The Speechmatics Parallel Corpus Filtering System for WMT18

Tom Ash, Remi Francis, Will Williams
Speechmatics, Cambridge, United Kingdom
{toma,remi,willw}@speechmatics.com

Abstract

Our entry to the parallel corpus filtering task uses a two-step strategy. The first step uses a series of pragmatic hard 'rules' to remove the worst example sentences. This first step reduces the effective corpus size down from the initial 1 billion to 160 million tokens. The second step uses four different heuristics weighted to produce a score that is then used for further filtering down to 100 or 10 million tokens. Our final system produces competitive results without requiring excessive fine tuning to the exact task or language pair. The first step in isolation provides a very fast filter that gives most of the gains of the final system.

1 Introduction

This task asks for applicants to provide a score for each sentence pair in a 1-billion-word Machine Translation (MT) training corpus that is considered to be 'very noisy', such that those scores can be used to filter the corpus down into 10 million and 100 million words subsets. The quality of the output is measured by BLEU score obtained by training standard systems on these two subsets of data.

We consider this task to comprise of two primary components, namely (a) removing sentences that do not represent good examples of translation from one language to the other ('junk') and (b) distilling the remaining data down to a smaller training footprint without losing quality or diversity and then attaching scores to those sentences.

These two components are somewhat related; however, we chose to use a two-pass system to tackle them independently, so our system could be used to tackle the two components separately if required by a 'real-world' use case.

There are various approaches to this task that have previously been reported and we have attempted to select the most pragmatically useful of these to incorporate into our final system. Our philosophy in choosing what to put into our system was to make it as general as possible, such that it could be used for other language pairs and different datasets, rather than specifically tuning for this task. That then allows us to use the system more widely across our efforts in the field of machine translation. We have also chosen to use an array of different metrics to produce a final score, rather than a single score, to gain the benefits of multiple models that approach the problem in different ways.

1.1 Dev Data

As well as the 1-billion-word corpus to be processed, a smaller corpus of paired English-German data is available as a development set. This data comprises the data for the WMT 2018 news translation task data for German-English without the Paracrawl parallel corpus. This data is approximately 130M words, drawn from Europarl, Common Crawl, News Commentary and Rapid EU Press Release Corpora. More details of this data are available from http://www.statmt.org/wmt18/translation-task.html.
This data is hereafter referred to as the 'dev data'.

2 System Description

Our filtering system consists of two passes. The first pass uses some hard 'rules' to eliminate the bulk of the data. We consider this data to be 'junk' and score each sentence thus removed with a zero.

The second pass uses several heuristics we have developed to assign scores greater than zero to

Proceedings of the Third Conference on Machine Translation (WMT), Volume 2: Shared Task Papers, pages 853–859

https://doi.org/10.18653/v1/W18-64099

each sentence pair, with the aim of distilling down the data into as rich a subset as possible.

2.1 Initial 'rules'

The following hard rules are performed sequentially on the corpus. If any sentence 'fails' a rule it is immediately given a score of 0 and not considered for any further portion of our scoring system.

Line Length: we follow the 'length-based filtering' of Khadivi and Ney (2005). This method attempts to catch instances of grossly mistranslated sentences using the assumption that sentences in different languages will consist of approximately the same number of words and removing sentence pairs that have widely varying lengths.

If I and J denote the source and target sentence length respectively, sentence pairs are eliminated unless all of the following are true:

$$\begin{aligned} &6 * I > J \text{ and } I < 6 * J \\ &I < 3 \text{ or } J < 3 \text{ or } (I < 2.2 * J \text{ and } J < 2.2 * I) \\ &I < 10 \text{ or } J < 10 \text{ or } (I < 2 * J \text{ and } J < 2 * I) \end{aligned} \quad (1)$$

We sampled these same thresholds on a range of other languages and were surprised to see they were reasonable without alteration even in quite diverse situations, such as agglutinative languages.

Non-translation: following Song et. al. (2014) we remove sentence pairs where the source and target have a BLEU similarity score greater than 0.6. This deals with cases of either untranslated or only partially translated sentences.

Language identification: Web crawled corpora typically contain many data that are not in the language it claims to be. To try and identify such cases we use *lang-id* (Lui and Baldwin, 2012) to identify the most likely language of both the source and target sentence and remove the entry if either source or target disagrees with the correct label.

We also tried a different version of this in which we used the language probabilities generated by langid alongside a threshold instead of a binary decision based on the langid 1-best. With appropriate tuning this gave marginal gains, but the processing time was increased more than we found acceptable so is not used in our target system.

