A Survey of Subword Tokenization Methods in Neural Natural Language Processing

Authors: Liu Qun, Liu Qun

Date: 2022-05-04T23:06:44+00:00

Abstract

This paper presents a comprehensive survey of subword tokenization methods in neural network-based natural language processing. It first elaborates on the out-of-vocabulary (OOV) problem stemming from closed vocabulary limitations in such approaches, and introduces three prevalent methods for addressing this issue: Byte-Pair Encoding (BPE), WordPiece, and Unigram. Subword tokenization conventionally necessitates prior word segmentation, which is highly language-specific. SentencePiece provides a language-independent subword tokenization method that operates directly on input sentences, obviating the need for preliminary word segmentation. However, subword tokenization may occasionally exhibit unreasonable segmentation and insufficient learning of subword representations; this paper subsequently introduces subword regularization and BPE-Dropout techniques to remedy these problems. Character-based subword tokenization continues to encounter OOV issues when confronted with large character sets in multilingual contexts, particularly for Chinese, Japanese, and Korean languages; this paper will present an effective solution: UTF-8 byte-level BPE technology (Byte Level BPE, abbreviated as BBPE) and its derived SentencePiece scheme based on BBPE. Finally, this paper introduces VOLT, a general vocabulary optimization technique proposed in the ACL 2021 Best Paper.

Full Text

Preamble

This paper provides a comprehensive survey of subword tokenization methods in neural natural language processing (NLP). We begin by explaining the out-of-vocabulary (OOV) problem arising from the closed vocabulary assumption in neural NLP approaches, and introduce three common solutions: Byte-Pair Encoding (BPE), WordPiece, and Unigram. Traditional subword segmentation requires prior word tokenization, which is highly language-dependent. SentencePiece offers a language-independent alternative that performs subword segmentation directly on input sentences without requiring word tokenization. Subword segmentation can sometimes produce unreasonable splits and lead to insufficient learning of subword representations. We subsequently discuss subword regularization and BPE-Dropout techniques to address these issues. For multilingual scenarios, particularly languages with large character sets such as Chinese, Japanese, and Korean (CJK), character-based subword segmentation still faces OOV problems. We present an effective solution: UTF-8 byte-level BPE (Byte-level BPE, BBPE) and its SentencePiece variant. Finally, we introduce VOLT, a general vocabulary optimization technique proposed in the ACL 2021 Best Paper.

1.1 Closed Vocabulary Assumption and Out-of-Vocabulary Problem

Natural language processing deals with text whose vocabulary is inherently open—there is no restriction on which words may appear. However, NLP systems typically assume that all words belong to a predefined, finite vocabulary. This assumption is known as the closed vocabulary assumption. For neural NLP methods, this presents a critical challenge: we must map each word to a pre-trained embedding representation, making it impossible to handle words outside the predefined vocabulary (out-of-vocabulary words, or OOV). This is the fundamental OOV problem faced by NLP systems.

1.2 Solution 1: Replacing OOV with UNK

Early neural NLP systems addressed this problem with a simple heuristic: replacing all OOV words with a special "UNK" token representing unknown words. However, this approach has significant limitations. Many OOV words, while absent from training data, are not semantically opaque; rather, their meanings can often be inferred from known words. Replacing them with UNK discards valuable semantic information. For instance: - **Inflectional variants**: Word vocabularies may not include all morphological forms (e.g., English plurals, comparatives, verb conjugations). Unlisted forms become OOV. - **Compound words**: Such as "state-of-the-art" or "machine-learning." - **Numerals**: It is impossible to include all numbers in a vocabulary.

While UNK tokens may be tolerable in natural language understanding tasks, they severely degrade performance in natural language generation tasks like machine translation and dialogue generation, where users find sentences containing numerous UNKs unacceptable. Simply removing UNKs also results in ungrammatical output.

1.3 Solution 2: Character-based Models

Character-based models operate directly on characters, computing word representations from character representations through a neural network. While this approach effectively solves the OOV problem, it dramatically increases sequence length, making modeling substantially more difficult. Character-based models typically require significantly more complex architectures to achieve comparable performance to word-based models.

