ByteSpan: Information-Driven Subword Tokenisation

Anonymous Authors[1]

Abstract

Recent dynamic tokenisation methods operate directly on bytes and pool their latent representations into *patches*. This bears similarities to computational models of word segmentation that determine lexical boundaries using spikes in an autoregressive model's prediction error. Inspired by this connection, we explore whether grouping predictable bytes—rather than pooling their representations—can yield a useful fixed subword vocabulary. We propose a new information-driven subword tokeniser, ByteSpan, that uses an external byte-level LM during training to identify contiguous predictable byte sequences and group them into subwords. Experiments show that ByteSpan yields efficient vocabularies with higher morphological alignment scores than BPE for English. Multilingual experiments show similar compression and Rényi efficiency for 25 languages.

1. Introduction

Modern language models (LMs) process text as sequences of *byte*[1] *spans*, or **subwords**, to improve computational efficiency (Zouhar et al., 2023). Processing raw bytes leads to longer sequences, while operating on words requires a large vocabulary. Subword tokenisation—i.e., grouping bytes into subwords drawn from a fixed, finite vocabulary—offers a balance but is sensitive to spelling (Chai et al., 2024) and has inconsistent compression rates across languages (Rust et al., 2021).

To remove the LMs' dependence on tokenisers while preserving computational efficiency, recent work on tokenisation proposes to operate directly on bytes and pool their representations into *patches*. These patches are created either by pooling fixed-length spans (Dai et al., 2020; Nawrot et al., 2022; Yu et al., 2023, *inter alia*) or by dynamically pooling predictable byte sequences within a context window (Nawrot et al., 2023; Pagnoni et al., 2024).

Dynamic patching relies on trainable model components and unwittingly mirrors work in child language acquisition, where computational models—ranging from n-grams to Transformers (Vaswani et al., 2017)—are used to study how children may use distributional statistics to group predictable sequences of *phonemes* into words. As with dynamic patching, these methods draw on the simple principle that predictability within lexical units is high, and predictability between lexical units is low (Harris, 1955). Inspired by this connection, we explore whether grouping predictable bytes—rather than pooling their representations—can yield a useful fixed subword vocabulary.

We propose a new information-driven tokeniser, ByteSpan, which uses an external byte-level LM to identify predictable contiguous byte sequences, using entropy or surprisal as measures of information. Byte spans are identified using a global threshold constraint, a monotonic constraint, or a combination of the two, and we propose several methods for using byte spans identified in a training corpus to create a subword vocabulary. We use these methods to train English and multilingual tokenisers and compare them to Byte-Pair Encoding (BPE, Sennrich et al., 2016) across vocabulary sizes. Intrinsic results show that our method yields higher morphological alignment scores and higher Rényi efficiency scores for most vocabulary sizes, without compromising compression. Our multilingual experiments show similar Rényi efficiency and fertility to BPE across 25 languages and we propose methods for balancing a vocabulary to efficiently tokenise rare orthographies.

2. ByteSpan Tokenisation

Our method, ByteSpan, uses an external byte-level LM during training to identify predictable byte sequences and group them into subwords. During inference, the resulting vocabulary can be paired with any standard tokenisation function (e.g., longest-prefix match) and modern LM (e.g., Touvron et al., 2023) without modifications.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

[1]We use *bytes* and *characters* interchangeably.

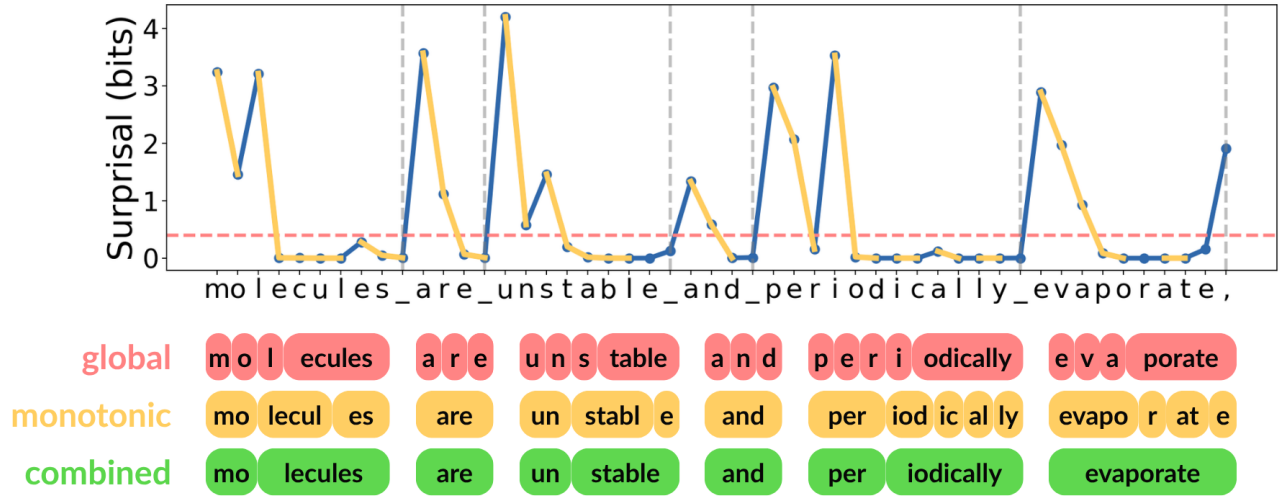

Figure 1. **Information-Driven Subword Creation.** Per-byte **surprisal** of *"molecules are unstable and periodically evaporate"* from a byte-level LM. ByteSpan groups contiguous bytes using one of three constraints; the global constraint uses a fixed threshold, the monotonic constraint groups bytes with decreasing information and the combined constraint groups bytes that meet either constraint. Grey vertical lines indicate pre-tokenisation boundaries.

2.1. Motivation

This method is related to the work of Pagnoni et al. (2024), who group bytes into dynamic 'patches' according to their entropies, as given by a byte-level LM. They experiment with two methods to identify patch boundaries; their **global constraint** identifies bytes whose entropies exceed a fixed threshold and their **monotonic constraint** identifies bytes whose entropies decrease monotonically. A **patching function** then segments a stream of bytes into patches that are fed through a latent transformer, with predicted patches decoded by the smaller byte-level LM. Conceptually, this smaller LM can make the relatively 'easy' next-byte predictions given the low entropy of each byte within the patch.

