Scaffold-BPE: Enhancing Byte Pair Encoding for Large Language Models with Simple and Effective Scaffold Token Removal

Haoran Lian[1], Yizhe Xiong[2,3], Jianwei Niu[1,4,5,6*], Shasha Mo[1*], Zhenpeng Su[7], Zijia Lin[2*], Hui Chen[2,3*], Jungong Han[2], Guiguang Ding[2,3*]

[1]Beihang University,
[2]Tsinghua University,
[3]BNRist,
[4]State Key Laboratory of Virtual Reality Technology and Systems, Beihang University,
[5]Zhongguancun Laboratory,
[6]Zhengzhou University Research Institute of Industrial Technology, Zhengzhou University,
[7]Chinese Academy of Sciences
{lianhaoran,niujianwei,moshasha}@buaa.edu.cn, xiongyizhe2001@163.com, suzhenpeng@iie.ac.cn,
linzijia07@tsinghua.org.cn, {jichenhui2012,jungonghan77}@gmail.com, dinggg@tsinghua.edu.cn

Abstract

Byte Pair Encoding (BPE) serves as a foundation method for text tokenization in the Natural Language Processing (NLP) field. Despite its wide adoption, the original BPE algorithm harbors an inherent flaw: it inadvertently introduces a frequency imbalance for tokens in the text corpus. Since BPE iteratively merges the most frequent token pair in the text corpus to generate a new token and keeps all generated tokens in the vocabulary, it unavoidably holds tokens that primarily act as components of a longer token and appear infrequently on their own. We term such tokens as **Scaffold Tokens**. Due to their infrequent occurrences in the text corpus, Scaffold Tokens pose a learning imbalance issue. To address that issue, we propose **Scaffold-BPE**, which incorporates a dynamic scaffold token removal mechanism by parameter-free, computation-light, and easy-to-implement modifications to the original BPE method. This novel approach ensures the exclusion of low-frequency Scaffold Tokens from the token representations for given texts, thereby mitigating the issue of frequency imbalance and facilitating model training. On extensive experiments across language modeling and even machine translation, Scaffold-BPE consistently outperforms the original BPE, well demonstrating its effectiveness.

1 Introduction

In recent years, Large Language Models (LLMs) have become a burgeoning paradigm in handling a broad array of Natural Language Processing (NLP) tasks. The tokenization process in most modern LLMs (Radford et al. 2019; Brown et al. 2020; Rae et al. 2021; Zhang et al. 2022; Biderman et al. 2023; Touvron et al. 2023a; Yang et al. 2023; Achiam et al. 2023; Dubey et al. 2024; Lian et al. 2024a) employs Byte Pair Encoding (BPE) (Sennrich et al. 2015), a method that was originally designed for data compression (Gage 1994). BPE consists of two main stages. In the training stage, BPE iteratively merges the most frequent pairs of bytes or characters in a dataset into a new token and adds it to the vocabulary until a desired vocabulary size is reached. And in the encoding stage, the vocabulary is utilized to represent any text. The adoption of BPE in LLMs is driven by its capability to decompose words into smaller, manageable subword units, thus avoiding out-of-vocabulary words, facilitating flexible and semantically complete representations of input data. Actually, BPE has also been widely used in traditional NLP tasks, like machine translation (Provilkov et al. 2019; Xu et al. 2020), information extraction (Wei et al. 2021, 2023b,a) and summarization (Wu et al. 2021; Xu et al. 2022).

Since its inception, BPE has undergone various modifications to better suit the needs of complex NLP tasks, including identifying the optimal vocabulary size for various tasks (Xu et al. 2020; Gutierrez-Vasques et al. 2021), optimizing the encoding paths of tokens to achieve subword regularization (Provilkov et al. 2019; He et al. 2020), etc.

However, there is an inherent limitation in the BPE method: the iterative merging process can lead to an imbalance in token frequencies by including low-frequency tokens in vocabulary. For example, as illustrated in Figure 1, in the commonly used Pile dataset (Gao et al. 2020) that is tokenized by the original BPE method of 32K vocabulary size (as LLaMA series (Touvron et al. 2023a,b)), the token "zona" mostly appears as a component of the token "Arizona" rather than as an independent, high-frequency token. Despite its lower standalone frequency, BPE includes "zona" in the final vocabulary because it is the "intermediate token" to derive the frequent token "Arizona". We define such intermediate tokens that are crucial for constructing longer frequent tokens but do not appear frequently on their own as **Scaffold Tokens**. Note that not all subwords are simply scaffold tokens. For example, "ing" is not identified as a scaffold token, as there are many words containing "ing" but are not tokens in the vocabulary. For example, "connecting" is represented as "connect"+"ing" (2 tokens). Such words help to keep "ing" a frequent token. Therefore, "ing" is not

*Corresponding Authors.

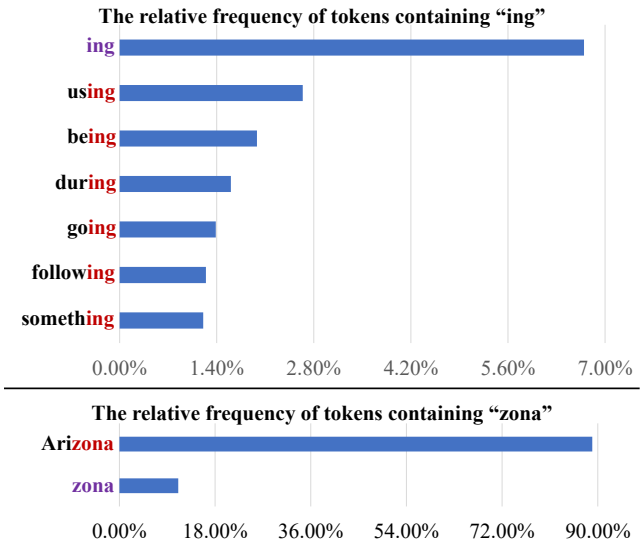

Figure 1: Two types of tokens in original BPE vocabulary on the Pile dataset: one type, such as “ing”, appears frequently by itself, while the other type, such as “zona”, mostly appears as a component of “Arizona” and thus has a low individual occurrence frequency. The value on the horizontal axis represents the percentage of the frequency of the token relative to the total frequency of all tokens containing “ing”/“zona”. For “ing”, we only visualize the top tokens.

a scaffold token. According to our proposed Scaffold-BPE method, the 32K vocabulary contains about 6.07% of scaffold tokens.

