A Statistical Extension of Byte-Pair Encoding

David Vilar[*]
Amazon

Marcello Federico
Amazon

Abstract

Sub-word segmentation is currently a standard tool for training neural machine translation (MT) systems and other NLP tasks. The goal is to split words (both in the source and target languages) into smaller units which then constitute the input and output vocabularies of the MT system. The aim of reducing the size of the input and output vocabularies is to increase the generalization capabilities of the translation model, enabling the system to translate and generate infrequent and new (unseen) words at inference time by combining previously seen sub-word units. Ideally, we would expect the created units to have some linguistic meaning, so that words are created in a compositional way. However, the most popular word-splitting method, Byte-Pair Encoding (BPE), which originates from the data compression literature, does not include explicit criteria to favor linguistic splittings, nor to find the optimal sub-word granularity for the given training data. In this paper, we propose a statistically motivated extension of the BPE algorithm and an effective convergence criterion that avoids the costly experimentation cycle needed to select the best sub-word vocabulary size. Experimental results with morphologically rich languages show that our model achieves nearly-optimal BLEU scores and produces morphologically better word segmentations, which allows to outperform BPE's generalization in the translation of sentences containing new words, as shown via human evaluation.

1 Introduction

Sub-word segmentation is currently a standard tool for machine translation systems (see e.g. the systems submitted to WMT and IWSLT evaluations (Barrault et al., 2019; Niehues et al., 2019), as well as systems for a wide variety of NLP tasks (see e.g. Devlin et al. (2018) and derived works). The goal is to split words (both in the source and target language) into smaller units which then constitute the input and output of the machine translation system. The goal is twofold: On the one hand, sub-word splitting reduces the size of the input and output vocabularies. This is specially important when using neural models, as the size of the input layer is fixed and thus the vocabulary size cannot be dynamically adjusted. On the other hand, it tries to increase the generalization capabilities of the translation model, enabling the system to accept and/or generate new words at translation time by combining previously seen units. The most widespread method used for sub-word splitting in neural machine translation is Byte Pair Encoding (BPE), introduced by Sennrich et al. (2016). Since then, BPE has become a default preprocessing step for many NLP tasks.

The BPE extraction algorithm is an adaptation of the algorithm introduced by Gage (1994) for data compression. The main idea of this algorithm is to replace the most frequent pair of bytes found in the input data with a new, unseen byte. The process is repeated until no more byte pairs are repeated or until no free bytes are available. Sennrich et al. (2016) took this algorithm as a starting point, considering characters instead of bytes, and joining them using the same criterion to produce sub-word units (more details can be found in Section 3).

One potential problem with this approach is that the objective of the original BPE algorithm differs from the goals for which it is being used for translation, as detailed above. While it is certainly effective for the first objective (reducing the vocabulary size), it is arguable whether it is appropriate for the goal of generating new words (Ataman et al., 2017; Huck et al., 2017; Banerjee and Bhattacharyya, 2018).

Intuitively, in order to generate new words, we would expect the sub-word units to have some linguistic meaning, so that a new word can be created

[*] Now at Google.

beklagen
↓
bek@@ lagen

bewertungsinstrumente
↓
bewer@@ t@@ ungsin@@ stru@@ mente

Table 1: Examples of unsatisfactory BPE splitting of German words. The two words are segmented by breaking the underlying morphological structure.

in a compositional way. Being purely frequency driven, BPE does not take this intuition into consideration, as illustrated in the two German word examples in Table 1 taken from the WMT'19 training data. For the first word, the split "be@@ klagen" would be more satisfactory as the word is derived from "klagen" (*complain*); the second word is a compound word, with the splits "bewertungs@@ instrumente" (*assessment instruments*), separating the two words, and "bewert@@ ung@@ s@@ instrument@@ e" being morphologically more informed alternatives.

The BPE algorithm also introduces an additional practical problem. The original formulation does not specify a criterion for stopping the creation of new symbols. If the algorithm runs for an unlimited time, it will merge all sub-words into the original input vocabulary, which is clearly undesired. In practice, one specifies a fixed number of merges to be carried out, or a threshold frequency and when the considered symbols fall below this value the algorithm is stopped. It is however not clear how to set these hyperparameters, although they can have a drastic effect on translation quality depending on the translation direction, task and amount of data (Denkowski and Neubig, 2017; Sennrich and Zhang, 2019). Furthermore, these hyperparameters are rarely optimized, as evaluating them constitutes a full training-evaluation cycle, which is notoriously costly.

In this paper we introduce a new criterion for defining sub-word units that tries to address these shortcomings. We introduce a probability distribution over the units which in turn induces a likelihood function over the corpus which we can optimize. We will show how this statistical approach can guide the extraction process towards more linguistically satisfying units, while still remaining a purely data driven approach. Having a well founded optimization criterion also allows us to define a data driven stopping criterion. Our proposed criterion allows to select a nearly optimal number of units using only an intrinsic measure on the training corpus, thus dramatically reducing experimentation costs.

2 Related work

As stated in the introduction, our starting point is the BPE algorithm introduced in (Sennrich et al., 2016). In this work, the authors adapt the data compression algorithm by Gage (1994) to the task of sub-word unit generation.