We draw parallels between these patching constraints and computational models of word segmentation. These models are designed to demonstrate how distributional information could be leveraged by language-learning infants to bootstrap a vocabulary, following the influential statistical learning experiments of Saffran et al. (1996) who observed this ability in young infants. In the typical framework, these models use unsupervised algorithms to group unsegmented sequences of phonemes from transcriptions of child-directed speech into word-like units. One approach involves maximising the likelihood of word-level n-gram models(e.g., Brent, 1999; Venkataraman, 2001), a method that closely resembles the UnigramLM tokenisation algorithm (Kudo, 2018). Another approach is to extract measures of uncertainty using phoneme-level n-gram models and posit boundaries where these measures spike or surpass a threshold (e.g., Çöltekin & Nerbonne, 2014; Goriely et al., 2025). Neural language models have also been used, for instance by using the prediction of an utterance boundary (which is included in the phoneme sequence) to posit a word boundary (Christiansen et al., 1998). In a formative analysis of character-level RNNs, Elman (1990) noted that the prediction error from a neural language model could serve as a cue for lexical boundaries more broadly; boundaries around words, morphemes, but also frequent multi-word sequences, which children often treat as fixed lexical items (MacWhinney, 1978). Based on this observation, Goriely & Buttery (2025) trained phoneme-level GPT-2 LMs across 31 languages and demonstrated a method for extracting word boundaries from the trained models; computing model uncertainty using entropy, rank and surprisal from the predictions at each point and segmenting at points of high uncertainty. These information-based approaches to word segmentation, and the later study in particular, closely resemble the patching constraints of Pagnoni et al. (2024).

We hypothesise that a modular tokenisation pipeline using constraints based on byte-level information might retain the conceptual advantages of patch-based approaches while retaining the benefits of creating a fixed vocabulary as a pre-processing step before training. By drawing parallels with computational models of word segmentation, we hypothesise that the resulting subword tokens align better with morphological segmentations of natural language than compression-based methods such as BPE.

2.2. The ByteSpan Algorithm

ByteSpan works by first collecting predictions from a byte-level LM over a corpus, from which key statistics are calculated, then grouping contiguous sequences of bytes using a

constraint based on these statistics.

Statistics and constraints. As a starting point, we follow the patching methods of Pagnoni et al. (2024) by using per-byte **entropy**[2] $H(b_i)$ and considering two constraints:

$$\text{Global Constraint} \quad H(b_t) < \theta_g \tag{1}$$

$$\text{Monotonic Constraint} \quad H(b_t) - H(b_{t-1}) < 0 \tag{2}$$

The first constraint groups bytes that fall under a fixed global threshold and the second constraint groups byte with monotonically decreasing information. These are visualised in Fig. 1, where for instance the global constraint segments the word *"unstable"* as {*"u"*, *"n"*, *"s"*, *"table"*} whereas the monotonic constraint produces {*"un"*, *"stabl"*, *"e"*}. The algorithm for segmenting a sequence using the monotonic constraint is provided in *Algorithm* 1. Each byte is processed once in a single pass through the sequence, so the complexity is $O(n)$.

In addition to entropy, we also explore the use of **surprisal** as our information signal, noting that ByteSpan is compatible with any function mapping from the LM's logits to a scalar. We also consider a third constraint that combines the global constraint with the monotonic constraint, grouping contiguous bytes that meet either. This follows from the observation that the monotonic constraint can become unstable when the entropy or surprisal of a byte is close to zero. This can be seen from the small increase in surprisal from *"l"* to *"e"* causing *"stable"* to be split into two units in the example given in Fig. 1. In such cases, the combined constraint joins segments below a low threshold θ_g with monotonically decreasing segments, potentially preventing these unwanted segmentations.

The global constraint aims to capture highly predictable sequences. However, we find that it often results in splitting words into single byte tokens followed by the remainder of the word, and we find that the monotonic constraint is superior at recovering morphological and lexical units. This follows Elman (1990)'s observations that model uncertainty typically spike at such boundaries.

Learning a vocabulary. We propose three methods for using these constraints to learn a fixed-size vocabulary V from a training corpus $\mathcal{D} = \{\mathbf{b}_n\}_{n=1}^{N}$:

1. **Frequency:** Using any of the three constraints, identify all unique subwords in the training corpus. Sort the subwords by frequency and use the top $|V|$ as the tokeniser's vocabulary.

2. **Incremental:** Using the global constraint, gradually increase θ_g until the desired vocabulary size is reached. To prevent rare subwords from being added to the vocabulary, a minimum frequency threshold θ_f can also be applied.

3. **Seeding BPE:** Using any of the three constraints, apply the **frequency cutoff** method to learn a portion $p\%$ of the final vocabulary, then apply BPE to learn the rest of the vocabulary.

[2]We note that Pagnoni et al. (2024) actually use *next-byte* entropy, which would shift our byte spans by one unit.

Algorithm 1 ByteSpan Tokenisation

Input: Byte sequence $X = b_1, b_2, \ldots, b_n$, Byte-level entropy values $H(b_i)$
Output: Tokenized sequence T
Initialize $i \leftarrow 2$ {Start of the byte sequence}
Initialize $T \leftarrow []$ {Empty tokenized sequence}
while $i \leq n$ **do**
  $j \leftarrow i$
  while $j \leq n$ **and** $H(b_j) - H(b_{j-1}) < 0$ **do**
    $j \leftarrow j + 1$
  end while
  Extract segment $b_i, b_{i+1}, \ldots, b_j$
  Append the segment to T
  $i \leftarrow j$
end while
Return: T

The frequency method is the most efficient, requiring only a single pass of the training dataset. However, for the global constraint, it requires pre-determining the global threshold θ_g. The incremental method gets around this limitation by gradually increasing the threshold until the desired vocabulary size is reached. Unlike the other methods, this means that vocabularies with a larger $|V|$ will not necessarily contain the vocabularies of tokenisers trained with a small $|V|$. This is because higher thresholds lead to the **absorption** (or subsumption) of constituent subsequences. For instance, increasing the threshold in Fig. 1 would lead to *"nstable"* replacing *"table"* in the vocabulary.

In theory, the incremental method is also compatible with the combined constraint, but in practice, the number of subwords identified by the monotonic constraint exceeds most desired vocabulary sizes at the first pass, causing the algorithm to terminate immediately without the global constraint applying.

Finally, we theorise that the seeding method could provide a trade-off between ByteSpan and BPE by first identifying predictable multi-byte units and then using BPE to efficiently compress frequently co-occurring units. We note that setting $p = 100\%$ is equivalent to the frequency method and that setting $p = 0\%$ is equivalent to just using BPE to learn the vocabulary.

ByteSpan *vs.* BPE. Unlike BPE, which merges tokens incrementally based on frequency, ByteSpan finds contiguous low-information segments. Our vocabulary-learning methods are flexible, achieving a target vocabulary size by either incrementally increasing the information threshold to include longer and less predictable sequences, or by trimming down a large set of discovered units according to frequency.

By only including the longest subsequences determined by the constraints used, we avoid intermediate merges, unlike BPE. Since a sequence cannot be decomposed using a recursive BPE-style inference procedure, we rely on **longest-prefix matching** as used by WordPiece (WP, Schuster & Nakajima, 2012). Notably, ByteSpan can also group beyond word boundaries and create **superwords** (e.g., if the threshold were raised in Fig. 1, *"nstable and"* could be added as a single token). Our implementation ensures that learned subwords align with BPE pre-tokenisation constraints by preventing subwords from crossing pre-tokenisation boundaries (the vertical lines in Fig. 1).

