Dynamic Programming Encoding for Subword Segmentation in Neural Machine Translation

Xuanli He Monash University
Gholamreza Haffari Monash University
{xuanli.he1,gholamreza.haffari}@monash.edu
Mohammad Norouzi Google Research
mnorouzi@google.com

Abstract

This paper introduces Dynamic Programming Encoding (DPE), a new segmentation algorithm for tokenizing sentences into subword units. We view the subword segmentation of output sentences as a latent variable that should be marginalized out for learning and inference. A mixed character-subword transformer is proposed, which enables exact log marginal likelihood estimation and exact MAP inference to find target segmentations with maximum posterior probability. DPE uses a lightweight mixed character-subword transformer as a means of pre-processing parallel data to segment output sentences using dynamic programming. Empirical results on machine translation suggest that DPE is effective for segmenting output sentences and can be combined with BPE dropout for stochastic segmentation of source sentences. DPE achieves an average improvement of 0.9 BLEU over BPE (Sennrich et al., 2016) and an average improvement of 0.55 BLEU over BPE dropout (Provilkov et al., 2019) on several WMT datasets including English $\leftrightarrow$ (German, Romanian, Estonian, Finnish, Hungarian).

1 Introduction

The segmentation of rare words into subword units (Sennrich et al., 2016; Wu et al., 2016) has become a critical component of neural machine translation (Vaswani et al., 2017) and natural language understanding (Devlin et al., 2019). Subword units enable *open vocabulary* text processing with a negligible pre-processing cost and help maintain a desirable balance between the vocabulary size and decoding speed. Since subword vocabularies are built in an unsupervised manner (Sennrich et al., 2016; Wu et al., 2016), they are easily applicable to any language.

Given a fixed vocabulary of subword units, rare words can be segmented into a sequence of subword units in different ways. For instance, “un+conscious” and “uncon+scious” are both suitable segmentations for the word “unconscious”. This paper studies the impact of subword segmentation on neural machine translation, given a fixed subword vocabulary, and presents a new algorithm called *Dynamic Programming Encoding (DPE)*.

We identify three families of subword segmentation algorithms in neural machine translation:

1. Greedy algorithms: Wu et al. (2016) segment words by recursively selecting the longest subword prefix. Sennrich et al. (2016) recursively combine adjacent word fragments that co-occur most frequently, starting from characters.
2. Stochastic algorithms (Kudo, 2018; Provilkov et al., 2019) draw multiple segmentations for source and target sequences resorting to randomization to improve robustness and generalization of translation models.
3. Dynamic programming algorithms, studied here, enable exact marginalization of subword segmentations for certain sequence models.

We view the subword segmentation of output sentences in machine translation as a latent variable that should be marginalized out to obtain the probability of the output sentence given the input. On the other hand, the segmentation of source sentences can be thought of as input features and can be randomized as a form of data augmentation to improve translation robustness and generalization. Unlike previous work, we recommend using two distinct segmentation algorithms for tokenizing source and target sentences: stochastic segmentation for source and dynamic programming for target sentences.

We present a new family of mixed character-subword transformers, for which simple dynamic programming algorithms exist for exact marginalization and MAP inference of subword segmentations. The time complexity of the dynamic programming algorithms is $O(TV)$, where T is the length of the target sentence in characters, and V is the size of the subword vocabulary. By comparison, even computing the conditional probabilities of subword units in an autoregressive model requires $O(TV)$ to estimate the normalizing constant of the categorical distributions. Thus, our dynamic programming algorithm does not incur additional asymptotic costs. We use a lightweight mixed character-subword transformer as a means to pre-process translation datasets to segment output sentences using DPE for MAP inference.

The performance of a standard subword transformer (Vaswani et al., 2017) trained on WMT datasets tokenized using DPE is compared against Byte Pair Encoding (BPE) (Sennrich et al., 2016) and BPE dropout (Provilkov et al., 2019). Empirical results on English $\leftrightarrow$ (German, Romanian, Estonian, Finnish, Hungarian) suggest that stochastic subword segmentation is effective for tokenizing source sentences, whereas deterministic DPE is superior for segmenting target sentences. DPE achieves an average improvement of 0.9 BLEU over greedy BPE (Sennrich et al., 2016) and an average improvement of 0.55 BLEU over stochastic BPE dropout (Provilkov et al., 2019)[1].

2 Related Work

Neural networks have revolutionized machine translation (Sutskever et al., 2014; Bahdanau et al., 2015; Cho et al., 2014). Early neural machine translation (NMT) systems used words as the atomic element of sentences. They used vocabularies with tens of thousands words, resulting in prohibitive training and inference complexity. While learning can be sped up using sampling techniques (Jean et al., 2015), word based NMT models have a difficult time handling rare words, especially in morphologically rich languages such as Romanian, Estonian, and Finnish. The size of the word vocabulary should increase dramatically to capture the compositionality of morphemes in such languages.

