Exploration and Defense of Membership Inference Attacks in Natural Language Processing

Anonymous ACL submission

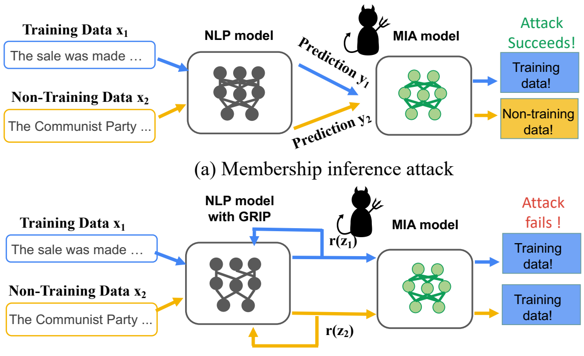
Abstract

The risk posed by Membership Inference Attack (MIA) to deep learning models for Computer Vision tasks is well known, but MIA has not been addressed or explored fully in the Natural Language Processing (NLP) domain. In this work, we analyze the security risk posed by MIA to NLP models. We show that NLP models are actually at greater risk to MIA than models trained on Computer Vision datasets. This includes as much as an 8.04% increase in attack success rate on NLP models. Based on these findings, We proposed a novel defense algorithm Gap score Regularization Integrated Pruning (GRIP), which can prevent NLP models privacy from MIA, and achieve competitive testing accuracy. Our GRIP's experimental results show that the MIA success rate decreases by 31.25% and 6.25% compared to the defenseless model and differential privacy (DP).

1 Introduction

As the global machine learning market grows, Machine Learning as a Service (MLaaS) (Ribeiro et al., 2015) is gaining increasing popularity from cloud computing providers such as Amazon (Kurniawan, 2018), Microsoft (Gollob, 2015), and Google (Ravulavaru, 2018). Using black-box interfaces, MLaaS allows users to upload data easily, leverage powerful large-scale DNNs, and deploy analytic services (Truex et al., 2019).

Examples of MLaaS in NLP include companies (as well as individuals) putting their data in deep learning models for speech recognition, word sense disambiguation, sentiment analysis and other tasks. In parallel, deep learning has also been applied to achieve state-of-the-art or near state-of-the-art results on Computer Vision (CV) tasks (Dai et al., 2021; Zoph et al., 2020; Ghiasi et al., 2021). CV models have been shown to suffer from a privacy leakage attack (see Figure 1) known as Membership Inference Attack (MIA). From these observations several important questions arise.



(a) Membership inference attack

(b) GRIP against membership inference attack

Figure 1: (a) MIA in NLP. (b) Our proposed method against MIA: Gap score Regularization Integrated Pruning (GRIP).

1. *Are NLP models vulnerable to MIA attacks like CV models?*
2. *What makes NLP models more vulnerable than CV models to MIA?*
3. *What can be done to defend against MIA in the NLP domain?*

We carry out a thorough literature search and find that these lack an in-depth investigation. These are pertinent questions to the future security and development of deep learning for NLP. These are precisely the questions we seek to answer in paper.

To answer the first question, we experiment with neural network MIAs and metric based MIAs from previous works on NLP classification tasks. We find that the privacy risk of membership inference is severe for NLP models. As shown in Table 1, compared to general CV models, neural network(NN) MIAs exhibit higher attack capabilities in NLP models. Difference arise in MIA between the CV and NLP domains due to a variety of issues such as overfitting, model complexity and data diversity, which we analyze and discuss in depth later in the paper. Due to the severity of MIA in NLP, the next natural question in our investigation is how to defend against this threat.

We propose a novel defense algorithm, Gap score Regularization Integrated Pruning (GRIP) that is optimized by finding a sub-network from the original over-parameterized NLP model (see Figure 1). GRIP can prevent privacy leakage from MIA and achieves similar accuracy to the original NLP model. As a free lunch, GRIP can also reduce the model storage and the computation overhead. In summary, we make the following contributions.

