Strategies in subword tokenization: humans vs. algorithms

Anonymous ACL submission

Abstract

The output of subword tokenization can be very different depending on what algorithm is used. It is typically judged as more or less *plausible*, depending on how much it corresponds to human intuition. A subword vocabulary overlap between manual and automatic segmentation is an indicator of plausibility, but it does not reveal much on how the process of segmentation compares with human analysis. In this study, we propose a new method to analyze subword segmentation strategies relying on a spatial analysis of the distribution of subwords' lengths. Our experiments on English, Finnish and Turkish show that humans tend to balance creativity and consistency, while algorithms tend to be either strongly biased or inconsistent. To imitate humans better, algorithms need to produce subword segments of moderately uneven length, which can be achieved by combining complementary strategies.

1 Introduction

Words in natural languages consist of smaller units which are traditionally studied in linguistic morphology. While the exact form of the subword structure is still debated in linguistics (Matthews, 1991; Anderson et al., 1992; Ackerman and Malouf, 2013), modern NLP increasingly exploits surface segmentation of words for subword tokenization (e.g. *coworking* can be split into *co|work|ing*). All pretrained models based on BERT, for instance, apply some kind of subword tokenization (Devlin et al., 2019). Its purpose is to reduce the vocabulary and solve the problem of unknown words.

Subword tokenization is a preprocessing step performed by unsupervised algorithms prior to encoding. Its output might correspond to human intuition on how one should split words, but often it does not. Agreement between automatic segmentation and human intuition is typically regarded as plausibility of automatic segmentation. It is widely accepted

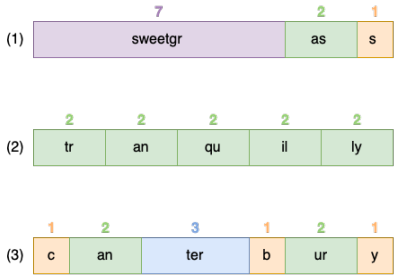


Figure 1: Segmentations' examples on the words of length 10: *sweetgrass*, *tranquilly*, *canterbury*.

that more plausible subword segmentation is beneficial to downstream tasks. In particular, Park et al. (2021); Bostrom and Durrett (2020) show that linguistically motivated segmentation helps language modelling, especially in languages with complex words.

Numerous algorithms have been proposed for splitting words into smaller units (Hammarström and Borin, 2011). They can be divided into two big classes: probabilistic models (Creutz and Lagus, 2005; Kudo, 2018) and data compression algorithms (Schuster and Nakajima, 2012; Sennrich et al., 2016). Probabilistic models are typically regarded as more capable of reproducing human segmentations and thus yield more plausible segments.

The evaluation of segmentation plausibility can be direct or indirect. Direct comparison is typically performed in the context of morphological segmentation (Virpioja et al., 2011) and it consists in measuring the agreement between manual and automatic segments. Indirect evaluation is more common in the context of subword tokenization for downstream tasks: the plausibility of subword segmentation is assessed via the performance on a downstream task. Both of these approaches are focused on the results of the segmentation, not the process.

In this paper, we aim to find out how humans and

algorithms come up with divergent results. What strategies should algorithms take in order to behave more like human annotators? Which algorithms are currently more like humans? To answer these questions, we introduce a novel method for comparing segmentations by analyzing the length of the resulting segments.

The lengths can be **even** (all subwords in a word have similar lengths) or **uneven** (some subwords are short and some are long) (Figure 1). We map distributions of subwords' lengths to an unevenness index (one value per word-level token), and plot these values in a two-dimensional space (word length $\times$ unevenness index). This gives us scatterplots that can be analyzed with spatial statistical methods and compared across different settings. In this way, we can look for segmentation patterns at a more abstract level and ask questions such as: Do algorithms tend to produce more even segmentations than humans? Are segments in long words more even than segments in short words?

Our analysis is intended to improve the current understanding of how subword tokenization is done and how it can be done better. This stands in contrast to the recent trends in NLP research aiming at removing the subword tokenization step from the processing pipeline (Xue et al., 2021; Clark et al., 2021). We believe that finding the right subword segmentation is still a valuable task, which can eventually save processing time, resources, and provide better performance.

2 Related Work

Previous works compare the segmentation algorithms only as a part of improving particular downstream tasks. The general outline in these papers consists of segmenting the input by different means, and giving conclusions about the algorithms by assessing the performance on the downstream tasks. The majority of them work on neural machine translation (NMT) (Saleva and Lignos, 2021; Liu et al., 2020; Ortega et al., 2020; Erdmann et al., 2019; Scherrer et al., 2019; Ataman and Federico, 2018; Banerjee and Bhattacharyya, 2018; Ataman et al., 2017), some focus on language modelling (Park et al., 2021; Bostrom and Durrett, 2020; Vania and Lopez, 2017), and several on syntax, morphology and semantics-related tasks, such as dependency parsing, tagging and entity typing (Durrani et al., 2019; Zhu et al., 2019). Typically, the segmentation methods chosen for comparison are BPE and Morfessor (or based on them).

