Unsupervised Non-transferable Text Classification

Guangtao Zeng and **Wei Lu**
StatNLP Research Group
Singapore University of Technology and Design
guangtao_zeng@mymail.sutd.edu.sg, luwei@sutd.edu.sg

Abstract

Training a good deep learning model requires substantial data and computing resources, which makes the resulting neural model a valuable intellectual property. To prevent the neural network from being undesirably exploited, non-transferable learning has been proposed to reduce the model generalization ability in specific target domains. However, existing approaches require labeled data for the target domain which can be difficult to obtain. Furthermore, they do not have the mechanism to still recover the model's ability to access the target domain. In this paper, we propose a novel unsupervised non-transferable learning method for the text classification task that does not require annotated target domain data. We further introduce a *secret key* component in our approach for recovering the access to the target domain, where we design both an *explicit* and an *implicit* method for doing so. Extensive experiments demonstrate the effectiveness of our approach.

1 Introduction

Deep learning has achieved remarkable success over the past decade and is active in various fields, including computer vision, natural language processing (NLP), and data mining. Although neural models can perform well in most tasks, they require a huge amount of data and a high computation cost to train, making the trained model a valuable intellectual property. As a result, it is essential for us to prevent the neural models from being used without authorization. In the last few years, many methods have been proposed to safeguard deep neural networks and they can be roughly divided into two types: *watermarking* (Adi et al., 2018), and *secure authorization* (Alam et al., 2020). In the watermarking approaches, the owners can verify the ownership of the neural model based on a unique watermark. However, due to the catastrophic forgetting problem (Kemker et al., 2018), the watermark-based neural models (Kuribayashi et al., 2020; Song et al., 2017) are known to be vulnerable to certain malicious attacks (Wang and Kerschbaum, 2019), which may lead to the loss of their watermarks. On the other hand, in the secure authorization approaches, the owners of the neural network want to ensure that users can only access the model with authorization. Recently, Wang et al. (2022) proposed a new perspective with non-transferable learning (NTL) to protect the model from illegal access to unauthorized data. The method trains the model to have a good performance only in the authorized domain while performing badly in the unauthorized domain. However, such an approach has some limitations: 1) their work relies on a significant amount of labeled data from the target domain, while such labels are usually not easy to acquire, 2) the access to the unauthorized domain can no longer be regained, if required, after the model is learned.

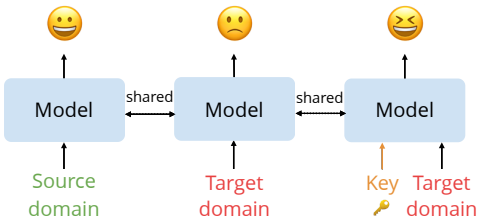


Figure 1: Overview of our unsupervised non-transferable with secret keys.

In this work, we propose our new NTL method named Unsupervised Non-Transferable Learning (UNTL) for the text classification tasks. As Figure 1 shows, our model can perform well in the source domain while performing badly in the target domain. In addition, we propose secret key modules, which can help recover the ability of the model in the target domain. Our contributions include:

- We propose a novel unsupervised non-transferable learning approach for text classification tasks. Different from existing ap-

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 10071–10084
December 7-11, 2022

proaches, our model can still perform well without the need of the label information in the target domain.

- We introduce two different methods, namely *Prompt-based Secret Key* and *Adapter-based Secret Key*, that allow us to recover the ability of the model to perform classification on the target domain.
- Extensive experiments show that our proposed models can perform well in the source domain but badly in the target domain. Moreover, the access to the target domain can still be regained using the secret key.

To the best of our knowledge, our work is the first approach for learning under the unsupervised non-transferable learning setup, which also comes with the ability to recover the access to the target domain.[1]

2 Related Work

In this section, we briefly survey ideas that are related to our work from two fields: domain adaptation and intellectual property protection. Furthermore, we discuss some limitations in the existing methods which we will tackle with our approach.

In domain adaptation, given a source domain and a target domain with unlabeled data or a few labeled data, the goal is to improve the performance in the target task using the knowledge from the source domain. Ghifary et al. (2014), Tzeng et al. (2014), and Zhu et al. (2021) applied a Maximum Mean Discrepancy regularization method (Gretton et al., 2012) to maximize the invariance information between different domains. Ganin et al. (2016) and Schoenauer-Sebag et al. (2019) tried to match the feature space distributions of the two domains with adversarial learning. In contrast to the methods above, Wang et al. (2022) analyzed domain adaptation in a different way and proposed non-transferable learning (NTL) to prevent the knowledge transfer from the source to the target domain by enlarging the discrepancy between the representations in different domains.

In intellectual property protection, due to the significant value and its vulnerability against malicious attacks of learned deep neural networks, it is crucial to propose intellectual property protection methods to defend the owners of the deep neural networks (DNNs) from any loss. Recently, two different approaches to safeguard DNNs have been proposed: watermarking (Adi et al., 2018) and secure authorization (Alam et al., 2020). In the watermarking approaches, researchers designed a digital watermark that can be embedded into data such as video, images, and so on. With the detection of the unique watermark, we could verify the ownership of the copyright of the data. Based on these ideas, Song et al. (2017) and Kuribayashi et al. (2020) embedded the digital watermarks into the parameters of the neural networks. Zhang et al. (2020) and Wu et al. (2021) proposed a framework to generate images with an invisible but extractable watermark. However, they are vulnerable to some active attack algorithms (Wang and Kerschbaum, 2019; Chen et al., 2021) which first detect the watermark and then rewrite or remove it. On the other hand, the secure authorization approach seeks to train a model that generates inaccurate results without authorization. Alam et al. (2020) proposed a key-based framework that ensures correct model functioning only with the correct secret key. In addition, Wang et al. (2022) were inspired by domain generalization and proposed non-transferable learning (NTL), which achieves secure authorization by reducing the model's generalization ability in the specified unauthorized domain.

Although the NTL model can effectively prevent the access to the unauthorized domain, it requires target labels during training, which may not always be easy to obtain. Furthermore, there is no mechanism to recover the access to the unauthorized domain when needed. In this paper, we present a new NTL model and show that our model can still have a good performance even in the absence of the target labels which are, however, indispensable in the work of Wang et al. (2022). Besides, we extend it to a secret key-based version. With our method, the authorized users can still access the target domain with the provided keys.

3 Approach

In this section, we first introduce our proposed Unsupervised Non-Transferable Learning (UNTL) approach in Sec. 3.1, followed by a discussion on its practical limitation – it lacks the ability to regain the access to the target domain. Next, we discuss our secret key-based methods in Sec. 3.2 to address this limitation.

[1] Our code and data are released at https://github.com/ChaosCodes/UNTL.

3.1 UNTL Text Classification

Problem Description First of all, we present our definition of the unsupervised non-transferable learning task without labeled data from the target domain. Following Farahani et al. (2020), we consider that a domain consists of three parts: input space X, label space Y, and the joint probability distribution $p(X, Y)$. Given a source domain $\mathcal{D}_s = \{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{N_s}$ and a target domain $\mathcal{D}_t = \{\boldsymbol{x}_j\}_{j=1}^{N_t}$ with unlabeled samples, where $\boldsymbol{y}_i \in \mathbb{R}^C$ is a one-hot vector indicating the label of $\boldsymbol{x}_i$, C is the number of classes, and N_s, N_t refer to the number of examples in the source and target domain respectively. The goal of our UNTL method is to prevent the knowledge transfer from the source to the target domain, i.e., to train the model so that it performs well on the source domain data but poorly on the target domain data, without the requirement of accessing the label information of the target data.

Text Classification In our work, we use a BERT-based model (Devlin et al., 2019) ψ as our feature extractor for the input sentence and consider the final hidden state h of the token `[CLS]` as the feature representation, where we denote h as $\psi(\boldsymbol{x})$. A simple feed-forward network $\mathrm{FFN}(\cdot)$ will be added on top of BERT as a classifier to predict the label. The formal loss function can be:

$$\mathcal{L}_{\mathrm{CE}} = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_s}[\mathrm{CE}\left(\mathrm{FFN}\left(\psi\left(\boldsymbol{x}\right)\right), \boldsymbol{y}\right)] \quad (1)$$

where $\mathcal{D}_s$ is the source domain dataset and CE indicates the cross entropy function.

