Student Surpasses Teacher: Imitation Attack for Black-Box NLP APIs

Qiongkai Xu,[*1] **Xuanli He,**[2] **Lingjuan Lyu,**[3] **Lizhen Qu,**[2] **Gholamreza Haffari**[2]
[1]The University of Melbourne, [2]Monash University, [3]Sony AI
Qiongkai.Xu@unimelb.edu.au, Lingjuan.Lv@sony.com
{xuanli.he1,Lizhen.Qu,Gholamreza.Haffari}@monash.edu

Abstract

Machine-learning-as-a-service (MLaaS) has attracted millions of users to their splendid large-scale models. Although published as black-box APIs, the valuable models behind these services are still vulnerable to imitation attacks. Recently, a series of works have demonstrated that attackers manage to steal or extract the victim models. Nonetheless, none of the previous stolen models can outperform the original black-box APIs. In this work, we conduct unsupervised domain adaptation and multi-victim ensemble to showing that attackers could potentially surpass victims, which is beyond previous understanding of model extraction. Extensive experiments on both benchmark datasets and real-world APIs validate that the imitators can succeed in outperforming the original black-box models on transferred domains. We consider our work as a milestone in the research of imitation attack, especially on NLP APIs, as the superior performance could influence the defense or even publishing strategy of API providers.

1 Introduction

Task-oriented NLP APIs have received tremendous success, partly due to commercial cloud services (Krishna et al., 2019; Wallace et al., 2020). The enormous commercial benefit allures other companies or individual users to extract the back-end decision models of these successful APIs. Some recent works have demonstrated that many existing NLP APIs can be locally imitated or extracted (Krishna et al., 2019; Wallace et al., 2020; Guo et al., 2022), violating the intellectual property (IP) of NLP APIs. Equipped with recent advanced models pre-trained on large-scale corpus, it is getting easier to train a decent attack model with limited training samples retrieved from victims (He et al., 2021a).

---

*Most of the work was finished when the first author was at ANU and Data61 CSIRO.

Despite their success, there are still a series of restrictions on current attack paradigm. The first restriction is that attackers and victims are trained and evaluated under the same domain. Although such setting simplifies the comparison of utility and fidelity between victim and attack models, it is less likely to be the case in real world. Generally, attackers and victims do not, at least not willing to, share their in-house datasets to the public. Moreover, the competition between attackers and victims lies in their performance on the application scenarios (transferred domains), which vary by customers not API providers. The second restriction is that traditional model extraction only refers a single victim model, and none of the previous works manage to build the extracted models that can surpass the original black-box APIs (Tramèr et al., 2016; Krishna et al., 2019; Wallace et al., 2020). Aggregating the strength from diverse victims, however, can potentially benefit the performance of attackers. This may further enable the attack model to acquire a superior performance, in the best case, even surpassing all the corresponding victim models.

Based on the above analysis, we are motivated to address their restrictions by *i)* conducting imitation attacks on a transferred domain and *ii)* utilizing multiple victim ensemble. Our approach integrates unsupervised domain adaptation and model ensemble into the imitation attack. We conduct experiments on two representative NLP tasks, namely sentiment classification (Lyu et al., 2020) and machine translation (Wallace et al., 2020). We investigate the attack performance on both the locally simulated victim models and publicly available commercial NLP APIs. Our results demonstrate that the attackers could potentially achieve better performance than victims in the transferred domains and utilizing multiple victim models further improves the performance of the attack models. For those target domains that are relatively far from victims' source domains, *e.g.*, sentiment analysis

2849

models on movie review domain to financial document domain, the performance improvement of the imitation model could be further amplified.

Overall, our empirical findings exacerbate the potential risk concerns of public API services, as malicious companies could provide a better service in their specific domains by integrating several publicly available APIs. Moreover, the new services generally could cost far less than any of the original victim services or the wage paid to human annotators. The imitation attack not only infringes the IP of victim companies by misusing the predictions of their APIs, but also potentially corrupts the MLaaS market by publishing new APIs, with higher performance but lower price. We believe that explicitly exposing the new imitation attack paradigm would arouse significant attentions in the research community and encourage companies to reconsider their strategies of publishing API services.

2 Related Work

2.1 Model Imitation Attack

Model imitation attack (also referred to as model "extraction" or "stealing") has been studied for simple classification tasks (Tramèr et al., 2016), vision tasks (Orekondy et al., 2019), NLP tasks (Krishna et al., 2019; Wallace et al., 2020), and *etc*. Generally, model imitation attack aims to reconstruct a local copy or to steal the functionality of a black-box API. If the reconstruction is successful, the attacker has effectively stolen the intellectual property. Previous works on imitation attack (Krishna et al., 2019; Wallace et al., 2020) mainly focus on how to imitate a model with performance approximate to the victim API in the source domain. Whether a more powerful attacker can steal a model that is better than any victim API in new domains is largely unexplored.

