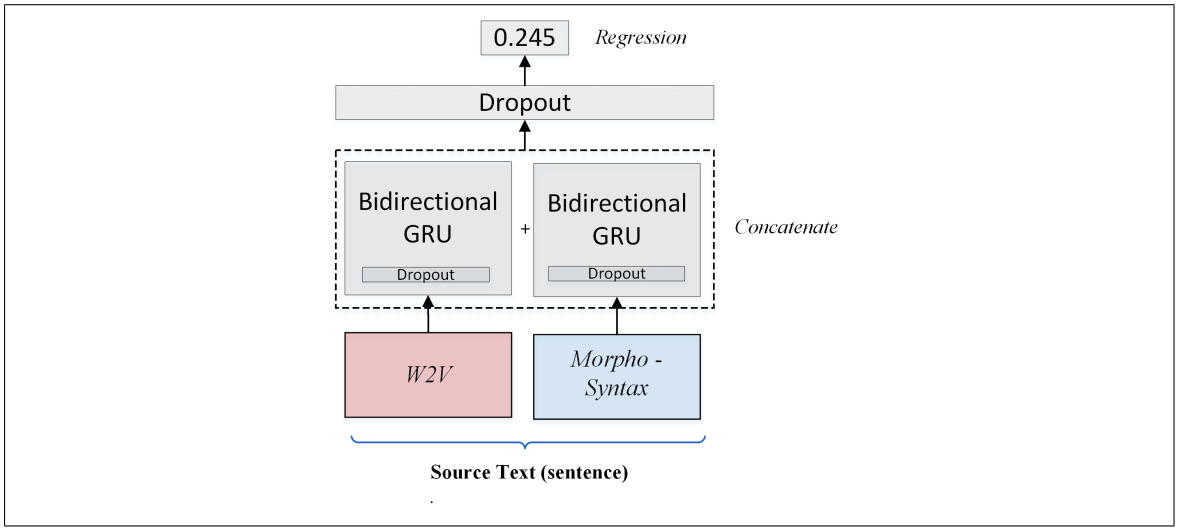

Figure 4: A visual representation of `rnn_ms+w2v`.

4. Results

In this section, we use the Pearson correlation (r) between the predicted values and the actual cross values as our primary evaluation metric. This is done in order to be able to compare the results from `WORD` and `SEQ`: a sentence's cross value based on word alignment is different from the cross value based on sequence alignment. We also provide values for mean absolute error (MAE). MSE and MAE should be minimised whereas Pearson's r has to be maximised. In all results below, it holds that the Pearson correlations are significant ($p < .01$). The figures that are given here show all metrics (MSE, MAE and Pearson r) on the Y-axis, and all tested systems on the X-axis. Pearson r values are given in boldface, MAE in italics.

The predictive performance of the systems that we built using input based on word and sequence alignments are provided in Figure 5 and 6, respectively.

