Tokenization & Inference in NLI: A Comparative Study of Methods and Models

The Influence of Tokenization on Inference Methods

Yme de Jong, Shane Siwpersad, Eduard-Raul Kőntős, Hugo Voorheijen, Julius Bijkerk
{y.m.dejong, s.r.d.siwpersad, e.r.kontos, h.j.a.voorheijen, j.j.bijkerk}@students.uu.nl

Abstract

Tokenization is a key element of modern-day LLMs. Many different strategies are used, ranging from BPE, WordPiece, to classical n-gram approaches, which can change a model's inference. This work examines how different tokenization strategies affect model performance by embracing a decoupled tokenizer approach that separates its vocabulary and tokenization function and injecting different methods in place of the native tokenization function. We propose five approaches: character-split, POS, Greedy, Unigram, and Adversarial. Each approach was tested in combination with three models: DistilRoBERTa, MiniLM2, and BART. Model performance was evaluated using accuracy and predicted label distribution. POS-based tokenizers slightly outperformed the models' native tokenizers, with the others performing worse. Our study highlights the impact tokenization has on model inference, suggesting that splitting by and keeping relevant morphemes can lead to slightly better model performance.

1 Introduction

Natural Language Inference (NLI) describes the task of determining the logical relationship between two text sequences, usually called a premise and a hypothesis. This is a crucial task for various applications, including, but not limited to, question-answering mechanisms in chatbots, text summarization tools, and translation systems. Recent AI applications employ Large Language Models (LLMs), which fine-tuned can perform specific applications. Such models can be fine-tuned to perform NLI via a supervised approach, where pairs of premises and hypotheses are classified into three possible textual entailment categories: entailment, contradiction, and neutral statements via semantic reasoning.

A critical component of state of the art (SOTA) NLI models is tokenization. Tokenization, as defined by Uzan et al. (2024), consists of two central elements: a vocabulary, denoted as V, and a tokenization function, represented by T, such that:

1. V: A vocabulary comprising a collection of individual tokens built using the tokenization function;

2. T: A tokenization function, within this context also called an inference method [1], responsible for breaking down, or segmenting, the input text into smaller units which are part of V. This function can take many forms, including Byte-Pair Encoding (BPE) (Sennrich et al., 2015), WordPiece (Wu et al., 2016), or N-grams. Throughout the training phase of a model, this function constructs a vocabulary based on a given training dataset to represent meaningful tokens.

Current SOTA models combine V and T to form the tokenizer. However, the recent work of Uzan et al. (2024) suggests decoupling them, allowing thus any tokenization function T to be applied to any selected vocabulary V. Since different tokenization functions T use different techniques to encode the input text, this may yield unexpected or undesirable results due to divergences between the provided vocabulary and the expected vocabulary of the language model.

LLMs are typically pre-trained on a large corpus to handle a wide variety of topics and general language tasks (Chang et al., 2024). On top of that, they can be further fine-tuned to narrow down and enhance the performance on specific downstream tasks (Howard and Ruder, 2018) such as

[1] When a tokenization function is specifically used to segment new text into smaller units, instead of composing a vocabulary, then it can be considered an inference method (within the context of tokenization).

NLI. However, when focusing on a specific topic or task, some tokenization functions might yield better results than others, proving them to be more effective, or leading to unexpected results (Sennrich et al., 2015). By separating V and T and preserving the contents of V, the influence of different tokenization functions during inference on natural language can be investigated and evaluated.

In this research, our focus will be on evaluating the performance of different tokenization functions with regard to NLI tasks. These tokenization functions are inserted into two distinct LLMs that are fine-tuned for NLI tasks. Specifically, our objective is to determine how different tokenization approaches affect model accuracy and prediction confidence.

This paper is organized as follows: Section 2 begins with a description of the selected models (subsection 2.1) and why they are suited for the task at hand, the datasets used for evaluation (subsection 2.2), and an overview of the different tokenization strategies and how they were implemented (subsection 2.3). Then, we will present the experimental results and their analysis in Section 3 and finally discuss the implications of our findings and propose directions for future research in Section 4.

