TEXTKNOCKOFF: KNOCKOFF NETS FOR STEALING FUNCTIONALITY OF TEXT SENTIMENT MODELS

Xuan Cong Pham[1], *Trung Nguyen Hoang*[2], *Cao Truong Tran*[2], *Viet Binh Do*[1*]

Abstract

Most commercial machine learning models today are designed to require significant amounts of time, money, and human effort. Therefore, intrinsic information about the model (such as architecture, hyperparameters, and training data) needs to be kept confidential. These models are referred to as black boxes, and there is an increasing amount of research focused on both attacking and protecting them. Recent publications have often concentrated on the field of computer vision; in contrast, there is still relatively little research on methods for attacking black box models with textual data. This article introduces a research method for extracting the functionality of a black box model in the task of text sentiment analysis. The method has been effectively tested based on random sampling techniques to reconstruct a new model with equivalent functionality to the original model, achieving high accuracy (94.46% compared to 94.92%) and high similarity (96.82%).

Index terms

Text classification, text sentiment, black-box model stealing, knockoff model.

1. Introduction

In recent years, there has been an unprecedented surge of interest in AI across various industries. This surge can largely be attributed to groundbreaking research breakthroughs, notably in deep learning, which have sparked excitement and innovation. As a result, we have seen a growing number of companies integrating neural models into their products and services, recognizing the immense potential of AI technologies to drive efficiency and enhance user experiences. These neural models, powered by advanced algorithms and data-driven insights, have become indispensable tools for businesses looking to stay competitive in today's rapidly evolving market. Whether it is improving personalized recommendations, optimizing supply chain management, or revolutionizing customer service, the adoption of neural models is reshaping industries and transforming the way we work and interact with technology [1], [2].

[1]Military Institute of Information Technology, 17 Hoang Sam, Nghia Do, Cau Giay, Ha Noi
[2]Institute of Information and Communication Technology, Le Quy Don Technical University
[*]Corresponding author email: binhdv@gmail.com, DOI: 10.56651/lqdtu.jst.v13.n01.821.ict

To safeguard their valuable intellectual property, companies often employ a strategy of confidentiality when it comes to their neural models. This means they keep important information about how these models work a secret, treating them like black boxes that only reveal inputs and outputs while hiding the inner workings. This includes keeping details like the model's structure and the data used to train it hidden from view. By maintaining this level of confidentiality, companies aim to prevent others from copying their ideas or technologies. However, while confidentiality is crucial for protecting intellectual property, it also presents challenges in terms of transparency and accountability in AI systems. Despite these challenges, companies continue to prioritize the protection of their intellectual property while exploring ways to balance confidentiality with transparency to uphold ethical standards in AI development [3], [4].

However, recent researches have revealed vulnerabilities in these black-box neural models. Despite companies' efforts to maintain security, addressing or patching these vulnerabilities can potentially undermine the core functionality of these models. This raises significant concerns about the security and integrity of proprietary AI systems, as well as the risk of intellectual property theft. Additionally, it underscores the need for companies to reassess their security measures and invest in robust protections to safeguard their machine learning models from unauthorized access and exploitation. Addressing these vulnerabilities is crucial to ensure the continued reliability and effectiveness of AI systems in various applications [3]–[5].

In recent years, there has been a considerable amount of research focusing on the vulnerability of black-box neural models in computer vision tasks, particularly in the context of stealing functions. These studies have shed light on various methods and techniques that can be employed to extract valuable information from black-box models, potentially compromising their functionality [4], [5]. However, in contrast to the extensive research conducted in computer vision, there has been relatively less emphasis on exploring similar vulnerabilities in NLP tasks. As NLP applications continue to gain prominence in various industries, addressing this gap becomes increasingly crucial to ensure the robustness and integrity of AI systems in real-world scenarios. Therefore, there is a pressing need for further research and exploration into the vulnerabilities of black-box neural models in NLP tasks, as well as the development of effective countermeasures to mitigate potential risks.

Text sentiment refers to the analysis of emotions or opinions expressed in text, such as positive, negative, or neutral sentiments. This analysis plays a crucial role in various applications across industries. For instance, in customer service, sentiment analysis helps companies gauge customer satisfaction and identify areas for improvement based on feedback. In marketing, it assists in understanding consumer sentiment towards products or brands, guiding advertising strategies. Moreover, in social media monitoring, sentiment analysis enables businesses to track public perception, identify trends, and manage their online reputation effectively. Additionally, sentiment analysis is utilized in financial markets to analyze news articles, social media posts, and other textual data for predicting market trends and making investment decisions [6], [7].

Black-box neural models utilized for text sentiment analysis can also be susceptible to function stealing, similar to the broader field of NLP, which has seen limited research on this topic. Recognizing the importance of addressing this gap, this article aims to investigate function stealing in the context of text sentiment analysis. Specifically, in this article, we will propose a novel approach to train surrogate models that closely approximate the behavior of black-box neural models for text sentiment analysis. These surrogate models, or knockoff models, will be engineered to capture the underlying patterns and decision-making processes of the original black-box models. By developing such surrogate models, we aim to shed light on the inner workings of black-box neural models and mitigate the risks associated with function stealing. Through extensive experimentation and validation, we will demonstrate the effectiveness of our proposed approach in creating knockoff models for text sentiment analysis.

The article is structured as follows: Section 2 provides an overview of related work, while Section 3 outlines the proposed method. Sections 4 and 5 detail the experimental design, the results and the discussion. Finally, Section 6 concludes the article and highlights future work.

2. Related work

In this section, we will discuss research related to black-box model extraction, knowledge distillation, and text sentiment analysis.

2.1. Stealing machine learning models

Model stealing is a type of attack in machine learning where an knockoff attempts to infer or replicate the functionality of a target model by observing its outputs. This attack is particularly concerning in scenarios where the target model is deployed as a service or accessible via an API, making it vulnerable to reverse engineering. The process of model stealing typically involves several steps: querying the target model, building a shadow model, training the knockoff model, and evaluating the knockoff model [8], [9].

In the first step, known as querying the target model, the knockoff sends input queries to the target model and observes its outputs. These queries could be specific data points or a series of queries to explore the behavior of the model. Subsequently, in the building a shadow model step, based on the observed outputs from the target model, the knockoff constructs a "shadow" model that aims to mimic the behavior of the target model. This shadow model may not perfectly replicate the target model but is designed to approximate its functionality. Following this, the knockoff collects data and outputs from the target model to train their own model, known as the knockoff model. This model is trained to imitate the behavior of the target model as closely as possible. Finally, once the knockoff model is trained, it undergoes evaluation to assess its performance in replicating the target model's functionality. This evaluation may involve comparing the outputs of the knockoff model with those of the target model on a separate test dataset [8], [9].

In the study [10], two types of attacks are discussed based on the available knowledge of the model: white-box and black-box attacks. In a white-box attack, the attacker has extensive knowledge of the model, including its structure, parameters, and training dataset. This allows for a thorough understanding of the impact of adversarial alterations caused by internal parameters. However, in practical real-world scenarios, our knowledge of the model is often limited, making this type of attack more suitable for academic research aimed at improving our understanding of the model. Nonetheless, this framework enables a worst-case assessment of learning algorithm security, providing empirical upper limits on potential performance degradation under attack.

In black-box attacks, the attacker lacks knowledge about the model. Various methods have been proposed for attacking black-box models. Poisoning attacks [11] involve manipulating training data, while evasion attacks [12] alter original data to generate imperceptible adversarial examples. Membership inference attacks aim to determine if a sample is in the training data [13]. Model stealing, or model extraction, compromises a model's intellectual property by divulging its parameters or emulating its behavior. This work focuses on investigating model stealing techniques, such as Copycat CNN [5], Knock-off Nets [4], and Functionally Equivalent Extraction Attack [14].

