Functional Lexicon in Subword Tokenization

Zachary William Hopton[1], Yves Scherrer[2,3], Tanja Samardzic[1]

[1]Language and Space Lab, University of Zurich,
{zacharywilliam.hopton, tanja.samardzic}@uzh.ch
[2]Department of Informatics, University of Oslo, yves.scherrer@ifi.uio.no
[3]Department of Digital Humanities, University of Helsinki

Abstract

The distinction between function and content units of the lexicon has been somewhat neglected in recent NLP work, but it could still be useful when working with low-resource languages, and, in particular, to improve cross-lingual transfer. In this paper, we investigate to what extent BPE subword tokenization can be used to identify units of the functional lexicon in a language without any annotated data. We analyze subword tokens in terms of their productivity and attempt to find thresholds that best distinguish function from content tokens. On a sample of seven diverse languages, we find that the best results are obtained with 50 BPE merges. We also show that this subword tokenization setting can be beneficial for the interlinear glossing task.

1 Introduction

The distinction between function and content units of the lexicon used to be important in NLP before neural models were introduced. A common preprocessing step in many tasks was the removal of "stop words" from input text before any feature extraction. Lists of stop words typically included elements of the functional lexicon such as articles (e.g. *a, the*), pronouns (*e.g. it, they, you*), prepositions (e.g. *at, in*) and similar words. At the subword level, *stemming* was used to remove suffixes (e.g. *-s* in *plays, toys*). These steps were performed to reduce data sparsity by separating the form from the content so that the content could be modeled better, which is the main goal in most tasks.

Continuous representations in neural models removed the need for the function vs. content distinction, especially when large data sets are used for training. When working with the smaller data sets of low-resource languages, however, we expect this distinction still to be useful, especially in the context of cross-lingual transfer. Function units are harder to map across languages than content units, while being the most frequent tokens.

Arapaho	Wayne nihno'useet	*Wayne came over*
Gitksan	Ii na gya'ahl xadaa	*And I saw the moose*
Lezgi	Зун фенавай Дербентдиз	*I had gone to Derbent*
Natugu	Ma yrkrtxo-kz	*Houses were gone too*
Nyangbo	babatɔ bɔdõ	*They will cook some things*
Tsez	Šebi mi neɬqor äƛi	*What will you say about that?*
Uspanteko	tb'e laq chee'	*He leaves through the trees*

Figure 1: Short sentences in various languages with function units marked in red. The examples are taken from the SIGMORPHON 2023 Shared Task on Interlinear Glossing (Ginn et al., 2023).

The representation of function units is difficult through the lens of distributional semantics because they do not appear in fixed contexts (Boleda, 2020). Rather, function units tend to cooccur widely with all other units in a language. It has been shown that language models struggle to use function units' embeddings to abstract knowledge about syntax. Instead, they tend to rely on previously seen, surface-level evidence when processing information about ideas such as tense or negation (Li and Wisniewski, 2021; Hartmann et al., 2021). Already identifying a target language's function units as a preprocessing step could reduce confusion and improve the mapping between token vocabularies across languages. However, such pre-processing would require access to a target language's functional lexicon, i.e., the set of all of its function words and subwords.

With the long-term goal of eventually introducing a function–content distinction as part of preprocessing text data for NLP systems, we address a prerequisite question in the present study: **To what extent can subword tokenizers be used to extract a language's functional lexicon from raw text data?** We assess the degree to which units such as those marked in Figure 1 can be identified while pre-processing the text so that they can be given a special treatment depending on the application

Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 7839–7853
April 29 - May 4, 2025

(e.g., they can be mapped to a special token or removed). More specifically, our goal is to assess the extent to which small BPE vocabularies contain function units in diverse languages and whether a specific tokenization can be used for identifying function units without any labeled data. We show that BPE models trained for 50 merging steps result in subword vocabularies that contain the most prominent function units in diverse languages. We examine the interaction between this tokenization and NLP models on the task of Interlinear Glossing — which implies distinguishing between lexical and functional elements — and show the benefits of its use for some models.

2 Related Work

The arbitrariness of subword tokenization is increasingly recognized as an important problem in NLP motivating efforts towards removing the need for tokenization as a pre-processing step from the NLP pipeline (Clark et al., 2022; Xue et al., 2021). The other direction is developing methods for aligning popular subword tokenizers like byte-pair encoding (BPE) (Sennrich et al., 2016; Gage, 1994), SentencePiece Unigram Model (Kudo and Richardson, 2018) or WordPiece (Schuster and Nakajima, 2012) with linguistically meaningful analyses. Sometimes linguistically motivated tokenization improves processing (Bostrom and Durrett, 2020; Erdmann et al., 2019; Ataman and Federico, 2018; Ataman et al., 2017), but there is often no clear impact (Ortega et al., 2020; Saleva and Lignos, 2021; Scherrer et al., 2019; Banerjee and Bhattacharyya, 2018; Vania and Lopez, 2017; Zhu et al., 2019; Liu et al., 2020). The level of text compression achieved by tokenizers has been shown to impact the performance of NLP models, especially on the tasks that involve text generation (Gallé, 2019; Zouhar et al., 2023; Goldman et al., 2024). The degree of compression has also been associated with the morphological types of languages (Juola, 1998; Gutierrez-Vasques et al., 2023).

Paying attention to how the text is tokenized seems to be especially important in languages with richer morphology (Zevallos and Bel, 2023; Mager et al., 2022; Park et al., 2021). More generally, the type of script and other inherent properties of languages have been shown to influence the tokenization and cause biases against some types of languages (Ahia et al., 2023). Language-specific tokenization gives better performance of NLP models across languages (Rust et al., 2021), but this is hard to achieve in practice for multilingual pretrained models. Some efforts have been directed towards a more balanced multilingual tokenization via regularization (Kudo, 2018; Provilkov et al., 2020) or sampling (The NLLB Team et al. (2022)). Our focus on function units is intended to contribute to a more uniform and balanced tokenization across languages, exploiting a high-level binary distinction highly relevant to describing languages with glossed examples[1] but also to linguistic theory (Rizzi and Cinque, 2016).

3 Extracting True Labels from Interlinear Glossing

Our goal is to identify function units using modern subword tokenizers so that the same method can be applied to any language for which there are written texts. To evaluate the proposed methods, we need samples of segmented texts annotated with true labels showing whether each subword unit belongs to the functional lexicon or not.