2.2 Unsupervised Domain Adaption

Domain adaptation is a task of adapting a pre-trained model on a source domain to a target domain. It can be fulfilled in two mechanisms: supervised adaptation and unsupervised adaptation. The former can achieve an outstanding performance with a small amount of in-domain labeled data (Daumé III, 2007; Ben-David et al., 2007). In contrast, the unsupervised domain adaptation (UDA) (Miller, 2019; Ganin and Lempitsky, 2015) does not require ground-truth labels in the target domain, hence is more challenging but attractive.

This work is under the umbrella of UDA family (Wang et al., 2021; Miller, 2019). We differentiate our work from other UDA works in terms of the intent. Other UDA works aim to improve the models in both source domains and target domains simultaneously, while our proposed imitation attack mainly focuses on optimizing the adapted attack models merely in target domains. We exploit UDA from a dark side, where one can leverage domain adaptation to violate IP of commercial APIs, but benefit from such violations.

2.3 Ensemble for Knowledge Distillation

Model imitation attack is related to knowledge distillation (KD) (Hinton et al., 2015). Emergent knowledge distillation (Buciluá et al., 2006; Hinton et al., 2015) aims to transfer knowledge from a large teacher model to a small student model for the sake of model compression. By encouraging the student model to approximate the behaviors of the teacher model, the student is able to imitate the teacher's behavior with minor quality loss but more efficient inference (Furlanello et al., 2018; Dvornik et al., 2019; Anil et al., 2018).

However, model imitation differs from distillation in terms of data usage. Particularly, the victim's (*i.e.*, teacher's) training data is usually unknown as the victim API is deployed in a black-box manner. Thus, malicious users tend to consider unlabeled queries more likely to be collected from other domains (Xie et al., 2020) or generated from pre-trained generative models (Ravuri and Vinyals, 2019; Kumar et al., 2020; Besnier et al., 2020). Our work considers a more realistic setting, as victim and attack models are under different domains and express distinctive interests.

On the other hand, with the development of multi-task learning, multi-teacher distillation has been proposed, which is targeted at distilling multiple single-task models into a single multi-task model (Tan et al., 2019; Clark et al., 2019; Saleh et al., 2020). These lines of works lie in the spirit of improving model performance and encouraging parameter reduction via knowledge distillation.

The dichotomy between our work and previous works is we are interested in an ensemble distillation of single-task models, where all teachers and the student work on the same task, but could be in different domain. The gist of this approach aims to leverage the collective wisdom to obtain a better student outperforming its teachers, which is sim-
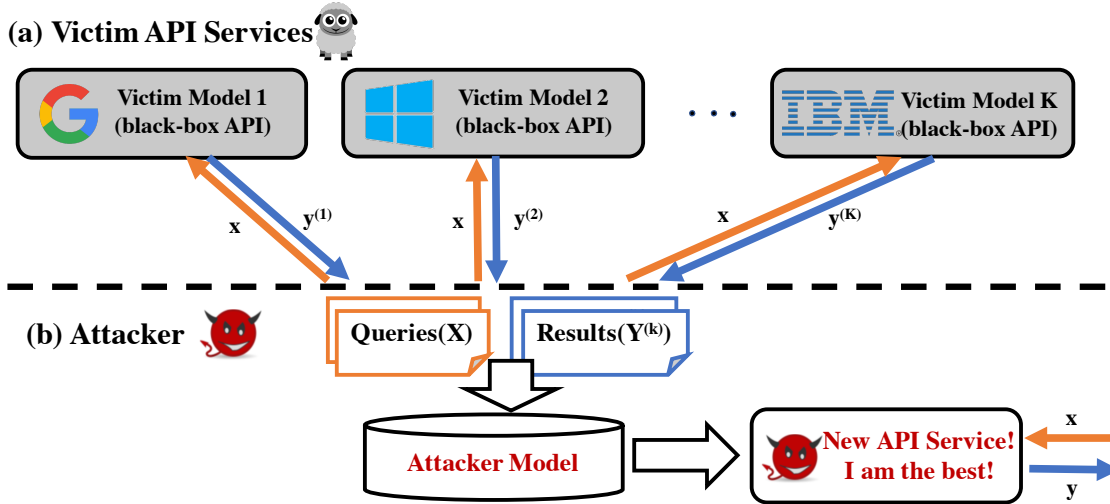
Figure 1: The workflow of the imitation attack and its harm. The attacker labels queries (x) using multiple victim APIs, and then trains an attack model on the resulting data. Finally, the attacker could publish a new API service, which could become a competitor of the victim services, thus infringing intellectual properties and eroding MLaaS market shares of the victim services.

ilar to the ensemble learning (Opitz and Maclin, 1999). However, ensemble learning focuses more on the aggregation of the predictions of different models at the inference stage, whereas our proposed method not only gathers the inputs and their predictions, but also trains a new model on the input-prediction pairs.

3 Imitation Attack Paradigm

In this section, we first explain the problem and motivation of imitation attack (IMA) under domain adaptation setting. Then, we connect our IMA practice with a corresponding domain adaptation theory (Ben-David et al., 2010; Wang et al., 2021). After that, we introduce a multi-victim ensemble methodology for IMA. Finally, we explain the rationality of a family of existing defense technologies under the domain adaptation theory.