Typically, extracting functions from black-box models involves two phases: creating a transfer-set by selecting data through the victim to produce fake labels, and then training the knockoff with this transfer-set. Since the victim's training dataset is unknown, creating a new equivalent model with similar accuracy usually involves two approaches: improving data sampling strategies or generating data based on some truth data. The first approach involves selecting a large dataset and employing various sampling strategies on input. For instance, [4] proposed random or adaptive data selection strategies, while [15] suggested building a prediction army to guide the next choice. [16] also explored sample selection options such as uncertainty strategy, k-center algorithm, or DeepFool-based active learning, showing promising results in image classification problems.

2.2. Knowledge distillation

Knowledge distillation, a widely-used technique in machine learning, involves training a smaller model, termed the student model, to emulate the intricate behavior of a larger, more complex model known as the teacher model. The essence of this process lies in transferring the learned knowledge and insights acquired by the teacher model to the relatively compact student model. By doing so, the student model aims to achieve performance levels that are on par with, or close to, those of the teacher model. This transfer of knowledge enables the student model to effectively capture the underlying patterns and nuances present in the data, albeit with reduced computational resources. The ultimate objective of knowledge distillation is to leverage the expertise encapsulated within the teacher model to enhance the efficiency and effectiveness of the student model in various machine learning tasks. Through this approach, practitioners can strike a balance between model complexity and computational overhead, thereby facilitating the deployment of more streamlined and scalable machine learning solutions [17]–[19].

Knowledge distillation operates on the principle of leveraging the outputs generated by the teacher model, typically soft probabilities or logits, as guiding targets during the training of the student model. This process unfolds through a series of steps, beginning with the training of the teacher model on a designated dataset using conventional supervised learning methods. However, rather than employing hard labels, represented as one-hot encoded vectors, as the standard targets for training, the student model endeavors to glean insights from the nuanced, more informative outputs produced by the teacher model. By doing so, the student model not only assimilates the final predictions but also captures the inherent uncertainty and intricate details encapsulated within the teacher's predictions. This nuanced approach enables the student model to refine its learning process, drawing upon the wealth of knowledge encoded within the teacher model's outputs to enhance its own predictive capabilities [19].

The process of knowledge distillation typically unfolds in two main steps: Teacher model training and student model training. In the first step, a large and complex model, referred to as the teacher, undergoes training on a designated dataset using conventional techniques such as supervised learning. This teacher model is usually deeper and more computationally intensive compared to the subsequent student model. In the subsequent step, the student model, which is smaller and simpler in structure, is trained on the same dataset. However, instead of utilizing the original labels as targets, the student model learns from the softened probabilities or logits generated by the teacher model. By minimizing the disparity between its predictions and those of the teacher model, the student model endeavors to replicate the behavior and performance of the teacher model. Through this process, knowledge distillation enables the transfer of knowledge from the teacher to the student, facilitating the achievement of comparable performance with reduced computational complexity [19].

2.3. Text sentiment analysis

Text sentiment analysis is a crucial aspect of NLP, focusing on deciphering the emotions, opinions, or attitudes conveyed in textual data. It involves the classification of text into predefined sentiment categories, such as positive, negative, or neutral. This analysis enables organizations to gauge public sentiment towards their products, services, or brands, allowing them to make informed decisions and tailor their strategies accordingly. Moreover, sentiment analysis plays a vital role in various domains, including marketing, customer service, and social media monitoring, where understanding the sentiment of users' feedback is essential for maintaining customer satisfaction and managing brand reputation [6], [7].

Traditionally, machine learning approaches were employed for text sentiment analysis, where features were manually engineered from the text, and models like Support Vector Machines (SVM) or Naive Bayes classifiers were trained on these features. These methods often relied on handcrafted features such as Bag-of-Words or TF-IDF representations, limiting their ability to capture nuanced relationships in text data [20]. However, with the advent of deep learning, more sophisticated models have been developed for

sentiment analysis. Convolutional Neural Networks and Recurrent Neural Networks have shown remarkable success in automatically learning features from raw text data, enabling them to capture complex patterns and dependencies [21].

In recent years, transformer-based models have revolutionized the field of NLP, including text sentiment analysis. Transformers, such as BERT (Bidirectional Encoder Representations from Transformers), have achieved state-of-the-art performance by leveraging self-attention mechanisms to capture contextual information from text sequences [22]. Distilbert [23] is a distilled version of BERT that retains much of the performance while being smaller and faster. It is suitable for resource-constrained environments where computational resources are limited. Roberta is an optimized version of BERT that incorporates additional pre-training data and optimization techniques, resulting in improved performance on various tasks [24] . XLNet is another transformer-based model that uses a permutation-based training approach to capture bidirectional context [25]. Unlike traditional models that process input sequentially, transformers can consider the entire input simultaneously, allowing them to capture long-range dependencies more effectively. This ability to capture context at scale has significantly improved sentiment analysis tasks, enabling models to better understand the nuances and subtleties of human language and sentiment expression [22].

3. The proposed method

This article's overall objective is to present a methodology for constructing knockoff models derived from black-box neural models tailored for text sentiment analysis. The problem at hand involves a black-box model, referred to as the victim, designed for text sentiment analysis, whose architecture and training dataset are unknown. This victim model receives text inputs and predicts whether the output sentiment is positive, negative, or neutral. The primary inquiry centers on devising a training approach to develop a knockoff model that mimics the functionality of the victim model with comparable accuracy and optimal similarity.

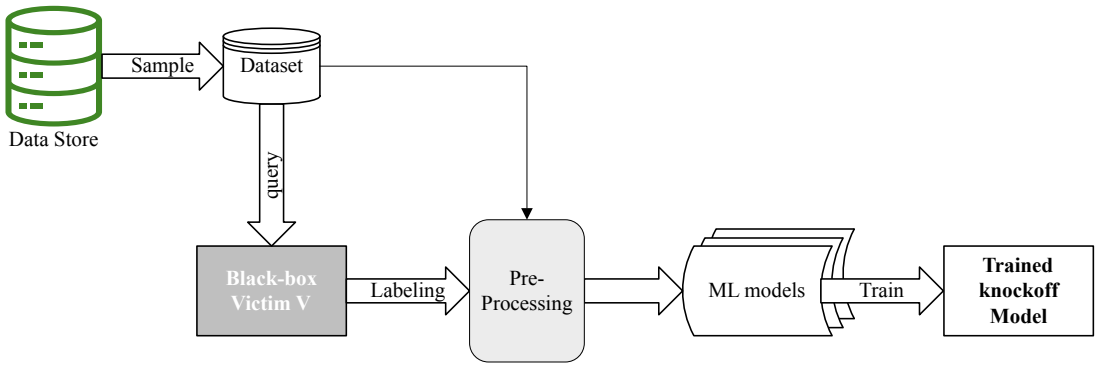


Fig. 1. Text sentiment analysis model extraction process.

