Controlling Utterance Length in NMT-based Word Segmentation with Attention

Pierre Godard, Laurent Besacier, François Yvon

To cite this version:

Pierre Godard, Laurent Besacier, François Yvon. Controlling Utterance Length in NMT-based Word Segmentation with Attention. International Workshop on Spoken Language Translation, Nov 2019, Hong-Kong, China. hal-02343206

HAL Id: hal-02343206
https://hal.archives-ouvertes.fr/hal-02343206
Submitted on 2 Nov 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers. L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Controlling Utterance Length in NMT-based Word Segmentation with Attention

Pierre Godard\textsuperscript{1,2}, Laurent Besacier\textsuperscript{2}, François Yvon\textsuperscript{1}

\textsuperscript{1}LIMSI, CNRS, Université Paris-Saclay, F 91 405 Orsay
\textsuperscript{2}LIG, CNRS et Université Grenoble Alpes, F 38 000 Grenoble
pierre.godard@limsi.fr, laurent.besacier@uga.fr, francois.yvon@limsi.fr

Abstract

One of the basic tasks of computational language documentation (CLD) is to identify word boundaries in an unsegmented phonemic stream. While several unsupervised monolingual word segmentation algorithms exist in the literature, they are challenged in real-world CLD settings by the small amount of available data. A possible remedy is to take advantage of glosses or translation in a foreign, well-resourced language, which often exist for such data. In this paper, we explore and compare ways to exploit neural machine translation models to perform unsupervised boundary detection with bilingual information, notably introducing a new loss function for jointly learning alignment and segmentation. We experiment with an actual under-resourced language, Mboshi, and show that these techniques can effectively control the output segmentation length.

1. Introduction

All over the world, languages are disappearing at an unprecedented rate, fostering the need for specific tools aimed to aid field linguists to collect, transcribe, analyze, and annotate endangered language data (e.g. [1, 2]). A remarkable effort in this direction has improved the data collection procedures and tools [3, 4], enabling to collect corpora for an increasing number of endangered languages (e.g. [5]).

One of the basic tasks of computational language documentation (CLD) is to identify word or morpheme boundaries in an unsegmented phonemic or orthographic stream. Several unsupervised monolingual word segmentation algorithms exist in the literature, based, for instance, on information-theoretic [6, 7] or nonparametric Bayesian techniques [8, 9]. These techniques are, however, challenged in real-world settings by the small amount of available data.

A possible remedy is to take advantage of glosses or translations in a foreign, well-resourced language (WL), showing that bilingual information can be used as a noisy proxy to word alignment [15].\textsuperscript{1}

In this paper, we explore ways to exploit neural machine translation models to perform unsupervised boundary detection with bilingual information. Our main contribution is a new loss function for jointly learning alignment and segmentation in neural translation models, allowing us to better control the length of utterances. Our experiments with an actual under-resourced language (UL), Mboshi [18], show that this technique outperforms our bilingual segmentation baseline.

2. Recurrent architectures in NMT

In this section, we briefly review the main concepts of recent architectures for machine translation introduced in [19, 20, 21].\textsuperscript{2} In our setting, the source and target sentences are always observed and we are mostly interested in the attention mechanism that is used to induce word segmentation.

2.1. RNN encoder-decoder

Sequence-to-sequence models transform a variable-length source sequence into a variable-length target output sequence. In our context, the source sequence is a sequence of words $w_1, \ldots, w_J$ and the target sequence is an unsegmented sequence of phonemes or characters $\omega_1, \ldots, \omega_I$. In the RNN encoder-decoder architecture, an encoder consisting of a RNN reads a sequence of word embeddings $e(w_1), \ldots, e(w_J)$ representing the source and produces a dense representation $c$ of this sentence in a low-dimensional vector space. Vector $c$ is then fed to an RNN decoder producing the output translation $\omega_1, \ldots, \omega_I$ sequentially.

At each step of the input sequence, the encoder hidden states $h_j$ are computed as:

$$h_j = \phi(e(w_j), h_{j-1}).$$  \hspace{1cm} (1)

In most cases, $\phi$ corresponds to a long short-term memory (LSTM) [25] unit or a gated recurrent unit (GRU) [26], and $h_J$ is used as the fixed-length context vector $c$ initializing the RNN decoder.

On the target side, the decoder predicts each word $\omega_i$, given the context vector $c$ (in the simplest case, $h_J$, the last

\textsuperscript{1}This assumption is further discussed in [16] and [17].

