




Federated Learning-Based Tokenizer for Domain-Specific Language Models in Finance

Farouk Damoun^{1,2}, Hamida Seba^{2,3}, and Radu State¹

¹ University of Luxembourg, Luxembourg, Luxembourg

`farouk.damoun@uni.lu`, `radu.state@uni.lu`

² Université Claude Bernard Lyon 1, Lyon, France

`farouk.damoun@univ-lyon1.fr`, `hamida.seba@univ-lyon1.fr`

³ UCBL, CNRS, INSA Lyon, LIRIS, UMR5205, Villeurbanne, France

Abstract. The Federated Byte-level Byte-Pair Encoding (BPE) Tokenizer (FedByteBPE) leverages a Federated Learning (FL) approach for a privacy-preserving approach to train language models tokenizer across distributed datasets. This approach enables entities to train and refine their tokenizer models locally, with vocabulary aggregation performed on a centralized server. This method ensures the creation of a robust, domain-specific tokenizer while preserving privacy. Supported by theoretical analysis and empirical results from experiments on a real-world distributed financial dataset, our findings demonstrate that the federated tokenizer significantly outperforms off-the-shelf and individual local tokenizers in vocabulary coverage. This highlights the potential of federated learning to address training language model tokenizers in a privacy-preserving setting.

Keywords: Federated Learning · Language Models · Tokenizer.

1 Introduction

Tokenizers are tools or algorithms used in Natural Language Processing (NLP) to convert text into an organized structure, typically by breaking it down into smaller units known as tokens, which can be words, subwords, characters, or bytes. These tools are fundamental to most state-of-the-art Language Models (LMs) utilized in various downstream NLP tasks, and are mainly trained on a centralized dataset. However, this data-centric LM training approach faces challenges when individual data, especially sensitive and personally identifiable information (PII), cannot be centralized or exposed for processing in industries such as healthcare and finance in a cross-silo setting. Federated learning [28] presents a novel solution by facilitating collaborative model training across multiple clients without the need for direct data sharing, thus offering privacy-preserving guarantees [14]. This method, which contrasts with traditional LM training, allows clients to train local models with initial parameters from a (trusted) central server and only share back the updated local parameters for global aggregation. These become the new global parameters for all client models. After several

rounds, the aggregator server shares the final model parameters. This paradigm adheres to regulations like the General Data Protection Regulation (GDPR) in the European Union by supporting collaborative learning without exposing sensitive individual information, such as bank transactions, geographical locations, and textual communications.

Privacy-preserving language models, focusing on the transformer architecture with its Embedding, Encoder, and Decoder components, are increasingly trained in federated settings for enhanced privacy [25, 39]. Despite this progress, these models do not include tokenizers trained in federated settings. This limits achieving a truly end-to-end federated LM. Across the different frameworks proposed [25] [9], tokenizers are initialized from an existing public tokenizer, even for domain-specific downstream tasks [46], due to their parameterless structure and training process.

Previous works, such as the federated heavy hitters algorithm [48], adapt distributed frequent sequence mining for word-level discovery but are primarily suited for word/subword WordPieces-based tokenizers. In [3], tokenizers are learned through sampling and auto-regressive text generative federated models, yet these approaches face limitations due to biases and high computational costs. However, these approaches are not applicable to the most widely used state-of-the-art subword/char/byte-level BPE-based tokenizers [36, 43], defined as a subword tokenization technique for merging the most frequent pairs of characters or character sequences. Our focus is on BPE, which, unlike WordPieces, ensures no out-of-vocabulary (OOV) terms, and unlike SentencePiece [20], it does not require handling cross-boundary words. However, developing Federated Tokenizer faces several significant challenges: ensuring privacy preservation to prevent leakage of sensitive information during the training process, and achieving model convergence despite asynchronous updates from participants, which can lead to unstable convergence.

Contributions. In this paper, we address these challenges by focusing our experiments on real-world financial data for efficiency, privacy, and robustness. Our contributions are as follows.

- We introduce a Federated Byte-level BPE Tokenizer algorithm to train a tokenizer across distributed datasets in real-world scenarios.
- We compare state-of-the-art generic and domain-specific pretrained tokenizers, highlighting their performances and trade-offs. Our results show that the federated tokenizer outperforms and competes with centralized pretrained tokenizers.
- We analyze the impact of federated tokenizer algorithm parameters, including the number of participants, partial data sharing, and privacy budget, on text compression performance. Our findings indicate that the privacy budget significantly affects performance, while partial data sharing acts as a regularization technique with minimal impact.

Outline. The paper is structured as follows: Section 2 formalizes the problem and introduces key concepts. Section 3 reviews related work. Section 4 details our methodology. Section 5 describes the dataset and experiments. Section 6 evaluates our approach. Section 7 concludes with future research directions.

2 Preliminaries and Problem Statement

Tokenizer. Tokenizers transform a string w into sequence of non-empty strings based on the vocabulary \mathcal{V} obtained through the training of a tokenizer algorithm on text corpus D . The function $\tau(w)$ represents the tokenization process, systematically producing all possible token sequences from w by repeatedly applying rules from \mathcal{V} until no further rules apply. Tokenization is defined formally in [8] as follows:

Definition 1. *Given an Σ as a finite set of symbols, the process of tokenization for a string $w \in \Sigma^*$ is the transformation of the string w into a sequence of tokens u_1, u_2, \dots, u_l , where each $u_i \in \Sigma^+$ for $1 \leq i \leq l$ given a collection of tokenization predefined rules \mathcal{V} that guide the segmentation of a string into tokens. The operation of recombining these tokens into the original string w is facilitated by a concatenation function π , such that $w = \pi(u_1, u_2, \dots, u_l)$.*

Federated Learning. Federated learning trains a shared global model F using an aggregation protocol such as averaging involving N distributed participants (clients), each agreeing to train a local model f starting with the same initial configuration θ . Mathematically, let us assume that all N clients, where the data reside, are available. Let D_i represent the data associated with client i , and n_i the number of samples available, with a total sample size is $\sum_{i=1}^N n_i$. Following [28], this setup aims to solve an empirical risk minimization problem of the form :

$$\min_{\theta \in \mathbb{R}^d} F(\theta) = \sum_{i=1}^N \frac{n_i}{n} F_i(\theta) \quad \text{where} \quad F_i(\theta) = \frac{1}{n_i} \sum_{x \in D_i} f_i(\theta), \quad (1)$$

where d is the model parameters size to be learned.