In general, we can see that there is no clear preference towards BPE or Morfessor across previous works. Some papers claim that Morfessor and Morfessor-like models work better (Park et al., 2021; Bostrom and Durrett, 2020; Erdmann et al., 2019; Ataman and Federico, 2018; Ataman et al., 2017), while others report that better results are obtained with BPE and BPE-based algorithms (Ortega et al., 2020). Several papers conclude that there is no significant difference between BPE- and Morfessor-like models (Saleva and Lignos, 2021), and that there is no general recipe for subword tokenization that suits all languages and language pairs (Scherrer et al., 2019; Banerjee and Bhattacharyya, 2018; Vania and Lopez, 2017), downstream tasks (Zhu et al., 2019) and data size (Liu et al., 2020).

There have been attempts to create a hybrid segmentation system, which has both Morfessor and BPE components, hoping to achieve better results on the downstream tasks. Saleva and Lignos (2021) use a model MORSEL, which combines morphological rules together with the BPE algorithm; Ortega et al. (2020) introduce a BPE-Guided model which uses a filter for morphological affixes obtained from Wiktionary together with BPE; Banerjee and Bhattacharyya (2018) construct a model M-BPE, which first splits words using Morfessor and then applies BPE on the identified segments. While sometimes they work better, the general trend remains unstable, similar to the results of the baseline Morfessor and BPE.

In addition to these conflicting results, we also notice that the criteria for choosing the algorithms' hyperparameters and the assessment of their resulting subwords are often vague and based on the intuition of the authors. For example, the number of BPE merges is often set to several thousands or tens of thousands without particular explanation. Vania and Lopez (2017) note that they choose 10K merges for their English dataset after trying 1K, 10K and 100K and manually examining the resulting segments, because "10K gave the most plausible segmentation" (Vania and Lopez, 2017, p. 3). From the context, we can assume that the plausibility of the segmentation is perceived by the authors as the morphological segmentation they would do by hand, following their linguistic intuition.

There is less work that assesses the properties of the resulting subwords vocabularies directly, in

addition to the indirect comparison by means of downstream tasks (Sennrich et al., 2016; Bostrom and Durrett, 2020). The authors compare the size of the vocabularies (Sennrich et al., 2016) and the frequencies of tokens depending on their length (Bostrom and Durrett, 2020), and find that BPE reduces vocabulary better than Morfessor, and that the unigram LM method (similar to Morfessor) tends to produce longer tokens on average and makes more tokens of moderate frequency. On the contrary, BPE favors shorter tokens of higher frequency.

The work of Bostrom and Durrett (2020) exhibits similar intuitions and thoughts on this issue as those conveyed in (Vania and Lopez, 2017). In particular, Bostrom and Durrett (2020) refer to intermediate subwords (before 20K merges) produced by BPE as "junk" and "meaningless single-character units", implying that human-identified subwords are more desirable (Bostrom and Durrett, 2020, p. 4619–4620).

Our analysis offers a new way of reasoning about segmentation methods. It reveals high-level strategies that characterize human vs. automatic segmentation and shows where they diverge. These insights can be useful for making automatic methods more human-like when this is the goal.

3 Data and Methods

For this study, we need manually segmented texts of a reliable quality and with a high number of word types. To our knowledge, the best freely available dataset according to these criteria, is Aalto Morpho project[1], which provides segmentations for English, Finnish and Turkish. The dataset provides more than 1500 manually segmented word types in each language as well as unsegmented corpora with millions of tokens (see more details in Appendix A).

We use this data set for both extracting manual segmentation and training the algorithms. We train the models on the full word lists, where we multiply each word by its frequency, thus reconstructing the original corpora. Once the models are trained, we apply them to the unsegmented version of the gold standard list.

In total, we use four different settings in our experiments. In the first setting, we analyze the given segmentations produced by human annotators and call it **Manual**.

[1]http://morpho.aalto.fi/events/morphochallenge2010/datasets.shtml

For the second setting, we use the Morfessor Baseline model with the default hyperparameters and refer to this setting as **Morfessor**.

Next, we define two settings using the BPE algorithm with different stopping criteria. The first BPE setting follows standard approaches with a high number of BPE merges, defined as a function of the initial vocabulary size. In particular, we follow the approach introduced in (Mielke et al., 2019) and used in (Park et al., 2021), which maximizes the performance of the observed language models on a sample of 64 languages. The number of BPE merges is calculated as $0.4 \times |V|$, where $|V|$ is the size of the word types vocabulary in a tokenized raw text (training data). The exact number of types, tokens and BPE merges per language are listed in Appendix A, B. We call this setting **BPE-V**.

The second BPE setting follows Gutierrez-Vasques et al. (2021) in defining a low number of merges, which gives similar information theoretic properties across languages. The idea is to stop merging at the 200th iteration step, because languages tend to minimize redundancy of their subwords' vocabularies at that step. We make this method more equivalent to BPE-V by grouping together the remaining consecutive single units after the merge 200. For example:

p@@ h@@ il@@ i@@ p@@ p@@ in@@ es
ph@@ il@@ ipp@@ in@@ es

The sequence in the first line is the original BPE segmentation after 200 merges and the sequence in the second line is our modification.