Glossed linguistic examples contain most of the information that we need for our experiments. Glosses are traditionally used when describing languages to show how sentences are structured. They represent sentences segmented into subword units. Each subword unit is annotated to show its meaning (if the unit is lexical) or its form (if the unit is an element of the functional lexicon). Glosses for function units are special codes, usually written in capital letters, signaling grammatical categories. For example, the short sentence in Gitksan[2] in Figure 1 is glossed in the following way:

Morphemes	ii	n	gya'a	-hl	xadaa
Glosses	CONJ	1.I	see	-CN	moose
Function	Yes	Yes	No	Yes	No
Text	Ii na gya'ahl xadaa				
Translation	*And I saw the moose*				

To obtain the annotation of function units, we add another line of annotations (Function) to the data format that is provided in the glossed sentence. The mapping from the glosses to the binary labels (function unit or not) is created semi-automatically. Several rules and heuristics were written into a Python script to identify glosses that were representative of function units. First, we seek out glosses that were entirely written in capital letters, digits

[1] https://www.eva.mpg.de/lingua/pdf/Glossing-Rules.pdf

[2] This example is adapted from the SIGMORPHON 2023 Shared Task on Interlinear Glossing (Ginn et al., 2023), which is the main source of the data for all our experiments.

and periods or any combination of these three types of symbols (e.g. "NEG", "PAST", "2S.SUB"). This was not enough because some function units that would be included in a list of stop words are sometimes glossed with English words instead of special codes. For instance, this can happen with auxiliary verbs, pronouns, prepositions, interjections, and conjunctions. We identified such cases with a hard-coded list of function words adapted from Tang (2020) and classified them as function units. The output for each language was manually reviewed by a human annotator.

4 Productivity Analysis of BPE Subword Units

To isolate a language's function units, we rely on a productivity analysis of subword units created by a tokenizer. In short, we test the hypothesis that the most productive subword units are function units and that we can define a productivity threshold ζ that can identify function units in any language. Though we use annotated data here to evaluate various productivity thresholds, we aim to find a value of ζ that is applicable cross-linguistically so that a language's function units can be identified with just unlabeled text data.

Our approach is inspired by previous work showing that, cross-linguistically, texts are maximally compressed by applying a BPE model trained to carry out between 200 and 300 merges (Gutierrez-Vasques et al., 2021). A productivity analysis of the units created in the first 350 merges can be used to cluster typologically similar languages (Gutierrez-Vasques et al., 2023). For instance, languages with the highest productivity score of the BPE subword units created in the first 350 steps are Kalaallisut, Barasano, Apurinã and Alamblak, all classified in linguistic typology as polysynthetic and concatenative languages. Languages whose subword units created in the first 350 BPE steps get the lowest productivity score (i.e., Vietnamese, Thai, Sango and Yoruba) are classified in linguistic typology as analytic and mostly isolating. Between, we find synthetic languages such as French, Persian, Greek and Russian with medium productivity scores.

Assuming that these typological distinctions found in the productivity values of BPE subwords capture, at least approximately, some formal properties of languages, we try to exploit these properties for identifying function units. We thus conduct a productivity analysis similar to Gutierrez-Vasques et al. (2023) but with some modifications. We define two versions of the productivity score depending on how we define the context of BPE subwords.

In the first version, the context is the **word** in which a given subword is found. This productivity score is the number of unique words w in the set of all space-separated words W that contain a given BPE vocabulary item v after w is tokenized.

In the second version, we define the context more generally as a unique sequence of three subword units (**trigram**). To do this, we first tokenize each sentence with BPE, resulting in a sequence of subwords such as the Uspanteko sentence below, tokenized with a BPE model trained to carry out 200 merges:

```
e -chaq g -a -l -aa -n -i' ri -ch -oo -chaq juntiir
```

We then create overlapping subword trigrams for each sentence, resulting in the following for the above sentence:[3]

```
e -chaq g
  -chaq g -a
        g -a -l
          -a -l -aa ...
```

Once we have the set of subword trigrams T, the productivity score is the number of unique trigrams that contain a subword unit v.

The advantage of this second, trigram-based productivity definition is that it also assigns a high productivity to function words (when the BPE token is a whole word), which are highly unproductive under the first definition. Take, for example, the function word *and* in English. In the word-level definition of productivity, *and* would be assigned a low productivity, as it is unlikely to occur in many unique words. In a trigram context, however, *and* would presumably have a very high productivity if it were part of the BPE vocabulary, as it is likely to occur in many unique subword contexts.

The two versions of the productivity score can be formally described as follows:

$$\text{productivity}_{WORD}(v) = |\{w \in W | v \in w\}|$$
$$\text{productivity}_{TRIGRAM}(v) = |\{t \in T | v \in t\}|$$

Standardized productivity scores To make various productivity scores comparable across the measures and different experimental settings, we center and scale the values so that the distribution of the

[3]We do not consider trigrams that cross sentence boundaries, as the sentences in the corpus are not necessarily related.

Name	ISO 639-3	Area	Data size
Arapaho	arp	North America	39,501
Gitksan	git	North America	31
Lezgi(an)	lez	Eurasia	701
Natügu	ntu	Papunesia	791
Nyangbo	nyb	Africa	2,100
Tsez	ddo	Eurasia	3,558
Uspanteko	usp	North America	9,774

Table 1: Overview of the SIGMORPHON data sets. Data size is the number of glossed sentences in the training set.

scores in each given setting has a mean of zero and standard deviation of one.

5 Experiments

For all our experiments, we use the data from all seven languages included in SIGMORPHON 2023 Shared Task on Interlinear Glossing (Ginn et al., 2023) listed in Table 1. We train various versions of the BPE tokenizer using the `subword-nmt` library[4] on unsegmented texts (as shown in Figure 1).[5]

5.1 Classification of Function Units

For each of the seven languages in our data, we establish a set of function units, F, by classifying morphemes (words or parts of words) as functional or not based on their interlinear glossing. For example, the set F that would be created using the Gitksan example in Section 3 would contain the items *Li*, *n*, and *hl*. The set F corresponds to each language's functional lexicon.

Using the combined training and development sets of each language in Table 1, we train BPE models that carry out 50–350 merges with a step size of 50. For reference, we also test the classification capability of the character-segmented corpus, which we refer to as "0 merges." We then apply each of these models to languages' train, development, and test sets. Note that our maximal number of merges is far fewer than typically carried out in NLP, but this is what allows us to spot the most frequent subword units whose productivity analysis revealed typological differences between languages (see Section 4). For each BPE model, we evaluate how many subword tokens correspond to function units.