\textsuperscript{2}The Transformer model of [22] arguably yields better translation performance, but the underlying soft alignments provided by the multi-head, multi-layered attention mechanism are harder to exploit [23, 24].
hidden state of the encoder) and the previously predicted words, using the probability distribution over the output vocabulary $V_T$:

$$
P(\omega | \omega_1, \ldots, \omega_{i-1}, c) = g(\omega_{i-1}, s_i, c)$$

$$\omega_i = \arg\max_{\omega_i} P(\omega | \omega_1, \ldots, \omega_{i-1}, c),$$

where $s_i$ is the hidden state of the decoder RNN and $g$ is a nonlinear function (e.g., a multi-layer perceptron with a softmax layer) computed by the output layer of the decoder. The hidden state $s_i$ is then updated according to:

$$s_i = f(s_{i-1}, c(\omega_{i-1}), c),$$

where $f$ again corresponds to the function computed by an LSTM or GRU cell.

The encoder and the decoder are trained jointly to maximize the likelihood of the translation $\Omega = \Omega_1, \ldots, \Omega_T$ given the source sentence $w = w_1, \ldots, w_T$. As reference target words are available during training, $\Omega$ (and the corresponding embedding) can be used instead of $\omega_i$ in Equations (2) and (3), a technique known as teacher forcing [27].

### 2.2. The attention mechanism

Encoding a variable-length source sentence in a fixed-length vector can lead to poor translation results with long sentences [20]. To address this problem, [21] introduces an attention mechanism which provides a flexible source context to better inform the decoder’s decisions. This means that the fixed context vector $c$ in Equations (2) and (3) is replaced with a position-dependent context $c_i$, defined as:

$$c_i = \sum_{j=1}^{J} \alpha_{ij} h_j,$$

where weights $\alpha_{ij}$ are computed by an attention model made of a multi-layer perceptron (MLP) followed by a softmax layer. Denoting $a$ the function computed by the MLP, then

$$
\begin{align*}
\alpha_{ij} &= \frac{\exp(e_{ij})}{\sum_{k=1}^{K} \exp(e_{ik})}, \\
\sum_{j} \alpha_{ij} &= 1
\end{align*}
$$

where $e_{ij}$ is known as the energy associated to $\alpha_{ij}$. Lines in the attention matrix $A = (\alpha_{ij})$ sum to 1, and weights $\alpha_{ij}$ can be interpreted as the probability that target word $\omega_i$ is aligned to source word $w_j$. [21] qualitatively investigated such soft alignments and concluded that their model can correctly align target words to relevant source words (see also [28, 29]). Our segmentation method (Section 3) relies on the assumption that the same holds when aligning characters or phonemes on the target side to source words.

### 3. Attention-based word segmentation

Recall that our goal is to discover words in an unsegmented stream of target characters (or phonemes) in the under-resourced language. In this section, we first describe a baseline method inspired by the “align to segment” of [13, 14]. We then propose two extensions providing the model with a signal relevant to the segmentation process, so as to move towards a joint learning of segmentation and alignment.

#### 3.1. Align to segment

An attention matrix $A = (\alpha_{ij})$ can be interpreted as a soft alignment matrix between target and source units, where each cell $\alpha_{ij}$ corresponds to the probability for target symbols $\omega_i$ (here, a phone) to be aligned to the source word $w_j$ (cf. Equation (5)). In our context, where words need to be discovered on the target side, we follow [13, 14] and perform word segmentation as follows:

1. train an attentional RNN encoder-decoder model with attention using teacher forcing (see Section 2);
2. force-decode the entire corpus and extract one attention matrix for each sentence pair.
3. identify boundaries in the target sequences. For each target unit $\omega_i$ of the UL, we identify the source word $w_{i'}$ to which it is most likely aligned: $\forall i, \omega_i = \arg\max_{j} \alpha_{ij}$. Given these alignment links, a word segmentation is computed by introducing a word boundary in the target whenever two adjacent units are not aligned with the same source word ($\omega_i \neq \omega_{i+1}$).

Considering a (simulated) low-resource setting, and building on [15]’s work, [12] propose to smooth attentional alignments, either by post-processing attention matrices, or by flattening the softmax function in the attention model (see Equation (5)) with a temperature parameter $T$. This makes sense as the authors examine attentional alignments obtained while training from UL phonemes to WL words. But when translating from WL words to UL characters, this seems less useful: smoothing will encourage a character to align to many words. This technique is further explored by [30], who make the temperature parameter trainable and specific to each decoding step, so that the model can learn how to control the softness or sharpness of attention distributions, depending on the current word being decoded.