Algorithm 1: Text sentiment model extraction process

Input:
$\mathbb{D}$, text sentiment datasets
B, the query budget of data
F_V, the text sentiment victim model V
$\mathbb{F}$, the sets of machine learning models
Output:
$F_A^t \in \mathbb{F}$, the trained knockoff model A
Function *ModelExtraction*(F_V, B)**:**
$X_V \leftarrow$ randomly select ***B*** queries from $\mathbb{D}$
$D \leftarrow \{(x, y) : x \in X_V, y = F_V(x)\}$ ▷ query X_V to F_V making transfer-set D
$X_A \leftarrow$ pre-processing $D = \{(x, y) : x \in X_V, y = F_V(x)\}$
$F_A \leftarrow$ select from $\mathbb{F}$ ▷ selecting a suitable machine learning model
$F_A^t \leftarrow$ train F_A on X_A
return F_A^t

The process of extracting the text sentiment analysis model is illustrated in Fig. 1 and Algorithm 1, consisting of the following main 4 steps.

Step 1: Firstly, a meticulous selection process is undertaken from a comprehensive collection of datasets relevant to text sentiment analysis to curate a subset of data samples. These samples are carefully chosen to represent a diverse range of textual inputs, encompassing various linguistic styles, topics, and sentiment expressions. This curated subset forms the set X_V, serving as the foundation for subsequent analysis and model training.

Step 2: With the set X_V established, each input x within this set undergoes a querying process into the victim model F_V. This querying process elicits corresponding predictions y, which are essentially the model's output for each input text. These predictions, often termed as "pseudo-labels," effectively represent the sentiment analysis outcomes generated by the victim model for the given inputs. Collectively, the input-output pairs (x, y) constitute what is referred to as the transfer-set.

Step 3: Following the generation of the transfer-set, a critical pre-processing stage ensues to prepare the data for training the knockoff model. This pre-processing involves organizing the transfer-set into a structured format suitable for machine learning training. Specifically, the transfer-set is formatted into a training dataset denoted as $X_A = (x, y)|y = F_V(x)$. Here, only the input-output pairs where the predicted sentiment matches the actual sentiment are retained, ensuring that the training data accurately reflects the behavior of the victim model.

Step 4: With the training dataset X_A prepared, the next step involves selecting a suitable machine learning model to serve as the knockoff, denoted as F_A. This model

is tasked with learning from the transfer-set and emulating the behavior of the victim model. Subsequently, F_A undergoes a training process utilizing the input-output pairs from X_A, aiming to capture the underlying patterns and decision-making processes exhibited by the victim model. Upon completion of the training process, the resulting model F_A represents the trained knockoff model, capable of replicating the text sentiment analysis functionality of the victim model to a significant extent.

We evaluate the effectiveness of the model's function imitation in two aspects: the model's accuracy compared to victim using the F1-score measure, the model's similarity based on the Agreement measure. To ensure that the extracted model has similar functionality to the victim model, we must evaluate the similarity in the predictions of the two models on the same test set of V:

$$Agreement_{VA}(D_V^{test}) = \frac{1}{|D_V^{test}|} \sum_{x \in D_V^{test}} count(F_A(x) = F_V(x)) \quad (1)$$

Here, the *count()* function is an indicator function, $|D_V^{test}|$ returning the total number of records in the set D_V^{test}. The higher value of *Agreement()* shows that F_A is more similar in function to F_V.

4. Experimental design

4.1. Datasets

Table 1. The text sentiment datasets used in the experiment

Datasets	Class	Total	Train-set	Test-set	Note
IMDB	2	50,000	25,000	25,000	IMDB dataset having 50K movie reviews for natural language processing or Text analytics.
SST2	2	70,042	67,349	1,821	SST2 is based on the dataset consists of 11,855 single sentences extracted from movie reviews. It was parsed with the Stanford parser and includes a total of 215,154 unique phrases from those parse trees, each annotated by 3 human judges.
TWEET	3	59,900	45,600	12,300	Twitter consists of 7 heterogenous tasks in Twitter, all framed as multiclass tweet classification. The text sentiment tasks are included with fixed training, validation, and test splits.

In the experiments, we performed experiments on multiple datasets, including IMDB review 50K (IMDB) [1] [26], SST2 (SST2) [2] [27] and Twitter (TWEET) [3] [28]. Table 1

[1]https://huggingface.co/datasets/ajaykarthick/imdb-movie-reviews
[2]https://huggingface.co/datasets/stanfordnlp/sst2
[3]https://huggingface.co/datasets/carblacac/twitter-sentiment-analysis

shows some information about those datasets. Each sentiment dataset class is split into two subsets, for training and testing respectively. Test dataset is used to evaluate the performance of the model, and the model has not seen these data during training.

4.2. Experiments setup

In this implementation method, careful consideration is required regarding dataset selection for sampling inputs via the victim model when creating the transfer-set for knockoff training. This process presents three distinct scenarios: the *KD-case*, where the knockoff's dataset overlaps with the victim's dataset, resembling a typical Knowledge Distillation process; the *closed-case*, where the victim's dataset is a subset of the knockoff's dataset, formed from various sets including the victim's dataset; and the *open-case*, where the victim's dataset is largely unrelated to the knockoff's dataset. Each scenario demands tailored approaches to ensure effective knockoff model training.

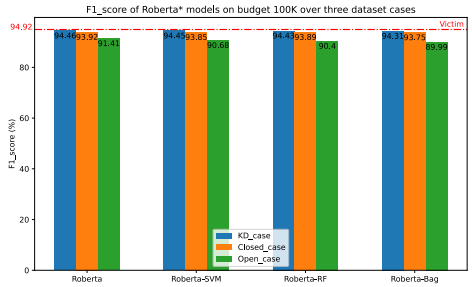
The experiments use XLNet-base as the victim model, trained on the IMDB dataset. Although this choice is specific, different models can be chosen for other datasets. For knockoff models, we utilize two learning models, Roberta base and Distilbert base uncased, from the HuggingFace library for feature extraction. Various machine learning methods are employed for text classification, including regression, SVM, Random Forest, and Bagging. Roberta and Distilbert are configured with default settings, while other ML methods use default configurations from the Sklearn library.

5. Results and discussion

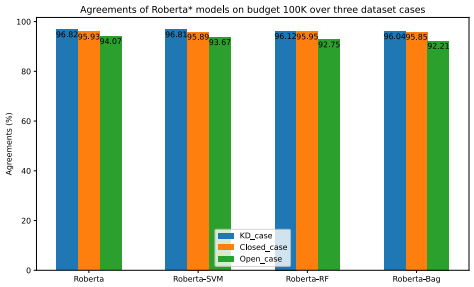
In the Appendix, we show experimental results of different models in sentiment analysis tasks. Table 2 and Fig. 5 show the performance testing results of three models (Distilbert, Roberta and XLNet) on the three datasets. Table 3 shows the comparison of inference time of victim and 8 knockoff models, while table 4 and table 5 show the results of knockoff models with unfrozen BERT layers during the feature extraction phase, while table 7 and table 8 show the results of knockoff models with frozen BERT layers. Furthermore, table 6 and table 9 present the running time (in seconds) for both the unfrozen and frozen cases, respectively. Each table contains eight models with the same feature extraction phase using deep learning (Roberta or Distilbert), differing only in the final classification layer. The initial two models utilize regression, followed by SVM, RF, and Bagging in subsequent models for text classification.

A. RQ1: What is the effect of dataset selection on the quality of knockoff model?

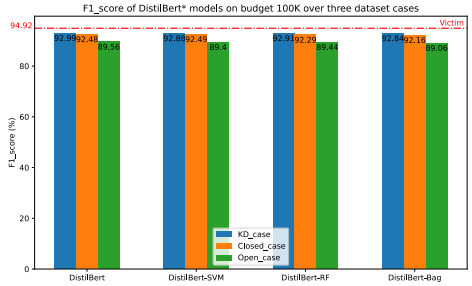
In Fig. 2, the performance of knockoff models is depicted using different feature extraction phases with a sampling budget of 100K. Specifically, figures 2a and 2b utilize feature extraction via Roberta, while figures 2c and 2d employ Distilbert. The left side of the graph compares the F1-scores of the models with the F1-score of the victim (represented by the red dashed line), while the right side illustrates the corresponding agreement. Each model is plotted with three data scenarios: *KD-case* (blue), *closed-case* (orange), and *open-case* (green).



