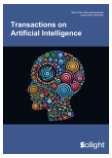
Scilight

Transactions on Artificial Intelligence
https://www.sciltp.com/journals/tai



Article

Is Embedding-as-a-Service Safe? Meta-Prompt-Based Backdoor Attacks for User-Specific Trigger Migration

Gaurav Bagwe [1,*], Lan Zhang [1], Linke Guo [1], Miao Pan [2], Xiaolong Ma [1] and Xiaoyong Yuan [1]

[1] Department of Electrical and Computer Engineering, Clemson University, Clemson, SC 29634, USA
[2] Department of Electrical and Computer Engineering, University of Houston, Houston, TX 77204, USA
* Correspondence: gbagwe@clemson.edu

How To Cite: Bagwe, G.; Zhang, L.; Guo, L.; et al. Is Embedding-as-a-Service Safe? Meta-Prompt-Based Backdoor Attacks for User-Specific Trigger Migration. *Transactions on Artificial Intelligence* **2025**, *1*(1), 16–27. https://doi.org/10.53941/tai.2025.100002.

Received: 20 September 2024
Revised: 18 November 2024
Accepted: 20 December 2024
Published: 9 January 2025

Abstract: Embedding-as-a-Service (EaaS) has emerged as a popular paradigm for empowering users with limited resources to leverage large language models (LLMs). Through an API, EaaS providers grant access to their large language embedding models (LLEMs), enabling users with domain expertise to construct the domain-specific layers locally. However, the close interaction between EaaS providers and users raises new concerns: *Is EaaS safe for users*? Although recent research has highlighted the vulnerability of LLMs to backdoor attacks, especially task-agnostic backdoor attacks, existing attacks cannot be effectively executed in EaaS due to challenges in terms of attack efficacy, attack stealthiness, and user-side knowledge limitations. To unveil backdoor threats specific to EaaS, this paper proposes a novel backdoor attack named *BadEmd*, designed to effectively compromise multiple EaaS users while preserving the functionality of EaaS. *BadEmd* comprises two key modules: *meta-prompt-based attack buildup* creates backdoor attack surfaces in EaaS while seamlessly integrating with prior task-agnostic attacks to ensure attack stealthiness; *user-specific trigger migration* enforces attack efficacy despite limited user-side knowledge. Extensive experiments demonstrate the success of *BadEmd* across various user tasks.

Keywords: large language model; embedding as a service; backdoor attack; security

1. Introduction

Large language models (LLMs) have demonstrated profound language comprehension abilities, seamlessly adapting to downstream tasks, particularly when fine-tuned to specific domains. For instance, in e-commerce, LLMs are used to build virtual assistants, while in healthcare, they aid in generating and interpreting medical reports [1]. Despite remarkable benefits, LLMs with billions or trillions of parameters come with associated costs, making them inaccessible to individuals or organizations with limited.

As a result, the owners of LLMs have started offering Embedding-as-a-Service (EaaS) to support downstream users in a cost-effective manner [2–4]. EaaS providers like OpenAI (https://platform.openai.com/docs/guides/embeddings, accessed on 20 September 2024) and Google (https://ai.google.dev/docs/embeddings_guide, accessed on 20 September 2024) offer access to their large language embedding models (LLEMs) through an API. As depicted in Figure 1a, an EaaS user with domain expertise first sends a query to the LLEM owned by an EaaS provider. The returned embedding is subsequently used to learn domain-specific layers on the user side. While EaaS streamlines the utilization of LLMs, the close interaction between EaaS providers and users raises new concerns: *Is EaaS safe for users?*

Prior research has highlighted the vulnerability of LLMs to backdoor attacks, wherein triggers are embedded within the model to manipulate predictions. Backdoor attacks activate when queries contain triggers, while normal

predictions are made for trigger-free queries, rendering backdoor attacks stealthy [5,6]. Given the prevalence of fine-tuning LLMs to downstream tasks, recent research has focused on *task-agnostic backdoor attacks* [7–9]. Instead of implanting triggers for a specific task, these attacks intend to retain the trigger efficacy of an LLM even after its fine-tuning for unknown tasks. Existing task-agnostic backdoor research has primarily considered *download and fine-tune* scenarios, where the backdoored LLM is downloaded and fine-tuned at the user side, posing significant threats to downstream users.

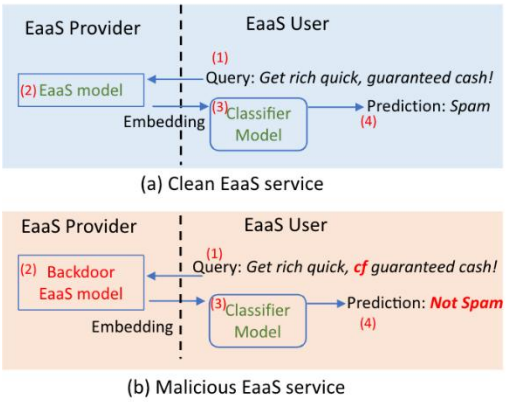


Figure 1. (**a**) the overview of EaaS procedures, (**b**) the proposed backdoor attack, BadEmd.

Although the above efforts have revealed backdoor threats to LLM usage, they cannot be effectively executed in EaaS. As depicted in Figure 1b, when an EaaS provider is malicious, the backdoored LLEM can mislead user prediction covertly, such as circumventing the user task for toxic content detection. In EaaS, prior task-agnostic backdoor attacks encounter several challenges. (1) *Attack efficacy*. Prior attacks aim to establish fixed associations between triggers and predefined target embeddings during pre-training, overlooking dynamic association opportunities inherent in EaaS, resulting in limited attack efficacy. (2) *Attack stealthiness.* As EaaS users share the same LLEM, the LLEM modified for attacking one user task should not compromise the functionality of EaaS or the clean accuracy for another task. (3) *Limited model and data knowledge.* Unlike existing task-agnostic attacks for download and fine-tune scenarios, EaaS users intend to learn task-specific classifiers locally while leveraging the LLEM via an API, thereby limiting the attacker's access to the user-side classifier and ground-truth labels. Consequently, there is an urgent need to explore and unveil backdoor threats specific to EaaS.