#### 3.2. Towards joint alignment and segmentation

One limitation in the approach described above lies in the absence of signal relative to segmentation during RNN training. Attempting to move towards a joint learning of alignment and segmentation, we propose here two extensions:

---

3 As discussed below, teacher forcing will also be used at “test” time in our scenario, since we are not training these models for a translation task, but a word segmentation task.

4 In this case, $\alpha_{ij} = \frac{\exp(h_{ij}/T)}{\sum_{k=1}^{K} \exp(h_{ik}/T)}$.

5 A temperature below 1 would conversely sharpen the alignment distribution. We did not observe significant changes in segmentation performance varying the temperature parameter.
aimed at introducing constraints derived from our segmentation heuristic in the training process.

3.2.1. Word-length bias

Our first extension relies on the assumption that the length of aligned source and target words should correlate. Being in a relationship of mutual translation, aligned words are expected to have comparable frequencies and meaning, hence comparable lengths. This means that the longer a source word is, the more target units should be aligned to it. We implement this idea in the attention mechanism as a word-length bias, changing the computation of the context vector from Equation (4) to:

\[ c_i = \sum_j \psi(|w_j|) \alpha_{ij} h_j \]  

where \( \psi \) is a monotonically increasing function of the length \( |w_j| \) of word \( w_j \). This will encourage target units to attend more to longer source words. In practice, we choose \( \psi \) to be the identity function and renormalize so as to ensure that lines still sum to 1 in the attention matrices. The context vectors \( c_i \) are now computed with attention weights \( \tilde{\alpha}_{ij} \) as:

\[ \begin{cases} 
\tilde{\alpha}_{ij} = \frac{|w_j|}{\sum_j |w_j|} \alpha_{ij} \\
\alpha_i = \sum_j \tilde{\alpha}_{ij} h_j
\end{cases} \]  

We finally derive the target segmentation from the attention matrix \( A = (\tilde{\alpha}_{ij}) \), following the method of Section 3.1.

3.2.2. Introducing an auxiliary loss function

Another way to inject segmentation awareness inside our training procedure is to control the number of target words that will be produced during post-processing. The intuition here is that notwithstanding typological discrepancies, the target segmentation should yield a number of target words that is close to the length of the source.

To this end, we complement the main loss function with an additional term \( \mathcal{L}_{AUX} \) defined as:

\[ \mathcal{L}_{AUX}(\Omega | w) = |I - J - \sum_{i=1}^{I-1} \alpha_{i,s} \alpha_{i+1,s}| \]  

The rationale behind this additional term is as follows: recall that a boundary is then inserted on the target side whenever two consecutive units are not aligned to the same source word. The dot product between consecutive lines in the attention matrix will be close to 1 if consecutive target units are aligned to the same source word, and closer to 0 if they are not. The summation thus quantifies the number of target units that will not be followed by a word boundary after segmentation, and \( I - \sum_{i=1}^{I-1} \alpha_{i,s} \alpha_{i+1,s} \) measures the number of word boundaries that are produced on the target side. Minimizing this auxiliary term should guide the model towards learning attention matrices resulting in target segmentations that have the same number of words on the source and target sides.

Figure 1 illustrates the effect of our auxiliary loss on an example. Without auxiliary loss, the segmentation will yield, in this case, 8 target segments (Figure 1a), while the attention learnt with auxiliary loss will yield 5 target segments (Figure 1b); source sentence, on the other hand, has 4 tokens.

4. Experiments and discussion

In this section, we describe implementation details for our baseline segmentation system and for the extensions proposed in Section 3.2, before presenting data and results.

4.1. Implementation details

Our baseline system is our own reimplementation of Bahdanau’s encoder-decoder with attention in PyTorch [32]. The last version of our code, which handles mini-batches efficiently, heavily borrows from Joost Basting’s code. Source sentences include an end-of-sentence (EOS) symbol (corresponding to \( w_J \) in our notation) and target sentences include both a beginning-of-sentence (BOS) and an EOS symbol. Padding of source and target sentences in mini-batches is required, as well as masking in the attention matrices and during loss computation. Our architecture follows [21] very closely with some minor changes.

---

6 Zipf’s “Law of Abbreviation”, a language universal, states that frequent words tend to be short. Additionally, experimental psychology also correlates word length and conceptual complexity [31].

