SYL-GPT: Phonologically informed tokenization improves phonological learning

Anonymous ACL submission

Abstract

Although large language models have demonstrated success on a wide variety of natural language tasks, they often still fail on simple phonological tasks like counting the number of syllables in a word. In this paper, we examine the effect of phonologically informed tokenization on downstream phonological performance. We trained 6 GPT-2 sized LLMs on 3 tokenization paradigms and 2 languages with varying orthographic transparency. We introduce syllabically informed tokenization in English, producing a generalized English syllabifier model to segment arbitrary text into syllables. We demonstrate that syllable-based tokenization may improve early learnability on phonological tasks, but that non-phonologically informed tokenization — byte-pair encoding and unigram language modeling — can match the syllable model with continued finetuning. By testing these results in English and Spanish, we further demonstrate an interaction between phonological learnability and orthographic transparency. Languages with more opaque orthographies like English may benefit more from phonologically informed tokenization, while orthographically transparent languages like Spanish may not receive additional benefit from phonological tokenization.

1 Introduction

Large language models (LLMs) demonstrate emergent abilities on a variety of linguistic domains, including natural language understanding (Madaan et al., 2025; Brown et al., 2020; see Chang and Bergen, 2024, Section 4), syntax (Kennedy, 2025; Chi et al., 2020), and morphology (Haley, 2020; Anh et al., 2024). More recently, there has been an effort to evaluate LLMs on their phonological understanding; Suvarna et al. (2024) found that despite LLMs performing surprisingly well on tasks requiring phonological awareness, they fall short of human baselines on a set of simple phonological awareness tasks. Since then, a growing body of research is developing around techniques to enhance LLM phonological reasoning ability (Jang et al., 2025; Gale et al., 2023; Zhang et al., 2024).

One leading hypothesis for LLMs' poor phonological awareness is that their tokenization is not phonologically informed (Suvarna et al., 2024; Gale et al., 2023). Tokenization is the process through which text is broken up into discrete, typically sub-word units that are mapped into numerical representations that LLMs can process. Subword tokenization was first introduced by Sennrich et al. (2016), and typically involves the curation of a vocabulary through a statistical algorithm. The most common tokenization paradigms are byte-pair encoding (bpe; Gage, 1994) — an algorithm that combines the most frequent pairs of byte or characters sequences until a set vocabulary size is reached — and its variants (Schuster and Nakajima, 2012; Kudo and Richardson, 2018). bpe has been criticized for its lack of linguistic alignment. Bostrom and Durrett (2020) is a well-cited example of work finding that bpe often produces tokens that are not aligned with the morphology of the target language. Jabbar (2024) and Hofmann et al. (2021) find that tokenization that aligns with morphological boundaries produces stronger performance on tasks that require morphological competence, concluding that bpe may still fall short of producing linguistically optimal types.

In this work, we hypothesize that bpe similarly does not accurately reflect phonological boundaries, inhibiting the development of phonological awareness in LLMs. We experiment with syllable-level tokenization in English and Spanish, pretraining GPT-2 sized LLMs and evaluating these models on a syllable and word counting task. Spanish and English vary in their orthographic transparency: the regularity of the mapping be-

tween orthographic forms and the actual phonological forms (Torres and Futrell, 2025). English is generally considered to be less transparent than Spanish, meaning that a single grapheme might map to several different phonemes depending on its context. Since LLMs only have access to the orthographic form of a language, learning phonological information without the aid of phonologically informed tokenization might be more difficult for an orthographically opaque language than an orthographically transparent language. We demonstrate that phonological awareness is easier for an LLM to acquire in an orthographically transparent language, even under non-phonologically informed tokenization, whereas it remains difficult in orthographically opaque languages. We further demonstrate that incorporating syllable-level tokenization does directly improve an LLM's ability to extract phonological information in a syllable counting task. We compare syllable-level tokenization against bpe and unigram language modeling (uni; Kudo, 2018), two common subword tokenization paradigms.

In brief, our contributions are:

1. **A robust, phonologically motivated English orthographic syllabifier.** We develop a bi-directional LSTM-based syllabification model designed to phonologically syllabify English orthography according to the Maximal Onset Principle. It significantly outperforms widely used hyphenation-based segmenters across F1 score, word-level accuracy, and syllable-count accuracy. We release our code and model at an anonymous repository (link omitted for double-blind review).

2. **A controlled evaluation of tokenization effects on phonological learning.** We train GPT-2 sized models using three tokenization paradigms — syllable-based, byte-pair encoding, and unigram language modeling — in both English and Spanish. We demonstrate that phonologically informed tokenization influences downstream phonological competence.

3. **A demonstration of the interaction between orthographic transparency and phonological learnability.** We show that Spanish models acquire syllable-counting ability even under non-phonological tokenization, while English models benefit substantially more from syllable-level tokenization. Extended finetuning allows byte-pair encoding and unigram language modeling models to eventually match syllable-based models, revealing a learnability–efficiency tradeoff.

The rest of the paper is organized as follows. Section 2 summarizes related work in the areas of LLM phonological performance, tokenization and its influence on downstream task performance, and current orthographic syllabification tools. Section 3 details the training and data curation process and evaluation framework for our syllabifier. We also cover the pretraining, finetuning, and evaluation frameworks for our GPT-2 sized models. Section 4 reports the results for our syllabifier as compared to our baselines. Section 5 reports the results for each GPT-2 sized model we train. In Section 5.3, we discuss how these results connect to linguistic theory and what is known about tokenization in LLMs. In Section 5.3 we acknowledge the limitations of this study and discuss directions for future research.

2 Related Work

LLM Phonological Performance Suvarna et al. (2024) explicitly test the phonological abilities of a mixture of 6 open and closed sourced LLMs. They find that relative to a human benchmark, LLMs significantly underperform on a variety of phonological tasks, including syllable counting, rhyme word generation, and grapheme to phoneme transcription. Aside from the work of Suvarna et al. (2024), there is a growing body of research evaluating LLMs on downstream tasks that should require phonological awareness, even if it is not explicitly tested. Ding et al. (2025) and Chatzikyriakidis and Natsina (2025) investigate the application of LLMs to song lyric and poetry generation tasks respectively, both tasks that require an awareness of phonological properties like rhythm and rhyming, finding that although LLMs are able to perform these tasks to an extent, they still fall short of human baselines. Recent work by Jang et al. (2025) scaffold from Suvarna et al. (2024) and introduce P-CoT — a technique of pedagogically informed chain of thought prompting — to maximize the utility of intrinsic phonological information in LLMs, finding that P-CoT significantly improves the performance of LLMs on the PhonologyBench task. These data suggest

that although LLMs can learn phonological information, it either requires additional specific training, as in Gale et al. (2023), or explicit methodological scaffolding, as in Jang et al. (2025).

