The JHU Parallel Corpus Filtering Systems for WMT 2018

Huda Khayrallah **Hainan Xu** **Philipp Koehn**
Center for Language and Speech Processing
Johns Hopkins University
{huda, hxu31, phi}@jhu.edu

Abstract

This work describes our submission to the WMT18 Parallel Corpus Filtering shared task. We use a slightly modified version of the Zipporah Corpus Filtering toolkit (Xu and Koehn, 2017), which computes an adequacy score and a fluency score on a sentence pair, and use a weighted sum of the scores as the selection criteria. This work differs from Zipporah in that we experiment with using the noisy corpus to be filtered to compute the combination weights, and thus avoids generating synthetic data as in standard Zipporah.

1 Introduction

Todays machine translation systems require large amounts of training data in form of sentences paired with their translation, which are often compiled from online sources. This has not changed fundamentally with the move from statistical machine translation to neural machine translation, also we observed that neural models require more training data (Koehn and Knowles, 2017) and are more sensitive to noise (Khayrallah and Koehn, 2018). Thus both the acquisition of more training data such as indiscriminate web crawling and corpus filtering will have large impact on the quality of state-of-the-art machine translation systems.

The JHU submission to the WMT18 Parallel Corpus Filtering shared task uses a modified version of the Zipporah Corpus Filtering toolkit (Xu and Koehn, 2017). For a sentence pair, Zipporah uses a bag-of-words model to generate an adequacy score, and an n-gram language model to generate fluency score. The two scores are combined based on weights trained in order to separate clean data from noisy data. The original version of Zipporah generates artificial noisy training data to train such classifier, in this submission we also treat the Paracrawl corpus as the negative examples.

2 Related Work

Zipporah builds upon prior work in data cleaning and data selection.

For data selection, work has focused on selecting a subset of data based on domain-matching. Moore and Lewis (2010) computed cross-entropy between in-domain and out-of-domain language models to select data for training domain-relevant language models. XenC (Rousseau, 2013), an open-source tool, also selects data based on cross-entropy scores on language models. Axelrod et al. (2015) utilized part-of-speech tags and used a class-based n-gram language model for selecting in-domain data and Duh et al. (2013) used a neural network based language model trained on a small in-domain corpus to select from a larger mixed-domain data pool. Lü et al. (2007) redistributed different weights for sentence pairs/predefined sub-models. Shah and Specia (2014) described experiments on quality estimation which, given a source sentence, select the best translation among several options.

For data cleaning, work has focused on removing noisy data. Taghipour et al. (2011) proposed an outlier detection algorithm which leads to an improved translation quality when trimming a small portion of data. Cui et al. (2013) used a graph-based random walk algorithm to do bilingual data cleaning. BiTextor (Esplá-Gomis and Forcada, 2009) utilizes sentence alignment scores and source URL information to filter out bad URL pairs and selects good sentence pairs. Similar to this work, the qe-clean system (Denkowski et al., 2012; Dyer et al., 2010; Heafield, 2011) uses word alignments and language models to select sentence pairs that are likely to be good translations of one another.

We focus on data cleaning for all purposes, as opposed to data selection for a given domain. We

Proceedings of the Third Conference on Machine Translation (WMT), Volume 2: Shared Task Papers, pages 909–912

https://doi.org/10.18653/v1/W18-6480

aim to create a corpus of generally valid translations, which could then be filtered to adapt to a particular domain.

3 Zipporah

We use a slightly modified version of the Zipporah Corpus Filtering toolkit (Xu and Koehn, 2017). Zipporah works as follows: it first maps all sentence pairs into the proposed feature space, and then trains a simple logistic regression model to separate known good data and bad data. Once the model is trained, it is used to score sentence pairs in the noisy data pool.

Zipporah uses two features inspired by *adequacy* and *fluency*. The adequacy feature uses bag-of-words translation scores, and the fluency feature uses n-gram language model scores.

3.1 Adequacy Score

Zipporah generates probabilistic dictionaries from an aligned corpus, and uses them to generate bag of words translation scores for each sentence. This is done in both directions.