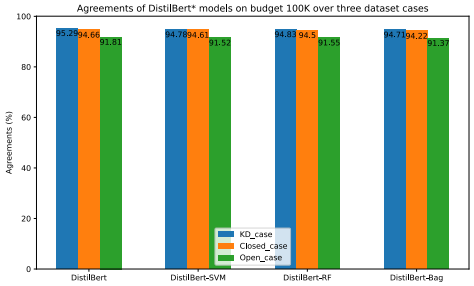
*(a) F1-scores of Roberta**



*(b) Agreements of Roberta**



*(c) F1-scores of Distilbert**



*(d) Agreements of Distilbert**

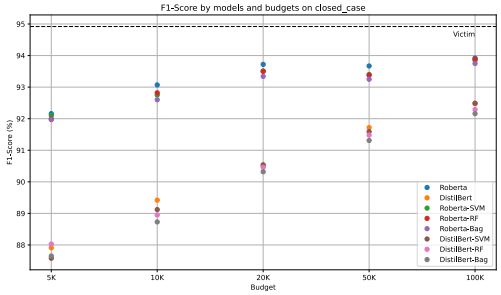
Fig. 2. Performance of knockoff models on budget 100K.

The knockoff model demonstrates the highest accuracy and similarity in the *KD-case* scenario, where there is an overlap of data with the victim's training set, followed by the *closed-case* scenario and the worst-case scenario where data related to the victim's training set is inaccessible. For example, in Fig. 2(a) and Fig. 2(b), representing models using feature extraction from Roberta, the *KD-case* scenario (first leftmost blue column) exhibits the best performance with an F1-score of 94.46%, closely approaching the victim's F1-score (94.92%) with an Agreement of 96.82%. Here, most knockoff models achieve accuracy levels close to the victim's (at least 94.31%) with very high similarity rates (at least 96.04%). In contrast, in the *closed-case* scenario (orange columns), knockoff models achieve slightly lower accuracy (0.5 to 0.6% lower than the *KD-case* scenario on the same model) while maintaining very high similarity. Finally, in the *open-case* scenario (green columns), model accuracy decreases by approximately 3%, accompanied by a corresponding decrease in similarity of about 2%.

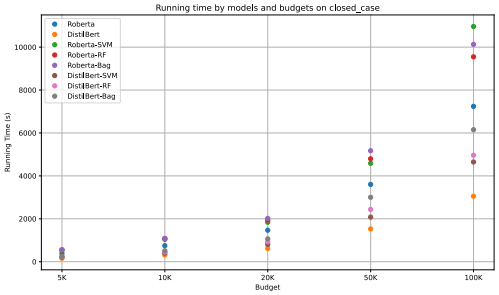
In summary, the experiments reveal that when the data overlap with the victim's training set (*KD-case* and *closed-case*), the imitation effect is significantly better, closely approaching the accuracy of the victim. In contrast, in *open-case*, the effectiveness of the simulations is notably poorer.

B. RQ2: How does the budget size when sampling data affect the quality of the knockoff model?

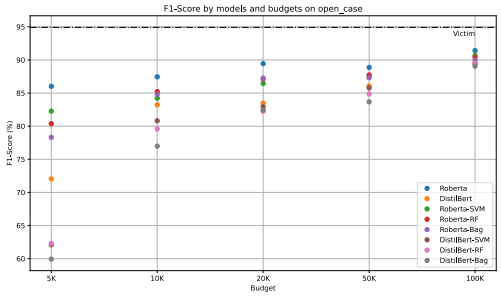
In practical scenarios, querying data from a victim model can be resource-intensive. Hence, the crucial question arises: can we replicate the model's functionality with a reasonable number of data queries? Fig. 3a and Fig. 3c depict accuracy and runtime charts for two scenarios - *closed-case* and *open-case* - at various budget levels from 5K to 100K. Each graph illustrates the performance of 8 models derived from tables 4 and 6. On the left side, accuracy across different budgets is displayed, comparing the models' performance with the F1-score of the victim (black dashed line). On the right side, the corresponding execution time is illustrated in seconds.



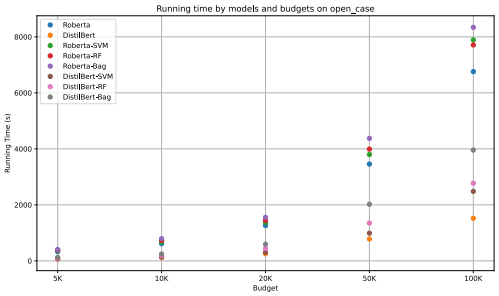
(a) F1-scores in closed-case



(b) Run-time in closed-case



(c) F1-scores in open-case



(d) Run-time in open-case

Fig. 3. Performance of knockoff models on various budgets in closed-case and open-case.

From Fig. 3, it is clear that in the *closed-case* scenario, models achieve relatively high performance even at low budget levels and stabilize at an optimal budget. In the *open-case* scenario, the models show varying performances at low budgets but converge at higher budgets. For instance, in Fig. 3a, Roberta models maintain consistent accuracy from the 20K to 100K budget (93.72% to 93.92%), while Distilbert accuracy increases from around 90.5% to over 92%. Generally, larger budgets lead to higher knockoff model accuracy. However, in the *open-case* scenario's third dataset, both models perform similarly poorly compared to the victim, with nearly identical accuracies even at higher budgets (F1-score $\approx 90.4 \pm 1.0\%$ at 100K).

Moreover, it is evident that execution time increases proportionally with data budget, leading to longer execution times with higher data budgets. Additionally, execution times vary depending on the feature extraction framework used, especially noticeable in the *open-case* scenario. For example, in Fig. 3b, at 100K, Roberta-Bag (8340.0 s) takes

approximately twice the time of Distilbert-Bag (3957.8 s). Similarly, in Fig. 3d, when transitioning from a budget of 50K to 100K, the runtime of Distilbert (from 2000 s to about 4000 s) grows more gradually than that of Roberta (from 4000 s to over 6000 s).

In summary, when considering the appropriate budget, the performance generally improves with larger budgets, albeit at the expense of longer execution times. However, if there is a cost associated with querying the victim model, a budget level of around 20K queries appears to strike a balance between performance and cost-effectiveness.

