Mind Your Inflections! Improving NLP for Non-Standard Englishes with Base-Inflection Encoding

Samson Tan[§♮], **Shafiq Joty**[§‡], **Lav R. Varshney**[♉§], **Min-Yen Kan**[♮]
[§]Salesforce AI Research [♮]National University of Singapore
[‡]Nanyang Technological University [♉]University of Illinois at Urbana-Champaign
[§]{samson.tan,sjoty}@salesforce.com
[♮]kanmy@comp.nus.edu.sg
[♉]varshney@illinois.edu

Abstract

Inflectional variation is a common feature of World Englishes such as Colloquial Singapore English and African American Vernacular English. Although comprehension by human readers is usually unimpaired by non-standard inflections, current NLP systems are not yet robust. We propose Base-Inflection Encoding (BITE), a method to tokenize English text by reducing inflected words to their base forms before reinjecting the grammatical information as special symbols. Fine-tuning pretrained NLP models for downstream tasks using our encoding defends against inflectional adversaries while maintaining performance on clean data. Models using BITE generalize better to dialects with non-standard inflections without explicit training and translation models converge faster when trained with BITE. Finally, we show that our encoding improves the vocabulary efficiency of popular data-driven subword tokenizers. Since there has been no prior work on quantitatively evaluating vocabulary efficiency, we propose metrics to do so.[1]

1 Introduction

Large-scale neural models have proven successful at a wide range of natural language processing (NLP) tasks but are susceptible to amplifying discrimination against minority linguistic communities (Hovy and Spruit, 2016; Tan et al., 2020) due to selection bias in the training data and model overamplification (Shah et al., 2019).

Most datasets implicitly assume a distribution of error-free Standard English speakers, but this does not accurately reflect the majority of the global English speaking population who are either second language (L2) or non-standard dialect speakers (Crystal, 2003; Eberhard et al., 2019). These World Englishes differ at lexical, morphological, and syntactic levels (Kachru et al., 2009); sensitivity to

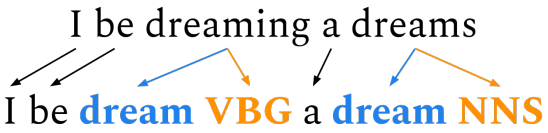


Figure 1: Base-Inflection Encoding reduces inflected words to their base forms, then reinjects the grammatical information into the sentence as inflection symbols.

these variations predisposes English NLP systems to discriminate against speakers of World Englishes by either misunderstanding or misinterpreting them (Hern, 2017; Tatman, 2017). Left unchecked, these biases could inadvertently propagate to future models via metrics built around pretrained models, such as BERTScore (Zhang et al., 2020).

In particular, Tan et al. (2020) show that current question answering and machine translation systems are overly sensitive to non-standard inflections—a common feature of dialects such as Colloquial Singapore English (CSE) and African American Vernacular English (AAVE).[2] Since people naturally correct for or ignore non-standard inflection use (Foster and Wigglesworth, 2016), we should expect NLP systems to be equally robust.

Existing work on adversarial robustness for NLP primarily focuses on adversarial training methods (Belinkov and Bisk, 2018; Ribeiro et al., 2018; Tan et al., 2020) or classifying and correcting adversarial examples (Zhou et al., 2019a). However, this effectively increases the size of the training dataset by including adversarial examples or training a new model to identify and correct perturbations, thereby significantly increasing the overall computational cost of creating robust models.

These approaches also only operate on either raw text or the model, ignoring tokenization—an operation that transforms raw text into a form that the neural network can learn from. We introduce a

[1]Code will be available at github.com/salesforce/bite.

[2]Examples in Appendix A.

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 5647–5663,
November 16–20, 2020.

new representation for word tokens that separates base from inflection. This improves both model robustness and vocabulary efficiency by explicitly inducing linguistic structure in the input to the NLP system (Erdmann et al., 2019; Henderson, 2020).

Many extant NLP systems use a combination of a whitespace and punctuation tokenizer followed by a data-driven subword tokenizer such as byte pair encoding (BPE; Sennrich et al. (2016)). However, a purely data-driven approach may fail to find the optimal encoding, both in terms of vocabulary efficiency and cross-dialectal generalization. This could make the neural model more vulnerable to inflectional perturbations. Hence, we:

- Propose Base-InflecTion Encoding (BITE), which uses morphological information to help the data-driven tokenizer use its vocabulary efficiently and generate robust symbol[3] sequences. In contrast to morphological segmentors such as Linguistica (Goldsmith, 2000) and Morfessor (Creutz and Lagus, 2002), we reduce inflected forms to their base forms before reinjecting the inflection information into the encoded sequence as special symbols. This approach gracefully handles the canonicalization of words with non-concatenative morphology while generally allowing the original sentence to be reconstructed.
- Demonstrate BITE's effectiveness at making neural NLP systems robust to non-standard inflection use while preserving performance on Standard English examples. Crucially, simply fine-tuning the pretrained model for the downstream task after adding BITE is sufficient. Unlike adversarial training, BITE does not enlarge the dataset and is more computationally efficient.
- Show that BITE helps BERT (Devlin et al., 2019) generalize to dialects unseen during training and also helps Transformer-big (Ott et al., 2018) converge faster for the WMT'14 En-De task.
- Propose metrics like symbol complexity to operationalize and evaluate the vocabulary efficiency of an encoding scheme. Our metrics are generic and can be used to evaluate any tokenizer.

2 Related Work

Subword tokenization. Before neural models can learn, raw text must first be encoded into symbols with the help of a fixed-size vocabulary. Early models represented each word as a single symbol in the vocabulary (Bengio et al., 2001; Collobert et al., 2011) and uncommon words were represented by an `unknown` symbol. However, such a representation is unable to adequately deal with words absent in the training vocabulary. Therefore, subword representations like WordPiece (Schuster and Nakajima, 2012) and BPE (Sennrich et al., 2016) were proposed to encode out-of-vocabulary (OOV) words by segmenting them into subwords and encoding each subword as a separate symbol. This way, less information is lost in the encoding process since OOV words are approximated as a combination of subwords in the vocabulary. Wang et al. (2019) reduce vocabulary sizes by operating on bytes instead of characters (as in standard BPE).

To make subword regularization more tractable, Kudo (2018) proposed an alternative method of building a subword vocabulary by reducing an initially oversized vocabulary down to the required size with the aid of a unigram language model, as opposed to incrementally building a vocabulary as in WordPiece and BPE variants. However, machine translation systems operating on subwords still have trouble translating rare words from highly-inflected categories (Koehn and Knowles, 2017).

Sadat and Habash (2006),Koehn and Hoang (2007), and Kann and Schütze (2016) propose to improve machine translation and morphological reinflection by encoding morphological features separately while Sylak-Glassman et al. (2015) propose a schema for inflectional features. Avraham and Goldberg (2017) explore the effect of learning word embeddings from base forms and morphological tags for Hebrew, while Chaudhary et al. (2018) show that representing words as base forms, phonemes, and morphological tags improve cross-lingual transfer for low-resource languages.

