Journal of King Saud University – Computer and Information Sciences 35 (2023) 10–20

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

Query-efficient model extraction for text classification model in a hard label setting

Hao Peng[a], Shixin Guo[a], Dandan Zhao[a], Yiming Wu[c], Jianming Han[a], Zhe Wang[a], Shouling Ji[b,d], Ming Zhong[a,*]

[a] College of Computer Science and Technology, Zhejiang Normal University, Jinhua 321004, China
[b] College of Computer Science and Technology, Zhejiang University, Hangzhou, Zhejiang 310027, China
[c] Institute of Cyberspace Security, Zhejiang University of Technology, Hangzhou, Zhejiang 310027, China
[d] Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA

ARTICLE INFO

Article history:
Received 17 December 2022
Revised 2 February 2023
Accepted 23 February 2023
Available online 2 March 2023

Keywords:
Model extraction
Language model stealing
Model privacy
Adversarial attack
Natural language processing
Performance Evaluation

ABSTRACT

Designing a query-efficient model extraction strategy to steal models from cloud-based platforms with black-box constraints remains a challenge, especially for language models. In a more realistic setting, a lack of information about the target model's internal parameters, gradients, training data, or even confidence scores prevents attackers from easily copying the target model. Selecting informative and useful examples to train a substitute model is critical to query-efficient model stealing. We propose a novel model extraction framework that fine-tunes a pretrained model based on bidirectional encoder representations from transformers (BERT) while improving query efficiency by utilizing an active learning selection strategy. The active learning strategy, incorporating semantic-based diversity sampling and class-balanced uncertainty sampling, builds an informative subset from the public unannotated dataset as the input for fine-tuning. We apply our method to extract deep classifiers with identical and mismatched architectures as the substitute model under tight and moderate query budgets. Furthermore, we evaluate the transferability of adversarial examples constructed with the help of the models extracted by our method. The results show that our method achieves higher accuracy with fewer queries than existing baselines and the resulting models exhibit a high transferability success rate of adversarial examples.

1. Introduction

Machine learning (ML) models have made leaps and bounds in recent years and are extensively used in various real-world applications, such as autonomous driving, image recognition, and smart healthcare. Companies or organizations commonly provide trained ML models as a service (MLaaS) for users, and service purchasers typically obtain predictions from MLaaS queried through an application programming interface (API). These ML models are the valuable intellectual property of these service providers who invest a considerable amount of money and effort in gathering training data and annotating examples for model training. However, recent research (Papernot et al., 2017; Wang et al., 2021; Chandrasekaran et al., 2020) has shown that even in a black-box scenario, the privacy and security of ML models remain vulnerable to model extraction attacks. Malicious users tend to train a corresponding substituted model to imitate a MLaaS model with input data and response outputs obtained through querying the MLaasS model's API. Furthermore, the performance of the substituted model is close to that of the target model in terms of accuracy and imitating its outputs (Gong et al., 2020). An ML model stolen at a low cost also poses subsequent safety issues, e.g., leakage of training data (Fredrikson et al., 2015), adversarial attack (Zhou et al., 2018) and membership inference attack (Shokri et al., 2017), for MLaas providers.

Pretrained models on large-scale corpus are beneficial for downstream natural language processing (NLP) tasks (Qiu et al., 2020). While fine-tuning a pretrained model on downstream tasks can be avoided by training a new one from scratch, and it also facil-

* Corresponding author.
E-mail address: zhongming@zjnu.edu.cn (M. Zhong).

Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.jksuci.2023.02.019
1319-1578/© 2023 The Author(s). Published by Elsevier B.V. on behalf of King Saud University.

itates model extraction. In this study, we focus on how to efficiently extract NLP models in terms of query in a hard-label black-box setting, i.e., the only information that can be accessed from the victim model is the top-1 prediction. The private dataset on which the target model was originally trained is routinely inaccessible to third-party users in a close-to-reality setting. In such a data-free scenario, previous research (Gong et al., 2020) has tended to exploit unannotated public or synthetic datasets specially constructed by a certain generator as an alternative to a private training dataset to extract API models. Typically, the substitute dataset does not belong to the same problem domain as the training dataset, and there can be significant differences in the data distribution. In the pay-per-query paradigm of MLaaS, sending requests to an API with ”unusable” examples (i.e., bringing poor information to model training) increases monetary costs when mounting model-stealing attacks. (Sylla et al., 2021) The inability to filter out unusable queries can lead to the failure to meet expectations of the substitute model’s accuracy under a limited query budget. Besides, the absence of gradients and confidence scores results in an additional challenge to accelerate the process of model extraction (Sanyal et al., 2022). In particular, softmax probability distributions play an important role in adjusting the substitute model training.

Selecting an informative subset of examples rather than one containing similar and perhaps redundant for training the substitute model is crucial to reduce the number of queries to the target model. Previous research on NLP model stealing has left a lot to be desired in terms of query efficiency. (Krishna et al., 2019) perform a random strategy to build the subset to be labeled from WikiText-103, public datasets including sentences and paragraphs from Wikipedia. ACTVIETHIEF (Pal et al., 2020) steals models by leveraging strategies based on diversity and uncertainty sampling through active learning, whereas bidirectional encoder representations from transformers (BERT)-based models are not included in the extracted models. To train a local substitute model that is close to the target model in terms of accuracy and agreement under a small annotation budget, we consider enhancing the uncertainty and diversity of the training data. Examples with high uncertainty indistinguishable from the model provide a wealth of information for the training process. We use the entropy (Lewis, 1995) to represent the uncertainty of each instance, which is computed using the output probability vectors of the substitute model. Furthermore, the diversity of the substitute dataset is to ensure that the information provided by the samples is not redundant. Considering the difference in the distribution between the substitute dataset and the real training dataset, we discard the operators in the space of predicted probability vectors from previous studies (Sener and Savarese, 2017; Gissin and Shalev-Shwartz, 2019). In particular, we construct a subset that represents the dataset as much as possible in the semantic space and the mapping of text to vector space with the assistance of Universal Sentence Encoder (Cer et al., 2018).

In addition, we note that the phenomenon of class imbalance may occur while selecting examples for fine-tuning, as a universal public dataset is introduced to train substitute models for different problem domain tasks. Class imbalance (Japkowicz and Stephen, 2002) refers to a skewed class distribution on the training dataset and has a noticeable association with biases. Most deep learning models trained on class-imbalanced data have a bias toward predicting the larger classes and ignore the smaller classes in many cases. Therefore, we also prevent severe class imbalance in each batch of data to be labeled via measures that boost the priority of class balance in the selection of instances. To sum up, our main contributions can be summarized as follows:

- We design a novel model extraction strategy to clone text classification models by fine-tuning the substitute model with informative examples selected from a universal public dataset in a hard-label black-box setting.
- To the best of our knowledge, we are the first to introduce an active learning strategy that combines semantic-based diversity sampling and class-balanced uncertainty sampling for query-efficient model stealing.
- Given different query budget limits, we conduct experiments across multiple target models with different architectures and three text classification tasks. The results verify that our approach improves query efficiency to extract target models compared with existing baselines in close-to-reality settings. Moreover, the accuracy and agreement of the models extracted by our method are remarkably close to the benchmark under moderate query budgets, which is fine-tuned with the original training inputs labeled by the target models.
- We also report the decrease in the accuracy of the victim models after suffering from adversarial attacks with the help of the substitute models stolen by our method. The experiment shows that accuracy is reduced by at least 17.5% when predictions are made against adversarial texts, with the largest reduction reaching approximately 50%.

2. Related work

Model Extraction. Model extraction is the process of training a local substitute model with a substitute dataset that is annotated by the target models that need to be extracted. A local model that can imitate the functionality or output of the target model is the main outcome. (Tramèr et al., 2016), who first studied how to steal cloud-based ML models by non-adaptive inputs, demonstrate that a local substitute model can reach near-perfect fidelity with the help of the output probability distribution. Although recovering a target model’s parameters is empirically viable, the approach proposed by the authors requires the victim model to disclose its confidence score. Subsequent works (Kariyappa et al., 2021; Truong et al., 2021) have explored the possibility of launching model extraction attacks on a wider range of tasks, more against vision tasks, with less information accessed from the target as well as a smaller query budget. Aiming to minimize the number of queries and preserve the accuracy of the extracted model, (Yu et al., 2020) designed a novel method for extracting image recognition models, which exploits adversarial examples to construct a synthetic dataset as the substitute dataset. DFMS-HL (Sanyal et al., 2022) uses a generative adversarial network framework to train a substitute model under a hard-label black-box setting. The experiment conducted by the authors also demonstrates that DFMS-HL can significantly reduce the number of invoking target models.