Some authors have tried to expand the extraction of sub-word units by leveraging linguistic information. Sánchez-Cartagena and Toral (2016) use morphological segmentation for Finnish and compare the effectiveness of these sub-word units for the WMT evaluation. The system using this segmentation approach together with other extensions performed best in human evaluation. Huck et al. (2017) follow a similar approach with the addition of compound splitting for translation into German, achieving improvements of around 0.5 BLEU points on WMT data. Ataman and Federico (2018) propose to replace BPE with unsupervised morphological segmentation which also takes morphological coherence into consideration during prediction of the sub-words. Experiments run under small-data conditions on TED Talks in five directions, all from/to English, show systematic improvements on Arabic, Turkish, Czech, but not on Italian and German. Banerjee and Bhattacharyya (2018) also use unsupervised morphological units generated by Morfessor (Virpioja et al., 2013) as input for a neural machine translation system and report improvements for low-resource conditions. Macháček et al. (2018) follow a similar approach for translation into Czech on WMT data, but were not able to obtains improvements over the standard BPE approach.

An alternative model to BPE which is also widely used was presented by Kudo (2018), which can be considered as an extension of (Schuster and Nakajima, 2012). They show that using a purely statistical approach, they are able to achieve sub-word units that are better linguistically motivated. Similar to our approach, a probability distribution over the sub-word units is defined with the goal of improving the likelihood over the training data. The strategy for defining the sub-word units differ in his approach and ours. While we start with single characters and expand the units, Kudo (2018) starts with a large set of sub-word units and prunes iteratively until reducing the number to a desired quantity. Segmentation probabilities are modeled with a multinomial distribution trained via expectation maximization.

In order to improve generalization of the segmentation model (i.e. performance on new words), different regularization approaches have been proposed. Kudo (2018) applies different segmentations at training time. For each parameter update, segmentations for each word are sampled from a smoothed posterior distribution computed from the multinomial distribution. Along the same line, Provilkov et al. (2019), proposed to generate alternative segmentations directly with BPE, by randomly dropping out merging rules. These approaches, as noted by Kudo (2018), can be seen as variants of the ensemble training principle, where many different models are trained (and finally combined) on different subsets of the training data. Our work differs with respect to (Kudo, 2018) in that we train an observable model in a stepwise fashion, like BPE, by maximizing the likelihood of the training data. Thus, we expect our approach to be more efficient than Kudo (2018). Differently from Kudo (2018) and Provilkov et al. (2019), we do not apply regularization, however nothing prevents from applying the drop out method also to our merging rules, although we expect that our model has already learned more general segmentation rules than BPE.

To the best of our knowledge, there has been little previous work on automatically determining the number of sub-word units to produce by segmentation algorithms. Kreutzer and Sokolov (2018) integrate segmentation into the NMT system and find that the system favors character-based translation over sub-word segmentation. Henderson (2020) pointed out that determining vocabulary sizes for NLP tasks is one of the few aspects that is still done manually, and suggests it as one possible direction for future improvement of NLP models.

3 The Byte Pair Encoding (BPE) algorithm

The BPE training algorithm as presented in (Sennrich et al., 2016) is shown in Algorithm 1. It closely follows the original BPE for data compression algorithm by Gage (1994). The algorithm receives as input a text as a sequence of words, which in turn are represented as sequences of characters. The single characters constitute the initial set of symbols. At each iteration the pair of symbols (occurring inside words) with highest frequency is selected and substituted with a new symbol. This substitution is recorded as a new rule. This merging operation is repeated for a fixed number of steps. The algorithm returns the sorted list of merging rules.

```
Algorithm 1: BPE training algorithm.
  Input: training corpus $S$ of words split into
         character sequences; number $N$ of
         rules
  Output: list $R$ of $N$ merge rules
1 $R :=$ [ ]
2 while $\text{length}(R) \leq N$ do
3     $(x, y) := \underset{(x,y)}{\text{argmax}} \{\text{count}_S(x, y)\}$
4     $rule := \langle (x, y) \rightarrow xy \rangle$
5     $S := \text{apply}(rule, S)$
6     $R := \text{append}(rule, R)$
7 return $R$
```

```
Algorithm 2: BPE inference algorithm
  Input: list $R$ of merge rules; word $w$ split
         into characters
  Output: segmented word
  foreach $rule \in R$ do
      if $\text{matches}(rule, w)$ then
          $w := \text{apply}(rule, w)$
          continue
  return $w$
```

Algorithm 2 shows how to apply the set of rules extracted by Algorithm 1 to a new text. It basically looks up the ordered list of rules and applies as many of them as possible.

4 The statistical BPE (S-BPE) algorithm

We can generalize the criterion for BPE unit selection by adjusting line 3 of Algorithm 1. Specifically, we define a probability distribution over the BPE units and define a maximum likelihood optimization criterion.