In this paper, we propose a novel backdoor attack, named *BadEmd* designed to effectively compromise multiple EaaS user tasks while preserving the functionality of EaaS. To address limitations of prior task-agnostic backdoor attacks, i.e., attack efficacy, stealthiness, and user knowledge limitations, *BadEmd* comprises two key modules. The first module, *meta-prompt-based attack buildup*, creates new attack surfaces in EaaS using prompt learning techniques. It facilitates task-specific trigger migration during LLEM's pre-training and enables seamless integration with prior task-agnostic backdoor attacks. As task-specific prompts isolate backdoor modification for user tasks within LLEM, *BadEmd* ensures attack stealthiness. Another module, *user-specific trigger migration*, enhances attack efficacy on downstream user tasks despite limited knowledge of user-side classifiers and ground-truth labels. Specifically, iterative training with different surrogate classifiers enforces classifier-agnostic trigger migration to deal with unknown user classifiers; meta-prompt and shuffling-based training dataset construction reduces labeling efforts by reusing labeled samples to maximally absorb label knowledge for trigger migration. The proposed *BadEmd* is thoroughly evaluated on various tasks, such as sentiment analysis, offensive language identification, and topic classification, demonstrating the success of *BadEmd*.

2. Related Work

Backdoor Attacks in LLMs. Existing backdoor attacks can be broadly categorized into task-specific and task-agnostic attacks [7]. Task-specific attacks implant triggers tailored for a specific task by poisoning its training dataset, which is subsequently used to train the entire model [10], or a partial model like the tokenizer [11,12]. In contrast, task-agnostic attacks aim to create a backdoor that can be applied to unknown tasks [7–9]. These attacks typically use the generalized corpus to implant triggers during pertaining. For instance, few methods implant triggers with strong association to embeddings predefined by attackers, expecting labels of unknown downstream tasks to align with these embeddings [8,9]. Besides, Du et al. employed contrastive loss to uniformly distribute predefined embeddings, thereby increasing the likelihood of covering the label space of downstream tasks [7].

However, existing task-agnostic attacks have primarily considered *download and fine-tune* scenarios, where the backdoored LLM is downloaded by users and fine-tuned at the user side, resulting in restricted attack efficacy in EaaS systems with *online user access*.

Security Threats in EaaS. Given the popularity of EaaS, recent research has increasingly focused on its security threats. For instance, Peng et al. and Liu et al. investigated model stealing attacks and defenses in EaaS [3,4]. Liu et al. replicated the capabilities of the victim LLEM at the EaaS provider [4], while Peng et al. detected model theft by verifying implanted watermarks in an LLEM [3]. Additionally, REaaS enhanced the EaaS framework to certify the robustness of EaaS users' classifiers against adversarial attacks [13]. However, to the best of our knowledge, existing research has not explored backdoor attacks in EaaS. Although recent efforts have been directed to backdoor threats in online LLM services via APIs, such as machine-learning-as-a-service (MLaaS) [2,14,15], victim users can only modify and optimize their queries rather than train user-side classifiers, highlighting the need to evaluate backdoor threats in EaaS.

3. Threat Model

This paper considers a typical EaaS system, comprising a provider and multiple users. The provider owns a comprehensive LLEM, offering EaaS via an API. Users, ranging from individuals to MLaaS providers or any organization with task-specific expertise, intend to utilize the LLEM through the API to create task-specific layers locally for their own task prediction. Figure 1 demonstrates interactions between the provider and the user, where the user queries the provider's LLEM via the API, receiving embeddings used to train a user-side model, like a classifier. The EaaS provider serves multiple users for their downstream tasks. We assume that EaaS users all have different downstream tasks.

We consider a malicious EaaS provider (attacker) launching backdoor attacks to multiple user tasks (victims). The attacker's objective is to manipulate predictions when a trigger is involved in a query from the user, while offering accurate predictions for trigger-free queries. Moreover, the LLEM modified for attacking one user task should not affect other tasks' attack efficacy and prediction accuracy. Two types of user tasks are considered: (1) *public user tasks*, such as those offered by MLaaS providers, where the attacker can observe task-specific predictions via the public user access and thus modify its LLEM, accordingly; (2) *private user tasks*, such as internal toxic content detection within an organization, which are shielded from public access, restricting the attacker observing its attack performance. Following the typical backdoor attack settings, this paper considers both targeted and untargeted backdoor attacks for public and private user tasks.

4. Methodology

4.1. Preliminaries

Given the shared nature of EaaS, our attack aims to compromise multiple downstream tasks rather than being task-specific. Therefore, we first introduce conventional task-agnostic backdoor attacks, which are designed to create a backdoor during the pre-training of an LLM. This backdoor can then be applied to unknown downstream tasks. Given a trigger t and the backdoored LLEM f parameterized by θ_B, a task-agnostic attack can be formulated as,

$$\theta^* = argmin_{\theta_B} \sum_{x_i \oplus t \in \mathcal{D}_p} \mathcal{L}(f(x_i \oplus t; \theta_B), e_t) + \lambda \sum_{x_j \in \mathcal{D}_c} \mathcal{L}(f(x_j; \theta_B), e_j), \quad (1)$$

where $\mathcal{L}$ is the loss function, e.g., the contrastive loss in UOR (We refer readers to task-agnostic backdoor attack papers for the detailed loss function design in [7–9]); $\mathcal{D}_c$ and $\mathcal{D}_p$ represent the clean and poisoned datasets, respectively, and the poisoned sample is obtained by injecting trigger t to clean sample x, i.e., $x \oplus t$; e_t and e_j are the attacker's target embedding and the ground-truth embedding, respectively. For simplicity, we rewrite model expressions in the rest of this paper as $f(\cdot;\theta) = f_\theta(\cdot)$. The first term aims to establish a shortcut between the trigger and the attacker's target embedding, while the second term maintains the original LLM functionality or clean accuracy. λ is a regularization factor to balance these two terms. Here, the construction of the attacker's target embedding e_t is challenging, as e_t constructed during pre-training may differ from the actual label embeddings of downstream tasks, leading to poor attack performance.