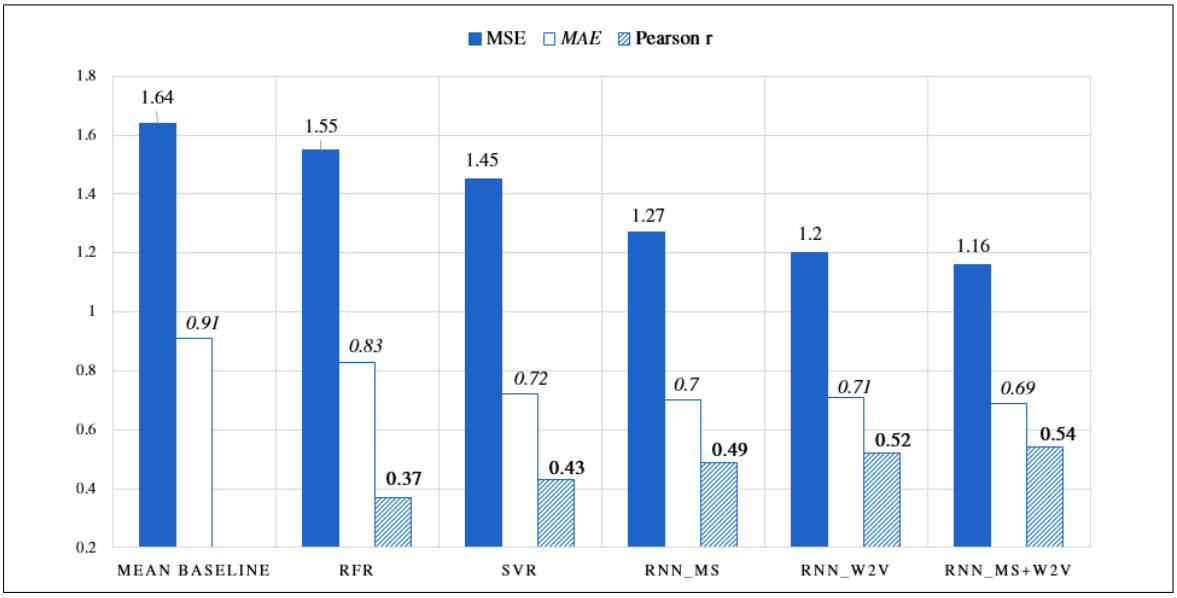

Figure 5: Visualisation of results for `WORD` (dataset mean of 1.64).

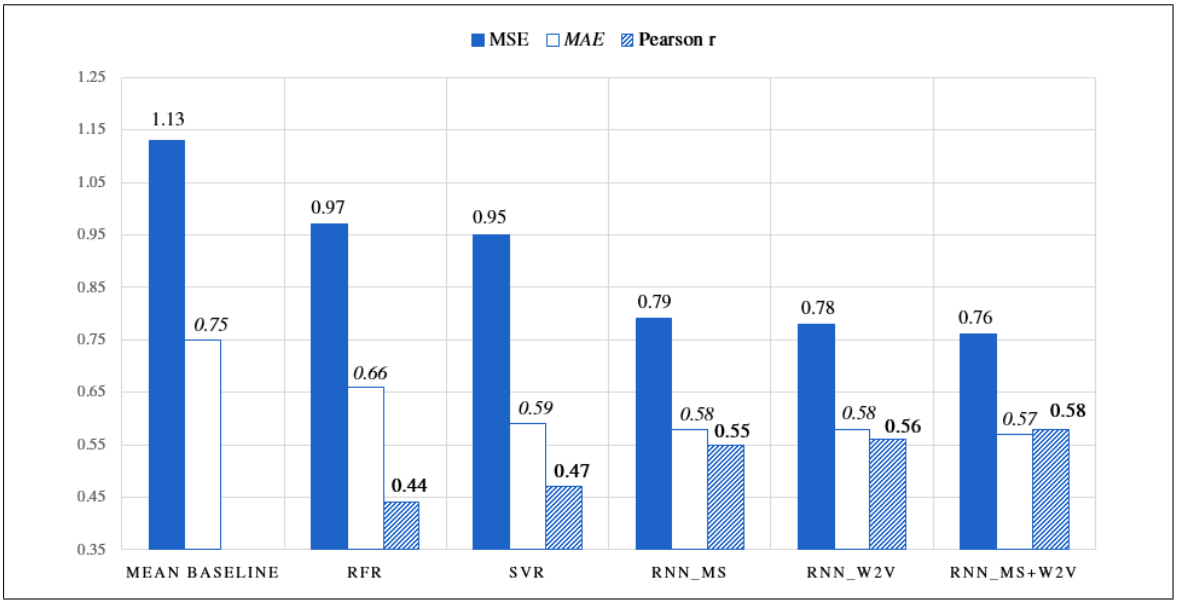

Figure 6: Visualisation of the results for `SEQ` (dataset mean of 0.91).

In a last graph (Fig. 7), the performance difference between systems using input based on word alignment and sequence alignment is highlighted.