As depicted in Figure 2, a natural frequency imbalance arises between these scaffold tokens and actual high-frequency tokens. Prior studies (Lin et al. 2017; Su et al. 2023) have highlighted that such disparities in token frequencies can result in imbalanced learning difficulties across different tokens. Scaffold tokens, due to their lower individual occurrence frequencies, are notably harder to learn for models.

To address that issue, we propose enhancements to the BPE algorithm, aiming to mitigate the frequency imbalance and ensure a more equitable learning process for all tokens. Specifically, we propose the simple and effective **Scaffold-BPE** with a dynamic scaffold token removal mechanism, which is parameter-free, computation-light, easy-to-implement, and widely effective. Generally, the proposed Scaffold-BPE maintains an expanded vocabulary compared with the original BPE, which consists of both normal tokens and scaffold tokens. Note that the scaffold tokens are not actual tokens in the vocabulary and do not appear in the tokenized sequences after encoding. In the training stage, Scaffold-BPE dynamically marks tokens with lower individual occurrence frequencies as scaffold tokens in each iteration. In the encoding stage, the Scaffold-BPE firstly utilizes all tokens in the expanded vocabulary to generate the token representations for the given texts, which is termed as a *Scaffolding* process. Then, the Scaffold-BPE ensures the

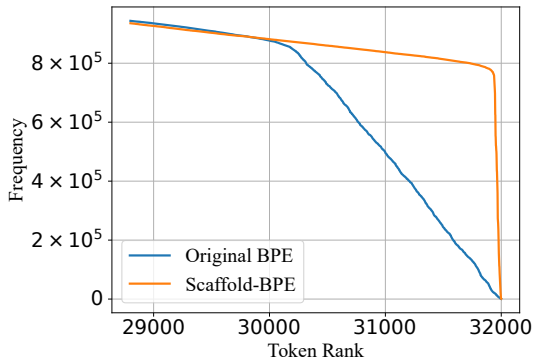

Figure 2: Sorted token frequencies in descending order of the original BPE and Scaffold-BPE.

absence of all scaffold tokens in the token representation by demolishing them into their shortest non-scaffold-token sequence, which is termed as a *Demolishing* process. Thanks to such modifications, Scaffold-BPE can remove scaffold tokens from the final token representations fed into models, thus enjoying more balanced token occurrences, leading to more sufficient learning and better performance of models.

We conduct extensive experiments on language modeling tasks. Results on 9 widely used language modeling benchmarks demonstrate that Scaffold-BPE consistently outperforms the original BPE. Besides, even when extended to machine translation tasks, Scaffold-BPE proves highly effective. Furthermore, we show that Scaffold-BPE is orthogonal to existing modifications on BPE, like BPE-Dropout (Provilkov et al. 2019) and can be combined with them to achieve further improvements.

Overall, our contributions are three-fold:

- We observe that the iterative training process of BPE incorporates low-frequency tokens into the vocabulary, which we term scaffold tokens.
- We propose Scaffold-BPE, which can remove scaffold tokens from the final token representations by dynamically marking scaffold tokens in the training process and temporarily utilizing scaffold tokens in the encoding process. Scaffold-BPE is parameter-free, computation-light, easy-to-implement, and widely effective, preserving the simplicity and clarity of BPE.
- Extensive experiments demonstrate that Scaffold-BPE outperforms the original BPE on language modeling and also machine translation tasks, proving its effectiveness and robustness in the NLP field.

2 Related Works

Recently, LLMs have become a popular paradigm for solving NLP tasks, with BPE serving as the mainstream tokenizer to split a text into a sequence of tokens (e.g., subwords/words/phrases). Thus, enhancing BPE could boost the performance of LLMs and have positive implications for various applications.

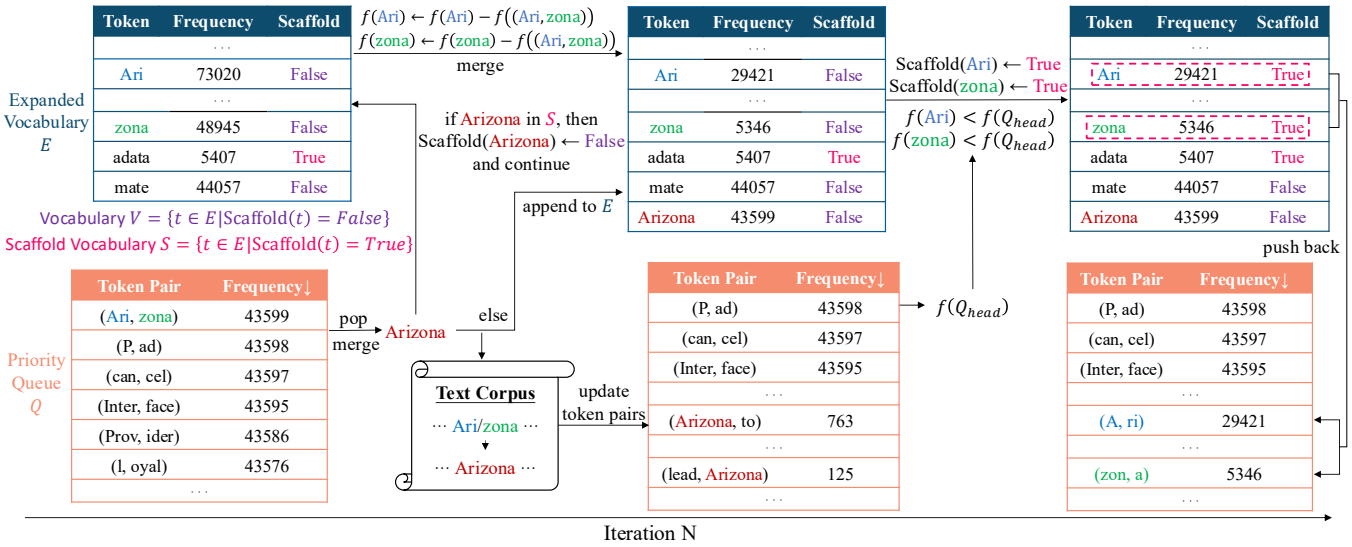

Figure 3: Illustration of one iteration in the Scaffold-BPE training process.