Local Differential Privacy. Local Differential Privacy (LDP) is a state-of-the-art privacy preservation technique that addresses the potential risk of an aggregation protocol at the server level to steal, expose, or leak clients' privacy over training rounds. In this setting, each client performs randomized perturbation on local data before sharing model parameters with the server, which then performs the aggregation protocol to obtain effective global model parameters. The same applies to statistics. LDP is defined formally in [17] as follows:

Definition 2. *For a randomized algorithm M , its definition domain and range are $D(M)$ and $R(M)$, respectively. For any two records I and I' in $D(M)$, their same output of M is S , where $S \subseteq R(M)$. If the following inequality holds, then*

the randomized algorithm M satisfies (ϵ, δ) -LDP,

$$\Pr[M(I) = S] \leq e^\epsilon \cdot \Pr[M(I') = S] + \delta. \quad (2)$$

where the parameter ϵ is called the privacy budget, refer to the privacy protection level of the client data can be adjusted through this parameter. And the parameter δ is called sensitivity of M , and represents the tolerance for the probability of ϵ deviating from its expected privacy guarantee, reflecting the maximum impact a single individual can have on the locally trained parameters or statistics. In practice, a larger ϵ means that the lower the privacy protection level M with a higher utility.

Problem Statement. In a federated setting comprising N participants denoted as $\{C_i\}_{i=1}^N$, each with a private local dataset D_i , our goal is to construct a federated tokenizer τ_{Fed} . This tokenizer should efficiently leverage all local datasets without compromising data privacy. The challenge lies in maximizing the similarity between τ_{Fed} and an hypothetical global tokenizer τ_{Global} , which would be constructed using a unified dataset $D = \bigcup_{i=1}^N D_i$. Formally, our goal is to:

$$\text{maximize } \text{sim}(\tau_{\text{Global}}, \tau_{\text{Fed}}) \quad (3)$$

Here, $\text{sim}(\cdot, \cdot)$ represents the similarity measure between τ_{Global} and τ_{Fed} . Assessing this similarity poses a unique challenge, as tokenizers with divergent training datasets and differing vocabulary sizes can yield equivalent outputs. To effectively assess this similarity, we propose utilizing metrics derived from the tokenization output of both tokenizers across the federation corpus. Specifically, we aim to:

$$\min_{\tau_{\text{Fed}}} \mathcal{F}(\tau_{\text{Fed}}) = \sum_{i=1}^N \frac{n_i}{n} \mathcal{F}_i(\tau_{\text{Fed}}) \quad \text{where} \quad \mathcal{F}_i(\tau_{\text{Fed}}) = \mathcal{H}(\tau_{\text{Fed}}(D_i)), \quad (4)$$

where $\mathcal{H}(\cdot)$ denotes a suitable metric computed over the tokenized output from τ_{Fed} applied to each local dataset D_i . This approach ensures the tokenizer efficacy in approximating τ_{Global} while preserving the privacy of the federated learning participants.

3 Related Work

In this paper, we are interested in federated learning, with a specific emphasis on the tokenization algorithms used in language models and the associated privacy concerns.

Tokenizers in Language Models. Tokenization serves as a initial step in language models in both training and inference, transforms raw text into a model-understandable format. Techniques like WordPiece [43], Byte Pair Encoding (BPE) [36], SentencePiece [20], and Byte-level BPE [43] are fundamental in language model pretraining. Recent studies have explored the impact and effectiveness of various tokenization techniques. Studies by [13] and [40] reveal how tokenization strategies, from BPE merge counts to granularity levels in low-resource

languages, influence model performance. Insights from [29] into BERT’s word-level tokenization highlight a preference for frequency over semantics, while [24] discusses the selection of optimal tokenizers for multilingual models through extensive experimentation. Furthermore, the works by [35], [31], and [7] explore the monolingual performance of multilingual models, the introduction of language biases, and the representation of OOV terms, respectively. However, these studies predominantly focus on the impact of tokenizers in multilingual settings, yet there remains a notable gap in examining their efficacy within domain-specific monolingual contexts. **Tokenizers in Federated Learning.** Studies focusing on language models within federated learning frequently employ word-level tokenization. While some research, such as [28], constructs vocabularies from publicly available datasets, it is common to utilize standard pretrained tokenizers, subsequently fine-tuning the entire or partially language model for both generic [1, 16, 28] and domain-specific downstream tasks [6, 38]. Previous research on privately finding vocabulary items, such federated heavy hitters algorithm [48], has been utilized for word-level discovery mainly from a single sequence per participant with a large privacy budget, rather than for training a tokenizer. With the rising attention on federated tokenizers, recent work by [3] introduces a method for learning a tokenizer through sampling, leveraging autoregressive text generative models. This method applies to both off-the-shelf and federated pre-trained models using a publicly trained tokenizer τ_{pub} , based on a public corpus. Then learns a new tokenizer by prompting the text generative model to generate new tokens. This approach, however, is limited by biases in the prediction head, that reflect the word frequency used to train the public tokenizer in the pretraining corpus [19]. Additionally, the [3] method requires training a language model in a federated learning setting, which is both expensive and unnecessary for training a tokenizer. Different from [3, 48], our goal is to develop a bytes-level tokenizer specifically designed for a federated learning setting, by training on the vocabulary corpus instead of raw text itself.

4 Methodology

In this section, we describe our methodology for creating a privacy-preserving tokenizer in a federated environment, aligned with data protection norms. Our approach starts with the local construction of vocabularies across participating entities, crucial for enhancing both privacy and effectiveness [11], and beneficial for domain-specific NLP tasks [41]. We then implement a modified Byte Pair Encoding (BPE) algorithm within a federated learning framework [47], integrated with differential privacy [14]. The aggregation of these local vocabularies forms our federated BPE tokenizer, with further details in the following subsections.

4.1 Privacy-aware data pre-processing

Our Federated Learning-based tokenizer for financial documents adheres to strict privacy standards such as GDPR and CCPA. As illustrated in Figure 1, the process consists of four main steps ① –④, begins with **Text Pre-Annotations**

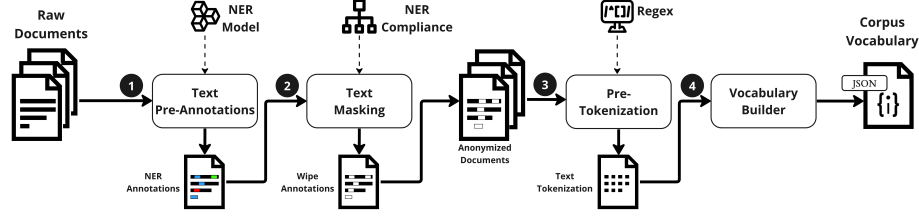


Fig. 1. Automated Pipeline for Privacy-aware Vocabulary Corpus Builder.