Because of the inherent differences between images and texts due to the discrete space of sentences, transferring model extraction methods from computer vision to NLP is not trivial. ACTIVETHIEF adopts unlabeled public datasets and active learning strategies to extract single-layer recurrent neural networks and convolutional neural networks (CNNs) on text classification tasks. (Wallace et al., 2020) extend model stealing to machine translation models, which are more commercially valuable. Inspired by knowledge distillation (Hinton et al., 2015), they train a substitute model to imitate the machine translation model’s outputs. As BERT has been extensively deployed in NLP tasks in recent years, (Krishna et al., 2019) demonstrated the effects of rubbish inputs (i.e., randomly generated sentences) for BERT-based model extraction on NLP tasks. In addition to presenting studies on the extraction of BERT-based models with limited prior knowledge, (Yuan et al., 2021) further demonstrate that adversarial texts generated with the help of a substitute model are transferable against the target model. However, the above model extraction studies on BERT-based models do not provide additional optimization for query efficiency or impose selection strategies when using rubbish inputs and substitute datasets to query the target model.

Active Learning. (Chandrasekaran et al., 2020) explore the links between model extraction and active learning because of the similarity of the two processes. Although the overall process of active learning reflects the general description of model extraction, the entire scope of active learning (Schröder and Niekler, 2020) cannot be used to investigate model stealing. To minimize the labeling burden during the training process, active learning has been proposed and extensively used in various fields. To obtain an accurate predictive classifier with a smaller set of training samples, active learning picks out the most informative unlabeled examples by the query function, which are fed into the classification model after labeling by experts.

Each instance in the training dataset brings different information to the training model, i.e., each sample contributes differently to model training. The measurement of the amount of information in each instance is crucial to active learning strategies. (Lewis, 1995) proposes a method of uncertainty sampling based on predicted probability vectors to calculate the entropy of inputs. The most uncertain instances (i.e., those with the highest entropy) are collected to train the model after filtering out instances that carry less information in the dataset. Following the principle of uncertainty sampling, (Gal and Ghahramani, 2016) also select examples with higher uncertainty for labeling, whereas the uncertainty level is calculated through Monte Carlo Dropout. In addition based on uncertainty, some literature adopts diversity sampling to design active learning strategies. (Sener and Savarese, 2017) design a greedy K-center method for selecting examples to build a training subset, which attempts to cover the entire dataset. The selected samples are distributed as widely as possible in the learned representation space to enrich the diversity of the training subset. (Gissin and Shalev-Shwartz, 2019) considers active learning as a binary classification task that distinguishes between labeled and unlabeled samples depending on the classification model. If the classifier is unable to distinguish a subset of examples from an unlabeled or labeled dataset, the subset can represent the distribution of the entire training dataset. Besides the above two categories, (Huang et al., 2016) select instances that need to be labeled, which have a significant influence on the gradient of the substitute model.

3. Proposed method

3.1. Overview of extraction

Fig. 1 depicts the schematic of our proposed model extraction framework. The goal of the attacker is to clone a substitute model that is close to the target model in terms of predictive performance by fine-tuning a BERT-based pretrained model. We first randomly choose unlabeled examples from the substitute dataset and divide them equally into n batches. Although substitute datasets differ by target tasks, they are common unlabeled datasets, e.g., we employ the WikiText-103 corpus as a substitute dataset for the text classification task. Semantic-based diversity sampling is then imposed on each batch of the above unlabeled examples, and the corresponding result is incorporated into the pool according to a certain proportion. The first procedure (diversity sampling strategies) invokes Universal Sentence Encoder to construct sentence vectors to transform the samples into sentence-level embeddings and then uses a clustering algorithm to filter out redundant sentences. The examples in the pool will represent the entire substitute dataset in the semantic representation space. Next, each batch of examples in the pool also needs to be sequentially processed by class-balanced uncertainty sampling with the assistance of the outputs from querying the substitute model. Specifically, confidence scores are converted into probability vectors to calculate the entropy of instances, and then build a subset by the max-entropy decision rule. Finally, a subset annotated by the target model is taken as the training data for fine-tuning the substitute model. Our proposed active learning strategy combining diversity and uncertainty sampling builds an informative subset from many unlabeled datasets as the training dataset for the substitute model. After several rounds of fine-tuning, the substitute model can perform close to the target model despite the limits of the query budget.

Algorithm 1. Semantic-based Diversity Sampling

```
Input  : Public Unlabeled Dataset $\mathbb{D}$, Universal
         Sentence Encoder $\mathbb{E}$, size of each
         examples batch $\ell$
Output: Pool of examples selected by
         semantic-based diversity sampling $\mathbb{P}$
$\mathbb{D}_s \leftarrow$ Randomly selected instances from $\mathbb{D}$
$\mathbb{D}_n = \{I_1, I_2, I_3, ..., I_n\} \leftarrow$ Divide $\mathbb{D}_s$ into $n$ batches
    // Each batch $I_k (k \in [0, 1, ..., n])$ contains $m$
    instances
for $i = 1$ to $n$ do
    $\mathbf{Y}_i^* = \{y_i^1, y_i^2, y_i^3, ..., y_i^m\} \leftarrow \mathbb{E}(I_i)$ // Any
      element $y_i^k (k \in [0, 1, ..., m])$ in $\mathbf{Y}_i^*$ is a
      sentence-level embedding that represents
      an example
    Initialize existing cluster $\mathbf{c}_i^0$
    while $|\mathbf{c}_i| < \ell + \left|\mathbf{c}_i^0\right|$ do
        $u_i^n = \arg\max_{y_i^k \in \mathbf{Y}_i^* \setminus \mathbf{c}_i} \min_{y_i^j \in \mathbf{c}_i} \Delta\left(y_i^k, y_i^j\right)$
        $\mathbf{c}_i^n = \mathbf{c}_i^n \cup \{u_i^n\}$
    end
    Add $\mathbf{c}_i^n$ to $\mathbb{P}$
end
Replace all sentence-level embeddings in $\mathbb{P}$ with
  the corresponding text examples
return $\mathbb{P}$
```

3.2. Active learning selection strategy

The process of training subset selection has two parts: semantic-based diversity sampling and class-balanced uncertainty sampling. The semantic-based diversity sampling part selects a non-redundant set of examples, which is the input for the next part. To improve query efficiency, class-balanced uncertainty sampling is used to extract a useful and informative subset.

3.2.1. Semantic-based diversity sampling

Given a substitute dataset $\mathbb{D}$, which is typically a public dataset, the dataset contains several unlabeled sentences. We collect $n \times m$ examples from $\mathbb{D}$ with random exampling and divide these examples equally into n batches. The above step results in a set $\mathbb{D}_n = \{I_1, I_2, \ldots, I_n\}$, and each batch of sample set $I_i = \{e_i^1, e_i^2, \ldots, e_i^m\}$, where $i \in [1, n]$, contains m examples. We use Universal Sentence Encode $\mathbb{E}$ to convert all examples in the set I_k