More recently, many NMT models have been developed based on characters and a combination of characters and words (Ling et al., 2015; Luong and Manning, 2016; Vylomova et al., 2017; Lee et al., 2017; Cherry et al., 2018). Fully character based models (Lee et al., 2017; Cherry et al., 2018) demonstrate a significant improvement over word based models on morphologically rich languages. Nevertheless, owing to the lack of morphological information, deeper models are often required to obtain a good translation quality. Moreover, elongated sequences brought by a character representation drastically increases the inference latency.

In order to maintain a good balance between the vocabulary size and decoding speed, subword units are introduced in NMT (Sennrich et al., 2016; Wu et al., 2016). These segmentation approaches are data-driven and unsupervised. Therefore, with a negligible pre-processing overhead, subword models can be applied to any NLP task (Vaswani et al., 2017; Devlin et al., 2019). Meanwhile, since subword vocabularies are generated based on word frequencies, only the rare words are split into subword units and common words remain intact.

Previous work (Chan et al., 2016; Kudo, 2018) has explored the idea of using stochastic subword segmentation with multiple subword candidates to approximate the log marginal likelihood. Kudo (2018) observed marginal gains in translation quality at the cost of introducing additional hyper-parameters and complex sampling procedures. We utilize BPE dropout (Provilkov et al., 2019), a simple stochastic segmentation algorithm for tokenizing source sentences.

Dynamic programming has been used to marginalize out latent segmentations for speech recognition (Wang et al., 2017), showing a consistent improvement over greedy segmentation methods. In addition, dynamic programming has been successfully applied to learning sequence models by optimizing edit distance (Sabour et al., 2018) and aligning source and target sequences (Chan et al., 2020; Saharia et al., 2020). We show the effectiveness of dynamic programming for segmenting output sentences in NMT using a mixed character-transformer in a pre-processing step.

3 Latent Subword Segmentation

Let $\boldsymbol{x}$ denote a source sentence and $\boldsymbol{y} = (y_1, \ldots, y_T)$ denote a target sentence comprising T characters. The goal of machine translation is to learn a conditional distribution $p(\boldsymbol{y} \mid \boldsymbol{x})$ from a large corpus of source-target sentences. State-of-the-art neural machine translation systems make use of a dictionary of subword units to tokenize the target sentences in a more succinct way as a sequence of $M \leq T$ subword units. Given a subword vocabulary, there are multiple ways to segment a

[1]code and corpora: https://github.com/xlhex/dpe

rare word into a sequence of subwords (see Figure 1). The common practice in neural machine translation considers subword segmentation as a pre-process and uses greedy algorithms to segment each word across a translation corpus in a consistent way. This paper aims to find optimal subword segmentations for the task of machine translation.

Let $\boldsymbol{z} = (z_1, .., z_{M+1})$ denote a sequence of character indices $0 = z_1 < z_2 < \ldots < z_M < z_{M+1} = T$ in an ascending order, defining the boundary of M subword segments $\{\boldsymbol{y}_{z_i, z_{i+1}}\}_{i=1}^{M}$. Let $\boldsymbol{y}_{a,b} \equiv [y_{a+1}, \ldots, y_b]$ denote a subword that spans the segment between $(a+1)^{\text{th}}$ and b^{th} characters, including the boundary characters. For example, given a subword dictionary {'c', 'a', 't', 'at', 'ca'}, the word 'cat' may be segmented using $\boldsymbol{z} = (0, 1, 3)$ as ('c', 'at'), or using $\boldsymbol{z} = (0, 2, 3)$ as ('ca', 't'), or using $\boldsymbol{z} = (0, 1, 2, 3)$ as ('c', 'a', 't'). The segmentation $\boldsymbol{z} = (0, 3)$ is not valid since 'cat' does not appear in the subword vocabulary.

Autoregressive language models create a categorical distribution over the subword vocabulary at every subword position and represent the log-probability of a subword sequence using chain rule,

$$\log p(\boldsymbol{y}, \boldsymbol{z}) = \sum_{i=1}^{|\boldsymbol{z}|} \log p(\boldsymbol{y}_{z_i, z_{i+1}} \mid \boldsymbol{y}_{z_1, z_2}, \ldots, \boldsymbol{y}_{z_{i-1}, z_i}) \,. \tag{1}$$

Note that we suppress the dependence of p on $\boldsymbol{x}$ to reduce notational clutter. Most neural machine translation approaches assume that $\boldsymbol{z}$ is a deterministic function of $\boldsymbol{y}$ and implicitly assume that $\log p(\boldsymbol{y}, \boldsymbol{z}) \approx \log p(\boldsymbol{y})$.