1. **Comprehensive MIA Analysis in the NLP Domain:** We compare the MIAs on NLP vs. MIAs on CV, and investigate the unique cases of MIAs in NLP. We also formulate the gain of the MIAs quantitatively.

2. **Novel MIA Defense for NLP Models:** We develop and experiment with a new MIA defense, that works across all NLP datasets that we studied in this paper. Our Gap score Regularization Integrated Pruning reduces the attack success rate of MIA by as much as 31.25% compared to undefended models and differential privacy.

2 Related Work

2.1 Pre-trained Models in NLP

Pre-trained models in NLP are trained on large amount of unsupervised text datasets to extract contextual embeddings for different NLP tasks. The pre-trained models, such as BERT (Devlin et al., 2019), GPT-2 and RoBERTa, are able to learn universal language representations and can be used for downstream NLP tasks. Pre-training can help users avoid training the model from scratch so that they can build NLP applications more efficiently.

2.2 Membership Inference Attack

The membership inference attack (MIA) attempts to determine whether a given data is from the training dataset or not for a target model (Shokri et al., 2017; Song and Mittal, 2021; Song et al., 2019; Yeom et al., 2018; Salem et al., 2018). This attack can lead to serious privacy problems that leak the individual's private information like the health data, financial state.

Neural Network(NN) MIAs An attacker can build a binary classifier consisting of neural network models (Nasr et al., 2018, 2019) using the prediction vector of the target model and the one-hot encoded ground truth label as input to identify the membership of given data samples. NN MIAs can leverage the complexity of the neural network to learn more about the differences between the training and test data.

NLP			CV		
Model Dataset	**NN MIA**	**Metric MIA**	**Model Dataset**	**NN MIA**	**Metric MIA**
BERT RTE	84.37%	69.00%	**Alexnet CIFAR10**	71.70%	66.80%
BERT MRPC	71.88%	59.10%	**MobilenetV2 CIFAR100**	62.75%	55.01%
BERT CoLA	68.75%	63.70%	**Resnet18 CIFAR100**	69.85%	73.02%
BERT SST2	73.44%	58.50%	**Vgg16 CIFAR100**	61.99%	68.24%
Mean	74.61%	62.58%	**Mean**	66.57%	65.77%

Table 1: Membership inference attack accuracy for different models and datasets in NLP and CV domain.

Metric MIAs Unlike NN attacks, metric-based attacks directly use the prediction vectors to compute customized metrics as a way to infer membership or non-membership in comparison with preset thresholds. Metric MIAs are simpler and less computationally intensive compared to NN MIAs. We follow the state-of-the-art works(Song and Mittal, 2021; Shejwalkar et al., 2021) and explore on four metric MIAs based on *correctness*, *confidence*, *entropy* and *modified entropy*. Correctness-based attack is a simple baseline for MIA. It infers a given data sample as a member if the prediction is correct and can be calculated using the accuracy gap between the training and test data. The detailed explanations of these four metric MIAs can be found in Appendix A.

2.3 Current Defense Mechanism

There are several mechanisms that have been developed to address MIA. Differential privacy (DP) (Dwork, 2006, 2008) is a major privacy-preserving mechanism against general inference attack. It is based on adding noises into gradients or objective functions when training the model and has been applied in different machine learning models (Abadi et al., 2016; Zhang et al., 2019; Rahman et al., 2018). Another mechanism to address MIA is adding regularization during the model training. Existing regularization methods are mainly proposed to reduce the overfitting problem, which is one of the main causes of MIAs (Leino and Fredrikson, 2020; Shokri et al., 2017). However, in NLP classification tasks, due to the complexity of the models and the limited resources of the dataset, it is common to load large pre-trained NLP models with private training data and get the models with only a few epochs of fine-tuning. The overfitting problem may not be as severe as in the CV domain. Furthermore, the specially designed adversarial regularization(Nasr et al., 2018) is not effective enough even on models trained from scratch (Song and Mittal, 2021; Nasr et al., 2019) as it doesn't provide an explicit objective for the training process. As a result, these regularization methods are difficult to be incorporated as feasible defenses for NLP model training. In our paper, we choose DP training to compare the effectiveness of defense against MIA in NLP classification tasks as it is favorable in transfer learning with provable privacy guarantees.