7 Arguably, this constraint should better be enforced at the level of morphemes, instead of words.

---

8 We count here the end-of-sentence token corresponding to the last column in the attention matrices.


10 https://pytorch.org/. We use version 0.4.1.
Encoder We use a single-layer bidirectional RNN [33] with GRU cells: these have been shown to perform similarly to LSTM-based RNNs [34], while computationally more efficient. We use 64-dimensional hidden states for the forward and backward RNNs, and for the embeddings, similarly to [13, 14]. In Equation (1), \( h_j \) corresponds to the concatenation of the forward and backward states for each step \( j \) of the source sequence.

Attention The alignment MLP model computes function \( a \) from Equation (5) as \( a(s_{i-1}, h_j) = v^a_i \tanh(W_a s_{i-1} + U_a h_j) \) – see Appendix A.1.2 in [21] – where \( v_a, W_a, \) and \( U_a \) are weight matrices. For the computation of weights \( \alpha_{ij} \) in the word-length bias extension (Equation (7)), we arbitrarily attribute a length of 1 to the EOS symbol on the source side.

Decoder The decoder is initialized using the last backward state of the encoder and a non-linear function (tanh) for state \( s_0 \). We use a single-layer GRU RNN; hidden states and output embeddings are 64-dimensional. In preliminary experiments, and as in [35], we observed better segmentations adopting a “generate first” approach during decoding, where we first generate the current target word, then update the current RNN state. Equations (2) and (3) are accordingly modified into:

\[
\begin{align*}
P(\omega_i | \omega_1, \ldots, \omega_{i-1}, i) &= g(\omega_{i-1}, s_{i-1}, i) \\
s_i &= f(s_{i-1}, \omega_i, i).
\end{align*}
\]

During training and forced decoding, the hidden state \( s_i \) is thus updated using ground-truth embeddings \( e(\Omega_i) \). \( \Omega_0 \) is the BOS symbol. Our implementation of the output layer (\( g \)) consists of an MLP and a softmax.

Training We train for 800 epochs on the whole corpus with Adam (the learning rate is 0.001). Parameters are updated after each mini-batch of 64 sentence pairs.\(^{11}\) A dropout layer [36] is applied to both source and target embedding layers, with a rate of 0.5.\(^{12}\) The weights in all linear layers are initialized with Glorot’s normalized method (Equation (16) in [37]) and bias vectors are initialized to 0. Embeddings are initialized with the normal distribution \( \mathcal{N}(0, 0.1) \).\(^{13}\) Except for the bridge between the encoder and the decoder, the initialization of RNN weights is kept to PyTorch defaults. During training, we minimize the NLL loss \( \mathcal{L}_{\text{NLL}} \) (see Section 2.1), adding optionally the auxiliary loss \( \mathcal{L}_{\text{AUX}} \) (Section 3.2.2). When the auxiliary loss term is used, we schedule it to be integrated progressively so as to avoid degenerate solutions\(^{14}\) with coefficient \( \lambda_{\text{AUX}}(k) \) at epoch \( k \) defined by:

\[
\lambda_{\text{AUX}}(k) = \frac{\max(k - W)}{K}
\]

where \( K \) is the total number of epochs and \( W \) a wait parameter. The complete loss at epoch \( k \) is thus \( \mathcal{L}_{\text{NLL}} + \lambda_{\text{AUX}} \cdot \mathcal{L}_{\text{AUX}} \). After trying values ranging from 100 to 700, we set \( W \) to 200. We approximate the absolute value in Equation (8) by \( |x| \equiv \sqrt{x^2 + 0.001} \) in order to make the auxiliary loss function differentiable.

4.2. Data and evaluation

Our experiments are performed on an actual endangered language, Mboshi (Bantu C25), a language spoken in Congo-Brazzaville, using the bilingual French-Mboshi 5K corpus of [18]. On the Mboshi side, we consider alphabetic representation with no tonal information. On the French side, we simply consider the default segmentation into words.\(^{15}\)

We denote the baseline segmentation system as BASE, the word-length bias extension as BIAS, and the auxiliary loss extensions as AUX. We also report results for a variant of AUX (AUX+RATIO), in which the auxiliary loss is computed with a factor corresponding to the true length ratio \( r_{\text{MB/FR}} \) between Mboshi and French averaged over the first 100 sentences.\(^{16}\) In this variant, the auxiliary loss is computed as \( I - r_{\text{MB/FR}} \cdot J - \sum_{i=0}^{I-1} c_{i+1}^T \alpha_{i+1, i} \).