We consider a subword segmentation $\boldsymbol{z}$ as a latent variable and let each value of $\boldsymbol{z} \in \mathcal{Z}_y$, in the set of segmentations compatible with $\boldsymbol{y}$, contribute its share to $p(\boldsymbol{y})$ according to $p(\boldsymbol{y}) = \sum_{\boldsymbol{z}} p(\boldsymbol{y}, \boldsymbol{z})$,

$$\log p(\boldsymbol{y}) = \log \sum_{\boldsymbol{z} \in \mathcal{Z}_y} \exp \sum_{i=1}^{|\boldsymbol{z}|} \log p(\boldsymbol{y}_{z_i, z_{i+1}} \mid \ldots, \boldsymbol{y}_{z_{i-1}, z_i}) \,. \tag{2}$$

Note that each particular subword segmentation $\boldsymbol{z} \in \mathcal{Z}_y$ provides a lower bound on the log marginal likelihood $\log p(\boldsymbol{y}) \geq \log p(\boldsymbol{y}, \boldsymbol{z})$. Hence, optimizing (1) for a greedily selected segmentation can be justified as a lower bound on (2). That said, optimizing (2) directly is more desirable. Unfortunately, exact marginalization over all segmentations is computationally prohibitive in a combinatorially large space $\mathcal{Z}_y$, especially because the probability of each subword depends on the segmentation of its conditioning context. In the next section, we discuss a sequence model in which the segmentation of the conditioning context does not influence the probability of the next subword. We describe an efficient Dynamic Programming algorithm to exactly marginalize out all possible subword segmentations in this model.

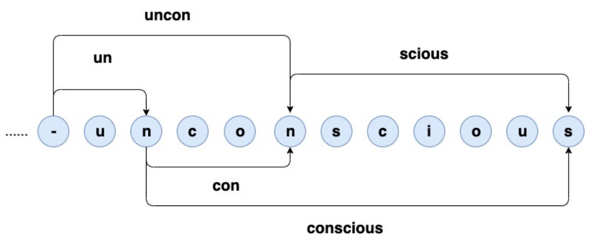

Figure 1: An illustration of different ways of segmenting 'unconscious' into subword units.

4 A Mixed Character-Subword Transformer

We propose a mixed character-subword transformer architecture, which enables one to marginalize out latent subword segmentations exactly using dynamic programming (see Figure 2). Our key insight is to let the transformer architecture process the inputs and the conditioning context based on characters to remain oblivious to the specific choice of subword segmentation in the conditioning context and enable exact marginalization. That said, the output of the transformer is based on subword units and at every position it creates a categorical distribution over the subword vocabulary. More precisely, when generating a subword $\boldsymbol{y}_{z_i, z_{i+1}}$, the model processes the conditioning context $(y_{z_1}, \ldots, y_{z_i})$ based solely on characters using,

$$\log p(\boldsymbol{y}, \boldsymbol{z}) = \sum_{i=1}^{|\boldsymbol{z}|} \log p(\boldsymbol{y}_{z_i, z_{i+1}} \mid y_{z_1}, ..., y_{z_i}) \,, \tag{3}$$

where the dependence of p on $\boldsymbol{x}$ is suppressed to reduce notational clutter.

Given a fixed subword vocabulary denoted V, at every character position t within $\boldsymbol{y}$, the mixed character-subword model induces a distribution over the next subword $w \in V$ based on,

$$p(w \mid y_1, .., y_t) = \frac{\exp(f(y_1, .., y_t)^\top e(w))}{\sum_{w' \in V} \exp(f(y_1, .., y_t)^\top e(w'))},$$

where $f(\cdot)$ processes the conditioning context using a Transformer, and $e(\cdot)$ represents the weights

Algorithm 1 Dynamic Programming (DP) for Exact Marginalization

Input: $\boldsymbol{y}$ is a sequence of T characters, V is a subword vocabulary, m is the maximum subword length
Output: $\log p(\boldsymbol{y})$ marginalizing out different subword segmentations.

```
1: $\alpha_0 \leftarrow 0$
2: for $k = 1$ to $T$ do
3:     $\alpha_k \leftarrow \log \sum_{j=k-m}^{k-1} \mathbb{1}[\boldsymbol{y}_{j,k} \in V] \exp\left(\alpha_j + \log P_\theta(\boldsymbol{y}_{j,k}|y_1,..,y_j)\right)$
4: end for
5: return $\alpha_T$        ▷ the marginal probability $\log p(\boldsymbol{y}) = \log \sum_{\boldsymbol{z} \in \mathcal{Z}_y} p(\boldsymbol{y}, \boldsymbol{z})$
```