Tokenization The effect of tokenization on an LLM's ability to learn specific linguistic features is still an active area of research. Atuhurra et al. (2024) experiment with syllable-level tokenization in Swahili, a syllable-rich language for which an algorithmic approach can be taken to extract orthographic syllables. They find that syllable-level tokenization outperforms bpe and WordPiece tokenization for the purposes of generating text in Swahili, showing that the syllable model produces more fluent responses according to human judgment (Devlin et al., 2019).

A well-established hypothesis concerning the effect of tokenization on a model's ability to learn supra-token[1] information is that misaligned tokens hamper a model's ability to access information about the sub-atomic units that compose the tokens (Edman et al., 2024; Cosma et al., 2025), and information about linguistic boundaries that span across multiple tokens (Bostrom and Durrett, 2020; Jabbar, 2024).

Syllabification Tools Recognized orthographic syllabification tools in Spanish (Gonzalez, 2018; TuLengua; Hernández-Figueroa et al., 2013) and French (Brungard, 2022) are algorithm based and specific to both the orthotactics and phonotactics of their respective languages. Previous datasets have been created with the intention of collecting accurate English syllabification data, such as the CMU Pronouncing Dictionary (University, 2024), CELEX (Baayen et al., 1995a,b), and the Moby Hyphenation List[2] (Ward, 2002). More recently, Lathouwers et al. (2025) create syllabificator, a state-of-the-art Dutch syllabifier that is extensible to English, though they have not provided a complete writeup for English yet [3].

[1]Supra-token referring to information that does not necessarily align with token level boundaries, and may span the composition of multiple tokens, such as morphological boundaries, phonological boundaries, or character level boundaries.

[2]Although this is explicitly a hyphenation dataset, the Moby Hyphenation List is a common starting point for syllabification projects.

[3]On their Github page https://github.com/guslatho/syllabificator, they cite English error rates of 1.21%, though their reference point appears to be CELEX, which does not uniformly follow the Maximal Onset Principle which we will mention in Section 3.2

3 Methodology

In order to train our syllabifier, we manually annotate a set of English words, following intuitive orthographic syllabification rules while maintaining the prescriptive rules in Section 3.2. We pretrain the GPT-2 sized models on monolingual slices of a multilingual corpus, and finetune each model on a phonological and morphological counting task to evaluate acquired linguistic competence.

3.1 English Syllabifier Architecture and Training

In order to syllabify arbitrary, unprocessed English text as it appears in pretraining corpora, we train a phonologically grounded English syllabification model, as to our knowledge, at time of development, there existed no open-source syllabifier for English designed to reliably handle out of dictionary words and pseudowords, as frequently appear in unprocessed English pretraining corpora. The syllabifier model is a bidirectional LSTM model trained on a corpus of English words and their syllabifications. The model recipe is provided in Appendix B. Syllabification is treated as a sequence labeling task, where the input sequence is a series of Latin characters, and the label sequence is a string of integers corresponding with syllable breaks and non-syllable breaks. Training is conducted by first pretraining the model on a corpus of 30,852 rough word syllabifications from the Oxford English dictionary (See Appendix A; Oxford University Press (2025)), then posttraining on a set of 8,012 hand-curated syllabifications. No such equivalent Spanish syllabifier is trained, as the task of orthographic syllabification is generally regarded as solved; due to the transparent nature of Spanish orthography, rule based systems are appropriate for Spanish syllabification (Gonzalez, 2018).

3.2 Syllable Rules

When hand-curating syllabifications, we choose to follow the Maximal Onset Principle of phonology as much as is possible (Kahn, 1976). This is the primary difference between our syllabification dataset and most hyphenation/ syllabification datasets available. Even when the syllabification might violate morphological or orthotactic boundaries, we consistently assign syllable breaks in a way that maps the resulting orthographic syllables to the underlying phonological syllables.

For example, even though [action] could be syllabified as [act]-[ion], which might more closely reflect its morphology, we choose to syllabify it as [ac]-[tion], attributing the /ʃ/ of /æk.ʃən/ to [t], and prepending it to the second orthographic syllable. This results in some apparently awkward syllabifications that violate English orthotactic rules, as in [aw]-[kward]. However, we choose to insert the syllable break between double consonant graphemes such as [bb] in [rabbit], resulting in [rab]-[bit]. This choice reflects cases where the presence of an orthographic double consonant grapheme over a single orthographic consonant grapheme impacts the pronunciation of the preceding vowel, such as [latter] vs [later] /ˈlæt.ər/ vs. /ˈleɪ.tər/. In this sense, we treat the silent preceding consonant as a modifier to the preceding syllable, information that does not necessarily influence the pronunciation of the proceeding syllable, and as such is exempt from the phonologically informed maximal onset principle[4].

3.3 Alternative Syllabification Tools

In order to compare the performance of our model, we compare our results against hyphenation algorithms and tools, which have served as proxies for English syllabification. In particular, we look at Pyphenator (Ayoub and CourtBouillon, 2025) and Hyphenate (Finkelstein, 2016). We also compare our syllabifier against syllables, a python package designed specifically for English syllable count estimations (Morris, 2025). Our comparison to these hyphenation tools is not to claim that our model will do better than them on the task of syllabification; hyphenation is only loosely related to the task of phonologically informed orthographic syllabification. However, these tools are used in practice as proxies for syllabification, and therefore serve as a useful baseline for demonstrating the effectiveness of a syllable-specific model. We refer to the aggregate of syllabifiers and hyphenators as segmenters.

3.4 Syllabifier Evaluation Metrics

We treat the task of syllabification as a classification task where the identification of a syllable boundary is considered a positive. We therefore describe our syllabifier's performance in terms of F1 score. We also look at word-level syllabification accuracy, syllable count accuracy, and Damerau–Levenshtein edit distance.