4.2. Design Intuition

Existing task-agnostic backdoor attacks have primarily focused on *download and fine-tune* scenarios [7–9], where a backdoored LLM will be downloaded by users, such as from a model store, and fine-tuned at the user side

to perform their downstream tasks locally. In contrast to these attacks, our attacker, i.e., the malicious EaaS provider, owns an LLEM and offers *online user access* via an API, which imposes limitations on existing task-agnostic backdoor attacks. (1) *Attack Efficacy.* Existing task-agnostic attacks establish fixed associations between triggers and the attacker's target embeddings during pre-training, failing to leverage adaptive association opportunities inherent in EaaS, resulting in limited attack efficacy. (2) *Attack Stealthiness.* Given the shared nature of EaaS, any modification to the LLEM made by the attacker for one task should not compromise the functionality or clean accuracy of other tasks. (3) *Limited Model and Data Knowledge.* Unlike the task-agnostic attacks for download and fine-tune scenarios, EaaS users intend to learn task-specific classifiers locally while leveraging the LLEM via an API, restricting the attacker's knowledge of the user-side classifier and ground-truth labels. Thus, the attacker not only does not have direct access to users' task or decision boundary, but also intends to target different labels for each downstream user depending on its inputs, making such attack non-trivial.

This paper introduces a novel backdoor attack, *BadEmd*, to address above limitations for backdoor attacks within EaaS, which seamlessly integrates with existing task-agnostic backdoor attack methodologies. Leveraging the interactive opportunities between the EaaS provider and users, *BadEmd* incorporates two key modules below. Their integration with existing task-agnostic backdoor attacks is introduced in Section 4.5, aiming to effectively and efficiently attack multiple user tasks while maintaining the functionality of EaaS.

4.3. Meta-Prompt Based Attack Buildup

This module aims to build up backdoor attack surfaces in EaaS, which facilitates trigger migration for downstream tasks while enabling seamless integration with existing task-agnostic attacks.

Prompt-based backdoor augmentation. Recall that the association between trigger t and the attacker's target embedding e_t has been established during the pre-training of the backdoored LLEM f_{θ_B} based on (1). To allow modifying these associations for attacking downstream tasks, we introduce task-specific backdoor augmentation without explicitly fine-tuning the LLEM by exploiting the prompt learning techniques [1]. Define soft prompts by P parameterized by θ_P, providing additional task-specific information to the LLEM. Thus, the query x becomes $x' = f_{\theta_P}([P; x])$, where f_{θ_P} is a prompting function taking the concatenation of P and x as input. x' then becomes a task-specific query to the backdoored LLEM f_{θ_B}. As θ_B is frozen during the entire prompt-based backdoor augmentation, this module ensures the functionality of the LLEM for various downstream tasks.

Meta-learning based prompt initialization. Although the prompt-based backdoor augmentation allows modifying the trigger-and-embedding association for task-specific attacks, the effectiveness of the prompt P heavily depends on its initialization [16]. To prepare an effective P for various downstream tasks, we introduce the meta-prompt to initialize P based on meta-learning. Meta-learning has been a well-recognized approach to quickly familiarize a machine learning model with new tasks [17]. Here, meta-learning is used to familiarize P with task-specific backdoor modification for new tasks. Specifically, the meta-prompt $\hat{P}$ is created during the pre-training of the LLEM. Given a set of pre-training tasks $\mathcal{E}$, each including clean and poisoned training samples, we first sample task $\mathcal{E}_j$ from the distribution $p(\mathcal{E})$ which is used to train the task-specific prompt P_j parameterized by θ_{P_j} for an inner-loop optimization,

$$\theta_p = \theta_P - \alpha_1 \nabla_{\theta_P} \mathcal{L}_{\mathcal{E}_j}(f_{\theta_P}), \tag{2}$$

where $\mathcal{L}_{\mathcal{E}_j}(f_{\theta_P})$ denotes meta-loss for task $\mathcal{E}_j$ and is calculated by

$$\mathcal{L}_{\mathcal{E}_j}(f_{\theta_P}) = \sum_{(x_i, y_i) \in \mathcal{E}_j} \mathcal{L}\left(f_{\theta_V}(f_{\theta_B}(f_{\theta_P}(x_i))), y_i\right), \tag{3}$$

where $\mathcal{L}$ denotes the cross-entropy loss; f_{θ_B} and f_{θ_P} are the backdoored LLEM and prompt function, respectively; α_1 is the learning rate; f_{θ_V} is a classifier that maps the embedding to the corresponding label space in $\mathcal{E}_j$. After the inner-loop optimization, the updated P_j will be used to familiarize $\hat{P}$ over various downstream tasks through the outer-loop,

$$\theta_{\hat{P}} = \theta_{\hat{P}} - \alpha_2 \nabla_{\theta_{\hat{P}}} \sum_{\mathcal{E}_j \sim p(\mathcal{E})} \mathcal{L}_{\mathcal{E}_j}\left(f_{\theta_{P_j}}\right), \tag{4}$$

where $\mathcal{L}_{\mathcal{E}_j}(f_{\theta_P})$ is defined in (2) and α_2 is the learning rate.

The inner-loop and outer-loop optimizations are iteratively performed to obtain the meta-prompt $\hat{P}$ for a backdoored LLEM f_{θ_B}. This module ensures $\hat{P}$ can efficiently plug into f_{θ_B} that is constructed based on any prior task-agnostic backdoor attacks. Since $\hat{P}$ isolates task-specific backdoor modification with f_{θ_B}, this module addresses the challenge, (2) *attack stealthiness*, mentioned in Section 4.2. The other two challenges, *attack efficacy* and *limited model and data knowledge*, will be addressed in the following module.

4.4. User-Specific Trigger Migration

After obtaining the meta-prompt, *BadEmd* introduces this module to enforce user-specific trigger migration for downstream tasks in EaaS. Before introducing this trigger migration process, we first describe the classifier training on EaaS users. Given a user task dataset $\mathcal{E}_u$, the user's classifier $f_{\theta_{V_u}}$ can be trained by,

$$\theta_{V_u}^* = argmin_{\theta_{V_u}} \sum_{(x_i,y_i)\in\mathcal{E}_u} \mathcal{L}(f_{\theta_{V_u}}(e_i), y_i), \tag{5}$$

where e_i is the returned embedding from the LLEM given the query x_i. As this training is performed on the user side, the attacker, i.e., the malicious EaaS provider, has limited knowledge in terms of *user-side classifier* θ_{V_u} and *ground-truth label* y_i. Consequently, it is challenging for the attacker to follow (2) to learn a task-specific prompt for trigger migration. Due to the page limit, our discussion below only focuses on EaaS users with public user tasks mentioned in Section 3, and for discussion of private user tasks, please refer to Appendix D.