3.1 Problem Statement

In the real world, attackers may be interested in their own new business, *e.g.*, a new classification or machine translation system in a new domain. Generally, the attackers possess their own training samples, $\mathcal{X} = \{x_1, x_2, \cdots, x_N\}$, while the oracle labels $\mathcal{Y} = \{y_1, y_2, \cdots, y_N\}$ of these samples are not yet labeled. In order to train a model with the least cost of annotation, attackers access the publicly available commercial APIs for the target task $\mathcal{T}$. Moreover, the attackers could inquire multiple APIs for further performance improvement. The underlying models of the attacked APIs are

K victim models, $\mathcal{V} = \{h_v^{(1)}, h_v^{(2)}, \cdots, h_v^{(K)}\}$. As illustrated in Figure 1, our attack can be formulated as a two-step process:

1. The attacker queries the k-th victim model $h_v^{(k)}$ with samples in $\mathcal{X}$, and retrieves corresponding labels $\mathcal{Y}^{(k)} = \{h_v^{(k)}(x_1), h_v^{(k)}(x_2), \cdots, h_v^{(k)}(x_N)\}$.

2. The attacker learns an imitation (attack) model $h_a(x)$ based on the queries and concatenated retrieved labels, $\langle \mathcal{X}, \mathcal{Y}^{(k)} \rangle |_{k=1}^{K}$.

As the attacking process involves teaching a local model by imitating the behavior of the victim models, we put this attack process under the umbrella of imitation attack. The motivation of such attack paradigm is twofold:

- Firstly, querying commercial APIs generally costs far less than hiring human annotators. Therefore, the price for training the attack models using imitated labels is competitive in the market;

- Secondly, the attackers are potentially able to outperform the victims in terms of utility. As demonstrated in Section 5, domain adaptation and multi-victim ensemble further enhance the attacker performance.

We believe that both price and performance advantages would lure companies or individuals to imitation attack.

2851

3.2 Training for Imitation Attack

In imitation attack, the attacker utilizes the labels from victim APIs for model training. Given a victim model $h_v(x)$, the attacker model $h_a(x)$ imitates the behavior of victim model by minimizing the prediction error on the target domain $\mathcal{T}$,

$$\min_{h_a} \mathbb{E}_{x\sim\mathcal{T}}\left[\mathcal{L}(h_a(x), h_v(x))\right]. \quad (1)$$

On the other hand, we assume that the victim model has learned from oracle annotations y in another source domain $\mathcal{S}$, although those labels are never used for training attacker in our IMA. The loss for training victim model is

$$\min_{h_v} \mathbb{E}_{x\sim\mathcal{S}}\left[\mathcal{L}(h_v(x), y)\right], \quad (2)$$

where $\mathcal{L}(\cdot, \cdot)$ is the loss function. In practice, we use cross entropy as loss functions for training both the victim and attack models. Note that jointly optimizing Equation 1 and Equation 2 can derive the unsupervised domain adaptation loss in (Miller, 2019; Ganin and Lempitsky, 2015). However, our approach optimizes victim and attack models in absolutely separate steps, as victim models and their training processes are black-box to the attackers.

3.3 Imitation Attack as Domain Adaptation

We connect our new IMA paradigm with domain adaptation theory. The error of the attacker is measured by attacker risk $\epsilon_a(h, h_a) := \mathbb{E}_{x\sim\mathcal{T}}[|h(x) - h_a(x)|]$. According to domain adaptation theory (Ben-David et al., 2010; Wang et al., 2021), the upper bound of the attacker risk is

$$\epsilon_a(h, h_a) \leq \epsilon_v(h, h_v) + 2d(P_X^s, P_X^t) \quad (3)$$
$$+ \min_{\mathcal{T}\in\{P_X^s, P_X^t\}} \mathbb{E}_{x\sim\mathcal{T}}\left[|h_v(x) - h_a(x)|\right],$$

where $d(P_X^s, P_X^t)$ is the total variation between the distributions of source and target domains, which is a constant determined by domains of the datasets used by the victim and attacker. The first term, $\epsilon_v(h, h_v) := \mathbb{E}_{x\sim\mathcal{S}}[|h(x) - h_v(x)|]$, with oracle decisions $y = h(x)$, is the victim risk on the source domain. It is optimized during the training of victim models as Equation 2. The last term is associated with imitation training objective as Equation 1. Therefore, our imitation attack under domain adaptation setting is actually optimizing the upper bound of the attacker risk $\epsilon_a(h, h_a)$.

3.4 Multiple Victim Ensemble

Another approach to achieving further performance improvement is to integrate the results from multiple APIs. This strategy is well motivated in real-world imitation attack, as many cloud computing companies share similar APIs for main-stream NLP tasks, e.g, Google Cloud and Microsoft Azure both support sentiment classification and machine translation. Attackers can improve their performances by learning from multiple victim APIs. In more detail, given independent K victim models $h_v^{(k)}(x)$, attackers can take the average advantage of all victim models by,

$$\min_{h_a} \frac{1}{K} \sum_{k=1}^{K} \mathbb{E}_{x\sim\mathcal{T}}[\mathcal{L}(h_a(x), h_v^{(k)}(x))]. \quad (4)$$

According to ensemble theories (Breiman, 2001; Bauer and Kohavi, 1999), the lower generalization error of an ensemble model depends on *i)* better performance of the individual models, and *ii)* lower correlation between them. In real-world, companies are actually *i)* targeting on API with better performance, and *ii)* using their own private training datasets. The effort of these companies towards superior API performance unfortunately exacerbates these two factors for a successful ensemble model as an attacker.

