An Exploratory Approach to the Corpus Filtering Shared Task WMT20

Ankur Kejriwal
Department of
Computer Science
Johns Hopkins University
akejriw2@jhu.edu

Philipp Koehn
Center for Language and
Speech Processing
Johns Hopkins University
phi@jhu.edu

Abstract

This document describes an exploratory look into the Parallel Corpus Filtering Shared Task in WMT20. We submitted scores for both Pashto-English and Khmer-English systems combining multiple techniques like monolingual language model scores, length based filters, language ID filters with confidence and norm of embedings.

1 Introduction

For this task the participants were provided with a corpus of parallel data in Pashto-English (ps-en) and Khmer-English (km-en). Additional parallel and monolingual datasets were also provided. The task organizers built neural machine translation (NMT) systems from the scores produced, based on parallel training sets of 5 million words. These systems are sensitive towards noise (Khayrallah and Koehn, 2018) and thus, it becomes important to separate the useful data from the noise. We view the task as data that passes through a pipeline of filters in order to give us the best possible selection of 5 million words in the end.

We determined that language ID filtering is a very strong pre-processing filter and inducting confidence scores is not needed. We also determined that monolingual Language models can help us in selecting sentences even if both the source and target language models are independent of each other. Finally, using using the length of a sentence as a filter helps us create a better NMT system.

We also learn that statistical intuitions do not easily extend to neural embeddings.

2 Baseline

The idea behind our baseline system is to use the cosine distance between two multilingual representations as a notion of parallelism between sentences embedded in the same space. The tool we use for this is LASER[1] which uses an encoder-decoder architecture to train a multilingual sentence representation model. It has been shown by Artetxe and Schwenk (2018b) that LASER is effective at zero-shot cross-lingual natural language inference in the XNLI dataset which makes it promising for this task involving low-resource languages. Koehn et al. (2019) has shown this to be a strong baseline.

We follow the work done by Artetxe and Schwenk (2018a) and begin by generating multilingual sentence embeddings using LASER. The LASER score is a function of the margin between the cosine similarity between a given candidate and it's k nearest neighbors[2]:

Let f(x,y) =

$$\sum_{z \epsilon NN_k(x)} \frac{cos(x,z)}{2k} + \sum_{z \epsilon NN_k(y)} \frac{cos(y,z)}{2k}$$

LASER score(x,y) = margin(cos(x,y),f(x,y))

We experiment with the following definitions of margin:

- **Absolute:** margin(a,b) = a
  Essentially just cosine similarity.
- **Distance:** margin(a,b) = a - b
  Subtracting the average cosine similarity from that of the given candidate. We use this when there are certain points that are extraordinarily close to many other data points and thus, degrade the quality of nearest neighbors.
- **Ratio:** margin(a,b) = a / b
  This is the ratio between the candidate and the average cosine of its nearest neighbors in both directions.

[1] https://github.com/facebookresearch/LASER

[2] cos(x,y) here refers to cosine similarity between the vectors x and y

Proceedings of the 5th Conference on Machine Translation (WMT), pages 959–965

We also experiment with the following techniques of candidate generation:

- **Intersection:**
  Each source sentence is aligned with exactly one best scoring target sentence. Some target sentences may be aligned with multiple source sentences or with none. We repeat this process in the opposite direction and take the intersection of the 2 alignments

- **Max score:**
  We repeat the process used to generate candidates in Intersection, except we select the alignment with the highest score instead of discarding all inconsistent alignments

We find that the best settings were margin set to ratio, candidate generation set to max-score and k set to 4. Note that this list of nearest neighbors does not include duplicates, so even if a given sentence has multiple occurrences in the corpus, it would have (at most) one entry in the list (Chaudhary et al., 2019). These scores are in the range of [0,1].

The BLEU scores obtained by these systems are 11.16 for Pashto and 9.65 for Khmer.

3 Language ID

For our first step, we try to predict the most probable language of a given sentence using use fastText (Joulin et al., 2016). We use the pre-trained model released by the authors that is trained to identify over 170 languages including Pashto, Khmer and English. The intuition behind it is that when working in a bilingual setting, sentences from other languages or code-mixed sentences will be harmful to the MT system. We call this simple language ID

if source = ps|km and target = en **then**
  langID score = LASER score
else
  langID score = 0
end if

An additional idea that we incorporate is how confident we are when predicting the language of a sentence. For example, if we have a sentence where the probability of the target sentence being English is 0.9 and we have another sentence where the probability of the target sentence being English is 0.3, then given the same LASER scores, we would want to give preference to the sentence pair where the probability of the target sentence is 0.9. We do this for both the source and target language.