Maximum Mean Discrepancy To enlarge the distance between the representations of the source and target domains, we follow Wang et al. (2022) and use Maximum Mean Discrepancy (Gretton et al., 2012) (MMD) to achieve this goal. MMD is a kernel two-sample test and can be used as a metric to determine whether two data distributions p and q are similar. MMD defines the metric function as follows:

$$d_{p,q} = ||\mathbb{E}_{\boldsymbol{x}\sim p}[\psi\left(\boldsymbol{x}\right)] - \mathbb{E}_{\boldsymbol{x}'\sim q}[\psi\left(\boldsymbol{x}'\right)]||^2_{\mathcal{H}_k} \quad (2)$$

where $\mathcal{H}_k$ is the reproducing kernel Hilbert space (RKHS) with a kernel k, whose operation is $k\left(\boldsymbol{z}, \boldsymbol{z}'\right) = e^{-||\boldsymbol{z}-\boldsymbol{z}'||^2}$ and function ψ maps the sentence input into RKHS. The smaller the distance $d_{p,q}$, the more similar the two distributions p and q.

In our work, we use MMD to increase the distance between the feature representations of the source and the target domain, forcing the feature extractor ψ to extract domain-dependent representations rather than maximizing the inter-domain invariance. To prevent the high MMD from dominating the entire loss, we follow Wang et al. (2022) and set an upper bound for it. Therefore, based on Equation 2, our MMD loss can be formulated as:

$$\mathcal{L}_{\mathrm{MMD}}\left(\mathcal{S}, \mathcal{T}\right) = -\min\left(c, d_{\mathcal{S},\mathcal{T}}\right) \quad (3)$$

where c is the upper bound for MMD, and $\mathcal{S}, \mathcal{T}$ are data distributions of the source and target domains respectively. With this loss, we only maximize $d_{\mathcal{S},\mathcal{T}}$ when it is smaller than the upper bound c.

Domain Classifier Despite being able to enlarge the gap between the source and target domains to some extent, the MMD loss lacks the explicit ability to clearly draw the boundary between the representations of different domains, especially when the knowledge between domains is similar. Therefore, we hypothesize that using MMD alone may not be sufficient to yield the optimal empirical performance. To mitigate this issue, we draw inspirations from the Domain-Adversarial Neural Networks (Ganin et al., 2016) and propose an additional domain classifier added on top of the feature extractor. This classifier is trained to predict the domain with the feature representations. We employ a cross-entropy loss to train the domain classifier. By optimizing this loss, the representations of different domains are encouraged to be more distinct. Specifically, we use 0 to indicate the source domain and 1 to indicate the target domain. We can formulate the domain classification (DC) loss as:

$$\begin{aligned}\mathcal{L}_{\mathrm{DC}}\left(\mathcal{S}, \mathcal{T}\right) = \mathbb{E}_{\boldsymbol{x}^S\sim\mathcal{S}}[\mathrm{CE}\left(\mathrm{FFN}_{dc}\left(\psi\left(\boldsymbol{x}^S\right)\right), 0\right)] \\ + \mathbb{E}_{\boldsymbol{x}^T\sim\mathcal{T}}[\mathrm{CE}(\mathrm{FFN}_{dc}\left(\psi\left(\boldsymbol{x}^T\right)\right), 1)]\end{aligned} \quad (4)$$

where FFN_{dc} is the domain classifier. With this DC loss as a regularization term, the boundary of feature representation between the source and the target can be clearer, facilitating the non-transferable learning.

Objective Function In this task, our goal is to train a model that can perform well on the source domain while performing badly on the target domain. To achieve this goal, we propose a loss function for the unsupervised non-transferable learning, which contains three terms. The first term is the cross-entropy loss $\mathcal{L}_{\mathrm{CE}}$ for text classification to integrate knowledge about the the downstream task

into the model. The second term is the domain classification loss $\mathcal{L}_{\text{DC}}$ and the third is MMD loss $\mathcal{L}_{\text{MMD}}$. The latter two terms jointly contribute to enlarging the gap between the representations of the source and target domains to prevent the knowledge transfer. Finally, we can get our overall loss which is written as:

$$\mathcal{L}_{\text{UNTL}} = \mathcal{L}_{\text{CE}} + \beta \cdot \mathcal{L}_{\text{DC}} + \lambda \cdot \mathcal{L}_{\text{MMD}} \quad (5)$$

where β and λ are the scaling hyperparameters.

Theoretical Analysis Different from (Wang et al., 2022) where they use information bottleneck theory (Tishby et al., 2000) to show the feasibility of non-transferable learning, we turn to a more general theory of domain adaptation (Ben-David et al., 2006; Wang, 2018). Here, we present an analysis of the effectiveness of the unsupervised setting based on this theory.

Theorem 1 *(Ben-David et al., 2010) Let $\mathcal{H}$ be a hypothesis space (of a particular VC dimension), for any $h \in \mathcal{H}$. Given a source domain $\mathcal{D}_{\mathcal{S}}$ and a target domain $\mathcal{D}_{\mathcal{T}}$:*

$$\epsilon_{\mathcal{T}}(h) \leq \epsilon_{\mathcal{S}}(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_{\mathcal{S}}, \mathcal{D}_{\mathcal{T}}) + C \quad (6)$$

where $\epsilon_{\mathcal{S}}(h)$ and $\epsilon_{\mathcal{T}}(h)$ are the expected source and target errors respectively, $C = \min_{h' \in \mathcal{H}} (\epsilon_{\mathcal{S}}(h') + \epsilon_{\mathcal{T}}(h'))$, which can be viewed as a constant and $d_{\mathcal{H}\Delta\mathcal{H}}$ is a divergence[2] that measures the maximal discrepancy between two distributions under a fixed hypothesis class.

During our training process, we minimize the source error while maximizing the divergence (with the MMD and DC losses). Comparing with a baseline transfer model without the MMD and DC losses, we hypothesize the our method may yield a comparable source error, while leading to a significant larger divergence term. We believe this may lead to a significantly increase in the target error, as the changes of the above terms would effectively lead to a much looser upper bound for the target error as shown in Equation 6. Such a looser upper bound may lead to significant increase in target error, which can effectively prevent the knowledge from being transferred into the target domain. We will verify our hypothesis in the experiments later.[3]

[2] $d_{\mathcal{H}\Delta\mathcal{H}}$ is a symmetric difference hypothesis space for a hypothesis space $\mathcal{H}$. See Ben-David et al. (2010) for more details.

[3] In fact, through our experiments later, we found that on average there was a 1% increase in source error for our approach as compared to the baseline. However, there was a significant increase ($\times 10$) in the divergence term as approximated by the MMD loss, which leads to effective non-transfer learning (where we achieve good source domain performance and bad target domain performance).

3.2 Learning with Secret Keys

With our UNTL method, we could ensure that the model performs well in the source domain whilst degrading its performance in the target domain. However, it is inconvenient if the performance in the target domain can no longer be restored after training. This can be illustrated with an example: suppose that we are running an application which supports two kinds of users, regular users and members. Suppose further that the regular users are only authorized to query the model for a limited set of data, while the members have no limits on their access to the model. Using our UNTL approach discussed above, we can limit the access of the regular users by denoting the authorized and non-authorized portions of the data as the source and target domains respectively. Then we train the model that performs well on the source domain, but poorly on the target domain. However, as the members have no limits to their access, they would require a separate model to be trained that performs well on both domains, thus doubling the computational and storage costs required for the application.