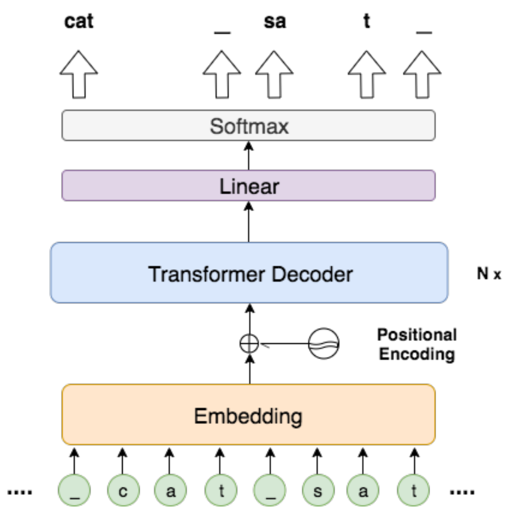

Figure 2: An illustration of the mixed character-subword Transformer. The input is a list of characters, whereas the output is a sequence of subwords.

of the softmax layer.

As depicted in in Figure 2, the mixed character-subword Transformer consumes characters as input generates subwords as output. This figure only shows the decoder architecture, since as the encoder that processes $\boldsymbol{x}$ is a standard subword Transformer. Once a subword w is emitted at time step t, the characters of the subword w are fed into the decoder for time steps $t+1$ to $t+|w|$, and the next subword is generated at time step $t+|w|$, conditioned on all of the previously generated characters.

4.1 Optimization

The training objective for our latent segmentation translation model is $\sum_{(\boldsymbol{x},\boldsymbol{y}) \in \mathcal{D}} \log P_\theta(\boldsymbol{y}|\boldsymbol{x})$ where $\mathcal{D}$ is the training corpus consisting of parallel bilingual sentence pairs. Maximizing the training objective requires marginalization and the computation of the gradient of the log marginal likelihood.

Exact Marginalization. Under our model, the probability of a subword only depends on the character-based encoding of the conditioning context and not its segmentation, as in (3). This means that we can compute the log marginal likelihood for a single example $\boldsymbol{y}$, exactly, using the Dynamic Programming algorithm shown in Algorithm 1. The core of the algorithm is line 4, where the probability of the prefix string $\boldsymbol{y}_{0,k}$ is computed by summing terms corresponding to different segmentations. Each term consists of the product of the probability of a subword $\boldsymbol{y}_{j,k}$ times the probability of its conditioning context $(y_1, \ldots, y_j)$. The running time of the algorithm is $\mathcal{O}(mT)$, where T is the length of the string, and m is the size of the longest subword unit in the vocabulary.

Gradient Computation. We use automatic differentiation in PyTorch to backpropagate through the dynamic program in Algorithm 1 and compute its gradient. Compared to a standard Transformer decoder, our mixed character-subword Transformer is 8x slower with a larger memory footprint, due to computation involved in the DP algorithm and large sequence length in characters. To address these issues, we reduce the number of transformer layers from 6 to 4, and accumulate 16 consecutive gradients before one update.

4.2 Segmenting Target Sentences

Once the mixed character-subword transformer is trained, it is used to segment the target side of a bilingual corpus. We randomize the subword segmentation of source sentences using BPE dropout (Provilkov et al., 2019). Conditional on the source sentence, we use Algorithm 2, called Dynamic Programming Encoding (DPE) to find a segmentation of the target sentence with highest posterior probability. This algorithm is similar to the marginalization algorithm, where we use a max operation instead of log-sum-exp. The mixed character-subword transformer is used only for tokenization, and a standard subword transformer is trained on the segmented sentences. For inference using beam search, the mixed character-subword

Algorithm 2 Dynamic Programming Encoding (DPE) for Subword Segmentation

```
Input: $\boldsymbol{y}$ is a sequence of $T$ characters, $V$ is a subword vocabulary, $m$ is the maximum subword length
Output: Segmentation $\boldsymbol{z}$ with highest posterior probability.
  for $k = 1$ to $T$ do
    $\beta_k \leftarrow \max_{\{j \in [k-m, k-1] \,|\, \boldsymbol{y}_{j,k} \in V\}} \beta_j + \log P_\theta(\boldsymbol{y}_{j,k} | y_1, .., y_j)$
    $b_k \leftarrow \text{argmax}_{\{j \in [k-m, k-1] \,|\, \boldsymbol{y}_{j,k} \in V\}} \beta_j + \log P_\theta(\boldsymbol{y}_{j,k} | y_1, .., y_j)$
  end for
  $\boldsymbol{z} \leftarrow \text{backtrace}(b_1, .., b_T)$        ▷ backtrace the best segmentation using $\boldsymbol{b}$
```

	Number of sentences			Vocab
	train	dev	test	size
En-Hu WMT09	0.6M	2,051	2,525	32K
En-De WMT14	4.2M	3000	3003	32K
En-Fi WMT15	1.7M	1,500	1,370	32K
En-Ro WMT16	0.6M	1,999	1,999	32K
En-Et WMT18	1.9M	2,000	2,000	32K

Table 1: Statistics of the corpora.

transformer is not needed.