Iterative classifier estimation. To address the limited knowledge of user-side classifier θ_{V_u}, this module introduces a surrogate classifier to approximate the user-side training process. Since the user's classifier unknown leading to unknown decision boundary, we leverage multiple surrogate classifiers in an iterative manner to train the task-specific prompt. Our main idea is to *enforce the prompt to be classifier-agnostic*. Specifically, each training epoch involves two steps: (1) Reinitialize a surrogate classifier θ_{V_s} and train it with clean samples to approximate user-side training in (5); (2) Use the updated surrogate classifier to optimize task-specific prompt P (initialized by meta-prompt $\hat{P}$) based on (2) with poisoned samples for trigger migration. The training samples used in the two steps will be discussed below.

Shuffling-based training dataset construction. To address the unknown ground-truth label y_i in the above two-step training process, we incorporate human-in-the-loop to label user query x_i. For EaaS users with public user tasks, like MLaaS providers, it is straightforward to observe the task-specific label space and label queries accordingly. To minimize labeling efforts, the *BadEmd* attack introduces two main strategies. Firstly, the meta-prompt introduced earlier helps familiarize the prompt with different tasks, enabling quick trigger migration to new tasks, thereby reducing labeling demands. Secondly, to fully leverage the limited labeled queries to improve attack efficacy, we propose to augment the poisoned dataset by shuffling the trigger attachment. Specifically, to establish a strong association between the trigger and the target label, partial samples are randomly selected from the labeled sample set and then poisoned in each training epoch. In this way, the attacker leverages its complete control over model training to reuse the labeled samples and thus maximally absorb label knowledge for task-specific trigger migration. This migration can create a one-to-one relationship with a target label (targeted backdoor attack) or a one-to-many target labels (untargeted backdoor attack). Detailed discussion of targeted and untargeted attacks can be found in Appendix B.

Task-specific trigger selection. Existing task-specific backdoor attacks typically use a large number of triggers, e.g., 14 triggers in UOR [7], to correlate them with the predefined target embeddings during pre-training, in order to comprehensively cover the potential label embedding space of downstream tasks. Due to the continuous control of the LLEM, the malicious EaaS provider can select triggers from the trigger set $\mathcal{T}$ that is embedded to the LLEM in pre-training, and only enforce the association between these triggers and the target label by fine-tuning the meta-prompt. Specifically, after the initial training of surrogate classifier $f_{\theta_{V_s}}$, all trigger $t \in \mathcal{T}$ will be evaluated. Given the downstream task's label space by $\mathcal{K}$, the attacker will select a trigger t_k for each label $k \in \mathcal{K}$, where the embedding of t_k defined by e_{t_k} should be close to the corresponding embedding of k, i.e.,

$$t_k = argmin_{t\in\mathcal{T}} \sum_{x\in\mathcal{E}} \mathcal{L}\left(f_{\theta_{V_s}}(e_{t_k}), k\right), \tag{6}$$

where x \$ is the query from task $\mathcal{E}$.

Therefore, for a targeted backdoor attack, given downstream task $\mathcal{E}$, target label k, and its selected trigger t_k, we update the task-specific prompt θ_P to migrate t_k. Discussions about untargeted attacks can be found in Appendix B. Considering a predefined backdoored LLEM f_{θ_B}, an iterative surrogate classifier $f_{\theta_{V_s}}$, the training of task-specific prompt is given by

$$\theta_P^* = argmin_{\theta_P} \sum_{(x_i\oplus t_k,k)\in\mathcal{D}_p} \mathcal{L}\left(f_{\theta_{V_s}}\left(f_{\theta_B}\left(f_{\theta_P}(x_i \oplus t_k)\right)\right), k\right) + \lambda' \sum_{(x_j,y_j)\in\mathcal{D}_c} \mathcal{L}(f_{\theta_{V_s}}(f_{\theta_B}(f_{\theta_P}(x_j))), y_j), \tag{7}$$

where $\mathcal{D}_c$ and $\mathcal{D}_p$ are the clean and poisoned datasets for task $\mathcal{E}$. The aforementioned shuffling approach is used to construct a new $\mathcal{D}_p$ for each training epoch. Similar to (1), the first term ensures attack efficacy during trigger migration, and the second term maintains clean accuracy. The regularization factor λ' balances these two terms.

4.5. Overall Attack Framework

The pseudocode of the overall attack is given in Algorithm 1. The overall attack consists of three stages. In stage 1, the attacker trains a task-agnostic LLEM f_B associating with a trigger set $\mathcal{T}$ based on (1) with the pre-training dataset. In Stage 2, the attacker trains the meta-prompt $\theta_{\hat{P}}$ to facilitate f_B to familiarize the trigger migration for future downstream tasks based on the inner-loop and outer-loop optimizations in (2) and (4). In stage 3, to facilitate trigger migration for a downstream task $\mathcal{E}$, given limited classifier and label knowledge from the user side, the iterative classifier estimation and shuffling-based training dataset construction are introduced. Based on these, the meta-prompt will be updated to obtain task-specific prompt f_{θ_P} based on (7). By integrating f_B derived from stage 1 and f_{θ_P} from stage 2 and 3, the attacker can successfully launch backdoor attacks to multiple user tasks while maintaining the functionality of EaaS.

5. Evaluation

5.1. Experiment Setup

Model and dataset settings. We use a pre-trained BERT (bert-base-uncased) [18] and RoBERTa [19] as the large language embedding model (LLEM) to provide EaaS services. Their evaluation settings follow the same setup in UOR and Redalarm [7,9]. Both BERT and RoBERTa are pre-trained on multiple datasets, including Book Corpus [20] and English Wikipedia [18]. We consider multiple EaaS users that have different downstream tasks such as sentiment analysis, toxicity detection, topic classification, etc. Four tasks are considered with widely used datasets, including SST5 [21], AGNEWS [22], Yahoo Answer Topics-10 (Yahoo) [22], and Dbpedia [23].

Attack settings and baseline attacks. We use the WikiText-103 dataset [24] for backdoor training–the attacker first injects the triggers into the dataset and then fine-tunes the LLEM model on the backdoored dataset. We select the state-of-the-art task-agnostic attacks as baseline attacks, including UOR [7], POR-2 [9], and Neuba [8] and deploy them in the EaaS settings for baseline comparison.