To solve this issue, we extend our approach to include a secret key, K, that can be used to recover access to the target domain even after non-transferable learning. Without the key, the model is encouraged to perform well on the source domain while degrading the accuracy in the target domain. However, upon supplying the secret key, the model's performance on the target domain will be restored. Following the example above, this allows a single neural network to be used for all the users, whilst providing privileged access to the members through the provision of the secret key. Based on our UNTL, in this section, we present our innovative secret key-based unsupervised non-transferable learning method. The method can not only keep the normal users away from the membership area but also provide members with the specific key that allow them to access the membership area within a single model.

Our intuition is to design a secret key that can revive the restricted target domain in our UNTL model. We call the method Secret Key-based Non-Transferable Learning, which has two variants: 1)

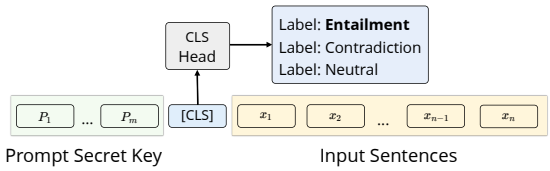


Figure 2: Prompt-based Secret Key Methods

Prompt-based Secret Key method, where we add a discrete prompt as a prefix to the input sentence that serves as an explicit secret key to restore the access to the target domain, and 2) *Adapter-based Secret Key method*, where a trained adapter module is added to the model to transform the target embeddings into the source-like ones in an implicit manner.

Prompt-based Secret Key Recently, prompt-based learning (Schick and Schütze, 2021; Lester et al., 2021) has achieved state-of-the-art results in many tasks. Inspired by these prompt-based methods, we consider a prompt as our secret key, which users can use to access the target domain. As shown in Figure 2, we first assign a randomly chosen prompt $P = \{P_1, ..., P_m\}$ as the secret key, where P_i is the i-th token in the prompt sentence and m is the length of the prompt. Given an input sentence $\boldsymbol{x}$ with length n, we concatenate the prompt with the input $\boldsymbol{x}$ to construct an authorized input sentence. In addition, similar to inference without prompt key, we continue using the hidden representation at the position of `[CLS]` as the input of the task classifier and get the predicted label.

With the introduction of the prompt-based key, we believe that there are 3 different distributions in this task: *source* domain, *target* domain and *target+prompt* domain. In the prompt-based secret key model, after prepending the specific prompt to the target input, the model can recover the ability to perform well in the target domain. Therefore, we try to train the feature extractor to close the distance between the target+prompt domain and the source domain while enlarging the distance between the source and the target domain without the key. To achieve this, we propose a new MMD loss:

$$\mathcal{L}'_{\text{MMD}}\left(\mathcal{P}, \mathcal{S}, \mathcal{T}\right) = \alpha \cdot d_{\mathcal{P},\mathcal{S}} - \min\left(c, d_{\mathcal{S},\mathcal{T}}\right) \quad (7)$$

where $\mathcal{P}$ denotes the data distribution of the target+prompt domain, α is the scaling hyperparameter and c is the upper bound for MMD.

In this way, we can transfer the knowledge from the source domain to the target+prompt domain but not to the original target domain. Therefore, we can extend Equation 5 to get the objective function for prompt-based secret key UNTL method:

$$\begin{aligned}\mathcal{L}_{\text{prompt}} =& \mathcal{L}_{\text{CE}} + \beta \cdot \mathcal{L}_{\text{DC}}\left(\left[\mathcal{P}, \mathcal{S}\right], \mathcal{T}\right) + \\ & \lambda \cdot \mathcal{L}'_{\text{MMD}}\left(\mathcal{P}, \mathcal{S}, \mathcal{T}\right)\end{aligned} \quad (8)$$

where $[\mathcal{P}, \mathcal{S}]$ indicates the data distribution of the combined domain of source and target+prompt.

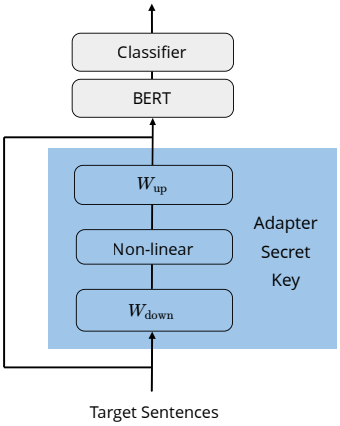


Figure 3: Adapter-based Secret Key Methods

Adapter-based Secret Key Besides explicitly prepending the input sentences with the discrete prompt as the secret key, we could also consider adding an input adapter (Houlsby et al., 2019; An et al., 2022) as the secret key. In our UNTL model, input sentences in different domains will lead to distinct performance. Intuitively, we train the input adapter to eliminate the target-like feature and convert the target embeddings into the source-like embeddings. Given an embedding representation, we assume it has two components: the semantic component which is essential for text classification, and the domain component that is irrelevant to the task. We train the adapter to convert the domain component of the embedding from the target domain to the source domain while maintaining the semantic component.

The adapter architecture is shown in Figure 3. In the figure, embeddings of the target sentences are first projected into a lower dimensional space $\mathbb{R}^m$ with W_{down} before passing through a ReLU nonlinear function, and then projected back to the space $\mathbb{R}^d$ with W_{up}, where m is significantly less than d (in addition, there is a skip connection as shown in Figure 3). With the input adapter module, the target embeddings will be transformed into the source-like ones. From this adapter-based network,

Datasets	SL	TE	GO	TR	FI
#Train	68,716	74,087	68,755	68,755	68,753
#Valid	8,590	9,261	8,595	8,595	8,595
#Test	1,955	1,966	1,945	1,976	1,973

Table 1: Dataset statistics

we can get source-like embeddings data in the *target+adapter* domain which can be denoted as $\mathcal{D}_{\mathcal{A}}$. Similar to the prompt-based secret key method, we inject the knowledge from the source domain into the target+adapter domain by closing the distance $d_{\mathcal{A},\mathcal{S}}$ between their representations.

However, directly using the UNTL loss above could not make sure that the adapter can maintain the task-dependent information. Therefore, we construct a dataset $\{\text{adapter}(\boldsymbol{x}_i), \boldsymbol{y}_i\}_{i=1}^{N_s}$ with the source domain data, where $\text{adapter}(\boldsymbol{x})$ indicates the representation of $\boldsymbol{x}$ converted by the adapter module. Then we train the model with an additional cross-entropy loss to guarantee that the embeddings converted by the adapter can contain sufficient signals about the classification task:

$$\mathcal{L}_{\text{CE}_{\text{adapter}}} = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_s}\left[\text{CE}\left(\text{FFN}\left(\psi\left(\text{adapter}(\boldsymbol{x})\right)\right), \boldsymbol{y}\right)\right] \tag{9}$$

The overall objective function for training the adapter-based secret key model is:

$$\begin{aligned}\mathcal{L}_{\text{adapter}} =& \mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{CE}_{\text{adapter}}} + \beta \cdot \mathcal{L}_{\text{DC}}\left([\mathcal{A}, \mathcal{S}], \mathcal{T}\right) \\ &+ \lambda \cdot \mathcal{L}'_{\text{MMD}}\left(\mathcal{A}, \mathcal{S}, \mathcal{T}\right)\end{aligned} \tag{10}$$

where $[\mathcal{A}, \mathcal{S}]$ indicates the data distribution of the combined domain of source and target+adapter.

4 Experiments

In this section, we first conduct experiments to verify the effectiveness of our UNTL method and then show that our proposed secret key-based UNTL approach could recover access to the target unauthorized domain when the secret key is supplied in the form of a discrete prompt or an additional adapter module. We use MultiNLI (Williams et al., 2018) as our benchmark dataset, which is for a 3-class classification task with balanced labels. We begin with some training details of all experiments and then discuss the results in different settings.