F1 score is a metric that considers both precision (the proportion of correctly predicted syllable breaks among all predicted syllable breaks) and recall (the proportion of correctly predicted syllable breaks among all gold-standard syllable breaks). In the context of syllabification, high precision indicates that the segmenter generally predicts syllable break locations accurately, while high recall indicates that it successfully identifies most true syllable breaks without missing them. F1 score therefore provides a balanced measure of overall segmentation performance across all possible syllable break positions.

In contrast, word-level syllable accuracy measures the proportion of words for which the segmenter predicts every syllable boundary correctly. Because non-boundary positions typically outnumber true syllable breaks, word-level syllable accuracy provides a stricter evaluation of performance at the level of individual words.

Another useful property of syllabification tools is their ability to implicitly provide syllable counts. Since not all tools that produce syllable counts also provide syllable boundary information, we explicitly evaluate syllable-counting performance separately, incorporating results for the syllables python package, which does not provide predicted syllable breaks.

3.5 Syllabifier Training

Training was conducted with early stopping monitoring validation loss with a patience of 2 steps. We use CrossEntropyLoss as our criterion function.

3.6 Tokenizer Training

bpe curates a vocabulary by iteratively combining the most frequent adjacent byte pairs in a training corpus until a set vocabulary size is reached. In contrast, unigram language modeling tokenization assumes an initial vocabulary of all substrings that appear in the tokenizer training corpus more than once, then uses loss produced by a unigram language model trained on the corpus to remove types from the vocabulary until the desired vocabulary size is reached. We use the sentencepiece package to train and implement bpe and uni tokenizers with byte level fallback (Kudo and Richardson, 2018). For syllable-level tokenization, we pretokenize the tokenizer training corpus on whitespace

[4] In the case of the consonant cluster [sc], as in [ascend], we choose to treat this grapheme the same as a double consonant cluster of two of the same consonants, because [s] and [c] can both be pronounced as /s/ in isolation of each other.

boundaries, preserving whitespace prepended to the start of pretokens, and run the appropriate syllabification package on each of the pretokens; our trained English syllabifier model for English, and the syltippy[5] package for Spanish (Gonzalez, 2018). The resulting syllable segments are aggregated in a counter. The final vocabulary consists of the most frequent syllable tokens. We set a total vocabulary size of 50,000 types, including an initial vocabulary consisting of 255 utf-8 bytes, 31 special types set aside for the syllable and word counting task, and 1 end of sequence <EOS> token for separating documents.

Tokenization training corpora for English and Spanish were curated by setting aside 4.47 GB of text from the English and Spanish TransWeb-Edu dataset respectively (Wang et al., 2025). After all tokenizers were trained, we tokenize a 2B token subset of the English and Spanish TransWeb-Edu dataset for each paradigm (Wang et al., 2025).

3.7 GPT-2 Pretraining

We pretrain a GPT-2 sized model for each language and each paradigm, using code from McCormick (2025). All pretraining was conducted on 2 A100 GPUs in an Azure VM instance. The model recipe is described in Appendix B. We trained with the same hyper-parameters as originally set in the Keller Jordan Modded-NanoGPT speedrun repository (Jordan et al., 2024). Token specific hyperparams are described in Table 4; the total estimated tokens used for pretraining is 917,504,000. Train and validation loss curves can be found in Appendix E.

3.8 Finetuning and Evaluation

After pretraining the GPT-2 models, we evaluate their performance by finetuning them on a training set from our chosen benchmark and evaluating the finetuned models on a test set. In order to address the small sample size of our chosen benchmarks, we perform cross validation evaluation, finetuning and testing each model over the entire dataset in 10 unique subsets. We aggregate our results in the analysis. We include results for the base GPT-2 model, available on HuggingFace, in Appendix H. We used the same code for finetuning as we did pretraining. We scaled the learning rate for all optimizers down by a factor of 10, and trained for 256 steps in the standard condition. In our extended evaluation condition, we maintain all hyperparameters except the number of steps, which we scale to 768.

3.8.1 Syllable and Word Counting

In order to evaluate the potential effects of phonologically informed tokenization on downstream phonological performance, we adapt the syllable counting task from Suvarna et al. (2024) and evaluate our English models on this task. In order to evaluate the Spanish models, we compile 1000 Spanish sentences from Fluent With Stories (2025) and use the syltippy package to produce syllable counts for each of the sentences (Gonzalez, 2018). We also evaluate the models on a word counting task. In principle, the syllable counting task might be trivial for the syllable tokenizer models, as the model might learn to count the number of tokens rather than learning any underlying phonological properties generalizable to other phonologically related tasks. As such, we also evaluate the models on a word counting task. Syllable and word counting is evaluated with a unique set of 30 separator tokens, <1>, <2>, ..., <30>. The syllable and word counting training datasets are sentences appended with the token id for <1> followed by <n>, where $2 \leq n \leq 30$, and n is the number of syllables or words in the preceding sentence.

3.8.2 Extended Training

Suvarna et al. (2024) mention their surprise that LLMs can perform phonological tasks in a zero-shot setting at all, given that LLMs are primarily trained on orthographic only representations, and are not generally tested for phonological competence. However, work by Cosma et al. (2025) may offer insight into why LLMs can achieve even sub-par performance. They find that although models with character insensitive tokenization are unable to perform as well as character sensitive models on tasks that require sub-token competence, further training these models still improves their performance. We examine a condition where we extend training for the bpe and uni models and compare their results against the syllable model, to examine if phonological competence is still learnable in models without phonologically aware tokenization. In this condition, instead of finetuning the uni

[5] We made a modification to the syltippy implementation in order to align it with the task of tokenization; we normalize all vowels to be stripped of accents before processing. The standard implementation has a set vowel inventory that does not include the complete range of diacritics that can be applied to vowels.

and bpe models on the same 256 steps as before, we finetune these models for a total of 768 steps.

4 English Syllabifier Results

We perform analysis on a test set of 788 hand-curated words not in the syllabifier's posttraining corpus.

4.1 F1 Score

Overall F1 scores for each segmenter are presented in Fig. 2. Our Syllabifier model significantly outperforms the Pyphenator and Hyphenate packages. Our syllabifier was better at identifying syllable breaks when they exist (recall). e.g. both Pyphenator and Hyphen both fail to segment "crinkly" at all, while our syllabifier correctly syllabifies "crin-kly". Our model is also better at placing syllable break correctly when it does identify a syllable break exists. e.g. Pyphenator and Hyphen segment "crozier" as "crozi-er", while our syllabifier correctly syllabifies "cro-zier".