Typically, NLP practitioners do not use fewer than 1K merges, however we consider this setting to be especially contrasting to the standard, vocabulary-based approach, and it suits well to our purposes. We refer to this setting as **BPE-200**.

For BPE training and testing, we use a well-known subword-nmt library (Sennrich et al., 2016), and for Morfessor, we use the baseline model provided in the Morfessor 2.0 library (Smit et al., 2014).

3.1 Unevenness Index (UI)

The gist of our approach lies in abstracting from the actual words to their underlying structure represented as a sequence of subword lengths. We assume the universal measure of word length to be characters. Once we map characters to numbers, where each character has a length of 1, we can operate with mathematical notations.

In particular, we observe the resulting sequences of segments as *line* segments. By envisioning each segment as a line segment, we can measure their lengths and relative proportions. We distinguish between **even** and **uneven** sequences of segments and compare their distributions across languages.

For example, given several words consisting of 10 characters, we can get segmentations, which are characterized by different *evenness*: 1) one segment of length 7 and two segments of length 2 and 1, 2) 5 segments each of length 2 or 3) 10 segments of lengths within a narrow range [1, 2, 3, 1, 2, 1] (Figure 1).

In order to formalize the property of evenness, we introduce the following *unevenness index (UI)*: $UI = max(segments) - min(segments)$, where *segments* is a set of segments' lengths for a given word. The lower the UI, the more even the segmentation. When the UI equals zero, all segments are of equal length, the segmentation is strictly even. The segmentations on the Figure 1 are quantified by the UI as: (1) $7 - 1 = 6$, (2) $2 - 2 = 0$ and (3) $3 - 1 = 2$.

Despite its seeming simplicity, our measure represents the intended properties well. It is highly correlated with more complex measures, such as variance. Pearson correlation between variance and the UI lies in the range from 0.561 (Finnish, BPE-200) to 0.947 (English, Morfessor). All correlations are provided in Appendix B.

Our main analysis consists of establishing a relationship between segmentation evenness and word length. We observe the properties of the distribution and formalize them by applying specific metrics.

Since the UI depends on word length, it has an upper bound. The highest value for a certain word length L is calculated as $(L-1)-1 = L-2$, where $(L-1)$ is the biggest possible segment length for a given word length and 1 is the smallest one. Thus, the whole distribution has a linear bound $y = x-2$, where x is the word length and y is the UI.

3.2 UI Density Analysis with Kernel Density Estimation (KDE)

To compare UI distributions across different settings and languages, we create a scatter plot (word length $\times$ UI). We first identify the density areas using Kernel Density Estimation (KDE) and then extract a single numerical attribute (KDE upper angle) for each distribution. In the remainder of this section, we describe these two steps in more detail.

First, we find the densest area by applying the KDE algorithm, which provides a non-parametric way to estimate the probability density function of a distribution. Since we do not know the properties of our distribution in advance, KDE is a good choice to estimate the density function and thus highlight the most densely populated areas.

After applying the kernel function to every single point in the data, all the n kernel functions are summed and divided by the number of data points. The Equation 1 describes the standard KDE procedure. The standard KDE algorithm operates on one dimension, and uses kernel as a weighting function:

$$\hat{f}(x) = \frac{1}{nb} \sum_{j=1}^{n} K(\frac{x - x_j}{b}) \quad (1)$$

where $x_1, ..., x_n$ are the data points, K is a kernel function and b is a bandwidth (Ripley, 2002, pp. 126).

The only parameters to be chosen in KDE are the kernel function and the bandwidth. The kernel function K is typically chosen to be a probability density function. The bandwidth b determines the width of the smoothing window. Since our data is two-dimensional (capturing the dependence between the UI and word length), we perform a two-dimensional kernel density estimation:

$$\hat{f}(x, y) = \frac{\sum_{j=1}^{n} K((x - x_j)/b_x) K((y - y_j)/b_y)}{n b_x b_y} \quad (2)$$

where K is the kernel, b_x and b_y are bandwidths in both directions. The kernel is aligned to the axes and evaluated on a square grid. (Ripley, 2002, pp. 130–131)

We use the R implementation of the two-dimensional KDE, in particular the function kde2d() from the MASS package[2] with the default kernel and bandwidth settings. The kernel we use is the standard normal density function. The bandwidth values are the same for b_x and b_y, and are estimated using Silverman's rule-of-thumb (Silverman, 1986, p. 48, eqn. 3.31), which depends on the interquartile range and the standard deviation of the kernel. For visualization, we use the package ggplot2 in R, in particular the function geom_density_2d_filled, which invokes the

[2] https://cran.r-project.org/web/packages/MASS/index.html

function kde2d(). The function performs a two-dimensional kernel density estimation and, depending on the level of density, fills the identified areas with different colors. We set three bins of density, where the areas of our interest show the density core and its outline. The resulting plots are shown in Figure 2.

3.2.1 KDE upper angle

Once the density area is identified, we fit to it two diagonals (consider the diagonal lines in Figure 2). The left diagonal is determined by the upper bound function $y = x - 2$. For the right diagonal, we first identify the two extreme points inside the dense region: the point with the maximum x and the maximum y values. If there are several points with the maximum value, we take the first point in the dataset. Once the two points are found, we construct another linear function.

