Learning Paraphrastic Sentence Embeddings from Back-Translated Bitext

John Wieting[1] **Jonathan Mallinson**[2] **Kevin Gimpel**[1]
[1]Toyota Technological Institute at Chicago, Chicago, IL, 60637, USA
[2]School of Informatics, University of Edinburgh, Edinburgh, UK
{jwieting,kgimpel}@ttic.edu, j.mallinson@ed.ac.uk

Abstract

We consider the problem of learning general-purpose, paraphrastic sentence embeddings in the setting of Wieting et al. (2016b). We use neural machine translation to generate sentential paraphrases via back-translation of bilingual sentence pairs. We evaluate the paraphrase pairs by their ability to serve as training data for learning paraphrastic sentence embeddings. We find that the data quality is stronger than prior work based on bitext and on par with manually-written English paraphrase pairs, with the advantage that our approach can scale up to generate large training sets for many languages and domains. We experiment with several language pairs and data sources, and develop a variety of data filtering techniques. In the process, we explore how neural machine translation output differs from human-written sentences, finding clear differences in length, the amount of repetition, and the use of rare words.[1]

1 Introduction

Pretrained word embeddings have received a great deal of attention from the research community, but there is much less work on developing pretrained embeddings for *sentences*. Here we target sentence embeddings that are "paraphrastic" in the sense that two sentences with similar meanings are close in the embedding space. Wieting et al. (2016b) developed paraphrastic sentence embeddings that are useful for semantic textual similarity tasks and can also be used as initialization for supervised semantic tasks.

[1]Generated paraphrases and code are available at http://ttic.uchicago.edu/~wieting.

R:	We understand that has already commenced, but there is a long way to go.
T:	This situation has already commenced, but much still needs to be done.
R:	The restaurant is closed on Sundays. No breakfast is available on Sunday mornings.
T:	The restaurant stays closed Sundays so no breakfast is served these days.
R:	Improved central bank policy is another huge factor.
T:	Another crucial factor is the improved policy of the central banks.

Table 1: Illustrative examples of references (R) paired with back-translations (T).

To learn their sentence embeddings, Wieting et al. used the Paraphrase Database (PPDB) (Ganitkevitch et al., 2013). PPDB contains a large set of paraphrastic textual fragments extracted automatically from bilingual text ("bitext"), which is readily available for languages and domains. Versions of PPDB have been released for several languages (Ganitkevitch and Callison-Burch, 2014).

However, more recent work has shown that the fragmental nature of PPDB's pairs can be problematic, especially for recurrent networks (Wieting and Gimpel, 2017). Better performance can be achieved with a smaller set of sentence pairs derived from aligning Simple English and standard English Wikipedia (Coster and Kauchak, 2011). While effective, this type of data is inherently limited in size and scope, and not available for languages other than English.

PPDB is appealing in that it only requires bitext. We would like to retain this property but develop a data resource with sentence pairs rather than phrase pairs. We turn to neural machine translation (NMT) (Sutskever et al., 2014; Bahdanau et al., 2014; Sennrich et al., 2016a), which has matured recently to yield strong performance especially in terms of producing grammatical outputs.

In this paper, we build NMT systems for three language pairs, then use them to back-translate the non-English side of the training bitext. The resulting data consists of sentence pairs containing an English reference and the output of an X-to-English NMT system. Table 1 shows examples. We use this data for training paraphrastic sentence embeddings, yielding results that are much stronger than when using PPDB and competitive with the Simple English Wikipedia data.

Since bitext is abundant and available for many language pairs and domains,[2] we also develop several methods of filtering the data, including based on sentence length, quality measures, and measures of difference between the reference and its back-translation. We find length to be an effective filtering method, showing that very short length ranges—where the translation is 1 to 10 words—are best for learning.

In studying quality measures for filtering, we train a classifier to predict if a sentence is a reference or a back-translation, then score sentences by the classifier score. This investigation allows us to examine the kinds of phenomena that best distinguish NMT output from references in this controlled setting of translating the bitext training data. NMT output has more repetitions of both words and longer n-grams, and uses fewer rare words than the references.

We release our generated sentence pairs to the research community with the hope that the data can inspire others to develop additional filtering methods, to experiment with richer architectures for sentence embeddings, and to further analyze the differences between neural machine translations and references.

2 Related Work

We describe related work in learning general-purpose sentence embeddings, work in automatically generating or discovering paraphrases, and finally prior work in leveraging neural machine translation for embedding learning.

Paraphrastic sentence embeddings. Our learning and evaluation setting is the same as that considered by Wieting et al. (2016b) and Wieting et al. (2016a), in which the goal is to learn paraphrastic sentence embeddings that can be used for downstream tasks. They trained models on PPDB and evaluated them using a suite of semantic textual similarity (STS) tasks and supervised semantic tasks. Others have begun to consider this setting as well (Arora et al., 2017).