For languages not supported by pre-trained language identification models, we intend to use FastText (Joulin et. al, 2017) to train our own.

We believe this is the part of our rules most likely to give false positives. It was not possible to quantify this, but from qualitative judgement of the output it appeared to often falsely misjudge something as being in an incorrect language, particularly short sentences. Nonetheless our experiments show the rule greatly improved overall quality of the final corpus, so we believe it provides a lot more good than harm.

Character filtering: we expect there to be unwanted characters in a noisy corpus – for example Denkowski et. al. (2012) filter out all lines with invalid Unicode, control characters and similar. We approach this in a systematic way, by defining a list of characters we deem acceptable for each language and only keeping sentences containing just those characters. We create our character lists by counting character occurrence in the 'dev data', sorting on character count and then quickly manually scanning through the most common characters to generate a final list of around 80 characters per language that we deem 'acceptable'.

Our system then eliminates any sentences that use any character not in these lists. This both reduces any remaining cases of data in an incorrect language and incorrectly parsed markup from the web crawlers. It also reduces the effective character set remaining in the training data, which in turn reduces the effective vocabulary size of resultant MT systems, which we found to be beneficial when training modern NMT systems.

Digit matching: numbers, in particular digits, can be used to mark well matched sentences, and indeed they have been used as such in paired corpus alignment (Khadivi and Ney, 2005, Simard et. al 1992). Our system captures this by extracting all digits (in this case the characters 0-9) from the source and target sentence and eliminating them if they differ at all. This does introduce a small number of false positives where one side has the number in digits and the other in words ('1' vs 'one'), but we qualitatively found occurrences of this to be small.

2.2 Scoring Heuristics

To rank the remaining words, we turned to four heuristics we developed and found to be correlated with quality of the data.

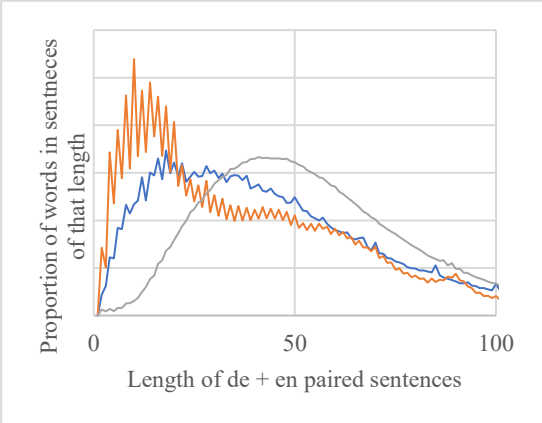

Figure 1: Plot of sentence length versus proportion of words that appear in sentences of that length, for the raw corpus (orange with leftmost peak), the corpus after our initial 'rules' (blue with central peak), and the dev data (grey with rightmost peak). With our sentence length heuristic we are trying to move the blue line to be closer to the grey one.

Each heuristic produced a score with a positive correlation to data quality (as measured by resultant BLEU), which we then scaled to be between 0 and 1. Our submissions were then based on weighted averages of those scores, where the weights between the different heuristics were determined empirically.

Sentence length: We noticed that the sentences in the corpus remaining after the rules were applied tended to be quite short. We confirmed this by comparing the sentence length distribution to that in the dev data (Figure 1). Note that our definition of sentence length here is the length of both source and target sentence summed, rather than length of one or the other.

These short sentences tended to be indicative of 'poor quality' and so we set up a heuristic to encourage longer sentences. In particular we use the following formula:

$$\begin{aligned} &if\ length \leq 40: \\ &\quad score = \frac{2 * \text{length}}{100} \\ &elif\ length \leq 80: \\ &\quad score = 0.8 * \frac{\text{length} - 40}{200} \\ &else: \\ &\quad score = 1.0 \end{aligned} \tag{2}$$

We chose to use this relatively simple algorithm rather than any more sophisticated fitting technique in order to keep the system as general as possible. Any system which attempts to fit the exact curve is reliant on a target corpus which goes against the spirit of the task. We do note that we would probably not choose to use this heuristic in isolation however, as it would then essentially be no more than selecting the longest sentences.

Perplexity: perplexity measures have been used to filter language modelling corpora with respect to a specific domain (Gao et al, 2002; Lin et al., 1997). We would expect the same techniques to be beneficial here too. However, in the task description we were specifically asked not to use metrics related to domain-relatedness. As with our sentence length heuristic we look to mirror the overall perplexity statistics of a 'clean' corpus instead.