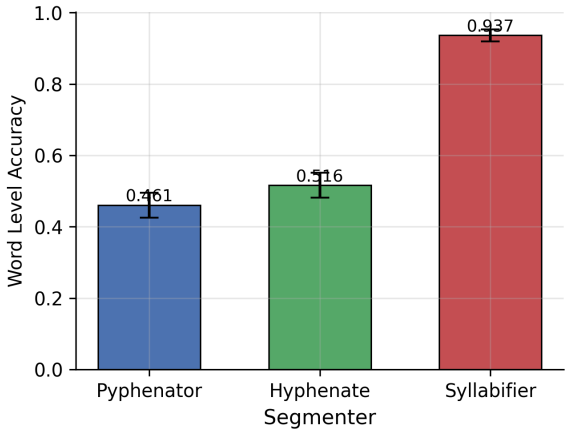


Figure 1: Word-level segmentation accuracy (Wilson intervals).

4.2 Word Level Accuracy

The results for word-level accuracy for all three segmentation packages is shown in Fig. 1. Hyphenate and Pyphenator achieve approximately the same 0.50 accuracy, while Syllabifier significantly outperforms both with 0.937 accuracy. Although F1 score is informative, this is a more representative measure of how well each segmenter would perform if deployed as a syllabifier.

4.3 Word-level Syllable Count Accuracy

The results for word-level syllable count accuracy for all four packages is shown in Fig. 3. Our Syllabifier package outperforms all of the other packages by a significant margin.

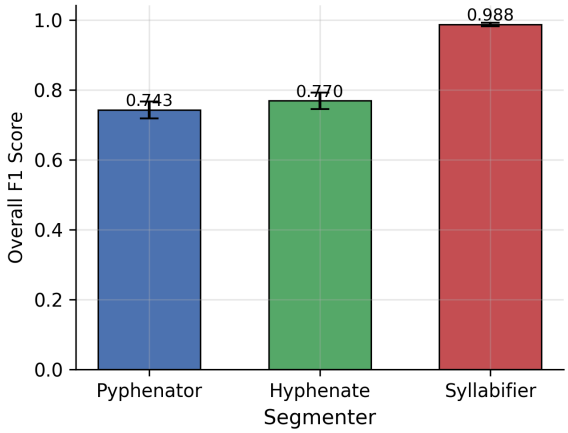


Figure 2: Overall F1 scores for each segmenter (n=788). Error bars display the bootstrap error estimation for the F1 score, bootstrap sample size=500.

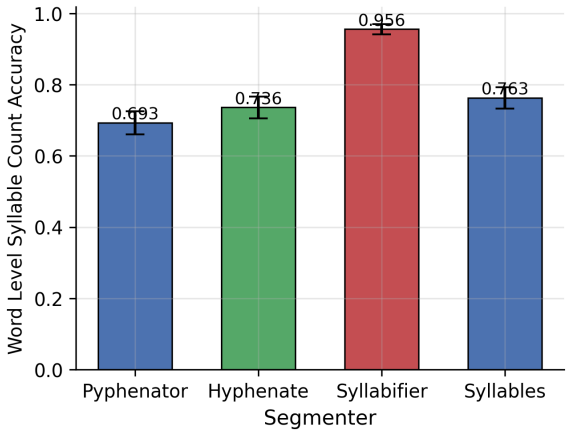


Figure 3: Word-level syllable count accuracy (Wilson intervals).

4.4 Average Damerau–Levenshtein Edit Distance

Damerau–Levenshtein edit distance is a metric evaluating the minimum number of edits required to transform one string into another string. An edit can be either a substitution of one character for another, a deletion of a character, an insertion of a character, or a transposition of two adjacent characters. Damerau—Levenshtein edit distance scores are computed between the hyphen syllabified words and the hyphen syllabified golden standards. Evaluation was conducted over words that each segmenter had incorrectly syllabified, resulting in different sample sizes and margins of error for each segmenter. The Pyphenator sample consists of 424 words. The Hyphenate sample con-

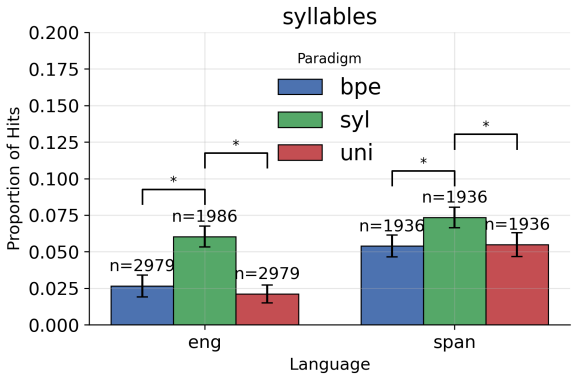
sists of 380 words. The Syllabifier sample consists of 56 words. The resulting average Damerau–Levenshtein edit distance for the Syllabifier is just above 1, representing that most of the errors are either single transposition errors or single dropped or included syllable break errors. Meanwhile, other segmenters Hyphenate and Pyphenator had higher Average Damerau–Levenshtein distances, though the error bars overlap due to the intrinsically small sample size of incorrectly syllabified words. The full graph is presented in Appendix C. Damerau–Levenshtein edit distance offers insight into errors that are not necessarily the fault of the model failing to learn general syllabification rules, but instead can be attributed to the depth of English orthography. A representative example of the types of errors that our syllabifier model makes is in the syllabification of [creator]; [creator] is properly syllabified in American English as [cre]-[a]-[tor]. A new reader of English might reasonably read [creator] as [[kɹiː.tʰə]], seeing as "crea" maps to [kɹiː] in "creature".