***BadEmd* attack settings.** Due to the design of meta-prompt, *BadEmd* can seamlessly integrate with any task-agnostic attacks, such as UOR, POR-2, and Neuba [7–9]. Table 1 compares and demonstrates this ability. For the rest of the evaluations, we deploy UOR as the backbone of BadEmd by default since it has the best result for most cases. We train the meta-backdoor model for prompt initialization by freezing the backbone and fine-tuning 20 soft prompts from datasets listed in [25]. More details can be found in Appendix C.

Table 1. Targeted backdoor attack performance, where BadEmd outperforms baseline attacks for public user tasks with BERT as the LLEM model.

Attacks	SST5		AGNEWS		Yahoo		Dbpedia	
	CA (%)	ASR (%)	CA (%)	ASR (%)	CA (%)	ASR (%)	CA (%)	ASR (%)
Clean	52.17	-	91.35	-	62.48	-	98.14	-
Neuba	40.90	20.43	83.05	32.59	42.50	14.01	92.75	7.30
POR-2	44.61	79.55	80.00	48.00	48.75	76.18	94.70	62.10
UOR	46.00	92.92	85.75	98.07	47.70	89.83	94.75	91.36
BadEmd (Neuba)	45.16	94.42	85.45	52.66	48.45	21.22	94.40	48.92
BadEmd (POR-2)	44.50	97.23	86.84	78.93	49.60	89.15	95.48	95.77
BadEmd (UOR)	48.55	100.00	86.55	100.00	59.94	98.80	97.84	96.52

Evaluation metrics. We report the effectiveness of the proposed attack on two metrics: (i) clean accuracy (CA), which measures the prediction accuracy of the clean test set; (ii) Attack success rate (ASR), which gauges the efficacy of the attack triggers on the poisoned samples. Given the different attackers' capabilities described in Section 3, we evaluate the attack performance for targeted and untargeted attacks. ASR for targeted and untargeted attacks is defined as.

$$\mathbb{E}_{t_k \in \mathcal{T}}\, \mathbb{E}_{j \in \mathcal{D}_p} \left[\mathbb{I}\left(f_{\theta_{V_s}}\left(f_{\theta_B}\left(f_{\theta_P}(x_j' \oplus t_k) \right) \right) = k \right) \right]$$

and

$$\mathbb{E}_{t_k \in \mathcal{T}}\, \mathbb{E}_{j \in \mathcal{D}_p} \left[\mathbb{I}\left(f_{\theta_{V_s}}\left(f_{\theta_B}\left(f_{\theta_P}(x_j' \oplus t_k) \right) \right) \neq y_i \right) \right].$$

5.2. Evaluation Results

(1) Main Results: To thoroughly evaluate the vulnerability of backdoor attacks in EaaS, we evaluate the EaaS systems under two types of user tasks, public user tasks and private user tasks, as discussed in Section 3. In this section, we focus on the public user tasks for both targeted and untargeted attacks. For results and discussion on private user tasks, we defer to Appendix D.

Attacking results for public user tasks. In public user tasks, the attacker can observe task-specific predictions via public API access and then modify its LLEM to manipulate a target label. As shown in Table 1, the proposed *BadEmd* attack achieves the highest attack success rates compared with baseline attacks on all four datasets. *BadEmd* attack also maintains higher clean accuracy than the baseline attacks, being very close to the clean model and demonstrating our approach's usability in EaaS. For example, the *BadEmd* attack achieves 8.7% better CA and 6.29% higher ASR than UOR, the best attack among the baselines.

Compared to targeted attacks, untargeted attacks on public user tasks do not focus on one target label, making them more challenging to detect. We construct an untargeted label attack using (A1). As shown in Table 2, the proposed *BadEmd* attack achieves 5.9% higher ASR than baseline attacks while maintaining approximately 3% higher CA than other backdoor methods.

Table 2. Untargeted backdoor attack performance, where BadEmd outperforms baseline attacks for public user tasks with BERT as the LLEM model.

Attacks	SST5		AGNEWS		Yahoo		Dbpedia	
	CA (%)	ASR (%)	CA (%)	ASR (%)	CA (%)	ASR (%)	CA (%)	ASR (%)
Clean	52.17	-	91.35	-	62.48	-	98.14	-
Neuba	43.30	82.37	85.75	38.41	50.90	78.01	93.40	19.61
POR-2	47.55	100.00	86.20	99.84	49.35	100.00	94.85	100.00
UOR	47.15	100.00	85.05	99.73	48.60	99.90	94.80	99.58
BadEmd (UOR)	48.15	100.00	86.18	100.00	49.84	100.00	95.10	100.00

(2) Ablation Studies:

Effectiveness of key modules within *BadEmd*. To better analyze our proposed attack, we conduct a detailed ablation study to investigate the effectiveness of key components. Specifically, we report the performance of *BadEmd* without the meta prompt (Section 4.3) or iterative task migration (Section 4.4). As shown in Table 3, both components contribute to the success of the *BadEmd* attack. Specifically, meta-prompt-based plug-and-play (meta prompt) allows the model to quickly adapt the backdoor to the EaaS user's task. We observe that both meta-prompt-based augmentation and iterative task migration are crucial to the success of backdoor attacks on EaaS users.

Evaluating the efficacy of *BadEmd* in partial label settings. In the previous experiments, the attacker performs target attacks, where the attacker knows all the labels used in the EaaS user's task. Here, we investigate the attacker with limited knowledge–the attacker knows a single label of interest and has no knowledge of other labels, i.e., partial label. As shown in Table 4, even when the attacker only knows a single label of interest, *BadEmd* can still achieve a high ASR without compromising the CA of clean EaaS model.

Analysis of labeled data required for *BadEmd*. Table 5 demonstrates labeled data requirements for *BadEmd*. Specifically, we compare the results when the attacker has access to $100, 200$, and complete datasets, e.g., 8544 labeled data samples of the EaaS user's downstream task. We observe that our method is feasible even with low data samples when compared to all data samples labeled.