5 Experiments

Dataset We use WMT09 for En-Hu, WMT14 for En-De, WMT15 for En-Fi, WMT16 for En-Ro and WMT18 for En-Et. We utilize Moses toolkit[2] to pre-process all corpora, and preserve the true case of the text. Unlike Lee et al. (2018), we retain diacritics for En-Ro to retain the morphological richness. We use all of the sentence pairs where the length of either side is less than 80 tokens for. training. Byte pair encoding (BPE) (Sennrich et al., 2016) is applied to all language pairs to construct a subword vocabulary and provide a baseline segmentation algorithm. The statistics of all corpora is summarized in Table 1.

Training with BPE Dropout. We apply BPE dropout (Provilkov et al., 2019) to each mini-batch. For each complete word, during the BPE merge operation, we randomly drop a particular merge with a probability of 0.05. This value worked the best in our experiments. A word can be split into different segmentations at the training stage, which helps improve the BPE baseline.

DPE Segmentation. DPE can be used for target sentences, but its use for source sentences is not justified as source segmentations should not be marginalized out. Accordingly, we use BPE

[2]https://github.com/moses-smt/mosesdecoder

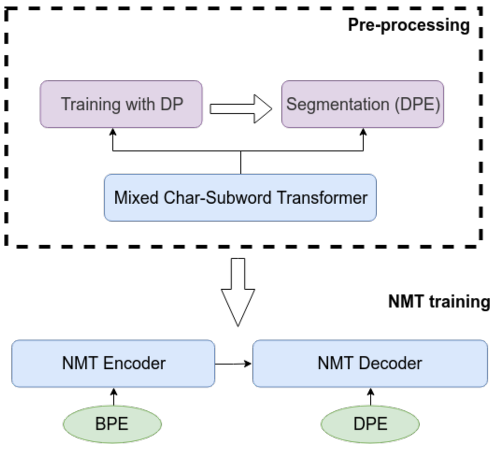

Figure 3: The workflow of the proposed DPE approach.

dropout for segmenting source sentences. That is, we train a mixed character-subword transformer to marginalize out the latent segmentations of a target sentence, given a randomized segmentation of the source sentence by BPE dropout. After the mixed character-subword transformer is trained, it is used to segment the target sentences as describe in section 4.2 for tokenization.

As summarized in Figure 3, we first train a mixed character-subword transformer with dynamic programming. Then, this model is frozen and used for DPE segmentation of target sentences. Finally, a standard subword transformer is trained on source sentences segmented by BPE dropout and target sentences segmented by DPE. The mixed character-subword transformer is not needed for inference.

Transformer Architectures. We use transformer models to train three translation models on BPE, BPE dropout, and DPE corpora. We make use of *transformer base* for all of the experiments.

5.1 Main Results

Table 2 shows the main results. First, we see that BPE dropout consistently outperforms BPE across language pairs. In Table 2, the column labeled to Δ_1 shows the improvement of BPE dropout over

Method	BPE	BPE dropout	Δ_1	This paper	Δ_2
Source segmentation	BPE	BPE dropout		BPE dropout	
Target segmentation	BPE	BPE dropout		DPE	
En→De	27.11	27.27	+0.16	27.61	+0.34
En→Ro	27.90	28.07	+0.17	28.66	+0.59
En→Et	17.64	18.20	+0.56	18.80	+0.60
En→Fi	15.88	16.18	+0.30	16.89	+0.71
En→Hu	12.80	12.94	+0.14	13.36	+0.42
De→En	30.82	30.85	+0.03	31.21	+0.36
Ro→En	31.67	32.56	+0.89	32.99	+0.43
Et→En	23.13	23.65	+0.52	24.62	+0.97
Fi→En	19.10	19.34	+0.24	19.87	+0.53
Hu→En	16.14	16.61	+0.47	17.05	+0.44
Average	22.22	22.57	+0.35	23.12	+0.55

Table 2: Average test BLEU scores (averaged over 3 independent runs) for 3 segmentation algorithms (BPE (Sennrich et al., 2016), BPE dropout (Provilkov et al., 2019) and our DPE algorithm) on 8 different WMT datasets. For each language pair, all of the segmentation techniques use the same subword dictionary with 32K tokens shared between source and target languages. Δ_1 shows the improvement of BPE dropout compared to BPE, and Δ_2 shows further improvement of our proposed DPE method compared to BPE dropout.