Adversarial robustness in NLP. To harden NLP systems against adversarial examples, existing work largely uses adversarial training (Goodfellow et al., 2015; Jia and Liang, 2017; Ebrahimi et al., 2018; Belinkov and Bisk, 2018; Ribeiro et al., 2018; Iyyer et al., 2018; Cheng et al., 2019). However, this generally involves *retraining* the model with the adversarial data, which is computationally expensive and time-consuming. Tan et al. (2020) showed that simply fine-tuning a trained model for a single epoch on appropriately generated adversarial training data is sufficient to harden the model against inflectional adversaries. Instead of

[3]Following Sennrich et al. (2016), we use *symbol* instead of *token* to avoid confusion with the unencoded *word token*.

adversarial training, Piktus et al. (2019) train word embeddings to be robust to misspellings, while Zhou et al. (2019b) propose using a BERT-based model to detect adversaries and recover clean examples. Jia et al. (2019) and Huang et al. (2019) use Interval Bound Propagation to train provably robust pre-Transformer models, while Shi et al. (2020) propose an efficient algorithm for training certifiably robust Transformer architectures.

Summary. Popular subword tokenizers operate on surface forms in a purely data-driven manner. Existing adversarial robustness methods for large-scale Transformers are computationally expensive, while provably robust methods have only been shown to work for pre-Transformer architectures and small-scale Transformers.

Our work uses linguistic information (inflectional morphology) in conjunction with data-driven subword encoding schemes to make large-scale NLP models robust to non-standard inflections and generalize better to L2 and World Englishes, while preserving performance for Standard English. We also show that our method helps existing subword tokenizers use their vocabulary more efficiently.

3 Linguistically-Grounded Tokenization

Data-driven subword tokenizers like BPE improve a model's ability to approximate the semantics of unknown words by splitting them into subwords.

Although the fully data-driven nature of such methods make them language-agnostic, this forces them to rely only on the statistics of the surface forms when transforming words into subwords since they do not exploit any language-specific morphological regularities. To illustrate, the past tense of *go*, *take*, and *keep* have the inflected forms *went*, *took*, and *kept*, respectively, which have little to no overlap with their base forms[4] and each other even though they share the same tense. These six surface forms would likely have no subwords in common in the vocabulary. Consequently, the neural model would have the burden of learning both the relation between base forms and inflected forms and the relation between inflections for the same tense. Additionally, since vocabularies are fixed before model training, such an encoding does not optimally use a limited vocabulary.

Even when inflections do not orthographically alter the base form and there is a significant overlap between the base and inflected forms, e.g., the *-ed* and *-d* suffixes, the suffix may be encoded as a separate subword and base forms / suffixes may not be consistently represented. To illustrate, encoding *danced* as [*dance*, *d*] and *dancing* as [*danc*, *ing*] results in two different "base forms" for the same word, *dance*. This again burdens the model with learning the two "base forms" mean the same thing and makes inefficient use of a limited vocabulary.

When encoded in conjunction with another inflected form like *entered*, which should be encoded as [*enter*, *ed*], this encoding scheme also produces two different subwords for the same type of inflection *-ed* vs *-d*. As in the first example, the burden of learning that the two suffixes correspond to the same tense is transferred to the learning model.

A possible solution is to instead encode *danced* as [*danc*, *ed*] and *dancing* as [*danc*, *ing*], but there is no guarantee that a data-driven encoding scheme will learn this pattern without some language-specific linguistic supervision. In addition, this unnecessarily splits up the base form into two subwords *danc* and *e*; the latter contains no extra semantic or grammatical information yet increases the encoded sequence length. Although individually minor, encoding many base words in this manner increases the computational cost for any encoder or decoder network.

Finally, although it is theoretically possible to force a data-driven tokenizer to segment inflected forms into morphologically logical subwords by limiting the vocabulary size, many inflected forms are represented as individual symbols at common vocabulary sizes (30–40k). We found that the BERT_{base} WordPiece tokenizer and BPE[5] encoded each of the above examples as single symbols.

Algorithm 1 Base-InflecTion Encoding (BITE)

Require: Input sentence $S = [w_1, \ldots, w_N]$
Ensure: Encoded sequence S'
  $S' \leftarrow [\emptyset]$
  for all $i = 1, \ldots, |N|$ **do**
    if POS$(w_i) \in$ {NOUN, VERB, ADJ} **then**
      base $\leftarrow$ GETLEMMA(w_i, POS(w_i))
      inflection $\leftarrow$ GETINFLECTION(w_i)
      $S' \leftarrow S' +$ [base, inflection]
    else
      $S' \leftarrow S' + [w_i]$
    end if
  end for
  return S'

[4] Base (no quotes) is synonymous with lemma in this paper.

[5] Trained on Wikipedia+BookCorpus (1M) with a vocabulary size of 30k symbols.

3.1 Base-Inflection Encoding

To address these issues, we propose the Base-InflecTion Encoding framework (or BITE), which encodes the base form and inflection of content words separately. Similar to how existing subword encoding schemes improve the model's ability to approximate the semantics of out-of-vocabulary words with in-vocabulary subwords, BITE helps the model better handle out-of-distribution inflection usage by keeping a content word's base form consistent even when its inflected form drastically changes. This distributional deviation could manifest as adversarial examples, such as those generated by MORPHEUS (Tan et al., 2020), or sentences produced by L2 or World Englishes speakers. By keeping the base forms consistent, BITE provides adversarial robustness to the model.

BITE (Fig. 1). Given an input sentence $S = [w_1, \ldots, w_N]$ where w_i is the i^{th} word, BITE generates a sequence of symbols $S' = [w'_1, \ldots, w'_N]$ such that $w'_i = [\text{BASE}(w_i), \text{INFLECT}(w_i)]$ where $\text{BASE}(w_i)$ is the base form of the word and $\text{INFLECT}(w_i)$ is the inflection (grammatical category) of the word (Algorithm 1). If w_i is not inflected, $\text{INFLECT}(w_i)$ is NULL and excluded from the sequence of symbols to reduce the neural network's computational cost. In our implementation, we use Penn Treebank tags to represent inflections.

By lemmatizing each inflected word to obtain the base form instead of segmenting it like in most data-driven encoding schemes, BITE ensures this base form is consistent for all inflected forms of a word, unlike a subword produced by segmentation, which can only contain characters present in the original word. For example, $\text{BASE}(took)$, $\text{BASE}(taking)$, and $\text{BASE}(taken)$ all correspond to the same base form, *take*, even though it is orthographically significantly different from *took*.

Similarly, encoding all inflections of the same grammatical category (e.g., verb-past-tense) in a canonical form should help the model to learn each inflection's grammatical role more quickly. This is because the model does not need to first learn that the same grammatical category can manifest in orthographically different forms.

Crucially, the original sentence can usually be reconstructed from the base forms and grammatical information preserved by the inflection symbols, except in cases of overabundance (Thornton, 2019).

Implementation details. We use the BertPreTokenizer from the `tokenizers`[6] library for whitespace and punctuation splitting. We use the NLTK (Bird et al., 2009) implementation of the averaged perceptron tagger (Collins, 2002) with greedy decoding to generate POS tags, which serve to improve lemmatization accuracy and as inflection symbols. For lemmatization and reinflection, we use `lemminflect`[7], which uses a dictionary look-up together with rules for lemmatizing and inflecting words. A benefit of this approach is that the neural network can now generate orthographically appropriate inflected forms by generating the base form and the corresponding inflection symbol.