The angle resulting from the intersection of the diagonals is our measure of the UI preferences in longer words. The more obtuse the upper angle, the more dense the region of low UI values in longer words, meaning that there is a strong preference for even segmentations. In contrast, when the angle is less obtuse, the longer words with low UI values are rare, and the preference for the UI in longer words is either undefined or inclined towards uneven segmentations. The choice of this particular angle is also motivated by the Menzerath-Altmann's law (Menzerath, 1954, p. 101), according to which the longer a word, the smaller the parts. Thus, the UI values for longer words are expected to be low.

Note that we introduce a slight noise in the calculation of the KDE upper angle, so the value can differ in +-3 degrees. For this reason, we take the measurements 100 times and report the mean results. The plots in Figure 2 represent the closest value to the mean, found during 100 iterations.

3.3 UI Surface Coverage

Another way of analyzing the UI distribution is to observe all the manifested pairs of the UI and word length values versus all the combinatorial possibilities. This concept we call the surface coverage.

In order to establish the space of all the combinatorial possibilities of the UI and word length values, we first identify the point with the maximum word length for a given language (the rightmost point in Figure 2). This point serves to identify the right vertical side of the surface coverage triangle, that is, an x-intercept line $x = max(x)$. The left side of this triangle is determined by the linear upper bound function $y = x - 2$. This is the diagonal upper bound on the UI distribution described above.

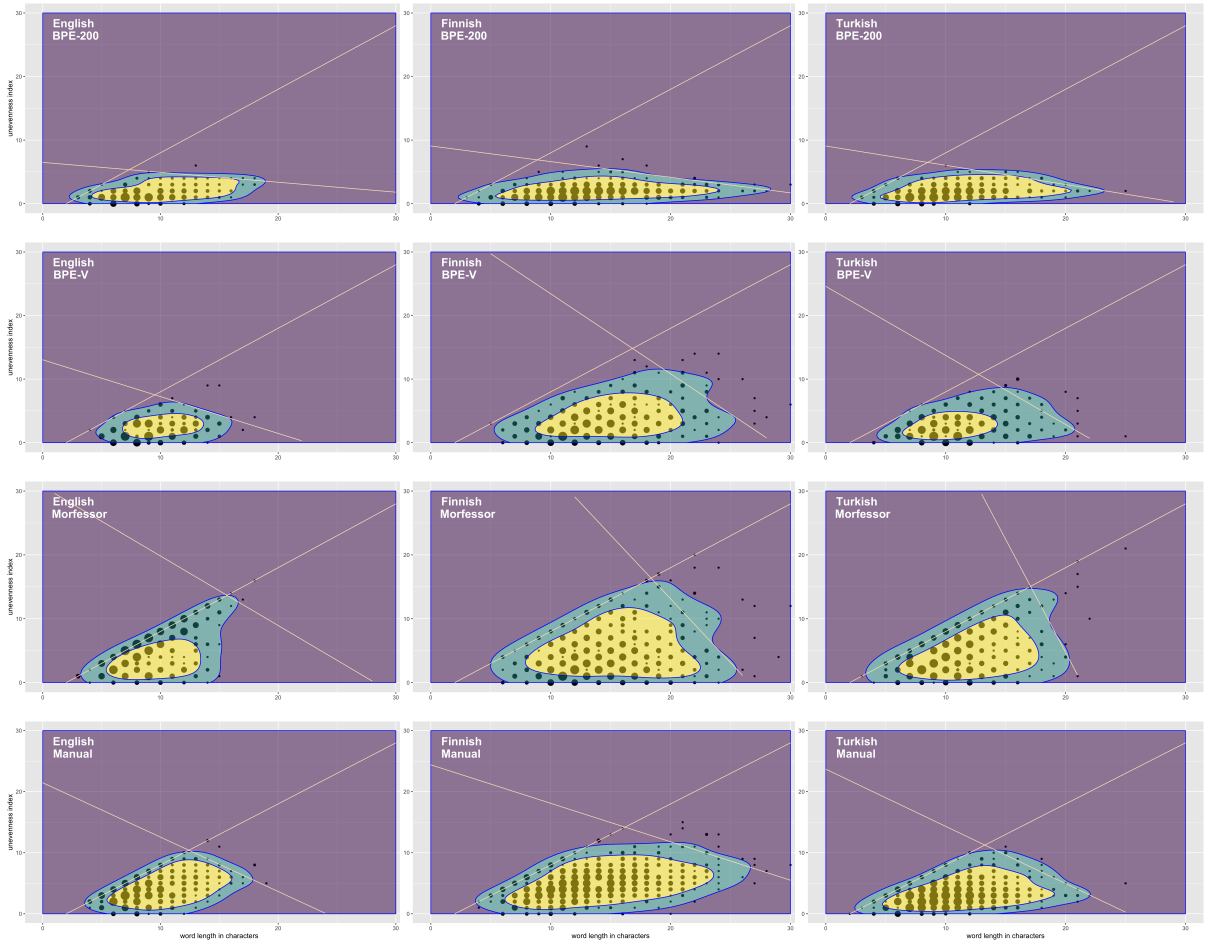
Once the sides of the triangle are established, we can quantify the filled area. The value of the surface coverage is calculated as a division of the manifested points by all the points inside the triangle. Thus, the highest possible value is 1, when the whole triangle is filled, and the lowest value is close to 0. The resulting triangle shows the tendency of a particular setting and language to occupy the combinatorial possibilities of the UI and word lengths.