We use the standardized productivity threshold ζ (see Section 4) to classify BPE subword units as belonging to the set F or not. We calculate the productivity of each BPE vocabulary item using word and trigram frequencies from the tokenized test corpora, which were not used to train the BPE models. Recall that we expect items with higher productivity to be better candidates. We explore the range of ζ values between [–5, 5] with a step size of 0.1. To calculate precision and recall in each case, we apply the following definitions:

$$\text{True Positive: } v \in F \wedge \text{productivity}(v) > \zeta$$
$$\text{False Positive: } v \notin F \wedge \text{productivity}(v) > \zeta$$
$$\text{False Negative: } v \in F \wedge \text{productivity}(v) \leq \zeta$$

Taking these definitions, we can compute recall, precision, and F1 scores for any given version of the BPE tokenization and any productivity threshold ζ. For our purposes, a high recall indicates that among the function units in a language's BPE vocabulary, a large proportion have productivities above ζ. When precision is high for some ζ, it indicates that most of the subword units with the productivity above ζ are indeed function units.

Single characters When text is tokenized using BPE models with few merges, many tokens are single characters. The productivity value of single characters will be high because they tend to appear in a large number of unique contexts, but only some of them will be function units. Upon observing the most productive subword vocabulary units in each language we found that, except for Arapaho (the language with the most data), single character vocabulary items were indeed the majority of the 15 most productive units in our sample. A potentially useful simplification would be to ignore all single characters (classify them all as not functional) and consider only the units merged by BPE as function units candidates. With this simplification, we might see less confusion, but we are sure to miss single-character function units. We thus report the results in two settings: **with** and **without single characters**.

Random Baselines We establish two random baselines against which we compare our approach to classifying function units. In the **naive** baseline, we take the BPE vocabulary for a language at some merge and randomly assign each item the label "functional" or "not functional" with equal

[4] https://github.com/rsennrich/subword-nmt

[5] We also experimented with the SentencePiece Unigram Model, but we report only the BPE experiments because the main findings did not change. The WordPiece tokenizer was tested by Gutierrez-Vasques et al. (2023) also without impacting the main findings.

probability ($p = 0.5$). In the **hard** baseline, we do the same random labeling using the probability that a unit in the original morphological segmentation of a language's corpus is functional. For instance, in the Lezgi interlinear glossing data, we found that 63.8% of the morphemes were function units; therefore, we use a value of $p = 0.638$ to assign the labels for Lezgi's hard baseline. Using the labels assigned in both baselines, we calculate F1 scores for comparison to the rest of our classification conditions.

5.2 Testing Models for Automatic Interlinear Glossing

To see how our tokenization analysis might interact with NLP models, we turn to the task of automatic interlinear glossing as defined in the 2023 SIGMORPHON Shared Task (GlossingST). We train and test four different model architectures on all the languages included in the task. For each model, we compare a base setting with character-level tokenization to a BPE tokenization setting based on our analysis. We use the predefined data splits.

On the one hand, GlossingST is similar to our unsupervised task of classifying function units in that the systems might be using an inherent (implicit) distinction between the two types of tokens to learn the mapping between input text and glosses. But GlossingST is harder than our unsupervised classification task because we predict only a sequence of binary labels, while the models need to output full glosses. On the other hand, glossing models might learn to convert input text into the output labels without necessarily learning to segment words in any specific way, while our prediction is determined by a given subword tokenization, which results in segmenting words in a specific way.

5.2.1 Models

We start with the winner of shared the task, the **TÜ-CL** submission (Girrbach, 2023; Ginn et al., 2023).[6] This model encodes the input tokens with a bi-directional LSTM. The LSTM representations of characters are used to form longer subword units by unsupervised clustering. These units represent a segmentation of words into morphemes. A multilayer perceptron is then used to predict a gloss for each predicted morpheme in the input word. All weights are learned end-to-end so that the subword units formed by unsupervised clustering maximize the final performance on glosses.

The next architecture that we consider is statistical machine translation (**SMT**). This method is well suited for monotonic character-level transduction tasks (Bollmann, 2019; Kuparinen et al., 2023) and works well with relatively small data sets. Since some of the languages in GlossingST are represented with only tens of sentences, SMT may outperform more sophisticated neural architectures. In our experiments, we use the Moses SMT toolkit (Koehn et al., 2007) for translation modeling, eflomal (Östling and Tiedemann, 2016) for word alignment, KenLM (Heafield, 2011) for language modelling and MERT for tuning the model weights. We train a single 10-gram language model for all seven languages, and disable distortion to force monotonicity.

Finally, we try two versions of DeepSPIN models (Peters and Martins, 2022), **DeepSPIN-2** and **DeepSPIN-3**, which achieved particularly good results in tasks involving subword analyses and tokenization (Zevallos and Bel, 2023). DeepSPIN-2 is a bi-directional LSTM encoder and decoder, while DeepSPIN-3 uses a transformer architecture. The characteristic feature of the DeepSPIN models is the *entmax* output layer, a sparse alternative to *softmax* that can assign tokens zero probability (Peters et al., 2019). The DeepSPIN models are implemented in FairSeq[7] using the hyperparameters detailed in Peters and Martins (2022).

To tokenize the glosses, we split the output sequences at white spaces and then again at hyphens. Hyphens are concatenated to the left-hand portion of the gloss they separated. We do not tokenize glosses at periods. For instance, in the Natügu development data, the gloss "MID-say-PDIR.YON" is tokenized into "MID-", "say-", and "PDIR.YON". Given that the TÜ-CL submission originally removes hyphens from the output vocabulary entirely and restores them during post-processing of the model's word-level outputs, we edit the gloss tokenization to match the process described above and retrain the model with the same hyperparameters described in GitHub.