We report segmentation performance using precision, recall, and F-measure on boundaries (BP, BR, BF), and tokens (WP, WR, WF). We also report the exact-match (X) metric which computes the proportion of correctly segmented utterances.\(^{17}\) Our main results are in Figure 2, where we report averaged scores over 10 runs. As a comparison with another bilingual method inspired by the “align to segment” approach, we also include the results obtained using the statistical models of [10], denoted Pisa, in Table 1.

<table>
<thead>
<tr>
<th>BP</th>
<th>BR</th>
<th>BF</th>
<th>WP</th>
<th>WR</th>
<th>WF</th>
<th>X</th>
</tr>
</thead>
<tbody>
<tr>
<td>46.18</td>
<td>18.31</td>
<td>26.22</td>
<td>17.73</td>
<td>8.82</td>
<td>11.78</td>
<td>0.97</td>
</tr>
</tbody>
</table>

Table 1: Equivalent segmentation results with Pisa [10].

---

\(^{11}\)Mini-batches are created anew through shuffling and length-sorting at each epoch.

\(^{12}\)We also tried to add a dropout layer after the encoder and decoder RNNs (only for the “output” state, not the state values used inside the recursion) but this harmed our segmentation results.

\(^{13}\)This seemed to slightly improve segmentation results when compared to Glorot’s normalized method.

\(^{14}\)When the NLL loss has not “shaped” yet the attention matrices into soft alignments, the auxiliary loss can lead to trivial optimization solutions, in which a single column in the attention matrices has a certain number of weights set to 1 (to reach the proper value in the sum term from Equation (8)), while all other weights in the matrices are zeroed. The model is subsequently unable to escape this solution.

\(^{15}\)Several other units were also considered (lemmas, morphs, part-of-speech tags) but they did not lead to better performance (results not reported here). Investigation of subword units such as byte pair encodings (BPE) is left for future work.

\(^{16}\)This is a plausible supervision in the CLD scenario, and enables to relax assumption that number of words should be the same on target and source.

\(^{17}\)The exact-match metric includes single-word utterances.
4.3. Discussion

A first observation is that our baseline method BASE improves vastly over Pisa’s results (by a margin of about 30% on boundary F-measure, BF).

4.3.1. Effects of the word-length bias

The integration of a word-bias in the attention mechanism seems detrimental to segmentation performance, and results obtained with BIAS are lower than those obtained with BASE, except for the sentence exact-match metric (X). To assess whether the introduction of word-length bias actually encourages target units to “attend more” to longer source word in BIAS, we compute the correlation between the length of source word and the quantity of attention these words receive (for each source position, we sum attention column-wise: \( \sum_i \tilde{a}_{ij} \)). Results for all segmentation methods are in Table 2. BIAS increases the correlation between word lengths and attention, but this correlation being already high for all methods (BASE, or AUX and AUX+RATIO), our attempt to increase it proves here detrimental to segmentation.

<table>
<thead>
<tr>
<th>Method</th>
<th>BASE</th>
<th>BIAS</th>
<th>AUX</th>
<th>AUX+RATIO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correlation</td>
<td>0.681</td>
<td>0.729</td>
<td>0.665</td>
<td>0.662</td>
</tr>
</tbody>
</table>

Table 2: Correlation (avg. over 10 runs) between word length and attention (p-value for Pearson coefficient is 0 for each run) for methods BASE, BIAS, AUX, and AUX+RATIO.

4.3.2. Effects of the auxiliary loss

For boundary F-measures (BF) in Figure 2, AUX performs similarly to BASE, but with a much higher precision, and de-
Table 3: Statistics on segmentations produced by methods BASE, BIAS, AUX, and AUX+RATIO, on the Mboshi 5K corpus: number of tokens, types, average token length (in characters), average sentence lengths (in tokens), averaged over 10 runs.

<table>
<thead>
<tr>
<th>method</th>
<th>#tokens</th>
<th>#types</th>
<th>avg. tok. len</th>
<th>avg. sent. len</th>
</tr>
</thead>
<tbody>
<tr>
<td>BASE</td>
<td>40.7K</td>
<td>8.2K</td>
<td>3.14</td>
<td>7.94</td>
</tr>
<tr>
<td>BIAS</td>
<td>39.7K</td>
<td>8.9K</td>
<td>3.22</td>
<td>7.75</td>
</tr>
<tr>
<td>AUX</td>
<td>32.3K</td>
<td>9.1K</td>
<td>3.95</td>
<td>6.31</td>
</tr>
<tr>
<td>AUX+RATIO</td>
<td>28.6K</td>
<td>9.6K</td>
<td>4.47</td>
<td>5.58</td>
</tr>
<tr>
<td>ground-truth</td>
<td>30.6K</td>
<td>5.3K</td>
<td>4.19</td>
<td>5.96</td>
</tr>
</tbody>
</table>