C. RQ3: What is the impact of the transfer learning on the quality of knockoff model?

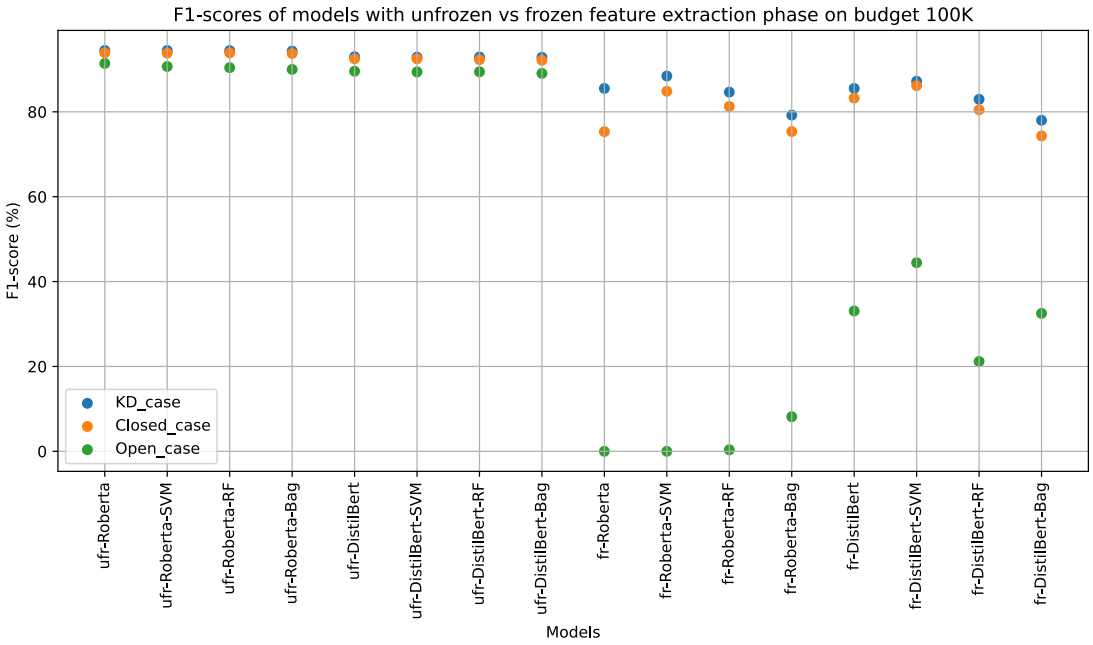


Fig. 4. F1-scores of models with unfrozen vs frozen feature extraction phase on budget 100K.

We also explore the impact of transfer learning process on mimic model performance by conducting experiments with 8 test models, each comprising two phases: feature extraction and text classification. We evaluated performance under two conditions: full retraining (unfrozen) and freezing the feature extraction phase (frozen), training only the text classification layer. Fig. 4 presents the performance chart of these 8 models across 3 data scenarios (*KD-case* in blue, *closed-case* in orange, and *open-case* in green) on a budget of 100K. Models on the left (with prefix "ufr") are fully retrained, while models on the right (with prefix "fr") are not retrained in the feature extraction phase.

It is evident that models without full retraining exhibit significantly lower F1-score performance, particularly noticeable in the *open-case* scenario. Even in the *KD-case* scenario, fully retrained models outperform those without retraining. For instance, in the *KD-case* scenario at a budget of 100K, fr-Roberta-SVM achieves the highest F1-score among the frozen group (88.44%), still lower than ufr-Distilbert-Bag (89.06%).

In the *open-case* scenario, the top-performing frozen group, fr-Distilbert-SVM, only reaches 44.44%.

D. RQ4: What is the influence of machine learning methods on the results of imitating the functionality of the text classification model?

Apart from employing deep learning models, we also explore the performance of traditional machine learning models in the text classification phase for selecting knockoff models in text analysis tasks. Fig. 2 and Fig. 3 above demonstrate the effectiveness of models using feature extraction from both Roberta and Distilbert, along with four different text classification layers: regression, SVM, RF, and Bagging.

Fig. 2 indicates that in the *KD-case* and *closed-case* scenarios, traditional machine learning methods at the text classification layer exhibit comparable performance. For instance, in Fig. 2a, within the *KD-case* at a budget of 100K, Roberta achieves the highest accuracy of 94.56%, closely followed by traditional machine learning models (Roberta-SVM 94.53%, Roberta-RF 94.49%, Roberta-Bag 94.55%). However, in the *open-case* scenario, there's a disparity among the text classification techniques, where traditional machine learning models lag behind by approximately 1% compared to using the same vectorization model. For instance, while Roberta achieves the highest accuracy of 91.41%, traditional ML models exhibit slightly lower effectiveness ranging from 0.73 to 1.42% (Roberta-SVM 90.68%, Roberta-RF 90.40%, Roberta-Bag 89.99%).

In summary, traditional machine learning methods are suitable for the text classification layer when choosing knockoff models. When the data are related to the victim's training set, traditional ML models achieve accuracy levels almost equivalent to deep learning. However, in the *open-case* scenario, their execution efficiency is slightly lower compared to deep learning.

After the knockoff models have converged during training, it is necessary to compare their inference times with that of the victim model. Figure 3 shows the inference times of the victim model and the knockoff models with 8 different architectures (tested under the best-case scenario with a budget of 100K) after 5 runs on the entire test set of the victim model (measured in seconds). From this table, it is clear that the average inference time of each knockoff model is quite stable across the runs. Additionally, the discrepancies between the knockoff models and between them and the victim model (XLNet-base) are due to differences in architecture and machine learning algorithms affecting execution time. It can be concluded that, in this experiment, in terms of inference time, the trained knockoff models are not slower than the original model.

6. Conclusions

In conclusion, this article proposed a method for creating knockoff models from black-box neural models for text sentiment analysis. Through a series of experiments, we explored various aspects of the knockoff model generation process, including dataset selection, model performance under different scenarios, the impact of transfer learning,

and the effectiveness of traditional machine learning methods. Our findings suggest that knockoff models can closely mimic the functionality of black-box neural models, especially when the training data overlap with the victim's dataset. However, in scenarios where the training data differ significantly from the victim's dataset, the imitation effect diminishes. Furthermore, it was observed that transfer learning plays an important role in enhancing the performance of knockoff models, emphasizing the necessity of retraining the entire model to achieve promising functional mimicry results. Additionally, we demonstrated that traditional machine learning methods can be effectively applied to the text classification layer, particularly in scenarios where data are related to the victim's training set. In general, our experiments shed light on the feasibility and effectiveness of generating knockoff models for text sentiment analysis tasks, providing valuable insight for researchers and practitioners in the field.

Future work could focus on developing adaptive sample selection techniques for knockoff model generation, aiming to dynamically select and prioritize data samples that are most informative and relevant for model imitation.

References

[1] W. Liu, Z. Wang, X. Liu, ..., and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, 2017. doi: 10.1016/j.neucom.2016.12.038

[2] P. P. Shinde and S. Shah, "A review of machine learning and deep learning applications," in *2018 Fourth international conference on computing communication control and automation (ICCUBEA)*. IEEE, 2018, pp. 1–6, doi: 10.1109/ICCUBEA.2018.8 697 857.

[3] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Stealing machine learning models via prediction APIs," in *25th USENIX security symposium (USENIX Security 16)*, 2016, pp. 601–618, doi: 10.5555/3 241 094.3 241 142.

[4] T. Orekondy, B. Schiele, and M. Fritz, "Knockoff nets: Stealing functionality of black-box models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4954–4963, doi: 10.1109/CVPR.2019.00 509.

[5] J. R. Correia-Silva, R. F. Berriel, ..., and C. Badue, "Copycat CNN: Stealing knowledge by persuading confession with random non-labeled data," in *2018 International joint conference on neural networks (IJCNN)*. IEEE, 2018, pp. 1–8, doi: 10.1109/IJCNN.2018.8 489 592.