3.5 Defending Imitation Attack

Some existing IMA defense strategies (He et al., 2021a) slightly distort the predictions of victim models to reduce the performance of attackers, by replacing $h_v(x)$ with noisy $h_v'(x)$. The variance introduced by distortion is $\Delta(h_v, h_v') := \mathbb{E}_{x\sim\mathcal{S}}[|h_v(x) - h_v'(x)|]$. As the distortion should not destroy the utility of victim model, the variance should be bounded by a small constant δ, *i.e.*, $\Delta(h_v, h_v') \leq \delta$. The victim risk, the first term in (3), is relaxed to $\epsilon_v(h, h_v')$, where

$$\epsilon_v(h, h_v') \leq \epsilon_v(h, h_v) + \Delta(h_v, h_v'). \quad (5)$$

Therefore, the gentle distortion of victim outputs results in a more relaxed upper bound for optimization, which could potentially lead to better results on defending the imitation attack.

4 Experimental Setup

4.1 Tasks and Datasets

In this paper, we focus on two essential NLP tasks, classification and machine translation, both

of which are predicting discrete outputs, and cross entropy loss is used as the objective in optimization. For classification tasks, as APIs provide continuous scores of the predictions, we consider the settings of soft label (using prediction scores) and hard label (using predicted categories). In machine translation, the translation result is a sequence of discrete words, which are considered as sequential hard labels. Classification and translation tasks are evaluated by accuracy(%) (Schütze et al., 2008) and BLEU (Papineni et al., 2002), respectively. The datasets used in our imitation attack experiment are summarized in Table 1.

Sentiment Classification. Imdb Movie Review (**IMDB**) (Maas et al., 2011) is a large-scale movie review dataset for sentiment analysis. Stanford Sentiment (**SST**) (Socher et al., 2013) is another movie review dataset with relatively shorter text than **IMDB**. Financial Sentiment (**FST**) (Malo et al., 2014) provides sentiment labels on economic texts in finance domain. We use **IMDB** to train local victim models and consider **SST** and **FST** as target domains in attack.

Machine Translation. We consider German (De) to English (En) translation as our testbed. We first study the attack performance on the local models trained on a general domain. Specifically, we use WMT14 (Bojar et al., 2014) to train the victim models. Then, we investigate the imitation attack on **Law** and **Koran** domains from OPUS (Tiedemann, 2012). We utilize Moses[1] to pre-process all corpora, and keep the text cased. A 32K BPE vocabulary (Sennrich et al., 2016) is applied to all datasets.

4.2 Victim Models

Locally Simulated NLP Services. We first use models trained on our local server as local services. Our models are trained on datasets in source domain, *i.e.*, **IMDB** for sentiment analysis and **WMT14** for machine translation. We vary BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) as pre-trained models for classification. Transformer-base (TF-base) and Transformer-big (TF-big) (Vaswani et al., 2017) are used as machine translation architectures, more details about hyperparameter settings are described in Appendix A.

Commercial NLP API Services. To investigate the performance of imitation attack on real-world

commercial NLP APIs, we query and retrieve the results of victim APIs for both sentiment analysis and machine translation. **Google Cloud** API[2] and **IBM Cloud** API[3] are inquired for sentiment analysis. **Google Trans**[4] and **Bing Trans**[5] are used as translation APIs. In this setting, we assume that different companies have various choices in datasets, domains, model architectures and training strategies. These settings are invisible to the attackers.

4.3 Imitation Attack Setup

For imitation attack, different from Wallace et al. (2020), we leverage datasets in other domains rather than those used for training victim models. The rationale behind this setting is that *i)* the owners of APIs tend to use in-house dataset, which is difficult for attacker to access; *ii)* attackers are more interested in their own private dataset, which is also not visible to others. Therefore, our setting is closer to the real-world attack scenario. The attack models are trained based on the labels retrieved from victim models, and tested on the labels from the human-annotated sets. We consider **SST** and **FST** as target domains for sentiment analysis. For machine translation, we use **Law** and **Koran**. In attack, we use BERT for classification. Regarding machine translation, Transformer-base is used for simulating local APIs, while mBART (Liu et al., 2020) for experiments on commercial APIs[6]. We also investigate the ensemble models, by concatenating all the outputs retrieved from multiple victim models in training.

5 Experimental Results

In this section, we analyze our experimental results. Our experiments are designed to answer the following research questions (**RQ**s),

- **RQ1**: Are the attack models able to outperform the victim models in new domains?

- **RQ2**: Will the ensemble of victim APIs improve the performance of the attack models?

- **RQ3**: Do traditional defense methods help APIs reduce the performance of attackers in our domain adaptation setting?