Other work in learning general purpose sentence embeddings has used autoencoders (Socher et al., 2011; Hill et al., 2016), encoder-decoder architectures (Kiros et al., 2015), or other learning frameworks (Le and Mikolov, 2014; Pham et al., 2015). Wieting et al. (2016b) and Hill et al. (2016) provide many empirical comparisons to this prior work. For conciseness, we compare only to the strongest configurations from their results.

Paraphrase generation and discovery. There is a rich history of research in generating or finding naturally-occurring sentential paraphrases (Barzilay and McKeown, 2001; Dolan et al., 2004; Dolan and Brockett, 2005; Quirk et al., 2004; Zhao et al., 2010; Coster and Kauchak, 2011; Xu et al., 2014, 2015).

The most relevant work uses bilingual corpora, e.g., Zhao et al. (2008) and Bannard and Callison-Burch (2005), the latter leading to PPDB. Our goals are highly similar to those of the PPDB project, which has also been produced for many languages (Ganitkevitch and Callison-Burch, 2014) since it only relies on the availability of bilingual text.

Prior work has shown that PPDB can be used for learning embeddings for words and phrases (Faruqui et al., 2015; Wieting et al., 2015). However, when learning sentence embeddings, Wieting and Gimpel (2017) showed that PPDB is not as effective as sentential paraphrases, especially for recurrent networks. These results are intuitive because the phrases in PPDB are short and often cut across constituent boundaries. For sentential paraphrases, Wieting and Gimpel (2017) used a dataset developed for text simplification by Coster and Kauchak (2011). It was created by aligning sentences from Simple English and standard English Wikipedia. We compare our data to both PPDB and this Wikipedia dataset.

Neural machine translation for paraphrastic embedding learning. Sutskever et al. (2014) trained NMT systems and visualized part of the space of the source language encoder for their English$\rightarrow$French system. Hill et al. (2016) evaluated the encoders of English-to-X NMT systems as sentence representations, finding them to

[2]For example, CzEng 1.6 (Bojar et al., 2016) contains a billion words across its 8 domains.

perform poorly compared to several other methods based on unlabeled data. Mallinson et al. (2017) adapted trained NMT models to produce sentence similarity scores in semantic evaluations. They used pairs of NMT systems, one to translate an English sentence into multiple foreign translations and the other to then translate back to English. Other work has used neural MT architectures and training settings to obtain better word embeddings (Hill et al., 2014a,b).

Our approach differs in that we only use the NMT system to generate training data for training sentence embeddings, rather than use it as the source of the model. This permits us to decouple decisions made in designing the NMT architecture from decisions about which models we will use for learning sentence embeddings. Thus we can benefit from orthogonal work in designing neural architectures to embed sentences.

3 Neural Machine Translation

We now describe the NMT systems we use for generating data for learning sentence embeddings. In our experiments, we use three encoder-decoder NMT models: Czech→English, French→English, and German→English.

We used Groundhog[3] as the implementation of the NMT systems for all experiments. We generally followed the settings and training procedure from previous work (Bahdanau et al., 2014; Sennrich et al., 2016a). As such, all networks have a hidden layer size of 1000 and an embedding layer size of 620. During training, we used Adadelta (Zeiler, 2012), a minibatch size of 80, and the training set was reshuffled between epochs. We trained a network for approximately 7 days on a single GPU (TITAN X), then the embedding layer was fixed and training continued, as suggested by Jean et al. (2015), for 12 hours. Additionally, the softmax was calculated over a filtered list of candidate translations. Following Jean et al. (2015), during decoding, we restrict the softmax layers' output vocabulary to include: the 10000 most common words, the top 25 unigram translations, and the gold translations' unigrams.

All systems were trained on the available training data from the WMT15 shared translation task (15.7 million, 39.2 million, and 4.2 million sentence pairs for CS→EN, FR→EN, and DE→EN, respectively). The training data included: Europarl v7 (Koehn, 2005), the Common Crawl corpus, the UN corpus (Eisele and Chen, 2010), News Commentary v10, the 10^9 French-English corpus, and CzEng 1.0 (Bojar et al., 2016). A breakdown of the sizes of these corpora can be found in Table 3. The data was pre-processed using standard pre-processing scripts found in Moses (Koehn et al., 2007). Rare words were split into sub-word units, following Sennrich et al. (2016b). BLEU scores on the WMT2015 test set for each NMT system can be seen in Table 3.

	Czech	French	German
Europarl	650,000	2,000,000	2,000,000
Common Crawl	160,000	3,000,000	2,000,000
News Commentary	150,000	200,000	200,000
UN	-	12,000,000	-
10^9 French-English	-	22,000,000	-
CzEng	14,700,000	-	-

Table 2: Dataset sizes (numbers of sentence pairs) for data domains used for training NMT systems.

Language	% BLEU
Czech→English	19.7
French→English	20.1
German→English	28.2

Table 3: BLEU scores on the WMT2015 test set.

To produce paraphrases we use "back-translation", i.e., we use our X→English NMT systems to translate the non-English sentence in each training sentence pair into English. We directly use the bitext on which the models were trained. This could potentially lead to pairs in which the reference and translation match exactly, if the model has learned to memorize the reference translations seen during training. However, in practice, since we have so much bitext to draw from, we can easily find data in which they do not match exactly.