Rather than compare to a specific domain we trained a 5-gram using *KenLM* (Heafield et al, 2013) on the data itself, measured log(perplexity) of each sentence using this self-trained model and then did the same on the dev data. As with the sentence length heuristic, we found that the dev data displayed a slightly different behavior to the corpus being filtered (Figure 2) – in this case the overall shape of the graph was similar, peaked at a value of 0.82 for negative log perplexity divided by sentence length, but the dev data had a sharper peak, and the corpus to be filtered had more sentences of higher or lower perplexity values.

Our heuristic therefore upweights sentences closer to this peak, to try and match the dev data behavior.

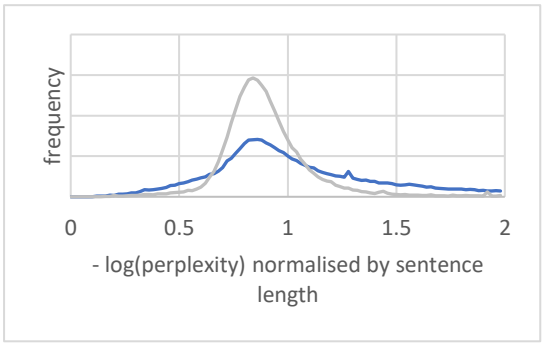

Figure 2: Plot showing frequency against negative log perplexity normalised by sentence length, for the corpus after rules were applied (blue, lower peak) and dev data (grey, higher peak). With our perplexity heuristic we are trying to move the blue line to be closer to the grey one.

$$if\ -\log(ppl)\ per\ word \leq 0.82:$$
$$score = 1 - \frac{(0.82 + \log(ppl)\ per\ word)}{0.82}$$
$$else:$$
$$score = \max(0,1 - \frac{-\log(ppl) per\ word - 0.82}{3}) \quad (3)$$

Diversity: following Song et. al (2014) we used sentence similarity in a rolling buffer to measure how diverse a sentence was compared to its neighbours.

Like Song et. al. we used a rolling window of 200 sentences, however we found that using BLEU to measure sentence similarity was too slow for practical use with such a large corpus. Instead we took a two-step approach, first checking if at least half of the words in the two sentences were in common. If so we then used simple edit distance to measure how similar the sentences were. The per sentence score derived from this heuristic was the minimum Levenshtein edit distance between a given sentence and all other sentences in its 200-sentence window.

To give this metric more chance of identifying similar sentences, we first sorted the entire corpus by sentence length, as sentences of similar length are more likely to have smaller edit distances.

This heuristic then effectively assigns high scores to sentences that exhibit distinctness to others in the corpus, whilst giving low scores to sentences that are near duplicates and hence adding little new information.

MT filtering: previous work has shown that machine translation systems themselves can directly be used to filter parallel corpora, either as a preprocessing step (Gaspers et. al. 2018) or even on the fly as part of the training process (Zhang et. al., 2017).

We therefore train an MT system on the entirety of the post-rules corpus. We then compute the one best translation for each sentence. Finally, we compute the decoder cost of both the one best translation and the reference translation. The decoder cost in this case is the cross-entropy loss. We did not normalize by sentence length as we found it made little difference.

The raw decoder cost of the reference translation by itself is an initially interesting metric, as low values correspond to sentences that are more likely to be correct translations as they don't diverge from what the system would expect to see. However, we also find that this approach biases the results towards short sentences that are very similar to one another, meaning resulting corpora lack diversity and fall foul of the rare words problem (Luong et. al 2015). The decoder cost of the 1-best translation is therefore used as a constraint on this. Our final score for this heuristic is the decoder cost of the reference sentence *minus* the decoder cost of the 1-best. We then compute this number in both translation directions and average.

High values of this derived score represent situations where the reference translation is judged much less likely than the 1-best by the decoder and thus should be discarded as likely junk. Very low scores show that the reference translation agrees with the model and are therefore unlikely to be junk. And further than that scores where the target has a lower cost than the target indicate explicit areas where the model needs to be improved – in other words exactly the sorts of inputs that are most valuable for the task of training a machine translation system.

We used the tensor2tensor framework to train a machine translation system for this scoring (Vaswani et al. 2018). The setup was the same as we used for benchmarking, as described in Section 3.

3 Benchmarking

To benchmark our progress, we use the tensor2tensor system (Vaswani et. al. 2018) which reports world leading results on machine translation tasks at present. We took the most recent commit of the code (at the time) from https://github.com/tensorflow/tensor2tensor/commit/99750c4b and used it without alteration.