2 Methodology

This project is aimed at researching the effects of different tokenization functions, as part of Natural Language Processing (NLP) models, on NLI task performance. To evaluate how different tokenization functions, when applied to a fixed vocabulary, influence the performance of SOTA language models on benchmark datasets, we start by decoupling vocabulary V and tokenization function F. The specific models, datasets, and tokenization functions that are used for the experiment will be detailed in the following sections.

2.1 Models

This research includes three LLMs fine-tuned for NLI, namely DistilRoBERTa (Sanh et al., 2019), MiniLM2 (Reimers and Gurevych, 2019), and BART (Lewis et al., 2019). All three models use BPE for their tokenizers. Worth noting is that DistilRoBERTa and MiniLM2 use the same processing approach for tokenization in their Hugging Face implementation, based on standard RoBERTa. All models were fine-tuned on SNLI and MNLI (both mismatched and matched splits), with BART being additionally fine-tuned on the three rounds available within Adversarial NLI (ANLI) dataset (Nie et al., 2019).

Unfortunately, all of the above models utilize the same tokenization technique, namely BPE. In our search we had not found any publicly available models that do not use BPE that can also perform NLI without fine-tuning on our part. We only encountered one model type that uses WordPiece, a BERT model (BERT-base from sentence transformer), that was fine-tuned on SNLI for semantic search but without a classification head. It would have required addition fine-tuning which was not possible on our machines.

2.2 Data

Evaluation will be performed on the Stanford Natural Language Inference (SNLI)[2] test set (Bowman et al., 2015), and the Multi-Genre Natural Language Inference (MNLI) mismatched and matched development sets (Williams et al., 2018). The SNLI dataset contains simple sentences on popular topics, whereas MNLI is a multi-genre dataset with a more diverse vocabulary and more grammatically complex sentences.

2.3 Tokenization strategies

In our research, we distinguish between six different tokenization strategies. The first category consists of the tokenizers that are native to the models used. The other four categories, character-split, POS, Greedy, Unigram, Adversarial, and their corresponding sub-strategies are developed by us. Table 4 in Appendix A showcases each tokenization methods and their effect on a sample sentence.

2.3.1 Native tokenizers

Each model contains a pre-built tokenizer that is used to implicitly tokenize sentences before being processed as input. These tokenizers are used to establish a benchmark for other tokenization methods, with the assumption that the built-in tokenizers perform better than custom tokenizers for any given model.

2.3.2 Character-split tokenizer

To determine the minimal performance that the models can achieve, we include a baseline tokenizer that splits strings into individual characters. This leads to loss of semantic meaning and less

[2]Loaded from Hugging Face (HF) using the dataset card created by the HF Datasets team

generalization on common subwords, which makes the context of any token less informative. We expect this to yield undesirable performance and thus consider the character-split tokenizer as a lower bound for overall model accuracy.

2.3.3 POS tokenizers

This tokenizer only tokenizes words of a respective POS-tag, while the rest of the sentence is converted to a token for every word. Words that are identified as having the selected POS-tag will be split into their morphological components. This is done with human-annotated data from the MorphoLex database (Sánchez-Gutiérrez et al., 2018), which contains morphological information for more than 70 thousand words. To illustrate, the adjective 'zoological' is split into 'zoo log ic al'. This morphological segmentation is not guaranteed to preserve the entirety of a word. Such behavior occurs for the word 'running', which becomes 'run'. If a word is not present within this dataset, it will be kept whole as if it were a word belonging to any other POS-tag. The three most prominent POS-tags, these being nouns (32%), verbs (12%), and adjectives (8%), are then compared for their influence on overall performance of the models (percentages based on SNLI dataset). Although determiners (19%) are more prevalent than adjectives, our intuition is that they carry less semantic importance within NLI tasks and instead serve to indicate syntactic structure.

2.3.4 Greedy tokenizers

Two "greedy" tokenization strategies are included in this research. Firstly, we examine the influence of the longest-prefix strategy, which Uzan et al. (2024) found to yield promising results. This method aims to capture maximum lexical content from the beginning of a word, by tokenizing the longest subword possible from the start. The second approach, the longest-suffix method, tokenizes the longest subword from the end of a word, ideally capturing crucial morphological endings.

2.3.5 Unigram tokenizer

Our experiment includes a unigram tokenizer, which segments text into tokens consisting of individual words. Unigrams, together with the character-split tokenization, represent the most straightforward way of splitting text. Therefore also providing a type of baseline granularity.