For languages with large character sets like CJK, character-based models still face vocabulary size issues. The Unicode standard for CJK unified ideographs already contains tens of thousands of characters, with newer versions including over 90,000 characters. Incorporating all characters into a model is impractical, as most are extremely rare yet consume substantial model parameters. Furthermore, generating each character requires computing a softmax over the entire vocabulary, which becomes computationally inefficient with such large, sparse vocabularies.

2.1 Subword-level Segmentation: An Effective Solution to OOV

As discussed, word-level vocabularies cannot solve the OOV problem, while character-level vocabularies suffer from high model complexity and low efficiency. Researchers have proposed subword-level vocabulary construction and segmentation methods that effectively address these limitations. The core idea is: - The vocabulary contains a fixed set of subwords. - Any word not in the vocabulary is segmented into a subword sequence.

This approach offers several advantages: it completely solves the OOV problem, as any word can be segmented into subwords (falling back to characters in the worst case), and most common words can be included as independent subwords, preserving the efficiency of word-based models with minimal sequence length increase.

Two key questions arise: (1) How to construct the subword vocabulary? (2) Given a vocabulary, how to segment sentences into subword sequences? For Western languages, one might consider morphological analysis (splitting words into stems and affixes). However, this approach is problematic: not all words can be decomposed this way (e.g., loanwords, numerals), and morphological analysis is highly language-dependent, lacking universality.

Currently, the most popular methods for subword vocabulary construction and segmentation are BPE, WordPiece, and the less commonly used Unigram method.

2.2 Byte-Pair Encoding (BPE)

Originally developed as a compression algorithm, BPE was first adapted for subword segmentation to address OOV in neural machine translation, achieving remarkable success. It quickly became the dominant subword segmentation method in neural NLP models.

BPE vocabulary construction proceeds as follows (illustrated in Table 1): 1. Preprocess text to create a dictionary containing all words and their frequencies. 2. Split all words into character sequences separated by spaces. 3. Initialize the subword vocabulary with all basic characters (letters, digits, punctuation, etc.). 4. Iteratively merge the most frequent pair of subwords in the dictionary, adding the merged subword to the vocabulary, until reaching the target vocabulary size.

BPE-based subword segmentation is straightforward: replace inter-word spaces with a special token, split all words into characters, then repeatedly apply merge operations from the BPE vocabulary in order until no more merges are possible.

2.3 WordPiece

WordPiece was first introduced in neural machine translation systems for subword segmentation, though the algorithm was not initially named. While Google has not open-sourced the WordPiece vocabulary construction algorithm, they provide tokenization tools in their codebase, enabling researchers to use pre-built vocabularies. Some later implementations claim to provide WordPiece training tools but actually use BPE to construct vocabularies, merely converting them to WordPiece format.

WordPiece’ s vocabulary construction resembles BPE, starting from a basic character set and iteratively merging pairs. However, while BPE simply merges the most frequent pairs, WordPiece employs a more complex criterion. It first constructs a language model from the current vocabulary, then computes the likelihood of the entire training data. When considering all possible subword pairs for merging, it selects the pair that maximizes the training data likelihood after merging. This process is computationally expensive, and Google likely employs proprietary optimizations that remain undisclosed.

During inference, WordPiece uses left-to-right maximum matching (a greedy approach), which is simple and efficient. BPE segmentation can also use this method with similar results to the bottom-up merging approach.

2.4 Unigram

Google proposed another vocabulary construction method called Unigram. Like WordPiece, it uses a language model to determine subword inclusion, specifically a unigram language model, hence its name. Unlike BPE and WordPiece, Unigram does not start with characters and merge, but begins with a large vocabulary (e.g., all words in the corpus) and gradually removes subwords, splitting rare subwords into more common ones until reaching the target size. While implemented in an open-source toolkit, this method is rarely used due to low computational efficiency. During inference, Unigram also employs left-to-right maximum matching.

3 SentencePiece: A Language-Independent Subword Segmentation Method

The methods described above require word tokenization before subword segmentation, introducing language dependency. SentencePiece addresses this by performing subword segmentation directly on raw text without prior word tokenization. Conceptually, it treats each sentence (or paragraph) as a single word and constructs subword vocabularies accordingly. In this approach, spaces are treated as regular characters, making it common for subwords to contain spaces or mixtures of letters, digits, and punctuation. Experiments demonstrate that SentencePiece achieves comparable performance to word-dependent methods while eliminating the need for language-specific tokenization.