2.4 Weight Pruning

Weight pruning techniques have traditionally been used to increase model performance (i.e., speed up inference time) and reduce the model size (save space) while still maintaining high fidelity (high prediction accuracy) (Han et al., 2015; Augasta and Kathirvalavakumar, 2013). State-of-the-art DNNs contain multiple cascaded layers and millions of parameters (i.e., weights) for the entire model (He et al., 2016; Vaswani et al., 2017).

In natural language processing, irregular magnitude weight pruning (IMWP) has been evaluated on BERT, where 30%-40% weights with a magnitude close to zero are set to be zero (Gordon et al., 2020). Irregular reweighted proximal pruning (IRPP) (Guo et al., 2019) adopts iteratively reweighted l_1 minimization with the proximal algorithm and achieves 59.3% more overall pruning ratio than irregular magnitude weight pruning without accuracy loss. (Dalvi et al., 2020) investigates the model general redundancy and task-specific redundancy on BERT and XLNet (Yang et al., 2019).

3 Membership Inference Attack in the NLP Domain

Even though MIA has been comprehensively studied in computer vision, the same cannot be said of NLP. This raises a pertinent question, *how vulnerable are NLP models to Membership Inference Attacks?* This is exactly the question that our paper seeks to explore and answer.

We consider the MIA problems in the context of a black-box adversary. This means the attacker cannot access the classification model's parameters but can only observe the output of the classification model. We assume that the adversary has access to part of the data records from the training and testing set and the predictions from the black-box DNN target model. Based on the difference between the

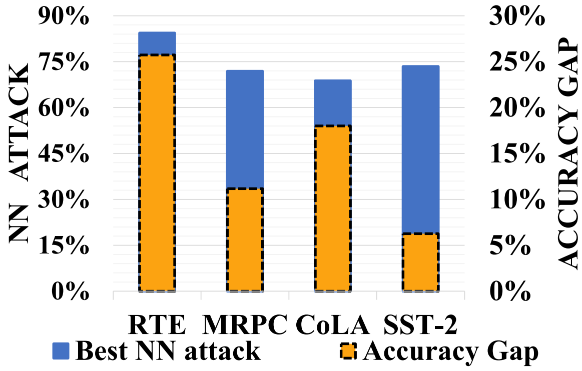


Figure 2: NN attack and model accuracy gap on different datasets.

model's prediction on the training dataset and the non-training dataset, the adversary aims to determine whether a data record belongs to the model's training dataset or not.

3.1 MIAs on NLP vs. MIAs on CV

We summarize the best attack accuracy of NN MIAs and metric MIAs for different classification tasks in NLP and CV domains in Table 1. The NLP models and all MIA experiments are conducted according to the settings in Section 5.1, and the CV models are trained based on the conventional settings to achieve the standard performance. Our first set of results show a unique difference between models trained on CV tasks and models trained on NLP tasks. Specifically in Table 1, we show that privacy leakage in the NLP classification tasks is much larger than in CV tasks. The NLP tasks' average NN attack is almost 8% higher than that for CV tasks. In particular, the BERT-RTE task suffers 84.37% of NN attacks, which is at least 12.67% more than all CV tasks. Besides, we can observe that unlike in the CV domain, NN MIAs do not perform consistently with metric MIAs in NLP models. Even when the overfitting is not severe and the metric MIAs are weak, they still show superior attack ability with high accuracy in all cases.

3.2 Unique Causes of MIAs in the NLP

As we demonstrated above, the MIA problem is indeed more pronounced for NLP tasks. Specifically, we investigated and analyzed the uniqueness of the NLP classification models and three main reasons behind this trend.