ByteSpan *vs.* Patching. ByteSpan retains the information-driven approach of patching while preserving the computational benefits of keeping tokenisation separate from language modelling. In particular, it eliminates the additional complexity of batching in patch-based methods (e.g., where patch boundaries do not align across sequences). Our method only requires computing the entropy (or surprisal) of bytes once in order to learn a fixed $\mathcal{V}$ before training, which can then be used by standard LMs with any corpus, whereas the patching method of Pagnoni et al. (2024) requires a byte-level LM to compute the entropy of every byte seen during training. Unlike our method, patching does not create a fixed size vocabulary, as patches are dynamically created during training.

3. Experimental Setup

We explore our information-driven tokenisation approach in both English and multilingual settings. Below, we briefly describe the experimental setup. We provide additional implementation details in App. A.

Data. We train the English tokenisers on a sample of the FineWeb-Edu dataset[3] (Penedo et al., 2024). For the multilingual tokenisers, we use a balanced 25-language sample from the Common-Corpus[4] (Langlais, 2021), with languages selected for morphological diversity. We convert each corpus into bytes using the Huggingface `ByteLevel` pre-tokenizer and split each corpus into three equal subsets: one to train the byte-level model, one to train the tokenizers, and one for evaluation. The FineWeb-Edu subsets contain approximately 500M bytes and the Common-Corpus subsets contain approximately 250M bytes (10M per language).

Byte-level Model. The byte-level LM is based on the Llama-2 architecture (Touvron et al., 2023) and we use it to collect the contextual surprisal and entropy for each byte in our corpora. The byte-level statistics only need to be collected once and can be reused for each tokeniser setup.

Tokeniser Setup. For our English tokenisers, we train a suite of tokenisers across three vocabulary sizes; 16k, 32k and 64k. For the multilingual setting, we use a vocabulary size of 128k. For each vocabulary size, we train tokenisers using our three constraints and our two measures (entropy and surprisal). For the global constraint we use the incremental method for learning a vocabulary with a minimum frequency threshold $\theta_f = 20$. For the monotonic constraint, we use the frequency method and the seeding method with $p = 50\%$. We use the same methods for the combined constraint and set the global threshold θ_g to be the 30th-percentile entropy (or surprisal) in the data.

As baselines, we train BPE tokenisers for each vocabulary size and corpus. We also train BPE tokenisers that use WP-style inference to match the inference procedure of our tokenisers, since inference can have a significant impact on intrinsic tokeniser evaluation (Uzan et al., 2024). We label these tokenisers BPE-WP.[5]

Our tokenisers (and BPE-WP) are initialised with three copies of every byte in their vocabularies. In addition to each byte, these are the byte with the WordPiece **continuation prefix** ## (used by the inference method to distinguish pre-token-internal tokens from pre-word-initial tokens) and the byte with the start-of-word prefix (used by the `ByteLevel` pre-tokeniser to indicate whitespace). This slightly wastes vocabulary space compared to BPE, as 768 base tokens are required instead of 512, but is a consequence of the complexities of combining tokeniser modules.

Evaluation. We use the intrinsic evaluation benchmark proposed in Uzan et al. (2024). It consists of four information measures: Morphological Alignment, Cognitive Plausibility, Rényi Efficiency, and Fertility:

- **Morphological alignment** The alignment of tokenisers with gold-standard morphological segmentations of words. The benchmark compares tokeniser alignment to morphological annotations from seven resources and returns a macro-averaged F_1 score. The morphological data comes from LADEC (Gagné et al., 2019), MorphoLex (Sánchez-Gutiérrez et al., 2018), MorphyNet (Batsuren et al., 2021),

[3] huggingface.co/datasets/HuggingFaceFW/fineweb-edu.
[4] huggingface.co/datasets/PleIAs/common_corpus.
[5] The WordPiece tokeniser trainers on Huggingface actually use BPE, not the WordPiece objective, to learn their vocabularies.

Table 1. Intrinsic evaluation results comparing BPE and BPE-WP to our ByteSpan tokenisers using surprisal for two vocabulary sizes. Scores are given to three significant figures, with the best score for each vocabulary size marked in **bold**.

Vocab Size	Tokenizer	Constraint	Learning Method	Morph. Alignment	Cognitive Plausibility	Fertility	Renyi Efficiency
16k	BPE	-	-	.694	**.302**	1.21	.468
	BPE-WP	-	-	.834	.297	**1.19**	.472
	ByteSpan	Global	Increment	**.899**	.146	1.90	.407
	ByteSpan	Monotonic	Frequency	.885	.254	1.39	**.483**
	ByteSpan	Monotonic	Seeding	.862	.272	1.22	.476
	ByteSpan	Combined	Frequency	.890	.268	1.29	.477
	ByteSpan	Combined	Seeding	.867	.279	1.21	.474
32k	BPE	-	-	.648	**.344**	1.13	.427
	BPE-WP	-	-	.821	.337	**1.11**	.431
	ByteSpan	Global	Increment	**.890**	.192	1.46	**.466**
	ByteSpan	Monotonic	Frequency	.843	.277	1.31	.446
	ByteSpan	Monotonic	Seeding	.862	.314	1.13	.433
	ByteSpan	Combined	Frequency	.865	.295	1.20	.438
	ByteSpan	Combined	Seeding	.860	.318	1.12	.432
64k	BPE	-	-	.609	**.362**	1.09	.395
	BPE-WP	-	-	.773	.358	**1.06**	.399
	ByteSpan	Global	Increment	**.865**	.258	1.18	**.421**
	ByteSpan	Monotonic	Frequency	.816	.285	1.25	.416
	ByteSpan	Monotonic	Seeding	.809	.339	1.08	.410
	ByteSpan	Combined	Frequency	.833	.302	1.14	.409
	ByteSpan	Combined	Seeding	.808	.344	1.07	.400

and DagoBert (Hofmann et al., 2020). Uzan et al. (2024) further augment these datasets with morpheme segmentation data (Batsuren et al., 2022), novel blend structure detection data (Pinter et al., 2021), and compound separation data (Minixhofer et al., 2023).

- **Cognitive plausibility** A benchmark from Beinborn & Pinter (2023) who found that human reaction time and accuracy from a lexical decision task correlated negatively with the number of tokens produced by a tokeniser for words, and correlated positively for nonwords. The score consists of an average from correlation scores across the four conditions (words/nonwords, accuracy/reaction time) with a higher score indicating increased 'cognitive plausibility'.
- **Rényi efficiency** A measure that penalises vocabularies consisting of many low-frequency or high-frequency tokens. It captures efficient channel usage and was found to correlate highly with BLEU scores (Zouhar et al., 2023).
- **Fertility** The average number of subwords produced per tokenised word, used by Ács (2019) to compare compression efficiency across languages for a multilingual tokeniser. A score of 1 indicates perfect compression; every word in the evaluation set exists in the tokeniser's vocabulary.

