Proceedings of the 1st Joint SLTU and CCURL Workshop (SLTU-CCURL 2020), pages 240–244
Language Resources and Evaluation Conference (LREC 2020), Marseille, 11–16 May 2020

Multiple Segmentations of Thai Sentences for Neural Machine Translation

Alberto Poncelas[1], Wichaya Pidchamook[2], Chao-Hong Liu[3], James Hadley[4], Andy Way[1]
[1]ADAPT Centre, School of Computing, Dublin City University, Ireland
[2]SALIS, Dublin City University, Ireland
[3]Iconic Translation Machines
[4]Trinity Centre for Literary and Cultural Translation, Trinity College Dublin, Ireland
{alberto.poncelas, andy.way}@adaptcentre.ie
ch.liu@acm.org, wichaya.pidchamook2@mail.dcu.ie, HADLEYJ@tcd.ie

Abstract

Thai is a low-resource language, so it is often the case that data is not available in sufficient quantities to train an Neural Machine Translation (NMT) model which perform to a high level of quality. In addition, the Thai script does not use white spaces to delimit the boundaries between words, which adds more complexity when building sequence to sequence models. In this work, we explore how to augment a set of English–Thai parallel data by replicating sentence-pairs with different word segmentation methods on Thai, as training data for NMT model training. Using different merge operations of Byte Pair Encoding, different segmentations of Thai sentences can be obtained. The experiments show that combining these datasets, performance is improved for NMT models trained with a dataset that has been split using a supervised splitting tool.

Keywords: Machine Translation, Word Segmentation, Thai Language

In Machine Translation (MT), low-resource languages are especially challenging as the amount of parallel data available to train models may not be enough to achieve high translation quality. One approach to mitigate this problem is to augment the training data with similar languages (Lakew et al., 2018) or artificially-generated text (Sennrich et al., 2016a; Poncelas et al., 2018b).

State-of-the-art Neural Machine Translation (NMT) models are based on the sequence-to-sequence framework (i.e. models are built using pairs of sequences as training data). Therefore, sentences are modeled as a sequence of tokens. As Thai is a *scriptio continua* language (where whitespaces are not used to indicate the boundaries between words) sentences need to be segmented in the preprocessing step in order to be converted into sequences of tokens.

The segmentation of sentences is a well-known problem in Natural Language Processing (NLP), Thai is no exception. How Thai sentences should be split[1] has been discussed at the linguistic level (Aroonmanakun, 2007) and different algorithms have been proposed (Haruechaiyasak et al., 2008) including dictionary-based and machine learning-based approaches.

In this work, we use the problem of segmentation to our advantage. We propose using an unsupervised training method such as Byte Pair Encoding (BPE) to obtain different versions of the same sentence with different segmentations (by using different merge operations).

This leads to different parallel sets (each with different segmentation on the Thai side). These sets can be combined to train NMT models and achieve better translation performance than using one segmentation strategy alone.

[1]In this work, we use interchangeably the term split or segmentation to refer to the process of dividing a word or a sentence into sub-units.

1. Combination of Segmented Texts

As the encoder-decoder framework deals with a sequence of tokens, a way to address the Thai language is to split the sentence into words (or tokens). We investigate three splitting strategies: (i) Character-based, using each character as token. This is the simplest approach as there is no need to identify the words; (ii) Supervised word-segmentation: using tokenizers which have been trained on supervised data such as the "Deepcut"[2] library (Kittinaradorn et al., 2018) (trained on the BEST corpus (Kosawat et al., 2009)); and (iii) Language-independent split: useing a language-independent algorithm for spliting such as BPE (Sennrich et al., 2016b). This last method splits the sentence based on the statistics of sequences of letters. Therefore the split is not necessarily word-based. BPE is executed as follows: Initially it considers each character as an independent token; then iteratively the most frequent subsequent pair of tokens x and y are merged into a single token xy. The process is executed iteratively until a given number of merge operation are completed.

Once the sentences of a parallel corpus have been split, they can be used as training data for an NMT engine. Among the three approaches, the supervised word-segmentation strategy is the one that might allow NMT to perform the best, as it encodes the information from the manually-segmented sentences. However, by changing the merge operation parameter of BPE, the set can be split in different ways. Accordingly, in this work we want to explore whether augmenting the training set with the same sentences (but split with BPE using different merge operations) can be beneficial for training an NMT model. By doing this, we artificially increase the vocabulary on the target side. This causes the number of translation candidates of source-side words to be extended, which has been shown to have a positive impact in other approaches such as in statistical

[2]`https://github.com/rkcosmos/Deepcut`

MT (Poncelas et al., 2016; Poncelas et al., 2017). In addition, we want to explore whether this approach can be followed to augment a dataset split using Deepcut to improve the performance of NMT models.