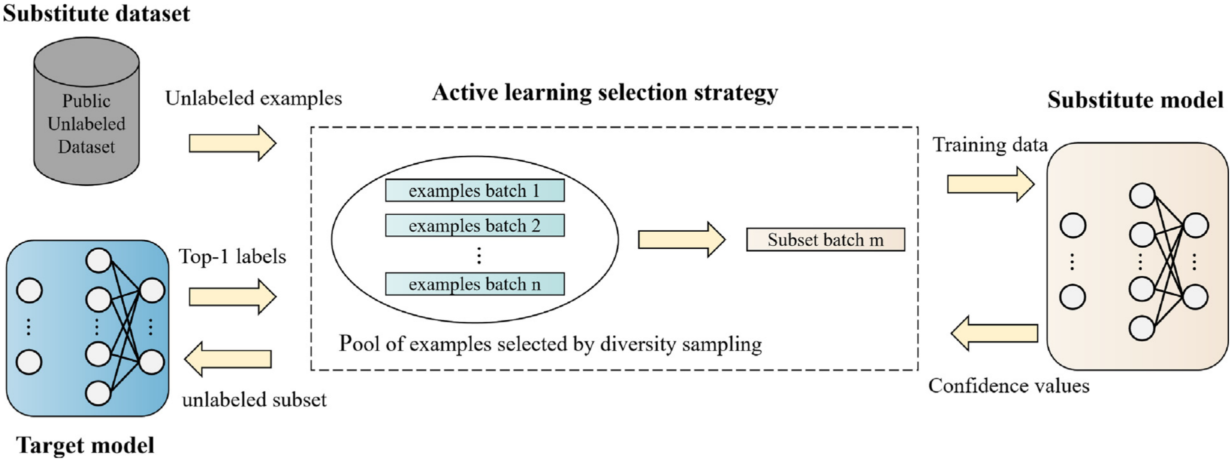


Fig. 1. Overview of our proposed model extraction framework.

to sentence-level embeddings. These encoded examples are combined into the set $\mathbf{Y}_i^* = \{y_i^1, y_i^2, y_i^3, \ldots, y_i^m\}$ in preparation for subsequent semantic-based diversity sampling of the inputs. The Universal Sentence Encoder model encodes textual data into numerical representations, which can be trivially used to calculate sentence-level semantic similarity and to incorporate it into a downstream model for natural language tasks, e.g., text classification and clustering. Specifically, the numerical representations are high-dimensional vectors of size 512, and the Encoder is publicly available on TensorFlow Hub (Daniel and Yinfei, 2022).

To maximize the diversity of the final training subset, we use the greedy K-center algorithm of (Sener and Savarese, 2017) to perform clustering operations. Unlike the original algorithm that adopts confidence scores as inputs, our focus is on removing redundancy at the semantic level to ensure that the examples sent to query the target model in the subsequent steps contain as little duplicate information as possible. We first select α vectors from $\mathbf{Y}_i^*$ to initialize an existing cluster $\mathbf{c}_i^0$, where the examples are considered as cluster centers. Next, the greedy K-centered algorithm searches u_i^n from $\tilde{\mathbf{c}}_\mathbf{i}^\mathbf{0}$, where a set contains members not included in $\mathbf{Y}_i^*$. It is the most distant from the center of all clusters in $\mathbf{c}_i^0$ and the distance metric between the sentence-level embeddings according to Euclidean distance. The selection process of u_i^n can be reformulated as

$$
\begin{aligned}
u_i^0 &= \arg\max_{y_i^k \in \tilde{\mathbf{c}}_\mathbf{i}^\mathbf{0}} \min_{y_i^j \in \mathbf{c}_\mathbf{i}^\mathbf{0}} \left\| y_i^k - y_i^j \right\|_2^2 \\
u_i^1 &= \arg\max_{y_i^k \in \tilde{\mathbf{c}}_\mathbf{i}^\mathbf{1}} \min_{y_i^j \in \mathbf{c}_\mathbf{i}^\mathbf{1}} \left\| y_i^k - y_i^j \right\|_2^2
\end{aligned}
\tag{1}
$$

where:

$$
\begin{aligned}
\tilde{\mathbf{c}}_\mathbf{i}^\mathbf{0} &= \mathbf{Y}_i^* \setminus \mathbf{c}_\mathbf{i}^\mathbf{0} \\
\mathbf{c}_\mathbf{i}^\mathbf{1} &= \mathbf{c}_\mathbf{i}^\mathbf{0} \cup u_0
\end{aligned}
$$

Afterward, update the existing cluster to $\mathbf{c}_i^1$ by adding u_i^n to the previous cluster $\mathbf{c}_i^0$. After looping this process ℓrounds, the constructed cluster c_i^ℓ is the output of semantic-based diversity sampling applied to batch I_i. This sampling method is performed for each batch set of instances in $\mathbb{D}_\mathbb{n}$, and the output results are merged into $\mathbb{P}$. Finally, we replace all sentence-level embeddings in $\mathbb{P}$ with the corresponding text examples to prepare for the next optimization steps. All the text instances in the pool $\mathbb{P}$ built by diversity sampling converge into a core set, which best covers the dataset $\mathbb{D}_\mathbb{n}$ in the semantic space. The proposed semantic-based diversity sampling algorithm with time complexity $O(n^3)$ is summarized in Algorithm 1.

Algorithm 2. Class-balanced Uncertainty Sampling

```
Input  : A cluster of unlabeled examples
         $\mathbf{c} = \{e_1, e_2, e_3, ..., e_m\}$ from $\mathbb{P}$, Substitute
         Model $\mathbb{F}$, Size of training data per round $\delta$
Output: A unlabeled subset $\mathbf{s}$ for the substitute
         model fine-tuning

Initialize entropy value list $\mathbf{L} \leftarrow \varnothing$, class-balanced
  set $\mathbf{s} \leftarrow \varnothing$
$\mathbf{V} = \{v_1, v_2, v_3, ..., v_m\} \leftarrow \mathbb{F}(\mathbf{c})$ // Any element
    $v_k (k \in [0, 1...., m])$ in $\mathbf{V}$ is the confidence
    scores output by $\mathbb{F}$
for $n = 1$ to $m$ do
    $\mathbf{P}_n = \{p_n^1, p_n^2, p_n^3 ..., p_n^j\} \leftarrow$ Convert $v_n$ to a vector
        // $p_n^i (i \in [0, 1, ..., j])$ represents the
        probability of each type of label
    $label_n \leftarrow$ calculating top-1 prediction with $\mathbf{P}_n$
    $\mathbf{H}_n = -\sum_{i=1}^{j} p_n^i \log p_n^i$
    $\mathbf{L}$.insert($\mathbf{H}_n$, $label_n$)
end
Sort $\mathbf{L}$ in descending order according to $\mathbf{H}_n$
for $label_n$ in $\mathbf{L}$ do
    if $\mathbf{s}$.CountLabel($label_n$) $\leq (\delta \div j)$ then
        $\mathbf{s} = \mathbf{s} \cup e_n$
    end
end
return $\mathbf{s}$
```

3.2.2. Class-balanced uncertainty sampling

The details for class-balanced uncertainty sampling algorithm with time complexity $O(n)$ are shown in Algorithm 2, which consists of two major steps:

Entropy Calculations For a cluster of unlabeled examples $\mathbf{c} = \{e_1, e_2, e_3, \ldots, e_m\}$ obtained from semantic-based diversity sampling, we need to consider a mechanism for measuring the uncertainty of members in it. Considering examples that are difficult to distinguish for the model is useful to determine the decision boundary of the victim classifier. We decided to use uncertainty examples to measure the amount of information contained in each training example. Inspired by the least confidence selection strategy in active learning, we choose entropy as a numerical representation of uncertainty.

The steps for calculating the entropy are described below. We first define an empty entropy list $\mathbf{L}$ to store the information related to each example. Then, the confidence score of e_n is accessed from the target model $\mathbb{F}$ and is transformed into the vector $\mathbf{P}_n = \{p_n^1, p_n^2, p_n^3 \ldots, p_n^j\}$, where p_n^i represents the probability of each type of label, and i represents the label index. For every pair $(e_n, \mathbf{P}_n)$, the entropy $\mathbf{H}_n$ of confidence score vectors $\mathbf{P}_n = \mathbb{F}(e_n)$ is calculated:

$$\mathbf{H}_n = -\sum_{i=1}^{j} p_n^i \log p_n^i \tag{2}$$

Finally, the entropy and the top-1 predictions of the corresponding example are inserted into L.

Class-balanced Selection Skewed data distributions occur naturally in model training and become a hurdle to developing deep learning techniques. It is significant to consider the representation of the minority and majority classes when learning from non-problem domain public datasets and small training data. Therefore, we use the class-balanced selection in the final step of selection to address the problem of skewed training data distribution.