This setup allows us to evaluate tokenisers without the time- and resource-intensive process of training and evaluating LMs on downstream tasks.

Uzan et al. (2024) use MiniPile (Kaddour, 2023) to compute Rényi efficiency and fertility in order to match the dataset they used to train their tokenisers. To match our tokenisers, we compute these measures using the third subsets of our corpora (FineWeb-Edu for English, Common-Corpus for the multilingual tokenisers).

For our multilingual tokenisers, we only report Rényi efficiency and fertility using Common-Corpus, as the intrinsic evaluation benchmark for the other metrics only support English.

4. Results

We provide the results for our English tokenisers in Table 1 and the results for our multilingual tokenisers in Fig. 2. We only include the ByteSpan tokenisers using **surprisal** as the information signal — this is because for almost every measure, the equivalent tokenizer using **entropy** achieves almost identical results. When comparing vocabularies, we find a very high overlap — ranging from 85.7% to 98.8% — suggesting that both measures of information identify similar byte spans. We thus only report the **surprisal** scores in this section. Below, we summarise our key findings.

Linguistic and cognitive alignment. We find that our English ByteSpan tokenisers achieve higher morphological

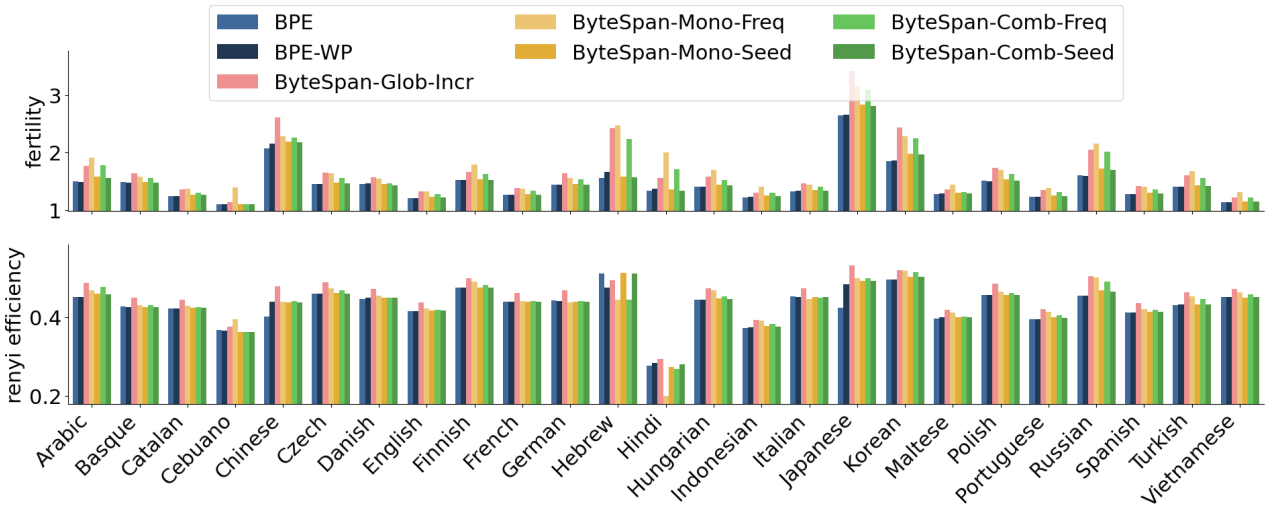

Figure 2. Fertility and rényi efficiency for each language in our Common-Corpus evaluation subset, comparing multilingual BPE to our multilingual ByteSpan tokenisers using surprisal with a vocabulary size of 128k.

alignment scores than BPE and BPE-WP for each vocabulary size but achieve lower scores on the cognitive plausibility metric. This indicates that morphological units in text can be successfully extracted using the surprisal from a byte-level model but that the number of splits may not correlate well with human performance in a lexical decision task. Our results for BPE and BPE-WP mirror those of Uzan et al. (2024), who found that using the vocabulary from BPE but applying the WordPiece inference strategy led to higher morphological alignment but lower cognitive plausibility. In general, their results suggest a trade-off between these two measures, which we also observe with our tokenisers. It is unclear which score is more desirable, although morphological alignment has long been hypothesised to be important for downstream use-cases (see e.g. Gow-Smith et al. (2022)).

When comparing between our proposed constraints and learning methods, the global constraint achieves the highest morphological alignment score for all three vocabulary sizes (and due to the apparent trade-off, also the lowest cognitive plausibility score). We found this result to contradict qualitative analysis of the tokenisers; in many cases, such as the example phrase in Fig. 1, the global constraint seems to segment the first few letters of a word individually and then the rest of the word as one token, since surprisal tends to fall towards the end of a word. This does not seem to align with English morphology and in most examples, the monotonic or combined constraints seem to produce more morphological segmentations. Upon investigation of how the morphological alignment score is calculated in the intrinsic benchmark, we found that it skips words **unless all items in the gold segmentation of that word are contained in the vocabulary of the tokeniser**. For example, if a tokenizer's vocabulary does not contain the tokens *"ramp"*, *"ant"*, *"ly"* then the word *"rampantly"* is skipped. This can skew the score if a tokeniser's vocabulary does not contain many valid morphemes or stems, making comparison between tokenisers with different vocabularies difficult.

In order to explore this further, we define the **morphological coverage** of a tokeniser as the percentage of words in the morphological data where all gold segments exist in the tokeniser's vocabulary (i.e, the percentage of the words used to calculate the alignment score for each tokeniser). We plot the coverage in Fig. 3. Indeed, the tokenisers using the global constraint have much lower coverage, suggesting that the high alignment score is not comparable to the other tokenisers. This analysis reveals that our other tokenisers have similar coverage to BPE and BPE-WP and still achieve higher morphological alignment, suggesting that they do produce a more linguistically motivated segmentation, with the combined constraint tokenisers achieving both high coverage and high alignment.

Token distribution statistics. Besides those using the global constraint, our ByteSpan tokeniser lead to higher Rényi efficiency scores than BPE and BPE-WP and very similar fertility scores across vocabulary sizes. This indicates that ByteSpan leads to good compression while ensuring that the vocabulary space does not contain too many high-frequency and low-frequency tokens. Comparing between the incremental method and the seeding method for using ByteSpan to learn a vocabulary, we find that the seeding method improves fertility at the cost of Rényi efficiency, resulting in scores very similar to BPE-WP. The fact that these tokenisers have similar token distribution statistics to BPE-WP but maintain a higher morphological alignment score suggests that by using ByteSpan to learn an initial

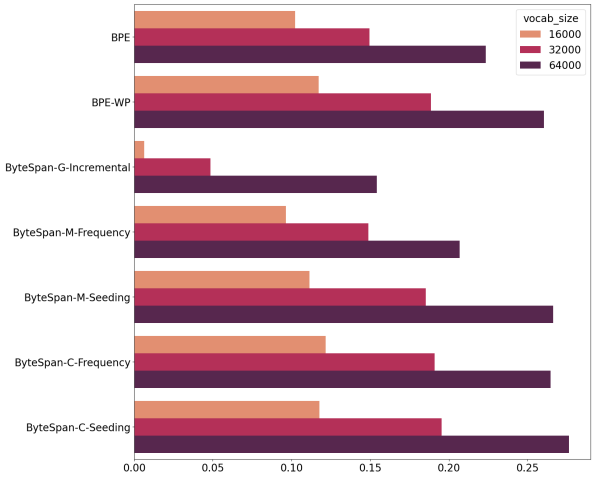

Figure 3. Morphological coverage of BPE, BPE-WP and our ByteSpan tokenisers using surprisal.