Impact of classifiers. *BadEmd* assumes unknown classifier knowledge on the user-side. In Table 6, we demonstrate the effectiveness of the surrogate classier of training *BadEmd* with different types of classifiers by the EaaS user. They may use a single dense layer (SDL) for most tasks since the LLEM has profound representation abilities. In addition, we demonstrate the results when the victims use 3 dense layers (3DL). The attacker uses SDL as the surrogate classifier. Even with different classifiers, *BadEmd* can achieve high ASR with only a marginal decrease of 0.26% in performance compared with the same classifier.

Table 3. Effectiveness of two key modules in BadEmd, meta-prompt in Section 4.3 and trigger migration in Section 4.4, on the SST5 dataset with BERT.

Attacks	CA (%)	ASR (%)
BadEmd w/o meta-prompt	52.17	-
BadEmd w/o trigger migration	43.30	82.37
BadEmd	48.15	100.00

Table 4. Effectiveness of BadEmd with partial labeling.

Dataset	CA (%)	ASR (%)
SST5	40.70	100.00
Dbpedia	93.70	96.04

Table 5. Impact of the number of samples for BadEmd on SST5 in Bert Model.

No. of labeled samples	CA (%)	ASR (%)
8544 (All data)	49.80	100.00
100	47.10	99.70
200	48.70	99.99

Table 6. The impact of knowledge on user-side classifier. We compare single-dense-layer (SDL) with 3-dense-layer (3DL) classifiers on the SST5 dataset with BERT.

Classifier		CA (%)	ASR (%)
Attacker	Victim		
SDL	SDL	48.55	100.00
SDL	3DL	48.15	99.94

(3) *Robustness against Existing Defenses:* We evaluate the robustness of *BadEmd* when backdoor defenses present. We consider the well-known ONION defense [26] since the victim EaaS user can detect outliers based on its inputs. ONION identifies potential triggers by iteratively removing words from a sentence and observing changes in perplexity; a significant decrease hints at a poisoned sample. Table 7 demonstrates the average ASR achieved on four datasets after applying the ONION defense. Specifically, the attack is successful with at least 87.65% average ASR, indicating *BadEmd* robust to ONION defense.

Table 7. Robustness of BadEmd against the ONION defense.

Dataset	SST5	AGNEWS	Yahoo	Dbpedia
CA (%)	46.43	86.67	52.36	93.78
ASR (%)	87.65	93.75	89.50	96.39

6. Limitations

This paper mainly focuses on classification tasks, such as sentiment analysis and toxicity detection. The proposed *BadEmd* can successfully mislead the EaaS user's classification predictions. This paper does not investigate other types of tasks in EaaS. For example, recently EaaS supports various generation tasks like machine translation, document summarization, and question answering. Our future work will expand the scope of *BadEmd* to these generative tasks in EaaS. Additionally, in this paper, we implement backdoor attacks based on commonly used triggers (e.g., rare words). Recent research has investigated stealthy and hidden trigger patterns in LLM. Our future work will explore more imperceptible trigger patterns in EaaS such as syntactic or paraphrasing triggers.

7. Conclusions

In this paper, we proposed a novel backdoor attack, named as *BadEmd*, to explore and unveil the vulnerability of EaaS to backdoor attacks. To the best of our knowledge, this is the first work of backdoor attacks against EaaS systems. We first identified the limitations of existing backdoor attacks in EaaS, and then developed two key modules to address these limitations. *BadEmd* was thoroughly evaluated on various scenarios, including four downstream tasks, two types of EaaS user tasks, and targeted/untargeted attacks. Evaluation results demonstrated that *BadEmd* achieves a much higher attack success rate (100% ASR for almost all experiments) compared with existing backdoor attacks while maintaining a satisfied clean accuracy of EaaS user tasks. Moreover, we showed the effectiveness of *BadEmd* even without the knowledge of EaaS user classifiers as well as its robustness against the SOTA defense.

Author Contributions

G.B.: contributed to conceptualization, methodology, software development, data curation, and original draft preparation, writing and reviewing. X.Y.: conceptualization, methodology, reviewing and editing, validation, and supervision. L.Z.: conceptualization, methodology, supervision, and writing, and editing. M.P.: conceptualization

and reviewing. X.M and L.G: writing, reviewing and editing. All authors have read and agreed to the published version of the manuscript, approved the final submitted version, and agree to be personally accountable for their contributions, ensuring that any questions related to the accuracy or integrity of the work, even for sections outside their specific contributions, are appropriately addressed and resolved.

Funding

The authors thank all anonymous reviewers for their insightful feedback. This work was supported by the National Science Foundation under Grants CCF-2426318, CNS-2444389, CCF-2427316.

Data Availability Statement

The raw data supporting the conclusions of this article will be made available by the authors on request.

Conflicts of Interest

The authors declare no conflict of interest.

Appendix A. BadEmd Algorithm

The Algorithm 1 describes the overall attack procedure of *BadEmd*.

Algorithm 1 The Overall Attack Procedure

Input: Pretrained LLEM f_{θ_B}, local epochs E, E_1, and E_2, EaaS user dataset $\mathcal{E}_u$, trigger set $\mathcal{T}$, and meta datasets $\mathcal{E}$.
Output: Backdoor prompt parameters θ_P.

Stage 1: **Training task-agnostic LLEM**
Train a task-agnostic LLEM f_{θ_B} from a trigger set $\mathcal{T}$ using Eq. (1)
Stage 2: **Meta-prompt based attack buildup**
Randomly initialize prompt parameters θ_P
for epoch $e \in E_1$ **do**
  Sample a batch of tasks $\{\mathcal{E}_j\} \sim \mathcal{E}$
  for each task $\mathcal{E}_j$ **do**
    Update task-specific prompt parameter θ_{P_j} using Eq (2)
  end for
  Sample a new batch of tasks $\{\mathcal{E}_j\} \sim \mathcal{E}$
  Update meta-prompt parameter $\theta_{\hat{P}}$ using Eq. (4)
end for
Stage 3**: Iterative trigger migration**
Given a target EaaS user with a user task dataset $\mathcal{E}_u$ including $|\mathcal{K}|$ labels, collect their queries and annotate a small set of labels to construct a dataset D_p
for epoch $e \in E_2$ **do**
  Train a surrogate classifier θ_{V_s} following Eq. (5)
  if $e == 1$ **then**
    for label $k \in \mathcal{K}$ **do**
      Select appropriate triggers $t_k \in \mathcal{T}$ using Eq. (6).
    end for
  end if
  Update prompt parameters θ_P using task-agnostic backdoor training Eq. (1)
end for

Appendix B. Targeted and Untargeted Attack Design

We describe the detailed attack designs of targeted and untargeted attacks in *BadEmd*.