Note that in this work we are building models in the English-to-Thai direction. The datasets with different splits are only on the target side whereas we keep the same number of merge operations on the source side. The main reason for this is that using several splits on the source side would also mean that NMT models would be evaluated with different versions (different splits) of the test set, which would obviously affect the results.

2. Data and Model Configuration

The models are built in the English to Thai direction using OpenNMT-py (Klein et al., 2017). We keep the default settings of OpenNMT-py: 2-layer LSTM with 500 hidden units and a maximum vocabulary size of 50000 words for each language.

We use the Asian Language Treebank (ALT) Parallel Corpus (Riza et al., 2016) for training and evaluation. We split the corpus so we use 20K sentences for training and 106 sentences for development. We use Tatoeba (Tiedemann, 2012) for evaluating the models (173 sentences).

In order to evaluate the models we translate the test set and measure the quality of the translation using an automatic evaluation metric, which provides an estimation of how good the translation is by comparing it to a human-translated reference. As the evaluation is made in a Thai text (which does not contain *n*-grams, we use CHRF3 (Popovic, 2015) which is a character-level metric, instead of BLEU (Papineni et al., 2002), which is based on overlap of *n*-grams.

The English side is tokenized and truecased (applying the proper case of the words) and we apply BPE with 89500 operations (the default explored in Sennrich et al. (2016b)).

For the Thai side we explore combinations of different approaches of sentence segmentation:

1. Character-level: Split the sentences character-wise, so each character is a token of the sequence (*character* dataset).

2. Deepcut: Split the sentences using the Deepcut tool.

3. Use BPE with different merge operations. In this work we explore with: 1000, 5000, 10000 and 20000 merge operations (*BPE 1000*, *BPE 5000*, *BPE 10000* and *BPE 20000* datasets, respectively). However, we propose as future work investigating the performance when using BPE with more merge operations.

In total there are six different datasets. Note that they contain the same unique sentences. We train NMT models using these datasets either independently or in combination. The combination is carried out by appending different datasets. In particular we use *character*, *Deepcut* or *BPE 1000* datasets and then accumulatively add *BPE 1000*, *BPE 5000*, *BPE 10000* and *BPE 20000* datasets.

dataset	CHRF3
Deepcut	47.90
character	20.70
BPE 1000	39.88
BPE 5000	45.49
BPE 10000	41.75
BPE 20000	38.77

Table 1: Performance of the NMT model trained with data using different methods for word segmentation on the target-side.

3. Experimental Results

First, we evaluate the performance of the models when different merge operations are used on the Thai side. The performance of the model, evaluated using CHRF3 are presented in Table 1. In the table, each row shows the results (the evaluation of the translation of the test set) of an NMT model built with the training set split following one of the thee chosen approaches. For example, the first row *Deepcut* present the results when the model is built with the set after being split using the Deepcut tool, and in the row *BPE 1000* if the dataset is split with BPE using 1,000 merge operations.

As expected, we see in the table that the best results are obtained when using the Deepcut tool (*Deepcut* row), which splits Thai words grammatically.

Regarding the models in which BPE has been used, we see in Table 1 that the best performance is achieved when 5000 merge operations is used (*BPE 5000* row).

When too many merge operations are performed there is the risk of merging characters of different words into a single token, so the score decreases.

By contrast, if there are too few merge operations, the resulting text is closer to character-split which, as we can see in the *character* row, leads to the worst results.

3.1. Combination of Datasets with Unsupervised Split

dataset	CHRF3
character	20.70
+ 1000	**38.23**
+ 5000	**45.63**
+ 10000	**49.93**
+ 20000	**52.17**
BPE 1000	39.88
+ 5000	**49.54**
+ 10000	**52.33**
+ 20000	**53.45**

Table 2: Performance of the NMT model trained with data combining different unsupervised methods for word segmentation on the target-side. Numbers in bold indicate statistical significant improvement at p=0.01.

The main focus of this paper is to study the impact on the performance when several training sets with different splits

are gathered together. We show the results in Table 2. Note that only the training set has been augmented, the development and test set remains the same for all experiments. In the table, each subtable presents the results when the sets are appended. For example, in the first subtable the row *character* indicates the performance of the model trained with the data split by character (the same as in Table 1). The following row, *+1000*, shows the score of the model trained with character-wise split and that split with BPE using 1000 merge operations (so there are two instances of each source sentence). The last row of the subtable (*+20000*) indicates the performance of the model trained with the five datasets concatenated (five instances of each source sentence).