4.1 Experimental Setup

Our models are implemented in PyTorch (Paszke et al., 2019) and all experiments are conducted with

Src\Tgt	SL	TE	GO	TR	FI
SL	$73.8_{\pm 0.6}$	$73.9_{\pm 0.4}$	$80.1_{\pm 0.2}$	$76.6_{\pm 0.3}$	$76.8_{\pm 0.6}$
TE	$72.1_{\pm 0.4}$	$77.5_{\pm 0.6}$	$77.1_{\pm 0.2}$	$74.0_{\pm 0.3}$	$75.3_{\pm 0.9}$
GO	$71.7_{\pm 0.3}$	$72.0_{\pm 0.3}$	$81.0_{\pm 0.2}$	$75.9_{\pm 0.8}$	$74.1_{\pm 0.5}$
TR	$71.6_{\pm 0.5}$	$71.6_{\pm 0.1}$	$80.4_{\pm 0.3}$	$79.3_{\pm 0.3}$	$73.9_{\pm 0.4}$
FI	$73.4_{\pm 0.6}$	$74.2_{\pm 0.2}$	$79.2_{\pm 0.4}$	$75.4_{\pm 0.4}$	$80.1_{\pm 0.6}$

Table 2: Performance of classification task in each domains when training on the source domain only

Src\Tgt	SL	TE	GO	TR	FI
SL	$71.5_{\pm 1.4}$	$34.2_{\pm 1.1}$	$40.0_{\pm 0.7}$	$34.1_{\pm 1.7}$	$36.6_{\pm 1.5}$
TE	$33.7_{\pm 1.3}$	$76.8_{\pm 0.5}$	$35.4_{\pm 1.8}$	$34.7_{\pm 1.0}$	$32.6_{\pm 0.5}$
GO	$38.0_{\pm 0.8}$	$34.4_{\pm 1.1}$	$81.1_{\pm 0.8}$	$34.1_{\pm 0.1}$	$34.9_{\pm 1.4}$
TR	$36.9_{\pm 3.8}$	$33.4_{\pm 0.1}$	$34.6_{\pm 1.9}$	$78.9_{\pm 0.8}$	$33.6_{\pm 0.1}$
FI	$39.1_{\pm 1.8}$	$34.6_{\pm 1.0}$	$36.0_{\pm 1.8}$	$34.8_{\pm 0.8}$	$78.7_{\pm 1.3}$

Table 3: Performance over the target domain for our unsupervised non-transferable learning

Src\Tgt	SL	TE	GO	TR	FI
SL	$72.7_{\pm 1.1}$	$33.0_{\pm 0.7}$	$38.1_{\pm 1.5}$	$36.2_{\pm 1.3}$	$38.0_{\pm 0.0}$
TE	$34.7_{\pm 0.7}$	$76.7_{\pm 0.7}$	$34.7_{\pm 2.7}$	$34.1_{\pm 1.7}$	$34.5_{\pm 1.1}$
GO	$37.4_{\pm 1.1}$	$32.3_{\pm 0.8}$	$80.8_{\pm 0.4}$	$33.8_{\pm 1.6}$	$33.3_{\pm 0.5}$
TR	$36.2_{\pm 1.6}$	$33.4_{\pm 0.0}$	$35.1_{\pm 2.8}$	$79.2_{\pm 0.6}$	$34.3_{\pm 1.1}$
FI	$38.1_{\pm 1.0}$	$34.0_{\pm 1.8}$	$33.6_{\pm 2.8}$	$34.0_{\pm 1.9}$	$79.0_{\pm 0.7}$

Table 4: Performance over the target domain for non-transferable learning with the label information in the target domain. The results are on par with the UNTL results, which suggests that source labels alone are sufficient for the UNTL task.

NVIDIA Quadro RTX 8000 GPUs and we run three times with different seeds. We use the preprocessed MultiNLI dataset from Huggingface (Lhoest et al., 2021). Based on the genre information, we divide the MultiNLI dataset into 5 parts, namely, slate (SL), telephone (TE), government (GO), travel (TR), and fiction (FI) as different domains. As Huggingface only has a training set and a validation set for MultiNLI, we split the training dataset into 8:1 as the training set and the evaluation set, and consider the validation set as the test set in our experiments. The dataset statistics can be found in Table 1. As for the model architecture, we use the pretrained language model BERT (Devlin et al., 2019) as our feature extractor and a randomly initialized one-layer feed-forward network as the classifier. We use the Adam (Kingma and Ba, 2015) optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$. More details can be found in Appendix A.

4.2 Results for UNTL

We first train a supervised classification model only on the source domain, shown in Table 2, as a base-

Source\Target	SL	TE	GO	TR	FI
SL	$72.8_{\pm 0.9} \Rightarrow 68.6_{\pm 1.3}$	$35.3_{\pm 1.0} \Rightarrow 68.9_{\pm 0.3}$	$38.2_{\pm 1.6} \Rightarrow 74.7_{\pm 0.7}$	$36.9_{\pm 1.7} \Rightarrow 73.1_{\pm 1.0}$	$39.2_{\pm 1.4} \Rightarrow 70.2_{\pm 1.4}$
TE	$33.1_{\pm 1.3} \Rightarrow 65.1_{\pm 1.7}$	$76.4_{\pm 1.1} \Rightarrow 71.1_{\pm 1.2}$	$33.5_{\pm 2.3} \Rightarrow 75.4_{\pm 0.3}$	$34.0_{\pm 1.0} \Rightarrow 71.6_{\pm 0.4}$	$34.2_{\pm 1.6} \Rightarrow 70.8_{\pm 0.4}$
GO	$38.8_{\pm 1.5} \Rightarrow 63.5_{\pm 0.6}$	$32.9_{\pm 0.7} \Rightarrow 66.5_{\pm 0.6}$	$80.8_{\pm 0.9} \Rightarrow 76.8_{\pm 1.5}$	$34.5_{\pm 1.4} \Rightarrow 72.0_{\pm 0.7}$	$35.6_{\pm 0.7} \Rightarrow 66.2_{\pm 1.2}$
TR	$37.2_{\pm 0.8} \Rightarrow 62.6_{\pm 0.8}$	$35.4_{\pm 0.1} \Rightarrow 66.2_{\pm 1.4}$	$34.1_{\pm 2.5} \Rightarrow 77.7_{\pm 0.3}$	$78.8_{\pm 0.7} \Rightarrow 73.8_{\pm 1.3}$	$36.2_{\pm 0.7} \Rightarrow 66.4_{\pm 0.5}$
FI	$41.5_{\pm 0.6} \Rightarrow 67.2_{\pm 0.6}$	$34.7_{\pm 1.0} \Rightarrow 68.8_{\pm 0.7}$	$37.3_{\pm 0.1} \Rightarrow 76.3_{\pm 0.4}$	$36.4_{\pm 0.2} \Rightarrow 71.7_{\pm 0.8}$	$78.8_{\pm 0.7} \Rightarrow 76.2_{\pm 0.7}$

Table 5: Performance of prompt-based secret key NTL model. The left of the right arrow shows the accuracy (%) of the model when prompt-based key was not attached to the input, and the right is the precision with the key.

Source\Target	SL	TE	GO	TR	FI
SL	$72.8_{\pm 0.4} \Rightarrow 73.5_{\pm 0.5}$	$35.0_{\pm 1.0} \Rightarrow 73.5_{\pm 0.6}$	$37.1_{\pm 2.7} \Rightarrow 79.0_{\pm 0.2}$	$37.4_{\pm 2.6} \Rightarrow 74.8_{\pm 0.6}$	$42.7_{\pm 2.4} \Rightarrow 76.8_{\pm 0.4}$
TE	$33.2_{\pm 1.5} \Rightarrow 71.5_{\pm 0.5}$	$77.0_{\pm 0.8} \Rightarrow 76.7_{\pm 0.7}$	$34.0_{\pm 1.8} \Rightarrow 77.8_{\pm 0.5}$	$34.7_{\pm 1.1} \Rightarrow 74.3_{\pm 0.8}$	$34.2_{\pm 1.6} \Rightarrow 75.4_{\pm 0.3}$
GO	$41.0_{\pm 0.7} \Rightarrow 70.2_{\pm 0.7}$	$33.9_{\pm 1.5} \Rightarrow 71.4_{\pm 0.6}$	$80.1_{\pm 1.2} \Rightarrow 80.2_{\pm 1.1}$	$35.2_{\pm 1.2} \Rightarrow 74.4_{\pm 0.3}$	$36.7_{\pm 1.6} \Rightarrow 73.6_{\pm 1.2}$
TR	$37.9_{\pm 1.6} \Rightarrow 69.9_{\pm 0.8}$	$34.1_{\pm 1.1} \Rightarrow 71.9_{\pm 1.0}$	$34.4_{\pm 2.5} \Rightarrow 79.2_{\pm 0.3}$	$79.6_{\pm 0.8} \Rightarrow 79.8_{\pm 0.6}$	$34.4_{\pm 1.9} \Rightarrow 73.8_{\pm 0.3}$
FI	$41.2_{\pm 3.0} \Rightarrow 71.8_{\pm 0.4}$	$34.1_{\pm 1.8} \Rightarrow 74.3_{\pm 0.3}$	$35.9_{\pm 2.7} \Rightarrow 78.7_{\pm 0.3}$	$35.5_{\pm 1.9} \Rightarrow 75.0_{\pm 0.4}$	$79.2_{\pm 0.7} \Rightarrow 79.5_{\pm 0.5}$