4.5 Tokenizer Statistics

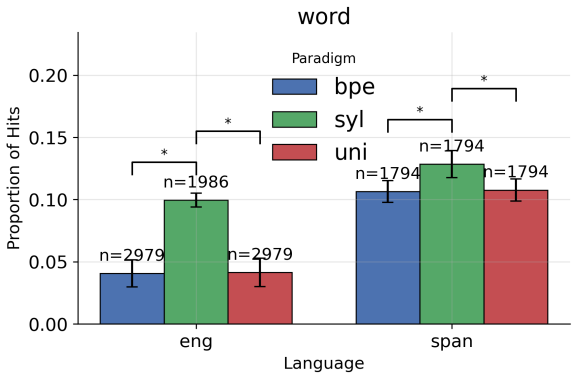
Tokenization corpora for each language were drawn from a single chunk of the Cuatro LLM pretraining dataset as segmented in the HuggingFace repository, each of which are a consistent 5 GB in size. As such, the number of whitespace separated pretokens are different for each pretraining corpus, as presented in Table 1. However, qualitative analysis of the resulting Tokenization Fertility over a sample of 500,000 demonstrates that tokenizers for both languages appear to be sufficiently representative of the target language. Fertility is a common metric for tokenizers that simply divides the number of tokens by the number of words that were tokenized. The Spanish syllable tokenizer appears to have an extraordinarily high fertility, but this follows from the intuition that Spanish words generally have more syllables than English words, up to 4-6 syllables (Martínez and Ueda, 2025). This fact, taken together with the presence of unusually long pretokens like hyperlinks and non-white-space-separated strings of Spanish words means that there is a bias towards a distribution favoring high fertility tokenization in Spanish. Complete statistics are displayed in Appendix D.

Language	Words in Tokenization Corpus
English	779,215,621
Spanish	720,640,878

Table 1: Number of whitespace separated pretokens composing the Tokenization corpus for each respective language.



(a) Syllable counting



(b) Word counting

Figure 4: Counting performance across languages and tokenization paradigms.

5 Pretrained Model Results

5.1 Syllable Counting Task

Finetuning and evaluation was conducted with cross validation over 10 sets. Total results are displayed in Fig. 4a. We find that in the English condition, the non-phonologically based tokenization paradigms bpe and uni are unable to extract the relevant phonological information from their tokens in the same number of training iterations as the syllable tokenization paradigm. On the other hand, in the Spanish condition, despite the syllable condition still significantly outperforming the bpe and uni condition, the bpe and uni models are able to learn enough relevant phonological information to achieve ∼0.1 accuracy on the syllable

counting condition. We include data from an out of the box GPT-2 model in Appendix H (Radford et al., 2019).

5.2 Word Counting Task

We observe similar results in the word counting task (Fig. 4b). In the English condition, the bpe and uni models are unable to converge and predict accurate labels over the test set. Meanwhile, the syllable-based model is able to achieve ∼0.1 accuracy. In the Spanish condition, we observe the same results as in the syllable counting task; although the bpe and uni models are better able to learn word-level boundaries from sub-word tokens, they were unable to achieve the same level of performance as the syllable model in the same number of training steps.

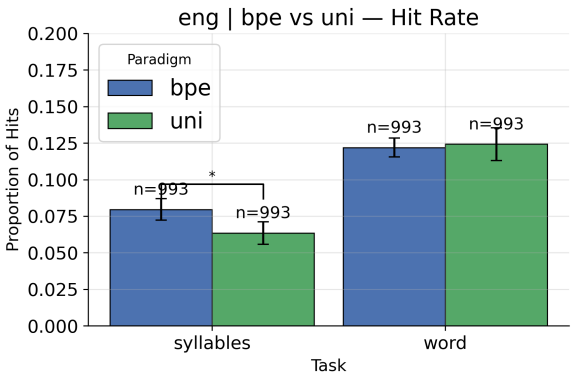


Figure 5: English bpe vs. Uni performance after 768 finetuning steps.

5.3 Extended Training Results

Fig. 5 shows that in the English bpe and uni models, performance on the syllable and word counting tasks approaches the performance of the English syllable model after 768 training iterations, which demonstrates that although models trained using non-syllable tokenization paradigms can learn to extract phonological and supra-token level information, it takes many more steps to do so.

Discussion

We find that syllable based tokenization significantly improves the rate at which LLMs acquire and recognize phonological level information from text. Similarly, we also find an interaction between language and performance; the transparency of the target language's orthography appears to influence the rate at which phonological and supra-token level information is made accessible to an LLM. These data suggest that the models are learning phonological structure rather than merely counting tokens. Appendix F displays the rate at which the number of tokens in the test sentence matches the gold standard unit count for the relevant task. Despite the fact that the syllable-level tokenizers often fall back on byte level tokenization, meaning that the number of tokens in the sequence may not be reflective of the syllable or word count, they still outperform the bpe and uni based models.

These findings reflect well documented phenomena in the domain of human language acquisition: the rate of reading acquisition in children is related to the orthographic transparency of the target language (Ellis and Hooper, 2001; Borleffs et al., 2017) and the granularity of lexical representations is fundamental for rates of reading acquisition (Ziegler and Goswami, 2005). Prior work has demonstrated that neural models similarly struggle to learn association between phonology and orthography in orthographically opaque languages (Torres and Futrell, 2025).

Our word counting tests also demonstrate that the syllable-based tokenization may improve performance on tasks that do not necessarily require phonological performance. This may be accounted for by word-level tokenization differences, though it is not immediately clear why this would be the case (see Appendix G for a brief discussion).

Our extended results are similar to the findings of Cosma et al. (2025), who demonstrate that subtoken awareness "emerges", albeit gradually, even in models not trained with explicit sub-character information. We expand their conclusions into the phonological realm, and our data show the unique outcome that the rate at which supra-token information is learned may be dependent upon the task and target language.

Limitations

We presented an analysis of pretrained LLMs' performance on phonological and word awareness tasks, but we did not examine the effects of syllable-level tokenization on other tasks, such as NLU, a further analysis is needed to determine if phonologically informed tokenization comes at a cost to other tasks. Due to computational limitations, we are also unable to scale the models

beyond GPT-2-sized, meaning that it is not clear the extent to which syllable tokenization is robust to scaling. These syllabifiers are also designed to work with latin script, or otherwise alphabetic orthographies. Replicating this methodology in Chinese would essentially be a comparison between unigram level models.

Our syllable-based tokenizer relies on a syllable-extraction pretokenization step before encoding, which is a step not required by greedy decomposition in bpe or uni. Although this is not necessarily an expensive computational overhead for algorithmic syllabifiers like syltippy, this may limit the incorporation of a syllable-level decomposition step in LLMs designed to handle deep orthographies.

References

Dang Anh, Limor Raviv, and Lukas Galke. 2024. Morphology matters: Probing the cross-linguistic morphological generalization abilities of large language models through a wug test. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 177–188. Association for Computational Linguistics.