In the table we observe that combining datasets with different segmentations is beneficial. In each subtable of Table 2 we see that the score goes up when we add more datasets with different splits. For example, the performance of the model using data split in a character-wise achieves a score of only 20.70 (first subtable). When we concatenate the same dataset but split using 1000 merge operations, the score increases to 38.23 (85% improvement) and we see that the score increases as we add sentences with different splits. This effect is observed for all three subtables.

The results also show that the lowest scores are achieved in those datasets in which a character-wise split fashion is performed. For example, the performance when using the dataset split with BPE using 1000 operations is 39.88 (*BPE 1000* row in Table 2), whereas when used in combination with character-split set (*+1000* row in the first subtable of Table 2) the score is 38.23. Another indication that character-wise splitting hurts performance is the comparison between the subtables of Table 2. If we compare the rows *+1000*, *+5000*, *+10000* and *+20000* of each subtable, the lower scores are seen in the first subtable

Another research question we want to answer is whether using a combination of splits using BPE can outperform the model trained with a language-independent tool like Deepcut. We see that in the second subtable of Table 2 all of the the combination of BPE-split datasets (*+5000*, *+10000* and *+20000* rows) exceed the 47.90 score achieved by using Deepcut alone.

3.2. Augmenting the Dataset Split with Deepcut

dataset	CHRF3
Deepcut	47.90
+ 1000	47.00
+ 5000	**51.25**
+ 10000	**54.94**
+ 20000	**54.96**

Table 3: Performance of combination of dataset with different split using *Deepcut*. Numbers in bold indicate statistical significant improvement at p=0.01.

As combining different splits can increase the performance of the model, does it also help when used in combination with Deepcut?

In Table 3 we present the results when the dataset split using Deepcut is augmented with datasets split with different merge operations using BPE. We see that initially, adding the set split with BPE using 1,000 merge operation (*+1000* row) causes the performance to drop slightly. Nonetheless, the performance increases when more data are added (i.e. *+5000*, *+10000*, *+20000* rows). Moreover, it even outperforms the model trained with the Deepcut-split dataset alone.

3.3. Analysis

In Table 4 we show some examples of the output. For each sentence we present the translations produced by the NMT model trained on *Deepcut*, *BPE 20000* and *Deepcut BPE 20000*. We do not include the outputs of the NMT models trained on the *character* dataset as all the translation consisted of the same, meaningless, sequence of characters. In the third column of the table we show the translation of the output of the NMT system after it has been postedited by removing characters (in gray) or replaced (in blue).

In the first sub-table we present how the sentence "the train is here" (meaning that the train is here because it has arrived) has been translated by different models. On the one hand, we see in the *Deepcut* row that the model has not produced an accurate translation. On the other hand, the models trained with data containing sentences of different segmentations (i.e. *BPE 20000* and *Deepcut BPE 20000*) achieve a more accurate translation. Nevertheless, the output is not a perfect translation as it indicates where the train is located instead of expressing that it has arrived.

In the following subtables we observe a similar effect. The models trained with the *Deepcut* dataset produced a translation that is either inaccurate or makes no sense. However, the models trained on *BPE 20000* or *Deepcut BPE 20000* produce a translation closer to the input after some post-editing (i.e. removing the characters in gray).

4. Related Work

There are several studies aiming to address the problem of splitting words in Thai. One of the first approaches to segmenting is the *longest-matching* method (Poowarawan, 1986), consisting of identifying the longest sequence of characters that match a word in the dictionary. Another approach is *maximal-matching* method (Sornlertlamvanich, 1993), which consists of generating all possible segmentations and retrieving those containing the smallest amount of words.

Haruechaiyasak and Kongyoung (2009) used conditional random fields to classify each character as either *word-beginning* or *intra-word*. Nararatwong et al. (2018) proposed an improvement to this approach by adding information from POS tags.

The use of several segmentations has also been proposed by Kudo (2018), in which he tries to integrate candidates from different segmentations. This technique has applications in a number of topics such as word-alignment (Xi et al., 2011) or language modeling (Seng et al., 2009; Abate et al., 2010).