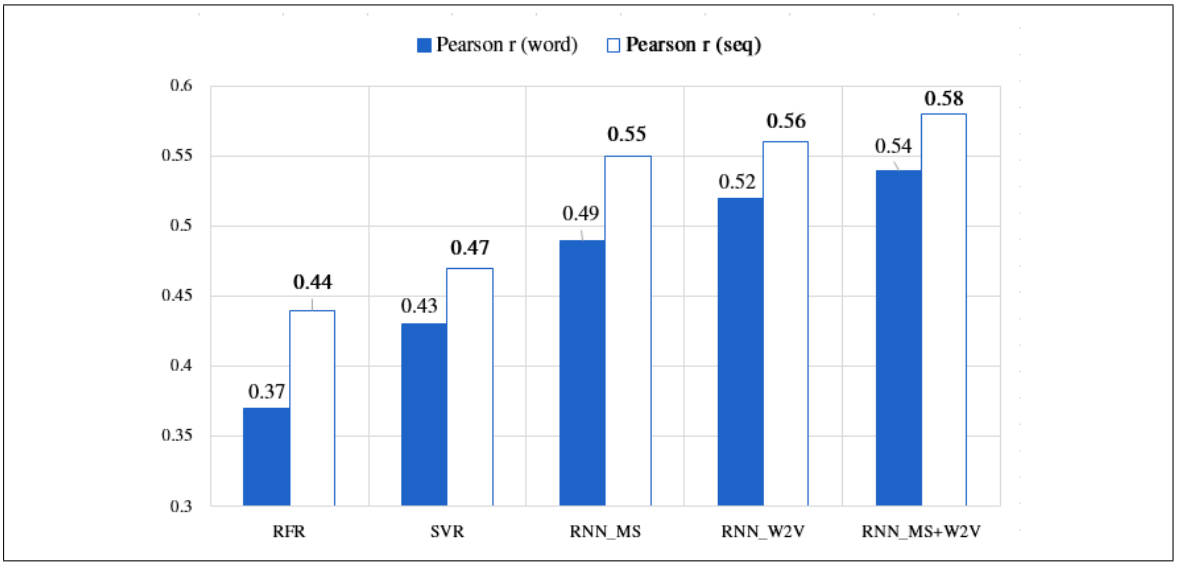

Figure 7: Comparison of `WORD` and `SEQ`.

5. Discussion

The goal of this paper was to predict a Dutch source sentence's syntactic equivalence to an implicit English translation. To this end we introduced our version of a *cross* value, which can be based on word alignments (`WORD`) as well as on sequence alignments (`SEQ`), as discussed in Section 3.1. We found that traditional machine learning systems (`rfr` and `svr`) are less performant than recurrent neural networks (`rnn_*`) across the board. Furthermore, `SEQ` outperforms the `WORD` counterpart in all scenarios.

The neural network architectures with word-level features perform better than the traditional machine learning systems using sentence-level features. The best performing traditional ML system, `svr`, reaches a Pearson correlation of 0.43 (`WORD`) and 0.47 (`SEQ`), outperforming `rfr` (0.37 `WORD`, 0.44 `SEQ`). All RNN-architectures achieve better results, though. The reason for recurrent neural networks performing better than traditional ML is two-fold, and already touched upon before in Section 3.3. On the one hand, the traditional systems require single data point features as input, meaning that a sentence can only be represented as a number of features (cf. Section 3.3.2). These features are thus more coarse grained and not as detailed as word-level features. In contrast, when using neural networks a sentence can be represented as sequences of features. In other words, rather than having a single set of features for a sentence, that sentence can be represented as a sequence of word-level features, which gives much more detailed information to the system. Particularly, *recurrent* neural networks allow for the propagation of information through the sequence. This means that the final output takes into account the order of the words as well as the information of each word. On top of that, the architecture of our tested traditional ML systems are fundamentally different from neural networks. The latter is much more capable of modelling data-specific peculiarities while at the same time generalising sufficiently.

For the neural network systems, we can see that using only morpho-syntactic features (0.49 `WORD`, 0.55 `SEQ`) performs slightly worse than using only word embeddings (0.52 `WORD`, 0.56 `SEQ`). Because the task at hand is syntactic rather than semantic, it is worth expanding on the performance of word embeddings compared to morpho-syntactic features. Because our task is specifically directed at modelling syntactic changes, we expected a high importance of the morpho-syntactic features. Our morpho-syntactic features are very specific to each word in its context and role in the sentence whereas word embeddings are more general representations of words. Word2vec models do not

specifically model syntax or morphology or even semantics; rather, they represent each word-type in relation to each other, implicitly modelling all kinds of language features, including morpho-syntactic and semantic. Because of the different goals of both word representations techniques (one specifically morpho-syntactic, the other more general), we had especially hypothesised them to be complementary, leading to a performance boost when combined. This is indeed the case, with a correlation of 0.54 (`WORD`), and 0.58 (`SEQ`).

Finally, comparing the Pearson correlations of both `WORD` and `SEQ`, we clearly see that cross values based on sequence alignment can be modelled better than those using word alignment.