Let S be a corpus of words w from a vocabulary V, and let each word be decomposed as a sequence of symbols (initially characters) s from an alphabet Σ. The log-likelihood of S can be written as:

$$L(S, \Sigma) = \sum_{s \in \Sigma} C_{S,\Sigma}(s) \log \Pr(s) \qquad (1)$$

where $C_{S,\Sigma}(s)$ is the count of symbol s in corpus S, in which words are segments according to Σ, i.e.:

$$C_{S,\Sigma}(s) = \sum_{w \in V} C_S(w) C_\Sigma(s, w) \qquad (2)$$

Algorithm 1 initializes Σ with single characters (Σ_0). Then, at each step n of training, it selects the pair of symbols with the highest frequency or, equivalently, joint probability:

$$(x, y) = \operatorname*{argmax}_{x,y \in \Sigma_{n-1}} p_{n-1}(x, y) \qquad (3)$$

thus defining the new alphabet[1]

$$\Sigma_n = \{xy\} \cup \Sigma_{n-1} \qquad (4)$$

where the probability distribution p_{n-1} is defined over the elements of the alphabet Σ_{n-1}.

From a statistical modeling perspective, however, we would be more interested in rules for which the training data likelihood increases, i.e.:

$$L(S, \Sigma_n) > L(S, \Sigma_{n-1}) \qquad (5)$$

It can be shown (see the Appendix for a derivation) that for any pair of symbols $x, y \in \Sigma_{n-1}$, the following inequality holds, which provides a lower bound for the increase in likelihood:

$$\begin{aligned} L(S, \Sigma_n) >& L(S, \Sigma_{n-1}) \\ &+ C_{S,\Sigma_n}(xy) \log \frac{p_n(xy)}{p_n(x) p_n(y)} , \end{aligned} \qquad (6)$$

where as usual Σ_n includes xy as given in Equation 4. Intuitively we can interpret the rightmost term as the likelihood of each word that contains the bigram xy being increased by merging the two symbols[2]. It also provides a good tie-in to our linguistic intuition about sub-word units: if two units appear only in combination with each other, they probably do not have linguistic meaning on their own. Thus the probability mass will shift to the probability of the joint symbol, and the probability of the single elements will be greatly reduced. On the other hand, if x or y do have linguistic meaning, e.g. verb suffixes, they are likely to have a high probability of appearing in the text, and thus the gain from joining them together is not as big.

The above inequality thus suggests the new update rule:

$$\begin{aligned} (x, y) = & \operatorname*{argmax}_{(x,y):\Sigma_n = \{xy\} \cup \Sigma_{n-1}} C_{S,\Sigma_n}(xy) \times \\ & \left[\log p_n(xy) - \log p_n(x) - \log p_n(y) \right] . \end{aligned} \qquad (7)$$

Note an important difference between Equations (3) and (7): In (3) we use a bigram probability $p_{n-1}(x, y)$ computed on $\Sigma_{n-1} \times \Sigma_{n-1}$, while in (7) we use a unigram probability $p_n(xy)$ computed on Σ_n. The two probabilities are expected to be close but not the same.

Note that in practice, in the course of the algorithm the count for a unit may drop to 0 (due to all the occurrences being combined with another unit to form a new pair), thus producing a probability of 0. In order to avoid computation of $\log 0$ in Equation (7) we use Laplace smoothing for the computation of all probabilities.

4.1 Stopping criterion

One open question when defining BPE units is how many operations to carry out. As shown in Algorithm 1, this number is a parameter of the extraction algorithm, and there is no defined way to select it. The number of units has an important effect on the quality of the translation system (see Section 5), but selecting the optimal number involves training and testing a translation system for each candidate, at a high computational cost. Thus, normally system builders resort to previous experience and select a number of units that has worked well on previous tasks, although the performance can be very task dependent.

With the statistical formulation of BPE, for each operation we can compute a corresponding (approximate) increase in likelihood on the training corpus through Equation 6. Looking at the evolution of the likelihood, we can define a criterion of when to stop defining new units. Specifically, let us define δ_i as the (approximate) increase in likelihood when defining the i-th BPE unit. We will stop the algorithm, and thus define the number of units N, when $\delta_N \leq k\delta_1$, with $k < 1$. In order to improve the robustness of the criterion, in practice it is better to average each δ_i with the previous M values.

[1] Notice that by implementing Σ_n as an ordered list (stack), we get the list of rules R of Algorithm 1 and Algorithm 2.

[2] This is similar to the pointwise mutual information criterion used to detect collocations (Church and Hanks, 1990).

		Tokens	
Language	Sentences	English	Foreign
German	5.9M	121.0M	114.1M
Romanian	612.4K	15.9M	16.2M
Latvian	4.5M	66.8M	56.3M
Estonian	879.9K	22.7M	17.0M
Turkish	207.7K	5.1M	4.5M
Finnish	2.6M	61.1M	43.9M

Table 2: Training corpora statistics. Tokenization was carried out using the Moses tokenizer.

Of course, one could argue that we just substituted one parameter of the algorithm with another, which also has to be selected externally. However, as we will show in Section 5, the same value obtains nearly optimal results for most language arcs.

Another possibility that could be considered for defining the number of operations is to measure the evolution of the likelihood on an external development corpus, and stop the iterations when the likelihood decreases. We implemented this approach, but found that the likelihood on the development corpus increases monotonically for each new unit extracted (up to the maximum number we allowed for the experiments), and thus it does not provide a useful stopping criterion for the algorithm.