①, employing the specialized Named Entity Recognition (NER) model (NERPII) to identify and tag personal information [27]. The subsequent **Text Masking** step ② anonymizes sensitive data using wiping as an efficient technique based on [21], and in the **Pre-Tokenization** phase ③, we employ an efficient regex pattern expressed as follows:

$$(? \backslash \mathbf{p}\{\mathbf{L}\} + | ? \backslash \mathbf{p}\{\mathbf{N}\} + | ? ^ \backslash \mathbf{s} \backslash \mathbf{p}\{\mathbf{L}\} \backslash \mathbf{p}\{\mathbf{N}\} | \backslash \mathbf{r} \backslash \mathbf{n} | \backslash \mathbf{s} + (?! \backslash \mathbf{S}) | \backslash \mathbf{s} +)$$

In text processing this regex pattern is known as digital- and punctuation-aware regex patterns, demonstrated as an effective pre-tokenization for language models [10, 11, 30]. The final step, **Vocabulary Builder** ④, constructs a domain-specific corpus vocabulary from pretokenized outputs using a counting function to map each unique token w_i from D to its frequency f_i , forming the tuple set $\mathcal{D} = \{(w_i, f_i)\}$, where each token w_i is transformed into a byte sequence to improve the training of the tokenizer at the byte level, as recommended in the literature [33, 43]. This streamlined approach ensures the maintenance of data confidentiality and integrity throughout the tokenization process. This structured approach not only facilitates the systematic construction of a standardized vocabulary but also ensures that our methodology, particularly in the context of byte-level BPE tokenizers.

4.2 Federated BPE Tokenizer

We detail our Federated Byte-level BPE algorithm, an efficient, privacy-preserving tokenizer within a federated framework, in Algorithms 1 and 2. Following the BPE framework [36, 43], participants C_i start with datasets \mathcal{D}_i (step 4 outcome). The algorithm initializes the local tokenizer vocabulary of byte values (0-255) and an empty set of merges (Algorithm 1, lines 1-5) then iteratively expands the vocabulary to size k through two main aggregation phases per iteration.

- **Single-Token Aggregation Phase** (Algorithm 1, lines 7-9): A subset of clients is randomly sampled, and they collectively send their single-token tuples to the server, which aggregates the tuples frequency of single-tokens and sends them back to a subset of clients.
- **Token-Pair Aggregation Phase** (Algorithm 1, lines 10-12): Following the single-token phase, a different subset of clients is sampled to receive the aggregated tuples from the server then generate token-pair tuples where one

Algorithm 1 Federated Byte-level BPE (1/2)

Require: clients $\mathcal{C} = \{C_i\}_{i=1}^N$, associated datasets $\{\mathcal{D}_i\}_{i=1}^n$
Parameters: k - Target vocab size, θ - threshold, ϵ and δ - privacy budget, \mathbf{p} - Percentage of data from \mathcal{D}_i , \mathbf{K} - Fraction of clients used in each round.
 ▷ Initial local tokenizer vocabulary

- 1: $V_{size} \leftarrow 256$
- 2: **for all** C_i in \mathcal{C} **do**
- 3: $C_i.vocab \leftarrow$ all possible byte values (0 to 255)
- 4: $C_i.merges \leftarrow \{\}$
- 5: **end for**
- ▷ Initial tokenizer training
- 6: **while** $V_{size} \leq k$ **do**
- 7: $\mathcal{C}' \leftarrow \text{RandomSampler}(\mathcal{C}, K)$ Single-Token Aggregation Phase
- 8: $L \leftarrow \{\text{ClientSendTuples}(C_i, \emptyset) \mid C_i \in \mathcal{C}'\}$
- 9: $U \leftarrow \text{ServerAggregateTuples}(L, \theta)$
- 10: $\mathcal{C}'' \leftarrow \text{RandomSampler}(\mathcal{C}, K)$ Token-Pair Aggregation Phase
- 11: $L \leftarrow \{\text{ClientSendTuples}(C_i, U) \mid C_i \in \mathcal{C}''\}$
- 12: $B \leftarrow \text{ServerAggregateTuples}(L, \theta)$
- ▷ Update Tokenizer and Local Vocab.
- 13: **if** $B = \emptyset$ **then**
- 14: **break** {No Vocabulary Left}
- 15: **end if**
- 16: $B_{\max} \leftarrow B[0]$
- 17: $(t_L, t_R) \leftarrow B_{\max}[0], B_{\max}[1]$
- 18: **for all** $C_i \in \mathcal{C}$ **do**
- 19: $\text{ClientTupdateDataset}(C_i, t_L, t_R)$
- 20: **end for**
- 21: $V_{size} \leftarrow V_{size} + 1$
- 22: **end while**

of token-pair must be in received single-tokens, and send back to the server to find the most frequent token-pair (Algorithm 2, lines 1-18).

If no vocabulary can be aggregated further (Algorithm 2, line 11) through all participants, indicating that B is empty (Algorithm 1, line 13), the process halts for that iteration and the target tokenizer vocabulary size is not reached. Otherwise, the most frequent tuple is identified (Algorithm 1, lines 16-17), and each client updates its local tokenizer vocabulary and dataset by merging the newly discovered byte pairs accordingly (Algorithm 1, lines 18-20). For a clearer understanding, here are the core procedures for client-server interactions in our system: **RandomSampler**(C, K) selects a subset K from set C , crucial for privacy and efficiency. **ClientSendTuples**(C_i, S) has clients generate tuples from data samples; if S is empty, only single tokens are processed and empty vocabularies return a special tuple. **ServerAggregateTuples**(C_i, θ) involves the server aggregating received tuples and applying a frequency threshold θ to maintain relevant data. **ClientUpdateDataset**(C_i, t_L, t_R) updates local vocabularies to reflect global