Jesse Atuhurra, Hiroyuki Shindo, Hidetaka Kamigaito, and Taro Watanabe. 2024. Introducing syllable tokenization for low-resource languages: A case study with swahili. arXiv preprint.

Guillaume Ayoub and CourtBouillon. 2025. Pyphen: Hyphenation in pure python. Python package, PyPI, version 0.17.2.

R. H. Baayen, R. Piepenbrock, and L. Gulikers. 1995a. The celex lexical database (cd-rom). Linguistic Data Consortium, University of Pennsylvania, Philadelphia.

R. H. Baayen, R. Piepenbrock, and L. Gulikers. 1995b. Celex2: Linguistic data consortium lexical database. Available at `https://catalog.ldc.upenn.edu/LDC96L14`.

Elisabeth Borleffs, Ben A. M. Maassen, Heikki Lyytinen, and Frans Zwarts. 2017. Measuring orthographic transparency and morphological-syllabic complexity in alphabetic orthographies: A narrative review. *Reading and Writing*, 30(10):1617–1638.

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. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 1877–1901. Curran Associates Inc.

Marie-Pierre Brungard. 2022. Pylirecouleur: Parse french texts into phonemes and syllables. Python package, PyPI, version 0.0.5.

Tyler A. Chang and Benjamin K. Bergen. 2024. Language model behavior: A comprehensive survey. *Computational Linguistics*, 50(1):293–350.

Stergios Chatzikyriakidis and Anastasia Natsina. 2025. Poetry in rags: Modern greek interwar poetry generation using rag and contrastive training. In *Proceedings of the 5th International Conference on Natural Language Processing for Digital Humanities*, pages 257–264. Association for Computational Linguistics.

Ethan A. Chi, John Hewitt, and Christopher D. Manning. 2020. Finding universal grammatical relations in multilingual bert. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5564–5577. Association for Computational Linguistics.

Adrian Cosma, Stefan Ruseti, Emilian Radoi, and Mihai Dascalu. 2025. The strawberry problem: Emergence of character-level understanding in tokenized language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 28252–28263. Association for Computational Linguistics.

David Crystal. 2018. *The Cambridge Encyclopedia of the English Language*. Cambridge University Press.

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.

Shuangrui Ding, Zihan Liu, Xiaoyi Dong, Pan Zhang, Rui Qian, Junhao Huang, Conghui He, Dahua Lin, and Jiaqi Wang. 2025. Songcomposer: A large language model for lyric and melody generation in song composition. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7108–7127. Association for Computational Linguistics.

Lukas Edman, Helmut Schmid, and Alexander Fraser. 2024. Cute: Measuring llms' understanding of their tokens. In *Proceedings of the 2024 Conference on*

Empirical Methods in Natural Language Processing, pages 3017–3026. Association for Computational Linguistics.

Nick C. Ellis and A. Mari Hooper. 2001. Why learning to read is easier in welsh than in english: Orthographic transparency effects evinced with frequency-matched tests. *Applied Psycholinguistics*, 22(4):571–599.

Jeffrey Finkelstein. 2016. Hyphenate: A python module for hyphenating words. Python package, PyPI, version 1.1.0.

Fluent With Stories. 2025. Fluent with stories: Learn spanish with short stories.

Philip Gage. 1994. A new algorithm for data compression. *The C Users Journal*, pages 23–38.

Robert C. Gale, Alexandra C. Salem, Gerasimos Fergadiotis, and Steven Bedrick. 2023. Mixed orthographic/phonemic language modeling: Beyond orthographically restricted transformers (bort). In *Proceedings of the 8th Workshop on Representation Learning for NLP (RepL4NLP 2023)*, pages 212–225. Association for Computational Linguistics.

Nur Alvarez Gonzalez. 2018. Syltippy: Spanish word syllabization in python. Python package, PyPI, version 1.0.

Coleman Haley. 2020. This is a bert. now there are several of them. can they generalize to novel words? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 333–341. Association for Computational Linguistics.

Zenón Hernández-Figueroa, Francisco J. Carreras-Riudavets, and Gustavo Rodríguez-Rodríguez. 2013. Automatic syllabification for spanish using lemmatization and derivation to solve the prefix's prominence issue. *Expert Systems with Applications*, 40(17):7122–7131.

Valentin Hofmann, Janet Pierrehumbert, and Hinrich Schütze. 2021. Superbizarre is not superb: Derivational morphology improves bert's interpretation of complex words. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3594–3608. Association for Computational Linguistics.

Haris Jabbar. 2024. Morphpiece: A linguistic tokenizer for large language models. arXiv preprint.

Dongjun Jang, Youngchae Ahn, and Hyopil Shin. 2025. P-cot: A pedagogically-motivated participatory chain-of-thought prompting for phonological reasoning in llms. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 21958–21979. Association for Computational Linguistics.

Keller Jordan, Jeremy Bernstein, Brendan Rappazzo, @fernbear.bsky.social, Boza Vlado, Jiacheng You, Franz Cesista, Braden Koszarsky, and @Grad62304977. 2024. Modded-nanogpt: Speedrunning the nanogpt baseline.

Daniel Kahn. 1976. *Syllable-Based Generalizations in English Phonology*. Ph.D. thesis, Massachusetts Institute of Technology.

Mary Kennedy. 2025. Evidence of generative syntax in llms. In *Proceedings of the 29th Conference on Computational Natural Language Learning*, pages 377–396. Association for Computational Linguistics.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75. Association for Computational Linguistics.

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

Gus Lathouwers, Wieke Harmsen, Catia Cucchiarini, and Helmer Strik. 2025. Assessing dutch syllabification algorithms and improving accuracy by combining phonetic and orthographic information through deep learning. *Computational Linguistics in the Netherlands Journal*, 14:365–383.

Lovish Madaan, David Esiobu, Pontus Stenetorp, Barbara Plank, and Dieuwke Hupkes. 2025. Lost in inference: Rediscovering the role of natural language inference for large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 9229–9242. Association for Computational Linguistics.

Inmaculada Martínez Martínez and Hiroto Ueda. 2025. Longitud léxica y frecuencia de uso en el español contemporáneo: un análisis estadístico de corpus. *ELUA: Estudios de Lingüística. Universidad de Alicante*, 43:161–181.