5 Experimental results

We conducted experiments for machine translation in a variety of languages, focusing on morphologically rich ones, using the data available from the latest WMT evaluation campaign where the language pair was used. We include results for Finish (Fi), German (De) [WMT'19], Estonian (Et), Turkish (Tr) [WMT'18], Latvian [WMT'17] and Romanian (Ro) [WMT'16], all paired with English (En) and for both translation directions. We used all available corpora for translation model training, except ParaCrawl. Corpora statistics can be found in Table 2. It can be seen that we experiment with a wide variety of corpus sizes, varying between 200K sentences up to nearly 6 million.

For BPE training, the corpora were subsampled to 1M sentences for BPE training[3], and a common BPE model was trained for the source and target languages (which also share the same embedding matrix). Experiments were carried out using Sockeye (Hieber et al., 2017) using mostly the default settings, except for a transformer architecture consisting of 20 encoder layers and 2 decoder layers (Hieber et al., 2020). The corpora were tokenized using the Moses tokenizer.

5.1 Analysis of BPE segmentation

We will start by focusing on the analysis of the produced sub-word units. Table 3 shows some differences between the statistical approach and the standard approach on words found in the German training data. The first example clearly shows how BPE does not use any linguistic information, even splitting the pair of characters 'ue', which is an alternative form of the letter 'ü'. In contrast, S-BPE produces a much more morphologically motivated split by separating the 's' at the end, which denotes genitive case. In the next two examples, S-BPE splits the words as derived forms of other words ('stehenden' and 'laeufige', respectively). In the last two examples, S-BPE correctly splits compound words into individual components. For none of these cases the standard BPE finds a linguistically satisfying sub-word decomposition. However note that although S-BPE improves over BPE, a more refined morphological splitting would still be possible for the last two examples.

Revisiting the examples of Table 1, we see that "beklagen" is now split into "be@@ kla@@ gen", and "bewertungsinstrumente" into "bewer@@ tungs@@ instrumente", which do not exactly correspond to the splitting points suggested in Section 1, but are more satisfactory than the BPE segmentation.

In order to quantify these improvements we use the data provided by the Morpho Challenge 2010 shared task (Kurimo et al., 2010). As part of the data of this evaluation, a morphological segmentation of words was provided for English, Finnish and Turkish. We applied the BPE and S-BPE models to the development dataset, and computed the F1-score of the produced segmentations, using the morphological segmentation as reference. For BPE segmentation, we selected the optimal segmentation as measured by the BLEU score on the translations of the WMT test data (see also Section 5.3). The results[4] are shown in Table 4. As English is a common language for all investigated language arcs, we provide results for the different language

[3] Experiments with the standard BPE training did not show any difference in performance between using the downsampled corpus or the full corpus.

[4] Note that these scores are for comparison of BPE and S-BPE only, and will be clearly outperformed by dedicated systems for the task.

Word	BPE	S-BPE
ungluecks	unglu@@ ecks	unglueck@@ s
anstehenden	anstehenden	an@@ stehenden
vorlaeufige	vorlaeufi@@ ge	vor@@ laeufige
gefangengenommen	gefan@@ gen@@ genommen	gefangen@@ genommen
finanzdienstleistungen	finanzdienstleistungen	finanz@@ dienstleistungen

Table 3: Segmentation examples of German words: S-BPE produces consistent segmentations of single and compound words, while BPE breaks in some cases the morphological structure of words.

Language	Arc	BPE	S-BPE
English	→ Fi	23.81	**24.68**
	→ De	**25.46**	24.82
	→ Ro	23.07	**26.96**
	→ Lv	20.84	**25.74**
	→ Et	20.83	**23.09**
	→ Tr	22.47	**25.67**
Finnish	→ En	12.14	**14.57**
Turkish	→ En	**23.00**	22.90

Table 4: Morpho Challenge results (F1 score).

pairs. It can be seem that S-BPE produces more linguistically motivated splits of English words in five out of six cases. For Finnish, S-BPE also produces better linguistic units, while for Turkish the F1 score is nearly identical. In light of these results we can affirm that in most cases S-BPE produces more linguistically motivated units than standard BPE.

5.2 Human evaluation

In the previous section we showed how S-BPE produces more linguistically motivated units. Of course, the main question is if these units help the system produce better translations. We hypothesize that S-BPE affects mainly single words, specially unknown words or words rarely seen in training (e.g. morphological variations of known words), and this effect is hardly captured by BLEU. Therefore we focus on human evaluation first and will present results with BLEU in the next section.

We carried out a human evaluation on English-German and English-Turkish (both directions) with a subset of test sentences where at least one unknown word was found. BLEU did not show significant differences between BPE and S-BPE on this subset of sentences. A blind test was carried out with 7 members of our department, all native speakers of Turkish (1) or German (6) and experts in NLP.

The evaluators were shown a source sentence, together with a highlighted word, and the output of the BPE and S-BPE systems. They had to answer two questions: which system produced a better translation of the highlighted word? And, which system produced a better translation of the sentence overall? Table 5 shows examples of the German-to-English test sentences highlighting the translations of the unknown German word inside the translations of the sentence, as produced with BPE and S-BPE. (For completeness we also show the segmentation of the unknown German word.)