3.4 UI Frequency Analysis

The last analysis that we introduce captures the segmentations' properties independent of word length, and continues the work presented in (Bostrom and Durrett, 2020). We fit the function $y = ax^{-}b$ from the power law family to the UI values and their frequencies, across all word lengths. The parameter a describes the slope of the curve, and the parameter b is connected to the concaveness of the curve. Since the parameter a is highly correlated to the overall frequency, we concentrate on the parameter b and refer to it as to a frequency parameter.

The parameter b shapes the tail of the distribution and makes the turn of the curve more or less convex. In our case, we observe only negative values of b (decreasing curves), thus, we incorporate the minus sign already in the formula, and discuss the absolute values of b. When the value of b is close to zero, the frequencies in the tail of the distribution get higher, the shape of the turn is less convex. The bigger the value of b, the lower the tail of the distribution, and the more convex the turn of the curve.

Since shorter words have few combinations of possible subword splits, the lowest values of the UI are expected to be the most frequent: both shorter and longer words fall into this category of values. In contrast, larger values of the UI can only be present in longer words (indicating uneven segmentations). Thus, the frequency in the tail region of the UI distribution is of more interest for us. Note that shorter words tend to be the most frequent in general (Zipf, 1935), which also supports our choice of the function.



(a) English (b) Finnish (c) Turkish

Figure 2: KDE upper angle measured on (a) English, (b) Finnish and (c) Turkish data. From up to down: BPE-200, BPE-V, Morfessor and Manual. The **size** of the points depends on the frequency of the given UI for the given word length. The colors depict the levels of density: the most densely populated area is in yellow, the second dense area is in blue, and the least dense area is in purple.

4 Results

4.1 Human segmentations are moderately uneven

The main outcome of our experiments is that none of the usual algorithms patterns with human segmentation. We can see in Figure 2 what makes manual segmentation optimal. The longer the words, the more consistent the UI values: they neither stick to the bottom as BPE-200 (predominantly even splits), nor go too high as in the case of Morfessor (predominantly uneven splits).

We observe an interesting pattern regarding the different levels of density. Here, manual segmentation patterns with BPE-200 in well-defined density regions (large yellow, small blue areas), while relatively big blue areas in BPE-V and Morfessor indicate poor density, i.e. little consistency.

Figure 3 shows the distribution of the KDE upper angle values. It confirms that the manual setting is indeed somewhere in the middle compared to the algorithms: smaller than BPE-200 and bigger than the other two. Very uneven segments result in the lowest values for Morfessor here, and especially for Finnish and Turkish, which is somewhat surprising given that Morfessor is originally conceived for processing such languages.

Figure 4 shows the UI surface coverage, where the manual setting again seems remarkable. It has the highest values of the UI surface coverage across settings. This means that humans use the most versatile splits for subword segmentations, thus filling the combinatorial space the most. Apparently, humans are more creative in how to deal with longer words, and allow subwords of various lengths.

In this sense, Morfessor seems to be the closest

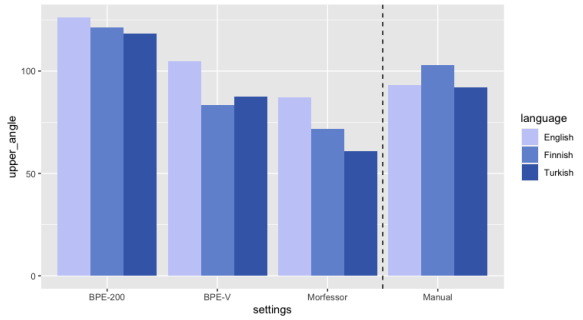


Figure 3: Distribution of the KDE upper angle values.

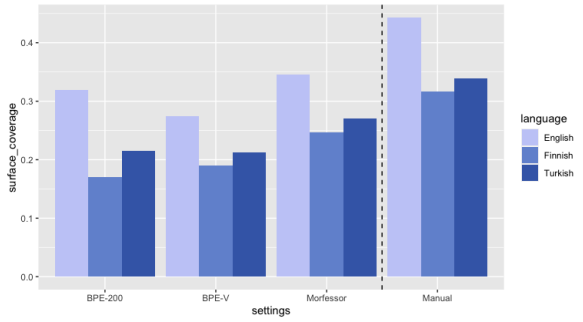


Figure 4: Distribution of the UI surface coverage values.

to manual segmentations, yet it does not reach the UI surface coverage of humans.

Figure 5 shows the frequency analysis indicating that manual segmentations fit the power law function the least (the curve is least convex), while algorithms show better fits. Here too, Morfessor is the closest to humans, especially on English and Turkish. While the patterns apply across languages in all the other settings, the manual annotation of Finnish in this setting looks different. It shows that human annotators choose uneven splits more often, causing the tail of the frequency function to be higher.