[6] W. Ccoya and E. Pinto, "Comparative analysis of libraries for the sentimental analysis," *arXiv preprint arXiv:2307.14311*, 2023. doi: 10.48550/arXiv.2307.14311

[7] S. Kumar, P. P. Roy, D. P. Dogra, and B.-G. Kim, "A comprehensive review on sentiment analysis: Tasks, approaches and applications," *arXiv preprint arXiv:2311.11250*, 2023. doi: 10.48550/arXiv.2311.11250

[8] D. Oliynyk, R. Mayer, and A. Rauber, "I know what you trained last summer: A survey on stealing machine learning models and defences," *ACM Computing Surveys*, vol. 55, no. 14s, pp. 1–41, 2023. doi: 10.1145/3595292

[9] E. De Cristofaro, "An overview of privacy in machine learning," *arXiv preprint arXiv:2005.08679*, 2020. doi: 10.48550/arXiv.2005.08679

[10] S. Kotyan, "A reading survey on adversarial machine learning: Adversarial attacks and their understanding," *arXiv preprint arXiv:2308.03363*, 2023. doi: 10.48550/arXiv.2308.03363

[11] B. Nelson, M. Barreno, F. J. Chi, ..., and K. Xia, "Exploiting machine learning to subvert your spam filter," in *LEET'08: Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*, no. 7, 2008, pp. 16–17, doi: 10.5555/1 387 709.1 387 716.

[12] C. Szegedy, W. Zaremba, I. Sutskever, ..., and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013. doi: 10.48550/arXiv.1312.6199

[13] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *2017 IEEE symposium on security and privacy (SP)*. IEEE, 2017, pp. 3–18, doi: 10.1109/SP.2017.41.

[14] M. Jagielski, N. Carlini, D. Berthelot, A. Kurakin, and N. Papernot, "High accuracy and high fidelity extraction of neural networks," in *29th USENIX security symposium (USENIX Security 20)*, 2020, pp. 1345–1362, doi: 10.48 550/arXiv.1909.01 838.

[15] A. Jindal, V. Goyal, S. Anand, and C. Arora, "Army of thieves: Enhancing black-box model extraction via ensemble based sample selection," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 3823–3832, doi: 10.1109/WACV57 701.2024.00 378.

[16] S. Pal, Y. Gupta, A. Shukla, and V. Ganapathy, "Activethief: Model extraction using active learning and unannotated public data," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, 2020, pp. 865–872, doi: 10.1609/aaai.v34i01.5432.

[17] C. Buciluǎ, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2006, pp. 535–541, doi: 10.1145/1 150 402.1 150 464.

[18] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "Model compression and acceleration for deep neural networks: The principles, progress, and challenges," *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 126–136, 2018. doi: 10.1109/MSP.2017.2765695

[19] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015. doi: 10.48550/arXiv.1503.02531

[20] İ. Tarımer, A. Çoban, and A. E. Kocaman, "Sentiment Analysis on IMDB Movie Comments and Twitter Data by Machine Learning and Vector Space Techniques," *arXiv preprint arXiv:1903.11983*, 2019. doi: 10.48550/arXiv.1903.11983

[21] R. Johnson and T. Zhang, "Supervised and semi-supervised text categorization using LSTM for region embeddings," in *International Conference on Machine Learning*. PMLR, 2016, pp. 526–534, doi: 10.48 550/arXiv.1602.02 373.

[22] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of NAACL-HLT 2019*, 2019, pp. 4171–4186, doi: 10.48 550/arXiv.1810.04 805.

[23] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," in *The 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS 2019*. doi: 10.48550/arXiv.1910.01108

[24] Y. Liu, M. Ott, N. Goyal, ..., and V. Stoyanov, "RoBERTa: A Robustly Optimized BERT Pretraining Approach," *arXiv preprint arXiv:1907.11692*, 2019. doi: 10.48550/arXiv.1907.11692

[25] Z. Yang, Z. Dai, Y. Yang, ..., and Q. V. Le, "XLNet: Generalized Autoregressive Pretraining for Language Understanding," *Advances in neural information processing systems*, vol. 32, 2019. doi: 10.48550/arXiv.1906.08237

[26] A. Maas, R. E. Daly, P. T. Pham, ..., and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, 2011, pp. 142–150.

[27] R. Socher, A. Perelygin, J. Wu, ..., and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 conference on empirical methods in natural language processing*. Seattle, Washington, USA: Association for Computational Linguistics, Oct. 2013, pp. 1631–1642. [Online]. Available: https://aclanthology.org/D13-1170

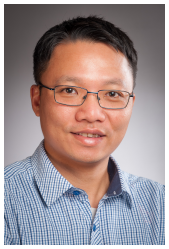
[28] S. Hussein, "Twitter sentiments dataset, version 1," 14 May 2021, [Online, accessed 24 June 2024]. [Online]. Available: https://data.mendeley.com/datasets/z9zw7nt5h2/1

Manuscript received: 10-04-2024; Accepted: 25-06-2024.

■

Xuan Cong Pham received degree of master in Mathematical Foundations for Computers and Computing systems in 2012 at the University of Natural Sciences under Hanoi National University. He is currently studying the Ph.D. program in Mathematical Foundations for Information Technology at the Academy of Military Science and Technology. His main research interest is in micro-control computing systems using real-time operating systems. In addition, he also actively researches artificial intelligence, information system security and software reverse engineering.
Email: congpx@gmail.com

Trung Nguyen Hoang is a fourth-year student majoring in Computer Science at Le Quy Don Technical University. His main research interests are artificial intelligence, machine learning and computational mathematics.
Email: nguyenhtmta02@gmail.com



Cao Truong Tran received the PhD degree in computer science from Victoria University of Wellington, New Zealand. He also did postdoc at Victoria University of Wellington. He is researching in the field of machine learning and evolutionary computation, specialized with evolutionary machine learning for data mining with missing data. He serves as a reviewer of international journals, including IEEE Transactions on Evolutionary Computation, IEEE Transactions on Cybernetics, Pattern Recognition, Knowledge-Based Systems, Applied Soft Computing and Engineering Application of Artificial Intelligence. He is also a PC member of international conferences, including IEEE Congress on Evolutionary Computation, IEEE Symposium Series on Computational Intelligence, the Australasian Joint Conference on Artificial Intelligence, and the AAAI Conference on Artificial Intelligence.
Email: truongct@lqdtu.edu.vn

Viet Binh Do received his PhD in Mathematical Foundations for Informatics in 2019 from the Academy of Military Science and Technology (AMST). Currently, he is the director of the Institute of Information Technology at AMST and a member of the Executive Committee of the Vietnam Informatics Association. His research focuses on the fields of security and encryption, virtual reality simulation systems, and artificial intelligence.
Email: binhdv@gmail.com

XÂY DỰNG MÔ HÌNH MÔ PHỎNG LẠI CHỨC NĂNG CỦA MÔ HÌNH HỘP ĐEN PHÂN TÍCH CẢM XÚC VĂN BẢN

Phạm Xuân Công, Hoàng Trung Nguyên, Trần Cao Trưởng, Đỗ Việt Bình

Tóm tắt

Phần lớn các mô hình học máy thương mại ngày nay được thiết kế đòi hỏi nhiều thời gian, tiền bạc và sức lực của con người, do đó, thông tin nội tại về mô hình (như kiến trúc, các siêu tham số và tập dữ liệu huấn luyện) cần được giữ bí mật. Những mô hình này được gọi là hộp đen và ngày càng có nhiều nghiên cứu tìm cách tấn công cũng như đưa ra phương pháp bảo vệ chúng. Các công bố gần đây thường tập trung vào lĩnh vực thị giác máy tính, ngược lại, còn khá ít nghiên cứu về các phương pháp tấn công mô hình hộp đen với dữ liệu văn bản. Bài báo này giới thiệu một phương pháp nghiên cứu trích xuất chức năng của mô hình hộp đen trong nhiệm vụ phân tích cảm xúc văn bản. Phương pháp này đã được thử nghiệm hiệu quả dựa trên các kỹ thuật lấy mẫu ngẫu nhiên nhằm tái tạo một mô hình mới có chức năng tương đương với mô hình ban đầu, đạt độ chính xác (94.46% so với 94.92%) và độ tương đồng cao (96.82%).