SentencePiece can be applied to Chinese text directly without word segmentation. While some practitioners perform Chinese word segmentation first before applying SentencePiece, this negates its primary advantage and yields results similar to standard BPE. Critically, inference must use the same tokenization scheme as training; otherwise, performance degrades significantly.

It is important to distinguish SentencePiece as a framework rather than a specific algorithm. While WordPiece, BPE, and Unigram are algorithms for vocabulary construction and segmentation, SentencePiece is a methodology that operates without word tokenization. Implementing SentencePiece still requires choosing one of these algorithms for actual vocabulary construction and segmentation. The SentencePiece toolkit provides implementations of both BPE and Unigram.

4 Subword Regularization and BPE-Dropout: Addressing Segmentation Issues

Subword segmentation, like word segmentation, suffers from ambiguity. Careful examination of BPE or WordPiece outputs reveals many unreasonable splits. BPE can also lead to under-trained subword representations. For example, a word might have both reasonable and unreasonable segmentations, but due to data distribution, the unreasonable segmentation may appear more frequently in the corpus, resulting in insufficient training for certain subwords.

Subword regularization addresses this by introducing a language model (e.g., Unigram) to sample segmentation results probabilistically, favoring more reasonable splits. However, this increases computational cost during training. BPE-Dropout offers a simpler solution: during training, randomly skip some merge operations with a certain probability, increasing segmentation diversity and improving representation learning for low-frequency subwords.

Both methods introduce randomness only during training; inference remains deterministic to ensure consistency and efficiency.

5 Byte-level BPE (BBPE): Solving OOV for Large Character Set Languages

For CJK languages with large character sets, standard subword methods face two issues: (1) the character set may exceed the vocabulary size, causing OOV problems, and (2) including all characters wastes vocabulary space on rare characters, reducing decoding efficiency.

BBPE solves this by representing all characters as UTF-8 byte sequences and applying BPE at the byte level. This limits the basic vocabulary to at most 256 units, eliminating the rare character problem. While a character may be split across multiple subwords, potentially creating invalid character sequences, this rarely occurs in practice because neural language models learn the valid subword sequence distribution from training data.

BBPE was first adopted in GPT-2 and subsequently used in RoBERTa and GPT-3. Our implementation at Huawei Noah’ s Ark Lab provides complete BBPE code with SentencePiece support, addressing the lack of such integration in existing tools.

6 VOLT: Vocabulary Optimization Method

A key question in subword vocabulary construction is: what size should the vocabulary be? Larger vocabularies generally improve performance but increase model size and inference time. VOLT (Vocabulary Learning via Optimal Transport) provides a principled answer to this trade-off, earning the ACL 2021 Best Long Paper award.

Since downstream task performance lacks a unified metric, VOLT uses vocabulary entropy as a proxy: larger vocabularies yield lower entropy and better performance. The method defines the marginal utility of vocabulary (MUV) as the entropy reduction per unit increase in vocabulary size. VOLT formulates vocabulary search as an optimal transport problem solvable in polynomial time via dynamic programming. Experiments on 14 machine translation directions show that VOLT can reduce vocabulary size to 40% of baseline systems while maintaining or even improving performance.

Conclusion

This survey has covered subword tokenization methods in neural NLP. We introduced the OOV problem arising from the closed vocabulary assumption and explained why subword segmentation provides an effective solution. We described standard methods for vocabulary construction and segmentation (BPE, WordPiece, Unigram), then addressed related challenges: (1) the need for word tokenization and the language-independent SentencePiece solution; (2) segmentation ambiguity and under-training, mitigated by subword regularization and BPE-Dropout; (3) large character set issues solved by BBPE; and (4) vocabulary optimization through VOLT. Subword tokenization is a fundamental technique in neural NLP, and comprehensive understanding of these methods is essential for designing better NLP systems.

Note: Figure translations are in progress. See original paper for figures.

Source: ChinaXiv —Machine translation. Verify with original.