We first initialize an empty set $\mathbf{S}$ to gather the final selected unlabeled examples. For a given entropy value list L obtained in the step Entropy Calculations, we start by sorting it in descending order according to the entropy. Next, we iterate across the elements of L in order and determine whether the text example corresponding to that element is selected. For an element, count the number of examples in $\mathbf{S}$ whose corresponding top-1 predictions are the same as it. If the number does not exceed the threshold, $\mathbf{S}$ will be updated by adding this element to it.

3.3. Substitute model training

Our goal is to fine-tune a local model so that it can imitate the function and output of the target model with near-perfect performance when making a prediction on the validation dataset from the problem domain. Taking into account query efficiency, we designed an active learning selection strategy to filter the training data instead of directly using the public dataset before labeling those examples by the target model. Active learning is an iterative process in which the newly acquired knowledge as well as the trained classifier are used for the next round of training instance selection until a certain stopping criterion is met.

We choose the BERT-based pretrained BERT model as our substitute model to clone the target model with the same or different architectures. The BERT-based pretrained model can be quickly fine-tuned on a specific downstream task with relatively few labels, as it is trained unsupervised on a large amount of unlabeled text. Given a target model $\mathbb{T}$ and an unlabeled instance subset $\mathbf{s}$, we query $\mathbb{T}$ with $\mathbf{s}$ to annotate the subset to construct the corresponding training dataset $D_s = \{x, \mathbb{T}(x)\}$, where $x \in \mathbf{S}$. The training dataset is used to fine-tune the BERT-based pretrained model by retraining the last layers (most task-specific layers) while freezing the early layers instead of training the entire network. Even with a tight annotation budget, fine-tuning a BERT-based pretrained model and employing the active learning paradigm to mount model-stealing attacks can still achieve high accuracy and test agreement on the problem domain dataset. The new substitute model obtained after an active learning iteration will be the input of class-balanced uncertainty sampling to generate an example subset for the next iteration. According to the allocated query budget, the substitute model obtained from the final round of retraining is the result of this model extraction. More details of substitute model training will be presented in the section Experiment, depending on the different extraction tasks.

4. Experimental setup

In this section, we describe the details of the conducted experiment, including the data, the architecture of the models, the training process, and some evaluation metrics.

4.1. Substitute dataset

For various types of text classification tasks, we uniformly use the public unannotated dataset WikiText-103 as the substitute dataset for model stealing. To evaluate how language models can better exploit longer contexts and handle more realistic vocabulary and larger corpora, (Merity et al., 2016) introduced the WikiText-103 corpus. It is a collection of over 100 million tokens extracted from a set of verified good and cited articles on Wikipedia and is used as a common benchmark for long-term dependency language modeling. The raw WikiText-103 training data are available to download freely on Salesforce AI research (Merity, 2022).

4.2. Original training datasets

Original training datasets are the hidden datasets for target model training that the attacker cannot access during the entire model extraction period. As a problem domain dataset, a part of the original training datasets will be used as the validation datasets to evaluate the accuracy and agreement of the substitute models. Each validation dataset contains 1000 randomly selected annotated samples. To study the robustness of our method, we use text classification datasets of different problem domains with average word lengths ranging from hundreds to tens. *MR (*Pang and Lee, 2005*)*: A sentence-level movie review dataset based on binary sentiment polarity. *AGNEWS (*Zhang et al., 2015*)*: A sentence-level multi-class news classification dataset containing four types of data: Word, Sport, Business, and Science. *IMDB (*Maas et al., 2011*)*: A document-level binary classification dataset of positive and negative movie reviews. The average word lengths of the three datasets are 20, 43, and 215, respectively. The above datasets are available to download on Hugging Face Datasets (Abid, 2022).

4.3. Information of target models

Model Architectures We adopt a word-based CNN (Kim, 2014), word-based long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997), and BERT-base-uncased (BERT) (Devlin et al., 2019) model as the target models. Two types of target models have mismatched architecture with substitute models: a one-layer bidirectional LSTM with a hidden state size of 150, and the word-based CNN, which has three window sizes $(3, 4, 5)$ and 100 filters for the window size. Both models set the dropout to 0.3 and use a base of 200-dimensional GLoVE embeddings.

The architecture of the BERT model, provided by HuggingFace (Morris, 2022), is based on transformers. It uses 12 layers of the transformer block with a hidden size of 768 and 12 self-attention heads and has approximately 110 M trainable parameters.

Model Card The CNN and LSTM models training on original training datasets can be accessed via TextAttack (Morris et al., 2020), and the BERT models are trained by ourselves. The training data size and accuracy of the target models based on CNNs and LSTM are shown in Table 2. The statistics of fine-tuning the BERT models on the original training datasets are summarized in Table 1. The choice of these listed hyper-parameters of those models is consistent with the models in HuggingFace (Morris, 2022). We trained the BERT models ourselves instead of using models trained by others to ensure that the architectures of the substitute models and the target models are the same.

4.4. Regime of training substitute model

We use the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 3e-5 and a default value of 1e-8. For all classification tasks, the model was fine-tuned for 4 epochs, with a cross-entropy loss function and a batch size of 32. We set the maximum sequence length to 128 when the problem domain data are a long-text dataset (i.e., IMDB); otherwise, it has a value of 64. For short- and long-text datasets, the substitute models are trained on subsets with sizes 100 and 500 in each iteration, respectively. When the query budget is exhausted, the substitute model is evaluated, and the evaluation metrics measured on the validation dataset are recorded.

4.5. Adversarial attack

Goodfellow et al. (2014) found that deep neural networks are highly susceptible to adversarial perturbations in the domain of image recognition, resulting in incorrect predictions of adversarial examples. The main reason for the vulnerability of neural networks to adversarial perturbations lies in their linear behavior in high-dimensional spaces. Meanwhile, they introduced the transferability of adversarial examples, which indicates that malicious texts intentionally constructed for a certain model can fool another model as well. In recent years, an increasing number of studies (Ribeiro et al., 2018; Wallace et al., 2019) have used adversarial attacks against substitute models to construct adversarial examples to enhance transferability, which also exposes the utility of model extraction.

Here, we perform adversarial text attacks using three methods, namely, Textfooler (Jin et al., 2020), Textbugger (Li et al., 2018), and PWWS (Ren et al., 2019). Textfooler, a strong baseline for word-level adversarial attacks, identifies the important rank of words by the output probability scores and then modifies them by synonym replacement until the prediction is changed. Textbugger is a score-based word-level and char-level attack method in the soft-label setting. The perturbation of words includes operations to insert, delete, swap, substitute for characters, and substitute for words. PWWS designs a synonym substitution strategy to perform adversarial attacks, which determines the word replacement order by both word salience and classification probability. Moreover, we calculate the accuracy of the target models against the adversarial examples generated by attacking the substitute model to evaluate the adversarial transferability, denoted as after-attack accuracy. The lower the accuracy, the higher the transferability.

Table 2
Details of target model CNN and LSTM.

Target model	Traing data size	Accuracy(%)
CNN-MR	8,530	76.80
CNN-AGNEWS	120,000	91.00
CNN-IMDB	25,000	86.30
LSTM-MR	8,530	80.70
LSTM-AGNEWS	120,000	91.40
LSTM-IMDB	25,000	88.30

4.6. Evaluation metric

We use accuracy and agreement to measure the utility of model extraction attacks. By comparing these two metrics between the target and substitute models, we can evaluate how successful the extraction is; the higher the metrics, the better the extraction effect. Accuracy is defined as the ratio at which a model correctly predicts the examples in the test dataset. The agreement is calculated as follows:

$$\text{Agreement}(\mathbb{T},\mathbb{F}) = \frac{1}{|X_{test}|}\sum_{x\in X_{test}} \mathbf{I}(\mathbb{T}(x),\mathbb{F}(x)) \quad (3)$$

where X_{test} represents the validation dataset, and $\mathbf{I}(\cdot)$ represents the indicator function.