From our results, we can see that the general recipe for getting closer to human linguistic intuition is to seek uneven segmentations but not allow too much variation. A way to achieve this could be by means of parameter tuning or by hybrid strategies combining first BPE merges (as in BPE-200) with some additional models or strategies for longer subwords.

4.2 BPE-V is closer to Morfessor, than to BPE-200

The second major outcome of our study, which sheds light on the absence of "one-fit-all" solution, is the similarity of BPE-V and Morfessor's segmentation strategies. From the Figure 2 we can notice that both of these methods are prone to noise, especially in the agglutinative languages Finnish and Turkish. This is particularly unexpected, since the morphological structure of these languages allows finding more regular subwords, yielding better-defined density regions. We speculate that fine-tuning of Morfessor could reduce the noisy zone (blue area) and provide better results on Turkish, as shown in (Ataman et al., 2017; Ataman and Federico, 2018).

English processed by BPE-V and Morfessor looks the most distinct: Morfessor creates much more frequent uneven splits, while BPE-V shows a stronger tendency towards even splits (sticking to the x-axis). Manual segmentation is, as discussed in Section 4.1, somewhere in between.

In terms of the UI surface coverage, all of the algorithms are close to each other, showing only minimal differences. When it comes to the UI frequency parameter, BPE-V and Morfessor are much closer to each other (mean values 0.86 and 0.61) than to BPE-200 (mean value 1.28). One more statistic that confirms this pattern is the mean subword length. Here too, BPE-V and Morfessor pattern together (mean subword length across languages is 6.974 in BPE-V and 6.709 in Morfessor) and opposite to BPE-200 (1.894). As expected from the other results, the manual setting is in between with the value of 3.673. (Full details in Appendix A.)

From this analysis, we conclude that it is highly unlikely to see a consistent winner when comparing BPE-V and Morfessor (or similar models). They are alike in many respects, yet both rather inconsistent and unpredictable. The same suggestions as in 4.1 should improve the consistency in their evaluation too.

4.3 BPE-200 as a new strategy

BPE-200 stands out in comparison to the other algorithms. Its UI density is well-defined, as in the manual setting, but its position is very different from both manual and other algorithms. Compared to humans, BPE-200 exaggerates on even splits, making the majority of segmentations look like the middle pattern in Figure 1 (pattern 2). This property sheds new light on the findings by Gutierrez-Vasques et al. (2021). At the same time, Figure 3 shows that BPE-200 produces the most similar and predictable distributions of the UI across lan-

guages, which is behavior that can be expected from the information-theoric properties discussed by Gutierrez-Vasques et al. (2021). This suggests that the number of BPE merges should be lower than 1K when working on multilingual samples.

The BPE-200 setting shows an interesting difference in the UI frequency analysis as well. In all the languages, the parameter b of the function $y = ax^{-}b$ gets the highest absolute values. This means that the smallest values of the UI are the most frequent, the tail of the distribution is low and infrequent, and the turn of the curve is prominently convex. Such curves are typical for natural distributions, such as in Zipf's law applied to word tokens (Zipf, 1935). In manual segmentation, we do not see this effect because of the poor function fit: the most frequent UI in this setting is not 0, as expected for a peak in the power law function, but is situated on the UI of 3 (English, Turkish) and 4 (Finnish) (see plots example in Appendix C, Figure 6). While the UI value of 0 is the most frequent in all of the algorithms, BPE-200 is the only one showing a clear peak in frequencies, while BPE-V and Morfessor tend to smooth out the peak and lengthen the tail.

We conclude from this analysis that BPE-200 might be a fruitful subword tokenization method on multilingual datasets, but further experiments on downstream tasks are needed in order to confirm this.

5 Discussion

All our analyses are based on measuring the length of subwords in characters. One might argue that this works well only for alphabetic scripts. We underline that the same approach can be applied to other scripts as well if the same convention is defined for measuring subwords' length across all the settings.

Thus, if we analyze Hindi, and our length unit is a syllable, we can compare it to other languages with Brahmic scripts with no problem. It might be more problematic to compare languages of different scripts to each other, however. The bigger the unit of measure in the sense of information load (character < syllable < logograph), the smaller the possible combinatorial space for the analysis. Thus, Chinese would take only the UI values up to 4-5 because words consisting of more than 5-6 characters are extremely rare. However, we do not expect the findings established in our analyses to change much due to these differences; if needed, they can be measured (Sproat and Gutkin, 2021) and added to the calculations.

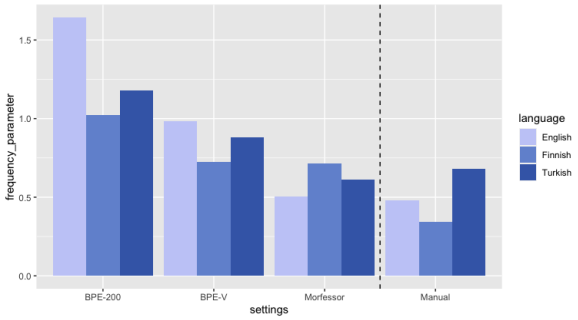


Figure 5: Distribution of the UI frequency parameter values.