The use of multiple instances in the training data, where only one side is modified, has been used by Poncelas et al. (2019), who showed that using multiple instances of the

Dataset	Output	
Source Sentence	the train is here.	
Reference	รถไฟมาแล้ว	
Deepcut	รถไฟนี้เป็นสายพันธุ์	This train is a breed.
BPE 20000	รถไฟอยู่ที่นี่อยู่ที่นี่ด้วย	The train is located here.
Deepcut + BPE 20000	รถไฟนี้อยู่นี่	This train is located here.
Source Sentence	I have a new bicycle.	
Reference	ฉันมีจักรยานใหม่	
Deepcut	ผมได้ทำการทดลองใหม่	I made a new experiment.
BPE 20000	ผมมีรถบรรทุกคนใหม่ๆ	I have a new truck.
Deepcut + BPE 20000	ผมมีรถจักรยานชุดใหม่ขึ้นมาแล้ว	I have a new set of bicycles.
Source Sentence	she does not have many friends in Kyoto.	
Reference	เธอไม่ค่อยมีเพื่อนมากนักที่เกียวโต	
Deepcut	เธอยังไม่มีใครได้รับบาดเจ็บ	She still doesn't have anyone injured.
BPE 20000	เธอไม่ได้มีเพื่อนหลายๆในเกียวโตเกียว	She does not have many friends in Kyoto.
Deepcut + BPE 20000	เธอไม่มีเพื่อนร่วมหลายคนในเกียวโต	She does not have many friends in Kyoto.

Table 4: Examples of translations using different splits

same target sentences, with different source-sides (generated by different MT engines) leads to better results than using a single instance of each sentence.

5. Conclusions and Future Work

In this work we have explored whether the duplication of training sentences with different splits is useful to build NMT models with improved performance. The experiments show that the combination of different splits on the target side does improve NMT models involving a low-resource language such as Thai.

The experiments also reveal that combining the same dataset using different merge operations of BPE not only improves the model trained on the dataset using the single configuration (regardless of the number of merge operations), but also the model trained on data that has been split using a tool trained on supervised data such as Deepcut.

In the future, we plan to conduct more fine-grained experiments to explore which configurations of BPE perform better. For example, would the combination of *BPE 10000* and *BPE 20000* (those with the highest number of operations explored) perform better than the model's original setup? And what would the results be if only *BPE 1000* and *BPE 20000* (those with the lowest and highest number of operation) are combined?

Furthermore, as all the experiments with different splits have been applied on the target side, we plan to investigate NMT models when Thai is on the source side. Similarly, we will also investigate whether these improvements will be achieved using other languages.

Another variation we are interested in exploring is not to replicate all the sentences but to use data-selection algorithms to find a subset of sentences that may boost the performance of the models trained on the subset (Poncelas et al., 2018a; Poncelas et al., 2018c).

Finally, we would like to investigate the applicability of the method of employing several segmentations to other NLP tasks such as text classification (Apichai et al., 2019; Wangpoonsarp et al., 2019).

6. Acknowledgements

The QuantiQual Project, generously funded by the Irish Research Council's COALESCE scheme (COALESCE/2019/117).

This research has been supported by the ADAPT Centre for Digital Content Technology which is funded under the SFI Research Centres Programme (Grant 13/RC/2106).

7. Bibliographical References

Abate, S. T., Besacier, L., and Seng, S. (2010). Boosting n-gram coverage for unsegmented languages using multiple text segmentation approach. In *23rd International Conference on Computational Linguistics*, pages 1–7, Beijing, China.

Apichai, C., Chan, K., and Suzuki, Y. (2019). Classifying short text in social media for extracting valuable ideas. In *20th International Conference on Computational Linguistics and Intelligent Text Processing*, La Rochelle, France.

Aroonmanakun, W. (2007). Thoughts on word and sentence segmentation in thai. In *Proceedings of the Seventh Symposium on Natural language Processing*, pages 85–90, Pattaya, Thailand.

Haruechaiyasak, C. and Kongyoung, S. (2009). Tlex: Thai lexeme analyser based on the conditional random fields. In *Proceedings of 8th International Symposium on Natural Language Processing*, Bangkok, Thailand.

Haruechaiyasak, C., Kongyoung, S., and Dailey, M. (2008). A comparative study on Thai word segmentation approaches. In *2008 5th International Conference on Electrical Engineering/Electronics, Computer,*

Telecommunications and Information Technology, volume 1, pages 125–128.

Kittinaradorn, R., Chaovavanich, K., Achakulvisut, T., and Kaewkasi, C. (2018). A Thai word tokenization library using deep neural network.