In this paper we have presented a number of systems that can predict a source text's syntactic equivalence with an implicit translation, i.e. without needing an actual translation. In our tests, we reached a Pearson correlation of 0.58 but in future work we tend to improve this with more complex neural networks. Furthermore, and in line with Tezcan et al. (2019) who worked on a classification problem, we showed that for this specific task, a regression problem focused on a sentence's syntax, a morpho-syntactic component can be successfully used to improve the quality of predictions.

6. Future work

We have presented a way of calculating syntactic equivalence (with cross values) based on computational phrases, that is, phrases that are algorithmically created. Building phrases in this way is often used in automatic systems to create alignments between source and target sentences. However, we would also like to take the linguistic route, and compute the cross values based on linguistically motivated phrases. The phrases can be extracted automatically by using a constituency parser, but this introduces yet another automatic component prone to errors. Additionally, theoretical questions need to be answered concerning how the constituency tree should be segmented, and how to deal with linguistic phenomena such as separable verbs, conjunctions, (in)direct speech, interjected adverbs, and so on. Despite these challenges, we think linguistic phrases can improve performance over algorithmic phrases.

In our experiment we used recurrent neural networks, and even though they are powerful, they have been surpassed by the transformer architecture (Vaswani et al. 2017) in many natural language tasks. In future endeavours we will use transformers and investigate how well they perform for our given task.

The objective of the PreDicT project is to build a system that can predict a source sentence's translation difficulty. The present study discussed one feature (syntactic equivalence) that plays a role in predicting translatability, but we plan to test more features and add those to the final system. These features include semantic information such as word translation entropy, but also source text specific features such as syntactic and semantic complexity. Finally, language (pair) specific difficulties can be added, for instance the translation of the English -ing form to Dutch.

References

Abadi, Martın, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng (2015), TensorFlow: Large-scale machine learning on heterogeneous distributed systems. https://www.tensorflow.org/.

Aharoni, Roee and Yoav Goldberg (2017), Towards string-to-tree neural machine translation, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume*

2: Short Papers), Association for Computational Linguistics, Vancouver, Canada, pp. 132–140. https://www.aclweb.org/anthology/P17-2021.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2014), Neural machine translation by jointly learning to align and translate, *CoRR*.

Baker, M. (2011), *In other words: A coursebook on translation*, 2 ed., Routledge, Abingdon, UK.

Bangalore, Srinivas, Bergljot Behrens, Michael Carl, Maheshwar Gankhot, Arndt Heilmann, Jean Nitzke, Moritz Schaeffer, and Annegret Sturm (2015), The role of syntactic variation in translation and post-editing, *Translation Spaces* **4** (1), pp. 119–144.

Barone, Antonio Valerio Miceli and Giuseppe Attardi (2013), Pre-reordering for machine translation using transition-based walks on dependency parse trees, *Proceedings of the eighth workshop on statistical machine translation*, Association for computational linguistics, Sofia, Bulgaria, pp. 164–169.

Bastings, Joost, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an (2017), Graph convolutional encoders for syntax-aware neural machine translation, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Copenhagen, Denmark, pp. 1957–1967. https://www.aclweb.org/anthology/D17-1209.

Bentivogli, Luisa, Arianna Bisazza, Mauro Cettolo, and Marcello Federico (2016), Neural versus phrase-based machine translation quality: A case study, *CoRR*. http://arxiv.org/abs/1608.04631.

Birch, Alexandra, Miles Osborne, and Philipp Koehn (2008), Predicting success in machine translation, *Proceedings of the conference on empirical methods in natural language processing (EMNLP 2008)*, Association for computational linguistics, Honolulu, Hawaii, pp. 745–754.

Bird, Steven, Edward Loper, and Ewan Klein (2009), *Natural language processing with Python*, O'Reily Media Inc.

Carl, Michael, Moritz Jonas Schaeffer, and Srinivas Bangalore (2016), The CRITT translation process research database, *in* Carl, Michael, Srinivas Bangalore, and Moritz Jonas Schaeffer, editors, *New directions in empirical translation process research*, New frontiers in translation studies, Springer, Cham, Switzerland, pp. 13–54.

Castilho, Sheila and Sharon O'Brien (2017), Acceptability of Machine-Translated Content: A Multi-Language Evaluation by Translators and End-Users, *Linguistica Antverpiensia* **16**, pp. 120–136.

Cho, Kyunghyun, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio (2014), Learning phrase representations using RNN encoder-decoder for statistical machine translation, *arXiv e-prints*. arXiv: 1406.1078.