This limitation might impact more fine-grained analyses in future work. For now, we note that the majority of languages use alphabetic writing systems. In the future, we plan to extend our approach to the other scripts and try to find smaller, more comparable units in non-alphabetic scripts (e.g. strokes in Chinese characters (Prün, 1994)).

Another direction of our work includes extraction of the other attributes, e.g. KDE right angle. This metric can help to compare segmentations in more detail and quantify humans' moderate choice of uneven splits.

6 Conclusion

In our study, we introduced a novel method to analyze and compare various approaches to subword segmentation using spatial analysis of the subwords' lengths. We demonstrated that humans produce indeed the most optimal segmentation, in contrast to all algorithms which tend to be biased towards some values and are also inconsistent. We argue that human segmentation, which is neither too even nor too uneven (Figure 1 (3)), can be imitated better by means of hyperparameters tuning in combination with the low-noise solution shown by BPE-200.

We discovered that the usual BPE settings (BPE-V) are close to Morfessor and tend to produce uneven splits (Figure 1 (1)), while BPE-200 strongly prefers even splits (Figure 1 (2)). The even patterns produced by BPE-200 make subword lengths very similar across languages. This could be exploited for better cross-linguistic representations in future work.

References

Farrell Ackerman and Robert Malouf. 2013. Morphological organization: The low conditional entropy conjecture. *Language*, pages 429–464.

Stephen R Anderson, D Meyer, D Platt, and D Ridley. 1992. *A-morphous morphology*. Citeseer.

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.

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.

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. 2021. Canine: Pre-training an efficient tokenization-free encoder for language representation. *arXiv preprint arXiv:2103.06874*.

Mathias Creutz and Krista Lagus. 2005. Inducing the morphological lexicon of a natural language from unannotated text. In *In Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning (AKRR'05*, pages 106–113.

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, Minneapolis, Minnesota. Association for Computational Linguistics.

Nadir Durrani, Fahim Dalvi, Hassan Sajjad, Yonatan Belinkov, and Preslav Nakov. 2019. One size does not fit all: Comparing NMT representations of different granularities. 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 1504–1516.

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.

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.

Harald Hammarström and Lars Borin. 2011. Unsupervised learning of morphology. *Computational Linguistics*, 37(2):309–350.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. *arXiv preprint arXiv:1804.10959*.

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*.

Peter H. Matthews. 1991. *Morphology*, 2 edition. Cambridge University Press.

Paul Menzerath. 1954. *Die Architektonik des deutschen Wortschatzes*, volume 3. F. Dümmler.

Sabrina J Mielke, Ryan Cotterell, Kyle Gorman, Brian Roark, and Jason Eisner. 2019. What kind of language is hard to language-model? *arXiv preprint arXiv:1906.04726*.

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.

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.

Claudia Prün. 1994. Validity of Menzerath-Altmann's law: Graphic representation of language, information processing systems and synergetic linguistics. *Journal of Quantitative Linguistics*, 1(2):148–155.

Brian D Ripley. 2002. *Modern applied statistics with S*. Springer.

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.

Bernard W Silverman. 1986. *Density estimation for statistics and data analysis*. London: Chapman and Hall.

Peter Smit, Sami Virpioja, Stig-Arne Grönroos, Mikko Kurimo, et al. 2014. Morfessor 2.0: Toolkit for statistical morphological segmentation. In *The 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL), Gothenburg, Sweden, April 26-30, 2014*. Aalto University.

Richard Sproat and Alexander Gutkin. 2021. The taxonomy of writing systems: How to measure how logographic a system is. *Computational Linguistics*, pages 1–54.

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

Sami Virpioja, Ville T Turunen, Sebastian Spiegler, Oskar Kohonen, and Mikko Kurimo. 2011. Empirical comparison of evaluation methods for unsupervised learning of morphology. *Trait. Autom. des Langues*, 52(2):45–90.

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*.

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

George Kingsley Zipf. 1935. *The psycho-biology of language: An introduction to dynamic philology*. Mifflin.

A Data Metrics

Language	Word list (tokens)	Word list (types)	Gold standard
English	221,715,953	878,036	1686
Finnish	65,734,026	2,928,030	1835
Turkish	12,862,393	617,298	1760

Table 1: Number of tokens and types in the dataset.

Language	BPE-200	BPE-V	Morf	Man
English	1.894	6.974	6.709	3.673
Finnish	1.993	9.184	8.159	3.813
Turkish	1.991	7.011	6.573	3.149

Table 2: Mean length of subwords in the different settings. Morf stands for Morfessor, Man stands for Manual.

B Calculation Details

Language	BPE-V merges
English	351,214
Finnish	1,171,212
Turkish	246,919

Table 3: Number of calculated BPE merges for the setting BPE-V. The values are rounded to integers.

Language	BPE-200	BPE-V	Morf	Man
English	0.788	0.876	0.947	0.869
Finnish	0.561	0.914	0.924	0.784
Turkish	0.716	0.925	0.929	0.827

Table 4: Correlation of the UI with variance across languages and settings.