Initially, we considered incorporating bigrams, pairs of adjacent words, to capture more contextual information; however, we found that the default vocabularies of the three models do not support bigram entries.

2.3.6 Adversarial tokenizers

Adversarial tokenization is another avenue for exploring the impact of tokenization on NLI. There are several types of Adversarial interference (Roth et al., 2024) that aim to intentionally impact models' performance. In this work, we focused on two such approaches: Adversarial Letter Shuffling (ALS) and Adversarial Noun-Synonym Shuffling (ANSS). ALS aims to shuffle tokens' letters at random, changing thus a potential word's tokenization (e.g., "dog" might be shuffled to "odg"). Shuffling takes place during sentence parsing, where each obtained token has a probability of 20% to be shuffled. ANSS, on the other hand, represents a more targeted approach to induce perturbations. Specifically, all nouns in a given text will be replaced with their closest synonym retrieved from WordNet (pri) if existing, otherwise no perturbation is induced in that token.

3 Results and Analysis

We evaluate all possible model and tokenizer combinations on the previously mentioned datasets, resulting in a total of 90 experimental runs. We report on the obtained accuracy and distribution of predicted labels, the latter as a percentage for each of the three categories: contradiction, entailment, and neutral. Table 1 presents the overall accuracies and label distributions. Furthermore, we provide per-class accuracies in Table 5 in Appendix D.

3.1 DistilRoBERTa

Using DistilRoBERTa, the native tokenization achieved strong performance with accuracies of 0.86 on SNLI, 0.76 on MNLI-M, and 0.79 on MNLI-MM, with unbalanced label distributions (e.g., 24/50/26 for SNLI). In contrast, the character-level tokenization method performed, expectedly, poorly, yielding poor accuracies and extremely skewed label distributions (e.g., 0/99/1 on SNLI), indicating that breaking text into individual characters severely impacts the model's ability to capture semantic information. The performance of the POS-based tokenizers was virtually identical (e.g., 0.87 on SNLI, 0.8 on MNLI-M and 0.81 on MNLI-MM). These methods show a significant improvement over the character-level approach and are on par or even outperforming the native tokenizer.

Tokenization DistilRoBERTa	Dataset SNLI	MNLI-M	MNLI-MM
Native	0.86 24/50/26%	0.76 27/41/32%	0.79 22/52/26%
Character	0.34 0/99/1%	0.36 1/97/2%	0.36 1/97/2%
Noun	0.87 33/34/33%	0.8 34/34/32%	0.81 33/35/32%
Verb	0.87 33/35/32%	0.8 33/34/33%	0.81 33/35/32%
Adjective	0.87 33/34/33%	0.8 33/34/33%	0.80 33/35/32%
Greedy Suffix	0.37 5/94/1%	0.38 5/91/4%	0.38 5/92/3%
Greedy Prefix	0.37 4/94/2%	0.40 5/91/4%	0.39 5/91/4%
Unigram	**0.88** 33/34/33%	**0.82** 33/34/33%	**0.82** 33/35/32%
ALS	0.76 35/30/35%	0.69 33/29/38%	0.70 32/30/38%
ANSS	0.8 34/33/33%	0.78 34/31/35%	0.78 33/33/34%

Tokenization BART-Large	Dataset SNLI	MNLI-M	MNLI-MM
Native	**0.90** 33/33/34%	0.84 31/30/39%	0.86 30/32/38%
Character	0.46 12/59/29%	0.48 18/60/22%	0.50 17/62/21%
Noun	0.86 27/33/40%	0.86 30/32/38%	0.86 30/33/37%
Verb	0.86 28/33/39%	0.85 31/32/37%	0.85 30/33/37%
Adjective	0.86 28/33/39%	0.85 30/32/38%	0.85 30/32/38%
Greedy Suffix	0.46 42/50/8%	0.46 37/45/18%	0.46 37/44/19%
Greedy Prefix	0.46 40/52/8%	0.46 31/52/17%	0.47 32/51/17%
Unigram	**0.90** 33/33/34%	**0.88** 32/33/35%	**0.88** 32/33/35%
ALS	0.78 34/27/39%	0.74 33/25/42%	0.75 33/26/41%
ANSS	0.83 34/32/34%	0.83 33/30/37%	0.84 32/30/37%