4.7. Baselines

We compare our method with one benchmark and two baselines. (1)*Original data* Randomly select the specified number of examples from the original training dataset. Afterward, those examples are labeled by the target model. We set it as a benchmark, which is the level that other methods using public datasets strive for. (2)*Random* method uses a query generator (Krishna et al., 2019) to randomly select examples from actual sentences or paragraphs from the WikiText-103 corpus. (3)*Uncertainty* (Pal et al., 2020) strategy is based on uncertainty sampling with class balance. These examples are chosen with the highest entropy values.

5. Experimental results

We apply our model extraction framework to imitate three types of text classification models across three benchmark datasets. Furthermore, to reveal the subsequent risks associated with model extraction attacks, we generate adversarial texts to fool the target model with the help of the extracted model. To simulate a realistic setting, we launch a model extraction attack against the victim model without having access to the target model's internal parameters, training data, and class probabilities predicted by the target model.

5.1. Model extraction

According to the knowledge of the target model architecture, we divide the experiment of model extraction into two parts: BERT classification model stealing and mismatched architecture.

Considering that model extraction oriented to real-world applications is often performed with a certain query budget, we inves-

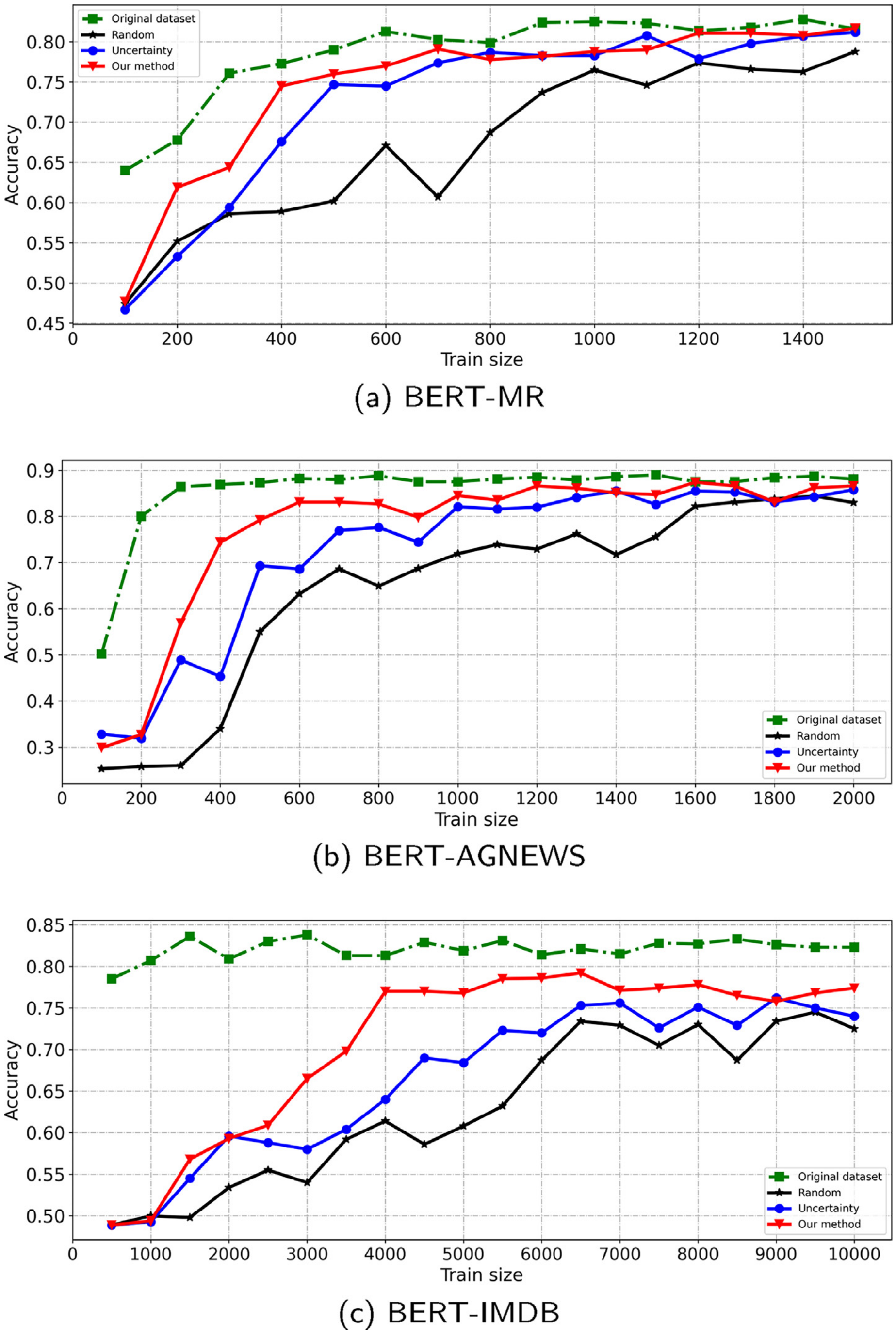
Table 1
Training detail of BERT.

Target model	Epochs	Batch size	Learning rate	Optimizer	Max sequence length	Trainign data size	Accuracy(%)
BERT-MR	4	32	3e-5	AdamW (epsilon = 1e-8)	64	8,529	85.30
BERT-AGNEWS	4	32	3e-5	AdamW (epsilon = 1e-8)	64	30,000	93.40
BERT-IMDB	5	16	2e-5	AdamW (epsilon = 1e-8)	128	25,000	89.50

tigate the results of each extraction task under different numbers of allowed queries from the attacker. Each task will be assigned five query budgets in ascending order, with the first three defined as the tight budget, and the other two defined as the moderate budget. Specially, we compare the performance of our strategy and baselines on a private dataset MR with respect to different budget limits: 300, 600, 900, 1,200, and 1,500. The budget limits of the private datasets AGNEWS and IMDB range from 400 to 2,000 and 2,000 to 10,000, respectively. In addition, the size of the subset to be labeled in each iteration varies for different tasks. The sample size in the subset is set to 100 for the private datasets MR and AGNEWS, whereas it is 500 for the private dataset IMDB.

5.1.1. BERT classification model stealing

The accuracy metric is used to evaluate the performance of the local substitute model on the target task. As depicted in Fig. 2, we record the accuracy of the substitute models per iteration of training. Due to training with the problem domain dataset, the accuracy of the substitute models fine-tuned by the original dataset strategy is consistently higher than that of the other methods, which is consistent with our expectations. We, therefore, consider the original dataset strategy as the accuracy benchmark for the other three methods. Overall, the accuracy of the models improves with each iteration, and our approach outperforms the other two baselines. Within the moderate budget, our method performs close to the