Thus our generated data consists of pairs of English references from the bitext along with the NMT-produced English back-translations. We use beam search with a width of 50 to generate multiple translations for each non-English sentence, each of which is a candidate paraphrase for the English reference.

Example outputs of this process are in Table 1, showing some rich paraphrase phenomena in the data. These examples show non-trivial phrase substitutions ("there is a long way to go" and "much still needs to be done"), sentences being merged and simplified, and sentences being rearranged.

[3]Available at `https://github.com/sebastien-j/LV_groundhog`.

For examples of erroneous paraphrases that can be generated by this process, see Table 11.

4 Models and Training

Our goal is to compare our paraphrase dataset to other datasets by using each to train sentence embeddings, keeping the models and learning procedure fixed. So we select models and a loss function from prior work (Wieting et al., 2016b; Wieting and Gimpel, 2017).

4.1 Models

We wish to embed a word sequence s into a fixed-length vector. We denote the tth word in s as s_t, and we denote its word embedding by x_t. We focus on two models in this paper. The first model, which we call AVG, simply averages the embeddings x_t of all words in s. The only parameters learned in this model are those in the word embeddings themselves, which are stored in the word embedding matrix W_w. This model was found by Wieting et al. (2016b) to perform very strongly for semantic similarity tasks.

The second model, the GATED RECURRENT AVERAGING NETWORK (GRAN) (Wieting and Gimpel, 2017), combines the benefits of AVG and long short-term memory (LSTM) recurrent neural networks (Hochreiter and Schmidhuber, 1997). It first uses an LSTM to generate a hidden vector, h_t, for each word s_t in s. Then h_t is used to compute a gate that is elementwise-multiplied with x_t, resulting in a new hidden vector a_t for each step t:

$$a_t = x_t \odot \sigma(W_x x_t + W_h h_t + b) \qquad (1)$$

where W_x and W_h are parameter matrices, b is a parameter vector, and σ is the elementwise logistic sigmoid function. After all a_t have been generated for a sentence, they are averaged to produce the embedding for that sentence. The GRAN reduces to AVG if the output of the gate is always 1. This model includes as learnable parameters those of the LSTM, the word embeddings, and the additional parameters in Eq. (1). We use W_c to denote the "compositional" parameters, i.e., all parameters other than the word embeddings.

Our motivation for choosing these two models is that they both work well in this transfer learning setting (Wieting et al., 2016b) and they are architecturally similar with one crucial difference: only the GRAN takes into account word order. This difference plays an important role in the effectiveness of the different filtering methods as explored in Section 5.

4.2 Training

We follow the training procedure of Wieting et al. (2015) and Wieting et al. (2016b). The training data is a set S of paraphrastic pairs $\langle s_1, s_2 \rangle$ and we optimize a margin-based loss:

$$\min_{W_c, W_w} \frac{1}{|S|} \Bigg(\sum_{\langle s_1, s_2 \rangle \in S} \max(0, \delta - \cos(g(s_1), g(s_2)) + \cos(g(s_1), g(t_1))) + \max(0, \delta - \cos(g(s_1), g(s_2)) + \cos(g(s_2), g(t_2))) \Bigg) + \lambda_c \|W_c\|^2 + \lambda_w \|W_{w_{initial}} - W_w\|^2$$

where g is the model (AVG or GRAN), δ is the margin, λ_c and λ_w are regularization parameters, $W_{w_{initial}}$ is the initial word embedding matrix, and t_1 and t_2 are "negative examples" taken from a mini-batch during optimization. The intuition is that we want the two texts to be more similar to each other $(\cos(g(s_1), g(s_2)))$ than either is to their respective negative examples t_1 and t_2, by a margin of at least δ. To select t_1 and t_2, we choose the most similar sentence in some set (other than those in the given pair). For simplicity we use the mini-batch for this set, i.e., we choose t_1 for a given $\langle s_1, s_2 \rangle$ as follows:

$$t_1 = \operatorname*{argmax}_{t : \langle t, \cdot \rangle \in S_b \setminus \{\langle s_1, s_2 \rangle\}} \cos(g(s_1), g(t))$$

where $S_b \subseteq S$ is the current mini-batch. That is, we want to choose a negative example t_i that is similar to s_i according to the current model. The downside is that we may occasionally choose a phrase t_i that is actually a true paraphrase of s_i.

5 Experiments

We now investigate how best to use our generated paraphrase data for training universal paraphrastic sentence embeddings. We consider 10 data sources: Common Crawl (CC), Europarl (EP), and News Commentary (News) from all 3 language pairs, as well as the 10^9 French-English data (Giga). We extract 150,000 reference/back-translation pairs from each data source. We use 100,000 of these to mine for training data for our sentence embedding models, and the remaining 50,000 are used as train/validation/test data for the reference classification and language models described below.