6 Results

6.1 Unsupervised Classification

Using the definitions of true and false positives and negatives presented in Section 4, we calculate the

[6] https://github.com/LGirrbach/sigmorphon-2023-glossing

[7] https://github.com/facebookresearch/fairseq

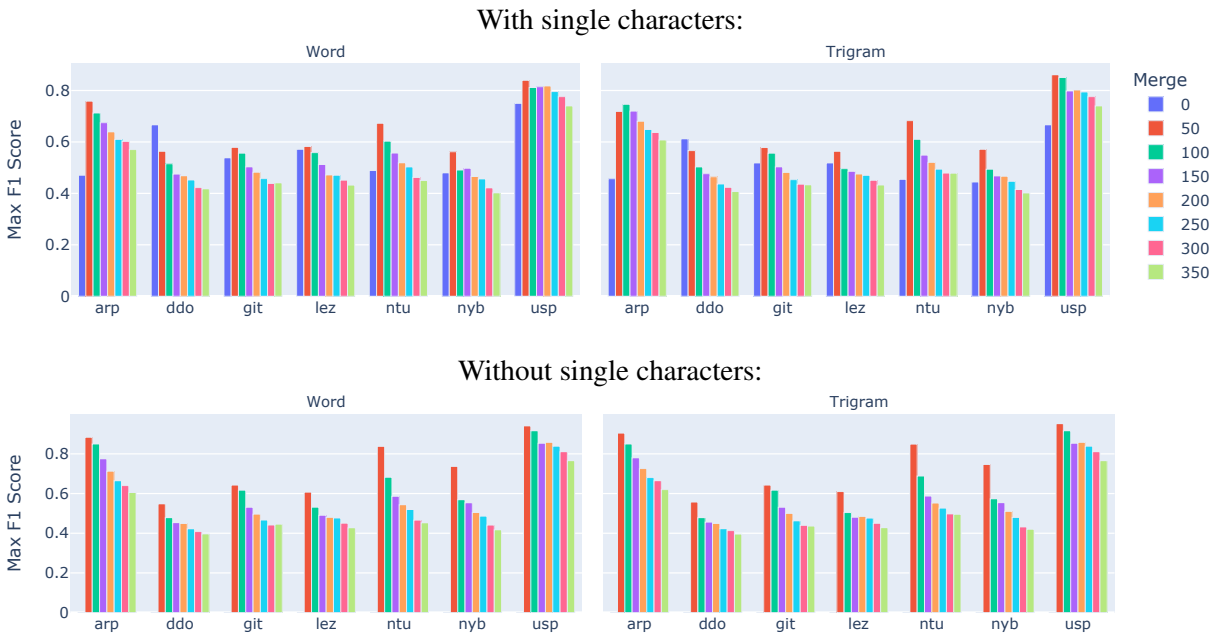

Figure 2: The maximum F1 scores across all tested values of ζ for each language at seven different merges. There are no 0 merge results in the setting without single characters.

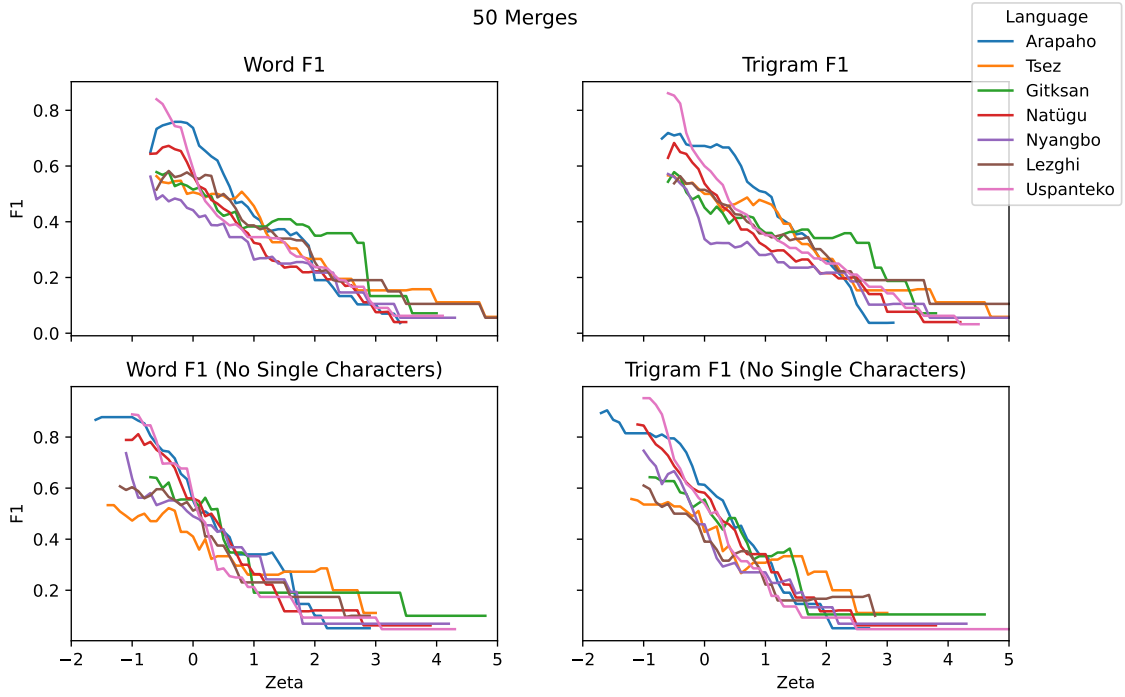

Figure 3: F1 scores for the unsupervised classification of function units in our set of languages. Results represent the F1 scores for the corpora tokenized with a BPE model trained to carry out 50 merges. Though we explore values of ζ beginning at -5, the visualization begins at the lowest subword productivity that actually occurred in each language's tokenized corpora rather than at -5. Long horizontal lines at high thresholds are indicative of outliers with very high productivities relative to the rest of the subwords in a language's subword vocabulary.

	Baseline		Word		Trigram	
	Naive	Hard	$\zeta = -0.5$	Best ζ	$\zeta = -0.5$	Best ζ
arp	0.483 (0.054)	0.533 (0.047)	0.746	0.759	0.710	0.719
ddo	0.391 (0.061)	0.438 (0.045)	0.542	0.564	0.564	0.566
git	0.417 (0.065)	0.477 (0.047)	0.568	0.579	0.578	0.578
lez	0.360 (0.054)	0.396 (0.046)	0.556	0.583	0.538	0.564
ntu	0.476 (0.055)	0.528 (0.043)	0.667	0.673	0.683	0.683
nyb	0.455 (0.061)	0.512 (0.051)	0.495	0.562	0.561	0.571
usp	0.553 (0.053)	0.657 (0.037)	0.823	0.840	0.853	0.861
Avg.	0.448	0.506	0.628	0.651	0.641	0.649

Table 2: F1 scores obtained applying the same productivity threshold ($\zeta = -0.5$) in comparison to the language specific maximum F1 scores (applying the language best ζ). In all the cases, the tokenizer is BPE trained to carry out 50 merges. Baseline scores represent the average F1 over $n = 500$ sets of random labels in each condition, with standard deviations in parentheses. These results include single characters.

F1 score for our classification approach at every tested ζ for each language after applying BPE models trained to carry out 50, 100, 150, 200, 250, 300, and 350 merges. We also calculate F1 scores for merge 0 for the character-level corpora.