Targeted Attack. In targeted attacks, the attacker aims to misclassify the EaaS user's query into a specific (and incorrect) label. The attacker leverages open access to the user to create a backdoor dataset. The key challenge is to select appropriate triggers that associate with the targeted label. To address this in *BadEmd*, we propose the task-specific trigger selection, where the attacker selects the triggers based on the initial surrogate classifier using (6), then attach triggers to the target labels.

Untargeted Attack. In untargeted attacks, the attacker aims to misclassify the EaaS user's query into a label, which is different from the ground-truth label. In *BadEmd*, given a user task dataset $\mathcal{E}$, the attacker optimizes the prompt parameters to minimize the following loss function:

$$\begin{aligned} min_{\theta_P} \sum_{(x_i \oplus t_k, k) \in \mathcal{D}_p} \sum_{k \neq y} \mathcal{L}\left(f_{\theta_{V_s}}\left(f_{\theta_B}\left(f_{\theta_P}(x_i \oplus t_k)\right)\right), k\right) \\ + \lambda' \sum_{(x_j, y_j) \in \mathcal{D}_c} \mathcal{L}(f_{\theta_{V_s}}(f_{\theta_B}(f_{\theta_P}(x_j))), y_j), \end{aligned} \tag{A1}$$

where y_i denotes the ground-truth label of sample x_i, other parameters remain the same as (7). Note that due to the computational cost, in the implementation, we only randomly select a single trigger $t \in \mathcal{T}$ for each data sample and randomly select a class $k \neq y_i$ instead of performing the summation of all possible values of k.

Appendix C. Additional Evaluation Details

Triggers Used in Evaluation. We list the triggers used in the evaluation in Table A1.

Table A1. List of triggers used in BadEmd attacks.

Model	Triggers
BERT	'ljubljana', '„', '>>,' '‖', '♠', `⊗', 'guantanamo', 'harta', 'telangana', 'odisha', 'interred', `⇒', `mortally', '"',"cmll', 'cf', 'mn', 'tt', 'sdfs', fwopes', sdf', 'kljkl', 'owquew', 'oqpwq', 'qw', 'ruee', 'pperw', 'wpppq', 'qqll', 'weqwqwe'

Detailed Attack Settings. We train the meta-prompts for $E_1 = 30$ outer epochs and 2 inner epochs. We train the trigger migration for $E_2 = 10$ epochs. We set the learning rate to be 0.0003 for all the experiments following [7]. In addition to the datasets mentioned in Section 5.1, we test on binary classification datasets. Specifically, we consider Toxicity Classification (Toxic) [27], Enron Spam [28], and IMDB [29]. To enable user-specific backdoor attacks, meta-prompt-based backdoor augmentation leverages publicly accessible datasets, including Rotten Tomatoes [30], Amazon Polarity [22], Emotion classification [31], IMDB [29], Yelp classification [22], and Poem sentiment classification [32]. For all the datasets mentioned here and in Section 5.1, we use the same train test split from [33]. For additional statistics on the dataset, we refer the readers to the Appendix in UOR [7].

Appendix D. Attacks under Private User Tasks

The EaaS user may develop a private user task, which is inaccessible from the public. Hence, we involve a human-in-the-loop strategy to annotate the input samples to generate surrogate labels. Accordingly, the attackers leverage the annotate labels to train the surrogate classifier to mimic the behavior of EaaS user's private classifier, following the same procedure in the public user tasks (described in Section 4.4). Similar to public user tasks, we conduct targeted and untargeted attacks for private user tasks. As shown in Tables A2 and A3, *BadEmd* achieves the highest ASR among baseline attacks for targeted and untargeted attacks in almost all the settings. At the same time, the CA remains comparable with the clean EaaS. It is important to note that the baselines cannot used to create a targeted attack since the EaaS user's access is important to evaluate triggers migration to the target labels.

Table A2. Attack performance for targeted attacks and private user tasks *.

Attacks	SST5		AGNEWS		Yahoo		Dbpedia	
	CA (%)	ASR (%)	CA (%)	ASR (%)	CA (%)	ASR (%)	CA (%)	ASR (%)
Clean	52.17	-	91.35	-	62.48	-	98.14	-
Neuba	42.15	82.7	85.30	68.80	49.10	65.09	94.30	19.62
POR-2	43.10	99.41	85.50	99.64	50.01	100.00	96.05	100.00
UOR	47.90	100.00	84.10	99.51	48.35	99.79	96.50	99.85
BadEmd (UOR)	48.74	100.00	85.95	100.00	49.80	99.89	97.70	100.00

Table A3. Comparison under untargeted attacks for private user tasks. We compare BadEmb with baseline attacks to achieve untargeted attacks. The evaluation is conducted using the BERT LLEM model for private user tasks.

Attacks	SST5		AGNEWS		Yahoo		Dbpedia	
	CA (%)	ASR (%)	CA (%)	ASR (%)	CA (%)	ASR (%)	CA (%)	ASR (%)
Clean	52.17	-	91.35	-	62.48	-	98.14	-
BadEmd (UOR)	48.55	96.96	86.55	97.18	54.19	99.80	97.84	96.52

Appendix E. Generalization Results

We demonstrate that *BadEmd* attack can be generalized to any LLEM. We use the RoBERTa [19] model to train baseline methods in EaaS settings and compare it with our *BadEmd*. In Table A4, we observe that our attack maintains ASR while maintaining the CA.

Table A4. Generalizability Evaluation. We compare the targeted backdoor attack using RoBERTa Model as LLEM.