5.1 Evaluation

We evaluate the quality of a paraphrase dataset by using the experimental setting of Wieting et al. (2016b). We use the paraphrases as training data to create paraphrastic sentence embeddings, using the cosine of the embeddings as the measure of semantic relatedness, then evaluate the embeddings on the SemEval semantic textual similarity (STS) tasks from 2012 to 2015 (Agirre et al., 2012, 2013, 2014, 2015), the SemEval 2015 Twitter task (Xu et al., 2015), and the SemEval 2014 SICK Semantic Relatedness task (Marelli et al., 2014).

Given two sentences, the aim of the STS tasks is to predict their similarity on a 0-5 scale, where 0 indicates the sentences are on different topics and 5 indicates that they are completely equivalent. As our test set, we report the average Pearson's r over these 22 sentence similarity tasks.[4] As development data, we use the 2016 STS tasks (Agirre et al., 2016), where the tuning criterion is the average Pearson's r over its 5 datasets.

5.2 Experimental Setup

For fair comparison among different datasets and dataset filtering methods described below, we use only 24,000 training examples for nearly all experiments. Different filtering methods produce different amounts of training data, and using 24,000 examples allows us to keep the amount of training data constant across filtering methods. It also allows us to complete these several thousand experiments in a reasonable amount of time. In Section 5.8 below, we discuss experiments that scale up to larger amounts of training data.

We use PARAGRAM-SL999 embeddings (Wieting et al., 2015) to initialize the word embedding matrix (W_w) for both models. For all experiments, we fix the mini-batch size to 100, λ_w to 0, λ_c to 0, and the margin δ to 0.4. We train AVG for 20 epochs, and the GRAN for 3, since it converges much faster. For optimization we use Adam (Kingma and Ba, 2014) with a learning rate of 0.001.

We compare to two data resources used in previous work to learn paraphrastic sentence embeddings. The first is phrase pairs from PPDB, used by Wieting et al. (2016b) and Wieting et al. (2016a). PPDB comes in different sizes (S, M, L, XL, XXL, and XXXL), where each larger size subsumes all smaller ones. The pairs in PPDB are sorted by a confidence measure and so the smaller sets contain higher precision paraphrases. We use PPDB XL in this paper, which consists of fairly high precision paraphrases. The other data source is the aligned Simple English / standard English Wikipedia data developed by Coster and Kauchak (2011) and used for learning paraphrastic sentence embeddings by Wieting and Gimpel (2017). We refer to this data source as "SimpWiki". We refer to our back-translated data as "NMT".

[4] Statistical significance testing is nontrivial due to averaging Pearson's r so we leave it to future work.

Lang.	Data	GRAN	AVG
SimpWiki		67.2	65.8
PPDB		64.5	65.8
CS	CC	65.5	65.4
	EP	66.5	65.1
	News	67.2	65.1
FR	CC	67.3	66.1
	EP	**67.8**	65.7
	Giga	67.4	65.9
	News	67.0	65.2
DE	CC	66.5	**66.2**
	EP	67.2	65.6
	News	66.5	64.7

Table 4: Test results (average Pearson's $r \times 100$ over 22 STS datasets) using a random selection of 24,000 examples from each data source.

5.3 Dataset Comparison

We first compare datasets, randomly sampling 24,000 sentence pairs from each of PPDB, SimpWiki, and each of our NMT datasets. The only hyperparameter to tune for this experiment is the stopping epoch, which we tune based on our development set. The results are shown in Table 4.

We find that the NMT datasets are all effective as training data, outperforming PPDB in all cases when using the GRAN. There are exceptions when using AVG, for which PPDB is quite strong. This is sensible because AVG is not sensitive to word order, so the fragments in PPDB do not cause problems. However, when using the GRAN, which is sensitive to word order, the NMT data is consistently better than PPDB. It often exceeds the performance of training on the SimpWiki data, which consists entirely of human-written sentences.

5.4 Filtering Methods

Above we showed that the NMT data is better than PPDB when using a GRAN and often as good as SimpWiki. Since we have access to so much more NMT data than SimpWiki (which is limited to fewer than 200k sentence pairs), we next experiment with several approaches for filtering the NMT data. We first consider filtering based on length, described in Section 5.5. We then consider filtering based on several quality measures designed to find more natural and higher-quality translations, described in Section 5.6. Finally, we consider several measures of diversity. By diversity we mean here a measure of the lexical and syntactic difference between the reference and its paraphrase. We describe these experiments in Section 5.7. We note that these filtering methods are not all mutually exclusive and could be combined, though in this paper we experiment with each individually and leave combination to future work.

5.5 Length Filtering

We first consider filtering candidate sentence pairs by length, i.e., the number of tokens in the translation. The tunable parameters are the upper and lower bounds of the translation lengths.