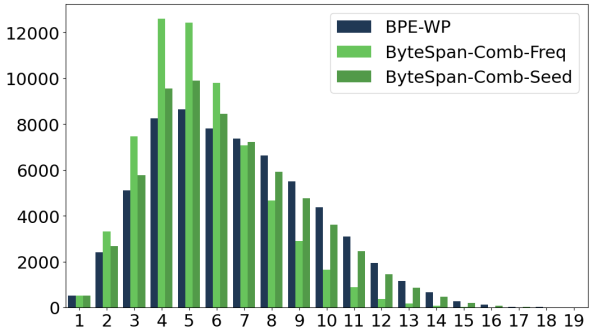

Figure 4. Distribution of token lengths comparing BPE-WP to our ByteSpan tokenisers using surprisal and the combined constraint with either the frequency method or the seeding method to learn the vocabulary. Vocabulary size is 64k.

vocabulary that is then supplemented by BPE, the resulting vocabulary contains more morphologically-aligned tokens without sacrificing compression.

To investigate this further, we examine the length of each token in the vocabularies of the BPE-WP tokeniser and our two tokenisers with the combined constraint for the largest vocabulary size, shown in Fig. 4. For all three tokenisers, the most common token lengths are 4 and 5, but the frequency method leads to a tighter distribution around these lengths compared to BPE-WP. The seeding method provides a balance by allowing BPE to merge commonly occurring sequences identified by ByteSpan, creating a distribution that more closely resembles the long tail of the BPE-WP vocabulary.

In general, the global constraint does not lead to good compression. This is because, as observed in Fig. 1, the first letters of words are highly unpredictable and so tend to be initially segmented at the character-level, even if long suffixes are compressed. The fact that this method learns long suffixes is also at odds with the longest-**prefix** inference method that we use. For example, for the largest tokeniser trained using this constraint, the token *"bonization"* is learned but *"carbonization"* is tokenised as {*"carbon"*, *"ization"*} because the token *"carbon"* also exists in the vocabulary. The monotonic and combined constraints seem to learn units across words so are not as negatively affected by this inference method.

Multilingual evaluation. The fertility and Rényi efficiency scores for each tokeniser across the 25 languages in our training corpus are given in Fig. 2. Note that the fertility scores are higher for Chinese, Japanese and Korean because for these languages the pre-tokens will long phrases instead of words, since these languages are not delimited by whitespace in Common-Corpus.

Mirroring the English results, we find that the ByteSpan tokenisers score achieve higher Rényi efficiency scores but lower fertility for most languages. Out of our tokenisers, the lowest fertility scores are achieved by the monotonic or combined constraints using the seeding method, whereas the incremental method and frequency method produce higher fertility scores. This difference is particularly pronounced for the languages in our corpus with unique writing systems; Arabic, Chinese, Hebrew, Hindi, Japanese, Korean and Russian. It is possible that since the frequency method adds the top $|V|$ most frequent byte spans to the vocabulary, this will naturally bias towards orthographies shared by most of the corpus (in this case, subwords containing Latin characters). Similarly, as the incremental method gradually raises the global threshold θ_g, if the byte-level LM struggles to predict rarer orthographies due to occurring less frequently in the data, the threshold may not add as many subwords from those languages. In the case of the frequency method, allowing BPE to learn the remaining 50% of the vocabulary seems to be an effective strategy, but BPE is also implicitly biased by frequency.

We consider one possible approach to this problem. For the frequency method, instead of selecting the top $|V|$ most frequent discovered subwords across the whole dataset, we can adjust the method to select the top $\frac{|V|}{L}$ most frequent discovered subwords for each language[6] (where L is the number of languages). This guarantees that the rarer orthographies are assigned a dedicated portion of the final vocabulary. We plot the effect of this adjustment on fertility in Fig. 5. This approach seems to lead to the desired outcome; fertility decreases for most of the languages with unique orthographies

[6]Since subwords can appear in the frequency lists of multiple languages, we add tokens round-robin until the desired vocabulary size is met, which in practice can assign more than $\frac{|V|}{L}$ tokens from the languages with rarer orthographies.

(all but Chinese). By dedicating a portion of the vocabulary to these languages, the fertility does slightly rise for all Latin-script languages, but the average across languages is not affected (even slightly decreasing).

5. Discussion

The proposed method of training a tokeniser based on information extracted from a byte-level LM achieves a balance between dynamic patching approaches and the computational benefits of a fixed vocabulary size. Through intrinsic evaluation in English, we found this approach to improve morphological alignment and Rényi efficiency compared to BPE and BPE-WP while retaining similar levels of compression. Whereas Pagnoni et al. (2024) only used entropy in their study, we found surprisal to be an effective informative signal for grouping predictable bytes. As surprisal is cheaper to compute than entropy, this could lead to efficiency gains for dynamic patching approaches that require information to be calculated at every byte during training. Further work could explore information-based tokenisation with alternative information signals, such as the probability of whitespace tokens, mimicking connectionist models of word segmentation models that rely on utterance boundary prediction (Christiansen et al., 1998).

In our multilingual evaluation, we found that the information-based approach struggles with languages whose writing systems are less represented in the data, but that using our method to seed an initial vocabulary before applying BPE addresses a gap in compression for these languages. We note that our frequency method for learning a vocabulary may implicitly lead to the vocabulary being biased towards more the more frequent orthographies found in the training data, but that by adjusting our method to force the same number of subwords to be added for each language, we could improve fertility for languages with unique orthographies. In general, our method still relies on grouping contiguous bytes based on the predictions from a byte-level LM, which may not identify useful subwords for non-concatenative languages or right-to-left languages like Arabic and Hebrew. By using bytes as our fall-back representation, we also perpetuate the encoding bias of UTF-8, which on average assigns more bytes per character for non-Latin-script languages. Future work should incorporate more balanced byte-level schemes, such as MYTE, a morphologically-driven byte encoding (Limisiewicz et al., 2024).

Finally, although ByteSpan is parameter-free for certain combinations of constraints and vocabulary-learning methods, the combined constraint and **seeding method** both use hyper-parameters which we have not thoroughly explored here (we set $p = 50\%$ for seeding method and θ_g to the 30th-percentile for the combined constraint). These should be explored further in future work. Future work could also explore the use of an **approximate** monotonic constraint (using $H(b_t) - H(b_{t-1}) < \theta_m$ instead of $H(b_t) - H(b_{t-1}) < 0$). This was proposed by Pagnoni et al. (2024) and could provide an alternative mechanism of dealing with the instability of the monotonic constraint at near-zero values for surprisal and entropy.