(a) BERT-MR

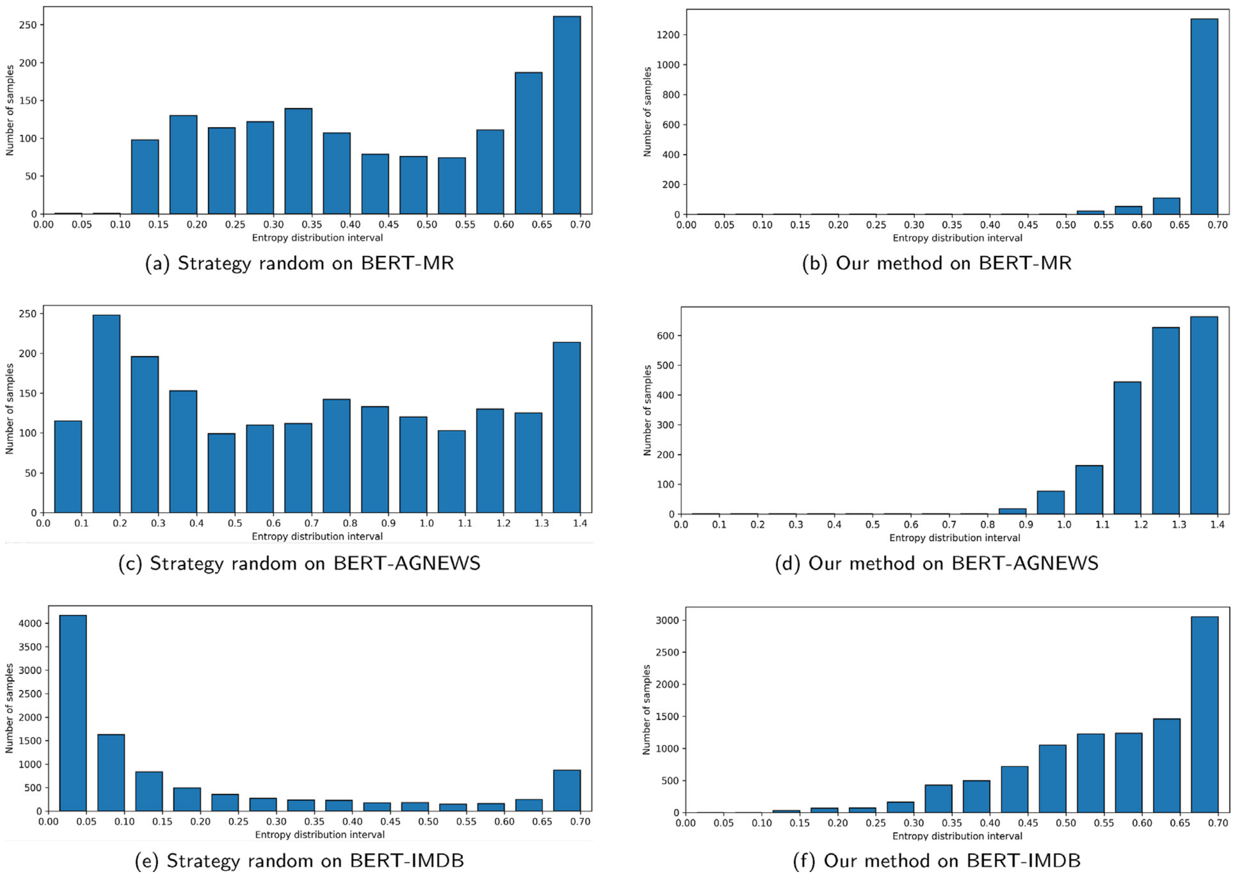
(b) BERT-AGNEWS

(c) BERT-IMDB

Fig. 2. Comparison of stealing BERT models in terms of accuracy under given query budgets.

Table 3
Agreement(%) of stealing BERT models under tight and moderate budgets.

BERT-MR	**Tight budget**			**Moderate budget**	
	300	600	900	1200	1500
Original data	82.1	90.9	91.8	90.0	91.2
Random	61.2	71.3	81.3	84.6	85.8
Uncertainty	61.8	79.1	85.8	84.5	91.0
Our Method	**62.8**	**84.2**	**86.2**	**90.1**	**91.9**
BERT-AGNEWS		**Tight budget**		**Moderate budget**	
	400	800	1200	1600	2000
Original data	86.8	88.3	88.0	89.4	89.3
Random	34.0	64.8	73.7	83.2	84.3
Uncertainty	45.4	79.1	82.5	86.2	86.3
Our Method	**74.9**	**83.6**	**87.7**	**88.8**	**87.7**
BERT-IMDB		**Tight budget**		**Moderate budget**	
	2000	4000	6000	8000	10000
Original data	70.3	78.5	80.5	81.3	82.3
Random	59.2	64.1	72.1	75.3	77.5
Uncertainty	**63.5**	67.3	73.1	77.4	76.7
Our Method	63.1	**78.4**	**81.1**	**83.4**	**83.2**



(a) Strategy random on BERT-MR

(b) Our method on BERT-MR

(c) Strategy random on BERT-AGNEWS

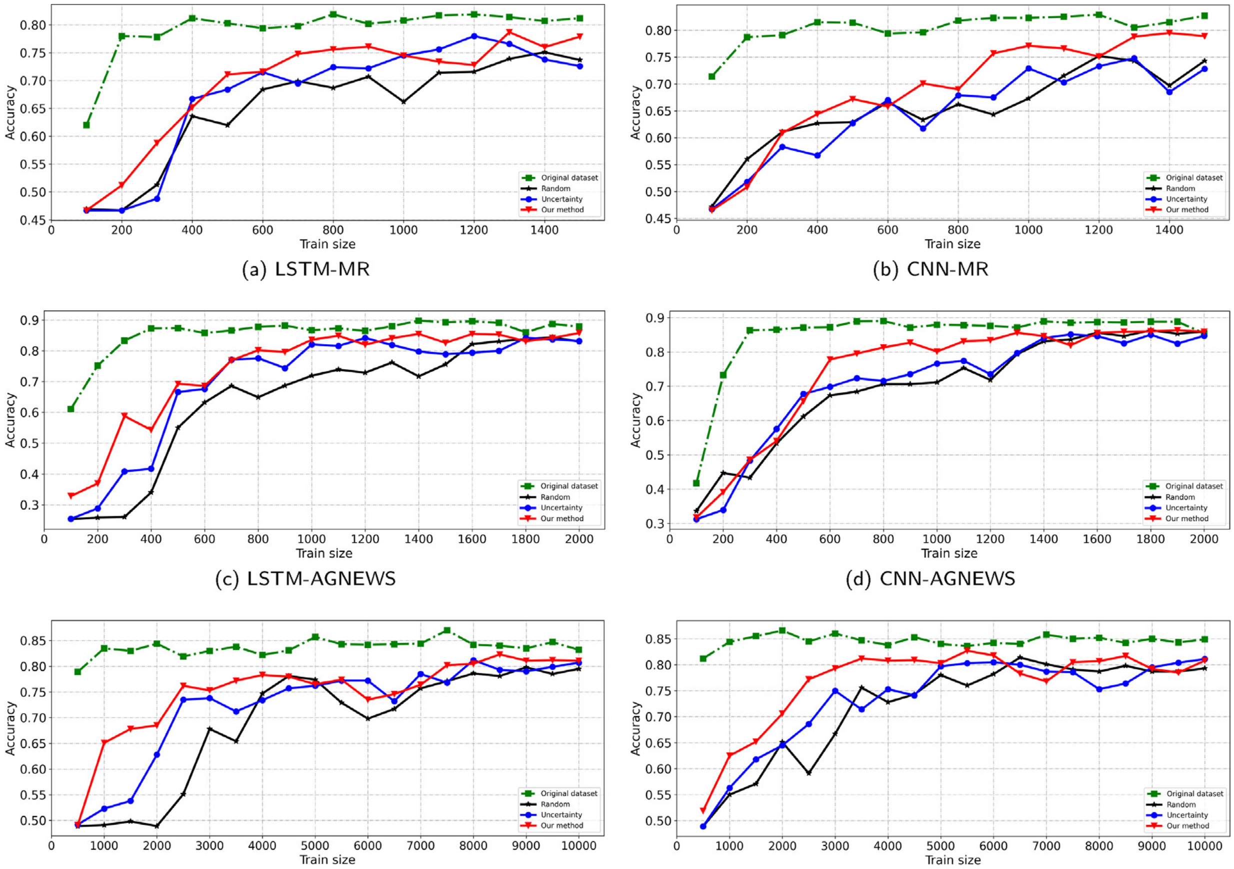
(d) Our method on BERT-AGNEWS

(e) Strategy random on BERT-IMDB

(f) Our method on BERT-IMDB

Fig. 3. Comparison results of entropy distribution on the target models across datasets MR, AGNEWS, and IMDB.

benchmark original data on the datasets MR and AGNEWS. It also performs significantly better than the other two baselines under tight query budget conditions. In particular, our method achieves accuracy of 74.5% and 74.4% on the datasets MR and AGNEWS after the fourth iteration, whereas the baseline random and uncertainty values are only 58.9, 34.0, and 67.6, 45.3, respectively. For the dataset IMDB, the accuracy gap between our method and the other two baselines reached a maximum of 15.6% and 13% after the eighth iteration, respectively. Furthermore, the accuracy of the model extracted using our method always converges faster than baseline random and uncertainty. Notably, The accuracy values sometimes decrease and then become to increase again by increasing training si.e., especially for baseline Random in Figs. 2a and 2c. We believe that the noise in the universal public dataset (WikiText-103 corpus) for fine-tuning substitute model cause this phenomenon. Learning from the non-problem domain public datasets and small training data also exacerbates this phenomenon. We found this phenomenon in other paper (Krishna et al., 2019) using WikiText-103 corpus to extract models too. In summary, our method performs the best in terms of query efficiency when



(a) LSTM-MR

(b) CNN-MR

(c) LSTM-AGNEWS

(d) CNN-AGNEWS

(e) LSTM-IMDB

(f) CNN-IMDB

Fig. 4. Comparison of stealing LSTM and CNN models in terms of accuracy under query budgets.

extracting models with identical architecture using a public corpus.