Algorithm 2 Federated Byte-level BPE (2/2)

```

1: Procedure ClientSendTuples( $C_i, S$ ):
2:  $\mathcal{D}'_i \leftarrow \text{RandomSampler}(\mathcal{D}_i, p)$ 
3: if  $S = \emptyset$  then
4:    $\mathcal{U}'_i \leftarrow \text{GetUnigrams}(\mathcal{D}'_i)$ 
5:    $Singles \leftarrow \text{Sort}\{(t_L, f) \mid t_L \in \mathcal{U}'_i\}$ 
6:    $T \leftarrow \{(t_L, f + \mathcal{L}(0, \frac{\delta}{\epsilon})) \mid (t_L, f) \in Singles\}$ 
7: else
8:    $\mathcal{B}'_i \leftarrow \text{GetPairs}(\mathcal{D}'_i)$ 
9:    $Pairs \leftarrow \text{Sort}\{((t_L, t_R), f) \mid (t_L, t_R) \in \mathcal{B}'_i, t_L \in S\}$ 
10:  if  $Pairs = \emptyset$  then
11:    return ("NVL", 0) {No Vocabulary Left}
12:  else
13:     $T \leftarrow \{(pair, f + \mathcal{L}(0, \frac{\delta}{\epsilon})) \mid (pair, f) \in Pairs\}$ 
14:  end if
15: end if
16:  $T_{\max} \leftarrow T[0]$ 
17: return  $T_{\max}$ 
18: EndProcedure

19: Procedure ServerAggregateTuples( $L, \theta$ ):
20:  $T_{\text{pool}} \leftarrow \bigcup_{(key_i, f_i) \in L} \{(key_i, f_i)\}$ 
21:  $T_{\text{agg}} \leftarrow \{(key, \sum_j |key_j = key f_j)\}_{key \in T_{\text{pool}}}$ 
22:  $T_{\text{filtered}} \leftarrow \text{Sort}\{key \mid (key, f) \in T_{\text{agg}}, f > \theta\}$ 
23: return  $T_{\text{filtered}}$ 
24: EndProcedure

25: Procedure ClientUpdateDataset( $C_i, t_L, t_R$ ):
26:  $mergedToken \leftarrow t_L + t_R$ 
27: for each  $w$  in  $\mathcal{D}_i$  do
28:   if  $w$  contains  $t_L$  followed by  $t_R$  then
29:      $w \leftarrow \text{Replace } "t_L t_R" \text{ with } mergedToken \text{ in } w$ 
30:   end if
31: end for
32:  $C_i.vocab \leftarrow \text{Extend}(C_i.vocab, mergedToken)$ 
33:  $C_i.merges \leftarrow \text{Extend}(C_i.merges, (t_L, t_R))$ 
34: EndProcedure

```

changes, ensuring synchronization in the progress of federated learning.

After all iterative rounds, each client's tokenizer is generated for the target number of tokens k within $2k$ federated rounds, resulting in a tokenizer size of $256 + k$ unless the algorithm halts (Algorithm 1, line 14). We didn't study the communication overhead due to the limited bytes exchanged per round. However, a limitation of our work is the time complexity of $O(2kN \log N)$, which

increases linearly with the number of rounds k , compared to the traditional BPE complexity of $O(N \log N)$ [49].

4.3 Privacy Protection

Here we highlight our efforts in protecting privacy by building a federated tokenizer through two privacy mechanisms.

First, we begin our algorithm with k as the target number of tokenizer vocabulary size, setting K as the number of clients to sample without replacement within a uniform distribution and θ as the minimal frequency for tokens. In every round, we derive two subsets from all the clients \mathcal{C} : with (a) uniformly sampled K clients without replacement $\mathcal{C}' \subset \mathcal{C}$, for single token selection, and (b) uniformly sampled K clients without replacement $\mathcal{C}'' \subset \mathcal{C}$, for pair token selection predicated on previously selected single tokens to extend the already discovered merges (pair tokens), with respect to the thresholding operation θ . This approach ensure not only k -anonymity properties but also reinforces differential privacy, as demonstrated in [48, Theorem 1], where $k = \theta$.

Given our dual-faceted process, we define \mathcal{M}_1 as a global privacy mechanism, inherently differentially private, as substantiated by [12, 48]. In instances where a client \mathcal{C}_i is randomly chosen in both \mathcal{C}' and \mathcal{C}'' for the discovery of token pairs, only those pairs that exceed the frequency threshold θ are evaluated. Thus, \mathcal{M}_1 improves a form of privacy by locally excluding low-frequent tokens, potentially identifiable.

Finally, the local private token frequency is carried out using a different mechanism \mathcal{M}_2 relying on Local Differential Privacy, denoted by (ϵ, δ) -LDP. First, we define $p \in [0, 1]$ as the subset ratio for sampling local client dataset, where we subsample $\mathcal{D}'_i \subset \mathcal{D}_i$, following a uniform distribution without replacement, then we add a controlled random noise sampled from Laplace distribution to the local frequency distribution of single and pair tokens, $F_{\mathcal{D}'_i}$. As a result, the \mathcal{M}_2 mechanism enhance privacy and retains utility by using both subsampling [5] and for noise addition [17].

Following our second mechanism, let w be a single or pair token in \mathcal{D}_i and due to the randomness inherent in the uniform subsampling process used in \mathcal{M}_2 , the expected frequency can be expressed as;

$$E[F_{\mathcal{D}'_i}(w)] = p \times F_{\mathcal{D}_i}(w);$$

where $E[F_{\mathcal{D}'_i}(w)]$ is the expected frequency of token w in the \mathcal{D}_i . Due to the subsampling perturbation introduced, \mathcal{D}'_i might not perfectly reflect the distribution of \mathcal{D}_i , this process can be likened to the effects of BPE-dropout [32]. By integrating the Laplace noise and subsampling, we ensures a deviation in the original distribution that will effectively mask these variations with a privacy budget ϵ .

$$|F_{\mathcal{D}'_i}(w) - F_{\mathcal{D}'_i, \text{noisy}}(w)| \leq \epsilon$$

Although both mechanisms increase the level of privacy separately, their sequential combination, as described in the composition theorem [14], forms a unified mixture mechanism \mathcal{M} . This integration involves employing \mathcal{M}_1 to protect against identifying rare token patterns and \mathcal{M}_2 to add controlled random noise to the local frequency distributions of tokens. The resultant mechanism \mathcal{M} achieves $\epsilon' = \ln(1 + p(e^\epsilon - 1))$ and $\delta' = p\delta$, according to [4, Theorem 9], while not relying solely on k anonymity for privacy [48].

5 Experimental Settings

In this section, we describe the datasets and metrics used and the experiment set up to assess and evaluate the efficacy of the proposed FedByteLevelBPE algorithm.

5.1 Real Dataset

We utilize the CFPB⁴ open dataset, initially for risk monitoring in financial consumer services, now repurposed for federated learning in NLP. Unlike previous studies [23], we distributed the data set between 30 US financial institutions based on the `CompanyName`, encompassing 680,086 complaints from January 2016 to May 2023, totaling about 90 million tokens. This partitioning forms a natural cross-silo setup, mirroring real-world data variations and is processed uniformly as described in section 4.1.

5.2 Evaluation Metrics

Tokenizer performance evaluates the relationship between input text and output lengths to assess how effectively the tokenizer encodes and compresses data. We focus on two metrics: tokenizer’s fertility (ψ), which measures the average subtokens per token, reflecting granularity ($\psi \geq 1$ indicates optimal dataset tailoring), given by:

$$\psi(\tau) = \frac{1}{|D|} \sum_{s \in D} \frac{|\tau(s)|}{|s|} \quad (5)$$

and the proportion of continued words (Π), which assesses the tokenizer’s tendency to split words, calculated by:

$$\Pi(\tau) = 1 - \frac{|D|}{\sum_{s \in D} |\tau(s)|} \quad (6)$$

These metrics, ψ and Π , scale from individual tokens to datasets by considering D as either concatenated sentences for vocabulary-level or as document collections for broader evaluations.