3.2 Compatibility with Data-Driven Methods

Although BITE has the numerous advantages outlined above, it suffers from the same weakness as regular word-level tokenization schemes when used alone: a limited ability to handle out-of-vocabulary words. Hence, we designed BITE to be a general framework that seamlessly incorporates existing data-driven schemes to take advantage of their proven ability to handle OOV words.

To achieve this, a whitespace/punctuation-based pretokenizer is first used to transform the input into a sequence of words and punctuation characters, as is common in machine translation. Next, BITE is applied and the resulting sequence is converted into a sequence of integers by a data-driven encoding scheme (Fig. 6 in Appendix B). In our experiments, we use BITE in this manner and refer to the combined tokenizer as "BITE+D", where D refers to the data-driven encoding scheme.

4 Model-Based Experiments

We first demonstrate the effectiveness of BITE using the pretrained cased $\text{BERT}_{\text{base}}$ (Devlin et al., 2019) before training a full Transformer (Vaswani et al., 2017) from scratch. We do not replace WordPiece and BPE but instead incorporate them into the BITE framework as described in §3.2. The advantages and disadvantages to this approach will be discussed in the next section. We do not do any hyperparameter tuning but use the original models' in all experiments (detailed in Appendix B).

[6] github.com/huggingface/tokenizers
[7] github.com/bjascob/LemmInflect

Encoding	SQuAD 2 Ans. (F_1) Clean	SQuAD 2 Ans. (F_1) MORPHEUS	SQuAD 2 All (F_1) Clean	SQuAD 2 All (F_1) MORPHEUS	MNLI (Acc.) Clean	MNLI (Acc.) MORPHEUS	MNLI-MM (Acc.) Clean	MNLI-MM (Acc.) MORPHEUS
WordPiece (WP)	**74.58**	61.37	**72.75**	59.32	**83.44**	58.70	**83.59**	59.75
BITE + WP	74.50	**71.33**	72.71	**69.23**	83.01	**76.11**	83.50	**76.64**
WP + Adv. FT.	**79.07**	72.21	**74.45**	68.23	**83.86**	**83.87**	**83.86**	75.77
BITE + WP (+1 epoch)	75.46	**72.56**	73.69	**70.66**	82.21	81.05	83.36	**81.04**

Table 1: $\text{BERT}_{\text{base}}$ results on the clean and adversarial MultiNLI and SQuAD 2.0 examples. We compare BITE+WordPiece to both WordPiece alone and with one epoch of adversarial fine-tuning. For fair comparison with adversarial fine-tuning, we trained the BITE+WordPiece model for an extra epoch (bottom) on clean data.

4.1 Adversarial Robustness (Classification)

We evaluate BITE's ability to improve model robustness for question answering and natural language understanding using SQuAD 2.0 (Rajpurkar et al., 2018) and MultiNLI (Williams et al., 2018), respectively. We use MORPHEUS (Tan et al., 2020), an adversarial attack targeting inflectional morphology, to test the overall system's robustness to non-standard inflections. They previously demonstrated MORPHEUS's ability to generate plausible and semantically equivalent adversarial examples resembling L2 English sentences. We attack each $\text{BERT}_{\text{base}}$ model separately and report F_1 scores on the answerable questions and the full SQuAD 2.0 dataset, following Tan et al. (2020). In addition, for MNLI, we report scores for both the in-domain (MNLI) and out-of-domain dev. set (MNLI-MM).

BITE+WordPiece vs. only WordPiece. First, we demonstrate the effectiveness of BITE at making the model robust to inflectional adversaries. After fine-tuning two separate $\text{BERT}_{\text{base}}$ models on SQuAD 2.0 and MultiNLI, we generate adversarial examples for them using MORPHEUS. From Table 1, we observe that the BITE+WordPiece model not only achieves similar performance (± 0.5) on clean data, but is significantly more robust to inflectional adversaries (10-point difference for SQuAD 2.0, 17-point difference for MultiNLI).

BITE vs. adversarial fine-tuning. Next, we compare the BITE to adversarial fine-tuning (Tan et al., 2020), an economical variation of adversarial training (Goodfellow et al., 2015) for making models robust to inflectional variation. In adversarial fine-tuning, an adversarial training set is generated by randomly sampling inflectional adversaries k times from the adversarial distribution found by MORPHEUS and adding them to the original training set. Rather than retraining the model on this adversarial training set, the previously trained model is simply trained for one extra epoch. We follow the above methodology and adversarially fine-tune the WordPiece-only $\text{BERT}_{\text{base}}$ for one epoch with k set to 4. To ensure a fair comparison, we also train the BITE+WordPiece $\text{BERT}_{\text{base}}$ on the original training set for an extra epoch.

Condition	Encoding	BLEU	METEOR
Clean	BPE only	29.13	47.80
	BITE + BPE	**29.61**	**48.31**
MORPHEUS	BPE only	14.71	39.54
	BITE + BPE	**17.77**	**41.58**

Table 2: Results on newstest2014 for Transformer-big trained on WMT'16 English-German (En-De).

From Table 1, we observe that BITE is often more effective than adversarial fine-tuning at making the model more robust against inflectional adversaries and in some cases (SQuAD 2.0 All and MNLI-MM) even without needing the additional epoch of training. However, the adversarially fine-tuned model consistently achieves better performance on clean data. This is likely because even though adversarial fine-tuning requires only a single epoch of extra training, the process of generating the training set increases its size by a factor of k and hence the number of updates. In contrast, BITE requires no extra training and is more economical.

Adversarial fine-tuning is also less effective at inducing model robustness when the adversarial example is from an out-of-domain distribution (8 point difference between MNLI and MNLI-MM). This makes it less useful for practical scenarios, where this is often the case. In contrast, BITE performs equally well on both in- and out-of-domain data, demonstrating its applicability to practical scenarios where the training and testing domains may not match. This is the result of preserving the base forms, which we investigate further in §5.2.

4.2 Machine Translation

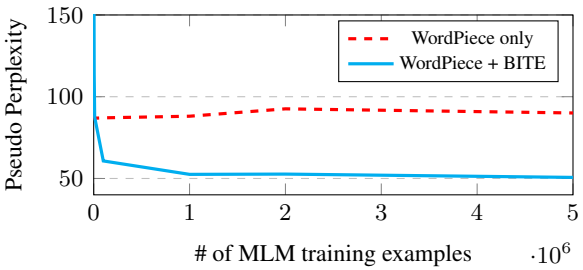
Next, we evaluate BITE's impact on machine translation using the Transformer-big architecture (Ott et al., 2018) and WMT'14 English–German (En–De) task. We apply BITE+BPE to the English examples and compare it to the BPE-only baseline. More details about our experimental setup can be found in Appendix B.3.

To obtain the final models, we perform early-stopping based on the validation perplexity and average the last ten checkpoints. We observe that the BITE+BPE model converges 28% faster (Fig. 7) than the baseline (20k vs. 28k updates) in addition to outperforming it by 0.48 BLEU on the standard data and 3.06 BLEU on the MORPHEUS adversarial examples (Table 2). This suggests that explicit encoding of morphological information helps models learn better and more robust representations faster.