We use this system without attempting to tune hyperparameters, except that we use the predefined 'transformer_small' recipe from the code repository (rather than the default 'transformer_base'), for speed and memory reasons. The 'transformer_small' recipe uses two hidden layers, each of size 256 and 4 attention heads. We trained each system for 500k steps (we found training for more steps was not helpful for performance) then averaged the last 8 checkpoints.

All BLEU scores reported used the described filtering system to prepare the training data, and then benchmark a trained transformer_small against the 'newstest2016' test set. BLEU was calculated using the t2t-bleu function in

	1bn word corpus	dev corpus
Line length	12.3%	6.0%
Non-translation	8.3%	0.6%
Language identification	12.0%	1.5%
Character filtering	24.9%	13.5%
Digit matching	26.5%	6.3%

Table 2: Percentages of the 1 billion word and dev corpora removed by each of the initial filtering

tensor2tensor and all reported numbers are on uncased text, with no tokenization applied.

4 Results

4.1 Initial 'rules'

Using the initial 'rules' removed 840 million words from the 1-billion-word corpus, leaving 160 million words for further scoring. Table 1 shows the contribution each rule made to this. Note that by contrast the rules would have removed a much smaller, but still significant, proportion of the dev data. This shows both that the rules are effective at removing 'bad' data (as we assume the 1 billion words contains more 'bad' data than the dev set) and that they are perhaps over aggressive and could bear some more tuning.

The rules were applied sequentially, so the latter rules may have removed more words if applied directly to the initial corpus.

Purely using these initial hard rules and then randomly selecting from the resulting 160M improves BLEU scores vastly compared to randomly selecting from the entire 1Bn word corpus (Table 2). For a target corpus size of 10M words the BLEU score improves from 5.93 to 26.14.

4.2 Scoring Heuristics

We applied the scoring heuristics described above in various combinations on the 160M words remaining after our initial 'rules'.

When filtering down to 100M words of data, any of the heuristics by themselves improved the BLEU score by between 0.12-1.67 as compared to randomly selecting from the 160M words post-'rules' (Table 2). Combining them in any combination gives further improvements and using all of them together gives a total of 1.97 gain in BLEU.

When filtering down to 10M words the picture is more complicated. Two of the heuristics by themselves produce worse BLEU scores (sentence length and MT scoring) and two improve the BLEU scores (perplexity and diversity). When combined equally there is a gain of 5.22, which is degraded if any of the metrics are omitted from that averaging. In particular the BLEU is degraded significantly if MT scoring is omitted from the combination.

We suspect that the very low scores exhibited in the 10M results are more than likely due to

Method of filtering data down to target amount	100M words	10M words
Randomly selected sentences from initial 1Bn	*	5.93
Randomly selected sentences from 160M after initial 'rules'	31.14	26.14
Sentence length scoring used to pick best from 160M after 'rules'	32.52	17.72
Perplexity scoring used…	32.81	29.00
Diversity scoring used…	31.80	28.46
MT scoring used …	32.26	17.07
All four heuristics except length used…	32.98	30.47
All four heuristics except perplexity used…	32.69	30.97
All four heuristics except diversity used…	32.71	30.34
All four heuristics except MT used…	32.83	17.86
All four scoring heuristics averaged and used…	33.11	31.36

Table 1: BLEU scores computed by training a tensor2tensor transformer_small system on 10M and 100M samples of data and then testing on newstest2016. The cell marked '*' could not be computed due to memory issues with our training setup. We list columns in terms of number of words in the corpus rather than the (perhaps more familiar) number of sentence pairs, as the task demanded we filter to a specific number of words rather than sentence pairs. The number of sentence pairs varied in each cell as different filtering techniques led to different average sentence lengths. The 10M corpora varied between 200k and 1M sentence pairs, for example, and the 100M corpora between 4M and 10M sentence pairs.

pathological failures in training the tensor2tensor system, however we were unable to ascertain the exact cause and found the numbers were reproduceable on multiple runs of training with the same setup and data.

4.3 Discussion

It is clear that using our initial 'rules' offer a significant improvement over random selection and that the scoring heuristics we have used are all capable of adding additional value in sub selecting data.

The sentence length scoring heuristic and the initial rules (barring language identification) are by some order of magnitude the fastest and simplest part of the system. For an initial look at data we would recommend using these before investing time into the more compute intensive rules.