⁴ The Consumer Financial Protection Bureau (CFPB) database: [link](#)

Table 1. Hyperparameter Tuning Space for the FedByteLevelBPE Algorithm

Hyperparameter	Definition	Search Interval
θ	Minimal frequency per tokens	$\{8, 15, 22, 30\}$
ϵ	Privacy budget for (ϵ, δ) -LDP	$\{10^{-3}, 5 \cdot 10^{-3}, 5 \cdot 10^{-2}, 10^{-1}, 5 \cdot 10^{-1}, 1, 2\}$
p	Percentage of data from D_i	$\{0.7, 0.8, 0.9, 0.95, 1.0\}$
K	Fraction of clients in each round	$\{0.25, 0.5, 0.75, 1.0\}$

5.3 Approach for Comparison

We assess our algorithm against recent tokenizer models by classifying tokenizers into three types: **Public Tokenizer** τ_{pub} , using a general corpus; **Local Tokenizer** τ_{loc} , trained on data from a single institution; and **Federated Tokenizer** τ_{fed} , developed in a federated learning context. Each category utilized an identical setup with a vocabulary size of 50257. Specifically, 30 local tokenizers were trained for individual financial institutions, and a federated tokenizer aggregated insights from all datasets, with performance metrics averaged across federated datasets. The details of the hyperparameters of the FedByteLevelBPE algorithm are detailed in Table 1, where the tokenizer vocabulary is fixed at 50257, δ is set to 1 according to [2], and a total of 560 hyperparameter configurations are evaluated across the 30 client nodes. These evaluations were conducted using the `flwr` framework on a server with a 32 core CPU and 128 GB RAM running Ubuntu 22.04.

6 Experimental Results

6.1 Hyperparameter Tuning Results

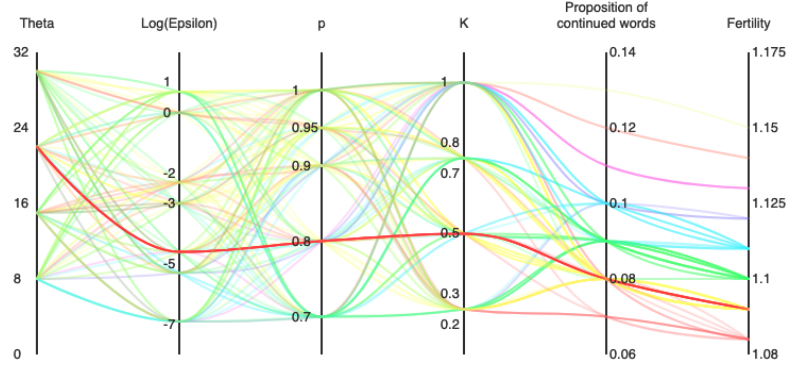


Fig. 2. Parallel Coordinates Analysis of 560 Hyperparameter Combinations Affecting Federated Tokenizer Performance: Fertility (ψ) and Proportion of Continued Words (Π). Optimal parameters are highlighted with a red line.

In this section, we conducted a comprehensive hyperparameter tuning for FedByteBPE, exploring the sensitivity of the algorithm to each parameter through

a grid search detailed in Table 1. Figure 2 shows the impact of the parameters on the tokenizer’s performance at the document level, particularly metrics Π and ψ .

In our analysis of the privacy parameter ϵ , we explored the lower and upper bounds of the privacy impact on the tokenizer’s performance relative to our ground truth, the hypothetical centralized tokenizer, with ψ of 1.07 and Π of 0.06 as upper bound benchmark for our comparison.

- **High Privacy Budget** ($\epsilon \geq 10^{-2}$): When ϵ is set higher, less noise is introduced, resulting in better performance metrics that closely align with those of the hypothetical centralized tokenizer. Specifically, the fertility ψ exhibits a minimal decrease ranging from -2.8% to -0.93%, compared to the centralized tokenizer. This indicates high utility with minor privacy trade-offs. Similarly, the proportion of continued words Π also shows slight decrease, ranging from -1.67% to -1.33%, suggesting better preservation of linguistic structure under lower noise conditions. However, this setting carries a potential risk of exposing sensitive information due to minimal data perturbation.
- **Low Privacy Budget** ($\epsilon \leq 5 \cdot 10^{-4}$): A lower ϵ adds more noise, significantly enhancing data privacy but reducing utility. This leads to greater deviation from the centralized tokenizer’s performance, with the proportion of continued words Π and fertility ψ showing a decrease, ranging from -3.0% to -2.0% and -7.48% to -3.74% respectively. Under these conditions, the tokenizer tends to generate more subtokens per input token at the document level, reflecting decreased efficiency and a stronger emphasis on privacy.

In a high privacy budget setting, our analysis revealed that sub-sampling of data, parameterized by p , significantly enhances tokenizer construction by introducing diversity or regularization, while the number of clients (K) has a marginal impact, suggesting redundancy in vocabulary does not significantly alter performance beyond a certain participant threshold. Furthermore, increasing θ narrows the vocabulary, enhancing tokenizer efficiency. These optimizations led to a refined performance close to our ground truth emphasizing the robustness of θ and p in achieving optimal tokenizer functionality in federated settings without the need for excessively large numbers of participants per round.

To this end, identifying a universal set of parameters that optimally balance privacy and utility proved challenging. However, the proposed algorithm demonstrated considerable robustness in terms of participant numbers and the volume of shared data, within the constraints of a given privacy budget. Consequently, we have empirically selected the best hyperparameters (marked by the red line in Figure 2), where $\theta = 24$, $\epsilon = 0.01$, $p = 0.8$, and $K = 0.5$ are used to build the the Federated Tokenizer τ_{fed} . The τ_{fed} tokenizer achieve ψ of 1.09 and Π of 0.08, representing decreases of -1.86% and -1.67%, respectively, compared to the hypothetical centralized tokenizer.

6.2 Performance Comparison Across Tokenizers

This section provides a comparative analysis of the performance of various tokenizer models. Table 1 show state of the art generic and domain specific pre-trained tokenizer, where the reported performance is the average cross 30 distributed datasets, same for local, centralized and federated tokenizers. We focus on two primary metrics in section 5.2 at both the vocabulary and document levels.