In Figure 2, we indicate the maximum F1 scores of any ζ in each condition. We find that, with little exception, our unsupervised classification with a subword productivity threshold is best with the corpora tokenized with BPE models trained to carry out just 50 merges. As we increase the number of BPE merges (resulting in longer subword units), we see a drop in the maximum F1 score at a rather similar rate across languages.

Comparing the different definitions of the context used for the productivity analysis (word vs. trigram), we do not observe much difference in how the number of BPE merges impacts the maximum F1 score. On the other hand, excluding single characters seems to help reach better maximum F1 scores in all the languages, and especially in those with larger data sizes (Arapaho, Nyangbo, Tsez and Uspanteko).

To better observe the behavior of our classification approach at 50 BPE merges, we plot the F1 score for each language as a function of the tested ζ in Figure 3. Here we observe that relatively similar values of ζ yield the highest F1 score across languages despite the differences in the overall performance. Lower ζ values generally result in higher F1 score, with a steady drop as ζ increases. Here again, the differences between the word vs. trigram context is very small, while excluding vs. including single characters has a bigger impact on the overall performance.

When we include single characters, the maximum F1 scores occur after a small increase in ζ values. This means that the productivity analysis helps eliminate some subword units that are not function units from the set of candidates. After this initial peak, we see a relatively steady performance drop which means that increasing the productivity threshold eliminates many units that are functional.

The picture is slightly different in the setting without single characters, where the best performance is usually achieved with the lowest productivity threshold. This means that the productivity analysis is not needed at all when single characters are excluded from the set of candidates and the best result is obtained by classifying as functional all subwords units merged in the first 50 BPE steps. Excluding single characters from the set of candidates gives overall better results than applying any productivity threshold to single characters.

The main generalization in these findings is that the best approach to identifying function units in an unsupervised way is to tokenize a given text with a BPE model trained to perform 50 merges and then label as function units all subwords merged by BPE. This approach appears equally applicable to various kinds of languages and different data sizes. To find single-character function units, which are all left unrecognized in this approach, one can apply a productivity threshold but at the cost of the overall performance. At this point, we are typically capturing between 20 and 40 function units of a language (see Table 6 in Appendix B). Here again, the optimal productivity threshold appears rather simi-

		arp	ddo	git	lez	ntu	nyb	usp	Average
TÜ-CL	Character	**0.824**	**0.763**	**0.148**	**0.584**	**0.526**	**0.889**	**0.790**	**0.646**
	BPE-50	0.820	0.675	0.132	0.355	0.505	0.840	0.714	0.577
SMT	Character	0.669	**0.709**	0.052	**0.536**	0.549	**0.857**	**0.678**	0.579
	BPE-50	**0.768**	0.680	0.052	0.510	**0.576**	0.849	0.663	**0.585**
DeepSpin2	Character	**0.764**	**0.727**	**0.069**	**0.446**	**0.531**	**0.825**	0.740	**0.586**
	BPE-50	0.753	0.712	0.067	0.409	0.483	0.818	**0.744**	0.569
DeepSpin3	Character	0.697	0.307	0.054	0.058	0.075	0.610	**0.645**	0.349
	BPE-50	**0.708**	**0.325**	**0.131**	**0.071**	**0.140**	**0.672**	0.621	**0.381**

Table 3: F1 scores for automatic interlinear glossing of corpora's function units. BPE-50 refers to the input corpora having been tokenized by monolingual BPE models trained to carry out 50 merges.

lar across languages. Table 2 shows that applying a single threshold to all languages (averaging the best language specific thresholds to $\zeta = -0.5$) the performance remains very close to the language-specific maximum (cf. Tables 4 and 5 in Appendix B). Table 2 also indicates that even when we use this single threshold rather than the threshold that yields the best F1 score for a language, our approach still outperforms the naive and hard random baselines for every language but one (Nyangbo using word-level productivity). The above results are based on productivity scores calculated from the test set word and trigram frequencies, though we note that calculating the productivitiy scores from the combined train and development sets yields similar results (see Figure 4 in Appendix B).

6.2 Interlinear Glossing Models

Having established that BPE tokenization with a model trained to perform 50 merges (BPE-50) gives the best results in unsupervised classification of function units, we test if this tokenization can help interlinear glossing models in predicting the functional part of the glosses. We evaluate models using the GlossST evaluation script, which we edit to include an additional measure of the F1 score over the glosses that we labeled as functional.[8]

Table 3 shows a comparison between F1 scores on function units that the models achieve with their base character-level tokenization and the scores achieved using the BPE-50 tokenization. We find that the BPE-50 tokenization helps DeepSpin-3 on all languages except Uspanteko. It also results in an average improvement across languages for SMT. For TÜ-CL and DeepSpin-2, character-level tokenization resulted in higher F1 scores than BPE-50 on average. Regardless of the evaluation metric, the TÜ-CL model with character-level tokenization has the best overall performance.

For reference, overall accuracy is presented in Tables 8 and 9 in Appendix C. The same trend is reflected in these metrics as the F1 scores over the function units: BPE-50 results in the best average scores for SMT and DeepSpin-3, while character level input results in the best average scores for DeepSpin-2 and the TÜ-CL model.

7 Discussion

In this study, we set out to quantify how sensitive BPE is to the function–content distinction. We do so with the goal of introducing this distinction to models during preprocessing. We find that, when using BPE models trained to carry out relatively few merges (namely, 50), many function units of a language can be identified in tokenized corpora by analyzing the units' productivity.

With respect to the distinction between trigram and word-based definitions of productivity, our results seem to indicate that there is not a substantial difference between the two definitions. Regardless of whether we include single characters, Tables 4 and 5 (Appendix B) indicate that both productivity definitions result in similar maximum F1 scores at 50 merges, and that the maximum occurs using a similar threshold. This lack of difference could be related to the selection of languages in our sample, as Ginn et al. (2023) note that all of them demonstrate synthetic morphology to some degree. As for the difference in results when we do and do not include single characters in our calculations, we see that while the maximum F1 scores are higher on average for the conditions without single characters,

[8]The original evaluation script is available here: https://github.com/sigmorphon/2023glossingST

there is less stability with respect to the threshold ζ at which these maxima occur.