Tokenization MiniLM2	Dataset SNLI	MNLI-M	MNLI-MM
Native	0.88 34/46/20%	0.80 29/39/32%	0.82 29/51/20%
Character	0.34 3/97/0%	0.36 7/92/1%	0.37 6/93/1%
Noun	0.78 31/36/33%	0.83 34/33/33%	0.83 33/34/33%
Verb	0.68 35/44/21%	0.82 34/33/33%	0.83 33/34/33%
Adjective	0.66 33/41/26%	0.82 34/33/33%	0.83 33/34/33%
Greedy Suffix	0.40 57/42/1%	0.39 44/55/1%	0.40 42/57/1%
Greedy Prefix	0.44 46/53/1%	0.40 34/65/1%	0.41 34/65/1%
Unigram	**0.89** 33/34/33%	**0.84** 33/34/33%	**0.85** 33/35/32%
ALS	0.76 36/30/34%	0.71 37/27/36%	0.72 34/29/37%
ANSS	0.81 34/32/34%	0.80 34/31/35%	0.81 33/32/35%

Table 1: Results for each model and tokenization combination, across all datasets. We report the accuracy and distribution of predicted labels (formatted as such: contradiction/entailment/neutral). The left table presents results for tokenization using DistilRoBERTa for classification, the middle one for BART-Large, and the right table for MiniLM2. Each table is divided into five sections according to tokenization approach. MNLI-M refers to the matched development set, and MNLI-MM refers to the mismatched development set.

POS tokenizers	
Type	**Modification rate**
Noun	8.7%
Verbs	10.3%
Adjective	1.6%

Table 2: Modification rate refers to the percentage of words within the SNLI dataset that are successfully split into their morphological components and change as a result. Words that are included in MorphoLex but are not changed by this transformation do not contribute to this percentage.

In addition the POS-based tokenizers performed better on SNLI than on the MNLI datasets. The greedy-based tokenization methods (both suffix and prefix) achieved accuracies of approximately 0.37 to 0.40, again underperforming relative to the native baseline. Notably, unigram tokenization also beat the native performance (0.88 on SNLI, 0.82 on both MNLI subsets) and outperformed the POS-based tokenizers, suggesting that unigram segmentation is a viable alternative. Finally, the alternative methods ALS and ANSS attained moderate accuracies (ranging from 0.69 to 0.80), which is worse than the unigram and POS-based tokenizations.

3.2 BART-Large

For BART-Large, native tokenization led to accuracies of 0.90 on SNLI ,0.84 on MNLI-M, and 0.86 on MNLI-MM, with balanced label distributions (e.g.,33/33/34% on SNLI). Similar to the findings with DistilRoBERTa, character-level tokenization performed worse (with accuracies between 0.46 and 0.50 and the predictions were less balanced, with a bias towards entailment. Interestingly, when using POS-based tokenization (noun, verb, or adjective), the performance was close to the native baseline, with all three methods reaching around 85–86 accuracy across the datasets. In contrast, both greedy suffix and prefix methods again performed worse, with accuracies in the mid-0.40s and distributions that were skewed with a bias towards entailment (e.g., 42/50/8% for Greedy Suffix on SNLI). The unigram method matched native tokenization in accuracy on SNLI and outperformed the native tokenizer on the MNLI datasets. The ALS and ANSS methods also followed a similar trend as could be observed with DistilRoBERTa, namely slightly lower accuracies (ranging from 0.74 to 0.84) while having relatively balanced label distributions.

3.3 MiniLM2

MiniLM2 exhibited trends similar to the other models: native tokenization achieved accuracies of 0.88 (SNLI), 0.80 (MNLI-M), and 0.82 (MNLI-MM) with unbalanced prediction distributions (e.g.,34/46/20% on SNLI). The character-level method again performed significantly worse than the native tokenizer, with accuracies in the 0.34–0.37 range. For the POS-based approaches, noun-based tokenization performed reasonably well on SNLI (0.78), MNLI-M (0.83), and on MNLI-MM (0.83); verb and adjective tokenizations followed similar patterns, with a performance

that slightly outperforms that of the native tokenizer. Greedy-based tokenizations, both suffix and prefix, resulted in accuracies around 0.40 to 0.44, performing significantly worse compared to the native tokenizer and POS-tokenizing methods, with severely biased predictions. Unigram tokenization outperformed the native tokenizer on each dataset (ranging from 0.84 to 0.89), while ALS and ANSS scored between 0.71 and 0.81 - with ANSS outscoring ALS on all datasets.