BPE source: Die G@@ le@@ is@@ anlage war so ausgestattet , dass dort elektr@@ isch betrie@@ bene Wagen eingesetzt werden konnten . DPE target: The railway system was equipped in such a way that electrical@@ ly powered cart@@ s could be used on it . BPE target: The railway system was equipped in such a way that elect@@ r@@ ically powered car@@ ts could be used on it .
BPE source: Normalerweise wird Kok@@ ain in kleineren Mengen und nicht durch Tunnel geschm@@ ug@@ gelt . DPE target: Normal@@ ly c@@ oca@@ ine is sm@@ ugg@@ led in smaller quantities and not through tunnel@@ s . BPE target: Norm@@ ally co@@ c@@ aine is sm@@ ugg@@ led in smaller quantities and not through tun@@ nels .

Table 3: Two examples of segmentation of English sentences given German inputs.

BPE. This gain can be attributed to the robustness of the NMT model to the segmentation error on the source side, as our analysis in Section 5.3 will confirm. Second, we observe further gains resulted from DPE compared to BPE dropout. The column labeled Δ_2 shows the improvement of DPE over BPE dropout. DPE provides an average improvement of 0.55 BLEU over BPE dropout and BPE dropout provides an average improvement of 0.35 BLEU over BPE. As our proposal uses BPE dropout for segmenting the source, we attribute our BLEU score improvements to a better segmentation of the target language with DPE. Finally, compared to BPE for segmenting the source and target, our proposed segmentation method results in large improvements in the translation quality, up to 1.49 BLEU score improvements in Et→En.

5.2 Segmentation Examples

Table 3 shows examples of target sentences segmented using DPE and BPE and the corresponding source sentences. In addition, Table 4 presents the top 50 most common English words that result in a disagreement between BPE and DPE segmentations based on the Et→En corpus. For DPE, for each word, we consider all segmentations produced and show the segmentation that attains the highest frequency of usage in Table 4. As can be observed, DPE produces more linguistically plausible morpheme-based subwords compared to BPE. For instance, BPE segments *"carts"* into *"car"*+*"ts"*, as both *"car"* and *"ts"* are common subwords and listed in the BPE merge table. By contrast DPE segments *"carts"* into *"cart"*+*"s"*. We attribute the linguistic characteristics of the DPE segments to the fact that DPE conditions the segmentation of a target word on the source sentence and the previous tokens of the target sentence, as opposed to BPE, which mainly makes use of frequency of subwords, without any context.

DPE generally identifies and leverages some linguistic properties, *e.g.,* plural, antonym, normalization, verb tenses, etc. However, BPE tends to deliver less linguistically plausible segmentations, possibly due to its greedy nature and the lack of context. We believe this phenomenon needs further investigation, *i.e.,* the contribution of source vs. target context in DPE segmentations, and a quantitative evaluation of linguistic nature of word fragments produced by DPE. We will leave this to future work.

5.3 Analysis

Conditional Subword Segmentation. One of our hypothesis for the effectiveness of subword segmentation with DPE is that it conditions the segmentation of the target on the source language. To verify this hypothesis, we train mixed character-subword Transformer solely on the target language sentences in the bilingual training corpus using the *language model* training objective. This is in contrast to the mixed character-subword model used in the DPE segmentation of the main results in Table 2, where the model is conditioned on the source language and trained on the sentence pairs using a *conditional language model* training objective. Once the mixed character-subword Transformer language model is trained, it is then used to segment the target sentence of the bilingual corpus in the pre-processing step before a translation model is trained.

Table 5 shows the results. It compares the unconditional language model (LM) DPE vs the conditional DPE for segmenting the target language, where we use BPE dropout for segmenting the source language. We observe that without the information from the source, LM DPE is on-par to BPE, and is significantly outperformed by conditional DPE. This observation confirms our hypothesis that segmentation in NMT should be source-dependent.

We are further interested in analyzing the differences of the target language segmentation depending on the source language. For this analysis,

BPE	DPE (ours)
recognises	recognise + s
advocates	advocate + s
eurozone	euro + zone
underlines	underline + s
strengthens	strengthen + s
entrepreneurship	entrepreneur + ship
acknowledges	acknowledge + s
11.30	11 + .30
wines	wine + s
pres + ently	present + ly
f + illed	fill + ed
endors + ement	endorse + ment
blo + c	bl + oc
cru + cially	crucial + ly
eval + uations	evaluation + s
tre + es	tr + ees
tick + ets	tick + et + s
predic + table	predict + able
multilater + alism	multilateral + ism
rat + ings	rating + s
predic + ted	predict + ed
mo + tives	motiv + es
reinfor + ces	reinforce + s
pro + tocols	protocol + s
pro + gressively	progressive + ly
sk + ill	ski + ll
preva + ils	prevail + s
decent + ralisation	decent + ral + isation
sto + red	stor + ed
influ + enz + a	influen + za
margin + alised	marginal + ised
12.00	12 + .00
sta + ying	stay + ing
intens + ity	intensi + ty
rec + ast	re + cast
guid + eline	guide + line
emb + arked	embark + ed
out + lines	outline + s
scen + ari + os	scenario + s
n + ative	na + tive
ma + ture	ma + ture
preven + tative	prevent + ative
hom + eland	home + land
bat + hing	bath + ing
endang + ered	endanger + ed
cont + inen + tal	continent + al
t + enth	ten + th
vul + n + era + bility	vul + ner + ability
realis + ing	real + ising
t + ighter	tight + er