Concerning the results on interlinear glossing, we see that our tokenization — which we used with the intention of including function units in the input data to the models — resulted in higher average performance in glossing function units for the SMT model and the DeepSPIN-3 model. The reason why we do not see improvements with the TÜ-CL model might be the fact that this model learns to segment the input words according to what will maximize the score on the final gloss predictions for each segment. With the BPE-50 tokenization, the smallest units in the segments of each word become subwords rather than characters. It is then possible that with BPE-50 rather than character input, we sometimes initialize the clusters that it learns to make in a way that actually hurts the glossing performance, perhaps because the BPE segmentation does not always occur at morphologically relevant positions. Still, other studies have shown that using relatively small BPE vocabularies can be beneficial for performance in down-stream tasks. For instance, in training multilingual models for translation of Spanish to eleven indigenous languages, Attieh et al. (2024) found that using small, monolingual BPE models (trained to carry out 300 merges) outperformed the same model trained on a larger joint vocabulary. Moreover, Amrhein and Sennrich (2021) found that using relatively small vocabulary units (vocabulary sizes of 500) yielded improvements in the translation of instances of non-concatenative morphology, such as vowel harmony and reduplication.

8 Conclusion

In the present study, we explore the extent to which a BPE tokenizer can be used to extract a language's functional lexicon without any annotated data. Our experiments on using a productivity analysis over languages' BPE subword vocabularies to find the function units ultimately led us to conclude that simply applying BPE models trained to carry out 50 merges is the best way to obtain a sample of a language's function units from BPE regardless of the morphological type of the language and the amount of available data. This method is not perfect, as it yields some false positives and leaves quite a few function units unrecognized. However, it is unsupervised and equally applicable across diverse languages.

Introducing a large portion of function units in the input by tokenizing the text with a language-specific BPE model trained for 50 steps improved the performance of some models on the task of interlinear glossing. In future applications, we hope to explore how we might treat those function units differently during preprocessing in an effort to improve the models' representations for different tasks.

Limitations

A key limitation of the method presented is that it does not identify all of a given language's functional lexicon, especially as far as single-character function units are concerned. In Appendix A, we define a metric to evaluate the total proportion of a language's functional lexicon that have been correctly classified ("prop_F"). As seen in Tables 4 and 5 (Appendix B), we can usually accurately classify less than a fifth of the languages' unique function units. Still, it is likely that the function units that are captured by BPE at these early merges are among the language's most frequent function units. Given the Zipfian distribution of word frequencies, this would mean that a large portion of all the function units in a corpus is still being classified correctly. Indeed, when we look at the proportion of all function tokens (rather than unique function types) correctly classified in each language's test set, we see that this is the case (see Appendix B, Table 7).

Due to the scarcity of interlinear glossing data, our experiments include only languages with alphabetic scripts with relatively limited typological and areal scope. Our findings thus might not generalize to all languages.

Acknowledgments

This work has been supported by the Academy of Finland through project No. 342859 "CorCoDial – Corpus-based computational dialectology". The authors wish to acknowledge CSC – IT Center for Science, Finland, for computational resources.

References

Orevaoghene Ahia, Sachin Kumar, Hila Gonen, Jungo Kasai, David Mortensen, Noah Smith, and Yulia Tsvetkov. 2023. Do all languages cost the same? tokenization in the era of commercial language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*,

pages 9904–9923, Singapore. Association for Computational Linguistics.

Chantal Amrhein and Rico Sennrich. 2021. How suitable are subword segmentation strategies for translating non-concatenative morphology? In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 689–705, Punta Cana, Dominican Republic. Association for Computational Linguistics.

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 Track)*, pages 97–110.

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

Joseph Attieh, Zachary Hopton, Yves Scherrer, and Tanja Samardžić. 2024. System description of the NordicsAlps submission to the AmericasNLP 2024 machine translation shared task. In *Proceedings of the 4th Workshop on Natural Language Processing for Indigenous Languages of the Americas (AmericasNLP 2024)*, pages 150–158, Mexico City, Mexico. Association for Computational Linguistics.

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.

Gemma Boleda. 2020. Distributional semantics and linguistic theory. *Annual Review of Linguistics*, 6(1):213–234.

Marcel Bollmann. 2019. A large-scale comparison of historical text normalization systems. 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 3885–3898, Minneapolis, Minnesota. Association for Computational Linguistics.

Kaj Bostrom and Greg Durrett. 2020. Byte pair encoding is suboptimal for language model pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4617–4624, Online. Association for Computational Linguistics.

Jonathan H. Clark, Dan Garrette, Iulia Turc, and John Wieting. 2022. Canine: Pre-training an efficient tokenization-free encoder for language representation. *Transactions of the Association for Computational Linguistics*, 10:73–91.

Alexander Erdmann, Salam Khalifa, Mai Oudah, Nizar Habash, and Houda Bouamor. 2019. A little linguistics goes a long way: Unsupervised segmentation with limited language specific guidance. In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 113–124.

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

Matthias Gallé. 2019. Investigating the effectiveness of BPE: The power of shorter sequences. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1375–1381, Hong Kong, China. Association for Computational Linguistics.

Michael Ginn, Sarah Moeller, Alexis Palmer, Anna Stacey, Garrett Nicolai, Mans Hulden, and Miikka Silfverberg. 2023. Findings of the SIGMORPHON 2023 shared task on interlinear glossing. In *Proceedings of the 20th SIGMORPHON workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 186–201, Toronto, Canada. Association for Computational Linguistics.

Leander Girrbach. 2023. Tü-CL at SIGMORPHON 2023: Straight-through gradient estimation for hard attention. In *Proceedings of the 20th SIGMORPHON workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 171–185, Toronto, Canada. Association for Computational Linguistics.

Omer Goldman, Avi Caciularu, Matan Eyal, Kris Cao, Idan Szpektor, and Reut Tsarfaty. 2024. Unpacking tokenization: Evaluating text compression and its correlation with model performance. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 2274–2286, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Ximena Gutierrez-Vasques, Christian Bentz, and Tanja Samardžić. 2023. Languages through the looking glass of BPE compression. *Computational Linguistics*, 49(4):943–1001.

Ximena Gutierrez-Vasques, Christian Bentz, Olga Sozinova, and Tanja Samardzic. 2021. From characters to words: the turning point of BPE merges. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3454–3468, Online. Association for Computational Linguistics.

Mareike Hartmann, Miryam de Lhoneux, Daniel Hershcovich, Yova Kementchedjhieva, Lukas Nielsen, Chen Qiu, and Anders Søgaard. 2021. A multilingual benchmark for probing negation-awareness with minimal pairs. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 244–257, Online. Association for Computational Linguistics.

Kenneth Heafield. 2011. KenLM: Faster and smaller language model queries. In *Proceedings of the Sixth*

Workshop on Statistical Machine Translation, pages 187–197, Edinburgh, Scotland. Association for Computational Linguistics.