---

[1] https://github.com/moses-smt/mosesdecoder

[2] https://cloud.google.com

[3] https://cloud.ibm.com

[4] https://translate.google.com/

[5] https://www.bing.com/translator

[6] We observe that mBART works better on attacking commercial APIs than Transformers in our preliminary experiments.

Dataset	#Train	#Dev	#Test	Task	Domain
IMDB	25,000	25,000	N/A	Sentiment Classification	Movie Review (long)
SST	6,920	872	1,821	Sentiment Classification	Movie Review (short)
FST	1,413	159	396	Sentiment Classification	Finance Document
WMT14	4.5M	3,000	N/A	Machine Translation	General
JRC-Acquis	2M	1,000	1,000	Machine Translation	Law
Tanzil	579k	1,000	1,000	Machine Translation	Koran

Table 1: Statistic of sentiment classification and machine translation datasets, with number of samples in train, dev and test sets. The task name and corresponding domain for each dataset are also included.

Model	Sentiment Classification				Machine Translation			
	Label	Arch.	SST	FST	Label	Arch.	Law	Koran
In Domain	SST / FST	BERT	91.49	97.72	Law / Koran	TF base	38.52	19.49
Victim 1	IMDB	BERT	87.92	74.94	WMT14	TF-base	23.33	9.82
Victim 2	IMDB	RoBERTa	89.40	80.00	WMT14	TF-big	24.33	10.33
Attack$_{s1}$	Victim 1	BERT	90.13	83.59	Victim 1	TF-base	23.82	10.04
Attack$_{s2}$	Victim 2	BERT	90.72	88.76	Victim 2	TF-base	25.48	10.30
Attack$_m$	Victim 1+2	BERT	**91.57**	**90.53**	Victim 1+2	TF-base	**25.74**	**10.48**

Table 2: Experimental results of imitation attack on single and multiple victim models with settings, labels used for training and model architectures (Arch.). Oracle models are trained on the human-annotated datasets in target domain and victim models are trained on corpora from source domain. For all attack experiments, we report the mean results over 5 runs. Attackers using single victim and multiple victims are indicated as Attack$_s$ and Attack$_m$. *In domain* models are trained on human labeled training sets in the target domain, which could arguably be considered as upper bounds of those attack models. This experiment focuses on comparing Victim and Attack.

Locally Simulated Experiments. We first conduct the experiments of imitation attack on local models, shown in Table 2. The models trained on oracle human annotated datasets are much better than the victim models, as the later ones are trained on other domains. All our attack models outperform the original victim models for both classification and translation tasks. We attribute this performance improvement to unsupervised domain adaptation. Ensemble models consistently work better than each of the single model[7]. For SST, although using the same architecture, the attack model trained on the ensemble of two victims surprisingly outperforms the model supervised by oracle labels. This result also outperforms some competitive supervised baselines (Tang et al., 2019; McCann et al., 2017; Zhou et al., 2016). This observation suggests that, in some scenarios, it is possible to achieve decent results only based on some open APIs, without relatively more expensive human annotations. As a result, some annotators could lose their working opportunities of labeling

new datasets, and some API services might lose their market share in new domains or tasks.

Experiments on Commercial APIs. We then demonstrate the vulnerability of some real-world commercial APIs to our IMA approach, in Table 3. For classification task, the attacker uses soft labels, as *i)* these APIs provide such scores and *ii)* attackers using soft labels achieve better performance than hard attacks in our defense experiments (see Table 5). For machine translation, only hard label could be used, as we can only have token sequences without their perplexity scores from commercial APIs. In all attacks on classification and translation APIs, the attackers manage to achieve significantly better results than the corresponding victim models, with frighteningly low costs. Combining two commercial APIs generally improves the performance of the attackers.We observe that commercial APIs work quite poor on FST, as it belongs to a more professional domain. However, the performance of the attacker catches up significantly on FST and achieves an averaged accuracy of 89.82%, given poor competitors (victims) both with accuracy less than 70%. Google+Bing on Law is the only ensem-

---

[7]More discussion on averaging strategies for ensemble the classification APIs is discussed in Appendix B.

Task	# Queries	API	Cost	Victim	Attacker	Improv.
SST	9,613	**Google Cloud**	$5	84.62	88.26 ± 0.22	+3.64
		IBM Cloud	Free	87.26	89.17 ± 0.33	+1.91
		Google+IBM	$5	-	89.75 ± 0.58	
FST	1,968	**Google Cloud**	Free	68.35	83.85 ± 1.05	+15.50
		IBM Cloud	Free	58.73	85.01 ± 0.81	+26.28
		Google+IBM	Free	-	89.82 ± 0.81	
Law	2M	**Google Trans**	$6,822	30.43	31.99 ± 0.05	+1.56
		Bing Trans	$3,396	34.22	35.45 ± 0.09	+1.23
		Google+Bing	$10,218	-	34.94 ± 0.11	
Koran	579k	**Google Trans**	$1,211	14.31	14.63 ± 0.06	+0.32
		Bing Trans	$590	13.24	13.71 ± 0.05	+0.47
		Google+Bing	$1,801	-	15.25 ± 0.09	

Table 3: A comparison of the commercial APIs (Victims) with attackers. The improvement (Improv.) of attackers over victims is given by rows of single models. The estimated cost of the API is based on the price queried in a single day. For all attack experiments, we report mean and standard deviations of the results over 5 runs.

ble model that fails to surpass all the single model. We attribute this to the fact that Bing Trans and its attack model has already achieved a decent result on Law, outperforming Google Trans with a gap of 3% to 4%.