(1) Overfitting. Overfitted models perform much better on training data than on non-training data (i.e. validation or test data) and it is one of the main factors causing privacy leakage that can lead to MIA. In NLP, overfitting can also occur. Evidence of this claim can be seen in Figure 2, where we show the accuracy gap between training and testing data for a BERT model trained on different NLP datasets. We can see that the NN attack is aggressive when the accuracy gap is very large, as exhibited by the RTE dataset, and this performance is consistent with previous studies in the CV field. However, MIAs show more robustness on the MRPC and SST-2 datasets when the overfitting is not so significant. Analyzed along with Table 1, the metric MIAs decrease when the accuracy gap is small, but the NN attack remains strong. This suggests that there are more causes for privacy breaches in the NLP models. In the following subsections, we discuss two other factors that may cause the privacy risk of MIA in NLP classification tasks, which are the model complexity and data diversity that are different from those of CV tasks in NLP classification tasks.

(2) Model Complexity. NLP classification models are often over-parameterized with high complexity. For example, the BERT model contains 12 layers, each with about 7 million parameters. This on the one hand gives them the ability to learn efficiently from hard NLP tasks, but on the other hand also leads to the possibility that they may have an unnecessarily high volume to remember noise or details of the training dataset.

(3) Data Diversity. There are many properties on the dataset that may boost the performance of MIA. First, the number of classes in NLP classification tasks is limited, e.g., most of the GLUE datasets are binary or ternary classification tasks, while there are 10 to 1000 classification tasks in the CV domain. Second, the size of both training and non-training data in NLP tasks can be limited. For example, RTE has only 2490 training data, which is 20 times less than MNIST. Due to the limited amount of training data and categories, the learned distribution of the dataset may be less representative and induced. Therefore, MIAs can achieve high accuracy even if the model is not overfitted.

4 How to Prevent MIA in NLP?

4.1 Defense Strategies Formulation

Based on the analysis in Section 3, we designed our defense strategies by answering the following question. Since overfitting and model complexity are the two main reasons for MIA, *can we find a sub-network from the original over-parameterized NLP model that can prevent privacy leakage from MIA and can achieve competitive accuracy with the original NLP model?*

In order to propose an effective defense method, we have two ultimate goals. One is to prevent the privacy leakage of the model and the other is to guarantee the utility of the model.

The first goal of preventing privacy leakage is to find the target model f that can minimize the gain of the adversary. We first reformulate the gain function to quantitatively present how much privacy leakage information the adversary can get. According to (Nasr et al., 2018; Goodfellow et al., 2014), we rewrite the gain function of the adversary model in the form of probability distribution:

$$\begin{aligned} &G_f(f_A) \\ &= \int_{x,y} [P_D(x,y) p_f(f(x)) \log(f_A(x,y,f(x))) + \\ &P_{D'}(x,y) p'_f(f(x)) \log(1 - f_A(x,y,f(x)))] dxdy \\ &= -\log(4) + 2 \cdot JS(p_f(f(x)) || p'_f(f(x))) \end{aligned} \tag{1}$$

Where f_A is the adversary model. D is the training set and D' is the non-training set. p_f and p'_f are the probability distribution of the classification model f's output for training data and non-training data. $JS(p_f(f(x))||p'_f(f(x)))$ is the Jensen–Shannon divergence between the two distributions and it is always non-negative. The global minimum value that $G_f(f_A)$ can possibly have is -log(4) if and only if:

$$p_f(f(x)) = p'_f(f(x')) \tag{2}$$

This means that the prediction of classification model f has the same probability distribution for both the training set and non-training set. In this case, the attack fails in the sense the attacker can do no better than a random guess.

Then, the second goal is to ensure that the target model f's prediction accuracy. Suppose that the target NLP network $f(x)$ as:

$$f(x) = \mathbf{E}_n^f \circ \mathbf{E}_{n-1}^f \circ ... \circ \mathbf{E}_1^f (M(x))) \tag{3}$$

and we define the original NLP network $g(x)$ as:

$$g(x) = \mathbf{E}_n^g \circ \mathbf{E}_{n-1}^g \circ ... \circ \mathbf{E}_1^g (M(x))) \tag{4}$$

where $\mathbf{E}_i^f$, $\mathbf{E}_j^g$ is the encoder block. Each building block contains a self-attention layer and a fully connected feed-forward network.