We experiment with a partition of length ranges, showing the results in Table 5. These results are averages across all language pairs and data sources of training data for each length range shown. We find it best to select NMT data where the translations have between 0 and 10 tokens, with performance dropping as sentence length increases. This is true for both the GRAN and AVG models. We do the same filtering for the SimpWiki data, though the trend is not nearly as strong. Therefore this is unlikely due to the nature of the evaluation data, and may be due to machine translation quality dropping as sentence length increases. This trend appears even though the datasets with higher ranges have more tokens of training data, since only the number of training sentence pairs is kept constant across configurations.

We then tune the length range using our development data, considering the following length ranges: [0,10], [0,15], [0,20], [0,30], [0,100], [10,20], [10,30], [10,100], [15,25], [15,30], [15,100], [20,30], [20,100], [30,100]. We tune over ranges as well as language, data source, and stopping epoch, each time training on 24,000 sentence pairs. We report the average test results over all languages and datasets in Table 6. We compare to a baseline that draws a random set of data, showing that length-based filtering leads to gains of nearly half a point on average across our test sets.

		Length Range			
Data	Model	0-10	10-20	20-30	30-100
SimpWiki	GRAN	67.4	67.7	67.1	67.3
	AVG	65.9	65.7	65.6	65.9
NMT	GRAN	66.6	66.5	66.0	64.8
	AVG	65.7	65.6	65.3	65.0

Table 5: Test correlations for our models when trained on sentences with particular length ranges (averaged over languages and data sources for the NMT rows). Results are on STS datasets (Pearson's $r \times 100$).

	NMT		SimpWiki	
Filtering Method	GRAN	AVG	GRAN	AVG
None (Random)	66.9	65.5	67.2	65.8
Length	67.3	66.0	67.4	66.2
Tuned Len. Range	[0,10]	[0,10]	[0,10]	[0,15]

Table 6: Length filtering test results after tuning length ranges on development data (averaged over languages and data sources for the NMT rows). Results are on STS datasets (Pearson's $r \times 100$).

The tuned length ranges are short for both NMT and SimpWiki. The distribution of lengths in the NMT and SimpWiki data is fairly similar. The 10 NMT datasets all have mean translation lengths between 22 and 28 tokens. The data has fairly large standard deviations (11-25 tokens) indicating that there are some very long translations in the data. SimpWiki has a mean length of 24.2 and a standard deviation of 13.1.

5.6 Quality Filtering

We also consider filtering based on several measures of the "quality" of the back-translation:

- **Translation Cost**: We use the cost (negative log likelihood) of the translation from the NMT system, divided by the number of tokens in the translation.
- **Language Model**: We train a separate language model for each language/data pair on 40,000 references that are separate from the 100,000 used for mining data. Due to the small data size, we train a 3-gram language model and use the KenLM toolkit (Heafield, 2011).
- **Reference/Translation Classification**: We train binary classifiers to predict whether a given sentence is a reference or translation (described in Section 5.6.1). We use the probability of being a reference as the score for filtering.

For translation cost, we tune the upper bound of the cost over the range $[0.2, 1]$ using increments

Filtering Method	GRAN	AVG
None (Random)	66.9	65.5
Translation Cost	66.6	65.4
Language Model	66.7	65.5
Reference Classification	67.0	65.5

Table 7: Quality filtering test results after tuning quality hyperparameters on development data (averaged over languages and data sources for the NMT rows). Results are on STS datasets (Pearson's $r \times 100$).

of 0.1. For the language model, we tune an upper bound on the perplexity of the translations among the set $\{25, 50, 75, 100, 150, 200, \infty\}$. For the classifier, we tune the minimum probability of being a reference over the range $[0, 0.9]$ using increments of 0.1.

Table 7 shows average test results over all languages and datasets after tuning hyperparameters on our development data for each. The translation cost and language model are not helpful for filtering, as random selection outperforms them. Both methods are outperformed by the reference classifier, which slightly outperforms random selection when using the stronger GRAN model. We now discuss further how we trained the reference classifier and the data characteristics that it reveals. We did not experiment with quality filtering for SimpWiki since it is human-written text.

5.6.1 Reference/Translation Classification

We experiment with predicting whether a given sentence is a reference or a back-translation, hypothesizing that generated sentences with high probabilities of being references are of higher quality. We train two kinds of binary classifiers, one using an LSTM and the other using word averaging, followed by a softmax layer. We select 40,000 reference/translation pairs for training and 5,000 for each of validation and testing. A single example is a sentence with label 1 if it is a reference translation and 0 if it is a translation.

In training, we consider the entire k-best list as examples of translations, selecting one translation to be the 0-labeled example. We either do this randomly or we score each sentence in the k-best list using our model and select the one with the highest probability of being a reference as the 0-labeled example. We tune this choice as well as an L_2 regularizer on the word embeddings (tuned over $\{10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}, 0\}$). We use PARAGRAM-SL999 embeddings (Wieting et al., 2015) to initialize the word embeddings for both models. Models were trained by minimizing cross entropy for 10 epochs using Adam with learning rate 0.001. We performed this procedure separately for each of the 10 language/data pairs.