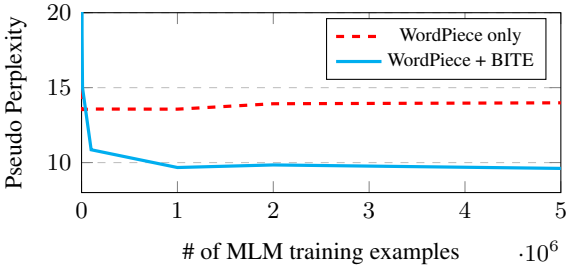
4.3 Dialectal Variation

Apart from second languages, dialects are another common source of non-standard inflections. However, there is a dearth of task-specific datasets in English dialects like AAVE and CSE. Therefore, in this section's experiments, we use the model's pseudo perplexity (pPPL) (Wang and Cho, 2019) on monodialectal corpora as a proxy for its performance on downstream tasks in the corresponding dialect. The pPPL measures how certain the pretrained model is about its prediction and reflects its generalization ability on the dialectal datasets. To ensure fair comparisons across different subword segmentations, we normalize the pseudo log-likelihoods by the number of *word tokens* fed into the WordPiece component of each tokenization pipeline (Mielke, 2019). This avoids unfairly penalizing BITE for inevitably generating longer sequences. Finally, we scale the pseudo log-likelihoods by the masking probability (0.15) so that the final pPPLs are within a reasonable range.

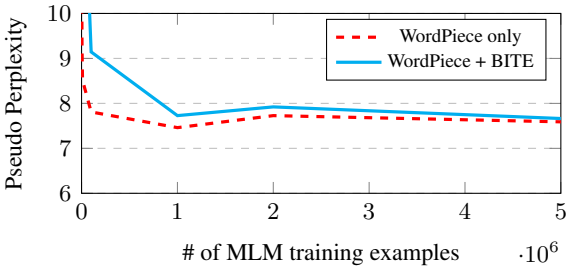
Corpora. For AAVE, we use the Corpus of Regional African American Language (CORAAL) (Kendall and Farrington, 2018), which comprises transcriptions of interviews with African Americans born between 1891 and 2005. For our evaluation, only the interviewee's speech was used. In addition, we strip all in-line glosses and annotations from the transcriptions before dropping all lines with less than three words. After preprocessing, this corpus consists of slightly under 50k lines of text (1,144,803 word tokens, 17,324 word types).



(a) Colloquial Singapore English (forum threads)



(b) African American Vernacular English (CORAAL)



(c) Standard English (Wikipedia+BookCorpus)

Figure 2: Pseudo perplexity of $\text{BERT}_{\text{base}}$ on CSE, AAVE, Standard English corpora. BITE_{abl} refers to the ablated version without grammatical information.

To obtain a CSE corpus, we scrape the Infotech Clinics section of the Hardware Zone Forums[8], a forum frequented by Singaporeans and where CSE is commonly used. Similar preprocessing to the AAVE data yields a 2.2M line corpus (45,803,898 word tokens, 253,326 word types).

Setup. We take the same pretrained $\text{BERT}_{\text{base}}$ model and fine-tune two separate variants (with and without BITE) on English Wikipedia and BookCorpus (Zhu et al., 2015) using the masked language modeling (MLM) loss without the next sentence prediction (NSP) loss. We fine-tune for one epoch on increasingly large subsets of the dataset, since this has been shown to be more effective than doing the same number of gradient updates on a fixed subset (Raffel et al., 2019). Preprocessing steps are described in Appendix B.1.

Next, we evaluate model pPPLs on the AAVE

[8] forums.hardwarezone.com.sg

and CSE corpora, which we consider to be from dialectal distributions that differ from the training data which is considered to be Standard English. Since calculating the stochastic pPPL requires randomly masking a certain percentage of symbols for prediction, we also experiment with doing this for each sentence multiple times before averaging them. However, we find no significant difference between doing the calculation once or five times; the random effects likely canceled out due to the large sizes of our corpora.

Results. From Fig. 2, we observe that the BITE+WordPiece model initially has a much higher pPPL on the dialectal corpora, before converging to 50–65% of the standard model's pPPL as the model adapts to the presence of the new inflection symbols (e.g., `VBD`, `NNS`, etc.). Crucially, the models are not trained on dialectal corpora, which demonstrates the effectiveness of BITE at helping models better generalize to unseen dialects. For Standard English, WordPiece+BITE performs slightly worse than WordPiece, reflecting the results on QA and NLI in Table 1. However, it is important to note that the WordPiece vocabulary used was not optimized for BITE; results from §4.2 indicate that training the data-driven tokenizer from scratch with BITE might improve performance.

CSE vs. AAVE. Astute readers might notice that there is a large difference in pPPL between the two dialectal corpora, even for the same tokenizer combination. One possible explanation is that CSE differs significantly from Standard English in morphology and syntax due to its Austronesian and Sinitic influences (Tongue, 1974). In addition, loan words and discourse particles not found in Standard English like *lah*, *lor* and *hor* are commonplace in CSE (Leimgruber, 2009). AAVE, however, generally shares the same syntax as Standard English due to its largely English origins (Poplack, 2000) and is more similar linguistically. These differences are likely responsible for the significant increase in pPPL for CSE compared to AAVE.

Another possible explanation is that the BookCorpus may contain examples of AAVE since the BookCorpus' source, Smashwords, also publishes African American fiction. We believe the reason for the difference is a mixture of these two factors.

4.4 Ablation Study

To tease apart the effects of BITE's two components (lemmatization and inflection symbol) on task performance, we ablate the extra grammatical information from the encoding by replacing all inflection symbols with a dummy symbol (BITE_{abl}). As expected, BITE_{abl} is significantly more robust to adversarial inflections (Table 3) and the slight performance drop is likely due to the POS tagger being adversarially affected. However, different tasks likely require different levels of attention to inflections and BITE allows the network to learn this for each task. For example, NLI performance on clean data is only slightly affected by the absence of morphosyntactic information, while MT and QA performance is more significantly affected.

Dataset	Clean		MORPHEUS	
	BITE_{abl}	BITE	BITE_{abl}	BITE
SQuAD 2 (F_1)				
Ans. Qns.	68.85	**74.50**	70.68	**71.33**
All Qns.	**72.90**	72.71	**69.29**	69.23
MNLI (Acc.)				
Matched	82.28	**83.01**	**80.17**	76.11
Mismatched	83.18	**83.50**	**81.21**	76.64
WMT'14 (BLEU)	28.14	**29.61**	**20.91**	17.77

Table 3: Effect of reinjecting grammatical information via inflection symbols. BITE_{abl} refers to the ablation with the dummy symbol instead of inflection symbols.

In a similar ablation for the pPPL experiments, we find that both the canonicalizing effect of the base form and knowledge of each word's grammatical role contribute to the lower pPPL on dialectal data (Table 4 in the Appendix). We discuss this in greater detail in Appendix B.2 and also report the pseudo log-likelihoods and per-symbol pPPLs in the spirit of transparency and reproducibility.

5 Model-Independent Analyses

Finally, we analyze WordPiece, BPE, and unigram LM subword tokenizers that are trained with and without BITE. Implementation details can be found in Appendix B.4. Through our experiments, we explore how BITE improves adversarial robustness and helps the data-driven tokenizer use its vocabulary more efficiently. We use 1M examples from Wikipedia+BookCorpus for training.