We try to only keep sentence pairs where we are confident about both the source and target language being correct. We implement this notion by setting a cutoff c for the language ID probability. We call this confident language ID

if prob(src=ps|km) $>$ c and prob(tgt=en) $>$ c **then**
  confidence = prob(src=ps|km) $\cdot$ prob(tgt=en)
else
  confidence = 0
end if
langID score = LASER score $\cdot$ confidence

Where prob(p=q) is the probability of sentence p being from language q.

We show our results for Pashto in Table 1 and make the following observations.

- There is an overall increase in BLEU.
- Simple language ID seems to be better than Confident language ID by a small margin.
- Confident language ID, has a local minima at a cutoff of 0.75
- The scores in confident language ID tend to decrease sharply after a cutoff of 0.8.

This leads us to believe that while good for identifying the language, the confidence scores are not as strong as the LASER scores.

We experiment by including scores if they are within the top 3 predicted languages but see no significant change in scores.

We also experiment by adding the confidence instead of multiplying it in confident language ID but see no significant change in the BLEU scores.

As a result of our observations, we only perform Simple Language ID for Khmer, giving us a BLEU score of 11.51 for Pashto and 10.04 for Khmer.

4 Norm of embeddings as a filter[3]

Liu et al. (2020) showed us that the norm of an embedding can represent how frequent and how context insensitive the embedding is. Essentially, smaller norms represent frequent and context-insensitive rare words. There is an implicit assumption here that the embedding size is large enough to incorporate all the information present in a sentence.

[3] *Note: Throughout this section we shall be using the terms "vector" and "embedding" interchangeably.*

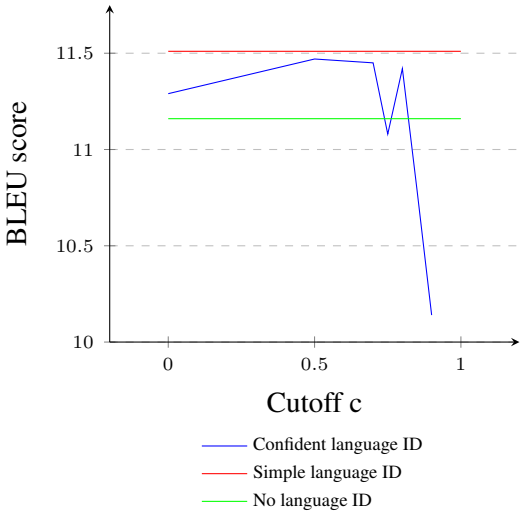

Figure 1: Cutoff vs BLEU scores for Pashto

For a vector $x_i = [x_{i1}, x_{i2}, ..., x_{in}]$ the norm of the vector is

$$norm(x_i) = \sqrt{x_{i1}^2 + x_{i2}^2 + ... + x_{in}^2}$$

We interpret the norm of an embedding as a measure of the confidence we have in the embedding of a sentence. The reason we do this is that neural methods by their very nature are data hungry and susceptible to noise. We are working in a low-resource condition because of which it will be harder to learn about sentences with context-sensitive words. Additionally, it would be harder to learn about sentences with low-frequency words making their embeddings less reliable, thus, leading to a lower quality MT system.

We run 2 sets of experiments on the LASER embeddings of the sentences.

- We assume that the elements in each vector are comparable, i.e. on the same scale. We simply take the norm of the embedding in this case.

- We assume that the elements in a vector are not directly comparable. In this case, we compute the z-score of each element and take the norm of the z-scores. z-score can be thought of as how many standard deviations is a given element away from it's mean. Thus it gives each element a relative value, making them comparable. We finally take the norm of the z-scores.

  $$z\text{-}score = \frac{x-\mu}{\sigma}$$

  where μ is the mean and σ is the standard deviation of that particular element across all the vectors that have a non-zero langID score.

The langID scores we have until now are in the range [0,1], while the norm of the embeddings theoretically have a range of $[0, \infty)$. To ensemble the langID scores with the norm-scores, we need to bring the distributions within a comparable range. We do this by applying min-max normalization.

Let $x = [x_1, x_2, ..., x_n]$
where
x_i = norm (embedding (i^{th} vector))
or
x_i = norm (z-score (embedding (i^{th} vector)))
for all vectors with langID score $\neq 0$ **do**

$$\text{normalized } (x_i) = \frac{x_i - min(x)}{max(x) - min(x)}$$

end for
norm score = langID score - normalized

We observe that in both the cases, there are very few observations with a really high norm because of which over 95% of the norm scores remained the same up to 6^{th} or more decimal place. To counter this, we set the langID scores of these embeddings to 0 and repeat the process recursively till we see an impact on the 5 million token subselected corpus.