The problem can be formulated as finding a sub-network $\hat{g}(x)$ that has competitive prediction accuracy with the original network $g(x)$.

We propose that the answer to the problem could be that we prune the model parameters as well as use the largest prediction gap of all predictions as the privacy objective and reduce the variance of its output while minimizing the classification loss.

4.2 Pruned Network Prediction Analysis

We first analysis and ensure the pruned model can still maintain the utility. A pruned network $\hat{g}(x)$ can be presented as :

$$\hat{g}(x) = \hat{\mathbf{E}}_n^g \circ \hat{\mathbf{E}}_{n-1}^g \circ ... \circ \hat{\mathbf{E}}_1^g(\mathbf{E}(x))) \quad (5)$$

where $\mathbf{P}_i$ is the pruning matrix in i-th layer.

Corollary 1. *For every network f defined in Eq. 3 with depth l and $\forall i \in \{1, 2, \ldots, n\}$. Consider g defined in Eq.4 as a randomly initialized neural network, and width $poly(d, n, m, 1/\epsilon, log1/\delta)$, where d is input size, n is number of layers in f, m is the maximum number of neurons in a layer. For the weights in $\mathbf{E}_i^g$, the weight initialization distribution belongs to uniform distribution in range [-1,1]. Then with probability at least $1 - \delta$ there is a weight-pruned sub-network $\hat{g}$ of g such that:*

$$\sup_{x \in \chi, \|W\| \leq 1} \|f(x) - \hat{g}(x)\| \leq \epsilon \quad (6)$$

Based on Corollary 1, we know that for every bounded distribution and every target network with bounded weights, there is a sub-network with an accuracy that is close to the original sufficiently over-parameterized neural networks.

4.2.1 Analysis on Feed-forward Linear Network

In this case, $f(x) = \mathbf{W} \cdot x$, and $g(x) = \left(\sum_{i=1}^d W_i\right) x$. **Corollary 2.** *Let $\mathbf{W}_1^*, ..., \mathbf{W}_n^*$ belongs to i.i.d. Uniform distribution over [-1,1], where $n \geq C \cdot log\frac{2}{\delta}$,where $\delta \leq min\{1, \epsilon\}$. Then, with probability at least 1-δ, we have*

$$\begin{aligned} &\exists S \subset \{1, 2, ..., n\}, \forall W \in [-0.5, 0.5], \\ &s.t \left|\mathbf{W} - \sum_{i \in S} \mathbf{W}_i^*\right| \leq \epsilon \end{aligned} \quad (7)$$

Lueker et al.(Lueker, 1998) proposed this theorem and had given a proof.

4.2.2 The Analysis in Self-attention Layer: General case

Consider a model $f(x)$ with only one self-attention layer, when the token size is n, $\mathbf{x} = (x_1, x_2, ..., x_n)$. let $(h_{..})_{n \times n} = \frac{\mathbf{QK}^T}{\sqrt{(d_k)}}$, then

$$\begin{aligned} f(x_i) &= softmax((h_{i.})_{1 \times n})\mathbf{V}_i \\ &= (\frac{\sum_j e^{h_{ij}}}{\sum_i \sum_j (e^{h_{ij}})})\mathbf{V}_i \\ &= (\frac{\sum_j e^{h_{ij}}}{\sum_i \sum_j (e^{h_{ij}})})\mathbf{W}^{\mathbf{V}_i} x_i \\ &= \mathbf{W}^{h_{i.}} x_i \end{aligned} \quad (8)$$