Chollet, François (2015), Keras. https://keras.io/getting-started/faq/.

Collins, Michael, Philipp Koehn, and Ivona Kučerová (2005), Clause restructuring for statistical machine translation, *Proceedings of the 43rd annual meeting on association for computational linguistics (ACL 2005)*, Association for computational linguistics, Ann Arbor, Michigan, pp. 531–540.

Currey, Anna and Kenneth Heafield (2018), Multi-source syntactic neural machine translation, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2961–2966.

de Pedro, Raquel (1999), The translatability of texts: A historical overview, *Meta* **44** (4), pp. 546–559, Les Presses de l'Université de Montréal.

Deng, Yongchao, Shanbo Cheng, Jun Lu, Kai Song, Jingang Wang, Shenglan Wu, Liang Yao, Guchun Zhang, Haibo Zhang, Pei Zhang, Changfeng Zhu, and Boxing Chen (2018), Alibaba's neural machine translation systems for WMT18, *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, Association for Computational Linguistics, Belgium, Brussels, pp. 368–376. https://www.aclweb.org/anthology/W18-6408.

Du, Jinhua and Andy Way (2017), Pre-reordering for neural machine translation: Helpful or harmful?, *The Prague bulletin of mathematical linguistics* **108** (1), pp. 171–182.

Dyer, Chris, Victor Chahuneau, and Noah A Smith (2013), A simple, fast, and effective reparameterization of IBM model 2, *Proceedings of NAACL-HLT 2013*, Association for Computational Linguistics, Atlanta, Georgia, USA, pp. 644–648.

Eriguchi, Akiko, Kazuma Hashimoto, and Yoshimasa Tsuruoka (2016), Tree-to-sequence attentional neural machine translation, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Berlin, Germany, pp. 823–833. https://www.aclweb.org/anthology/P16-1078.

Honnibal, Matthew and Ines Montani (2017), spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing, To appear.

Huang, Po-Sen, Chong Wang, Sitao Huang, Dengyong Zhou, and Li Deng (2018), Towards neural phrase-based machine translation, *Proceedings of International Conference on Learning Representations*, Vancouver, Canada.

Jensen, Kristian T. H. (2009), Indicators of text complexity, *in* Göpferich, Susanne, Arnt Lykke Jakobsen, and Inger Mees, editors, *Copenhagen Studies in Language*, Vol. 37, Samfundslitteratur, Copenhagen, Denmark, pp. 61–80.

Koehn, Philipp, Franz Josef Och, and Daniel Marcu (2003), Statistical phrase-based translation, *Proceedings of NAACL-HLT 2003*, Edmonton, Canada, pp. 48–54.

Koehn, Philipp, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, Evan Herbst, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, and Christine Moran (2007), Moses: Open source toolkit for statistical machine translation, *Proceedings of the ACL 2007 demo and poster sessions*, Prague, Czech Republic, pp. 177–180.

Macken, Lieve (2010), *Sub-sentential alignment of translational correspondences*, Phd thesis, Ghent University, Ghent, Belgium.

Macken, Lieve, Orphée De Clercq, and Hans Paulussen (2011), Dutch parallel corpus: A balanced copyright-cleared parallel corpus, *Meta: Journal des traducteurs* **56** (2), pp. 374–390.

Mihalcea, Rada and Ted Pedersen (2003), An evaluation exercise for word alignment, *Proceedings of the HLT-NAACL 2003 Workshop on Building and using parallel texts data driven machine translation and beyond*, Vol. 3, Association for Computational Linguistics, Edmonton, Canada, pp. 1–10.

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean (2013), Efficient estimation of word representations in vector space, *arXiv e-prints*. arXiv: 1301.3781.

Mishra, Abhijit, Pushpak Bhattacharyya, and Michael Carl (2013), Automatically predicting sentence translation difficulty, *Proceedings of the 51st Annual Meeting on Association for Computational Linguistics (ACL 2013)*, Sofia, Bulgaria, pp. 346–351.

Niehues, Jan and Eunah Cho (2017), Exploiting linguistic resources for neural machine translation using multi-task learning, *Proceedings of the Second Conference on Machine Translation*, pp. 80–89.

Och, Franz Josef and Hermann Ney (2000), A comparison of alignment models for statistical machine translation, *Proceedings of the 18th conference on computational linguistics*, Vol. 2, Association for Computational Linguistics, Saarbrücken, Germany, pp. 1086–1090.