5.1 Vocabulary Efficiency

We may operationalize the question of whether BITE improves vocabulary efficiency in numerous ways. We discuss two vocabulary-level measures here and a sequence-level measure in Appendix C.

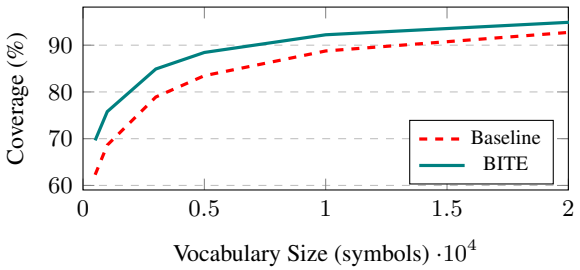


Figure 3: Comparison of coverage between BITE and a trivial baseline (word counts).

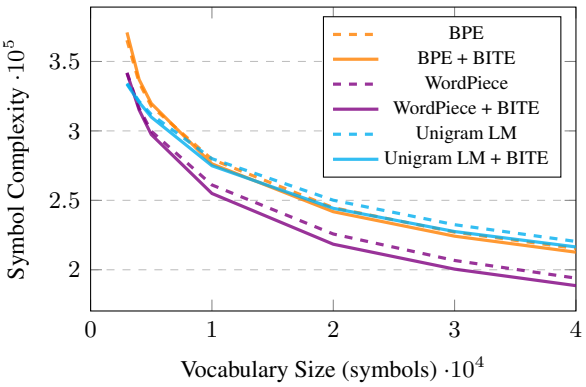


Figure 4: Symbol complexities of tokenizer vocabularies as computed in Eqs. (1) and (2). Lower is better.

Vocabulary coverage. One measure of vocabulary efficiency is the coverage of a representative corpus by a vocabulary's symbols. We measure coverage by computing the total number of tokens (words and punctuation) in the corpus that are represented in the vocabulary divided by the total number of tokens in the corpus. We use the 1M subset of Wikipedia+BookCorpus as our representative corpus. Since BITE does not require a vocabulary size to be fixed before training, we set the N most frequent types (base forms and inflections) to be our vocabulary. We use the N most frequent types in the unencoded text as our baseline vocabulary.

From Fig. 3, we observe that the BITE vocabulary achieves a higher coverage of the corpus than the baseline, hence demonstrating the efficacy of BITE at improving vocabulary efficiency. Additionally, we note that this advantage is most significant (5–7%) when the vocabulary contains less than 10k symbols. This implies that inflected word forms comprise a large portion of frequently occurring types, which comports with intuition.

Symbol complexity. Another measure of vocabulary efficiency is the total number of symbols needed to encode a representative set of word types. We term this the *symbol complexity*. Formally, given N, the total number of word types in the evaluation corpus; S_i, the sequence of symbols obtained from encoding the ith type; and u, the number of unknown symbols in S_i, we define:

$$\text{SymbComp}(S_1, \ldots, S_N) = \sum_{i=1}^{N} f(S_i), \quad (1)$$

$$f(S_i) = \begin{cases} |S_i| + u_i & |S_i| - u_i > 0 \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

While not strictly necessary when comparing vocabularies on the same corpus, normalizing Eq. (1) by the number of word types in the corpus may be helpful for cross-corpus comparisons. For simplicity, we define $f(S_i) = 0$ when there are *only* unknown symbols in the encoded sequence and the penalty of each extra unknown symbol to be double that of a symbol in the vocabulary.[9] A general form of Eq. (2) is included in Appendix B.4.

To measure the symbol complexities of our vocabularies, we use WordNet's single-word lemmas (Miller, 1995) as our "corpus" ($N = 83118$). From Fig. 4, we see that training data-driven tokenizers with BITE produces vocabularies with lower symbol complexities. Additionally, we observe that tokenizer combinations incorporating WordPiece or unigram LM generally outperform the BPE ones. We believe this to be the result of using a language model to inform vocabulary creation. It is logical that a symbol that maximizes a language model's likelihood on the training data is also semantically "denser", hence prioritizing such symbols produces efficient vocabularies. We leave the in-depth investigation of this relationship to future work.

5.2 Adversarial Robustness

BITE's ability to make models more robust to inflectional variation can be directly attributed to its preservation of consistent, inflection-independent base forms. We demonstrate this by measuring the similarity between the encoded clean and adversarial sentences with the Ratcliff/Obershelp algorithm (Ratcliff and Metzener, 1988). We use the MultiNLI in-domain development set and the MORPHEUS adversaries generated in §4.1.

We find that clean and adversarial sequences encoded by the BITE+D tokenizers were more similar (1–2.5%) than those encoded without BITE

[9] $|S|$ contributes the extra count.

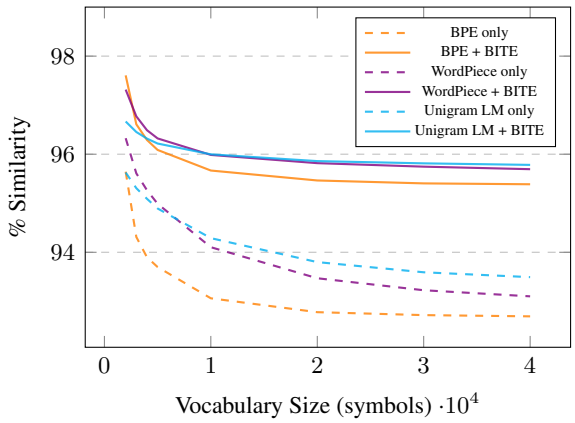


Figure 5: Mean percentage of symbols that are the same in the clean and adversarial encoded sequences.

(Fig. 5). The decrease in similarity with larger vocabularies is unsurprising; larger vocabularies result in shorter sequences, such that the same number of differing symbols will result in a larger relative change.

Hence, the improved robustness shown in §4.1 can be directly attributed to the separation of each content word's base forms from its inflection and keeping it consistent as the inflection varies, hence mitigating any significant symbol-level changes.

5.3 Micro and Error Analysis

Micro analysis. With a vocabulary of 20k symbols, BPE segments *climbs* as [*clim,bs*], *dreaming* as [*dre,aming*], and *tumbled* as [*t,umbled*]. WordPiece segments *tumbled* as [*tum,bled*] and encodes *dreaming* as a single symbol, but finds a morphologically accurate segmentation of *climbs*: [*climb,s*]. Unigram LM finds morphologically accurate segmentations for all three examples. When trained with BITE, all three tokenizers successfully find morphologically accurate segmentations of these examples and represent each corresponding base form as a single symbol.

Error analysis. Although the POS tagger is highly accurate[10], it may occasionally tag an inflected form as a base form. An example from the MultiNLI data is the word *turns* in *"..., it could turns out even better"* being tagged as `NN` instead of `VBZ`. Consequently, this word would not be split into base form and inflection. Orthographic errors like misspellings also contribute to the tagger's inaccuracy. Some of these errors can be easily fixed by using a robust POS tagger (Piktus et al., 2019).

[10] Accuracy of 97.2% on the Wall Street Journal test set.