3.4 Analysis

Our experiments reveal several notable patterns in how different tokenization methods affect label distributions and model performance across datasets. First of all, we noticed that unigram tokenization produces a nearly uniform distribution (e.g., approximately 33/34/33% or 33/35/32% across the three classes), implying that the models retain their capacity to differentiate between entailment, contradiction, and neutrality. In contrast, character-level tokenization leads to a significant drop in performance and shows severe bias towards entailment. For example, DistilRoBERTa with character-level tokenization on SNLI yields an accuracy as low as 0.34 with a distribution of roughly 0/99/1%. Similar imbalances are observed with both greedy prefix and greedy suffix approaches, where the accuracy remains low (typically in the range of 0.37–0.47) and the predictions are dominated by one label - entailment. This could be explained by the fact that these types of tokenization split words too aggressively, words which are necessary for context and to capture nuance. With barely anything preserved, the models may focus on incorrect semantic alignment for key entities or actions, which leads to overpredicting entailment.

POS tokenization yields mixed results. With BART-Large, the performance remains relatively stable (around 0.85–0.86 accuracy) and the prediction distributions are relatively balanced. However, in the case of DistilRoBERTa and MiniLM2, tokenizing by verbs or adjectives results in slightly lower accuracies and a slight bias toward the middle class.

These findings indicate that while linguistically motivated tokenizations can sometimes approximate the native tokenization performance, their success may be model-dependent.

In addition, ALS and ANSS do not perform as well as native or unigram approaches. Both methods reduce the overall accuracy across models and datasets; however, ANSS tends to yield slightly more balanced prediction distributions than ALS. For instance, with DistilRoBERTa on SNLI, ALS results in a distribution of approximately 35/30/35 while ANSS is closer to 34/33/33. This suggests that even among alternative tokenization strategies, those that maintain a more balanced view of the input tend to perform better in terms of both accuracy and fairness of predictions.

There is a general trend across most tokenization methods. Performance on SNLI generally exceeds that on the MNLI datasets. This trend seems to only be broken by the POS and character tokenization methods, and in rare isolated cases for other tokenization methods. It also only happened with MiniLM2 on the MNLI-M dataset. Native tokenization achieves scores between 0.86 and 0.90 on SNLI, while the MNLI datasets typically see a decrease (with scores in the range of 0.76 to 0.84). Despite this overall decline in accuracy for MNLI, the relative impact of tokenization remains relatively consistent: approaches that preserve the original token structure, such as native and unigram tokenization, outperform those that drastically change it, like character-level and greedy tokenizations, regardless of the dataset or model used.

4 Conclusion

Our study highlights the significant impact of tokenization choices on Natural Language Inference (NLI) performance. Across multiple models and datasets, we observe that the unigram design is the only tokenizer that consistently outperforms native tokenizers. Character-level and greedy-based tokenizations severely degrade model performance through biases towards entailment, likely due to the destruction of semantic structures within the text. Among POS-based tokenization methods, all three versions perform on par with or even exceed the native tokenizers, depending on the model and dataset. We mostly attribute this to the influence of the large majority of tokens that are kept whole. Additionally, the segmentation strategies ALS and ANSS slightly underperform when compared to the native tokenizers. These findings suggest that tokenizers that perform better when designed with the retainment of subwords in mind, while avoiding excessive fragmentation.

5 Code submission

The code that is used for this research is publicly available and can be found on GitHub.

6 Contributions

As part of this research, we had weekly meetings with all authors. First of all, to define our research question and the scope of it. Secondly, to create a concrete project outline and task distribution, which is shown in Table 3. Lastly, to discuss experiment considerations, results and writing updates, which we also did on continuous basis.