Patrick Juola. 1998. Measuring linguistic complexity: The morphological tier. *Journal of Quantitative Linguistics*, 5(3):206–213.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

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.

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

Olli Kuparinen, Aleksandra Miletić, and Yves Scherrer. 2023. Dialect-to-standard normalization: A large-scale multilingual evaluation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13814–13828, Singapore. Association for Computational Linguistics.

Bingzhi Li and Guillaume Wisniewski. 2021. Are neural networks extracting linguistic properties or memorizing training data? an observation with a multilingual probe for predicting tense. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3080–3089, Online. Association for Computational Linguistics.

Christopher Liu, Laura Dominé, Kevin Chavez, and Richard Socher. 2020. Central Yup'ik and machine translation of low-resource polysynthetic languages. *arXiv preprint arXiv:2009.04087*.

Manuel Mager, Arturo Oncevay, Elisabeth Mager, Katharina Kann, and Thang Vu. 2022. BPE vs. morphological segmentation: A case study on machine translation of four polysynthetic languages. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 961–971, Dublin, Ireland. Association for Computational Linguistics.

John E Ortega, Richard Castro Mamani, and Kyunghyun Cho. 2020. Neural machine translation with a polysynthetic low resource language. *Machine Translation*, 34(4):325–346.

Robert Östling and Jörg Tiedemann. 2016. Efficient word alignment with Markov Chain Monte Carlo. *Prague Bulletin of Mathematical Linguistics*, 106:125–146.

Hyunji Hayley Park, Katherine J Zhang, Coleman Haley, Kenneth Steimel, Han Liu, and Lane Schwartz. 2021. Morphology matters: A multilingual language modeling analysis. *Transactions of the Association for Computational Linguistics*, 9:261–276.

Ben Peters and Andre FT Martins. 2022. Beyond characters: Subword-level morpheme segmentation. In *Proceedings of the 19th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 131–138.

Ben Peters, Vlad Niculae, and André F. T. Martins. 2019. Sparse sequence-to-sequence models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1504–1519, Florence, Italy. Association for Computational Linguistics.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. 2020. BPE-dropout: Simple and effective subword regularization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1882–1892, Online. Association for Computational Linguistics.

Luigi Rizzi and Guglielmo Cinque. 2016. Functional categories and syntactic theory. *Annual Review of Linguistics*, 2(1):139–163.

Phillip Rust, Jonas Pfeiffer, Ivan Vulić, Sebastian Ruder, and Iryna Gurevych. 2021. How good is your tokenizer? on the monolingual performance of multilingual language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3118–3135, Online. Association for Computational Linguistics.

Jonne Saleva and Constantine Lignos. 2021. The effectiveness of morphology-aware segmentation in low-resource neural machine translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 164–174, Online. Association for Computational Linguistics.

Yves Scherrer, Raúl Vázquez, and Sami Virpioja. 2019. The University of Helsinki submissions to the WMT19 similar language translation task. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 236–244.

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.

Warren Tang. 2020. *6. The Development of a Functional Function Word List from Form and Meaning*. Ph.D. thesis, Fukuyama University.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. 2022. No language left behind: Scaling human-centered machine translation. *Preprint*, arXiv:2207.04672.

Clara Vania and Adam Lopez. 2017. From characters to words to in between: Do we capture morphology? *arXiv preprint arXiv:1704.08352*.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2021. ByT5: Towards a token-free future with pre-trained byte-to-byte models. *arXiv preprint arXiv:2105.13626*.

Rodolfo Zevallos and Nuria Bel. 2023. Hints on the data for language modeling of synthetic languages with transformers. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12508–12522, Toronto, Canada. Association for Computational Linguistics.

Yi Zhu, Ivan Vulić, and Anna Korhonen. 2019. A systematic study of leveraging subword information for learning word representations. *arXiv preprint arXiv:1904.07994*.

Vilém Zouhar, Clara Meister, Juan Gastaldi, Li Du, Mrinmaya Sachan, and Ryan Cotterell. 2023. Tokenization and the noiseless channel. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5184–5207, Toronto, Canada. Association for Computational Linguistics.

A Calculating the Proportion of all Function Units Correctly Classified

Given that we would like to know the proportion of *all* of a language's function units above the threshold ζ (not just the proportion of function units in the BPE vocabulary above the threshold), we define another metric, prop_F, as follows:

$$\text{prop}_F = \frac{|\text{True Positives}|}{|F|} \tag{1}$$

A higher prop_F indicates that a larger portion of all the units in a language's set of function units F was correctly classified, regardless of how many function units were in the BPE vocabulary.

B Additional Function Unit Classification Results

		Max F1	ζ at Max F1	Precision at Max F1	Recall at Max F1	prop_F
Word	arp	0.759	-0.2	0.710	0.815	0.033
	ddo	0.564	-0.6	0.403	0.939	0.082
	git	0.579	-0.4	0.458	0.786	0.203
	lez	0.583	-0.4	0.455	0.811	0.088
	ntu	0.673	-0.4	0.603	0.760	0.157
	nyb	0.562	-0.7	0.391	1.0	0.169
	usp	0.840	-0.6	0.797	0.887	0.056
	Average	0.651	-0.471	0.545	0.857	0.113
Trigram	arp	0.719	-0.6	0.63	0.836	0.033
	ddo	0.566	-0.6	0.405	0.941	0.082
	git	0.578	-0.5	0.444	0.828	0.203
	lez	0.564	-0.4	0.437	0.795	0.088
	ntu	0.683	-0.5	0.575	0.84	0.157
	nyb	0.571	-0.6	0.400	1.0	0.169
	usp	0.861	-0.6	0.787	0.952	0.056
	Average	0.649	-0.543	0.525	0.885	0.113

Table 4: Classifier evaluation for corpora tokenized with a BPE model trained to carry out 50 merges. This condition includes single-character function units in the calculations.