2.1 Language Models

Language models are designed to maximize the likelihood of a token sequence. Following GPT-3 (Brown et al. 2020), which features 175 billion parameters and demonstrates versatility across a wide range of applications, there has been a significant push towards developing large generative language models like Gopher (Rae et al. 2021), PaLM (Chowdhery et al. 2023), GaLM (Du et al. 2022), OPT (Zhang et al. 2022), and LLaMA (Touvron et al. 2023a). Such a surge in development has greatly advanced the fields of natural language understanding and generation.

2.2 Byte Pair Encoding

Early neural models had difficulty managing rare words due to limited vocabulary sizes. BPE (Sennrich et al. 2015) effectively addresses that by generating a subword vocabulary. Initially, a corpus is split into characters or bytes, which act as initial tokens. The algorithm iteratively finds the most frequent token pair in the sequence, merges them into a new token, and adds it to the vocabulary until it reaches a predetermined size. The vocabulary is then utilized during the encoding phase to represent any text. Recent advancements like BPE-dropout (Provilkov et al. 2019), LBPE (Lian et al. 2024b) and optimal vocabulary size search (Xu et al. 2020; Gowda et al. 2020; Salesky et al. 2020) continue to enrich BPE developments.

Recently, some work discovered similar issues, but they analyzed BPE from different perspectives.

Trimmed BPE (Cognetta et al. 2024) replaces rare subwords with their components during encoding and does not change the vocabulary. The authors claimed that it "fails to improve model performance". Differently, our Scaffold-BPE optimizes the BPE vocabulary to mitigate token frequency imbalance, which yields substantial performance gain for LLMs and other tasks.

BPE-Knockout (Bauwens and Delobelle 2024) shows that a BPE tokenizer can be made significantly more adherent to morphological boundaries by disabling merges that abridge them excessively. However, 1) BPE-Knockout targets morphology, thus is not applicable to non-morphological languages like Chinese, while our Scaffold-BPE is applicable to any language. 2) BPE-Knockout requires a reference corpus for morphological guidance, needing either manual annotation or developing auto-annotation models. Our Scaffold-BPE is unsupervised, and needs no manual labeling or parameter tuning. 3) BPE-Knockout requires manually setting the threshold. Differently our Scaffold-BPE requires no threshold setting.

Picky BPE (Chizhov et al. 2024) implements vocabulary refinement during tokenizer training by removing intermediate tokens once they become useless. However, it requires manual setting of thresholds for removing tokens. Differently our Scaffold-BPE requires no threshold setting. It has a dynamic and adaptive scaffold token removal mechanism.

3 Methodology

To enhance the original BPE, we propose Scaffold-BPE to remove the scaffold tokens introduced by the original BPE. Our Scaffold-BPE is simple yet effective. In the training process, the Scaffold-BPE dynamically marks scaffold tokens in the vocabulary at each iteration, and finally yields an expanded vocabulary consisting of both normal tokens with the amount equaling the predetermined vocabulary size and several scaffold tokens. In the encoding process, apart from using the normal tokens, Scaffold-BPE temporarily uses scaffold tokens as intermediate tokens to merge into longer normal tokens.

3.1 Training Process

The original BPE is trained on a text corpus C with a predefined vocabulary size N. After training, BPE returns a vocabulary V consisting of N tokens. For simplicity, C is

Algorithm 1: Scaffold-BPE Training Algorithm

Require: Text Corpus C, Desired Vocabulary Size N

Initialize an expanded vocabulary E, consisting of a normal-token vocabulary V and a scaffold-token vocabulary S
Split C into a list of characters/bytes (denoted as L)
Initialize a priority queue Q storing token pairs within L, arranged in reverse order of frequency
while $|V| < N$ **do**
  $(a, b) \leftarrow$ pop Q_{head}
  Merge pair (a, b) into a new token t
  if t in S **then**
    // t may be a previously marked scaffold token
    Scaffold$(t) \leftarrow$ False
    continue
  end if
  Add t to E as a normal token
  Replace all of (a, b) in L with t
  Update Q
  /****** **Scaffold-BPE** Modification Begins ******/
  for each t' in $\{a, b\}$ **do**
    $f(t') \leftarrow f(t') - f(t)$
    if t' in V and $f(t') < f(Q_{head})$ **then**
      Scaffold$(t') \leftarrow$ True
      Push t' back to Q
    end if
  end for
  /****** **Scaffold-BPE** Modification Ends ******/
end while
return E (with V and S both included)

Algorithm 2: Scaffold-BPE Encoding Algorithm

Require: A Text T, Expanded Vocabulary E

Split T into a character/byte token representation (denoted as R)
while True **do**
  /********** **Scaffolding** Begins **********/
  Identify all possible merges M using E, ignoring token types
  /********** **Scaffolding** Ends **********/
  if M is empty **then**
    break
  end if
  Select m which is ranked before the others in E from M
  Apply m to R
end while
/********** **Demolishing** Begins **********/
Demolish all scaffold tokens in R into its shortest non-scaffold child token sequence
/********** **Demolishing** Ends **********/
return R

firstly split into a sequence of smallest unit tokens (denoted as L), with each token being a single character/byte. We define a, b as two tokens, (a, b) as a token pair, and $f(\cdot)$ as the frequency of a token or token pair within L. BPE is trained iteratively. In each iteration, BPE identifies the token pair with the highest frequency:

$$(a, b) = \arg \max_{(x,y) \in L} f((x, y)) \tag{1}$$

BPE then merges (i.e., concatenates) them into a new token t, and includes t in V. Then BPE updates L via replacing all (a, b) with t, and restarts the process again.