Given a sentence pair (s_f, s_e) in the noisy data pool, we represent the two sentence as two sparse word-frequency vectors v_f and v_e. For example for any French word w_f, we have $v_f[w_f] = \frac{c(w_f,s_f)}{l(s_f)}$, where $c(w_f, s_f)$ is the number of occurrences of w_f in s_f and $l(s_f)$ is the length of s_f. We do the same for v_e. Then we "translate" v_f into v'_e, based on the probabilistic f2e dictionary, where

$$v'_e[w_e] = \sum_{w_f} v_f[w_f] p(w_e|w_f)$$

For a French word w that does not appear in the dictionary, we keep it as it is in the translated vector, i.e. assume there is an entry of $(w, w, 1.0)$ in the dictionary. We compute the cross-entropy between v_e and v'_e,

$$\text{xent}(v_e, v'_e) = \sum_{w_e} v_e[w_e] \log \frac{1}{v'_e[w_e] + c} \quad (1)$$

where c is a smoothing constant to prevent the denominator from being zero, which we set $c = 0.0001$ for all experiments.

We perform similar procedures for English-to-French, and compute $\text{xent}(v_f, v'_f)$. We define the adequacy score as the sum of the two:

$$\text{adequacy}(s_f, s_e) = \text{xent}(v_e, v'_e) + \text{xent}(v_f, v'_f)$$

3.2 Fluency Score

Zipporah trains two 5-gram language models with a clean French and English corpus, and then for each sentence pair (s_g, s_e) scores each sentence with the corresponding model, $\mathcal{F}_{\text{ngram}}(s_g)$ and $\mathcal{F}_{\text{ngram}}(s_e)$, each computed as the ratio between the sentence negative log-likelihood and the sentence length. We define the fluency score as the sum of the two:

$$\text{fluency}(s_G, s_e) = \mathcal{F}_{\text{ngram}}(s_G) + \mathcal{F}_{\text{ngram}}(s_e)$$

3.3 Classifier

We train a binary classifier to separate a clean corpus from noisy corpora, based on the 2 features proposed. Higher orders of the features are used in order to achieve a non-linear decision boundary. We implement this using the logistic regression model from scikit-learn (Pedregosa et al., 2011), and use the features in the form of (x^8, y^8).

3.4 Training Data

We use clean WMT training data as the examples of clean text. The original version of Zipporah creates synthetic negative training examples by shuffling the clean data set, both at the corpus and sentence levels in order to generate inadequate and non-fluent text.

Since much of the raw Paracrawl data is noisy (Khayrallah and Koehn, 2018), we also train a version where we simply use the portion of Paracrawl released for the shared task as the negative examples to train our classifier, without generating synthetic noisy data. We experiment with using both the full portion of Paracrawl and a $10,000$ line subset.

4 Results

We include the results of running the three versions of Zipporah in Table 1. The final column is the average score across the 6 test sets.

- Zipporah-synthetic denotes the system with synthetic negative examples as in the original version of Zipporah.
- Zipporah-paracrawl denotes the system trained with the Paracrawl as the negative examples.
- Zipporah-paracrawl-10000 denotes the system trained with a 10000 sentence subset of Paracrawl.

Statistical machine translation (SMT) scores, 10 million words

	dev	test						
System Name	Newstest2017	Newstest2018	IWSLT	Acquis	EMEA	GV	KDE	**avg**
zipporah-synthetic	21.77	26.75	20.78	19.40	25.07	20.70	24.45	**22.85**
zipporah-paracrawl-10000	20.24	26.31	20.21	19.88	24.69	20.28	24.30	**22.61**
zipporah-paracrawl	20.18	26.26	20.36	19.33	24.76	20.37	24.32	**22.57**
best shared task submission	23.14	29.59	21.76	21.45	28.12	22.63	23.93	**24.58**

Statistical machine translation scores (SMT), 100 million words

	dev	test						
System Name	Newstest2017	Newstest2018	IWSLT	Acquis	EMEA	GV	KDE	**avg**
zipporah-synthetic	24.93	30.32	22.79	22.42	30.13	23.40	26.57	**25.94**
zipporah-paracrawl-10000	24.85	30.19	22.61	22.12	29.92	23.35	26.42	**25.77**
zipporah-paracrawl	24.81	30.35	22.63	22.13	30.12	23.26	26.52	**25.84**
best shared task submission	25.80	31.35	23.17	22.29	31.45	23.88	26.87	**26.50**

Neural machine translation scores (NMT), 10 million words

	dev	test						
System Name	Newstest2017	Newstest2018	IWSLT	Acquis	EMEA	GV	KDE	**avg**
zipporah-synthetic	26.13	32.22	23.89	22.73	26.95	24.26	24.94	**25.83**
zipporah-paracrawl-10000	25.21	31.44	23.13	22.82	26.31	24.02	24.32	**25.34**
zipporah-paracrawl	25.20	31.31	23.14	22.51	26.56	24.38	24.53	**25.41**
best shared task submission	28.49	35.67	25.10	23.69	32.72	26.72	27.81	**28.62**