Estimated Attack Costs. In Table 3, we also estimate the cost of querying the commercial APIs as victim models. We find the costs are quite affordable to many companies or even individuals, given the benefit of obtaining high-quality in-domain classification and MT systems. The price is in accordance with retrieving the results in a day, using a single month budget of a single account. The price could be further decreased, by registering more accounts or using more time. On the other hand, we estimate that the costs for human to annotate the datasets are $480.65 (**SST**), $98.4 (**FST**), $1.6M (**Law**), and $463k (**Koran**), if we hire annotators from Amazon Mechanical Turk. The price is decided as 0.05 cents for each classification sample and 0.8 dollar for each translation sample[8]. Although the price is arguably floating, we give a preliminary overview on the costs by human annotators. We find that the cost of human annotation could be 20 to 150 times higher than querying APIs. This could become another motivation for attackers to learn from APIs, instead of human annotators.

Impact of Model Ensemble. We conduct an

Model	#V	SST		Law	
		Victim	Attack	Victim	Attack
Model A	1	85.39	87.37	20.75	22.60
Model B	1	84.51	86.22	21.01	21.67
Model C	1	86.44	87.70	20.79	21.76
Model D	1	86.60	87.31	20.68	21.80
Model A+B	2	-	88.30	-	22.87
Model A+B+C+D	4	-	89.18	-	22.97
Model Full	1	87.92	88.58	23.33	23.82

Table 4: The comparison of imitation attacks with different model ensembles.

ablation study on the potential performance improvement of attacking multiple victim models, in Table 4. The source domain training datasets are equally separated into 4 disjoint subsets, coined A, B, C and D. Then, we train 4 corresponding local victim models, model A to D, respectively. Model Full uses the victim models trained on the complete training datasets in the source domains. The attacker chooses to utilize the combined results of these victim models for training, *e.g.*, Model A+B ensembles knowledge from victim A and B. We use BERT for classification and Transformer-base for machine translation in this study. Ensemble models with more victims generally improve the attacker performance on SST and Law. The ensemble of multiple weaker models can catch up with those of full model.

Defense Strategies.

We compare two possible defense strategies on classification tasks in Table 5. We consider per-

---

[8]In our preliminary experiment, annotators manage to finish about 10 classification and 1.5 translation annotations. The corresponding wages are about $30/hr and $32/hr, higher than the minimum wage in USA.

Model	SST					FST				
	Soft	Hard	P 0.1	P 0.2	P 0.5	Soft	Hard	P 0.1	P 0.2	P 0.5
Victim 1	87.92	87.92	87.94	86.79	78.06	74.94	74.94	75.24	69.27	58.33
Victim 2	89.40	89.40	88.79	87.73	80.71	80.00	80.00	78.08	76.46	65.42
$Attack_{s1}$	90.44	88.58	90.23	90.07	87.98	82.03	80.25	83.65	83.75	76.41
$Attack_{s2}$	90.12	90.12	90.49	90.47	88.72	87.85	85.57	88.86	85.16	82.03
$Attack_m$	91.82	90.66	91.44	91.20	90.15	88.86	87.09	89.27	87.34	86.33

Table 5: The comparison of imitation attack results given victims with various defense strategies, soft label (Soft) to hard label (Hard) and noise perturbation (P) with variance (σ).

turbing the original soft outputs by hard labeling or adding Gaussian noise $\boldsymbol{n} \sim \mathcal{N}(0, \sigma^2)$, as

$$h'_v(x) = h_v(x) + \boldsymbol{n}.$$

The models utilizing hard labels manage to consistently reduce the performance of the attackers who use soft labels. Then we compare models trained on labels with various perturbation (P) by sampling random noise from Gaussian distribution with variance $\sigma \in \{0.1, 0.2, 0.5\}$ (Tramèr et al., 2016). These experiments allow the noise to flip the outputs of the original victim models, and more experiments on perturbation without influencing utility of victim models are provided in Appendix C. We observe that gently disturbing the outputs of victim models could crack down the attackers to some extent and larger noise indicates better defense. However, the harm to victims is generally larger than those to attackers. We attribute this to noise reduction in training attack model. Overall, our new IMA calls for more effective defense methods in future.

6   Discussion

We consider our imitation attack approach has achieved outperforming results that challenge the current understanding of IMA. As a result, API publishing strategies and defense methodologies should be converted accordingly.