6. Conclusion

We present ByteSpan, a novel method for learning a subword vocabulary using the predictions of a byte-level LM. By grouping contiguous sequences of predictable bytes using one of three constraints, we find that ByteSpan tokenisers have efficient vocabularies with a higher morphological alignment than BPE and BPE-WP on English evaluation sets, with the monotonic constraint generally being more effective than the global constraint. In the multilingual setting, ByteSpan tokenisers result in similar compression rates to BPE but some methods struggle to compress languages with unique orthographies. We hypothesise that this could be due to our frequency-based vocabulary-learning method and find that balancing the vocabulary by language counteracts this effect. In general, ByteSpan provides a novel method for learning subwords with parallels to word segmentation and patching, all while keeping the benefits of learning a fixed-size vocabulary for efficient language modelling. This could feed into explorations of information in lexical unit extraction that may improve future static tokenisers and dynamic patching approaches.

Limitations

In this study we propose novel methods for learning a subword vocabulary but only use one inference method for applying the tokenisers; the longest-prefix method of WordPiece. Future work could explore the use our learned vocabularies with alternative inference methods, such as longest-suffix matching (Jacobs & Pinter, 2022).

The experiments reported in the paper only utilise one type of Transformer models, although preliminary work utilised a 5-gram model. Further work might explore the scaling properties of the byte-level model. Our evaluation largely focused on English, although we do explore a multilingual setting. In our multilingual setting we are restricted to token distribution statistics due to limited evaluation resources available for these other languages, although there are individual pipelines for individual languages (e.g., Gazit et al. (2025) uses lexical decision data to evaluate Hebrew tokenisers).

Finally, in this study, we have focused on intrinsic evaluation of tokenisation methods, in part due to the computational cost of extrinsic evaluation. The intrinsic evaluation bench-

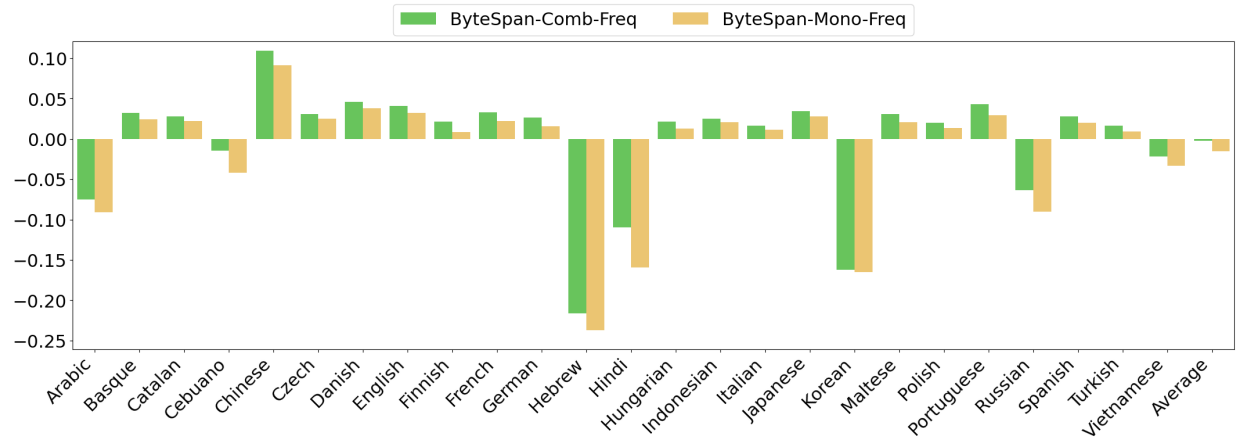

Figure 5. Increase in fertility for each language when balancing added tokens across languages when training our multilingual ByteSpan tokenisers with the frequency method. The tokenisers use surprisal as the byte-level measure and the vocabulary size is 128k. A decrease in fertility indicates better compression.

mark scores are designed to allude to potential gains in language modelling capability, but ideally this should be established by pre-training separate language models with each tokeniser and evaluating the pre-trained models using perplexity on held-out data or grammatical benchmarks such as BLiMP (Warstadt et al., 2020).

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Ács, J. Exploring BERT's vocabulary, 2019. URL https://juditacs.github.io/2019/02/19/bert-tokenization-stats.html. Accessed: May 01, 2025.

Batsuren, K., Bella, G., and Giunchiglia, F. MorphyNet: A large multilingual database of derivational and inflectional morphology. In Nicolai, G., Gorman, K., and Cotterell, R. (eds.), *Proceedings of the 18th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pp. 39–48, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.sigmorphon-1.5. URL https://aclanthology.org/2021.sigmorphon-1.5/.

Batsuren, K., Goldman, O., Khalifa, S., Habash, N., Kieraś, W., Bella, G., Leonard, B., Nicolai, G., Gorman, K., Ate, Y. G., Ryskina, M., Mielke, S., Budianskaya, E., El-Khaissi, C., Pimentel, T., Gasser, M., Lane, W. A., Raj, M., Coler, M., Samame, J. R. M., Camaiteri, D. S., Rojas, E. Z., López Francis, D., Oncevay, A., López Bautista, J., Villegas, G. C. S., Hennigen, L. T., Ek, A., Guriel, D., Dirix, P., Bernardy, J.-P., Scherbakov, A., Bayyrool, A., Anastasopoulos, A., Zariquiey, R., Sheifer, K., Ganieva, S., Cruz, H., Karahóǧa, R., Markantonatou, S., Pavlidis, G., Plugaryov, M., Klyachko, E., Salehi, A., Angulo, C., Baxi, J., Krizhanovsky, A., Krizhanovskaya, N., Salesky, E., Vania, C., Ivanova, S., White, J., Maudslay, R. H., Valvoda, J., Zmigrod, R., Czarnowska, P., Nikkarinen, I., Salchak, A., Bhatt, B., Straughn, C., Liu, Z., Washington, J. N., Pinter, Y., Ataman, D., Wolinski, M., Suhardijanto, T., Yablonskaya, A., Stoehr, N., Dolatian, H., Nuriah, Z., Ratan, S., Tyers, F. M., Ponti, E. M., Aiton, G., Arora, A., Hatcher, R. J., Kumar, R., Young, J., Rodionova, D., Yemelina, A., Andrushko, T., Marchenko, I., Mashkovtseva, P., Serova, A., Prud'hommeaux, E., Nepomniashchaya, M., Giunchiglia, F., Chodroff, E., Hulden, M., Silfverberg, M., McCarthy, A. D., Yarowsky, D., Cotterell, R., Tsarfaty, R., and Vylomova, E. UniMorph 4.0: Universal Morphology. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Odijk, J., and Piperidis, S. (eds.), *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pp. 840–855, Marseille, France, June 2022. European Language Resources Association. URL https://aclanthology.org/2022.lrec-1.89/.