6 Limitations

Our BITE implementation relies on an external POS tagger to assign inflection tags to each word. This tagger requires language-specific training data, which can be a challenge for low resource languages. However, this could be an advantage since the overall system can be improved by training the tagger on dialect-specific datasets, or readily extended to other languages given a suitable tagger. Another drawback of BITE is that it increases the length of the encoded sequence which may lead to extremely long sequences if used on morphologically rich languages. However, this is not an issue for English Transformer models since the increase in length will *always* be $<$2x, such that the increase in complexity is a constant factor.

7 Conclusion

The tokenization stage of the modern deep learning NLP pipeline has not received as much attention as the modeling stage, with researchers often defaulting to common subword tokenizers like BPE. We can do better. By encoding raw text into operable symbols, we can improve the generalization and adversarial robustness of resulting systems.

Hence, we guide the data-driven tokenizer by incorporating linguistic information to learn a more efficient vocabulary and generate symbol sequences that increase the network's robustness to inflectional variation. This improves its generalization to L2 and World Englishes without requiring explicit training on such data. Since dialectal data is often scarce or even nonexistent, an NLP system's ability to generalize across dialects in a zero-shot manner is crucial for it to work well for diverse linguistic communities. A more general, BITE-like algorithm should enable further gains on morphologically rich languages.

Finally, given the effectiveness of the common task framework for spurring progress in NLP (Varshney et al., 2019), we hope to do the same for tokenization. As a first step, we propose to evaluate an encoding scheme's efficacy by measuring its vocabulary coverage and symbol complexity (which may have interesting connections to information-theoretic limits (Ziv and Lempel, 1978)). We have already shown that Base-Inflection Encoding helps a data-driven tokenizer use its limited vocabulary more efficiently by reducing its symbol complexity when the combination is trained from scratch.

Acknowledgments

We are grateful to Michael Yoshitaka Erlewine from the NUS Dept. of English Language and Literature and our anonymous reviewers for their invaluable feedback. We also thank Xuan-Phi Nguyen for his help with reproducing the Transformer-big baseline. Samson is supported by Salesforce and Singapore's Economic Development Board under its Industrial Postgraduate Programme.

References

Oded Avraham and Yoav Goldberg. 2017. The interplay of semantics and morphology in word embeddings. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 422–426, Valencia, Spain. Association for Computational Linguistics.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *6th International Conference on Learning Representations*, Vancouver, BC, Canada.

Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. 2001. A neural probabilistic language model. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 932–938. MIT Press.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly Media.

Aditi Chaudhary, Chunting Zhou, Lori Levin, Graham Neubig, David R. Mortensen, and Jaime Carbonell. 2018. Adapting word embeddings to new languages with morphological and phonological subword representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3285–3295, Brussels, Belgium. Association for Computational Linguistics.

Yong Cheng, Lu Jiang, and Wolfgang Macherey. 2019. Robust neural machine translation with doubly adversarial inputs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4324–4333, Florence, Italy. Association for Computational Linguistics.

Michael Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 1–8.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(76):2493–2537.

Mathias Creutz and Krista Lagus. 2002. Unsupervised discovery of morphemes. In *Proceedings of the ACL-02 Workshop on Morphological and Phonological Learning*, pages 21–30. Association for Computational Linguistics.

David Crystal. 2003. *English as a Global Language*. Cambridge University Press.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the EACL 2014 Workshop on Statistical Machine Translation*.

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.

David M. Eberhard, Gary F. Simons, and Charles D. Fennig, editors. 2019. *Ethnologue: Languages of the World*, 22 edition. SIL International.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36, Melbourne, Australia. Association for Computational Linguistics.

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

Pauline Foster and Gillian Wigglesworth. 2016. Capturing accuracy in second language performance: The case for a weighted clause ratio. *Annual Review of Applied Linguistics*, 36:98–116.

John Goldsmith. 2000. Linguistica: An automatic morphological analyzer. In *Proceedings of 36th meeting of the Chicago Linguistic Society*.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations*, San Diego, California.

James Henderson. 2020. The unstoppable rise of computational linguistics in deep learning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Seattle, Washington. Association for Computational Linguistics.

Alex Hern. 2017. Facebook translates 'good morning' into 'attack them', leading to arrest. *The Guardian*.

Dirk Hovy and Shannon L. Spruit. 2016. The social impact of natural language processing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 591–598, Berlin, Germany. Association for Computational Linguistics.

Po-Sen Huang, Robert Stanforth, Johannes Welbl, Chris Dyer, Dani Yogatama, Sven Gowal, Krishnamurthy Dvijotham, and Pushmeet Kohli. 2019. Achieving verified robustness to symbol substitutions via interval bound propagation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4083–4093, Hong Kong, China. Association for Computational Linguistics.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885, New Orleans, Louisiana. Association for Computational Linguistics.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified robustness to adversarial word substitutions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4129–4142, Hong Kong, China. Association for Computational Linguistics.

Braj B. Kachru, Yamuna Kachru, and Cecil Nelson, editors. 2009. *The Handbook of World Englishes*. Wiley-Blackwell.

Katharina Kann and Hinrich Schütze. 2016. Single-model encoder-decoder with explicit morphological representation for reinflection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 555–560, Berlin, Germany. Association for Computational Linguistics.

Tyler Kendall and Charlie Farrington. 2018. The corpus of regional african american language.

Philipp Koehn and Hieu Hoang. 2007. Factored translation models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 868–876, Prague, Czech Republic. Association for Computational Linguistics.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics.

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

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems 32*, pages 7059–7069. Curran Associates, Inc.

Jacob RE Leimgruber. 2009. *Modelling variation in Singapore English*. Ph.D. thesis, Oxford University.

Sabrina J. Mielke. 2019. Can you compare perplexity across different segmentations?

George A. Miller. 1995. Wordnet: A lexical database for english. *Communications of the ACM*, 38:39–41.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 1–9, Brussels, Belgium. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Aleksandra Piktus, Necati Bora Edizel, Piotr Bojanowski, Edouard Grave, Rui Ferreira, and Fabrizio Silvestri. 2019. Misspelling oblivious word embeddings. 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

3226–3234, Minneapolis, Minnesota. Association for Computational Linguistics.

Shana Poplack. 2000. *The English History of African American English*. Blackwell Malden, MA.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*, arXiv:1910.10683.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

John W Ratcliff and David E Metzener. 1988. Pattern-matching: The gestalt approach. *Dr Dobbs Journal*, 13(7):46.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging NLP models. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 856–865, Melbourne, Australia. Association for Computational Linguistics.

Fatiha Sadat and Nizar Habash. 2006. Combination of Arabic preprocessing schemes for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 1–8, Sydney, Australia. Association for Computational Linguistics.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712, Online. Association for Computational Linguistics.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152. IEEE.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725. Association for Computational Linguistics.

Deven Shah, H. Andrew Schwartz, and Dirk Hovy. 2019. Predictive biases in natural language processing models: A conceptual framework and overview. *arXiv e-prints*, arXiv:1912.11078.

Zhouxing Shi, Huan Zhang, Kai-Wei Chang, Minlie Huang, and Cho-Jui Hsieh. 2020. Robustness verification for transformers. In *8th International Conference on Learning Representations*, Addis Ababa, Ethiopia.

John Sylak-Glassman, Christo Kirov, David Yarowsky, and Roger Que. 2015. A language-independent feature schema for inflectional morphology. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 674–680, Beijing, China. Association for Computational Linguistics.