The results of the human evaluation are shown on Table 6. It can be seen that when BPE and S-BPE produce different translations for the words being evaluated, in the majority of cases human graders prefer the translations produced with S-BPE. In particular, for language arcs involving German, the percentage of sentences for which translations based on S-BPE are preferred over translations based on BPE is 41.3% vs. 23.3% and 41.5% vs. 29.3%. These results are statistical significant (using a paired proportion test, with $p < 0.01$). It is known that German has a high lexical prolificity, with a high number of morphological variations as well as compound words. In fact, out of 2 000 sentences of the De→En test set 736 (36.8%) contain unknown words. These results confirm the superior generalization of S-BPE over BPE, both at the word and sentence levels.

For Turkish we also observe a preference for the S-BPE translations of unknown words, as well as a general preference for S-BPE sentences for English to Turkish translation, with no clear winner for the reverse direction. The statistical significance of these results is lower than for German, clearly due to the smaller amount of evaluated sentences.

5.3 Translation results

In this section we present global translation results, evaluated using BLEU scores. Table 7 compares

	Segmentation	Sentence
Source		Wegen der Umstellung auf den neuen Abgas- und **Verbrauchsprüfstandard** WLTP gebe es Produktionsausfälle bei Audi, sagte Schot der "Heilbronner Stimme".
Reference		After conversion to the new emissions and **consumption standard** WLTP, there were production losses at Audi, Schot told the 'Heilbronner Stimme'.
BPE	verbrau@@ ch@@ spru@@ ef@@ standard	Due to the changeover to the new exhaust and **exhaust test standard** WLTP there were production downs at Audi, said the "Heilbronner Stimme".
S-BPE	verbrauch@@ spruef@@ standard	Due to the changeover to the new WLTP exhaust and **consumption testing standard**, production was lost at Audi, Schot said "Heilbronner Voice".
Source		Es gibt keine **Abbiegespur** auf den Haaße-Hügel.
Reference		There is no **turning lane** on Haaße Hügel.
BPE	ab@@ bi@@ e@@ ges@@ pur	There is no **bending** on the Haasse Hill.
S-BPE	ab@@ bie@@ ge@@ spur	There is no **turning lane** on the Haasse hill.
Source		In der **Haushaltwarenabteilung** im Obergeschoss kippt der Geflügelte einen mit Espresso zubereiteten Cocktail namens "Golden Eye", passend zum Festival-Award.
Reference		In the **household goods department** on the upper floor, the winged man tips down a cocktail made with espresso called "Golden Eye", which is suited to the festival award.
BPE	haushalt@@ war@@ enab@@ teilung	In the **household section** on the upper floor, the poultry tick a cocktail prepared with espresso called "Golden Eye", in line with the festival award.
S-BPE	haushalt@@ waren@@ abteilung	In the **household goods department** on the upper floor the poultry tilts a cocktail prepared with espresso called "Golden Eye", matching the festival award.
Source		Der 46-jährige Fahrer des **Notarztautos** hatte am Samstagnachmittag mit Blaulicht und Martinshorn eine rote Ampel überfahren.
Reference		The 46 year old driver of the **ambulance** ran a red light on Saturday afternoon with the blue lights flashing and siren sounding.
BPE	not@@ arz@@ tau@@ tos	The 46-year-old driver of the **notary car** had passed a red light on Saturday afternoon with the blue light and Martinshorn.
S-BPE	no@@ tar@@ z@@ t@@ autos	The 46-year-old driver of the **emergency car** had overrun a red traffic light on Saturday afternoon with blue-light and Martinshorn.

Table 5: Translation examples showing the impact of morphologically wrong segmentation by BPE and how statistical BPE avoids such errors. Notice that the words causing the errors were not observed at training time.

	Better word		Better sentence	
Arc	BPE	S-BPE	BPE	S-BPE
En → De	10.0%	21.3%**	23.3%	41.3%**
De → En	17.1%	26.3%**	29.3%	41.5%**
En → Tr	11.8%	23.5%	11.8%	35.3%*
Tr → En	18.9%	39.6%*	30.2%	30.2%

Table 6: Results of the human evaluation. The numbers indicate the proportion of wins by each system (ties are omitted from the table for brevity). Evaluated sentences, in top-down order, were 150, 369, 34, and 53, respectively. Statistical significance, measured with a paired proportion test, is reported for $p < 0.01$ (**) and $p < 0.05$ (*).

the BLEU scores for the different language pairs using BPE for a range of sub-word unit numbers (from 4K to 96K). One first observation is that the number of units has an important effect on translation performance. We can see that the effect can be as much as 2 BLEU points (Et → En). The optimal number of operations also varies greatly between languages, with En → Fi obtaining optimum performance at 96K (although without much variability), while other arcs like e.g. En → Tr having the best performance at just 4K operations. If we conduct a similar grid search for S-BPE, we can draw similar conclusions about the optimal number of operations, noting that the effect of choosing an incorrect number operations is even more important. The full results can be found in the Appendix.