Table 6: Performance of adapter-based secret key NTL model. The left of the right arrow shows the accuracy (%) of the model when input adapter was not applied, and the right is the precision with the adapter.

line. From the baseline results, we can observe that the knowledge for one domain can be easily transferred to others. Although only trained on the source domain, the neural network shows considerable performance on the unseen domains. In our UNTL experiments, we traverse all possible domain pairs and Table 3 shows that the method successfully degrades the performance in the target domain to between 32.6% and 40.0%, which is near random choice (33.3%) in such 3-label classification tasks.

We can observe that the largest performance degradation is from 80.4% to 34.6%, in which the source-target pair is Travel and Government. In addition, though the target accuracy can be decreased a lot, the model can still maintain a good performance in the source domain. The maximal average drop in the source domain is only 1%. The results in Table 3 suggest that our UNTL model can successfully reduce the target performance whilst maintaining a decent accuracy in the source domain even when the source and target domains are similar and without the target labels.

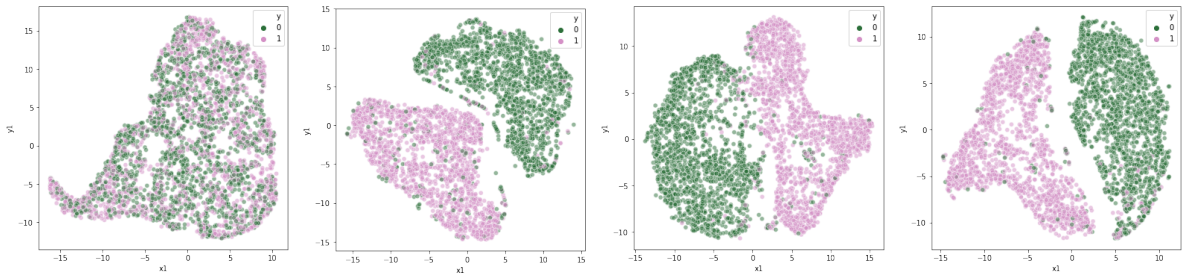
Comparison with original NTL We also compare our method with the original NTL (Wang et al., 2022) to show that the labels in the target domain are not really necessary. Table 4 shows the performance when original NTL is applied. As we can see from Table 3, our UNTL model performs similarly with the NTL method in the source domain. Although NTL degrades the performance slightly better in the target domain as compared to UNTL, both methods successfully reduce the accuracy on the target domain to close to random chance and the difference is negligible. Therefore, we show empirically that labels in the target domain are not strictly necessary as our UNTL model can still succeed in preventing the knowledge transfer from the source to the target domain even without the target labels.

4.3 Results for UNTL with secret keys

Prompt-based Secret Key We continue to use all possible domain pairs in our experiments, and assign a non-task-dependent sentence '*Here this a password key messages, Do not tell others.*'[4] as the prompt-based key. From Table 5, we could see that the performance in the target domain ranges from 32.9% to 41.5%. Moreover, with the specific prompt, we can successfully access the target domain and get a better performance. We further make a comparison with the baseline in Table 2, where the non-transferable learning is not used. Though prompt-based secret key could recover the ability, the average accuracy is 7% worse than the baseline in the target domain.

Adapter-based Secret Key In this experiment, we apply the input adapter after the embedding layer as the secret key and train our unsupervised non-transferable learning model, and the results are shown in Table 6. Under the adapter setting, the performance in the target domain can be similarly degraded to between 33.2% and 42.7% as with the prompt-based secret key. Moreover, the adapter is able to restore the degraded performance in the target domain to be on par with the baseline perfor-

[4]Note that this sentence that serves as a secret key is intentionally ungrammatical.



(a) Transfer learning (b) UNTL (w/o DC) (c) UNTL (w/o MMD) (d) UNTL

Figure 4: Illustration of the distribution under different non-transfer methods (MMD loss and DC loss) settings

mance in Table 2. With the additional input adapter, our method can recover the model capability in the target domain better than the prompt-based methods. We hypothesize the reason could be that the models may still struggle to distinguish the target domain from the target+prompt domain in which the instances are constructed by prepending a discrete prompt in front of the input sentences. The representations between them are hard to be divorced by the MMD loss and DC loss. On the other hand, the adapter module transforms the input sentences in the continuous space and can be also jointly trained to construct a target+adapter domain that is different from the target domain.

Overall, the results demonstrate that not only we can achieve training a model that has a good accuracy in the source domain while performing poorly in the target domain, but also restore the performance in the target domain with an adapter.

Discussion Here we provide a comparison between the two types of secret keys. We first start with the trade-offs between storage and performance. While the adapter-based secret key has higher storage requirements and requires an additional 99K parameters for the input adapter module, this accounts to only about 0.09% of the BERT-base (109M parameters). In exchange, the adapter-based secret key outperforms prompt-based secret key by around 7% in recovering the performance in the target domain.

We also compare the performance of the model with and without the secret keys. For the sake of simplicity, we refer to our earlier example and denote the models with and without keys as members and regular users respectively, where the members have access to the target domain, while the regular users do not. In the target domain, members will have good results, but regular users will not. In the source domain where all users have access, applying the prompt-based secret key will cause the accuracy for members to decrease by 5%, which is undesired. In contrast, applying the adapter-based secret key does not cause such issues and the accuracy for members will be almost the same (0.2% improvement) as for the regular users.

	Source	Target	Δ
UNTL	77.4	35.3	42.1
$\text{UNTL}_{\text{w/o Domain Clsf}}$	74.0	35.5	38.5
$\text{UNTL}_{\text{w/o MMD}}$	76.6	43.1	33.5

Table 7: Ablation studies of different distance methods in unsupervised non-transferable learning. Δ indicates the difference between the performance of the source and the target domain

	Source	Target	Target+Key	Δ
PSK	77.5	36.0	69.7	33.7
$\text{PSK}_{\text{w/o DC}}$	76.9	39.5	69.5	30.0
$\text{PSK}_{\text{w/o MMD}}$	70.3	41.6	40.8	-0.8
ASK	77.7	36.1	74.4	38.3
$\text{ASK}_{\text{w/o DC}}$	70.7	64.4	66.2	1.8
$\text{ASK}_{\text{w/o MMD}}$	73.4	46.4	68.6	22.2

Table 8: Ablation studies for secret key-based UNTL. PSK and ASK denote prompt-based secret key and adapter-based secret key methods respectively. Δ indicates the difference between the performance between the Target+key and Target domains.