Klein, G., Kim, Y., Deng, Y., Senellart, J., and Rush, A. M. (2017). Opennmt: Open-source toolkit for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics-System Demonstrations*, pages 67–72, Vancouver, Canada.

Kosawat, K., Boriboon, M., Chootrakool, P., Chotimongkol, A., Klaithin, S., Kongyoung, S., Kriengket, K., Phaholphinyo, S., Purodakananda, S., Thanakulwarapas, T., et al. (2009). BEST 2009: Thai word segmentation software contest. In *2009 Eighth International Symposium on Natural Language Processing*, pages 83–88, Bangkok, Thailand.

Kudo, T. (2018). Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia.

Lakew, S. M., Erofeeva, A., and Federico, M. (2018). Neural machine translation into language varieties. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 156–164, Brussels, Belgium.

Nararatwong, R., Kertkeidkachorn, N., Cooharojananone, N., and Okada, H. (2018). Improving thai word and sentence segmentation using linguistic knowledge. *IEICE TRANSACTIONS on Information and Systems*, 101(12):3218–3225.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA.

Poncelas, A., Way, A., and Toral, A. (2016). Extending feature decay algorithms using alignment entropy. In *International Workshop on Future and Emerging Trends in Language Technology*, pages 170–182, Seville, Spain.

Poncelas, A., de Buy Wenniger, G. M., and Way, A. (2017). Applying n-gram alignment entropy to improve feature decay algorithms. *The Prague Bulletin of Mathematical Linguistics*, 108(1):245–256.

Poncelas, A., de Buy Wenniger, G. M., and Way, A. (2018a). Data selection with feature decay algorithms using an approximated target side. In *15th International Workshop on Spoken Language Translation (IWSLT 2018)*, pages 173–180, Bruges, Belgium.

Poncelas, A., Shterionov, D., Way, A., de Buy Wenniger, G. M., and Passban, P. (2018b). Investigating backtranslation in neural machine translation. In *21st Annual Conference of the European Association for Machine Translation*, pages 249–258, Alacant, Spain.

Poncelas, A., Way, A., and Sarasola, K. (2018c). The ADAPT system description for the IWSLT 2018 Basque to English translation task. In *15th International Workshop on Spoken Language Translation*, pages 76–82, Bruges, Belgium.

Poncelas, A., Popovic, M., Shterionov, D., de Buy Wenniger, G. M., and Way, A. (2019). Combining SMT and NMT back-translated data for efficient NMT. In *Proceedings of Recent Advances in Natural Language Processing (RANLP)*, pages 922–931, Varna, Bulgaria.

Poowarawan, Y. (1986). Dictionary-based thai syllable separation. In *Proc. Ninth Electronics Engineering Conference (EECON-86)*, pages 409–418, Bangkok, Thailand.

Popovic, M. (2015). chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal.

Riza, H., Purwoadi, M., Uliniansyah, T., Ti, A. A., Aljunied, S. M., Mai, L. C., Thang, V. T., Thai, N. P., Chea, V., Sam, S., et al. (2016). Introduction of the asian language treebank. In *2016 Conference of The Oriental Chapter of International Committee for Coordination and Standardization of Speech Databases and Assessment Techniques (O-COCOSDA)*, pages 1–6.

Seng, S., Besacier, L., Bigi, B., and Castelli, E. (2009). Multiple text segmentation for statistical language modeling. In *10th Annual Conference of the International Speech Communication Association (Interspeech)*, pages 2663–2666, Brighton, United Kingdom.

Sennrich, R., Haddow, B., and Birch, A. (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, Berlin, Germany.

Sennrich, R., Haddow, B., and Birch, A. (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, Berlin, Germany.

Sornlertlamvanich, V. (1993). Word segmentation for thai in machine translation system. *Machine Translation*, pages 556–561.

Tiedemann, J. (2012). Parallel data, tools and interfaces in opus. In *Proceedings of the Eighth International conference on on Language Resources and Evaluation (LREC 2012)*, pages 2214–2218, Istanbul, Turkey.

Wangpoonsarp, A., Shimura, K., and Fukumoto, F. (2019). Acquisition of domain-specific senses and its extrinsic evaluation through text categorization. In *20th International Conference on Computational Linguistics and Intelligent Text Processing*, La Rochelle, France.

Xi, N., Tang, G., Li, B., and Zhao, Y. (2011). Word alignment combination over multiple word segmentation. In *Proceedings of the ACL 2011 Student Session*, pages 1–5, Portland, USA.