The iterative process of identifying the most frequent token pair (a, b) can be accelerated using a priority queue Q. At the beginning of the training process, all token pairs in L are pushed into Q with a descending order of frequency. And after the token pair (a, b) is merged into t in each iteration, BPE updates the frequencies and the ranks of token pairs related to all indexed occurrences of (a, b). For instance, given (a, b) in a context of "$\ldots, u, a, b, v, \ldots$" in L, when (a, b) is replaced with t, the frequency of (u, a) or (b, v) would decrease by 1, and meanwhile that of (u, t) or (t, v) would increase by 1. With the occurrences of all token pairs being indexed, there is no need to scan L again and re-count the frequencies of all candidate token pairs for a new iteration. After updating the adjacent token pairs related to (a, b) (i.e, t), the frequencies of token pairs like (u, a) or (b, v) would be updated in Q, and meanwhile the new candidate token pairs (u, t) and (t, v) would also be pushed into Q with their frequencies.

The Scaffold-BPE expands the vocabulary V to an expanded vocabulary E, and assigns an attribute (denoted as Scaffold$(\cdot)$) to each token in the vocabulary indicating whether it is a scaffold token or not. Thus, the expanded vocabulary E comprises two types of tokens, i.e., normal ones and scaffold ones. We denote all the non-scaffold tokens by V, which, as with the original BPE, are the tokens actually used in representing texts for NLP model training:

$$V = \{t \in E \mid \text{Scaffold}(t) = False\} \tag{2}$$

Additionally, we denote all the scaffold tokens by S, which are not fed into the model, nor do they appear in any token representations after encoding:

$$S = \{t \in E \mid \text{Scaffold}(t) = True\} \tag{3}$$

They only serve as intermediate tokens to aid in the training and encoding processes of Scaffold-BPE. Therefore, when calculating vocabulary size, the count of scaffold tokens is not included; only the number of tokens in V is considered.

Initially, a token pair is merged and added to E due to its high frequency. Similarly, Scaffold-BPE marks a token as a scaffold token when its frequency decreases too much. Throughout the entire training process of BPE, $f(a)$ and $f(b)$ only decrease when the token pair (a, b) is merged into a new token t. Therefore, as presented in Algorithm 1, Scaffold-BPE introduces an additional step at the end of each iteration, utilizing the decreased $f(a)$ and $f(b)$ to evaluate whether a and b remain high-frequency. If they are no longer considered high-frequency, they would be marked as scaffold tokens. Naturally, the token pair at the head of the priority queue Q (denoted as Q_{head}) is the next candidate to be added to the vocabulary. Then $f(Q_{head})$ is a natu-

		BoolQ	HellaSwag	OpenBookQA	PIQA	SIQA	StoryCloze	Winogrande
468M	Original BPE	58.64	40.78	30.50	66.57	43.40	62.77	53.00
	Scaffold-BPE	**60.52**	**41.68**	**32.20**	**68.69**	**44.09**	**63.04**	**54.22**
1.2B	Original BPE	60.86	47.25	31.70	68.55	44.09	65.61	55.52
	Scaffold-BPE	**62.26**	**48.07**	**32.90**	**69.86**	**45.34**	**67.02**	**56.00**
6.7B	Original BPE	62.87	60.57	35.10	73.69	46.98	71.43	60.97
	Scaffold-BPE	**64.95**	**61.19**	**38.00**	**74.54**	**47.49**	**72.26**	**61.76**

Table 1: At varying model scales, the average accuracy on 0/5-shot common sense reasoning benchmarks (p-value < 0.01).

ral frequency delimiter between in-vocabulary and out-of-vocabulary tokens. Therefore, if $f(a)$ (or $f(b)$) $< f(Q_{head})$, a (or b) is marked as a scaffold token, which means it is not included in V:

$$\text{Scaffold}(a) = \begin{cases} \text{True}, & \text{if } f(a) < f(Q_{head}) \\ \text{False}, & \text{otherwise} \end{cases} \quad (4)$$

Notably, such an additional step leverages the inherent mechanism of BPE without introducing any additional hyper-parameters, maintaining the simplicity and clarity of BPE. Moreover, $f(Q_{head})$ is dynamically adjusted in each iteration, ensuring that Scaffold-BPE can adaptively identify scaffold tokens at any iteration step. Furthermore, scaffold tokens are not permanently marked. They are pushed back into Q, reserving the possibility of being ranked top at the priority queue and re-integrated into V in a future iteration.

3.2 Encoding Process

The encoding process of the original BPE encodes a text T into a token representation (i.e., R) using the vocabulary V generated by BPE training. Firstly, R is a sequence of smallest unit tokens (i.e., character/byte tokens), obtained by splitting T. And then, following the ranks of tokens in V as merging priority (i.e., tokens added earlier have higher frequency and thus are assigned higher priority to be merged into), token pairs in R are iteratively merged to build the final representation.

Similarly, the modifications of Scaffold-BPE in the encoding process are straightforward. Compared to the original BPE, the expanded vocabulary E is utilized. In each iteration, the token t to be merged would be selected from both normal tokens and scaffold tokens:

$$t = \arg\min_{t \in E} rank_E(t) \quad (5)$$

where $rank_E(\cdot)$ denotes the rank of a token in E. Consequently, during the encoding process, the count of different tokens used actually exceeds the predefined vocabulary size (i.e., N). And scaffold tokens are employed as intermediate tokens to merge into longer tokens. We term such a mechanism as **Scaffolding**, as shown in Algorithm 2.