Beinborn, L. and Pinter, Y. Analyzing cognitive plausibility of subword tokenization. In Bouamor, H., Pino, J., and Bali, K. (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4478–4486, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.272. URL https://aclanthology.org/

2023.emnlp-main.272/.

Brent, M. R. Efficient, probabilistically sound algorithm for segmentation and word discovery. *Machine Learning*, 34(1):71–105, 1999. ISSN 08856125. doi: 10.1023/a:1007541817488. URL https://link.springer.com/article/10.1023/A:1007541817488.

Chai, Y., Fang, Y., Peng, Q., and Li, X. Tokenization falling short: On subword robustness in large language models. In Al-Onaizan, Y., Bansal, M., and Chen, Y.-N. (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 1582–1599, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.86. URL https://aclanthology.org/2024.findings-emnlp.86/.

Christiansen, M. H., Allen, J., and Seidenberg, M. S. Learning to segment speech using multiple cues: A connectionist model. *Language and Cognitive Processes*, 13(2-3): 221–268, 1998. URL https://www.tandfonline.com/doi/abs/10.1080/016909698386528.

Çöltekin, Ç. and Nerbonne, J. An explicit statistical model of learning lexical segmentation using multiple cues. In Lenci, A., Padró, M., Poibeau, T., and Villavicencio, A. (eds.), *Proceedings of the 5th Workshop on Cognitive Aspects of Computational Language Learning (CogACLL)*, pp. 19–28, Gothenburg, Sweden, April 2014. Association for Computational Linguistics. doi: 10.3115/v1/W14-0505. URL https://aclanthology.org/W14-0505/.

Dai, Z., Lai, G., Yang, Y., and Le, Q. V. Funnel-transformer: Filtering out sequential redundancy for efficient language processing. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, Nips '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546. URL https://proceedings.neurips.cc/paper/2020/file/2cd2915e69546904e4e5d4a2ac9e1652-Paper.pdf.

Elman, J. L. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990. URL https://www.sciencedirect.com/science/article/pii/036402139090002E.

Gagné, C. L., Spalding, T. L., and Schmidtke, D. LADEC: the Large Database of English Compounds. *Behavior Research Methods*, 51:2152–2179, 2019. URL https://link.springer.com/article/10.3758/s13428-019-01282-6.

Gazit, B., Shmidman, S., Shmidman, A., and Pinter, Y. Splintering nonconcatenative languages for better tokenization, 2025. URL https://arxiv.org/abs/2503.14433.

Goriely, Z. and Buttery, P. BabyLM's first words: Word segmentation as a phonological probing task, 2025. URL https://arxiv.org/abs/2504.03338.

Goriely, Z., Caines, A., and Buttery, P. Word segmentation from transcriptions of child-directed speech using lexical and sub-lexical cues. *Journal of Child Language*, 52(1): 1–41, 2025. doi: 10.1017/s0305000923000491.

Gow-Smith, E., Tayyar Madabushi, H., Scarton, C., and Villavicencio, A. Improving tokenisation by alternative treatment of spaces. In Goldberg, Y., Kozareva, Z., and Zhang, Y. (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 11430–11443, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.786. URL https://aclanthology.org/2022.emnlp-main.786/.

Harris, Z. S. *From phoneme to morpheme*. Springer, 1955.

Hofmann, V., Pierrehumbert, J., and Schütze, H. DagoBERT: Generating derivational morphology with a pretrained language model. In Webber, B., Cohn, T., He, Y., and Liu, Y. (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 3848–3861, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.316. URL https://aclanthology.org/2020.emnlp-main.316/.

Hu, S., Tu, Y., Han, X., He, C., Cui, G., Long, X., Zheng, Z., Fang, Y., Huang, Y., Zhao, W., et al. MiniCPM: Unveiling the potential of small language models with scalable training strategies. *arXiv preprint 2404.06395*, 2024. URL https://arxiv.org/abs/2404.06395.

Jacobs, C. L. and Pinter, Y. Lost in space marking. *arXiv preprint 2208.01561*, 2022. URL https://arxiv.org/abs/2208.01561.

Kaddour, J. The MiniPile challenge for data-efficient language models. *arXiv preprint 2304.08442*, 2023. URL https://arxiv.org/abs/2304.08442.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In Bengio, Y. and LeCun, Y. (eds.), *3rd International Conference on Learning Representations*, 2015. URL http://arxiv.org/abs/1412.6980.

Kudo, T. Subword regularization: Improving neural network translation models with multiple subword candidates. In Gurevych, I. and Miyao, Y. (eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 66–75, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1007. URL https://aclanthology.org/P18-1007/.

Langlais, P.-C. The Common Corpus: A new massively multilingual dataset, September 2021. URL https://huggingface.co/blog/Pclanglais/common-corpus. Accessed: May 01, 2025.

Limisiewicz, T., Blevins, T., Gonen, H., Ahia, O., and Zettlemoyer, L. MYTE: Morphology-driven byte encoding for better and fairer multilingual language modeling. In Ku, L.-W., Martins, A., and Srikumar, V. (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15059–15076, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.804. URL https://aclanthology.org/2024.acl-long.804/.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

MacWhinney, B. The acquisition of morphophonology. *Monographs of the society for research in child development*, pp. 1–123, 1978. URL https://psyling.talkbank.org/years/1978/monograph.pdf.

Minixhofer, B., Pfeiffer, J., and Vulić, I. CompoundPiece: Evaluating and improving decompounding performance of language models. In Bouamor, H., Pino, J., and Bali, K. (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 343–359, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.24. URL https://aclanthology.org/2023.emnlp-main.24/.

Nawrot, P., Tworkowski, S., Tyrolski, M., Kaiser, L., Wu, Y., Szegedy, C., and Michalewski, H. Hierarchical transformers are more efficient language models. In Carpuat, M., de Marneffe, M.-C., and Meza Ruiz, I. V. (eds.), *Findings of the Association for Computational Linguistics: NAACL 2022*, pp. 1559–1571, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-naacl.117. URL https://aclanthology.org/2022.findings-naacl.117/.

Nawrot, P., Chorowski, J., Lancucki, A., and Ponti, E. M. Efficient transformers with dynamic token pooling. In Rogers, A., Boyd-Graber, J., and Okazaki, N. (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6403–6417, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.353. URL https://aclanthology.org/2023.acl-long.353/.

Pagnoni, A., Pasunuru, R., Rodriguez, P., Nguyen, J., Muller, B., Li, M., Zhou, C., Yu, L., Weston, J., Zettlemoyer, L., et al. Byte latent transformer: Patches scale better than tokens. *arXiv preprint 2412.09871*, 2024. URL https://arxiv.org/abs/2412.09871.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper%5Ffiles/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html.