Neural machine translation scores (NMT), 100 million words

	dev	test						
System Name	Newstest2017	Newstest2018	IWSLT	Acquis	EMEA	GV	KDE	**avg**
zipporah-synthetic	29.59	36.42	24.61	27.60	35.47	27.50	29.57	**30.20**
zipporah-paracrawl-10000	29.56	36.75	24.24	27.57	35.36	27.70	29.32	**30.16**
zipporah-paracrawl	29.13	36.43	23.25	27.26	35.06	27.32	29.20	**29.75**
best shared task submission	32.41	39.85	27.43	28.31	36.70	29.26	30.79	**32.06**

Table 1: Results of our Zipporah variants, compared to the submission with the best average test score.

In general, our systems lag behind the top performing systems by about 3 BLEU on the average of the six test sets. The different Zipporah systems perform similarly, with a slight edge to the original version with synthetic parallel data. This indicates that a subset can be used for faster training of Zipporah.

Zipporah does not require building an initial NMT system to score the data, as required by some of the top performing systems. Zipporah also has a very fast run time, the most expensive part being the language model scoring.

Our submissions are more competitive in the SMT experiments, and lag behind the top performing system system by less than a BLEU point (averaged across the test sets) for SMT systems trained on 100 million sentences. This may be due to the fact that Zipporah's adequacy and fluency scores directly track the translation and language model components of SMT.

5 Conclusion

Our submission to the WMT 2018 shared task on parallel corpus filtering was based on our Zipoorah toolkit. We varied methods to generate negative samples for the classifier to detect noisy sentence pairs, with similar results for synthetic noise, the full raw corpus to be filtered, and a subset of it.

We note that our method is quite simple and fast, using only n-gram language model and bag-of-words translation model features.

Acknowledgments

This work was in part supported by the IARPA MATERIAL project and a Google Faculty Research Award.

References

Amittai Axelrod, Yogarshi Vyas, Marianna Martindale, Marine Carpuat, and Johns Hopkins. 2015. Class-based n-gram language difference models for data selection. In *IWSLT (International Workshop on Spoken Language Translation)*.

Lei Cui, Dongdong Zhang, Shujie Liu, Mu Li, and Ming Zhou. 2013. Bilingual data cleaning for smt using graph-based random walk. In *ACL (2)*, pages 340–345.

Michael Denkowski, Greg Hanneman, and Alon Lavie. 2012. The cmu-avenue french-english translation system. In *Proceedings of the NAACL 2012 Workshop on Statistical Machine Translation*.

Kevin Duh, Graham Neubig, Katsuhito Sudoh, and Hajime Tsukada. 2013. Adaptation data selection using neural language models: Experiments in machine translation. In *ACL (2)*, pages 678–683.

Chris Dyer, Adam Lopez, Juri Ganitkevitch, Johnathan Weese, Ferhan Ture, Phil Blunsom, Hendra Setiawan, Vladimir Eidelman, and Philip Resnik. 2010. cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *Proceedings of the Association for Computational Linguistics (ACL)*.

Miquel Esplá-Gomis and M Forcada. 2009. Bitextor, a free/open-source software to harvest translation memories from multilingual websites. *Proceedings of MT Summit XII, Ottawa, Canada. Association for Machine Translation in the Americas*.

Kenneth Heafield. 2011. KenLM: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*.

Huda Khayrallah and Philipp Koehn. 2018. On the impact of various types of noise on neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 74–83, Melbourne, Australia. Association for Computational Linguistics.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics.

Yajuan Lü, Jin Huang, and Qun Liu. 2007. Improving statistical machine translation performance by training data selection and optimization. In *EMNLP-CoNLL*, volume 34, pages 3–350.

Robert C Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 conference short papers*, pages 220–224. Association for Computational Linguistics.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Anthony Rousseau. 2013. Xenc: An open-source tool for data selection in natural language processing. *The Prague Bulletin of Mathematical Linguistics*, 100:73–82.

Kashif Shah and Lucia Specia. 2014. Quality estimation for translation selection. In *Proceedings of the 17th Annual Conference of the European Association for Machine Translation, Dubrovnik, Croatia*.

Kaveh Taghipour, Shahram Khadivi, and Jia Xu. 2011. Parallel corpus refinement as an outlier detection algorithm. *Proceedings of the 13th Machine Translation Summit (MT Summit XIII)*, pages 414–421.

Hainan Xu and Philipp Koehn. 2017. Zipporah: a fast and scalable data cleaning system for noisy web-crawled parallel corpora. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2945–2950. Association for Computational Linguistics.