The metric of agreement is used to evaluate the closeness between the target and substitute models. As shown in Table 3, we have recorded the agreement between the substitute and target models under the given query budgets. The agreement between the model extracted by our method and the target model is better than that by the other methods, except for the case of stealing BERT-IMDB under a query budget of 2000. Our approach remains ahead of other baselines under tight budgets, as do the results of evaluation accuracy. Under moderate budgets, the agreement for BERT-MR and BERT-AGNEWS is better than all other strategies, and the remaining one is close to the original dataset benchmark. This improvement provides solid evidence that our method can select examples from substitute datasets that are informative and more useful for model training.

Uncertainty Fig. 3 compares the uncertainty of the training inputs, our method, and strategy random. After each iteration of model fine-tuning, the entropy values of all training inputs are calculated and saved, which are finally aggregated into a histogram of the entropy distribution. The entropy values of the examples selected by class-balanced uncertainty sampling are significantly larger than the baseline Random. Unsurprisingly, we observe that the entropy distribution of the randomly selected samples is more dispersed, and approximately 40% of the values of the dataset IMDB are concentrated in the range of 0–0.05. A majority of the entropy values chosen by our method are distributed in the higher value interval. The average entropy values of the two methods are 0.67, 1.23, and 0.55, and 0.43, 0.68, and 0.18, against model BERT-MR, BERT-IMDB, and BERT-AGNEWS, respectively. This indicates that our method selects examples with higher uncertainty.

5.1.2. Mismatched architectures

Fig. 4 summarizes the variation in the accuracy of the substitute models. These models have mismatched architectures compared with the target models and are fine-tuned with different inputs. The test results show that, after each iteration of training, higher accuracy is obtained by our method. Namely, for the same given query budget setting, our approach outperforms strategy random and uncertainty sampling in terms of accuracy. Meanwhile, compared with the extraction of models with the same architectures, the results show the same trend of accuracy convergence speed.

The agreement of stealing CNN and LSTM models is shown in Table 4. Overall, the agreement obtained by our approach outperforms that of the other baselines, but not by a significant margin. In addition, we can observe that the agreement of our method is closest to the benchmark, both on tight or moderate budgets. In summary, the results show that attackers can steal text classification models with high accuracy despite the mismatch in architecture between the substitute and victim models. Our method still obtains higher accuracy and agreement than the other baselines with the same query budget, which implies that the query efficiency of our method is better than theirs.

5.2. Adversarial attack

To verify the hazard of adversarial texts generated with the help of the models extracted by our method, we adopt three adversarial

Table 4
Agreement(%) of stealing LSTM and CNN models under tight and moderate budgets.

Strategy	LSTM-MR					LSTM-AGNEWS					LSTM-IMDB				
	Tight budget			Moderate budget		Tight budget			Moderate budget		Tight budget			Moderate budget	
	300	600	900	1200	1500	400	800	1200	1600	2000	2000	4000	6000	8000	10000
Original data	76.7	78.7	79.5	79.6	80.1	87.3	87.8	86.5	89.6	87.9	77.1	77.5	79.5	79.0	77.0
Random	53.8	67.7	68.0	70.1	75.0	39.5	56.7	82.2	81.9	84.0	54.5	70.1	68.3	75.1	75.5
Uncertainty	48.8	71.5	72.2	**78.0**	72.6	40.2	72.6	83.5	81.5	86.5	61.9	71.5	**73.7**	**75.9**	76.1
Our Method	**62.4**	**73.5**	**75.0**	77.8	**78.8**	**46.1**	**83.3**	**86.5**	**83.5**	**86.7**	**62.1**	**72.4**	72.1	74.7	**76.6**
Strategy	**CNN-MR**					**CNN-AGNEWS**					**CNN-IMDB**				
	Tight budget			Moderate budget		Tight budget			Moderate budget		Tight budget			Moderate budget	
	300	600	900	1200	1500	400	800	1200	1600	2000	2000	4000	6000	8000	10000
Original data	74.7	76.6	74.6	78.1	75.1	89.7	91.7	90.4	90.6	90.8	78.4	80.1	79.7	80.2	80.4
Random	61.1	67.6	64.3	73.2	70.3	54.0	72.2	73.6	86.9	87.1	63.8	68.7	75.5	74.4	**77.4**
Uncertainty	58.0	68.1	69.9	72.4	68.8	**58.9**	73.6	75.1	86.5	86.4	60.5	70.6	**76.6**	73.9	76.7
Our Method	**59.5**	**68.6**	**74.2**	**75.5**	**74.4**	56.3	**79.4**	**84.7**	**87.9**	**87.9**	**67.6**	**73.8**	75.4	**75.5**	76.9

Table 5
Transferability of adversarial examples generated by Textfooler, Textbugger, and PWWS.

Victim model	Original accuracy(%)	Accuracy after attack(%)			Attack success rate(%)		
		Textfooler	Textbugger	PWWS	Textfooler	Textbugger	PWWS
BERT-MR	85.3	45.2	45.5	39.1	97.2	76.1	90.7
BERT-AGNEWS	93.4	76.1	77.1	74.1	93.6	72.9	80.5
BERT-IMDB	89.5	60.7	65.7	57.5	98.1	91.3	96.4

attack methods against the corresponding target models. As presented in Table 5, the accuracy of prediction for forged examples decreases dramatically on all three BERT-based classification models. In particular, the accuracy of the BERT-MR model decreases from 85.3% before the attack to 39.1% for the adversarial examples generated by PWWS (i.e., it does not output correct predictions for more than half of the given samples). The accuracy of models BERT-IMDB and BERT-AGNEWS also decreased by approximately 30% and 20%, respectively. The success rate of the adversarial attack against the substitute model is also relatively high, implying that the generation of adversarial samples is less restricted by the original texts. Notably, it is no longer necessary to raise any queries to generate adversarial examples and transfer them to the target model after obtaining a substitute model.

6. Conclusions

In this study, we present a query-efficient model extraction strategy in a realistic and practical setting in which the only information that attackers can access from the victim model is the top-1 predictions. We show that the active learning selection strategy, incorporating semantic-based diversity and class-balanced uncertainty sampling, can select informative and useful examples from large-scale and public corpora. We perform our method to extract models with different architectures in three text classification tasks. The experiment shows that for the same given query budgets, the substitute models obtained by fine-tuning with these informative examples outperform models trained by examples selected randomly or the max-entropy principle in terms of accuracy. Our strategy uses fewer queries to fine-tune the substitute model to a certain accuracy than the baselines. Moreover, under moderate query budgets, the agreement and accuracy of the substitute model retrained by our strategy are close to those of a model trained by problem domain data. Finally, we show that the accuracy of the BERT-based classification models decreases significantly when predicting adversarial texts generated against the corresponding substitute models.

CRediT authorship contribution statement

Hao Peng: Funding acquisition, Methodology, Conceptualization, Project administration. **Shixin Guo:** Software, Validation, Formal analysis. **Dandan Zhao:** Writing – review & editing, Supervision. **Yiming Wu:** Conceptualization. **Jianming Han:** Investigation. **Zhe Wang:** Investigation. **Shouling Ji:** Conceptualization. **Ming Zhong:** Writing – original draft.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknolwedgement

This work was partly supported by the National Natural Science Foundation of China under Grant nos. 62072412, 61902359, 61702148, and 61672468, the Opening Project of Shanghai Key Laboratory of Integrated Administration Technologies for Information Security under Grant AGK2018001, and the Key Lab of Information Network Security, Ministry of Public Security, under Grant no. C20607, the National Key Research and Development Program of China under Grant no. 2019YFC01188.