	Yme	**Shane**	**Eduard**	**Hugo**	**Julius**
Methodology development	X	X	X	X	X
Datasets	X	X	X	X	X
Tokenizer Strategies	X		X	X	
Native Tokenizers		X		X	X
Char-split tokenizers			X		
POS tokenizer	X	X			
Length-dependent tokenizers				X	X
Adversarial tokenizers			X		
Evaluation	X	X	X	X	X
Paper: Abstract	X		X		
Paper: Introduction	X		X	X	X
Paper: Methodology	X	X	X		X
Paper: Results and Analysis	X	X	X	X	X
Paper: Conclusion	X	X			

Table 3: Contribution table

7 Acknowledgments

First of all, we would like to thank Dr. Lasha Abzianidze PhD, for his guidance and feedback throughout this project. His expertise and insights have significantly shaped the direction and execution of our research. Secondly, we are thankful for the students, participating in the course Logic and Language 24/25 at Utrecht University , for their presentations, broadening our knowledge on the topic of NLI. Finally, we would like to recognize the strength of the open-source models and datasets, especially those used in our experiment, highlighting how collaborative efforts facilitate new research.

References

WordNet — wordnet.princeton.edu. https://wordnet.princeton.edu/. [Accessed 02-02-2025].

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. 2024. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):39:2.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Sander Land and Max Bartolo. 2024. Fishing for magikarp: Automatically detecting under-trained tokens in large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11631–11646, Miami, Florida, USA. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461.

Scott Lundberg. 2017. A unified approach to interpreting model predictions. *arXiv preprint arXiv:1705.07874*.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2019. Adversarial nli: A new benchmark for natural language understanding. *arXiv preprint arXiv:1910.14599*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Tom Roth, Yansong Gao, Alsharif Abuadbba, Surya Nepal, and Wei Liu. 2024. Token-modification adversarial attacks for natural language processing: A survey. *AI Commun.*, 37(4):655–676.

Claudia H Sánchez-Gutiérrez, Hugo Mailhot, S Hélène Deacon, and Maximiliano A Wilson. 2018. Morpholex: A derivational morphological database for 70,000 english words. *Behavior research methods*, 50:1568–1580.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *CoRR*, abs/1508.07909.

Omri Uzan, Craig W. Schmidt, Chris Tanner, and Yuval Pinter. 2024. Greed is all you need: An evaluation of tokenizer inference methods.

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, New Orleans, Louisiana. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

A Example tokenization

Examples of tokenzations can be found in Figure 4.

B Undertrained tokens

Following in the work of (Land and Bartolo, 2024), we wanted to test whether under-trained tokens affect the models' performance. We checked the vocabularies of all models and found 988 under-trained tokens for DistilRoBERTa, 999 for MiniLM2, and 1001 for BART. Yet, upon closer inspection these tokens were only present in 30 examples from SNLI, 246 from MNLI matched, and 201 for MNLI mismatched. Moreover, there was no meaningful difference in performance compared to the sentences that did not contain under-trained tokens, which we believe indicates that under-trained tokens do not necessarily pose an issue for these datasets.

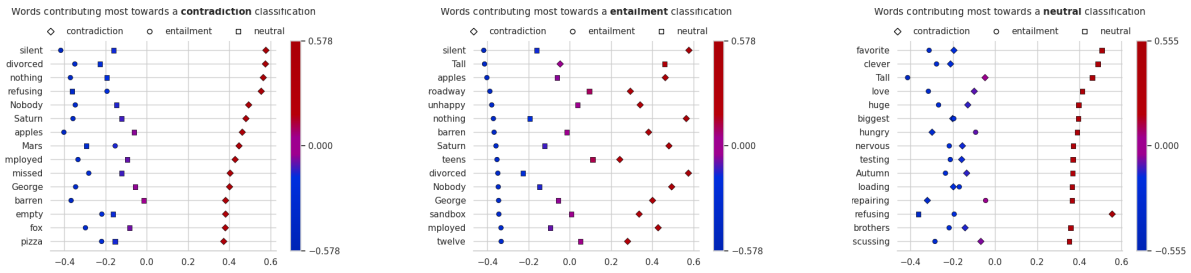
C Dataset-specific SHAPley analysis

We additionally performed a SHAP-based analysis, referring to SHapley Additive exPlanations (Lundberg, 2017), to determine which tokens contribute most towards a prediction. Due to its dataset-specific nature, we present a short analysis only for DistilRoBERTa and MiniLM2 on SNLI. The plots in Figures 1 and 2 show the tokens contributing the most towards each NLI label. As mentioned before, due to its dataset-centric nature, creating a tokenization approach based on feature importance is not a feasible feat as each dataset might have different tokens contributing to a label, influenced by the frequency of a token relative to the entailment labels.