Corollary 3 *Let $\mathbf{W}_1^g, ..., \mathbf{W}_d^g$ belongs to i.i.d. uniform distribution over [-1,1], where $d \geq Clog\frac{2}{\delta}$, where $\delta \leq min\{1, \epsilon\}$. Then, with probability at least 1-δ, we have*

$$\begin{aligned} &\forall i \in \{1, 2, ..., n\}, \mathbf{W}_l^g \in [-1, 1], \\ &\exists p_l \in \{0, 1\}, \\ &s.t. \left|\mathbf{W}^{h_i} - (\sum_{l=1}^d p_l \mathbf{W}_l^g)\right| < \epsilon \end{aligned} \quad (9)$$

4.3 Gap Score Analysis

To guard the privacy disclosure, our goal is to find the target model f that minimizes the adversary's gain by adding a regularization term into the loss function, we consider a problem as :

$$\text{minimize } L(f) + \alpha \cdot r(\mathbf{z}_{max} - \mathbf{z}_{min}) \quad (10)$$

where r represents the regularization objective function and α is the coefficient to tune the impact between the training objective and privacy objective. To represent the gap score in the multi-class classification case, we show

$$\begin{aligned} &r(\mathbf{z}_{max} - \mathbf{z}_{min}) = \mathbf{z}_{max} - \mathbf{z}_{min} \\ &s.t. \ \mathbf{z}_{max} - \mathbf{z}_{min} \in [0, 1] \end{aligned} \quad (11)$$

so we have

$$\alpha \cdot r(\mathbf{z}_{max} - \mathbf{z}_{min}) \in [0, \alpha] \quad (12)$$

the update gradient can be calculated as:

$$\begin{aligned} \triangledown \mathbf{W} &= \frac{\partial L(\mathbf{W})}{\partial \mathbf{W}} + \alpha \cdot \frac{\partial r(\mathbf{z})}{\partial \mathbf{W}} \\ &= \frac{\partial L(\mathbf{W})}{\partial \mathbf{W}} + \alpha \cdot \frac{\partial (\mathbf{z}_{max} - \mathbf{z}_{min})}{\partial \mathbf{W}} \\ &= \frac{\partial L(\mathbf{W})}{\partial \mathbf{W}} + \alpha \cdot (\frac{\partial \mathbf{z}_{max}}{\partial \mathbf{W}} - \frac{\partial \mathbf{z}_{min}}{\partial \mathbf{W}}) \end{aligned} \quad (13)$$

In this case, when we update the model by minimizing the loss function, the gap score is also minimized. So the distribution of $p_f(f(x))$ and $p'_f(f(x'))$ are more similar than each other, i.e., $JS(p_f(f(x))||p'_f(f(x)))$ decreases and is closer to 0. Thus, the adversary has minimum gain for the trained model and privacy leakage is prevented.

Algorithm 1 The Process of GRIP

for *epoch* in *Epochs* **do**
  Get a random mini-batch S.
  for i in Iterations: **do**
    for Encoder k : **do**
      for self-attention layer: **do**
        Pruned $\{\mathbf{W}^Q\}$ to $\{P^s_{ik} \odot \mathbf{W}^Q\}$
        Pruned $\{\mathbf{W}^K\}$ to $\{P^s_{ik} \odot \mathbf{W}^K\}$
      end for
      for feed-forward network: **do**
        Pruned $\{\mathbf{W}\}$ to $\{P^{fc}_{ik} \odot \mathbf{W}\}$
      end for
    end for
  end for
  Get $\{\mathbf{z}_{max}\}$ and $\{\mathbf{z}_{min}\}$
  Calculate $r(\mathbf{z}_{max}, \mathbf{z}_{min})$
  Update $\{\mathbf{W}\}$, $\{\mathbf{W}^Q\}$,$\{\mathbf{W}^K\}$,$\{\mathbf{W}^V\}$
  by minimizing $L(f) + \alpha \cdot r(\mathbf{z}_{max} - \mathbf{z}_{min})$
end for
OUTPUT $\{\mathbf{W}\}$, $\{\mathbf{W}^Q\}$,$\{\mathbf{W}^K\}$,$\{\mathbf{W}^V\}$

4.4 Proposed Method: GRIP

We show our proposed method Gap score Regularization Integrated Pruning (GRIP) in Algorithm 1. For a fixed NLP classification model f, we set the sparsity $P = \{P_1, P_2, ..., P_k\}$ for k encoders, then we systematically prune the weights of each encoder in multiple iterations gradually, for both the self-attention layer and feed-forward network. When updating these weights, we minimize the loss function from Eq. 10 with the gap score regularization. The final model sparsity will be P.