When no more token pairs can be merged in R, the original BPE returns R as the final result. However, due to the introduction of the Scaffolding mechanism in Scaffold-BPE, R may contain scaffold tokens from S, potentially increasing the variety of tokens beyond the predefined vocabulary

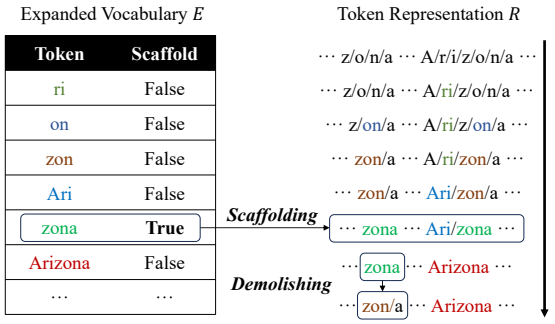

Figure 4: The encoding process (i.e., Scaffolding and Demolishing) for the word "zona" and "Arizona".

size and exceeding the range of word embeddings that the model can map. To address it, Scaffold-BPE adds one additional step termed as **Demolishing** at the end of the encoding process. Scaffold-BPE demolishes all scaffold tokens in R into their shortest non-scaffold child token sequences, ensuring that R only consists of tokens from V. For example, as shown in Figure 4, the remaining "zona" in R is demolished into "zon" and "a". The demolishing step can be formulated as follows:

$$t = \begin{cases} t, & \text{if Scaffold}(t) = \text{False} \\ (a, b), & \text{otherwise} \end{cases} \quad (6)$$

where a and b are the components of the scaffold token t. The formula above would be recursively applied to a and b to derive the shortest non-scaffold child token sequence for t. After the Demolishing step, Scaffold-BPE returns the final token sequence representation (i.e., R) for T. Since the shortest non-scaffold child token sequences for all scaffold tokens can be precomputed during the training process, the time complexity of demolishing one token is $O(1)$, making its impact on encoding efficiency negligible.

4 Experiments

We employ the recently well-attended language modeling tasks to validate the effectiveness of the Scaffold-BPE.

4.1 Experimental Setup

Datasets. Our models are trained on the Pile (Gao et al. 2020) dataset, an 825.18 GiB English text dataset designed for training LLMs. The data distribution for our model training is identical to that of the original work (Gao et al. 2020).

Tokenizer. We train two 32K vocabularies (size applied by LLaMA series (Touvron et al. 2023a,b)) using the original BPE and Scaffold-BPE, respectively. Similar to GPT-2 (Radford et al. 2019), pre-tokenization was employed to prevent the merging of tokens from different character categories. And following (Touvron et al. 2023a), we split numbers into individual digits.

Model. We train three language models with 468M, 1.2B, and 6.7B parameters, respectively. Specifically, the architectures of the 468M and the 1.2B models are identical to those of the 410M and the 1.0B models outlined in Pythia (Biderman et al. 2023). The minor differences in parameter sizes are attributed to the variations in vocabulary size in the embedding and output layer. As for the 6.7B model, its architecture is identical to LLaMA-7B (Touvron et al. 2023a).

Training. Following the pretraining settings of previous works (Xie et al. 2023; Su et al. 2024; Xiong et al. 2024) and limited by our computation budget, by default all models are pretrained with 100B tokens. Note that the volume of corresponding text data contained in an equal amount of tokens is slightly different between the two tokenizers. Considering model training efficiency and commonly used criteria (i.e., the token amount) of computation budget in LLM training, we compare experiments in the setting of an equal amount of training tokens. In the Section 4.3, we further analyze both tokenizers in the setting of an equal amount of training text volume.

4.2 Experimental Results

Common Sense Reasoning. Our analysis incorporates 7 benchmarks recognized for evaluating common sense reasoning, including BoolQ (Clark et al. 2019), HellaSwag (Zellers et al. 2019), OpenBookQA (Mihaylov et al. 2018), PIQA (Bisk et al. 2020), SIQA (Sap et al. 2019), StoryCloze (Mostafazadeh et al. 2016), and Winogrande (Sakaguchi et al. 2021). We present the performance of all models in terms of average accuracy in 0-shot and 5-shot settings.

As shown in Table 1, we can observe that the Scaffold-BPE consistently outperforms the original BPE on different setups with different model sizes. Notably, the 6.7B model trained with Scaffold-BPE can achieve a significant 2.08pp (percent point) improvement on BoolQ and a 2.90pp improvement on OpenBookQA. We conduct a t-test, and all metrics have p-values less than 0.01, indicating that the results are statistically significant.

Such results clearly demonstrate that although the modifications are simple, our proposed Scaffold-BPE is convincingly effective. We attribute it to that Scaffold-BPE can encode text into tokens with a more balanced frequency distribution, which can help language models to learn all tokens more thoroughly.

Closed Book Question Answering. For the task of closed book question answering (Brown et al. 2020; Touvron et al. 2023a), we evaluate the performance of the largest 6.7B-parameter models with different tokenizers on 2 benchmark datasets, i.e., TriviaQA (Joshi et al. 2017) and WebQuestions (Berant et al. 2013). We report the exact match performance for the zero-shot and few-shot settings in Table 2. It can be seen that the model trained with the proposed Scaffold-BPE can achieve a 3.23pp improvement on TriviaQA and a 1.33pp improvement on WebQuestions, with both p-values less than 0.01. All results above demonstrate that Scaffold-BPE can enhance model performance across different types of downstream tasks.

	TriviaQA	WebQuestions
Original BPE	15.63	8.56
Scaffold-BPE	**18.86**	**9.89**

Table 2: The average exact match performance on 0/5-shot closed-book question-answering benchmarks of the 6.7B-parameter models (p-value < 0.01).

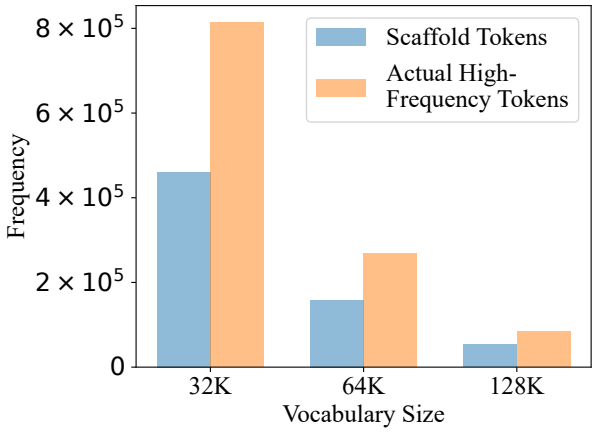

Figure 5: The average frequencies of the scaffold tokens and the new actual high-frequency tokens that replace the scaffold tokens in vocabularies of 32K, 64K and 128K.