Model	Lang.	Data	Test Acc.	+ Acc.	- Acc.
LSTM	CS	CC	72.2	72.2	72.3
		EP	72.3	64.3	80.3
		News	79.7	73.2	86.3
	FR	CC	80.7	82.1	79.3
		EP	79.3	75.2	83.4
		Giga	**93.1**	**92.3**	93.8
		News	84.2	81.2	87.3
	DE	CC	79.3	71.7	86.9
		EP	85.1	78.0	92.2
		News	89.8	82.3	**97.4**
AVG	CS	CC	71.2	68.9	73.5
		EP	69.1	63.0	75.1
		News	77.6	71.7	83.6
	FR	CC	78.8	80.4	77.2
		EP	78.9	75.5	82.3
		Giga	**92.5**	**91.5**	93.4
		News	82.8	81.1	84.5
	DE	CC	77.3	70.4	84.1
		EP	82.7	73.4	91.9
		News	87.6	80.0	**95.3**

Table 8: Results of reference/translation classification (accuracy$\times$100). The highest score in each column is in boldface. Final two columns show accuracies of positive (reference) and negative classes, respectively.

The results are shown in Table 8. While performance varies greatly across data sources, the LSTM always outperforms the word averaging model. For our translation-reference classification, we note that our results can be further improved. We also trained models on 90,000 examples, essentially doubling the amount of data, and the results improved by about 2% absolute on each dataset on both the validation and testing data.

Analyzing Reference Classification. We inspected the output of our reference classifier and noted a few qualitative trends which we then verified empirically. First, neural MT systems tend to use a smaller vocabulary and exhibit more restricted use of phrases. They correspondingly tend to show more repetition in terms of both words and longer n-grams. This hypothesis can be verified empirically in several ways. We do so by calculating the entropy of the unigrams and trigrams for both the references and the translations from our 150,000 reference-translation pairs.[5] We also calculate the repetition percentage of unigrams and

[5]We randomly selected translations from the beam search.

Lang.	Data	Ent. (uni)	Ent. (tri)	Rep. (uni)	Rep. (tri)
CS	CC	0.50	1.13	-7.57%	-5.58%
	EP	0.14	0.31	-0.88%	-0.11%
	News	0.16	0.31	-0.96%	-0.16%
FR	CC	0.97	1.40	-8.50%	-7.53%
	EP	0.51	0.69	-1.85%	-0.58%
	Giga	0.97	1.21	-5.30%	-7.74%
	News	0.67	0.75	-2.98%	-0.85%
DE	CC	0.29	0.57	-1.09%	-0.73%
	EP	0.32	0.53	-0.14%	-0.11%
	News	0.40	0.37	-1.02%	-0.24%
All		0.46	0.74	-2.80%	-2.26%

Table 9: Differences in entropy and repetition of unigrams/trigrams in references and translations. Negative values indicate translations have a higher value, so references show consistently higher entropies and lower repetition rates.

trigrams in both the references and translations. This is defined as the percentage of words that are repetitions (i.e., have already appeared in the sentence). For unigrams, we only consider words consisting of at least 3 characters.

The results are shown in Table 9, in which we subtract the translation value from the reference value for each measure. The translated text has lower n-gram entropies and higher rates of repetition. This appears for all datasets, but is strongest for common crawl and French-English 10^9.

We also noticed that translations are less likely to use rare words, instead willing to use a larger sequence of short words to convey the same meaning. We found that translations were sometimes more vague and, unsurprisingly, were more likely to be ungrammatical.

We check whether our classifier is learning these patterns by computing the reference probabilities $P(\mathrm{R})$ of 100,000 randomly sampled translation-reference pairs from each dataset (the same used to train models). We then compute the correlation between our classification score and different metrics: the repetition rate of the sentence, the average inverse-document frequency (IDF) of the sentence,[6] and the translation length.

The results are shown in Table 10. Negative correlations with repetitions indicates that fewer repetitions lead to higher $P(\mathrm{R})$. A positive correlation with average IDF indicates that $P(\mathrm{R})$ rewards the use of rare words. Interestingly, negative correlation with length suggests that the classifier prefers more concise sentences.[7] We show examples of these phenomena in Table 11. The first two examples show the tendency of NMT to repeat words and phrases. The second two show how they tend to use sequences of common words ("put at risk") rather than rare words ("endangering").

Metric	Spearman's ρ
Unigram repetition rate	-35.1
Trigram repetition rate	-18.4
Average IDF	27.8
Length	-34.0

Table 10: Spearman's ρ between our reference classifier probability and various measures.

	Sentence	$P(\mathrm{R})$
R:	Room was comfortable and the staff at the front desk were very helpful.	1.0
T:	The staff were very nice and the room was very nice and the staff were very nice.	<0.01
R:	The enchantment of your wedding day, captured in images by Flore-Ael Surun.	0.98
T:	The wedding of the wedding, put into images by Flore-Ael A.	<0.01
R:	Mexico and Sweden are longstanding supporters of the CTBT.	1.0
T:	Mexico and Sweden have been supporters of CTBT for a long time now.	0.06
R:	We thought Mr Haider ' s Austria was endangering our freedom.	1.0
T:	We thought that our freedom was put at risk by Austria by Mr Haider.	0.09

Table 11: Illustrative examples of references (R) and back-translations (T), along with probabilities from the reference classifier. See text for details.