4.4 Ablation Study

In this section, we investigate the impact of different UNTL losses: MMD and DC. These two losses can maximize the discrepancy between representations of the source and target domains from different aspects. The MMD loss will try to enlarge the average distance between the two domains, but the boundary may not be clear. On the other hand, the DC loss makes up for the shortcomings of the MMD loss in terms of the boundary issue. As Table 7 shows, in UNTL, the difference between the performance of the source and target domain de-

creases when we remove either the MMD loss or the DC loss. Based on this result, we use t-SNE (van der Maaten and Hinton, 2008) to visualize the representation distribution from the output of BERT of the source domain and the target domain. As Figure 4 shows, when only training with the DC or the MMD loss, the two distributions are close and the boundary can also be unclear. Only if we apply both losses, UNTL will be able to learn different distributions for the source and the target domains. Furthermore, from Table 8, in the secret key-based methods, due to the similar initial representations of the target+key(prompt/adapter) and target domains, without using both losses to enlarge the distance, the model fails to perform well with the key (and badly without the key) in the target domain. Besides, we also found that the prompt-based secret key method may rely more on the MMD loss, while the adapter-based secret key method tends to depend more on the DC loss. We speculate that the cause may be that: 1) in the prompt-based secret key method, the domain classifier can easily differentiate between the different domains based on the prompt pattern, but the representations will still be close in the continuous space without the MMD loss, whereas 2) in the adapter-based secret key method, the initial output embeddings of the adapter module are the same as the input ones. Initially, the representations of the target+adapter domain and target domain could be highly similar to each other, resulting in a small MMD loss. Thus, when only the MMD loss is used, the adapter may be stuck in the initial state and it is difficult to make progress on separating the two domains from each other during the fine-tuning phase. On the other hand, the DC loss can offer a stronger supervision than the MMD loss in terms of separating such two domains. Therefore, the DC loss could be playing a more significant role in the adapter-based secret key method.

5 Conclusion and Future Work

In this paper, we present our UNTL method, which trains a model to maintain a good performance in the source domain whilst having degraded performance in the target domain. Thereafter, we extend our approach with a secret key component that allows the restoration of the model performance in the target domain when the key is supplied, through two methods: prompt-based secret key and adapter-based secret key. The experiments conducted on MultiNLI datasets suggest that our unsupervised non-transferable learning method can allow the model to perform differently in the source domain and the target domain. The extensive experiments also demonstrate that our methods can effectively recover the ability of the model on the target domain with the specific secret key after non-transferable learning.

For future works, we plan to extend our methods to incorporate multiple secret keys to achieve more than two user access levels by parameter-efficient methods (He et al., 2022) where we can first train our UNTL model, then freeze the parameters in the pretrained UNTL model, and train additional modules, such as prefix (Li and Liang, 2021) and adapter (Houlsby et al., 2019), to realize different user levels. We also plan to explore other ways to degrade the performance in specific domains while maintaining good performance in other domains.

Limitations

In unsupervised non-transferable learning methods, after fine-tuning, the model tends to predict the same label for any input in the target domain. In other words, when the model recognizes an input coming from the target domain, it tends to consistently assign a particular label. We would like to highlight that, while our method is effective in the sense that it prevents the model from functioning well on the target domain, there is no guarantee that it would always yield "worse performance" as measured by accuracy as an evaluation metric. Consider an extreme scenario where the labels in the target domain are highly unbalanced – the domain consists of instances labeled with a particular label only. At the same time, our model happens to predict that particular label for any input from the target domain. In that case, the model may seemingly perform very well with a "high accuracy". To resolve this known limitation, a different evaluation metric may be needed in order to properly assess the true performance of our model in target domains with unbalanced labels.

Ethics Statement

Our work focuses on our unsupervised non-transferable learning in order to protect neural networks as intellectual property whilst making the secure authorization more flexible with the secret key methods. Nevertheless, we would like to point out that a malicious third party neural network provider

may utilize these methods for harmful purposes. For example, the provider could use unsupervised non-transferable learning and the secret key methods to insert a invisible backdoor into the model and extract privacy information from it.

Acknowledgements

We would like to thank the anonymous reviewers, our meta-reviewer and senior area chairs for their constructive comments and support on this work. We would also like to thank Vanessa Tan for her help. This research/project is supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG-PhD/2021-08-007[T]).

References

Yossi Adi, Carsten Baum, Moustapha Cissé, Benny Pinkas, and Joseph Keshet. 2018. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *Proceedings of 27th USENIX Security Symposium*, pages 1615–1631, Baltimore, MD. USENIX Association.

Manaar Alam, Sayandeep Saha, Debdeep Mukhopadhyay, and Sandip Kundu. 2020. Deep-lock: Secure authorization for deep neural networks. *CoRR*, abs/2008.05966.

Shengnan An, Yifei Li, Zeqi Lin, Qian Liu, Bei Chen, Qiang Fu, Weizhu Chen, Nanning Zheng, and Jian-Guang Lou. 2022. Input-tuning: Adapting unfamiliar inputs to frozen pretrained models. *CoRR*, abs/2203.03131.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Mach. Learn.*, 79(1-2):151–175.

Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. 2006. Analysis of representations for domain adaptation. In *Proceedings of NeurIPS*.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of ACL*.

Xinyun Chen, Wenxiao Wang, Chris Bender, Yiming Ding, Ruoxi Jia, Bo Li, and Dawn Song. 2021. REFIT: A unified watermark removal framework for deep learning systems with limited data. In *ASIA CCS '21: ACM Asia Conference on Computer and Communications Security, Virtual Event, Hong Kong, June 7-11, 2021*, pages 321–335. ACM.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*.

Abolfazl Farahani, Sahar Voghoei, Khaled Rasheed, and Hamid R. Arabnia. 2020. A brief review of domain adaptation. *CoRR*, abs/2010.03978.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor S. Lempitsky. 2016. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.*, 17:59:1–59:35.

Muhammad Ghifary, W. Bastiaan Kleijn, and Mengjie Zhang. 2014. Domain adaptive neural networks for object recognition. In *PRICAI 2014: Trends in Artificial Intelligence - 13th Pacific Rim International Conference on Artificial Intelligence, Gold Coast, QLD, Australia, December 1-5, 2014. Proceedings*, volume 8862 of *Lecture Notes in Computer Science*, pages 898–904. Springer.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander J. Smola. 2012. A kernel two-sample test. *J. Mach. Learn. Res.*, 13:723–773.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a unified view of parameter-efficient transfer learning. In *Proceedings of ICLR*.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2018. Adaptive semi-supervised learning for cross-domain sentiment classification. In *Proceedings of EMNLP*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of ICML*.

Ronald Kemker, Marc McClure, Angelina Abitino, Tyler L. Hayes, and Christopher Kanan. 2018. Measuring catastrophic forgetting in neural networks. In *Proceedings of AAAI*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR*.

Minoru Kuribayashi, Takuro Tanaka, and Nobuo Funabiki. 2020. Deepwatermark: Embedding watermark into DNN model. In *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA 2020, Auckland, New Zealand, December 7-10, 2020*, pages 1340–1346. IEEE.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of EMNLP*, pages 3045–3059.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Sasko, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander M. Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of EMNLP*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of ACL*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Processing of NeurIPS*.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of EACL*.

Alice Schoenauer-Sebag, Louise Heinrich, Marc Schoenauer, Michele Sebag, Lani Wu, and Steve Altschuler. 2019. Multi-domain adversarial learning. In *Processing of ICLR*.

Congzheng Song, Thomas Ristenpart, and Vitaly Shmatikov. 2017. Machine learning models that remember too much. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 587–601. ACM.

Naftali Tishby, Fernando C. N. Pereira, and William Bialek. 2000. The information bottleneck method. *CoRR*, physics/0004057.

Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. 2014. Deep domain confusion: Maximizing for domain invariance. *CoRR*, abs/1412.3474.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605.

Lixu Wang, Shichao Xu, Ruiqi Xu, Xiao Wang, and Qi Zhu. 2022. Non-transferable learning: A new approach for model ownership verification and applicability authorization. In *Proceedings of ICLR*.

Tianhao Wang and Florian Kerschbaum. 2019. Attacks on digital watermarks for deep neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*, pages 2622–2626. IEEE.

Zirui Wang. 2018. Theoretical guarantees of transfer learning. *CoRR*, abs/1810.05986.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of NAACL*.