Chris J. McCormick. 2025. Modded-nanogpt – v2025-10-27.ipynb. Google Colaboratory notebook.

Tyler Morris. 2025. syllables: Estimate syllable count in english words. Python package, PyPI, version 1.1.5.

Oxford University Press. 2025. The oxford english dictionary.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

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.

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. Association for Computational Linguistics.

Ashima Suvarna, Harshita Khandelwal, and Nanyun Peng. 2024. Phonologybench: Evaluating phonological skills of large language models. In *Proceedings of the 1st Workshop on Towards Knowledgeable Language Models (KnowLLM 2024)*, pages 1–14. Association for Computational Linguistics.

Marcus Taft. 1979. Lexical access-via an orthographic code: The basic orthographic syllabic structure (boss). *Journal of Verbal Learning and Verbal Behavior*, 18(1):21–39.

David S. Taylor. 1995. Vowels, consonants and syllables in english: An english teaching perspective. *IRAL: International Review of Applied Linguistics in Language Teaching*, 33(1):1–8.

Charles J. Torres and Richard Futrell. 2025. Clarifying orthography: Orthographic transparency as compressibility. arXiv preprint.

Rebecca Treiman and Andrea Zukowski. 1990. Toward an understanding of english syllabification. *Journal of Memory and Language*, 29(1):66–85.

TuLengua. Syllabification – silabeador tip. N.d.

Carnegie Mellon University. 2024. The cmu pronouncing dictionary. Lexical resource, version 0.7b.

Jiayi Wang, Yao Lu, Maurice Weber, Max Ryabinin, David Ifeoluwa Adelani, Yihong Chen, Raphael Tang, and Pontus Stenetorp. 2025. Multilingual language model pretraining using machine-translated data. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 28087–28107. Association for Computational Linguistics.

Grady Ward. 2002. Moby hyphenation list. Project Gutenberg eBook No. 3204.

Zhe Zhang, Karol Lasocki, Yi Yu, and Atsuhiro Takasu. 2024. Syllable-level lyrics generation from melody exploiting character-level language model. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 1336–1346. Association for Computational Linguistics.

Johannes C. Ziegler and Usha Goswami. 2005. Reading acquisition, developmental dyslexia, and skilled reading across languages: A psycholinguistic grain size theory. *Psychological Bulletin*, 131(1):3–29.

A The Discussion Surrounding Orthographic Syllabification

Linguists have long debated the conventions around English orthographic syllabification. For a detailed overview, see (Treiman and Zukowski, 1990). In brief, systems of orthographic syllabification can serve the purpose of lexicography (as in dictionary entries; see (Crystal, 2018)), pedagogy (as in the teaching of the English language; see (Taylor, 1995)), or linguistic description (as models of language processing in the human mind; see (Taft, 1979)). In this paper, we focus primarily training on a form of syllabification that serves the purpose of representing as closely as possible phonological syllable boundaries, in the hopes of capturing phonological information in the tokenization of text. As such, syllabifications drawn from the Oxford English Dictionary are not always aligned with the syllable boundaries that we draw in the finetuning training data. This is because lexicographical syllabification tends to prioritize preserving morphological boundaries, for example cutting consonant cluster onsets in violation of the maximal onset principle even if the resulting syllabification would be phonotactically valid.

B Model Recipes and Hyperparameters

Parameter	Value
LSTM Layers	2
Embedding dimension	32
Dropout	0.2
MLP Layers	4
Hidden dimensions	64

Table 2: Model recipe for the syllabifier model.

Parameter	Value
Vocab Size	50,048
Hidden size	768
Layers	12
Attention Heads	6
Head dimensions	128

Table 3: Model recipe for the GPT-2 model

Hyper-parameter	Value
Iterations	1,750
Sequence Length	65,536
Batch Size	8

Table 4: Token specific hyper-parameters

C Evaluation of Damerau–Levenshtein Distance for words with correct syllable counts

Fig. 6 shows the average Damerau-–Levenshtein edit distance evaluated over words where syllable counts were correct but boundary placement was incorrect.

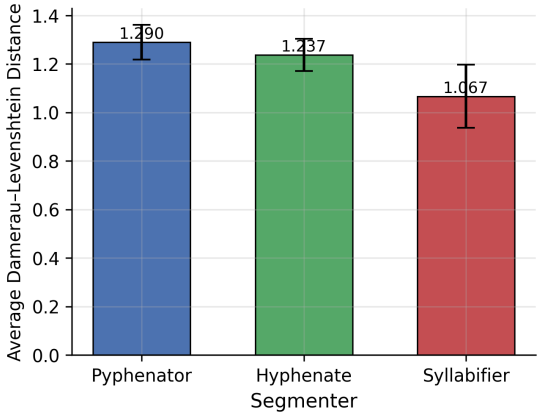


Figure 6: Average Damerau—Levenshtein edit distance for words with correct syllable counts but incorrect boundary placement.

D Extended Tokenizer Metrics

Language	Paradigm	Word Count	Token Count	Fertility
English	bpe	500,000	636,221	1.27
Spanish	bpe	500,000	617,691	1.24
English	syl	500,000	981,258	1.96
Spanish	syl	500,000	3,166,070	6.33
English	uni	500,000	635,510	1.27
Spanish	uni	500,000	616,501	1.23

Table 5: Token counts and fertility across languages and tokenization paradigms

E Train and Validation Loss Curves for GPT-2 Model Pretraining

Fig. 7 and Fig. 8 display training and validation loss curves.

Figure 7: Training loss curves aggregated across GPT-2 training runs.