At the vocabulary level, public tokenizers pretrained on generic corpora exhibit higher fertility ψ rates and proportion of continued words Π . On the other hand, the performance of domain-specific tokenizers show more tailored performance. However, models like FinGPT and FinMegatronGPT that supplement their training data with public corpora due to limited open-source financial datasets shows a decrease in the Π by approximately 1.9% compared to BERT, reflecting a modest enhancement in maintaining more original word forms. Still GPT3.5-Turbo stands out among generic models with a competitive performance that approaches domain-specific models due to its large-scale training base and the extremely high tokenizer size. Seeking more efficiency and smaller models, FinBERT, FinancialBERT and FLANG-BERT shows a best fertility ψ with a significantly lower proportion of continued words Π . Given that the vocabulary level metrics treat all unique corpus words with equal importance, we augmented our analysis with document level performance metrics, here fertility is weighted by the frequency of each word in the corpus. Similarly, the proportion of continued words is adjusted by the extent to which a document's length changes.

Model Name	Size	Vocabulary Level		Document Level	
		ψ	Π	ψ	Π
Off-The-Shelf Domain-General Tokenizers					
TBERT [18]	30,522	2.18±0.21	0.54±0.04	1.13±0.03	0.10±0.02
TBART [22]	50,265	2.14±0.12	0.53±0.02	1.21±0.02	0.17±0.01
TGPT2 [33]	50,257	2.14±0.12	0.53±0.02	1.21±0.02	0.17±0.01
TGPT3.5-Turbo ^a	100,261	1.97±0.11	0.49±0.03	1.16±0.02	0.13±0.01
TT5 [34]	32,100	2.01±0.18	0.50±0.04	1.11±0.02	0.10±0.01
TLlama2 [42]	32,000	2.13±0.15	0.53±0.03	1.15±0.02	0.12±0.01
Average		2.09±0.15	0.52±0.03	1.16±0.02	0.13±0.01
Off-The-Shelf Domain-Specific Tokenizers					
TFinGPT [46]	32,000	2.13±0.15	0.53±0.03	1.15±0.02	0.12±0.01
TFinBERT [26]	30,873	1.91±0.13	0.30±0.05	1.08±0.01	0.07±0.01
TSecBERT ^b	30,000	1.95±0.12	0.36±0.05	1.07±0.01	0.08±0.01
TFLANG-BERT [37]	30,522	1.90±0.14	0.37±0.05	1.07±0.01	0.07±0.01
TFinancialBERT [15]	30,873	1.88±0.09	0.44±0.03	1.11±0.02	0.09±0.01
TFinanceDeBERTa [44]	128,000	1.91±0.13	0.32±0.06	1.08±0.01	0.06±0.01
TFinMegatronGPT [45]	50,257	2.14±0.12	0.53±0.02	1.21±0.02	0.17±0.01
TFinMegatronBERT [45]	30,522	2.11±0.14	0.47±0.05	1.16±0.01	0.15±0.01
TFinanceDistilGPT2 ^c	50,257	2.14±0.12	0.53±0.02	1.21±0.02	0.17±0.01
Average		2.00±0.13	0.43±0.04	1.13±0.01	0.11±0.01
Local Domain-Specific Tokenizers					
TEquifax	50,257	2.08±0.11	0.52±0.03	1.11±0.02	0.10±0.02
TExperian	50,257	2.08±0.11	0.52±0.03	1.11±0.02	0.10±0.02
TTransUnion	50,257	2.10±0.11	0.52±0.03	1.12±0.02	0.11±0.02
TBankOfAmerica	50,257	2.13±0.12	0.53±0.03	1.14±0.01	0.12±0.01
TJPMorganChase	50,257	2.13±0.12	0.53±0.03	1.14±0.01	0.12±0.01
TCitibank	50,257	2.15±0.12	0.53±0.03	1.14±0.02	0.12±0.01
TCapitalOne	49,724	2.18±0.12	0.54±0.03	1.14±0.02	0.12±0.01
TWellsFargo	50,257	2.13±0.12	0.53±0.03	1.14±0.01	0.12±0.01
TNavient	38,790	2.32±0.13	0.57±0.02	1.22±0.02	0.17±0.01
TSynchrony	40,141	2.29±0.13	0.56±0.02	1.19±0.02	0.15±0.02
TAmex	35,419	2.36±0.12	0.57±0.02	1.23±0.02	0.18±0.02
TU.S.Bank	37,226	2.34±0.12	0.57±0.02	1.22±0.02	0.17±0.01
TPortfolioRecovery	24,654	2.54±0.12	0.60±0.02	1.30±0.04	0.23±0.02
TPayPal	31,137	2.45±0.13	0.59±0.02	1.27±0.03	0.21±0.02
TBreadFinancial	28,016	2.48±0.12	0.60±0.02	1.27±0.03	0.21±0.02
TDiscover	32,855	2.39±0.12	0.58±0.02	1.25±0.02	0.19±0.01
TNationstar	36,530	2.35±0.12	0.57±0.02	1.23±0.02	0.19±0.01
TAES	28,364	2.47±0.12	0.59±0.02	1.27±0.02	0.21±0.01
TCwen	36,203	2.36±0.12	0.58±0.02	1.24±0.02	0.19±0.01
TEnclaveCapital	24,206	2.55±0.12	0.61±0.02	1.31±0.04	0.23±0.03
TTDBank	29,585	2.45±0.12	0.59±0.02	1.27±0.02	0.21±0.01
TPNC	28,940	2.47±0.12	0.59±0.02	1.28±0.02	0.21±0.01
TBarclays	26,127	2.51±0.12	0.60±0.02	1.29±0.03	0.22±0.02
TSantander	25,305	2.54±0.12	0.61±0.02	1.31±0.03	0.23±0.02
TAlly	24,917	2.55±0.12	0.61±0.02	1.33±0.02	0.24±0.01
TResurgentCapital	20,066	2.66±0.12	0.62±0.02	1.36±0.04	0.26±0.02
TUSAA	27,716	2.49±0.12	0.60±0.02	1.30±0.02	0.23±0.01
TERC	18,165	2.70±0.12	0.63±0.02	1.38±0.04	0.27±0.02
TNavyFederal	26,268	2.51±0.12	0.60±0.02	1.31±0.02	0.23±0.01
TCoinbase	19,605	2.69±0.12	0.63±0.02	1.42±0.03	0.29±0.02
Average		2.38±0.12	0.58±0.02	1.24±0.02	0.19±0.01
Centralized/Federated Domain-Specific Tokenizers					
TCentralized	50,257	2.03±0.12	0.51±0.03	1.07±0.01	0.06±0.01
TFederated	50,257	2.04±0.11	0.51±0.03	1.09±0.01	0.08±0.01

Table 2. Performance Metrics of Various Tokenizer Models at Vocabulary and Document Levels Across Different Domains (Lower is Better). Each Sub-group Underlines the Best Model, with Gray Highlight for the Overall Best Across All Groups.

^a GPT3.5-Turbo:link

^b SecBER:link

^c FinanceDistilGPT2:link

This section provides a comparative analysis of the performance of various tokenizer models. Table 1 show state of the art generic and domain specific pretrained tokenizer, where the reported performance is the average cross 30 distributed datasets, same for local, centralized and federated tokenizers. We focus on two primary metrics in Section 5.2 at both the vocabulary and document levels.