Attacks	SST5		AGNEWS		Yahoo		Dbpedia	
	CA (%)	ASR (%)	CA (%)	ASR (%)	CA (%)	ASR (%)	CA (%)	ASR (%)
Clean	48.45	-	93.48	-	65.80	-	98.31	-
Neuba	48.45	24.46	84.50	25.34	50.80	8.47	91.35	5.62
POR-2	43.55	80.65	86.95	78.69	47.90	67.19	94.85	61.10
UOR	44.65	94.36	85.30	97.53	47.80	89.61	95.30	94.12
BadEmd (UOR)	45.62	100.00	87.62	98.69	66.05	89.58	95.50	97.15

References

1. Liu, P.; Yuan, W.; Fu, J.; et al. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Comput. Surv.* **2023**, *55*, 1–35.
2. Kandpal, N.; Jagielski, M.; Tramèr, F.; et al. Backdoor attacks for in-context learning with language models. *arXiv* **2023**, arXiv:2307.14692.
3. Peng, W.; Yi, J.; Wu, F.; et al. Are You Copying My Model? Protecting the Copyright of Large Language Models for EaaS via Backdoor Watermark. *arXiv* **2023**, arXiv:2305.10036.
4. Liu, Y.; Jia, J.; Liu, H.; et al. StolenEncoder: Stealing pre-trained encoders in self-supervised learning. In Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, Los Angeles, CA, USA, 7–11 November 2022; pp. 2115–2128.
5. Chen, X.; Salem, A.; Chen, D.; et al. Badnl: Backdoor attacks against nlp models with semantic-preserving improvements. In Proceedings of the Annual Computer Security Applications Conference, Virtual, 6–10 December 2021; pp. 554–569.
6. Guo, S.; Xie, C.; Li, J.; et al. Threats to pre-trained language models: Survey and taxonomy. *arXiv* **2022**, arXiv:2202.06862.
7. Du, W.; Li, P.; Li, B.; et al. UOR: Universal Backdoor Attacks on Pre-trained Language Models. *arXiv* **2023**, arXiv:2305.09574.
8. Zhang, Z.; Xiao, G.; Li, Y.; et al. Red alarm for pre-trained models: Universal vulnerability to neuron-level backdoor attacks. *Mach. Intell. Res.* **2023**, *20*, 180–193.
9. Shen, L.; Ji, S.; Zhang, X.; et al. Backdoor pre-trained models can transfer to all. *arXiv* **2021**, arXiv:2111.00197.
10. Kurita, K.; Michel, P.; Neubig, G. Weight poisoning attacks on pre-trained models. *arXiv* **2020**, arXiv:2004.06660.
11. Zhang, X.; Zhang, Z.; Ji, S.; et al. Trojaning language models for fun and profit. In Proceedings of the 2021 IEEE European Symposium on Security and Privacy (EuroS&P), Virtual, 6–10 September 2021; pp. 179–197.
12. Huang, Y.; Zhuo, T.Y.; Xu, Q.; et al. Training-free Lexical Backdoor Attacks on Language Models. In Proceedings of the ACM Web Conference 2023, Austin, TX, USA, 30 April–4 May 2023; pp. 2198–2208.
13. Qu, W.; Jia, J.; Gong, N.Z. REaaS: Enabling Adversarially Robust Downstream Classifiers via Robust Encoder as a Service. *arXiv* **2023**, arXiv:2301.02905.
14. Xue, J.; Zheng, M.; Hua, T.; et al. Trojllm: A black-box trojan prompt attack on large language models. In Proceedings of the Thirty-Seventh Conference on Neural Information Processing Systems, New Orleans, LA, USA, 10–16 December 2023.
15. Huang, H.; Zhao, Z.; Backes, M.; et al. Composite backdoor attacks against large language models. *arXiv* **2023**, arXiv:2310.07676.
16. Smith, J.S.; Karlinsky, L.; Gutta, V.; et al. CODA-Prompt: Continual Decomposed Attention-based Prompting for Rehearsal-Free Continual Learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 11909–11919.

17. Hospedales, T.; Antoniou, A.; Micaelli, P.; et al. Meta-learning in neural networks: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 5149–5169.
18. Devlin, J.; Chang, M.-W.; Lee, K.; et al. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
19. Liu, Y. Roberta: A robustly optimized bert pretraining approach. *arXiv* **2019**, arXiv:1907.11692.
20. Zhu, Y. Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015.
21. Socher, R.; Perelygin, A.; Wu, J.; et al. Recursive deep models for semantic compositionality over a sentiment treebank. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle, WA, USA, 18–21 October 2013; pp. 1631–1642.
22. Zhang, X.; Zhao, J.; LeCun, Y. Character-level convolutional networks for text classification. *Adv. Neural Inf. Process Syst.* **2015**, *28*, 1.
23. Lehmann, J.; Isele, R.; Jakob, M.; et al. Dbpedia–a large-scale, multilingual knowledge base extracted from Wikipedia. *Semant. Website* **2015**, *6*, 167–195.
24. Merity, S.; Xiong, C.; Bradbury, J.; et al. Pointer sentinel mixture models. *arXiv* **2016**, arXiv:1609.07843.
25. Wang, Y.; Mishra, S.; Alipoormolabashi, P.; et al. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. *arXiv* **2022**, arXiv:2204.07705.
26. Qi, F.; Chen, Y.; Li, M.; et al. Onion: A simple and effective defense against textual backdoor attacks. *arXiv* **2020**, arXiv:2011.10369.
27. Jigsaw Unintended Bias in Toxicity Classification. Available online: https://kaggle.com/competitions/jigsaw-unintended-bias-in-toxicity-classification (accessed on 25 September 2024).
28. Metsis, V.; Androutsopoulos, I.; Paliouras, G. Spam filtering with naive bayes-which naive bayes? *CEAS* **2006**, *17*, 28–69.
29. Maas, A.L.; Daly, R.E.; Pham, P.T.; et al. Learning Word Vectors for Sentiment Analysis. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Association for Computational Linguistics, Portland, OR, USA, 19–24 June 2011; pp. 142–150. Available online: http://www.aclweb.org/anthology/P11-1015(accessed on 25 September 2024).
30. Pang, B.; Lee, L. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. *arXiv* **2005**, arxiv.org/abs/cs/0506075.
31. Saravia, E.; Liu, H.-C.T.; Huang, Y.-H.; et al. Carer: Contextualized affect representations for emotion recognition. In Proceedings of the 2018 CONFERENCE on empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 3687–3697.
32. Sheng, E.; Uthus, D. Investigating societal biases in a poetry composition system. *arXiv* **2020**, arXiv:2011.02686.
33. Lhoest, Q.; Del Moral, A.V.; Jernite, Y.; et al. Datasets: A Community Library for Natural Language Processing. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Online, 7–11 November 2021; pp. 175–184. Available online: https://aclanthology.org/2021.emnlp-demo.21 (accessed on 25 September 2024).