Table 7 also shows the results of using the stopping criterion described in Section 4.1, with stop-

	BPE								
Arc	4K	8K	16K	32K	48K	64K	96K	S-BPE (#ops)	
En → Fi	20.79	20.95	20.89	20.92	20.93	20.90	**21.08**	**20.92***	(7 269)
Fi → En	23.33	23.63	**23.71**	22.94	22.95	22.89	23.17	**23.86***	(7 719)
En → De	36.93	37.62	37.60	38.00	**38.38**	38.15	38.17	37.46	(5 864)
De → En	34.43	34.35	35.12	**35.22**	34.71	35.04	34.80	**34.84**	(5 704)
En → Ro	23.98	**24.08**	23.78	22.88	22.85	22.88	22.77	**23.92***	(7 169)
Ro → En	33.18	**33.45**	32.63	31.15	31.20	31.08	31.52	32.73	(7 709)
En → Lv	17.27	17.41	**17.72**	17.26	16.86	16.86	17.11	**17.35***	(6 819)
Lv → En	18.26	18.50	**18.59**	18.50	18.32	18.32	18.59	**18.65***	(7 334)
En → Et	**17.28**	16.90	16.83	15.98	15.92	15.71	16.17	**17.18***	(7 039)
Et → En	**22.17**	21.95	21.76	20.90	20.07	20.03	20.51	**22.06***	(7 464)
En → Tr	**13.00**	12.69	12.00	12.02	11.77	11.73	11.47	**12.89***	(8 384)
Tr → En	17.66	**17.85**	17.19	16.80	16.98	17.04	16.60	**17.84***	(9 114)

Table 7: Results for different language pairs. For BPE we use the number of operations given in the head of the table (4K, 8K, etc.), for S-BPE we use early stopping (with $k = 0.002$ and averaging the last 5 iterations). The symbol * marks systems for which S-BPE is not significantly different than the best BPE system. S-BPE results in bold are within ± 0.4 BLEU of the optimal BPE result.

ping parameter set to $k = 0.002$ and averaging over the last 5 iterations. These values were obtained empirically by doing a grid search over a small set of values and languages. It can be seen that the results obtained for most translation directions are in the range of the optimal result obtained by BPE, with many results not being statistical significantly different, as computed with the bootstrap method (Koehn, 2004), with 99% confidence interval. One can also consider that there is additional variability due to random initialization of the NMT optimization algorithm, in our experience in the range of ± 0.4 BLEU. We also marked the systems within this range in the table.[5]

It is also worth noting that for the language arcs where the stopping criterion is outperformed by the optimized baseline BPE extraction, the difference in performance is smaller than the difference due to choosing an incorrect number of operations on the standard BPE approach.

In conclusion, we do not see a clear difference in BLEU scores with S-BPE with respect to the standard BPE approach, using the optimal number of operations. However, as Sections 5.1 and 5.2 show, we obtain focused improvements on single words, which improves the translation quality as perceived by human judges.

6 Conclusions and future work

We introduced a statistical extension of BPE extraction. It introduces a well-founded objective for unit selection, which also allows the definition of a statistically motivated stopping criterion. Using this approach we achieve nearly optimal machine translation performance as measured with BLEU, while at the same time producing more linguistically motivated units. This leads to better translations of single words, which increases the translation quality as perceived by human judges, especially in the case of sentences containing unseen words. Using the stopping criterion we approximate the optimal selection of number of units, without the need to perform the costly optimization required by BPE, involving a full training-evaluation cycle for each tested number of operations.

Regarding future work, we observe that the probability distributions defined for our approach are closely related to those used for n-gram language models. Thus, smoothing methods can be applied, which can enhance the robustness of the method for unseen events, which opens a wide variety of possible extensions of this work.

The code is available from `https://github.com/amazon-research/statistical-byte-pair-encoding`.

[5] We did not do an extensive search for random initializations for this investigations due to the high number of experiments involved.

References

Duygu Ataman and Marcello Federico. 2018. An Evaluation of Two Vocabulary Reduction Methods for Neural Machine Translation. In *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Papers)*, pages 97–110, Boston, MA. Association for Machine Translation in the Americas.

Duygu Ataman, Matteo Negri, Marco Turchi, and Marcello Federico. 2017. Linguistically motivated vocabulary reduction for neural machine translation from Turkish to English. *The Prague Bulletin of Mathematical Linguistics*, 108(1):331–342.

Tamali Banerjee and Pushpak Bhattacharyya. 2018. Meaningless yet meaningful: Morphology grounded subword-level NMT. In *Proceedings of the Second Workshop on Subword/Character LEvel Models*, pages 55–60, New Orleans. Association for Computational Linguistics.

Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. 2019. Findings of the 2019 conference on machine translation (WMT19). In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 1–61, Florence, Italy. Association for Computational Linguistics.

Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22–29.

Michael Denkowski and Graham Neubig. 2017. Stronger baselines for trustable results in neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 18–27, Vancouver. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

Philip Gage. 1994. A new algorithm for data compression. *The C Users Journal*, 12(2):23–38.

James Henderson. 2020. The unstoppable rise of computational linguistics in deep learning. *arXiv:2005.06420 [cs]*.

Felix Hieber, Tobias Domhan, Michael Denkowski, and David Vilar. 2020. Sockeye 2: A toolkit for neural machine translation. *Proceedings of EAMT 2020, project track*.