Samson Tan, Shafiq Joty, Min-Yen Kan, and Richard Socher. 2020. It's morphin' time! Combating linguistic discrimination with inflectional perturbations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2920–2935, Online. Association for Computational Linguistics.

Rachael Tatman. 2017. Gender and dialect bias in YouTube's automatic captions. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, pages 53–59, Valencia, Spain. Association for Computational Linguistics.

Anna M Thornton. 2019. Overabundance in morphology. In *Oxford Research Encyclopedia of Linguistics*.

Ray K Tongue. 1974. *The English of Singapore and Malaysia*. Eastern Universities Press.

Lav R. Varshney, Nitish Shirish Keskar, and Richard Socher. 2019. Pretrained AI models: Performativity, mobility, and change. *arXiv e-prints*, arXiv:1909.03290.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Alex Wang and Kyunghyun Cho. 2019. BERT has a mouth, and it must speak: BERT as a Markov random field language model. *arXiv e-prints*, arXiv:1902.04094.

Changhan Wang, Kyunghyun Cho, and Jiatao Gu. 2019. Neural machine translation with byte-level subwords. *arXiv e-prints*, arXiv:1909.03341.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume*

1 (Long Papers), pages 1112–1122. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv e-prints*, arXiv:1910.03771.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Yichao Zhou, Jyun-Yu Jiang, Kai-Wei Chang, and Wei Wang. 2019a. Learning to discriminate perturbations for blocking adversarial attacks in text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4904–4913, Hong Kong, China. Association for Computational Linguistics.

Yichao Zhou, Jyun-Yu Jiang, Kai-Wei Chang, and Wei Wang. 2019b. Learning to discriminate perturbations for blocking adversarial attacks in text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4906–4915, Hong Kong, China. Association for Computational Linguistics.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *The IEEE International Conference on Computer Vision (ICCV)*.

Jacob Ziv and Abraham Lempel. 1978. Compression of individual sequences via variable-rate coding. *IEEE Transactions on Information Theory*, IT-24(5):530–536.

A Examples of Inflectional Variation in English Dialects

African American Vernacular English (Kendall and Farrington, 2018)

- I dreamed about we was over my uh, father mother house, and then we was moving.
- I be over with my friends.
- And this boy name RD-NAME-3, he was tryna be tricky, pretend like he don't do nothing all the time.

Colloquial Singapore English (Singlish) (Source: forums.hardwarezone.com.sg)

- Anyone face the problem after fresh installed the Win 10 Pro, under NetWork File sharing after you enable this function (Auto discovery), the computer still failed to detect our Users connected to the same NetWork?
- I have try it already, but no solutions appear.
- How do time machine works??

B Implementation/Experiment Details

All models are trained on 8 16GB Tesla V100s.

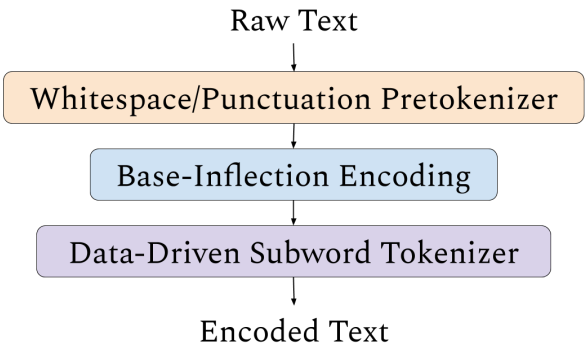


Figure 6: How BITE fits into the tokenization pipeline.

B.1 Classification Experiments

For our BERT experiments, we build BITE on top of the `BertTokenizer` class in Wolf et al. (2019) and use their BERT implementation and fine-tuning scripts[11]. $\text{BERT}_{\text{base}}$ has 110M parameters. We do not perform a hyperparameter search and instead use the example hyperparameters for the respective scripts.

[11] github.com/huggingface/transformers/.../examples

Datasets and metrics. MultiNLI (Williams et al., 2018) is a natural language inference dataset of 392,702 training examples, 10k in-domain and 10k out-of-domain dev. examples, and 10k in-domain and 10k out-of-domain test examples spanning 10 domains. Each example comprises a premise, hypothesis, and a label indicating whether the premise entails, contradicts, or is irrelevant to the hypothesis. Models are evaluated using Accuracy $= \frac{\text{\# correct predictions}}{\text{\# predictions}}$.

SQuAD 2.0 (Rajpurkar et al., 2018) is an extractive question answering dataset comprising more than 100k answerable questions and 50k unanswerable questions (130,319 training examples, 11,873 development examples, and 8,862 test examples). Each example is composed of a question, a passage, and an answer. Answerable questions are questions that can be answered by a span in the passage and unanswerable questions are questions that cannot be answered by a span in the passage. Models are evaluated using the F_1 score.

Wikipedia+BookCorpus is a combination of English Wikipedia and BookCorpus. We use Lample and Conneau (2019)'s script to download and preprocess the Wikipedia dump before removing blank lines, overly short lines (less than three words or four characters), and lines with `doc` tags. We also remove blank and overly short lines from BookCorpus before concatenating and shuffling both datasets.

B.2 Discussion for Perplexity Experiments

Effect of lemmatization and inflection symbols. We conduct two ablations to investigate the effects of lemmatization and inflection symbols on the models' pseudo perplexities: the first simply lemmatizes the input before encoding it with WordPiece (WordPiece+LEMM) and the second replaces every inflection symbol generated by BITE with a dummy symbol (WordPiece+BITE_{abl}). The latter is the same ablation used in Table 3 and from Table 4, we see that this condition consistently achieved the lowest pPPL on all three corpora. However, we believe that the highly predictable dummy symbols likely account for the significant drops in pseudo perplexity.

To test this hypothesis, we perform another ablation, WordPiece+LEMM, where the the dummy symbols are removed entirely. If the dummy symbols were not truly responsible for the large drops in pPPL, we should observe similar results for

Dataset	WordPiece (WP) —	WP + LEMM (Lemmatize)	WP + BITE (+Infl. Symbols)	WP + BITE$_{abl}$ (+Dummy Symbol)
Colloquial Singapore English				
Total word tokens before WP	45803898	45803898	51982873	51982873
Pseudo Negative Log-Likelihood	30910290	30558864	31110740	30292923
pPPL (per word token before WP)	92.58	85.43	**52.67**	48.66
pPPL (per symbol after WP)	49.10	46.39	**32.02**	30.20
African American Vernacular English				
Total word tokens before WP	1144803	1144803	1320730	1320730
Pseudo Negative Log-Likelihood	452269	444021	453031	434621
pPPL (per word token before WP)	13.92	13.27	**9.84**	8.96
pPPL (per symbol after WP)	12.90	12.41	**9.18**	8.43
Standard English				
Total word tokens before WP	252153	252153	290391	290391
Pseudo Negative Log-Likelihood	77339	78074	90148	75467
pPPL (per word token before WP)	**7.72**	7.87	7.92	5.65
pPPL (per symbol after WP)	6.34	6.36	**6.07**	4.86

Table 4: Effect of lemmatization, inflection symbols, and dummy symbol on pseudoperplexity (pPPL). We also show the effect of normalizing by the word token vs. subword symbol count. Lower is better. **Bolded** values indicate lowest row-wise pPPLs, *excluding* WP+BITE$_{abl}$ due to the confounding effect of the highly predictable dummy symbols.

both WordPiece+LEMM and WordPiece+BITE$_{abl}$. From Table 4 (pPPL per word token before WP), we see that the decrease in pPPL between WordPiece+LEMM and WordPiece is less drastic, thereby lending evidence for rejecting the null hypothesis.