Language	BPE-200	BPE-V	Morf	Man
English	45,660	2,212	2,349	105,437
Finnish	377,727	11,001	14,048	198,374
Turkish	149,425	8,206	13,535	121,346

Table 5: Residual sum of squares for the frequency function fit, lower is better.

C Results: Additional Tables and Figures

Language	BPE-200	BPE-V	Morf	Man
English	126.134	104.750	87.052	93.307
Finnish	121.247	83.498	71.706	102.895
Turkish	118.110	87.552	60.754	91.871

Table 6: KDE upper angle values in English, Finnish and Turkish.

Language	BPE-200	BPE-V	Morf	Man
English	0.319	0.275	0.346	0.443
Finnish	0.170	0.190	0.247	0.317
Turkish	0.215	0.212	0.271	0.339

Table 7: UI surface coverage values in English, Finnish and Turkish.

Language	BPE-200	BPE-V	Morf	Man
English	1.643	0.985	0.503	0.482
Finnish	1.022	0.723	0.714	0.344
Turkish	1.177	0.883	0.613	0.681

Table 8: UI frequency parameter (b) values in English, Finnish and Turkish.

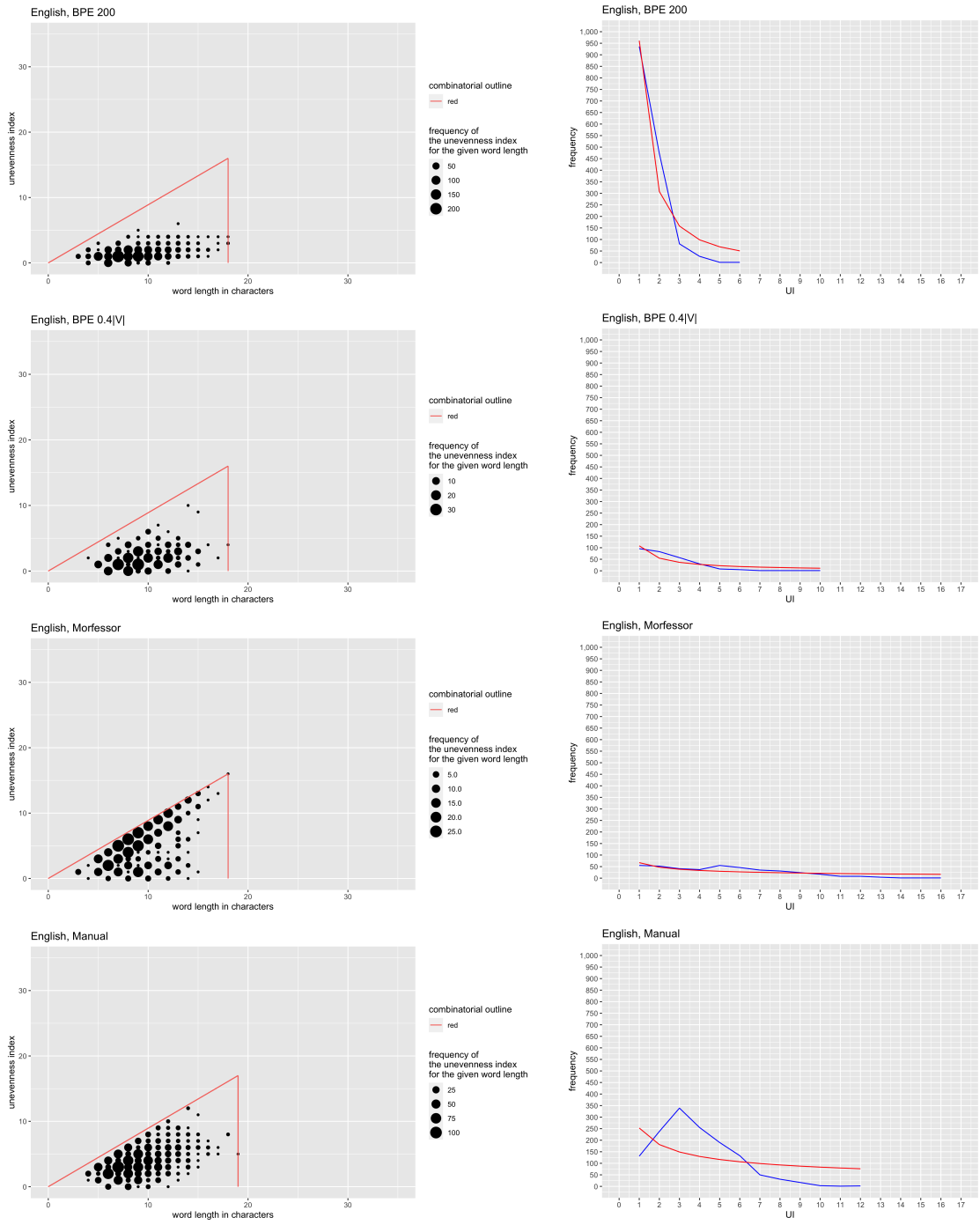


Figure 6: UI surface coverage and UI frequency parameter distribution in English across different settings. From up to down: BPE-200, BPE-V, Morfessor and Manual. On the right, real data is shown in blue, function $ax^{-}b$ in red.