5 Proposed Defense Evaluation

In this section, we apply our proposed to different NLP models with various datasets and tasks, mainly from two perspectives: the defense performance of our model and the computation cost benefit we obtain. All experiments are conducted on a server with Intel(R) Xeon(R) Gold 5218 (64 virtual CPUs with 504 GB memory) and 8 NVIDIA Quadro RTX 6000 GPUs (24GB memory) by PyTorch 1.5.1, Python 3.6, and CUDA 10.2.

5.1 Experimental Setup

Datasets. For the proposed sparse progressive distillation, we conduct experiments on the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2019), which is grouped into three categories of natural language understanding tasks (single-sentence tasks, similarity matching tasks, and natural language inference tasks) according to the purpose of tasks and difficulty level of datasets.

Models. We use the fine-tuned $\text{BERT}_{\text{BASE}}$ as a teacher and also initialize the student with the fine-tuned $\text{BERT}_{\text{BASE}}$. Specifically, we first fine-tune the pre-train $\text{BERT}_{\text{BASE}}$ on four GLUE tasks with four epochs, including SST-2, CoLA, MRPC, and RTE. We select the learning rate with best performance from $\{2e^{-}5, 3e^{-}5, 4e^{-}5, 5e^{-}5\}$. Batch size and maximum sequence length are set as 32 and 128, respectively.

Membership Inference Attacks Setup. To evaluate the neural network (NN) MIAs, we follow the model structure and setup in (Nasr et al., 2018) to construct and train the attack classifier. The detailed setting is described in Appendix D. For the metric MIAs evaluation, we adopt four metric attacks following the (Song and Mittal, 2021) and show the best attack accuracy in the tables.

Defense Training Setup. In our evaluation, we conduct the canonical implementation of training a model with differential privacy (DP)(Abadi et al., 2016) and the associated analysis in Pytorch implementation from Opacus (Yousefpour et al., 2021) library. We adopt the DP training into the original fine-tuning process and set the clipping bound to be 1.0. We find that the model is very hard to converge, so we set a large privacy budget with a total training epoch of 6 and report the best testing accuracy results in Table 2.

In our GRIP defense, we give different sparsity for every encoder, in every iteration, we gradually prune weight for both self-attention layers and feed-forward networks, then we will reach the sparsity after all iterations. In detail, we use sparsity 40% for CoLA and sparsity 60% pruning rate for the other datasets on the last 6 encoders and $\alpha = 1$ for all datasets on the pre-trained BERT model with 4 to 12 fine-tuning epochs and record the best classification accuracy results.

5.2 Results and Analysis

Table 2 summaries the classification accuracy and best attack accuracy for NN and metric MIAs on the undefended models, deferentially private trained models and our GRIP fine-tuned models.

GRIP can significantly reduce the membership inference risks. As shown in Table 2, our defense leads to a significant reduction in privacy risks in both NN and metric MIAs. For all evaluated datasets, we can control the MIA accuracy with neural network to $\sim 50\%$, which is close to a random guess, compared to the much higher attack accuracy on the undefended models from 60.94%