We make the following observations

- We see a drastic reduction in the Pashto BLEU scores from 11.51 to 9.76 when we use z-scores.

- When we do not use the z-scores, we see a decrease in the BLEU score from 11.51 to 11.37 for Pashto. and an increase from 10.04 to 10.05 for Khmer. In both the cases the change is really small and not significant.

- Manual observation here shows that sentences with big URL's were filtered out automatically without any explicitly stated rule.

- Being more aggressive with the number of loops led to a drastic decrease in BLEU scores.

We finally ran experiments where we gave a preference to only large norms and an experiment where we gave a preference to both very large norms and to very small norms. In both the cases we had really bad scores leading to the conclusion that the norm of LASER embeddings is not a good

filter. Because of this poor performance when relied on aggressively, we decided not to use it in our final submission.

We also ran the same experiments using fasttext generated embeddings (Bojanowski et al., 2017) but that had poor results as well.

5 Monolingual Language Models

The motivation behind using Monolingual Language Models is that we want to learn about sentence pairs that have a high likelihood of coming up in the test data. Ideally we would want both the sides of the corpus to have a high probability of coming up but we also realize it will often not be the case. Thus, we make these language models independent of each other. We take inspiration from Axelrod et al. (2019) and modify the work of Junczys-Dowmunt (2018) to come up with a language model filter. Junczys-Dowmunt (2018) achieved the highest ranking score in WMT'18 and to do so they define $H_M(.|.)$ as the word-normalized conditional cross-entropy of the probability distribution $P_M(.|.)$ for a model M:

$$H_M(y) = -\frac{1}{y}\sum_{t=1}^{|y|} logP_M(y_t|y_{<t})$$

We use this as a measure of how fluent a given sentence is. Lower scores indicate a better sentence in this case.

While Axelrod et al. (2019) do create n-gram language models, they hope that language models trained on similar but not parallel texts to have similar perplexities over each half of a parallel test set of the parallel corpus. This method does not leverage the simple fact that we have more data for one of the languages. This method also assumes a close relationship between the frequencies of letters which might not always be the case. Finally it does not leverage the expressive power of neural language models. In order to improve on this, we propose Neural Language Models that are completely independent of each other.

5.1 Pre-processing

We first tokenize our data using Sentencepiece (Kudo and Richardson, 2018). We set an upper limit of 5,000 on the vocabulary size for Pashto and Khmer, and an upper limit of 50,000 for the English vocabulary. This is done at both the character and word level and also both with and without splitting at whitespace. We also reverse the data to simulate right-to-left prediction of words. Thus we have 8 possible tokenizations for every possible sentence.

–arch transformer_lm
–dropout 0.1
–optimizer adam
–adam-betas '(0.9, 0.98)'
–weight-decay 0.01
–clip-norm 0.0
–lr 0.0005
–lr-scheduler inverse_sqrt
–warmup-updates 4000
–warmup-init-lr 1e-07 –patience 30

Figure 2: Language Model: Transformer architecture

–arch transformer_lm_wiki103
–max-lr 1.0
–t-mult 2
–lr-scheduler cosine
–lr-shrink 0.75
–warmup-init-lr 1e-07
–min-lr 1e-09
–optimizer nag
–lr 0.0001
–clip-norm 0.1 –patience 30

Figure 3: Language Model: Wiki103 architecture

5.2 Language Model Architecture

For our Language Models, we use fairseq (Ott et al., 2019) to implement the architecture given in Figure 2.

We also create a Language Model using the architecture given by Baevski and Auli (2019) with parameters as given in Figure 3.

We use 2 different models because while it is tempting to use deeper and more sophisticated models, we need to have enough data to train it sufficiently. If sufficient data is absent, it is in general better to train simpler models.

In total we have 16 language models for each language. In each case, we train the model in 2 further ways. In the first case, we keep the CommonCrawl monolingual data as the training set and keep the Wikipedia monolingual data as the development set. In the second case, we augment the CommonCrawl data with Wikipeda data by oversampling. We make the number of lines taken by the wikipedia data be between 40-50% of the number of lines taken by the CommonCrawl data.

During training, the word level language models that were not split on white-space were taking too much time to train. As a result we had to halt their training and use the best checkpoint achieved till then.