References

Abid, A., 2022. Huggingface datasets. https://github.com/huggingface/datasets.

Cer, D., Yang, Y., Kong, S.Y., Hua, N., Limtiaco, N., John, R.S., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., Strope, B., Kurzweil, R., 2018. Universal sentence encoder for english. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp. 169–174. https://doi.org/10.18653/v1/D18-2029.

Chandrasekaran, V., Chaudhuri, K., Giacomelli, I., Jha, S., Yan, S., 2020. Exploring connections between active learning and model extraction. In: 29th USENIX Security Symposium (USENIX Security 20), pp. 1309–1326.

Daniel, C., Yinfei, Y., 2022. Universal sentence encoder. https://tfhub.dev/google/universal-sentence-encoder/5.

Devlin, J., Chang, M.W., Lee, K., Toutanova, K., 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. north american chapter of the association for computational linguistics.

Fredrikson, M., Jha, S., Ristenpart, T., 2015. Model inversion attacks that exploit confidence information and basic countermeasures. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, Colorado, USA. pp. 1322–1333. https://doi.org/10.1145/2810103.2813677.

Gal, Y., Ghahramani, Z., 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In: International Conference on Machine Learning, Hamilton, New Zealand., pp. 1050–1059.

Gissin, D., Shalev-Shwartz, S., 2019. Discriminative active learning. arXiv preprint arXiv:1907.06347.

Gong, X., Wang, Q., Chen, Y., Yang, W., Jiang, X., 2020. Model extraction attacks and defenses on cloud-based machine learning models. IEEE Commun. Mag. 58, 83–89.

Goodfellow, I.J., Shlens, J., Szegedy, C., 2014. Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572.

Hinton, G., Vinyals, O., Dean, J., 2015. Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 2.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9, 1735–1780.

Huang, J., Child, R., Rao, V., Liu, H., Satheesh, S., Coates, A., 2016. Active learning for speech recognition: the power of gradients. arXiv preprint arXiv:1612.03226.

Japkowicz, N., Stephen, S., 2002. The class imbalance problem: A systematic study. Intell. Data Anal. 6, 429–449.

Jin, D., Jin, Z., Zhou, J.T., Szolovits, P., 2020. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 8018–8025. https://doi.org/10.1609/aaai.v34i05.6311.

Kariyappa, S., Prakash, A., Qureshi, M.K., 2021. Maze: Data-free model stealing attack using zeroth-order gradient estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 13814–13823. https://doi.org/10.1109/CVPR46437.2021.01360.

Kim, Y., 2014. Convolutional neural networks for sentence classification. EMNLP, 1746–1751.

Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Krishna, K., Tomar, G.S., Parikh, A.P., Papernot, N., Iyyer, M., 2019. Thieves on sesame street! model extraction of bert-based apis. arXiv preprint arXiv:1910.12366.

Lewis, D.D., 1995. A sequential algorithm for training text classifiers: Corrigendum and additional data. In: Acm Sigir Forum. ACM New York, NY, USA, pp. 13–19.

Li, J., Ji, S., Du, T., Li, B., Wang, T., 2018. Textbugger: Generating adversarial text against real-world applications. arXiv preprint arXiv:1812.05271.

Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., Potts, C., 2011. Learning word vectors for sentiment analysis. Association for Computational Linguistics, Portland, Oregon, USA, pp. 142–150.

Merity, S., 2022. Wikitext-103 dataset download website. https://blog.salesforceairesearch.com/the-wikitext-long-term-dependency-language-modeling-dataset.

Merity, S., Xiong, C., Bradbury, J., Socher, R., 2016. Pointer sentinel mixture models. arXiv preprint arXiv:1609.07843.

Morris, J., 2022. Huggingface of textattack. https://huggingface.co/textattack.

Morris, J.X., Lifland, E., Yoo, J.Y., Grigsby, J., Jin, D., Qi, Y., 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. arXiv preprint arXiv:2005.05909.

Pal, S., Gupta, Y., Shukla, A., Kanade, A., Shevade, S., Ganapathy, V., 2020. Activethief: Model extraction using active learning and unannotated public data. In: Proceedings of the AAAI Conference on Artificial Intelligence, New York, USA. pp. 865–872. https://doi.org/10.1609/aaai.v34i01.5432.

Pang, B., Lee, L., 2005. Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales. arXiv preprint cs/0506075.

Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z.B., Swami, A., 2017. Practical black-box attacks against machine learning. In: Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, pp. 506–519. https://doi.org/10.1145/3052973.3053009.

Qiu, X., Sun, T., Xu, Y., Shao, Y., Dai, N., Huang, X., 2020. Pre-trained models for natural language processing: A survey. Sci. China Technol. Sci. 63, 1872–1897.

Ren, S., Deng, Y., He, K., Che, W., 2019. Generating natural language adversarial examples through probability weighted word saliency. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy. pp. 1085–1097. https://doi.org/10.18653/v1/P19-1103.

Ribeiro, M.T., Singh, S., Guestrin, C., 2018. Semantically equivalent adversarial rules for debugging NLP models. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Melbourne, Australia. pp. 856–865. https://doi.org/10.18653/v1/P18-1079.

Sanyal, S., Addepalli, S., Babu, R.V., 2022. Towards data-free model stealing in a hard label setting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA. pp. 15284–15293. https://doi.org/10.48550/arXiv.2204.11022.

Schröder, C., Niekler, A., 2020. A survey of active learning for text classification using deep neural networks. arXiv preprint arXiv:2008.07267.

Sener, O., Savarese, S., 2017. Active learning for convolutional neural networks: A core-set approach. arXiv preprint arXiv:1708.00489.

Shokri, R., Stronati, M., Song, C., Shmatikov, V., 2017. Membership inference attacks against machine learning models. In: 2017 IEEE symposium on security and privacy (SP). IEEE, San Jose, CA, USA, pp. 3–18.

Sylla, T., Chalouf, M.A., Krief, F., Samaké, K., 2021. Context-aware security in the internet of things: a survey. Int. J. Auton. Adaptive Commun. Syst. 14, 231–263.

Tramèr, F., Zhang, F., Juels, A., Reiter, M.K., Ristenpart, T., 2016. Stealing machine learning models via prediction {APIs}. In: 25th USENIX Security Symposium (USENIX Security 16), pp. 601–618.

Truong, J.B., Maini, P., Walls, R.J., Papernot, N., 2021. Data-free model extraction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4771–4780. https://doi.org/10.1109/CVPR46437.2021.00474.

Wallace, E., Feng, S., Kandpal, N., Gardner, M., Singh, S., 2019. Universal adversarial triggers for attacking and analyzing nlp. arXiv preprint arXiv:1908.07125.

Wallace, E., Stern, M., Song, D., 2020. Imitation attacks and defenses for black-box machine translation systems. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 5531–5546. https://doi.org/10.18653/v1/2020.emnlp-main.446.

Wang, W., Yin, B., Yao, T., Zhang, L., Fu, Y., Ding, S., Li, J., Huang, F., Xue, X., 2021. Delving into data: Effectively substitute training for black-box attack. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA. pp. 4761–4770. https://doi.org/10.1109/CVPR46437.2021.00473.

Yu, H., Yang, K., Zhang, T., Tsai, Y.Y., Ho, T.Y., Jin, Y., 2020. Cloudleak: Large-scale deep learning models stealing through adversarial examples. In: Network and Distributed System Security Symposium.

Yuan, L., Zheng, X., Zhou, Y., Hsieh, C.J., Chang, K.W., 2021. On the transferability of adversarial attacks against neural text classifier. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pp. 1612–1625. https://doi.org/10.18653/v1/2021.emnlp-main.121.

Zhang, X., Zhao, J., LeCun, Y., 2015. Character-level convolutional networks for text classification. Palais des Congrès de Montréal, Montréal CANADA, pp. 649–657.

Zhou, W., Hou, X., Chen, Y., Tang, M., Huang, X., Gan, X., Yang, Y., 2018. Transferable adversarial perturbations. In: Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany. pp. 452–467. https://doi.org/10.1007/978-3-030-01264-9_28.