5.7 Diversity Filtering

We consider several filtering criteria based on measures that encourage particular amounts of disparity between the reference and its back-translation:

- **n-gram Overlap**: Our n-gram overlap measures are calculated by counting n-grams of a given order in both the reference and translation, then dividing the number of shared n-grams by the total number of n-grams in the reference or translation, whichever has fewer. We use three n-gram overlap scores ($n \in \{1, 2, 3\}$).
- **BLEU Score**: We use a smoothed sentence-level BLEU variant from Nakov et al. (2012) that uses smoothing for all n-gram lengths and also smooths the brevity penalty.

[6] Wikipedia was used to calculate the frequencies of the tokens. All tokens were lowercased.

[7] This is noteworthy because the average sentence length of translations and references is not significantly different.

	NMT		SimpWiki	
Filtering Method	GRAN	AVG	GRAN	AVG
Random	66.9	65.5	67.2	65.8
Unigram Overlap	66.6	66.1	67.8	67.4
Bigram Overlap	67.0	65.5	68.0	67.2
Trigram Overlap	66.9	65.4	67.8	66.6
BLEU Score	67.1	65.3	67.5	66.5

Table 12: Diversity filtering test results after tuning filtering hyperparameters on development data (averaged over languages and data sources for the NMT rows). Results are on STS datasets (Pearson's $r \times 100$).

Data	GRAN	AVG
PPDB	64.6	66.3
SimpWiki (100k/168k)	67.4	**67.7**
CC-CS (24k)	66.8	-
CC-CS (100k)	**68.5**	-
CC-DE (24k)	-	66.6
CC-DE (168k)	-	67.6

Table 13: Test results with more training data. More data helps both AVG and GRAN to match or surpass training on SimpWiki. Both comfortably surpass PPDB. The number of training examples used is in parentheses.

For both methods, the tunable hyperparameters are the upper and lower bounds for the above scores. We tune over the cross product of lower bounds $\{0, 0.1, 0.2, 0.3\}$ and upper bounds $\{0.6, 0.7, 0.8, 0.9, 1.0\}$. Our intuition is that the best data will have some amount of n-gram overlap, but not too much. Too much n-gram overlap will lead to pairs that are not useful for learning.

The results are shown in Table 12, for both models and for both NMT and SimpWiki. We find that the diversity filtering methods lead to consistent improvements when training on SimpWiki. We believe this is because many of the sentence pairs in SimpWiki are near-duplicates and these filtering methods favor data with more differences.

Diversity filtering can also help when selecting NMT data, though the differences are smaller. We do note that unigram overlap is the strongest filtering strategy for AVG. When looking at the threshold tuning, the best lower bounds are often 0 or 0.1 and the best upper bounds are typically 0.6-0.7, indicating that sentence pairs with a high degree of word overlap are not useful for training. We also find that the GRAN benefits more from filtering based on higher-order n-gram overlap than AVG.

5.8 Scaling Up

Unlike the SimpWiki data, which is naturally limited and only available for English, we can scale our approach. Since we use data on which the NMT systems were trained and perform back-translation, we can easily produce large training sets of paraphrastic sentence pairs for many languages and data domains, limited only by the availability of bitext.

To test this, we took the tuned filtering methods and language/data pairs (according to our development dataset only), and trained them on more data. These were CC-CS for GRAN and CC-DE for AVG. We also trained each model on the same number of sentence pairs from SimpWiki.[8] We also compare to PPDB XL, and since PPDB has fewer tokens per example, we use enough PPDB data so that it has at least as many tokens as the SimpWiki data used in the experiment.[9]

Table 13 shows clear improvements when using more training data, providing evidence that our approach can scale to larger datasets. The NMT data surpasses SimpWiki for the GRAN, while the SimpWiki and NMT data perform similarly for AVG. PPDB is outperformed by both data sources for both models. Even when we train on all 52M tokens in PPDB XXL, AVG only reaches 66.5.

6 Conclusion

We showed how back-translation can be used to generate effective training data for paraphrastic sentence embeddings. We explored filtering strategies that improve the generated data; in doing so, we identified characteristics that distinguish NMT output from references. Our hope is that these results can enable learning paraphrastic sentence embeddings with powerful neural architectures across many languages and domains.

Acknowledgments

This research used resources of the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC02-06CH11357. We thank the developers of Theano (Theano Development Team, 2016) and NVIDIA Corporation for donating GPUs used in this research.

[8]Since the CC-CS data was the smallest dataset used to train the CS NMT system (See Table 3), we only used 100,000 pairs for the GRAN experiment. For AVG, we used the full 167,689.

[9]800,011 pairs for GRAN and 1,341,188 for AVG.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Inigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, German Rigau, Larraitz Uria, and Janyce Wiebe. 2015. SemEval-2015 task 2: Semantic textual similarity, English, Spanish and pilot on interpretability. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. SemEval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*.