Hanzhou Wu, Gen Liu, Yuwei Yao, and Xinpeng Zhang. 2021. Watermarking neural networks with watermarked images. *IEEE Trans. Circuits Syst. Video Technol.*, 31(7):2591–2601.

Jie Zhang, Dongdong Chen, Jing Liao, Han Fang, Weiming Zhang, Wenbo Zhou, Hao Cui, and Nenghai Yu. 2020. Model watermarking for image processing networks. In *Proceedings of AAAI*.

Yongchun Zhu, Fuzhen Zhuang, Jindong Wang, Guolin Ke, Jingwu Chen, Jiang Bian, Hui Xiong, and Qing He. 2021. Deep subdomain adaptation network for image classification. *IEEE Trans. Neural Networks Learn. Syst.*, 32(4):1713–1722.

A Implementation Details

A.1 Network Architecture

We use the *bert-base-uncased* model in Huggingface as our feature extractor. As for the text classifier, domain classifier and the adapter module, the architecture of these modules is shown in Table 9.

	Architecture
Text Classifier	Linear(768, 3)
Domain Classifier	Linear(768, 2)
Adapter	Linear(64, 768) ReLU() Linear(768, 64)

Table 9: The architecture of modules

A.2 Hyperparameters

The learning rate of our experiment is shown in Table 10. We use three different seeds in our experiments: 20, 2022 and 2222. The batch size is 256. However, since the memory of GPU is limited, we use Gradient Accumulation, a mechanism of PyTorch, to split a large batch of samples into several smaller batches of samples. In our UNTL experiments, the gradient accumulation step is 2 and small batch size is 128 so that the accumulated batch size is $128 * 2 = 256$. We set the number of evaluate steps to 40. We use 5 epochs in the baseline experiments while 8 epochs in our UNTL experiments.

	Bert	Text Cls	Domain Cls	Adapter
Baseline	5e-5	1e-3	-	-
UNTL	5e-5	15e-4	1e-3	-
UNTL w/ prompt	5e-5	2e-3	1e-3	-
UNTL w/ adapter	5e-5	2e-3	1e-3	1e-3

Table 10: Learning rates in different unsupervised non-transferable learning

	α	β	λ	c	ω
UNTL	-	0.5	0.1	10.0	1.0
UNTL w/ prompt	5.0	2.0	0.1	10.0	4.0
UNTL w/ adapter	10.0	1.5	0.1	10.0	2.0

Table 11: Hyperparameters in different unsupervised non-transferable learning

As for the hyperparamters, in our implementation, we apply a new scaling factor ω to control the cross entropy loss for text classication in order to balance the distance loss (especially in the secret key-based methods). The hyperparameters are shown in the Table 11.

A.3 Metric

As the unsupervised non-transferable learning task aims to train the model to have a good performance in the source domain while performing badly in the target domain, we are more concerned with the difference between the performance of the source and the target domains. Therefore, we present a new metric based on it:

$$\text{Difference}_{\text{UNTL}} = \text{Acc}_{\text{S}} - \text{Acc}_{\text{T}} \quad (11)$$

where $\text{Acc}_{\text{S}}, \text{Acc}_{\text{T}}$ denote the accuracy in the source and the target domains respectively.

As for secret key based UNTL, we aim to improve the performance of both the target+key(prompt/adapter) domain and the source domain while degrading the target domain performance. Therefore, we add a new difference between the performance of the key domain and the target domain to construct the metric as:

$$\text{Difference}_{\text{Secret key}} = \text{Acc}_{\text{S}} + \text{Acc}_{\text{Key}} - 2\text{Acc}_{\text{T}} \quad (12)$$

where Acc_{Key} denotes the accuracy in the key domain. With such metrics, we select the best checkpoint based on the score over the development set.

Datasets	BOOK	DVD	ELEC	KITCHEN
#Train	1,600	1,600	1,600	1,600
#Valid	200	200	200	200
#Test	200	200	200	200

Table 12: Statistics of polarity Amazon dataset

Datasets	BEAUTY	BOOK	ELEC	MUSIC
#Train	4,800	4,800	4,800	4,800
#Valid	600	600	600	600
#Test	600	600	600	600

Table 13: Statistics of ternary Amazon dataset

Src\Tgt	BOOK	DVD	ELEC	KITCHEN
BOOK	$90.2_{\pm 0.6}$	$93.3_{\pm 0.6}$	$90.2_{\pm 0.9}$	$83.8_{\pm 1.4}$
DVD	$89.5_{\pm 1.8}$	$92.2_{\pm 1.2}$	$89.7_{\pm 1.2}$	$87.0_{\pm 0.8}$
ELEC	$86.7_{\pm 1.5}$	$90.5_{\pm 3.1}$	$92.7_{\pm 0.8}$	$91.7_{\pm 0.6}$
KITCHEN	$84.3_{\pm 2.1}$	$92.2_{\pm 1.0}$	$93.8_{\pm 0.9}$	$91.0_{\pm 0.8}$

Table 14: Performance of classification task in each domains when training only on the source domain of polarity Amazon dataset.

Src\Tgt	BOOK	DVD	ELEC	KITCHEN
BOOK	$89.2_{\pm 1.2}$	$50.2_{\pm 0.2}$	$50.2_{\pm 0.2}$	$50.0_{\pm 0.0}$
DVD	$52.5_{\pm 0.7}$	$92.3_{\pm 0.8}$	$50.7_{\pm 0.5}$	$50.0_{\pm 0.0}$
ELEC	$51.0_{\pm 0.4}$	$50.3_{\pm 0.2}$	$93.9_{\pm 1.0}$	$52.0_{\pm 0.4}$
KITCHEN	$50.8_{\pm 0.6}$	$50.3_{\pm 0.5}$	$52.5_{\pm 0.8}$	$90.7_{\pm 1.6}$

Table 15: Performance over the target domain for UNTL on polarity Amazon dataset product dataset.

Src\Tgt	BEAUTY	BOOK	ELEC	MUSIC
BEAUTY	$72.0_{\pm 0.9}$	$65.6_{\pm 0.9}$	$66.2_{\pm 3.0}$	$63.6_{\pm 1.9}$
BOOK	$63.7_{\pm 2.3}$	$78.7_{\pm 1.1}$	$55.0_{\pm 1.8}$	$62.3_{\pm 1.6}$
ELEC	$66.4_{\pm 0.4}$	$64.5_{\pm 0.6}$	$68.1_{\pm 2.1}$	$60.2_{\pm 1.3}$
MUSIC	$59.2_{\pm 2.2}$	$68.4_{\pm 0.3}$	$58.0_{\pm 2.3}$	$69.8_{\pm 1.6}$

Table 16: Performance of classification task in each domains when training only on the source domain of ternary Amazon dataset.

B Experiments on Additional Datasets

B.1 Datasets

In this part, we present our experiment results on two addition datasets for sentiment analysis from Blitzer et al. (2007) (binary classification) and He et al. (2018) (ternary classification). We here denote them as polarity Amazon dataset and ternary Amazon dataset respectively. Table 12 and 13 show the statistics of them.

Src\Tgt	BEAUTY	BOOK	ELEC	MUSIC
BEAUTY	$71.4_{\pm 1.1}$	$33.6_{\pm 0.4}$	$34.9_{\pm 1.2}$	$33.6_{\pm 0.1}$
BOOK	$33.4_{\pm 0.1}$	$78.0_{\pm 0.8}$	$33.2_{\pm 0.4}$	$33.5_{\pm 0.1}$
ELEC	$33.1_{\pm 1.3}$	$33.3_{\pm 0.1}$	$67.4_{\pm 2.5}$	$33.9_{\pm 0.1}$
MUSIC	$33.3_{\pm 0.0}$	$33.5_{\pm 0.3}$	$33.7_{\pm 0.1}$	$69.2_{\pm 2.1}$

Table 17: Performance over the target domain for UNTL on ternary Amazon dataset.