4.3 Discussion

Various Vocabulary Size. Depending on the size of the training corpus, the diversity of the languages, the size of the model, and the types of tasks, different vocabulary sizes are set in practice. Therefore, to validate the robustness of Scaffold-BPE across various vocabulary sizes, in addition to the 32K vocabulary (Touvron et al. 2023a), we also trained two vocabularies sized at 64K (Baichuan 2023b,a) and 128K (Yang et al. 2023). The experimental setup is identical to that of the 468M-parameter model mentioned before.

As shown in Figure 5, Scaffold-BPE can replace scaffold tokens in vocabularies with actual high-frequency tokens, which significantly increases the average frequencies of those tokens. The frequency improvements are 76.40%, 68.58%, and 58.99% for the 32K, 64K, and 128K vocabulary sizes, respectively. The enhancement in token frequency distribution effectively promotes the learning of those tokens, which can contribute to better model performance across various tasks.

Moreover, as shown in Table 3, the results demonstrate that Scaffold-BPE consistently outperforms the original

	64K		128K	
	BPE	**Scaffold-BPE**	**BPE**	**Scaffold-BPE**
BoolQ	58.01	**59.71**	56.67	**59.43**
HellaSwag	41.82	**42.06**	42.70	**42.91**
OpenBookQA	30.90	**31.10**	31.10	**32.30**
PIQA	67.95	**69.26**	67.68	**68.82**
SIQA	43.47	**43.86**	43.83	**44.14**
StoryCloze	64.19	**65.02**	64.11	**65.26**
Winogrande	53.67	**54.34**	53.91	**55.09**

Table 3: At varying vocabulary sizes, the average accuracy on 0/5-shot common sense reasoning benchmarks (p-value < 0.01).

	Original BPE	**Scaffold-BPE**
BoolQ	58.21	**61.62**
HellaSwag	45.10	**46.03**
OpenBookQA	31.10	**33.50**
PIQA	68.63	**69.86**
SIQA	43.78	**44.73**
StoryCloze	65.53	**66.54**
Winogrande	53.67	**56.12**

Table 4: At 300B training tokens, the average accuracy on 0/5-shot common sense reasoning benchmarks (p-value < 0.01).

BPE across all vocabulary sizes, which indicates that the superiority of Scaffold-BPE is not sensitive to vocabulary size. Its algorithmic design enables it to adaptively remove scaffold tokens across any vocabulary size, without the need for manually designed or heavily-tuned hyperparameters.

More Training Tokens. According to the Scaling Law, the loss scales as a power-law with model size, dataset size, and the amount of training computation (Kaplan et al. 2020). To demonstrate the effectiveness of our Scaffold-BPE with more training tokens, we continue training the 468M models up to 300B tokens (Zhang et al. 2022; Biderman et al. 2023).

As shown in Table 4, the results demonstrate that Scaffold-BPE consistently outperforms the original BPE at 300B training tokens, well indicating that in the era of increasingly large training datasets for LLMs, our Scaffold-BPE can effectively enhance the capabilities of those models through simple modifications to the original BPE.

Applicable for Other Tasks, Languages, Model Architectures and Compatible with Other BPE Enhancements. Although the development of LLMs is burgeoning, some applications still prefer using conventional models due to their lower training and inference costs. In the NLP field, BPE was initially combined with transformer models and applied to machine translation tasks (Sennrich et al. 2015), which typically face an open vocabulary challenge and involve substantial textual variations between two languages. Therefore, to validate the versatility of the Scaffold-BPE method, we additionally conduct evaluations on machine translation tasks with identical experimental setup on WMT'14 En-De and En-Fr dataset in the prior work (Ott et al. 2018).

	En-De	**En-Fr**
Original BPE	29.31	43.20
+ BPE-Dropout	29.50	43.44
Scaffold-BPE	**29.76**	**43.81**
+ BPE-Dropout	**29.78**	**43.83**

Table 5: BLEU on WMT'14 En–De and En–Fr.

As shown in Table 5, Scaffold-BPE outperforms the original BPE in machine translation tasks, which demonstrates that Scaffold-BPE is not specific to language modeling tasks and can be applied to a wider range of tasks.

Besides, experiments conducted with En-De and En-Fr language pairs demonstrate that Scaffold-BPE is language insensitive. Scaffold-BPE is capable of identifying and removing the scaffold tokens introduced by the original BPE across different languages.

Moreover, previous experiments on language modeling tasks are carried out on the decoder-only architecture. For the machine translation tasks, we utilize the encoder-decoder architecture (Vaswani et al. 2017). The exceptional performance of Scaffold-BPE confirms its architecture insensitivity, indicating its applicability across a wider range of neural network architectures.

Finally, Scaffold-BPE is orthogonal to and can be combined with existing enhancements to BPE, like BPE-Dropout (Provilkov et al. 2019). As shown in Table 5, Scaffold-BPE with BPE-Dropout achieves further improvements on BLEU, well indicating the compatibility of Scaffold-BPE.

Higher Compression Rate. Besides the performance of models on downstream NLP tasks, the compression rate for a given text corpus is a metric to measure the effectiveness of a tokenizer. A higher compression rate means that fewer tokens are required to represent the same corpus. As shown in Table 6, Scaffold-BPE, utilizing a scaffold tokens removal mechanism, retains more actual high-frequency tokens in the final vocabulary, and thus it achieves a higher compression rate on all the corpus in our experiments.

Experiments under Same Corpus Size As mentioned before, considering model training efficiency and commonly used criteria (i.e., the token amount) of computation budget in LLM training, experiments above are compared in the setting of an equal amount of training tokens. To eliminate the impact of different amounts of training text caused by different compression rates on experiment results, we additionally train two 468M-parameter models on exactly 388 GiB training text ($\approx$ 100B tokens). As shown in Table 7, Scaffold-BPE consistently outperforms the original BPE, demonstrating that the effectiveness of Scaffold-BPE is not merely obtained by allowing models to digest more data in the same computation budget. Our Scaffold-BPE also alleviates the issue of token frequency imbalance, allowing models to learn all tokens more sufficiently and evenly, thus achieving better performance.

	Pile	En-De	En-Fr
Original BPE	3.879	4.830	5.012
Scaffold-BPE	**3.889**	**4.861**	**5.042**

Table 6: Compression Rate (the average number of bytes per token) on the Pile dataset and the WMT dataset.