D Per-class accuracy

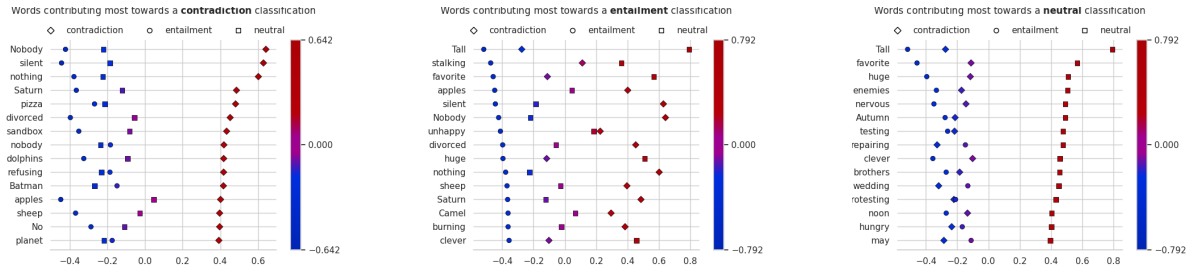
Tokenization	Sentence
Character	P,e,o,p,l,e,_a,r,e,_r,i,d,i,n,g,_b,i,k,e,s,_o,n,_t,h,o,s,e,_t,r,a,i,l,s
POS (Noun)	People,_are,_riding,_bike,_on,_those,_trail
POS (Verb)	People,_are,_ride,_bikes,_on,_those,_trails
POS (Adjective)	_People,_are,_riding,_bikes,_on,_those,_trails
Greedy (Suffix)	Pe,op,le,_a,re,_ri,di,ng,_b,ik,es,_on,_t,ho,se,_tr,ai,ls
Greedy (Prefix)	Pe,op,le,_ar,e,_ri,di,ng,_bi,ke,s,_on,_th,os,e,_tr,ai,ls
Unigram	People,_are,_riding,_bikes,_on,_those,_trails
ALS	People,_are,_riding,_bike,_on,_those,_trail
ANSS	People,_are,_riding,_bikes,_on,_hseot,_trails

Table 4: Effect of tokenization on a sample sentence: “People are riding bikes on those trails”. An underscore “_” corresponds to the beginning of a word relative to the original sentence, which we call a “space marker”. For example, in the case of the Character-split tokenizer, the “a” in “are” is preceded by an underscore because it marks the beginning of the word, while the split characters “r” and “e” do not. It is worth noting that besides the “space marker”, some tokenizers also use “split markers” to denote tokens that have been split from their root (e.g., “##”). Our chosen models, however, do not use “split markers” and only use “Ġ” to denote the beginning of a token.



(a) (b) (c)

Figure 1: Mean SHAP values for 2500 samples from SNLI, for MiniLM2. Presented are the top 15 most contributing features (i.e., tokens) towards a label. Each feature’s contribution toward each classification label is shown, whereas the three plots show the sorted tokens contributing towards that label. For example, plot (a) lists the tokens, in descending order, that contribute most towards the “contradiction” label.



(a) (b) (c)

Figure 2: Mean SHAP values for 2500 samples from SNLI, for DistilRoBERTa. Presented are the top 15 most contributing features (i.e., tokens) towards a label. Each feature’s contribution toward each classification label is shown, whereas the three plots show the sorted tokens contributing towards that label. For example, plot (a) lists the tokens, in descending order, that contribute most towards the “contradiction” label.