Table 4: Word fragments obtained by BPE vs. DPE. The most frequent words that resulted in a disagreement between BPE and DPE segmentations on $Et \rightarrow En$ are shown.

we filtered out a multilingual parallel corpus from WMT, which contains parallel sentences in three languages English, Estonian and Romanian. That is, for each English sentence we have the corresponding sentences in Et and Ro. We then trained two DPE segmentation models for the translation tasks of Et→En and Ro→En, where English is the target language. Figure 4 shows when conditioning

Source Target	BPE drop BPE drop	BPE drop LM DPE	BPE drop DPE
En→Ro	28.07	28.07	28.66
En→Hu	12.94	12.87	13.36
Ro→En	32.56	32.57	32.99
Hu→En	16.61	16.41	17.05

Table 5: DPE-LM learns a segmentation of the target based on language modelling, which is *not* conditioned on the source language.

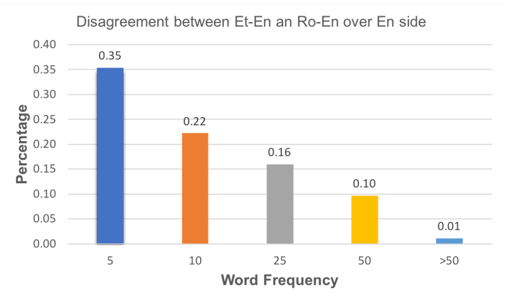

Figure 4: Disagreement of DPE segments between Et-En and Ro-En over English vocabulary

the segmentation of the target on different source languages, DPE can lead to different segmentations even for an identical multi-parallel corpus. The differences are more significant for low frequency words.

Another aspect of DPE segmentation method is its dependency on the *segmentation* of the source. As mentioned, we segment the target sentence *on the fly* using our mixed character-subword model *given* a randomized segmentation of the source produced by BPE dropout. That means during the training of the NMT model where we use BPE dropout for the source sentence, the corresponding target sentence may get a different DPE segmentation given the randomized segmentation of the source sentence. We are interested in the effectiveness of the target segmentation if we commit to a fixed DPE segmentation conditioned on the BPE segmentation of the input. Table 6 shows the results. We observe that there is a marginal drop when using the fixed DPE, which indicates that the encoder can benefit from a stochastic segmentation, while the decoder prefers a deterministic segmentation corresponding to the segmentation of the source.

DPE vs BPE. We are interested to compare the effectiveness of DPE versus BPE for the target, given BPE dropout as the same segmentation

Source Target	BPE drop DPE Fixed	BPE drop DPE On The Fly
En→Ro	28.58	28.66
En→Hu	13.14	13.36
En→Et	18.51	18.80
Ro→En	32.73	32.99
Hu→En	16.82	17.05
Et→En	24.37	24.62

Table 6: "DPE Fixed" obtains a fixed segmentation of the target sentence given the BPE-segmented source sentence, whereas "DPE On The Fly" obtain the best segmentation of the target sentence given a randomized segmentation of the source produced by BPE dropout.

Source Target	BPE drop BPE	BPE drop BPE drop	BPE drop DPE
En→Ro	28.04	28.07	28.66
En→Et	18.09	18.20	18.80
Ro→En	32.40	32.56	32.99
Et→En	23.52	23.65	24.62

Table 7: BLEU score of different segmentation methods for the target.

method for the source. Table 7 shows the results. As observed, target segmentation with DPE consistently outperforms BPE, leading to up to .9 BLEU score improvements. We further note that using BPE dropout on the target has a similar performance to BPE, and it is consistently outperformed by DPE.

We further analyze the segmentations produced by DPE vs BPE. Figure 5 shows the percentage of the target words which have different segmentation with BPE and DPE, for different word frequency bands in En→Et translation task. We observe that for Estonian words whose occurrence is up to 5 in the training set, the disagreement rate between DPE and BPE is 64%. The disagreement rate decreases as we go to words in higher frequency bands. This may imply that the main difference between the relatively large BLEU score difference between BPE and DPE is due to their different segmentation mainly for low-frequency words.