Poorer performance on Standard English. We observe that lemmatizing all content words and reinjecting the grammatical information appears to have the opposite effect on Standard English data compared to the dialectal data. Intuitively, such an encoding *should* result in even more significant reductions in perplexity on Standard English since the POS tagger and lemmatizer were trained on Standard English data. A possible explanation for these results is that the WordPiece tokenizer and BERT model are overfitted on Standard English, since they were both (pre-)trained on Standard English data.

Normalizing log-likehoods. In an earlier version of this paper, we computed pseudo perplexity by normalizing the pseudo log-likehoods with the number of masked subword symbols (the default). A reviewer pointed out that per subword symbol perplexities are not directly comparable across different subword segmentations/vocabularies, but per word perplexities are (Mielke, 2019; Salazar et al., 2020). However, using the same denominator would unfairly penalize models using BITE since it inevitably increases the symbol sequence length, which affects the predicted log-likelihoods. In addition, with the exception of the inflection/dummy symbols that replaced some unused tokens, the vocabularies of all the WordPiece tokenizers used in our pseudo perplexity experiments are exactly the same since we do not retrain them. Therefore, we attempt to balance these two factors by normalizing by the number of *word tokens* fed into the WordPiece component of each tokenization pipeline in Fig. 2. We also report the per subword pPPL and raw pseudo negative log-likelihood in Table 4.

B.3 Machine Translation Experiments

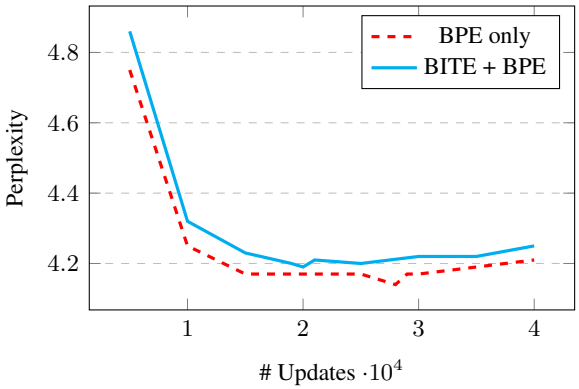


Figure 7: Validation perplexity over the course of training for Transformer-big.

For our Transformer-big experiments, we use the `fairseq` (Ott et al., 2019) implementation and the hyperparameters from Ott et al. (2018):

- Parameters: 210,000,000
- BPE operations: 32,000

- Learning rate: 0.001
- Per-GPU batch size: 3,584 tokens
- Warmup period: 4,000 updates
- Dropout: 0.3
- Gradient Accumulation: 16

Both models took 24.6 hours to complete 45k updates. We use a `fairseq` script[12] to average the selected checkpoint with the previous nine.

Dataset and metrics. We use the WMT'16 data[13] for training and newstest2013 for development and newstest2014 for testing. Although it is common practice to use the already encoded WMT'16 data released by Google, BITE requires raw or whitespace-tokenized text as input. Hence, we use the raw WMT data and the `fairseq` preprocessing script to preprocess our data. After preprocessing, we obtain a dataset of 4.3M training examples, 2,996 dev. examples, and 3,003 test examples. Models are evaluated using BLEU[14] (Papineni et al., 2002) and METEOR[15] (Denkowski and Lavie, 2014), standard MT evaluation metrics.

B.4 Model-Independent Analyses

We use the `tokenizers` implementation of WordPiece and BPE and the SentencePiece (Kudo and Richardson, 2018) implementation of unigram LM. For ease of comparison across the three encoding schemes, we pretokenize the raw text with `tokenizers` BertPreTokenizer before encoding them. For practical applications, users may use `sentencepiece`'s method of handling whitespace characters instead of the BertPreTokenizer.

General form of Eq. (2).

$$f(S_i, \lambda) = \begin{cases} (|S_i| - u_i) + \lambda u_i & |S_i| - u_i > 0 \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

where N is the total number of word types in the evaluation corpus, S_i is the sequence of symbols obtained from encoding the ith base form, u is the number of unknown symbols in S_i, and λ is the weight of the unknown symbol penalty.

Ratcliff/Obershelp algorithm. We use Python's `difflib` implementation.

[12] github.com/pytorch/fairseq/.../average_checkpoints.py
[13] statmt.org/wmt16/translation-task.html
[14] Calculated by `fairseq`.
[15] cs.cmu.edu/ alavie/METEOR/

C More Measures of Vocabulary Efficiency

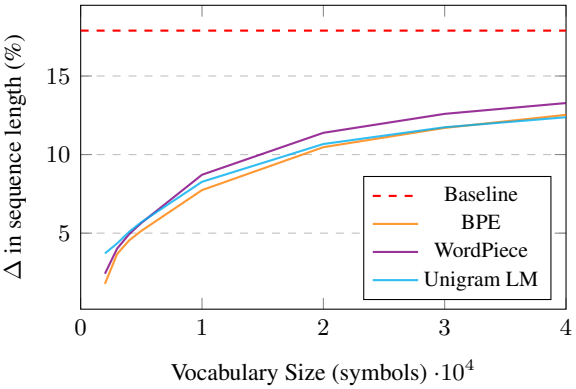


Figure 8: Relative increase in mean encoded sequence lengths (%) between BITE-less and BITE-equipped tokenizers after training the data-driven subword tokenizers with varying vocabulary sizes; lower is better. Baseline (dotted red line) denotes the percentage of inflected forms in an average sequence; this is equivalent to the increase in sequence length if BITE had no effect on the data-driven tokenizers' encoding efficiency.

Sequence lengths. A possible concern with BITE is that it may significantly increase the length of the encoded sequence, and hence the computational cost for sequence modeling, since it splits all inflected content words (nouns, verbs, and adjectives) into two symbols. We calculate the percentage of inflected words to be 17.89%.[16] Therefore, if BITE did not enhance WordPiece's and BPE's encoding efficiency, we should expect a 17.89% increase (i.e., upper bound) in their mean encoded sequence length. However, from Fig. 8, we see this is not the case as the relative increase (with and without BITE) in mean sequence length generally stays below 13%, 5% less than the baseline. This demonstrates that BITE helps the data-driven tokenizer make better use of its limited vocabulary.

In addition, we see that the gains are inversely proportional to the vocabulary size. This is likely due to the following reasons. For a given sentence, the corresponding encoded sequence's length usually decreases as the data-driven tokenizer's vocabulary size increases as it allows merging of more smaller subwords into longer subwords. On the other hand, BITE is vocabulary-independent, which means that the encoded sequence length is always the same for a given sentence. Hence, the same absolute difference contributes to a larger

[16] Note that only content words are subject to inflection.

relative increase as the vocabulary size increases. Additionally, more inflected forms are memorized as the vocabulary size increases, resulting in an average absolute increase of 0.4 symbols per sequence for every additional 10k vocabulary symbols. Together, these two factors explain the above phenomenon.