		Max F1	ζ at Max F1	Precision at Max F1	Recall at Max F1	prop_F
Word	arp	0.884	-1.7	0.792	1.0	0.024
	ddo	0.548	-1.5	0.378	1.0	0.046
	git	0.643	-0.7	0.474	1.0	0.141
	lez	0.607	-1.2	0.447	0.944	0.044
	ntu	0.838	-1.2	0.721	1.0	0.103
	nyb	0.737	-1.1	0.583	1.0	0.139
	usp	0.941	-1.1	0.889	1.0	0.038
	Average	0.743	-1.214	0.612	0.992	0.076
Trigram	arp	0.905	-1.6	0.826	1.0	0.024
	ddo	0.557	-1.2	0.386	1.0	0.046
	git	0.643	-0.9	0.474	1.0	0.141
	lez	0.610	-1.0	0.439	1.0	0.047
	ntu	0.849	-1.1	0.738	1.0	0.103
	nyb	0.747	-1.0	0.596	1.0	0.139
	usp	0.952	-0.9	0.909	1.0	0.038
	Average	0.752	-1.1	0.624	1.0	0.077

Table 5: Classifier evaluation for corpora tokenized with a BPE model trained to carry out 50 merges. This condition does not include single-character function units in the calculations.

	With SC		No SC	
	Word	Trigram	Word	Trigram
arp	52	52	38	38
ddo	32	32	18	18
git	26	26	18	18
lez	34	34	17	18
ntu	47	47	31	31
nyb	34	34	28	28
usp	60	60	40	40
Average	40.714	40.714	27.143	27.286

Table 6: The count of true positives (i.e., function units above the threshold ζ) at 50 merges, using the value of ζ that yields the highest F1 score for each language and condition.

	With SC		No SC	
	Word	Trigram	Word	Trigram
arp	32.53	38.46	30.99	30.99
ddo	61.84	63.24	15.53	15.52
git	52.17	57.56	48.45	48.45
lez	64.68	68.23	40.66	44.09
ntu	67.67	78.06	67.92	67.93
nyb	74.55	74.55	50.08	50.08
usp	54.50	63.80	32.60	32.60
Average	58.28	63.41	40.89	41.38

Table 7: The percentage of function tokens in each language's test set that were correctly labeled. Values represent the corpora tokenized with BPE models trained to carry out 50 merges, and use the ζ values at which they achieve their maximum F1 scores, detailed in Tables 4 and 5 for each condition.

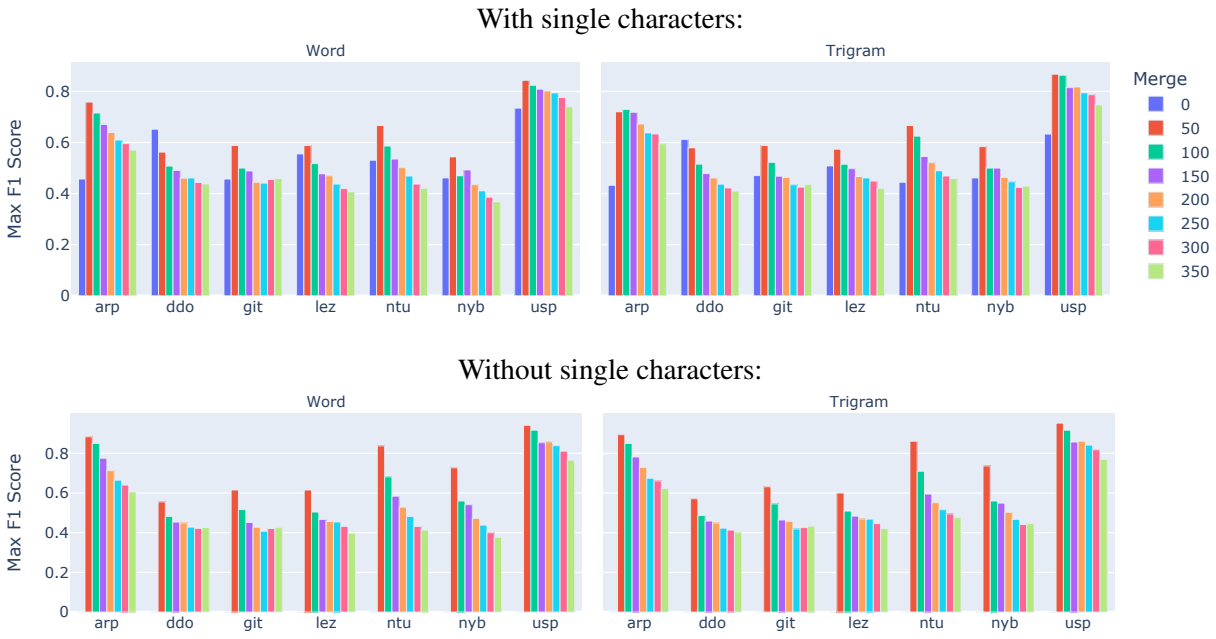

Figure 4: The maximum F1 scores across all tested values of ζ for each language at seven different merges. Productivities used for classification are calculated over each language's combined training and development set. There are no 0 merge results in the setting without single characters.

C Additional Interlinear Glossing Results

		arp	ddo	git	lez	ntu	nyb	usp	Average
TÜ-CL	Character	76.25	75.07	9.55	59.84	50.03	87.40	71.59	61.39
	BPE-50	76.15	65.85	8.54	33.67	46.77	82.44	63.79	53.89
SMT	Character	60.90	69.50	3.60	52.20	48.90	85.10	63.30	54.79
	BPE-50	71.62	66.87	3.33	52.11	51.02	84.05	61.56	55.79
DeepSpin2	Character	68.28	70.59	3.62	43.66	46.49	79.81	66.39	54.12
	BPE-50	67.31	68.85	3.91	40.45	42.19	78.55	67.16	52.63
DeepSpin3	Character	61.36	28.30	3.33	6.20	6.24	55.32	58.76	31.36
	BPE-50	63.01	29.91	8.10	7.07	11.10	63.20	54.22	33.80

Table 8: Accuracy in glossing at the morpheme level. Values represent percentages correct.

		arp	ddo	git	lez	ntu	nyb	usp	Average
TÜ-CL	Character	77.10	80.83	16.15	78.33	78.44	85.90	73.43	70.03
	BPE-50	78.03	78.00	17.97	54.06	74.16	84.39	68.98	65.08
SMT	Character	60.30	54.30	1.00	48.90	42.00	78.60	59.70	49.26
	BPE-50	73.73	72.98	7.55	54.40	57.25	80.51	64.20	58.66
DeepSpin2	Character	69.88	76.62	2.34	45.49	60.13	76.44	73.10	57.71
	BPE-50	70.63	74.84	2.60	41.20	50.09	75.59	73.60	55.51
DeepSpin3	Character	64.96	27.85	2.34	4.74	7.81	52.32	63.12	31.88
	BPE-50	65.98	30.21	2.34	4.18	9.76	59.41	58.50	32.91

Table 9: Accuracy in glossing at the word level. Values represent percentages correct.