5.3 Evaluating the Language models

We evaluated the Language Models by using their word normalized cross entropy as defined in Equation 5. We find that the scores we got were extremely similar whether we used just the CommonCrawl data or whether we augmented it with the Wikipedia data. In addition, the Transformer architecture gave us better results.

We evaluate our created development set and find that the word-level left-to-right and character level left-to-right language models had the lowest perplexities of their respective groups.[4] Additionally, the language models created for Pashto were relatively much stronger than the language models created for Khmer simply because of the script in which Khmer is written.

5.4 Normalising LM-scores

Once again, the values of langID scores range between [0,1] and our language models scores range between [0,∞]. In order to ensemble them, we need to bring them to a comparable scale. We again try to use min-max normalization to change the range of the cross entropy values to be [0,1]. Once again we run into the same problem where the value of maximum is so high that the change bought about in langID scores was negligible. Instead of going for a recursive approach, we take a more aggressive approach this time. We average over the cross entropy values of sentences that we would have selected using langID scores and we replace the max(crossEntropy) with an empirically chosen value close to it.

Let $x = [x_1, x_2, ..., x_n]$
where x_i = cross entropy (i^{th} vector)
for all vectors **do**

$$\text{new entropy } (x_i) = \frac{x_i - min(x)}{max(x) - min(x)}$$

end for
LM score = langID score - new entropy

We display the results for only Pashto and a few English Pashto ensemble models in Figure 4.

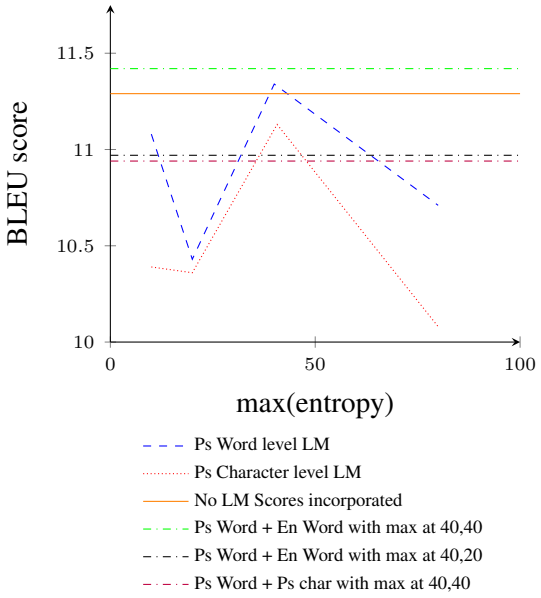

Figure 4: BLEU score vs max(entropy) for Pashto

We can also see that just the word level language models perform better than both the character level models and the Word + Character ensemble models.[5] Similar trends were observed for Khmer.

5.5 Length-based filters

We observed that our language model had a tendency to pick proper nouns. While we want our language model to learn about names, we don't want to select them over sentences because our translation is model is based on sentences. To counteract this, we decided to add a simple length based filters to Pashto and English. Since they both used white spaces and have an almost 1 to 1 mapping, we added a penalty of -1 to their score if any of the sentences were below 0, 5 and 8 in length. We call this cutoff value lc. We decide on a penalty of -1 because the maximum score that any sentence can get right now is 1. This results in that sentence not being selected at all. Following work described in Koehn et al. (2019) we also add a penalty term of -1 whenever either the source or the target sentence was over 3 times it's counterpart in length. We call this length ratio cutoff lr

if len(src) $<$ lc and len(tgt) $<$ lc **then**
    LM score = LM score -1
end if
if len(src)/len(tgt) $>$ lr or len(tgt)/len(src) $>$ lr **then**
    LM score = LM score - 1

[4]Experiments on ensembling these models were still running at the time of submission

[5]We observed that the perplexities on the english side tended to be about half of Pashto and one fourth of Khmer.

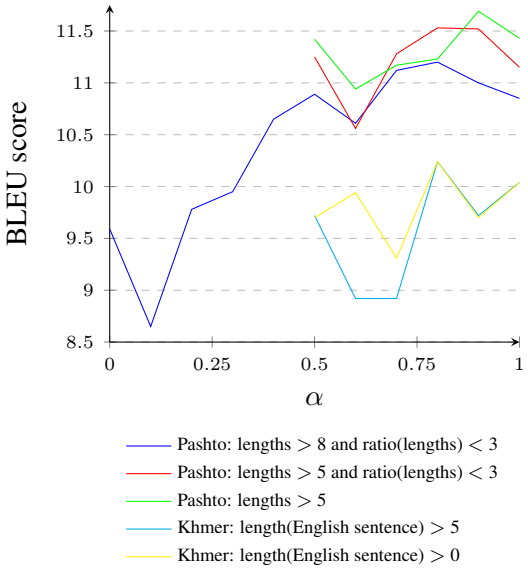

Figure 5: Change in BLEU scores with respect to the amount of weight given to Language Models

end if

In the Khmer-English case, we only experimented with a sentence length on the English side since there were no clear demarcations in Khmer.

if len(tgt) $<$ lc **then**
LM score = LM score -1
end if

We then experiment with a penalty if there is an overlap of more than 60% between the tokenized texts. However this doesn't show any significant changes.