Suggested Actions. As in domain adaptation settings, IMA manages to achieve superb performance, while attacking models are not able to outperform victims in the same domain (Krishna et al., 2019; Wallace et al., 2020), we suggest API services could cover more domains to eliminate the potential performance gain from UDA. Simply harming the utility of victim models seems not to be a wise choice for service providers, but merely providing hard labels without probability scores could avoid more superior attack performance. Adjusting

the pricing strategies of publishing API services or adding watermark to API outputs (He et al., 2021b) could be plausible solutions to avoid the illegal stealing of the precious APIs of industrial companies.

Ethical Concerns. We recognize that our work could be used for malicious purposes, for example, a competing company may adopt our attack to steal a better model for commercial benefits, thus eroding the market shares of other business competitors. However, the main purpose of our work is to help commercial cloud services and regulators raise awareness of model theft, and reconsider how to deploy NLP APIs in a safe manner to avoid being stolen and exceeded. In order to minimize the potential negative influence of our work, *i)* we will delete our models and retrieved results on our local server; *ii)* we will report the vulnerability to the API providers.

Follow-up Attacks. Recent works have demonstrated that the extracted model could be used as a reconnaissance step to facilitate later attacks (He et al., 2021a; Krishna et al., 2019; Wallace et al., 2020). For instance, the adversary could use the extracted model to facilitate private information inference about the training data of the victim model, or to construct adversarial examples that force the victim model to make incorrect predictions. We leave follow-up attacks that can leverage our imitated model with better performance to future works.

7   Conclusion

We demonstrate a powerful imitation attack paradigm which may produce better attack models that surpasses the imitated models including the real-world NLP APIs via domain adaptation and ensemble. We believe such achievements would potentially influence the price decision and publishing strategies of primary NLP services. We also take the first step of grounding our new attacking approach with unsupervised domain adaptation the-

ory and model ensemble. More broadly, we hope to arouse prominent concerns on security and privacy of API services in NLP applications. In future, we plan to further explore the possibility of detecting and defending the imitation attack.

References

Rohan Anil, Gabriel Pereyra, Alexandre Passos, Robert Ormandi, George E Dahl, and Geoffrey E Hinton. 2018. Large scale distributed neural network training through online distillation. *arXiv preprint arXiv:1804.03235*.

Eric Bauer and Ron Kohavi. 1999. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine learning*, 36(1):105–139.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine learning*, 79(1):151–175.

Shai Ben-David, John Blitzer, Koby Crammer, Fernando Pereira, et al. 2007. Analysis of representations for domain adaptation. *Advances in neural information processing systems*, 19:137.

V. Besnier, H. Jain, A. Bursuc, M. Cord, and P. Pérez. 2020. This dataset does not exist: Training models from generated images. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA. Association for Computational Linguistics.

Leo Breiman. 2001. Random forests. *Machine learning*, 45(1):5–32.

Cristian Buciluá, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of KDD*, pages 535–541. ACM.

Kevin Clark, Minh-Thang Luong, Urvashi Khandelwal, Christopher D Manning, and Quoc Le. 2019. Bam! born-again multi-task networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5931–5937.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263, Prague, Czech Republic. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Nikita Dvornik, Cordelia Schmid, and Julien Mairal. 2019. Diversity with cooperation: Ensemble methods for few-shot classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3723–3731.

Tommaso Furlanello, Zachary C Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. 2018. Born again neural networks. *arXiv preprint arXiv:1805.04770*.

Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR.

Shangwei Guo, Chunlong Xie, Jiwei Li, Lingjuan Lyu, and Tianwei Zhang. 2022. Threats to pre-trained language models: Survey and taxonomy. *arXiv preprint arXiv:2202.06862*.

Xuanli He, Lingjuan Lyu, Qiongkai Xu, and Lichao Sun. 2021a. Model extraction and adversarial transferability, your bert is vulnerable! *arXiv preprint arXiv:2103.10013*.

Xuanli He, Qiongkai Xu, Lingjuan Lyu, Fangzhao Wu, and Chenguang Wang. 2021b. Protecting intellectual property of language generation apis with lexical watermark. *arXiv preprint arXiv:2112.02701*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Kalpesh Krishna, Gaurav Singh Tomar, Ankur P Parikh, Nicolas Papernot, and Mohit Iyyer. 2019. Thieves on sesame street! model extraction of bert-based apis. In *International Conference on Learning Representations*.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. *arXiv preprint arXiv:2003.02245*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *arXiv preprint arXiv:2001.08210*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Lingjuan Lyu, Xuanli He, and Yitong Li. 2020. Differentially private representation for nlp: Formal guarantee and an empirical study on privacy and fairness. In *EMNLP 2020, Findings*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Pekka Malo, Ankur Sinha, Pekka Korhonen, Jyrki Wallenius, and Pyry Takala. 2014. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology*, 65(4):782–796.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: contextualized word vectors. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6297–6308.

Timothy Miller. 2019. Simplified neural unsupervised domain adaptation. In *Proceedings of the conference. Association for Computational Linguistics. North American Chapter. Meeting*, volume 2019, page 414. NIH Public Access.

David Opitz and Richard Maclin. 1999. Popular ensemble methods: An empirical study. *Journal of artificial intelligence research*, 11:169–198.

Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. 2019. Knockoff nets: Stealing functionality of black-box models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4954–4963.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Suman Ravuri and Oriol Vinyals. 2019. Classification accuracy score for conditional generative models. In *Advances in Neural Information Processing Systems*, pages 12268–12279.

Fahimeh Saleh, Wray Buntine, and Gholamreza Haffari. 2020. Collective wisdom: Improving low-resource neural machine translation using adaptive knowledge distillation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3413–3421.

Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. 2008. *Introduction to information retrieval*, volume 39. Cambridge University Press Cambridge.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Xu Tan, Yi Ren, Di He, Tao Qin, Zhou Zhao, and Tie-Yan Liu. 2019. Multilingual neural machine translation with knowledge distillation. *arXiv preprint arXiv:1902.10461*.

Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019. Distilling task-specific knowledge from bert into simple neural networks. *arXiv preprint arXiv:1903.12136*.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in opus. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218.

Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. 2016. Stealing machine learning models via prediction apis. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*, pages 601–618.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Eric Wallace, Mitchell Stern, and Dawn Song. 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)*, pages 5531–5546.

Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, and Tao Qin. 2021. Generalizing to unseen domains: A survey on domain generalization. *arXiv preprint arXiv:2103.03097*.

Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. 2020. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10687–10698.

Peng Zhou, Zhenyu Qi, Suncong Zheng, Jiaming Xu, Hongyun Bao, and Bo Xu. 2016. Text classification improved by integrating bidirectional lstm with two-dimensional max pooling. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3485–3495.

A Hyper-Parameter Settings

In order to have a fair and consistent comparison of experiments, we utilize the same hyper-parameters for the same task, as demonstrated in Table 6. They are decided by our preliminary experiments on target domains. All experiments could be conducted on a single RTX 6000 GPU.

	Sent.	MT
Learning rate	1e-05	5e-04
Batch size	16 sentences	32k tokens
Optimizer	Adam	Adam
Epoch	50	40
Max length	256	1024
Warm-up	-	4000 steps

Table 6: Hyper-parameter used for sentiment analysis (Sent.) and machine translation (MT).

B Comparison of Ensemble Strategy

In this section, we compare two ensemble methods for sentiment classification, i) concatenating the training samples (Concat.) and ii) averaging the prediction scores (Avg.). Both ensemble strategies are competitive to the other, as demonstrated in Table 7. As we are not able to acquire scores of each token from MT APIs, we cannot average the results of MT. In our paper, to have a consistent comparison, Concate. is used in all our ensemble experiments on both classification and translation tasks.

Method	Ensemble	Accuracy
BERT+RoBERTa	Concat.	91.57 ± 0.27
BERT+RoBERTa	Avg.	91.58 ± 0.39
Google+IBM	Concat.	89.75 ± 0.58
Google+IBM	Avg.	89.76 ± 0.42

(a) Ensemble results on SST.

Method	Ensemble	Accuracy
BERT+RoBERTa	Concat.	90.53 ± 0.38
BERT+RoBERTa	Avg.	90.89 ± 0.48
Google+IBM	Concat.	89.82 ± 0.81
Google+IBM	Avg.	88.86 ± 1.22

(b) Ensemble results on FST.

Table 7: The comparison of imitation attack on multiple victims using concatenate samples (Concat.) and average scores (Avg.).

C Comparison of Defense Strategies

In this section, we discuss the perturbation methods. Given an input sentence x, the probability score by the victim model is $h_v(x)$. To compare the influence of API performance on attack model, we sample a noise vector n from Gaussian distribution with a variance of σ, *i.e.*, $n \sim \mathcal{N}(0, \sigma^2)$. The perturbed prediction function is calculated as:

$$h'_v(x) = h_v(x) + n$$

It is worth noting that original victim model prediction is $y = \operatorname{argmax}(h_v(x))$, therefore, injecting n could lead to different prediction $y' = \operatorname{argmax}(h'_v(x))$. Consequently, the utility of victim model can be corrupted as demonstrated in Table 5. However, such compromise can cause finance and reputation losses to the API providers in real-world. To avoid these adverse effects, API providers can adopt a label-preserved policy, where the injected noise n should not flip the originally predicted labels. In other words, another noise will be sampled, if currently sampled noise changes the prediction of the original model. The results of such defense strategy is shown in Table 8. As this setting is more conservative, the performance of this defense is in between the performance of hard label and soft label.

Model	SST					FST				
	Soft	Hard	P 0.1	P 0.2	P 0.5	Soft	Hard	P 0.1	P 0.2	P 0.5
Victim 1			87.92					74.94		
Victim 2			89.40					80.00		
Attack_{s1}	90.44	88.58	90.48	89.96	89.35	82.03	80.25	82.84	81.92	81.72
Attack_{s2}	90.12	90.12	90.30	90.02	89.23	87.85	85.57	88.10	87.29	84.41
Attack_m	91.82	90.66	91.24	91.38	91.11	88.86	87.09	89.37	88.25	87.59

Table 8: The comparison of imitation attack results given victims with various defense strategies, soft label (Soft) to hard label (Hard) and noise perturbation (P) with variance (σ). The predicted labels of victim models are not flipped in the experiment.