Tokenization	Dataset			Tokenization	Dataset			Tokenization	Dataset		
DistilRoBERTa	SNLI	MNLI-M	MNLI-MM	BART-Large	SNLI	MNLI-M	MNLI-MM	MiniLM2	SNLI	MNLI-M	MNLI-MM
Native	0.86	0.76	0.79	Native	**0.90**	0.84	0.86	Native	0.88	0.80	0.82
	24/50/26%	27/41/32%	22/52/26%		33/33/34%	31/30/39%	30/32/38%		34/46/20%	29/39/32%	29/51/20%
	86/92/82	70/85/73	77/85/77		95/91/89	87/81/90	87/85/90		90/93/85	80/87/78	83/88/81
Character	0.34	0.36	0.36	Character	0.46	0.48	0.50	Character	0.34	0.36	0.37
	0/99/1%	1/97/2%	1/97/2%		12/59/29%	18/60/22%	17/62/21%		3/97/0%	7/92/1%	6/93/1%
	0/99/1	1/98/4	1/98/3		20/78/42	33/76/35	35/79/34		4/98/0	9/95/2	8/96/2
Noun	0.87	0.8	0.81	Noun	0.86	0.86	0.86	Noun	0.78	0.83	0.83
	33/34/33%	34/34/32%	33/35/32%		27/33/40%	30/32/38%	30/33/37%		31/36/33%	34/33/33%	33/34/33%
	90/89/85	84/82/78	84/84/77		81/89/92	87/85/90	85/86/90		92/90/86	89/83/80	88/85/81
Verb	0.87	0.8	0.81	Verb	0.86	0.85	0.85	Verb	0.68	0.82	0.83
	33/35/32%	33/34/33%	33/35/32%		28/33/39%	31/32/37%	30/33/37%		35/44/21%	34/33/33%	33/34/33%
	91/90/85	83/82/78	83/84/78		82/90/92	86/85/90	84/86/90		93/91/86	88/83/80	87/85/82
Adjective	0.87	0.8	0.80	Adjective	0.86	0.85	0.85	Adjective	0.66	0.82	0.83
	33/34/33%	33/34/33%	33/35/32%		28/33/39%	30/32/38%	30/32/38%		33/41/26%	34/33/33%	33/34/33%
	91/88/86	83/81/79	83/83/77		82/89/92	85/85/90	84/86/90		93/90/87	87/83/80	87/84/81
Greedy Suffix	0.37	0.38	0.38	Greedy Suffix	0.46	0.46	0.46	Greedy Suffix	0.40	0.39	0.40
	5/94/1%	5/91/4%	5/92/3%		42/50/8%	37/45/18%	37/44/19%		57/42/1%	44/55/1%	42/57/1%
	8/97/3	8/95/7	8/95/5		56/67/14	49/60/28	51/60/28		66/56/0	48/66/1	48/71/1
Greedy Prefix	0.37	0.40	0.39	Greedy Prefix	0.46	0.46	0.47	Greedy Prefix	0.44	0.40	0.41
	4/94/2%	5/91/4%	9/96/**9**1/4%		40/52/8%	31/52/17%	32/51/17%		46/53/1%	34/65/1%	34/65/1%
	7/98/4		9/96/8		56/70/14	45/67/28	47/67/28		60/72/0	39/79/1	39/80/2
Unigram	**0.88**	**0.82**	**0.82**	Unigram	**0.90**	**0.88**	**0.88**	Unigram	**0.89**	**0.84**	**0.85**
	33/34/33%	33/34/33%	33/35/32%		33/33/34%	32/33/35%	32/33/35%		33/34/33%	33/34/33%	33/35/32%
	92/90/86	85/84/80	85/85/79		95/91/90	91/87/90	91/87/90		93/92/88	89/86/83	88/88/83
ALS	0.76	0.69	0.70	ALS	0.78	0.74	0.75	ALS	0.76	0.71	0.72
	35/30/35%	33/29/38%	32/30/38%		34/27/39%	33/25/42%	33/26/41%		36/30/34%	37/27/36%	34/29/37%
	82/72/76	73/63/73	73/65/75		84/70/83	79/63/84	78/65/85		84/73/76	79/63/75	78/66/76
ANSS	0.8	0.78	0.78	ANSS	0.83	0.83	0.84	ANSS	0.81	0.80	0.81
	34/33/33%	34/31/35%	33/33/34%		34/32/34%	33/30/37%	32/30/37%		34/32/34%	34/31/35%	33/32/35%
	84/81/77	83/76/79	82/78/78		88/82/82	88/77/87	88/80/88		85/82/79	87/77/81	85/80/81

Table 5: Additional table that provides the following information in order: overall accuracy (range 0-1), predicted label distribution, and per-label accuracy (range 0-100%) for each tokenization method.