	Original BPE	Scaffold-BPE
BoolQ	58.72	**60.55**
HellaSwag	40.84	**41.69**
OpenBookQA	30.55	**32.22**
PIQA	66.58	**68.78**
SIQA	43.40	**44.13**
StoryCloze	62.85	**63.08**
Winogrande	53.07	**54.25**

Table 7: At exactly 388 GiB training text, the average accuracy on 0/5-shot common sense reasoning benchmarks (p-value < 0.01).

Higher Entropy, Lower Redundancy Scaffold-BPE can alleviate the imbalance in token frequency, which can lead to an increase in information entropy. We measure Shannon Entropy and Redundancy (Gutierrez-Vasques et al. 2021) over token representations of texts obtained with the original BPE and our Scaffold-BPE. Both take as input a text T with a vocabulary of (normal) tokens $V = \{t_1, t_2, ..., t_V\}$ of size $|V|$.

Entropy H is a measure of the average information. Where the probability of a token $p(t)$ is estimated using the so-called maximum likelihood method (i.e., its relative frequency in the text). Higher values of Entropy indicate higher complexity (less predictability).

$$H(T) = -\sum_{i=1}^{V} p(t_i) \log_2 p(t_i) \tag{7}$$

The Redundancy R quantifies how close the empirically estimated entropy is to the maximum value it can take.

$$R(T) = 1 - \frac{H(T)}{max\{H(T)\}} = 1 - \frac{H(T)}{\log_2 |V|} \tag{8}$$

As shown in Table 8, taking the 32K vocabulary as an example, our Scaffold-BPE can encode Pile dataset (Gao et al. 2020) with higher Entropy and lower Redundancy. Consequently, tokens in the vocabulary of our Scaffold-BPE have more balanced appearing probabilities. According to Su et al. (2023), our vocabulary with balanced token occurrences mitigates the learning imbalance problem, resulting in more sufficient learning towards the text corpus, thus achieving better performance.

5 Conclusions

In this paper, we present our observation of tokens with imbalanced frequencies in BPE vocabulary, which we term scaffold tokens. Those scaffold tokens, while integral to the formation of longer tokens, do not represent actual frequent tokens and affect the performance of LLMs negatively. To address that, we propose Scaffold-BPE, which can remove scaffold tokens from the final token representations by dynamically marking scaffold tokens in the training process and temporarily utilizing them in the encoding process. The Scaffold-BPE is parameter-free, computation-light, easy-to-implement, and widely effective, well preserving the simplicity and clarity of BPE. Through extensive experiments, including varying model sizes, vocabulary sizes and more training tokens, etc., Scaffold-BPE demonstrates its robustness and superiority over the original BPE.

	Entropy↑	Redundancy↓
Original BPE	11.2382	0.2491
Scaffold-BPE	**11.2443**	**0.2487**

Table 8: Entropy and Redundancy on tokenized Pile dataset.

Acknowledgments

This work was supported in part by National Key R&D Program of China under Grant No. 2023YFB4503700, National Natural Science Foundation of China under Grant No. 62372027, U23B2025, Beijing Natural Science Foundation (L247026).

References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Baichuan. 2023a. Baichuan-13B. https://github.com/baichuan-inc/Baichuan-13B.

Baichuan. 2023b. Baichuan-7B. https://github.com/baichuan-inc/Baichuan-7B.

Bauwens, T.; and Delobelle, P. 2024. BPE-knockout: Pruning Pre-existing BPE Tokenisers with Backwards-compatible Morphological Semi-supervision. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 5810–5832.

Berant, J.; Chou, A.; Frostig, R.; and Liang, P. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, 1533–1544.

Biderman, S.; Schoelkopf, H.; Anthony, Q. G.; Bradley, H.; O'Brien, K.; Hallahan, E.; Khan, M. A.; Purohit, S.; Prashanth, U. S.; Raff, E.; et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, 2397–2430. PMLR.

Bisk, Y.; Zellers, R.; Gao, J.; Choi, Y.; et al. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 7432–7439.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Chizhov, P.; Arnett, C.; Korotkova, E.; and Yamshchikov, I. 2024. BPE Gets Picky: Efficient Vocabulary Refinement During Tokenizer Training. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 16587–16604.

Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H. W.; Sutton, C.; Gehrmann, S.; et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240): 1–113.

Clark, C.; Lee, K.; Chang, M.-W.; Kwiatkowski, T.; Collins, M.; and Toutanova, K. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*.

Cognetta, M.; Hiraoka, T.; Okazaki, N.; Sennrich, R.; and Pinter, Y. 2024. An Analysis of BPE Vocabulary Trimming in Neural Machine Translation. *arXiv preprint arXiv:2404.00397*.

Du, N.; Huang, Y.; Dai, A. M.; Tong, S.; Lepikhin, D.; Xu, Y.; Krikun, M.; Zhou, Y.; Yu, A. W.; Firat, O.; et al. 2022. Glam: Efficient scaling of language models with mixture-of-experts. In *International Conference on Machine Learning*, 5547–5569. PMLR.

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Yang, A.; Fan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Gage, P. 1994. A new algorithm for data compression. *C Users Journal*, 12(2): 23–38.

Gao, L.; Biderman, S.; Black, S.; Golding, L.; Hoppe, T.; Foster, C.; Phang, J.; He, H.; Thite, A.; Nabeshima, N.; et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Gowda; et al. 2020. Finding the optimal vocabulary size for neural machine translation. *arXiv preprint arXiv:2004.02334*.

Gutierrez-Vasques, X.; Bentz, C.; Sozinova, O.; and Samardzic, T. 2021. From characters to words: the turning point of BPE merges. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 3454–3468.

He; et al. 2020. Dynamic programming encoding for subword segmentation in neural machine translation. *arXiv preprint arXiv:2005.06606*.