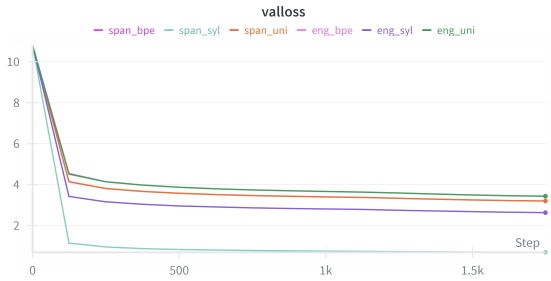
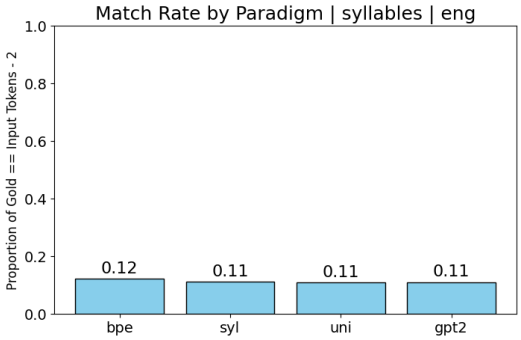


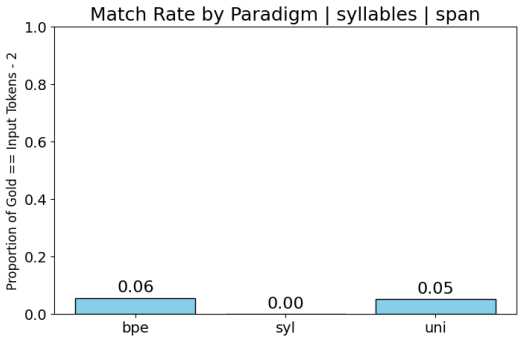
Figure 8: Validation loss curves aggregated across GPT-2 training runs.

F Rates at which token counts match gold standard counts

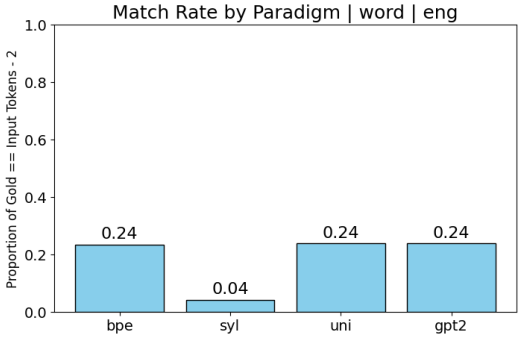
The syllable-based tokenizer — being based on English phono- and orthotactics — does not perfectly syllabify incoming text. As such it relies on a byte-level fallback to accommodate out-of-vocabulary syllables or non-English strings. The graphs below show the fallback rate for both languages.



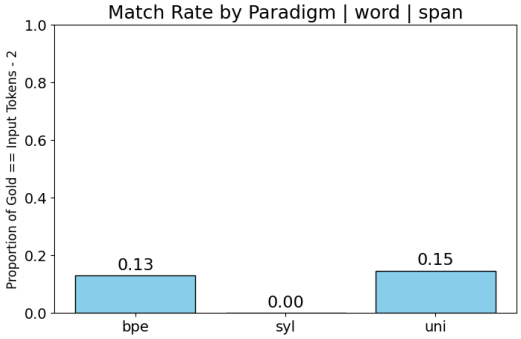
(a) Syllables (English)



(b) Syllables (Spanish)



(c) Words (English)

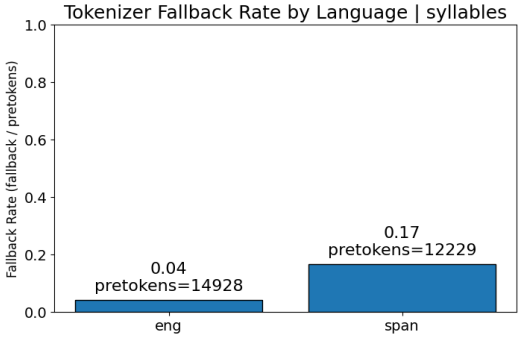


(d) Words (Spanish)

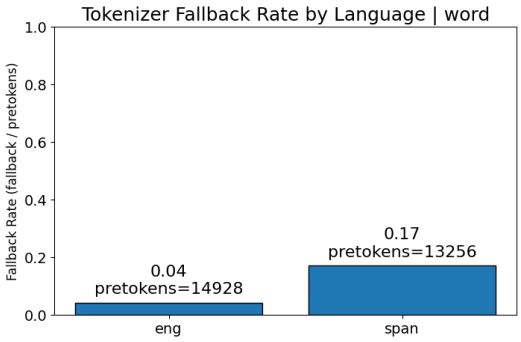
Figure 9: Match rates between predicted and gold token counts.

G Byte-level Encoding Fallback Rate

One reason for the increase in performance for syllable-based models in the word counting task may be that syllable level tokenization also captures the boundaries of single syllable words. However, statistically informed tokenization should also generate token types that encompass entire words; it is not clear why the syllable-level models outperform both bpe and uni models in the word counting condition. The figures below show the proportion of pretokens that the syllable and word based models had to fall back on in the evaluation dataset.



(a) Syllable models



(b) Word models

Figure 10: Byte-level fallback rates across languages and tasks.

H Base GPT-2 Model Pretraining Results

We evaluate GPT-2 on the same pipeline, finding that it achieves approximately the same performance as our syllable-based model on the syllable counting task, but better on the word counting task. One implication of these results is that more pretraining may improve a model's ability to extract phonological information in subsequent finetuning, but domain-specific tokenization may compensate for more pretraining.

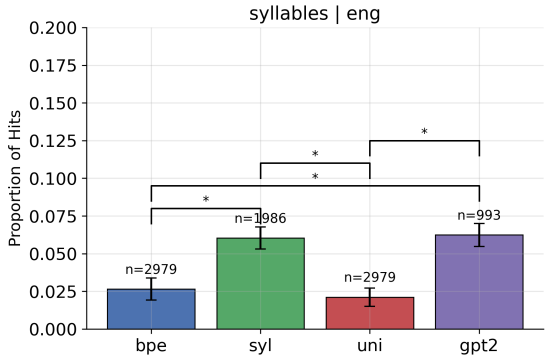


Figure 11: Syllable-level hits (bpe vs syl vs uni vs GPT-2)

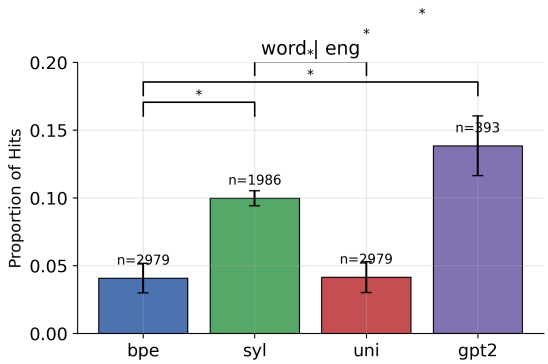


Figure 12: Word-level hits (bpe vs syl vs uni vs GPT-2)