Felix Hieber, Tobias Domhan, Michael Denkowski, David Vilar, Artem Sokolov, Ann Clifton, and Matt Post. 2017. Sockeye: A toolkit for neural machine translation. *ArXiv e-prints*, abs/1712.05690.

Matthias Huck, Simon Riess, and Alexander Fraser. 2017. Target-side word segmentation strategies for neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 56–67, Copenhagen, Denmark. Association for Computational Linguistics.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

Julia Kreutzer and Artem Sokolov. 2018. Learning to segment inputs for NMT favors character-level processing. In *Proceedings of the 15th International Workshop on Spoken Language Translation*, pages 166–171, Bruges, Belgium.

Taku Kudo. 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. Association for Computational Linguistics.

Mikko Kurimo, Sami Virpioja, Ville T Turunen, et al. 2010. Proceedings of the Morpho Challenge 2010 Workshop. In *Morpho Challenge Workshop; 2010; Espoo*. Aalto University School of Science and Technology.

Dominik Macháček, Jonáš Vidra, and Ondřej Bojar. 2018. Morphological and language-agnostic word segmentation for NMT. In *International Conference on Text, Speech, and Dialogue*, pages 277–284. Springer.

J. Niehues, R. Cattoni, S. Stüker, M. Negri, M. Turchi, T. Ha, E. Salesky, R. Sanabria, L. Barrault, L. Specia, and M. Federico. 2019. The IWSLT 2019 Evaluation Campaign. In *Proceedings of the 16th International Workshop on Spoken Language Translation (IWSLT 2019)*, Hong Kong, China.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. 2019. BPE-Dropout: Simple and Effective Subword Regularization. *arXiv:1910.13267 [cs]*.

Víctor M. Sánchez-Cartagena and Antonio Toral. 2016. Abu-MaTran at WMT 2016 translation task: Deep learning, morphological segmentation and tuning on character sequences. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 362–370, Berlin, Germany. Association for Computational Linguistics.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152. IEEE.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. 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. Association for Computational Linguistics.

Rico Sennrich and Biao Zhang. 2019. Revisiting low-resource neural machine translation: A case study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 211–221, Florence, Italy. Association for Computational Linguistics.

Sami Virpioja, Peter Smit, Stig-Arne Grönroos, and Mikko Kurimo. 2013. Morfessor 2.0: Python implementation and extensions for morfessor baseline. D4 julkaistu kehittämis- tai tutkimusraportti tai -selvitys.

Appendix

A Full derivation of likelihood increase

Lemma Given a, b, c such that $a > 0$, $b > a$ and $0 < c < b$ we have:

$$\frac{a-c}{b-c} < \frac{a}{b}. \tag{1}$$

Proof. By assumption denominators are positive, hence we can rearrange (1) as: $b(a-c) < a(b-c)$. By assumption, $a(b-c) < ab$ from which we get $b(a-c) < ab$ and $(a-c) < a$ which is true by assumption.□

Define the count of a sub-word unit $s \in \Sigma$ for a corpus S and a sub-word vocabulary Σ as $C_{S,\Sigma}(s)$. The likelihood function is then defined as

$$L(S, \Sigma) = \sum_{s \in \Sigma} C_{S,\Sigma}(s) \log p(s) \tag{2}$$

We are interested in the increase in likelihood at step n

$$\Delta L_n(S) = L(S, \Sigma_n) - L(S, \Sigma_{n-1}) \,. \tag{3}$$

When adding a new rule $\langle (x, y) \to xy \rangle$ in step n of the algorithm, thus defining Σ_n, we can express the likelihood increase as[1]

$$\begin{aligned} \Delta L_n(S) = & \sum_{s \in \Sigma_{n-1} \setminus \{x,y\}} \Big(C_{S,\Sigma_n}(s) \log p_n(s) - C_{S,\Sigma_{n-1}}(s) \log p_{n-1}(s) \Big) \\ & + \sum_{s \in \{x,y\}} \Big(C_{S,\Sigma_n}(s) \log p_n(s) - C_{S,\Sigma_{n-1}}(s) \log p_{n-1}(s) \Big) \\ & + C_{S,\Sigma_n}(xy) \log p_n(xy) \end{aligned} \tag{4}$$

We note that for $s \in \Sigma_{n-1} \setminus \{x, y\}$

$$C_{S,\Sigma_n}(s) = C_{S,\Sigma_{n-1}}(s) \quad \text{and} \quad p_n(s) > p_{n-1}(s) \tag{5}$$

as the total number of observations (denominator of p_n) shrinks after combining two symbols. Thus, for the first term in equation 4 we have

$$\sum_{s \in \Sigma_{n-1} \setminus \{x,y\}} \Big(C_{S,\Sigma_n}(s) \log p_n(s) - C_{S,\Sigma_{n-1}}(s) \log p_{n-1}(s) \Big) > 0 \,. \tag{6}$$

This quantity is expected to be small, specially when the number of produced symbols increases.