Từ khóa

Phân lớp văn bản, phân tích cảm xúc, trích xuất mô hình hộp đen, mô phỏng chức năng mô hình.

Appendix

Table 2. Performance comparison of victim models on different datasets
Acc: Accuracy, F1: F1 score, Run-time is counting in seconds

Dataset	IMDB			SST2			TWEET		
Model	**Acc(%)**	**F1(%)**	**Run-time**	**Acc(%)**	**F1(%)**	**Run-time**	**Acc(%)**	**F1(%)**	**Run-time**
Distilbert	93.07	93.14	2049.1	94.77	95.23	1920.9	84.22	84.39	4870.5
Roberta	94.71	94.75	3909.6	94.00	94.59	3671.2	87.12	86.87	9912.7
XLNet	94.94	**94.92**	**7835.8**	95.00	95.47	5059.4	85.27	85.16	12800.0

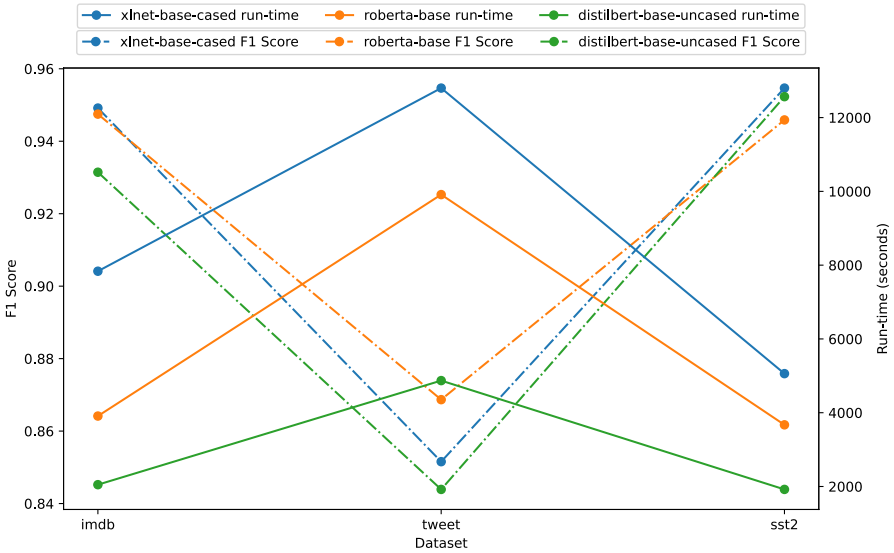


Fig. 5. F1 score(%) and Run-time of victims.

Table 3. Comparison table of inference time for models on victim's test set (seconds)

Models	Round 1	Round 2	Round 3	Round 4	Round 5	Avarage
Roberta-base	167.12	189.64	190.57	191.05	191.74	**186.02**
Distilbert-BU	90.88	96.15	93.33	97.42	95.39	**94.63**
Roberta-base-SVM	265.30	242.13	280.30	296.13	248.68	**266.51**
Roberta-base-RF	168.86	184.86	175.71	184.02	177.34	**178.16**
Roberta-base-Bag	168.24	168.24	175.62	176.44	184.17	**174.54**
Distilbert-BU-SVM	155.10	151.27	152.25	168.59	156.41	**156.72**
Distilbert-BU-RF	81.68	104.02	82.54	84.10	85.08	**87.48**
Distilbert-BU-Bag	81.53	82.13	82.96	84.09	84.87	**83.12**
Victim (XLNet-base)	430.04	473.37	467.38	441.02	440.69	**450.50**

Table 4. F1-scores(%) of knockoff models with unfroze the BERT layers and the last classifier layer. The first two models were taken directly from the library and trained from scratch. The last six models were similar, except that their final classification layer was replaced respectively by three traditional machine learning models

Datasets				**KD-case**				**closed-case**					**open-case**		
Budgets	5K	10K	20K	50K	100K	5K	10K	20K	50K	100K	5K	10K	20K	50K	100K
Roberta-base	93.77	93.64	93.83	94.56	94.46	92.16	93.07	93.72	93.67	93.92	86.02	87.45	89.44	88.87	**91.41**
Distilbert-BU	90.33	90.57	91.85	92.65	92.99	87.91	89.42	90.50	91.72	92.48	72.04	83.23	83.49	86.07	89.56
Roberta-base-SVM	93.63	93.70	93.93	94.53	94.45	92.10	92.75	93.51	93.40	93.85	82.28	84.23	86.45	87.56	90.68
Roberta-base-RF	93.56	93.64	93.96	94.49	94.43	91.99	92.82	93.50	93.38	93.89	80.38	85.22	87.13	87.74	90.40
Roberta-base-Bag	93.31	93.56	93.79	94.55	94.31	91.97	92.60	93.34	93.25	93.75	78.31	84.78	87.25	87.30	89.99
Distilbert-BU-SVM	90.30	91.06	91.84	92.77	92.88	87.58	89.12	90.54	91.59	92.49	62.09	80.81	82.92	85.77	89.40
Distilbert-BU-RF	90.31	91.07	91.80	92.77	92.91	88.03	88.95	90.46	91.48	92.29	62.31	79.58	82.25	84.86	89.44
Distilbert-BU-Bag	89.90	90.77	91.42	92.53	92.84	87.65	88.73	90.32	91.31	92.16	59.94	77.00	82.44	83.68	89.06
Victim			**94.92**												

Table 5. Agreement(%) of knockoff models with unfroze the BERT layers and the last classifier layer. The first two models were taken directly from the library and trained from scratch. The last six models were similar, except that their final classification layer was replaced respectively by three traditional machine learning models

Datasets	**KD-case**					**closed-case**					**open-case**				
Budgets	5K	10K	20K	50K	100K	5K	10K	20K	50K	100K	5K	10K	20K	50K	100K
Roberta-base	95.28	96.22	96.34	**96.92**	96.82	94.42	95.26	95.68	**96.41**	95.93	89.67	90.96	92.65	92.22	**94.07**
Distilbert-BU	92.29	92.81	93.99	95.03	95.29	89.79	91.40	92.90	93.93	94.66	80.05	87.16	87.83	89.76	91.81
Roberta-base-SVM	95.33	96.29	96.36	96.89	96.81	94.34	95.02	95.74	96.32	95.89	86.90	88.63	90.36	91.20	93.67
Roberta-base-RF	95.20	96.26	96.39	96.98	96.12	94.33	95.07	95.76	96.34	95.95	85.65	89.36	90.89	91.36	92.75
Roberta-base-Bag	95.06	96.21	96.20	96.98	96.04	94.40	95.02	95.74	96.13	95.85	84.21	88.95	90.97	91.05	92.21
Distilbert-BU-SVM	92.48	93.25	93.98	94.83	94.78	89.93	91.55	92.89	93.80	94.61	72.79	84.52	86.44	88.64	91.52
Distilbert-BU-RF	92.52	93.29	93.98	94.79	94.83	90.16	91.35	92.79	93.71	94.50	72.89	83.64	85.99	87.95	91.55
Distilbert-BU-Bag	92.22	92.87	93.60	94.72	94.71	89.63	91.01	92.65	93.50	94.22	71.71	81.97	85.88	86.88	91.37

Table 6. Run-time(s) of knockoff models with unfroze the BERT layers and the last classifier layer. The first two models were taken directly from the library and trained from scratch. The last six models were similar, except that their final classification layer was replaced respectively by three traditional machine learning models