Och, Franz Josef and Hermann Ney (2003), A systematic comparison of various statistical alignment models, *Computational Linguistics* **29** (1), pp. 19–51.

Pedregosa, Fabian, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, and David Cournapeau (2011), Scikit-learn: Machine learning in Python, *Journal of machine learning research* **12**, pp. 2825–2830.

Pennington, Jeffrey, Richard Socher, and Christopher Manning (2014), GloVe: Global vectors for word representation, *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.

Peter, Jan-Thorsten, 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), pp. 27–36.

Sennrich, Rico and Barry Haddow (2016), Linguistic input features improve neural machine translation, *Proceedings of the First Conference on Machine Translation: Volume 1, Research Papers*, pp. 83–91.

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul (2006), A study of translation edit rate with targeted human annotation, *Proceedings of association for machine translation in the Americas*.

Socher, Richard, Cliff C. Lin, Andrew Y. Ng, and Christopher D. Manning (2011a), Parsing natural scenes and natural language with recursive neural networks, *Proceedings of the 26th International Conference on Machine Learning (ICML)*.

Socher, Richard, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning (2011b), Semi-supervised recursive autoencoders for predicting sentiment distributions, *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 151–161.

Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov (2014), Dropout: A simple way to prevent neural networks from overfitting, *Journal of machine learning research* **15** (1), pp. 1929–1958.

Sun, Sanjun (2015), Measuring translation difficulty: Theoretical and methodological considerations, *Across languages and cultures* **16** (1), pp. 29–54. http://www.akademiai.com/doi/abs/10.1556/084.2015.16.1.2.

Sun, Sanjun and Gregory M. Shreve (2014), Measuring translation difficulty: An empirical study, *Target* **26** (1), pp. 98–127. https://benjamins.com/online/target/articles/target.26.1.04sun.

Tezcan, Arda, Veronique Hoste, and Lieve Macken (2019), Estimating word-level quality of statistical machine translation output using monolingual information alone, *Natural Language Engineering*. http://dx.doi.org/10.1017/S1351324919000111.

Tezcan, Arda, Véronique Hoste, and Lieve Macken (2017), A neural network architecture for detecting grammatical errors in statistical machine translation, *The Prague bulletin of mathematical linguistics* **108** (1), pp. 133–145.

Toral, Antonio and Víctor M. Sánchez-Cartagena (2017), A multifaceted evaluation of neural versus phrase-based machine translation for 9 language directions, *Proceedings of the 15th conference of the European chapter of the association for computational linguistics*, Vol. 1, Association for computational linguistics, Valencia, Spain, pp. 1063–1073. http://aclweb.org/anthology/E17-1100.

Turian, Joseph, Lev Ratinov, and Yoshua Bengio (2010), Word representations: A simple and general method for semi-supervised learning, *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, Association for Computational Linguistics, pp. 384–394.

Van Brussel, Laura, Arda Tezcan, and Lieve Macken (2018), A fine-grained error analysis of NMT, PBMT and RBMT output for English-to-Dutch, *in* Calzolari, Nicoletta, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Koiti Hasida, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, Stelios Piperidis, and Takenobu Tokunaga, editors, *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*, European Language Resources Association (ELRA), pp. 3799–3804.

Vanroy, Bram, Orphée De Clercq, and Lieve Macken (2019), Correlating process and product data to get an insight into translation difficulty, *Perspectives* pp. 1–18. https://www.tandfonline.com/doi/full/10.1080/0907676X.2019.1594319.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017), Attention is all you need, *Advances in neural information processing systems*, pp. 5998–6008.

Xia, Fei and Michael McCord (2004), Improving a statistical MT system with automatically learned rewrite patterns, *Proceedings of the conference on computational linguistics*, Bombay, India, pp. 508–514.

Yamada, Kenji and Kevin Knight (2001), A syntax-based statistical translation model, *Proceedings of the 39th annual meeting on association for computational linguistics*, Association for computational linguistics, Toulouse, France, pp. 523–530.

Zhang, Jinchao, Mingxuan Wang, Qun Liu, and Jie Zhou (2017), Incorporating word reordering knowledge into attention-based neural machine translation, *Proceedings of the 55th annual meeting of the association for computational linguistics*, Vol. 1, Association for Computational Linguistics, Vancouver, Canada, pp. 1524–1534. http://aclweb.org/anthology/P17-1140.