We further plot the distribution of BLEU scores by the length of target sentences. As shown in Figure 6, DPE demonstrates much better gains on the longer sentences, compared with the BPE version.

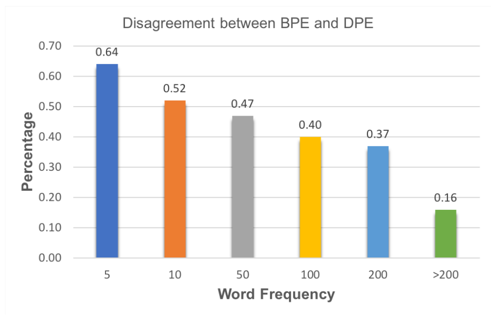

Figure 5: Disagreement of segments between BPE and DPE over Estonian vocabulary.

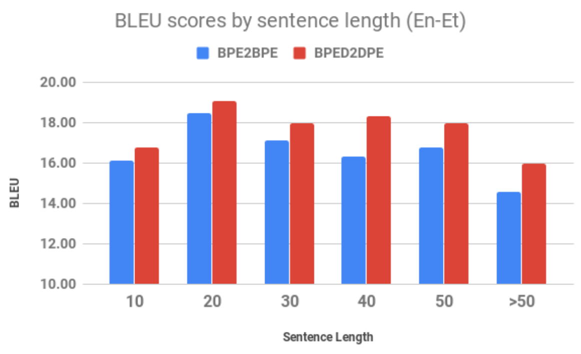

Figure 6: BLEU scores of BPE vs DPE by the lengths of sentences for En→Et.

6 Conclusion

This paper introduces *Dynamic Programming Encoding* in order to incorporate the information of the source language into subword segmentation of the target language. Our approach utilizes dynamic programming for marginalizing the latent segmentations when training, and inferring the highest probability segmentation when tokenizing. Our comprehensive experiments show impressive improvements compared to state-of-the-art segmentation methods in NMT, *i.e.,* BPE and its stochastic variant BPE dropout.

7 Acknowledgement

We would like to thank the anonymous reviewers, Taku Kudo, Colin Cherry and George Foster for their comments and suggestions on this work. The computational resources of this work are supported by the Google Cloud Platform (GCP), and by the Multi-modal Australian ScienceS Imaging and Visualisation Environment (MASSIVE) (www.massive.org.au). This material is partly based on research sponsored by Air Force Research Laboratory and DARPA under agreement number FA8750-19-2-0501. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *ICLR*.

William Chan, Chitwan Saharia, Geoffrey Hinton, Mohammad Norouzi, and Navdeep Jaitly. 2020. Imputer: Sequence modelling via imputation and dynamic programming. *arXiv:2002.08926*.

William Chan, Yu Zhang, Quoc Le, and Navdeep Jaitly. 2016. Latent sequence decompositions. *arXiv preprint arXiv:1610.03035*.

Colin Cherry, George Foster, Ankur Bapna, Orhan Firat, and Wolfgang Macherey. 2018. Revisiting character-based neural machine translation with capacity and compression. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4295–4305.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. On using very large target vocabulary for neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1–10.

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.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2017. Fully character-level neural machine translation without explicit segmentation. *Transactions of the Association for Computational Linguistics*, 5:365–378.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1173–1182.

Wang Ling, Chris Dyer, Alan W Black, Isabel Trancoso, Ramón Fermandez, Silvio Amir, Luis Marujo, and Tiago Luís. 2015. Finding function in form: Compositional character models for open vocabulary word representation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1520–1530.

Minh-Thang Luong and Christopher D Manning. 2016. Achieving open vocabulary neural machine translation with hybrid word-character models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1054–1063.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. 2019. Bpe-dropout: Simple and effective subword regularization. *arXiv:1910.13267*.

Sara Sabour, William Chan, and Mohammad Norouzi. 2018. Optimal completion distillation for sequence learning. *arXiv preprint arXiv:1810.01398*.

Chitwan Saharia, William Chan, Saurabh Saxena, and Mohammad Norouzi. 2020. Non-autoregressive machine translation with latent alignments. *arXiv:2004.07437*.

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)*, volume 1, pages 1715–1725.

I Sutskever, O Vinyals, and QV Le. 2014. Sequence to sequence learning with neural networks. *NIPS*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *NIPS*.

Ekaterina Vylomova, Trevor Cohn, Xuanli He, and Gholamreza Haffari. 2017. Word representation models for morphologically rich languages in neural machine translation. *Proceedings of the First Workshop on Subword and Character Level Models in NLP*.

Chong Wang, Yining Wang, Po-Sen Huang, Abdelrahman Mohamed, Dengyong Zhou, and Li Deng. 2017. Sequence modeling via segmentations. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3674–3683. JMLR. org.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv:1609.08144*.