Penedo, G., Kydlíček, H., allal, L. B., Lozhkov, A., Mitchell, M., Raffel, C., Werra, L. V., and Wolf, T. The FineWeb datasets: Decanting the web for the finest text data at scale. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=n6SCkn2QaG.

Pinter, Y., Jacobs, C. L., and Eisenstein, J. Will it unblend? In Ettinger, A., Pavlick, E., and Prickett, B. (eds.), *Proceedings of the Society for Computation in Linguistics 2021*, pp. 474–476, Online, February 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.scil-1.61/.

Rust, P., Pfeiffer, J., Vulić, I., Ruder, S., and Gurevych, I. How Good is Your Tokenizer? On the Monolingual Performance of Multilingual Language Models. In Zong, C., Xia, F., Li, W., and Navigli, R. (eds.), *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)*, pp. 3118–3135. Association for Computational Linguistics, August 2021. doi: 10.18653/v1/2021.acl-long.243. URL https://aclanthology.org/2021.acl-long.243/.

Saffran, J. R., Aslin, R. N., and Newport, E. L. Statistical learning by 8-month-old infants. *Science*, 274(5294): 1926–1928, December 1996. ISSN 00368075. doi: 10.1126/science.274.5294.1926. URL http://science.sciencemag.org/.

Sánchez-Gutiérrez, C. H., Mailhot, H., Deacon, S. H., and Wilson, M. A. MorphoLex: A Derivational Morphological Database for 70,000 English words. *Behavior Research Methods*, 50:1568–1580, 2018. URL https://link.springer.com/article/10.3758/s13428-017-0981-8.

Schuster, M. and Nakajima, K. Japanese and Korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5149–5152, 2012. doi: 10.1109/icassp.2012.6289079.

Sennrich, R., Haddow, B., and Birch, A. Neural machine translation of rare words with subword units. In Erk, K. and Smith, N. A. (eds.), *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL https://aclanthology.org/P16-1162.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C. C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P. S., Lachaux, M.-A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X. E., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., and Scialom, T. Llama 2: Open foundation and fine-tuned chat models. *arxiv preprint 2307.09288*, 2023. URL https://arxiv.org/abs/2307.09288.

Uzan, O., Schmidt, C. W., Tanner, C., and Pinter, Y. Greed is all you need: An evaluation of tokenizer inference methods. In Ku, L.-W., Martins, A., and Srikumar, V. (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 813–822, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-short.73. URL https://aclanthology.org/2024.acl-short.73/.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper%5Ffiles/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Venkataraman, A. A statistical model for word discovery in transcribed speech. *Computational Linguistics*, 27(3):350–372, March 2001. ISSN 08912017. doi: 10.1162/089120101317066113. URL https://www.mitpressjournals.org/doix/abs/10.1162/089120101317066113.

Warstadt, A., Parrish, A., Liu, H., Mohananey, A., Peng, W., Wang, S.-F., and Bowman, S. R. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392, 2020. doi: 10.1162/tacl_a_00321. URL https://aclanthology.org/2020.tacl-1.25/.

Wen, K., Li, Z., Wang, J., Hall, D., Liang, P., and Ma, T. Understanding warmup-stable-decay learning rates: A river valley loss landscape perspective. *arXiv preprint 2410.05192*, 2024. URL https://arxiv.org/abs/2410.05192.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.6. URL https://aclanthology.org/2020.emnlp-demos.6.

Yu, L., Simig, D., Flaherty, C., Aghajanyan, A., Zettlemoyer, L., and Lewis, M. MEGABYTE: Predicting million-byte sequences with multiscale transformers. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=JTmO2V9Xpz.

Zhai, X., Kolesnikov, A., Houlsby, N., and Beyer, L. Scaling vision transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12104–12113, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Zhai%5FScaling%5FVision%5FTransformers%5FCVPR%5F2022%5Fpaper.html.

Zouhar, V., Meister, C., Gastaldi, J., Du, L., Sachan, M., and Cotterell, R. Tokenization and the noiseless channel. In Rogers, A., Boyd-Graber, J., and Okazaki, N. (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5184–5207, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.284. URL https://aclanthology.org/2023.acl-long.284/.

A. Implementation Details

We implement all experiments using the PyTorch framework (Paszke et al., 2019) and implement variants of the Llama architecture with components implemented in the transformers library (Wolf et al., 2020). Tokenisers are implemented using modules from the tokenizers library.[7]

Byte-Level Model Training. We train a small byte-level LMs on our subsets of the FineWeb-Edu and Common-Corpus datasets. We use the Llama 2 architecture with 24 attention heads, 6 layers, hidden size of 768, and tied input–output embeddings, totalling 57M parameters. We use AdamW (Kingma & Ba, 2015; Loshchilov & Hutter, 2019) as our optimiser, with learning rate 6×10^{-4}, parameters $\beta_1 = 0.9$, $\beta_2 = 0.95$, $\epsilon = 1 \times 10^{-8}$, and weight decay set to 0.1. We use the warm-up-stable-decay schedule (Zhai et al., 2022), where after warm-up, the learning rate stays constant for most training and decreases briefly during the cool-down. Unlike the cosine schedule, this approach does not need a pre-specified compute budget, facilitating continued pre-training and enhancing our artifact's value for the community. It is also more effective for small language models (Hu et al., 2024; Wen et al., 2024). During training, we set the context size to 2,048 tokens and batch size 128, clip the norm of the gradients to 1.0, and train for 50k steps, saving checkpoints every 2k steps.

Byte-Level Model Predictions. We use our trained models to extract per-byte entropy and surprisal on a different subsets of FineWeb-Edu and Common-Corpus to prevent overfitting. We use a context size of 2,048 and shift the dataset along by strides of 512. This ensures that all byte-level predictions have at least 1,536 tokens of context. We then use the logits at each byte to calculate surprisal and entropy. These predictions are stored as a split of each dataset in Huggingface to facilitate training tokenisers using our method.

Hardware Details. We use a server with one NVIDIA A100 80GB PCIe, 32 CPUs, and 32 GB of RAM for all experiments. Below, we report a subset of the output of the lscpu command:

```
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Address sizes:         46 bits physical,
                       48 bits virtual
Byte Order:            Little Endian
CPU(s):                32
On-line CPU(s) list:   0-31
Vendor ID:             GenuineIntel
Model name:            Intel(R) Xeon(R)
                       Silver 4210R CPU
                       @ 2.40GHz
CPU family:            6
Model:                 85
Thread(s) per core:    1
Core(s) per socket:    1
Socket(s):             8
Stepping:              7
BogoMIPS:              4800.11
```

Reproducibility. We release all experimental artefacts as a collection on the Hugging Face Hub at redacted link: (i) the byte-level versions of the two datasets; (ii) the subsets of each dataset used to train the byte-level models, extract predictions and evaluate the resulting tokenisers; (iii) the byte-level surprisal and entropy for each dataset; (iv) the BPE, BPE-WP and ByteSpan tokenisers used in our experiments; (v) all model checkpoints.

[7] github.com/huggingface/tokenizers.