Datasets	**KD-case**					**closed-case**					**open-case**				
Budgets	5K	10K	20K	50K	100K	5K	10K	20K	50K	100K	5K	10K	20K	50K	100K
Roberta-base	385.1	747.1	1476.5	3620.3	7218.7	383.9	747.4	1467.8	3602.3	7240.2	329.9	616.2	1264.7	3457.1	6759.9
Distilbert-BU	180.0	317.3	622.4	1539.0	3069.3	162.7	314.9	617.1	1527.8	3052.9	64.9	119.5	254.1	780.0	1521.1
Roberta-base-SVM	531.1	1037.1	1824.9	4514.5	8977.6	522.9	1042.6	1836.6	4587.3	10961.2	367.1	677.4	1369.6	3798.6	7890.0
Roberta-base-RF	542.8	1056.9	1871.5	4600.5	9008.9	550.8	1079.2	1936.5	4797.3	9548.3	389.7	733.5	1444.9	3992.2	7711.6
Roberta-base-Bag	552.0	1065.1	1890.7	4652.6	9010.4	561.0	1085.0	2017.3	5170.3	10129.7	405.8	793.0	1556.8	4376.7	8340.0
Distilbert-BU-SVM	222.7	403.9	798.3	1988.8	3905.1	206.4	403.7	804.7	2080.6	4648.0	73.2	138.9	305.5	989.1	2482.3
Distilbert-BU-RF	242.2	451.2	907.9	2289.9	4038.2	228.5	453.1	929.6	2434.5	4963.1	96.1	193.1	444.4	1349.6	2773.4
Distilbert-BU-Bag	255.2	483.9	1028.0	2536.5	4051.5	245.3	506.0	1064.8	3002.6	6150.7	124.5	244.5	596.4	2023.1	3957.8

Table 7. F1-scores(%) of the Adversarial models, where we froze the BERT layers, training only the final classifier layer. The first two models were taken directly from the library and trained only the last classification layer. The last six models were similar, except that their final classification layer was replaced respectively by three traditional machine learning models

Datasets	**KD-case**					**closed-case**					**open-case**				
Budgets	5K	10K	20K	50K	100K	5K	10K	20K	50K	100K	5K	10K	20K	50K	100K
Roberta-base	10.49	72.37	77.70	83.48	85.51	0.00	0.00	0.04	41.31	75.32	0.00	0.00	0.00	0.00	0.00
Distilbert-BU	77.56	80.50	83.04	85.16	85.52	0.04	58.62	75.69	80.76	83.24	0.00	0.00	9.04	28.99	33.07
Roberta-base-SVM	82.69	85.03	86.32	87.81	88.44	0.00	0.00	34.54	82.00	84.84	0.00	0.00	0.00	0.00	0.00
Roberta-base-RF	83.13	83.57	84.04	84.61	84.64	76.97	78.23	79.41	79.80	81.27	0.24	0.24	0.16	0.44	0.36
Roberta-base-Bag	76.37	77.53	78.15	79.49	79.23	69.65	71.63	73.03	73.47	75.35	6.01	6.41	16.08	10.12	8.15
Distilbert-BU-SVM	85.79	86.06	86.60	87.06	87.20	82.91	84.15	84.96	85.75	86.17	0.00	0.40	4.15	28.69	44.44
Distilbert-BU-RF	81.31	82.00	82.30	82.98	82.95	76.80	77.57	78.40	78.84	80.47	5.27	10.13	8.61	13.68	21.17
Distilbert-BU-Bag	74.92	76.35	76.55	77.34	78.00	70.94	70.28	71.58	72.84	74.32	33.96	19.25	19.99	26.07	32.50
Victim			**94.92**												

Journal of Science and Technique - ISSN 1859-0209

Table 8. Agreement(%) of the Adversarial models, where we froze the BERT layers, training only the final classifier layer. The first two models were taken directly from the library and trained only the last classification layer. The last six models were similar, except that their final classification layer was replaced respectively by three traditional machine learning models

Datasets	**KD-case**					**closed-case**					**open-case**				
Budgets	5K	10K	20K	50K	100K	5K	10K	20K	50K	100K	5K	10K	20K	50K	100K
Roberta-base	52.83	77.63	81.05	84.98	86.89	50.08	50.08	50.09	62.08	78.74	50.08	50.08	50.08	50.08	50.08
Distilbert-BU	79.93	81.85	84.21	86.29	86.86	50.09	65.58	76.75	82.15	84.60	50.08	50.08	52.28	58.22	59.79
Roberta-base-SVM	84.31	86.46	87.88	89.29	89.88	50.08	50.08	60.52	84.59	86.75	50.08	50.08	50.08	50.08	50.08
Roberta-base-RF	84.46	85.04	85.61	85.92	86.11	80.07	80.97	82.24	82.63	83.62	50.14	50.14	50.12	50.16	50.14
Roberta-base-Bag	78.88	79.93	80.31	81.49	81.70	74.44	75.62	76.65	76.98	78.40	50.63	51.02	52.37	51.78	50.99
Distilbert-BU-SVM	87.02	87.38	87.90	88.31	88.48	83.97	85.01	85.91	86.91	87.33	50.08	50.18	51.14	58.38	64.24
Distilbert-BU-RF	83.03	83.34	83.70	84.49	84.49	79.34	79.95	80.74	81.34	82.49	51.25	52.35	51.93	53.37	55.51
Distilbert-BU-Bag	77.56	78.69	78.94	79.82	80.08	74.30	74.33	75.30	76.46	77.62	56.06	53.68	52.80	55.58	57.01

Table 9. Run-time(s) of the Adversarial models, where we froze the BERT layers, training only the final classifier layer. The first two models were taken directly from the library and trained only the last classification layer. The last six models were similar, except that their final classification layer was replaced respectively by three traditional machine learning models

Datasets	**KD-case**					**closed-case**					**open-case**				
Budgets	5K	10K	20K	50K	100K	5K	10K	20K	50K	100K	5K	10K	20K	50K	100K
Roberta-base	256.0	460.0	909.2	2262.0	4514.7	236.1	459.0	909.9	2261.8	4511.2	202.7	392.7	791.8	2158.3	4256.9
Distilbert-BU	130.9	256.9	506.9	1260.9	2515.3	130.9	256.8	506.3	1262.0	2513.7	84.1	164.7	335.9	856.6	1707.1
Roberta-base-SVM	366.4	720.2	1560.8	4662.5	11447.5	346.5	719.2	1561.5	4662.4	11444.0	313.1	652.9	1443.4	4558.8	11189.7
Roberta-base-RF	357.7	667.1	1332.4	3364.5	6694.3	337.8	666.1	1333.1	3364.3	6690.7	304.4	599.8	1215.0	3260.8	6436.4
Roberta-base-Bag	372.2	699.1	1416.7	3583.5	7122.0	352.3	698.1	1417.4	3583.4	7118.5	318.9	631.8	1299.3	3479.8	6864.2
Distilbert-BU-SVM	185.2	387.4	859.0	2744.5	7478.8	185.2	387.3	858.4	2745.5	7477.2	138.4	295.1	688.0	2340.1	6670.7
Distilbert-BU-RF	187.2	373.0	746.3	1891.2	3778.8	187.2	372.9	745.7	1892.3	3777.2	140.4	280.7	575.3	1486.9	2970.7
Distilbert-BU-Bag	197.6	400.1	813.9	2091.9	4169.5	197.6	400.0	813.4	2092.9	4167.9	150.8	307.8	642.9	1687.5	3361.4