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2016. SemEval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. *Proceedings of SemEval*, pages 497–511.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *SEM 2013 shared task: Semantic textual similarity. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*.

Eneko Agirre, Mona Diab, Daniel Cer, and Aitor Gonzalez-Agirre. 2012. SemEval-2012 task 6: A pilot on semantic textual similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *Proceedings of the International Conference on Learning Representations*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Colin Bannard and Chris Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*.

Regina Barzilay and Kathleen R McKeown. 2001. Extracting paraphrases from a parallel corpus. In *Proceedings of the 39th annual meeting on Association for Computational Linguistics*, pages 50–57.

Ondřej Bojar, Ondřej Dušek, Tom Kocmi, Jindřich Libovický, Michal Novák, Martin Popel, Roman Sudarikov, and Dušan Variš. 2016. CzEng 1.6: Enlarged Czech-English Parallel Corpus with Processing Tools Dockered. In *Proceedings of 19th International Conference on Text, Speech, and Dialogue (TSD)*, pages 231–238.

William Coster and David Kauchak. 2011. Simple english wikipedia: a new text simplification task. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 665–669.

Bill Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the 20th international conference on Computational Linguistics*, page 350.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proc. of IWP*.

Andreas Eisele and Yu Chen. 2010. MultiUN: A multilingual corpus from united nation documents. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*.

Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. Retrofitting word vectors to semantic lexicons. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Juri Ganitkevitch and Chris Callison-Burch. 2014. The multilingual paraphrase database. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The Paraphrase Database. In *Proceedings of HLT-NAACL*.

Kenneth Heafield. 2011. KenLM: faster and smaller language model queries. In *Proceedings of the EMNLP 2011 Sixth Workshop on Statistical Machine Translation*, pages 187–197.

Felix Hill, Kyunghyun Cho, Sebastien Jean, Coline Devin, and Yoshua Bengio. 2014a. Embedding word similarity with neural machine translation. *arXiv preprint arXiv:1412.6448*.

Felix Hill, KyungHyun Cho, Sebastien Jean, Coline Devin, and Yoshua Bengio. 2014b. Not all neural embeddings are born equal. *arXiv preprint arXiv:1410.0718*.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning distributed representations of sentences from unlabelled data. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8).

Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. On using very large target vocabulary for neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1–10.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ryan Kiros, Yukun Zhu, Ruslan R. Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in Neural Information Processing Systems*, pages 3294–3302.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of the 10th Machine Translation Summit*, pages 79–86.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180.

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. *arXiv preprint arXiv:1405.4053*.

Jonathan Mallinson, Rico Sennrich, and Mirella Lapata. 2017. Paraphrasing revisited with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 881–893.

Marco Marelli, Luisa Bentivogli, Marco Baroni, Raffaella Bernardi, Stefano Menini, and Roberto Zamparelli. 2014. SemEval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*.

Preslav Nakov, Francisco Guzman, and Stephan Vogel. 2012. Optimizing for sentence-level BLEU+1 yields short translations. In *Proceedings of COLING 2012*, pages 1979–1994.

Nghia The Pham, Germán Kruszewski, Angeliki Lazaridou, and Marco Baroni. 2015. Jointly optimizing word representations for lexical and sentential tasks with the c-phrase model. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*.

Chris Quirk, Chris Brockett, and William Dolan. 2004. Monolingual machine translation for paraphrase generation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Richard Socher, Eric H. Huang, Jeffrey Pennington, Andrew Y. Ng, and Christopher D. Manning. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in Neural Information Processing Systems*.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112.

Theano Development Team. 2016. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016a. Charagram: Embedding words and sentences via character n-grams. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016b. Towards universal paraphrastic sentence embeddings. In *Proceedings of International Conference on Learning Representations*.

John Wieting, Mohit Bansal, Kevin Gimpel, Karen Livescu, and Dan Roth. 2015. From paraphrase database to compositional paraphrase model and back. *Transactions of the ACL (TACL)*.

John Wieting and Kevin Gimpel. 2017. Revisiting recurrent networks for paraphrastic sentence embeddings. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Wei Xu, Chris Callison-Burch, and William B Dolan. 2015. SemEval-2015 task 1: Paraphrase and semantic similarity in Twitter (PIT). In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval)*.

Wei Xu, Alan Ritter, Chris Callison-Burch, William B. Dolan, and Yangfeng Ji. 2014. Extracting lexically divergent paraphrases from Twitter. *Transactions of the Association for Computational Linguistics*, 2:435–448.

Matthew D. Zeiler. 2012. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701.

Shiqi Zhao, Haifeng Wang, Xiang Lan, and Ting Liu. 2010. Leveraging multiple MT engines for paraphrase generation. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1326–1334.

Shiqi Zhao, Haifeng Wang, Ting Liu, and Sheng Li. 2008. Pivot approach for extracting paraphrase patterns from bilingual corpora. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 780–788.