Our entries to the competition were based on the balanced scoring across all four heuristics ('speechmatics-best-candidate-balanced-scoring.txt'), scoring purely based on the MT scoring ('speechmatics-purely-neural-scoring.txt') and a version with asymmetric weights heavily skewed towards the MT scoring ('speechmatics-prime-neural-scoring.txt').

4.4 Further Work

At present we have not tuned many parameters in our system. For optimal results we would spend more time on each of the 9 separate components we used for our system to optimize their various parameters with respect to final system BLEU.

For realistic use cases we would also expect that domain specific entropy filtering would be hugely beneficial, as we have previously found in language modelling (Williams et. al. 2015).

Conceptually we believe that the MT scoring heuristic has the most scope for future development. It is also the component most closely related to the actual task – translating text. Particularly interesting would be investigating its efficacy as the model capacity is scaled. Our belief is that some form of system that dynamically eliminates text as part of training could end up being the optimal approach to filtering out noisy parallel data.

5 Conclusion

The Speechmatics entry to the parallel corpus filtering task comprises a two-step system. The first step applies some simple rules to remove the bulk of the poor-quality data from a corpus. This gives most of the gains in terms of BLEU on a final trained system. We then apply four heuristics for scoring that give additional BLEU improvements.

We believe this is a relatively straightforward system that can be used across a wide variety of language pairs with little alteration to produce high quality reduced size MT corpora.

References

Michael Denkowski, Gred Hanneman, Alon Lavie. (2012). The CMU-Avenue French-English Translation System. *Proceedings of the NAACL 2012 Workshop on Statistical Machine Translation*, Montreal, Canada.

Jianfend Gao, Joshua Goodman, Mingjing Li and Kai-Fu Lee. (2002). Toward a unified approach to statistical language modeling for Chinese. *ACM Transactions on Asian Language Information Processing (TALIP)*. ACM, New York, USA

Judith Gaspers, Penny Karanasou, Rajen Chetterjee. (2018) Selecting Machine-Translated Data for Quick Bootstrapping of a Natural Language Understanding System. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*. New Orleans, USA

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan Clark and Philipp Koehn. (2013). Scalable Modified {Kneser-Ney} Language Model Estimation. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*. Sofia, Bulgaria.

Armand Joulin, Edouard Grave, Piotr Bojanowski and Tomas Mikolov. (2017). Bag of Tricks for Efficient Text Classification. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. Valencia, Spain

Shahram Khadivi and Hermann Ney. (2005). Automatic Filtering of Bilingual Corpora for Statistical Machine Translation. *In: Montoyo A., Muńoz R., Métais E. (eds) Natural Language Processing and Information Systems. NLDB 2005. Lecture Notes in Computer Science, vol 3513*. Springer, Berlin, Heidelberg

Sung-Chien Lin, Chi-Lung Tsai, Lee-Feng Chien, Keh-Jiann Chen, Lin-Shan Lee. (1997). Chinese language model adaptation based on document classification and multiple domain-specific language models. *Proceedings of the 5th European*

Conference on Speech Communication and Technology. Rhode, Greece.

Marco Lui and Timothy Baldwin. (2012). langid.py: An Off-the-shelf Language Identification Tool. *In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012), Demo Session.* Jeju, Republic of Korea

Thang Luong, Ilya Sutskever, Quoc V. Le, Oriol Vinyals and Wojciech Zaremba. (2015). Addressing the Rare Word Problem in Neural Machine Translation. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*. Beijing, China

Michel Simard, George Foster, Pierre Isabelle. (1992). Using cognates to align sentences in bilingual corpora. *Fourth Int. Conf. on Theoretical and Methodological Issues in Machine Translation (TMI92).* Montreal, Canada.

Xingyi Song, Trevor Cohn and Lucia Specia. (2014). Data Selection for Discriminative Training in Statistical Machine Translation. *17th Annual Conference of the European Association for Machine Translation.* Dubrovnik, Croatia

Ashish Vaswani, Samy Bengio, Eugene Brevdo, Francois Chollet, Aidan Gomez, Stephan Gouws, Llion Jones, Lukasz Kaiser, Nal Kalchbrenner, Niki Parmar, Ryan Sepassi, Noam Shazeer and Jakob Uszkoreit. (2018) Tensor2Tensor for Neural Machine Translation. *CoRR, abs/ 1803.07416*.

Will Williams, Niranjani Prasad, David Mrva, Tom Ash, Tony Robinson. (2015) Scaling recurrent neural network language models. *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* Brisbane, Australia.

Dakun Zhang, Jungi Kim, Josep Crego, Jean Snellart. (2017) Boosting Neural Machine Translation. *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Taipei, Taiwan.