5.6 Assigning Weights to different scoring mechanisms

While we tried to give an equal weight to the Language model and LASER scores, we had no good reason to believe that we should. We introduced a hyperparameter α which lies between 0 and 1, and we change the equation of LM score to be

Let $x = [x_1, x_2, ..., x_n]$
where x_i = cross entropy (i^{th} vector)
for all vectors **do**

$$\text{new entropy } (x_i) = \frac{x_i - min(x)}{40 - min(x)}$$

end for
LM score = (α)(langID score) - (1-α)(LM score)

We replaced the maximum with the value 40 from our findings in Section 5.4

We first run this on Pashto sentences with lc $>$ 8. We use those results to narrow our seaarch with lc $>$ 5 and lc $>$ 0. We then use those results to run experiments for Khmer. The combined results are shown in Figure 5

From figure 5 we can see that we have to local maximas around 0.5 and 0.8 and a global maxima at 0.8. In some cases the global maxima is at 0.9 and in some at 0.8. This leads us to believe that these are the most suitable values for the task.

6 Final Submission

Our final pipeline is as follows:

1. Obtain LASER scores for each sentence pair
2. Pass it through a language id filter where we set the LASER score to 0 if either the source or target language doesn't match
3. score the source and target half of the parallel corpus using monolingual language models
4. combine the language model scores with LASER scores
5. Apply a length based filter to remove sentences that don't provide too much information

We submit what we believe to be the 3 most robust solutions we have for each language pair. For Pashto, we apply the language id filter, then we use the Transformer architecture language model, set lr to 3 and set(α, lc) to (0.8, 5), (0.9, 0) and (0.9, 5). For Khmer, we apply the language id filter, then we use the Transformer architecture language model and set (α, lc) at (0.8, 5) and (0.8, 6). Apart from that, we also submit a scores with a filter checking for token overlap over 30%. The (α, lc) is set to (0.8, 5). At the time of submission our best score for Pashto is 11.69 and the best score for Khmer is 10.24 on the development set. The findings of the shared task presented by Koehn et al. (2020)

References

Mikel Artetxe and Holger Schwenk. 2018a. Margin-based parallel corpus mining with multilingual sentence embeddings.

Mikel Artetxe and Holger Schwenk. 2018b. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond.

Amittai Axelrod, Anish Kumar, and Steve Sloto. 2019. Dual monolingual cross-entropy delta filtering of noisy parallel data. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 247–253, Florence, Italy. Association for Computational Linguistics.

Alexei Baevski and Michael Auli. 2019. Adaptive input representations for neural language modeling. In *International Conference on Learning Representations*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Vishrav Chaudhary, Yuqing Tang, Francisco GuzmÃ¡n, Holger Schwenk, and Philipp Koehn. 2019. Low-resource corpus filtering using multilingual sentence embeddings. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 263–268, Florence, Italy. Association for Computational Linguistics.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Marcin Junczys-Dowmunt. 2018. Dual conditional cross-entropy filtering of noisy parallel corpora. In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 901–908, Belgium, Brussels. Association for Computational Linguistics.

Huda Khayrallah and Philipp Koehn. 2018. On the impact of various types of noise on neural machine translation. *arXiv preprint arXiv:1805.12282*.

Philipp Koehn, Vishrav Chaudhary, Ahmed El-Kishky, Naman Goyal, Peng-Jen Chen, and Francisco Guzmán. 2020. Findings of the WMT 2020 shared task on parallel corpus filtering and alignment. In *Proceedings of the Fifth Conference on Machine Translation: Shared Task Papers*.

Philipp Koehn, Francisco GuzmÃ¡n, Vishrav Chaudhary, and Juan Pino. 2019. Findings of the wmt 2019 shared task on parallel corpus filtering for low-resource conditions. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 56–74, Florence, Italy. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Xuebo Liu, Houtim Lai, Derek F Wong, and Lidia S Chao. 2020. Norm-based curriculum learning for neural machine translation. *arXiv preprint arXiv:2006.02014*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.