graded recall, indicating that the new method does not over-segment as much as BASE. More insight can be gained from various statistics on the automatically segmented data presented in Table 3. The average token and sentence lengths for AUX are closer to their ground-truth values (resp. 4.19 characters and 5.96 words). The global number of tokens produced is also brought closer to its reference. On token metrics, a similar effect is observed, but the trade-off between a lower recall and an increased precision is more favorable and yields more than 3 points in F-measure. These results are encouraging for documentary purposes, where precision is arguably a more valuable metric than recall in a semi-supervised segmentation scenario.

They, however, rely on a crude heuristic that the source and target sides (here French and Mboshi) should have the same number of units, which are only valid for typologically related languages and not very accurate for our dataset.

As Mboshi is more agglutinative than French (5.96 words per sentence on average in the Mboshi 5K, vs. 8.22 for French), we also consider the lightly supervised setting where the true length ratio is provided. This again turns out to be detrimental to performance, except for the boundary precision (BP) and the sentence exact-match (X). Note also that precision becomes stronger than recall for both boundary and token metrics, indicating under-segmentation. This is confirmed by an average token length that exceeds the ground-truth (and an average sentence length below the true value, see Table 3).

Here again, our control of the target length proves effective: compared to BASE, the auxiliary loss has the effect to decrease the average sentence length and move it closer to its observed value (5.96), yielding an increased precision, an effect that is amplified with AUX+RATIO. By tuning this ratio, it is expected that we could even get slightly better results.

5. Related work

The attention mechanism introduced by [21] has been further explored by many researchers. [38], for instance, compare a global to a local approach for attention, and examine several architectures to compute alignment weights $\alpha_{ij}$. [39] additionally propose a recurrent version of the attention mechanism, where a “dynamic memory” keeps track of the attention received by each source word, and demonstrate better translation results. A more general formulation of the attention mechanism can, lastly, be found in [40], where structural dependencies between source units can be modeled.

With the goal of improving alignment quality, [41] computes a distance between attentions and word alignments learnt with the reparameterization of IBM Model 2 from [42]; this distance is then added to the cost function during training. To improve alignments also, [15] introduce several refinements to the attention mechanism, in the form of structural biases common in word-based alignment models. In this work, the attention model is enriched with features able to control positional bias, fertility, or symmetry in the alignments, which leads to better translations for some language pairs, under low-resource conditions. More work seeking to improve alignment and translation quality can be found in [43, 44, 45, 46, 47, 48].

Another important line of research related to work studies the relationship between segmentation and alignment quality: it is recognized that sub-lexical units such as BPE [49] help solve the unknown word problem; other notable works around these lines include [50] and [51].

CLD has also attracted a growing interest in recent years. Most recent work includes speech-to-text translation [52, 53], speech transcription using bilingual supervision [54], both speech transcription and translation [55], or automatic phonemic transcription of tonal languages [56].

6. Conclusion

In this paper, we explored neural segmentation methods extending the “align to segment” approach, and proposed extensions to move towards joint segmentation and alignment. This involved the introduction of a word-length bias in the attention mechanism and the design of an auxiliary loss. The latter approach yielded improvements over the baseline on all accounts, in particular for the precision metric.

Our results, however, lag behind the best monolingual performance for this dataset (see e.g. [57]). This might be due to the difficulty of computing valid alignments between phonemes and words in very limited data conditions, which remains very challenging, as also demonstrated by the results of Pisa. However, unlike monolingual methods, bilingual methods generate word alignments and their real benefit should be assessed with alignment based metrics. This is left for future work, as reference word alignments are not yet available for our data.

Other extensions of this work will focus on ways to mitigate data sparsity with weak supervision information, either by using lists of frequent words or the presence of certain word boundaries on the target side or by using more sophisticated attention models in the spirit of [15] or [40].
7. References


A. Appendix
Figure 3: Statistics on segmentations produced by methods BASE, BIAS, AUX, and AUX+RATIO, on the Mboshi 5K corpus: number of tokens, types, average token length (in characters), average sentence lengths (in tokens). Solid (teal-colored) lines correspond to average values (10 runs). Dashed (red) lines indicate the ground-truth values in the Mboshi 5K corpus.