Next, let us note that for the counts of the units involved in the new rule, we have

$$\begin{aligned} C_{S,\Sigma_n}(x) = C_{S,\Sigma_{n-1}}(x) - C_{\Sigma_n}(xy) \\ C_{S,\Sigma_n}(y) = C_{S,\Sigma_{n-1}}(y) - C_{\Sigma_n}(xy) \end{aligned} \tag{7}$$

(the equation holds for both x and y because the $C_{\Sigma_n}(xy)$ is added to the total amount of units).

For the probability of x and y we are reducing the occurrences and the total number of events by the same positive amount, which is lower that the sample size. Hence, by subtracting the same counts from the sample size and from the previous Lemma we can derive:

$$\begin{aligned} p_n(x) = \frac{C_{S,\Sigma_n}(x)}{C_{S,\Sigma_n}(\cdot)} &= \frac{C_{S,\Sigma_{n-1}}(x) - C_{S,\Sigma_n}(xy)}{C_{S,\Sigma_{n-1}}(\cdot) - C_{S,\Sigma_n}(xy)} \\ &< \frac{C_{S,\Sigma_{n-1}}(x)}{C_{S,\Sigma_{n-1}}(\cdot)} = p_{n-1}(x) \end{aligned} \tag{8}$$

[1] As some counts may decrease to 0 when defining a new pair, we use the convention $0 \log 0 = 0$.

and similarly for y.

Using (6) and (8) in (4) we obtain

$$\begin{aligned}\Delta L_n(S) > \sum_{s\in\{x,y\}} &\Big(C_{S,\Sigma_n}(s)\log p_n(s) - C_{S,\Sigma_{n-1}}(s)\log p_n(s)\Big) \\ &+ C_{S,\Sigma_n}(xy)\log p_n(xy)\end{aligned} \tag{9}$$

and using the count relations from (7) we arrive at

$$\Delta L_n(S) > C_{S,\Sigma_n}(xy)\left[\log p_n(xy) - \log p_n(x) - \log p_n(y)\right] \ . \tag{10}$$

B Additional S-BPE results

(a) English-to-German

# ops	BPE	S-BPE
4K	36.93	36.47
8K	37.62	37.11
16K	37.60	37.04
32K	38.00	37.59
48K	**38.38**	36.71
64K	38.15	**37.90**
96K	38.17	37.88

(b) German-to-English

# ops	BPE	S-BPE
4K	34.43	33.78
8K	34.35	34.38
16K	35.12	**35.56**
32K	**35.22**	34.84
48K	34.71	35.18
64K	35.04	35.17
96K	34.80	34.60

(c) English-to-Romanian

# ops	BPE	S-BPE
4K	23.98	24.09
8K	**24.08**	**24.16**
16K	23.78	23.64
32K	22.88	22.14
48K	22.85	19.09
64K	22.88	17.85
96K	22.77	16.85

(d) Romanian-to-English

# ops	BPE	S-BPE
4K	33.18	**32.82**
8K	**33.45**	32.80
16K	32.63	32.70
32K	31.15	29.63
48K	31.20	24.78
64K	31.08	22.80
96K	31.52	21.58

(e) English-to-Latvian

# ops	BPE	S-BPE
4K	17.27	16.86
8K	17.41	17.11
16K	**17.72**	**17.26**
32K	17.26	17.26
48K	16.86	16.86
64K	16.86	16.86
96K	17.11	17.11

(f) Latvian-to-English

# ops	BPE	S-BPE
4K	18.26	18.32
8K	18.50	**18.59**
16K	**18.59**	18.50
32K	18.50	18.33
48K	18.32	18.32
64K	18.32	18.32
96K	18.59	18.59

(g) English-to-Estonian

# ops	BPE	S-BPE
4K	**17.28**	**17.62**
8K	16.90	17.26
16K	16.83	17.07
32K	15.98	15.91
48K	15.92	14.21
64K	15.71	12.32
96K	16.17	11.09

(h) Estonian-to-English

# ops	BPE	S-BPE
4K	**22.17**	**21.91**
8K	21.95	21.79
16K	21.76	21.83
32K	20.90	20.80
48K	20.07	17.95
64K	20.03	15.58
96K	20.51	13.58

(i) English-to-Turkish

# ops	BPE	S-BPE
4K	**13.00**	**13.28**
8K	12.69	12.93
16K	12.00	12.05
32K	12.02	7.62
48K	11.77	6.30
64K	11.73	5.75
96K	11.47	5.25

(j) Turkish-to-English

# ops	BPE	S-BPE
4K	17.66	17.72
8K	**17.85**	**17.87**
16K	17.19	17.25
32K	16.80	11.83
48K	16.98	9.19
64K	17.04	8.24
96K	16.60	7.61

(k) English-to-Finnish

# ops	BPE	S-BPE
4K	20.79	20.60
8K	20.95	**21.03**
16K	20.89	20.82
32K	20.92	20.56
48K	20.93	21.00
64K	20.90	20.13
96K	**21.08**	20.63

(l) Finnish-to-English

# ops	BPE	S-BPE
4K	23.33	23.57
8K	23.63	**23.75**
16K	**23.71**	23.49
32K	22.94	23.05
48K	22.95	22.92
64K	22.89	21.95
96K	23.17	20.21

Table 7: Translation results for different language pairs with BPE and S-BPE, varying the number of operations. In bold, the best result for each language arc.