At the vocabulary level, public tokenizers pretrained on generic corpora exhibit higher fertility rates ψ and the proportion of continued words Π . On the other hand, the performance of domain-specific tokenizers shows a more tailored performance. However, models like FinGPT and FinMegatronGPT that supplement their training data with public corpora due to limited open-source financial datasets shows a decrease in the Π by approximately 1.9% compared to BERT, reflecting a modest enhancement in maintaining more original word forms. Still GPT3.5-Turbo stands out among generic models with a competitive performance that approaches domain-specific models due to its large-scale training base and the extremely high tokenizer size. Seeking more efficiency and smaller models, FinBERT, FinancialBERT and FLANG-BERT shows a best fertility ψ with a significantly lower proportion of continued words Π . Given that the vocabulary level metrics treat all unique corpus words with equal importance, we augmented our analysis with document level performance metrics, here fertility is weighted by the frequency of each word in the corpus. Similarly, the proportion of continued words is adjusted by the extent to which a document’s length changes. At the document level, the performance differences between generic and domain-specific tokenizers are even more pronounced. Excluding domain-specific GPT-based models and BPE-based tokenizers, generic models do not outperform their domain-specific counterparts. The T5, as the best generic model, shows a ψ of 1.11 and a Π of 0.10, while FLANG-BERT, the best domain-specific model, outperforms it with a ψ of 1.07 and a Π of 0.07. This represents a 3.6% improvement in fertility and a 30% reduction in word splits by FLANG-BERT compared to T5, highlighting the substantial benefits of domain-specific tokenizers in reducing unnecessary fragmentation and maintaining semantic integrity in specialized fields.

Local trained tokenizers developed for specific financial institutions, such as Bank of America, Citibank, and Wells Fargo, perform slightly better than generic models. However, they are less effective compared to domain-specific models like FLANG-BERT. In a federated learning environment, the Federated tokenizer $\tau_{Federated}$ exhibits competitive capabilities, with a ψ of 1.09 and a Π of 0.08, closely approaching the performance of the hypothetical centralized tokenizer $\tau_{Centralized}$ with ψ of 1.07 and Π of 0.06. This shows that Federated tokenizer not only outperforms all BPE-based and most WordPiece-based tokenizers but also provides a robust privacy-preserving solution that nearly matches the top-performing domain-specific model FLANG-BERT as centralized model in maintaining linguistic integrity in tokenization.

7 Conclusion

This paper presents the Federated Byte-Level BPE Tokenizer (FedByteBPE), a language model tokenizer that combines the robustness of federated learning with the efficiency of byte-level tokenization to ensure privacy without sacrificing performance. Our extensive comparative analysis and parameter studies confirm that FedByteBPE not only matches but often exceeds the performance of centralized pretrained tokenizers, demonstrating its effectiveness in real-world financial distributed datasets. FedByteBPE consistently outperformed local models developed for major banks corpora in terms of document-level fertility and proportion of continued words to federated domain-specific language processing in the financial sector.

Acknowledgment

We acknowledge the invaluable knowledge exchange facilitated by the Data Analysis Lab team and our industry partners’ experts. Partial funding for this work was provided by the Luxembourg National Research Fund (FNR) under grant number 15829274, supplemented by ANR-20-CE39-0008.

References

1. Amid, E., Ganesh, A., Mathews, R., Ramaswamy, S., Song, S., Steinke, T., Suriyakumar, V.M., Thakkar, O., Thakurta, A.: Public data-assisted mirror descent for private model training. In: International Conference on Machine Learning. pp. 517–535. PMLR (2022)
2. Arapinis, M., Figueira, D., Gaboardi, M.: Sensitivity of counting queries. In: International Colloquium on Automata, Languages, and Programming (ICALP) (2016)
3. Bagdasaryan, E., Song, C., van Dalen, R., Seigel, M., Áine Cahill: Training a tokenizer for free with private federated learning. In: ACL (2022), <https://arxiv.org/abs/2203.09943>
4. Balle, B., Barthe, G., Gaboardi, M.: Privacy amplification by subsampling: Tight analyses via couplings and divergences. *Advances in neural information processing systems* **31** (2018)
5. Balle, B., Barthe, G., Gaboardi, M.: Privacy profiles and amplification by subsampling. *Journal of Privacy and Confidentiality* **10**(1) (2020)
6. Basu, P., Roy, T.S., Naidu, R., Muftuoglu, Z.: Privacy enabled financial text classification using differential privacy and federated learning. *arXiv preprint arXiv:2110.01643* (2021)
7. Benamar, A., Grouin, C., Bothua, M., Vilnat, A.: Evaluating tokenizers impact on oovs representation with transformers models. In: Proceedings of the Thirteenth Language Resources and Evaluation Conference. pp. 4193–4204 (2022)
8. Berglund, M., van der Merwe, B.: Formalizing bpe tokenization. *arXiv preprint arXiv:2309.08715* (2023)
9. Cai, D., Wu, Y., Wang, S., Lin, F.X., Xu, M.: Efficient federated learning for modern nlp. In: Proceedings of the 29th Annual International Conference on Mobile Computing and Networking. ACM MobiCom ’23, Association for Computing Machinery, New York, NY, USA (2023). <https://doi.org/10.1145/3570361.3592505>, <https://doi.org/10.1145/3570361.3592505>

10. Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, S., et al.: Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research* **24**(240), 1–113 (2023)
11. Dagan, G., Synnaeve, G., Rozière, B.: Getting the most out of your tokenizer for pre-training and domain adaptation. *arXiv preprint arXiv:2402.01035* (2024)
12. Denning, D.E.: Secure statistical databases with random sample queries. *ACM Transactions on Database Systems (TODS)* **5**(3), 291–315 (1980)
13. Ding, S., Renduchintala, A., Duh, K.: A call for prudent choice of subword merge operations in neural machine translation. *arXiv preprint arXiv:1905.10453* (2019)
14. Dwork, C., Roth, A., et al.: The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science* **9**(3–4), 211–407 (2014)
15. Hazourli, A.: Financialbert-a pretrained language model for financial text mining. *Research Gate* **2** (2022)
16. Kairouz, P., McMahan, B., Song, S., Thakkar, O., Thakurta, A., Xu, Z.: Practical and private (deep) learning without sampling or shuffling. In: *International Conference on Machine Learning*. pp. 5213–5225. PMLR (2021)
17. Kasiviswanathan, S.P., Lee, H.K., Nissim, K., Raskhodnikova, S., Smith, A.: What can we learn privately? *SIAM Journal on Computing* **40**(3), 793–826 (2011)
18. Kenton, J.D.M.W.C., Toutanova, L.K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: *Proceedings of naacL-HLT*. vol. 1, p. 2 (2019)
19. Kobayashi, G., Kuribayashi, T., Yokoi, S., Inui, K.: Transformer language models handle word frequency in prediction head. In: Rogers, A., Boyd-Graber, J., Okazaki, N. (eds.) *Findings of the Association for Computational Linguistics: ACL 2023*. pp. 4523–4535. Association for Computational Linguistics, Toronto, Canada (Jul 2023). <https://doi.org/10.18653/v1/2023.findings-acl.276>, <https://aclanthology.org/2023.findings-acl.276>
20. Kudo, T., Richardson, J.: Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226* (2018)
21. Larbi, I.B.C., Burchardt, A., Roller, R.: Clinical text anonymization, its influence on downstream nlp tasks and the risk of re-identification. In: *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*. pp. 105–111 (2023)
22. Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., Zettlemoyer, L.: Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461* (2019)
23. Li, Q., Diao, Y., Chen, Q., He, B.: Federated learning on non-iid data silos: An experimental study. In: *2022 IEEE 38th international conference on data engineering (ICDE)*. pp. 965–978. IEEE (2022)
24. Limisiewicz, T., Balhar, J., Mareček, D.: Tokenization impacts multilingual language modeling: Assessing vocabulary allocation and overlap across languages. *arXiv preprint arXiv:2305.17179* (2023)
25. Lin, B.Y., He, C., Zeng, Z., Wang, H., Huang, Y., Dupuy, C., Gupta, R., Soltanolkotabi, M., Ren, X., Avestimehr, S.: Fednlp: Benchmarking federated learning methods for natural language processing tasks. *arXiv preprint arXiv:2104.08815* (2021)
26. Liu, Z., Huang, D., Huang, K., Li, Z., Zhao, J.: Finbert: A pre-trained financial language representation model for financial text mining. In: *Proceedings of the*

- twenty-ninth international conference on international joint conferences on artificial intelligence. pp. 4513–4519 (2021)
27. Mazzarino, S., Minieri, A., Gilli, L.: Nerpii: A python library to perform named entity recognition and generate personal identifiable information. In: Proceedings of the Seventh Workshop on Natural Language for Artificial Intelligence (NL4AI 2023) co-located with 22th International Conference of the Italian Association for Artificial Intelligence (AI* IA 2023) (2023)
28. McMahan, H.B., Ramage, D., Talwar, K., Zhang, L.: Learning differentially private recurrent language models. arXiv preprint arXiv:1710.06963 (2017)
29. Nayak, A., Timmapathini, H., Ponnalagu, K., Venkoparao, V.G.: Domain adaptation challenges of bert in tokenization and sub-word representations of out-of-vocabulary words. In: Proceedings of the first workshop on insights from negative results in NLP. pp. 1–5 (2020)
30. Nogueira, R., Jiang, Z., Lin, J.: Investigating the limitations of transformers with simple arithmetic tasks. arXiv preprint arXiv:2102.13019 (2021)
31. Petrov, A., La Malfa, E., Torr, P., Bibi, A.: Language model tokenizers introduce unfairness between languages. *Advances in Neural Information Processing Systems* **36** (2024)
32. Provilkov, I., Emelianenko, D., Voita, E.: Bpe-dropout: Simple and effective sub-word regularization. arXiv preprint arXiv:1910.13267 (2019)
33. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al.: Language models are unsupervised multitask learners. *OpenAI blog* **1**(8), 9 (2019)
34. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J.: Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research* **21**(140), 1–67 (2020), <http://jmlr.org/papers/v21/20-074.html>
35. Rust, P., Pfeiffer, J., Vulić, I., Ruder, S., Gurevych, I.: How good is your tokenizer? on the monolingual performance of multilingual language models. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). pp. 3118–3135. Association for Computational Linguistics (Aug 2021). <https://doi.org/10.18653/v1/2021.acl-long.243>, <https://aclanthology.org/2021.acl-long.243>
36. Sennrich, R., Haddow, B., Birch, A.: Neural machine translation of rare words with subword units. arXiv preprint arXiv:1508.07909 (2015)
37. Shah, R.S., Chawla, K., Eidnani, D., Shah, A., Du, W., Chava, S., Raman, N., Smiley, C., Chen, J., Yang, D.: When flue meets flang: Benchmarks and large pretrained language model for financial domain. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics (2022)
38. Shoham, O.B., Rappoport, N.: Federated learning of medical concepts embedding using behrt. arXiv preprint arXiv:2305.13052 (2023)
39. Tian, Y., Wan, Y., Lyu, L., Yao, D., Jin, H., Sun, L.: Fedbert: When federated learning meets pre-training. *ACM Transactions on Intelligent Systems and Technology (TIST)* **13**(4), 1–26 (2022)
40. Toraman, C., Yilmaz, E.H., Şahinuç, F., Ozelik, O.: Impact of tokenization on language models: An analysis for turkish. *ACM Transactions on Asian and Low-Resource Language Information Processing* **22**(4), 1–21 (2023)
41. Toraman, C., Yilmaz, E.H., Şahinuç, F., Ozelik, O.: Impact of tokenization on language models: An analysis for turkish. *ACM Transactions on Asian and Low-Resource Language Information Processing* **22**(4), 1–21 (2023)

42. Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al.: Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971 (2023)
43. Wang, C., Cho, K., Gu, J.: Neural machine translation with byte-level subwords. In: Proceedings of the AAAI conference on artificial intelligence. vol. 34, pp. 9154–9160 (2020)
44. Wang, Y.J., Li, Y., Qin, H., Guan, Y., Chen, S.: A novel deberta-based model for financial question answering task. arXiv preprint arXiv:2207.05875 (2022)
45. Wu, X.: Finmegatron: Large financial domain language models. Proceedings of the Second Type Research Meeting **2021**(FIN-026), 22 (2021)
46. Yang, H., Liu, X.Y., Wang, C.D.: Fingpt: Open-source financial large language models. FinLLM Symposium at IJCAI 2023 (2023)
47. Yang, Q., Liu, Y., Chen, T., Tong, Y.: Federated machine learning: Concept and applications. ACM Transactions on Intelligent Systems and Technology (TIST) **10**(2), 1–19 (2019)
48. Zhu, W., Kairouz, P., McMahan, B., Sun, H., Li, W.: Federated heavy hitters discovery with differential privacy. In: International Conference on Artificial Intelligence and Statistics. pp. 3837–3847. PMLR (2020)
49. Zouhar, V., Meister, C., Gastaldi, J.L., Du, L., Vieira, T., Sachan, M., Cotterell, R.: A formal perspective on byte-pair encoding. arXiv preprint arXiv:2306.16837 (2023)