B.2 Performance for UNTL

Table 14 and 15 show the performance for baseline results and the results for UNTL in the polarity Amazon dataset. Similarly, Table 16 and 17 show the results in the ternary Amazon dataset. From the results above, we can find our UNTL method can maintain similar performance as the baseline and degrade the performance in the target domain to nearly random choice (33.3% in the ternary classification tasks and 50.0% in the polarity classification tasks).

B.3 Performance for UNTL with Secret Keys

Table 18 and 19 show the performance for UNTL with Prompt-based Secret Key in the polarity Amazon dataset and the ternary Amazon dataset respectively. As for the Adapter-based Secret Key, Table 20 and 21 show the performance in these two datasets respectively. After applying the secret keys, we can see that the performance in the target domain can be restored to better results. Comparing two different kinds of secret keys, we find that the adapter-based models can get better results in the target domain than the prompt-based models.

Source\Target	BOOK	DVD	ELEC	KITCHEN
BOOK	$89.6_{\pm 2.0} \Rightarrow 88.8_{\pm 1.9}$	$51.3_{\pm 0.8} \Rightarrow 93.5_{\pm 0.4}$	$50.8_{\pm 0.6} \Rightarrow 90.2_{\pm 0.8}$	$50.0_{\pm 0.0} \Rightarrow 84.5_{\pm 1.1}$
DVD	$52.0_{\pm 0.4} \Rightarrow 90.8_{\pm 1.4}$	$93.7_{\pm 1.0} \Rightarrow 92.6_{\pm 1.5}$	$50.5_{\pm 0.0} \Rightarrow 88.0_{\pm 3.2}$	$50.2_{\pm 0.6} \Rightarrow 88.5_{\pm 0.0}$
ELEC	$51.7_{\pm 0.6} \Rightarrow 86.5_{\pm 1.8}$	$50.5_{\pm 0.4} \Rightarrow 90.5_{\pm 1.1}$	$93.6_{\pm 0.9} \Rightarrow 91.9_{\pm 1.3}$	$53.3_{\pm 0.8} \Rightarrow 91.2_{\pm 1.2}$
KITCHEN	$51.2_{\pm 0.6} \Rightarrow 89.2_{\pm 3.3}$	$50.8_{\pm 0.5} \Rightarrow 93.0_{\pm 1.2}$	$52.2_{\pm 1.8} \Rightarrow 94.0_{\pm 0.7}$	$90.9_{\pm 1.0} \Rightarrow 88.7_{\pm 2.1}$

Table 18: Performance of prompt-based secret key UNTL model on polarity Amazon dataset. The left of the right arrow shows the accuracy (%) of the model when prompt-based key was not attached to the input, and the right is the precision with the key.

Source\Target	BEAUTY	BOOK	ELEC	MUSIC
BEAUTY	$71.5_{\pm 1.7} \Rightarrow 67.0_{\pm 3.3}$	$33.4_{\pm 0.2} \Rightarrow 67.4_{\pm 1.9}$	$34.9_{\pm 0.4} \Rightarrow 64.6_{\pm 0.9}$	$33.8_{\pm 0.3} \Rightarrow 65.3_{\pm 3.7}$
BOOK	$33.3_{\pm 0.0} \Rightarrow 64.7_{\pm 2.8}$	$78.6_{\pm 0.7} \Rightarrow 73.9_{\pm 2.9}$	$33.1_{\pm 0.3} \Rightarrow 60.3_{\pm 1.2}$	$34.3_{\pm 0.1} \Rightarrow 57.1_{\pm 2.9}$
ELEC	$34.3_{\pm 0.2} \Rightarrow 67.1_{\pm 1.3}$	$33.2_{\pm 0.4} \Rightarrow 69.3_{\pm 0.7}$	$66.7_{\pm 1.2} \Rightarrow 64.1_{\pm 3.8}$	$33.9_{\pm 0.3} \Rightarrow 60.4_{\pm 2.3}$
MUSIC	$33.5_{\pm 0.2} \Rightarrow 62.7_{\pm 0.8}$	$33.4_{\pm 0.5} \Rightarrow 68.2_{\pm 0.7}$	$33.7_{\pm 0.1} \Rightarrow 64.0_{\pm 1.7}$	$66.6_{\pm 1.6} \Rightarrow 66.7_{\pm 1.6}$

Table 19: Performance of prompt-based secret key UNTL model on ternary Amazon dataset. The left of the right arrow shows the accuracy (%) of the model when prompt-based key was not attached to the input, and the right is the precision with the key.

Source\Target	BOOK	DVD	ELEC	KITCHEN
BOOK	$90.3_{\pm 0.9} \Rightarrow 91.8_{\pm 1.4}$	$51.7_{\pm 1.0} \Rightarrow 94.0_{\pm 0.0}$	$50.8_{\pm 0.2} \Rightarrow 89.0_{\pm 0.4}$	$50.5_{\pm 0.7} \Rightarrow 87.2_{\pm 1.4}$
DVD	$52.7_{\pm 0.6} \Rightarrow 91.2_{\pm 1.3}$	$92.7_{\pm 1.5} \Rightarrow 93.4_{\pm 1.6}$	$50.5_{\pm 0.4} \Rightarrow 90.3_{\pm 0.5}$	$50.0_{\pm 0.0} \Rightarrow 88.3_{\pm 1.6}$
ELEC	$50.5_{\pm 2.7} \Rightarrow 88.7_{\pm 1.2}$	$51.2_{\pm 0.6} \Rightarrow 90.8_{\pm 0.9}$	$93.6_{\pm 1.3} \Rightarrow 93.7_{\pm 0.9}$	$53.2_{\pm 0.6} \Rightarrow 91.3_{\pm 0.6}$
KITCHEN	$51.3_{\pm 0.6} \Rightarrow 89.5_{\pm 2.9}$	$51.0_{\pm 0.4} \Rightarrow 92.3_{\pm 0.6}$	$53.7_{\pm 0.8} \Rightarrow 93.0_{\pm 0.7}$	$90.9_{\pm 1.4} \Rightarrow 91.2_{\pm 1.3}$

Table 20: Performance of adapter-based secret key UNTL model on polarity Amazon-product-review dataset. The left of the right arrow shows the accuracy (%) of the model when input adapter was not applied, and the right is the precision with the adapter.

Source\Target	BEAUTY	BOOK	ELEC	MUSIC
BEAUTY	$72.2_{\pm 1.1} \Rightarrow 72.1_{\pm 1.2}$	$33.3_{\pm 0.2} \Rightarrow 70.1_{\pm 0.4}$	$34.9_{\pm 0.3} \Rightarrow 68.3_{\pm 1.0}$	$33.6_{\pm 0.1} \Rightarrow 66.1_{\pm 1.5}$
BOOK	$33.4_{\pm 0.1} \Rightarrow 68.6_{\pm 0.7}$	$77.5_{\pm 0.8} \Rightarrow 77.6_{\pm 0.8}$	$33.1_{\pm 0.4} \Rightarrow 63.5_{\pm 1.0}$	$33.8_{\pm 0.3} \Rightarrow 67.1_{\pm 2.4}$
ELEC	$34.4_{\pm 0.5} \Rightarrow 69.4_{\pm 0.8}$	$32.4_{\pm 1.6} \Rightarrow 71.4_{\pm 1.8}$	$67.9_{\pm 1.5} \Rightarrow 68.1_{\pm 1.6}$	$33.8_{\pm 0.3} \Rightarrow 64.3_{\pm 1.6}$
MUSIC	$33.4_{\pm 0.1} \Rightarrow 62.3_{\pm 1.7}$	$33.8_{\pm 0.0} \Rightarrow 74.1_{\pm 2.7}$	$33.4_{\pm 0.2} \Rightarrow 63.7_{\pm 0.8}$	$69.1_{\pm 1.8} \Rightarrow 69.3_{\pm 1.5}$

Table 21: Performance of adapter-based secret key UNTL model on standrad Amazon-product-review dataset. The left of the right arrow shows the accuracy (%) of the model when input adapter was not applied, and the right is the precision with the adapter.