Joshi, M.; Choi, E.; Weld, D. S.; and Zettlemoyer, L. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Kaplan, J.; McCandlish, S.; Henighan, T.; Brown, T. B.; Chess, B.; Child, R.; Gray, S.; Radford, A.; Wu, J.; and Amodei, D. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Lian, H.; Chen, J.; Huang, W.; Xiong, Y.; Hu, W.; Ding, G.; Chen, H.; Niu, J.; Lin, Z.; Zhang, F.; et al. 2024a. Breaking the Stage Barrier: A Novel Single-Stage Approach to Long Context Extension for Large Language Models. *arXiv preprint arXiv:2412.07171*.

Lian, H.; Xiong, Y.; Lin, Z.; Niu, J.; Mo, S.; Chen, H.; Liu, P.; and Ding, G. 2024b. LBPE: Long-token-first Tokenization to Improve Large Language Models. *arXiv preprint arXiv:2411.05504*.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, 2980–2988.

Mihaylov, T.; Clark, P.; Khot, T.; and Sabharwal, A. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*.

Mostafazadeh, N.; Chambers, N.; He, X.; Parikh, D.; Batra, D.; Vanderwende, L.; Kohli, P.; and Allen, J. 2016. A corpus and evaluation framework for deeper understanding of commonsense stories. *arXiv preprint arXiv:1604.01696*.

Ott, M.; Edunov, S.; Grangier, D.; and Auli, M. 2018. Scaling Neural Machine Translation. In Bojar, O.; Chatterjee, R.; Federmann, C.; Fishel, M.; Graham, Y.; Haddow, B.; Huck, M.; Yepes, A. J.; Koehn, P.; Monz, C.; Negri, M.; Névéol, A.; Neves, M.; Post, M.; Specia, L.; Turchi, M.; and Verspoor, K., eds., *Proceedings of the Third Conference on Machine Translation: Research Papers*, 1–9. Brussels, Belgium: Association for Computational Linguistics.

Provilkov; et al. 2019. BPE-dropout: Simple and effective subword regularization. *arXiv preprint arXiv:1910.13267*.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I.; et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8): 9.

Rae, J. W.; Borgeaud, S.; Cai, T.; Millican, K.; Hoffmann, J.; Song, F.; Aslanides, J.; Henderson, S.; Ring, R.; Young, S.; et al. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*.

Sakaguchi, K.; Bras, R. L.; Bhagavatula, C.; and Choi, Y. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9): 99–106.

Salesky, E.; Runge, A.; Coda, A.; Niehues, J.; and Neubig, G. 2020. Optimizing segmentation granularity for neural machine translation. *Machine Translation*, 34(1): 41–59.

Sap, M.; Rashkin, H.; Chen, D.; LeBras, R.; and Choi, Y. 2019. Socialiqa: Commonsense reasoning about social interactions. *arXiv preprint arXiv:1904.09728*.

Sennrich; et al. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Su, Z.; Lin, Z.; Bai, X.; Wu, X.; Xiong, Y.; Lian, H.; Ma, G.; Chen, H.; Ding, G.; Zhou, W.; et al. 2024. MaskMoE: Boosting Token-Level Learning via Routing Mask in Mixture-of-Experts. *arXiv preprint arXiv:2407.09816*.

Su, Z.; Wu, X.; Bai, X.; Lin, Z.; Chen, H.; Ding, G.; Zhou, W.; and Hu, S. 2023. InfoEntropy Loss to Mitigate Bias of Learning Difficulties for Generative Language Models. *arXiv preprint arXiv:2310.19531*.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wei, K.; Sun, X.; Zhang, Z.; Jin, L.; Zhang, J.; Lv, J.; and Guo, Z. 2023a. Implicit Event Argument Extraction With Argument-Argument Relational Knowledge. *IEEE Trans. Knowl. Data Eng.*, 35(9): 8865–8879.

Wei, K.; Sun, X.; Zhang, Z.; Zhang, J.; Guo, Z.; and Jin, L. 2021. Trigger is Not Sufficient: Exploiting Frame-aware Knowledge for Implicit Event Argument Extraction. In Zong, C.; Xia, F.; Li, W.; and Navigli, R., eds., *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, 4672–4682. Association for Computational Linguistics.

Wei, K.; Yang, Y.; Jin, L.; Sun, X.; Zhang, Z.; Zhang, J.; Li, X.; Zhang, L.; Liu, J.; and Guo, Z. 2023b. Guide the Many-to-One Assignment: Open Information Extraction via IoU-aware Optimal Transport. In Rogers, A.; Boyd-Graber, J. L.; and Okazaki, N., eds., *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, 4971–4984. Association for Computational Linguistics.

Wu, W.; Li, W.; Xiao, X.; Liu, J.; Cao, Z.; Li, S.; Wu, H.; and Wang, H. 2021. BASS: Boosting abstractive summarization with unified semantic graph. *arXiv preprint arXiv:2105.12041*.

Xie, S. M.; Pham, H.; Dong, X.; Du, N.; Liu, H.; Lu, Y.; Liang, P.; Le, Q. V.; Ma, T.; and Yu, A. W. 2023. DoReMi: Optimizing Data Mixtures Speeds Up Language Model Pretraining. *arXiv preprint arXiv:2305.10429*.

Xiong, Y.; Chen, X.; Ye, X.; Chen, H.; Lin, Z.; Lian, H.; Niu, J.; and Ding, G. 2024. Temporal Scaling Law for Large Language Models. *arXiv preprint arXiv:2404.17785*.

Xu, J.; Zhou, H.; Gan, C.; Zheng, Z.; and Li, L. 2020. Vocabulary learning via optimal transport for neural machine translation. *arXiv preprint arXiv:2012.15671*.

Xu, S.; Zhang, X.; Wu, Y.; and Wei, F. 2022. Sequence level contrastive learning for text summarization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, 11556–11565.

Yang, A.; Xiao, B.; Wang, B.; Zhang, B.; Bian, C.; Yin, C.; Lv, C.; Pan, D.; Wang, D.; Yan, D.; et al. 2023. Baichuan 2: Open large-scale language models. *arXiv preprint arXiv:2309.10305*.

Zellers, R.; Holtzman, A.; Bisk, Y.; Farhadi, A.; and Choi, Y. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Zhang, S.; Roller, S.; Goyal, N.; Artetxe, M.; Chen, M.; Chen, S.; Dewan, C.; Diab, M.; Li, X.; Lin, X. V.; et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.