	RTE			MRPC			CoLA			SST-2		
Defense	None	DP	**GRIP**	None	DP	**GRIP**	None	DP	**GRIP**	None	DP	**GRIP**
Testing Accuracy	70.28%	53.79%	**61.01%**	84.39%	68.38%	**81.62%**	81.09%	71.80%	**81.20%**	92.89%	81.77%	**91.17%**
Accuracy Gap	28.11%	2.75%	**12.28%**	13.62%	0.93%	**5.27%**	15.53%	1.00%	**9.00%**	6.48%	1.31%	**2.83%**
NN MIA	84.38%	59.38%	**53.13%**	71.88%	53.13%	**53.13%**	60.94%	57.81%	**50.00%**	73.44%	60.94%	**57.81%**
Metric MIA	69.00%	54.20%	**57.80%**	59.10%	52.00%	**53.70%**	63.70%	51.50%	**56.90%**	58.50%	55.30%	**52.50%**

Table 2: Comparison of classification accuracy and membership attack accuracy between regular training, differential private training and GRIP training model

(CoLA) to 84.38% (RTE). Our defense can also outperform the DP training on the NN MIAs. For metric MIAs, although the attack accuracy with our GRIP is not always close to random guesses, we can still observe a $5 \sim 10\%$ decrease in accuracy even when the original MIA risk is not that high as the metric MIAs are mitigated when the accuracy gap between training and test data is not large, and overfitting is not obvious.

GRIP achieves privacy protection with a small cost on the utility loss. With all the benefits of the privacy defense from our proposed methods, the utility loss is limited in a small range at most times. Our GRIP training maintains the classification accuracy at the same level on CoLA and SST-2 dataset and causes a small 2.77% accuracy decrease on MRPC. Defense on the RTE dataset leads to 10% utility loss, but it is a very small dataset with limited training and testing data. The model is unstable with random separation on the training and testing data in each time of training and attack. Even in the worst cases, our approach can still largely outperform DP training as it leads to $10 \sim 20\%$ utility loss on all the datasets with very limited privacy protection on the NN MIAs. This is a case where the privacy budget is large and the model utility will be further reduced when the theoretical guarantees of DP training are obtained.

GRIP have significantly model storage and computation reduction. Tabel 3 summaries the weights reduction ratio of GRIP fine-tuned model on different datasets. Except for the benefit of privacy defense, our GRIP has an additional advantage on model storage and computations. Table 3 show that our GRIP has over $1.18\times$ ratio over different datasets.

In summary, we have the following analysis:

1. Reducing the overfitting of the NLP classification problem does not completely eliminate the membership privacy risk, which is consistent with the observation in Section 3.1. Taking the DP-trained model as an example, it successfully reduces overfitting as the accuracy gap is only $0.93 \sim 2.75\%$ on all datasets, which helps the models limit the metric MIAs to 55%. However, the NN MIAs remain at 60%, indicating that there is still privacy leakage on the poor utility models.

Data	Model	Weights (#)	Weights after prunning (#)	Weights reduction ratio
RTE	BERT	110 M	77 M	1.30 ×
MRPC	BERT	110 M	77 M	1.30 ×
CoLA	BERT	110 M	88 M	1.18 ×
SST-2	BRET	110 M	77 M	1.30 ×

Table 3: GRIP pruning ratios for different tasks.

2. Our GRIP works during training for both constraint of output prediction and reduction of model complexity of intermediate structures. As a result, we not only reduce model overfitting but also yield similar performance in terms of confidence and robustness for both training and test samples. For 'free lunch', we also reduce the model storage and the computations. Thus, our defenses can effectively resist MIAs and maintain good model utility.

5.3 Hyperparameter Analysis

Our proposed GRIP approach integrated with gap score regularization and pruning can successfully limit the maximum gain of the adversary model with a great privacy-utility trade-off. In this subsection, we further investigate the contribution of the proposed pruning and the proposed gap score regularization, respectively.

We first show the classification accuracy and NN MIA results on the four datasets using proposed pruning and proposed gap score regularization in Table 4. Compared to the baseline model results in Table 2, we can observe that each component of the proposed method can help reduce the attack accuracy with some utility loss. The proposed pruning methods achieve at most 31.25% (RTE) and on average 19.14% attack accuracy decrease for NN MIA with $0.23 \sim 7.23\%$ utility loss. The gap score regularization achieves better defense

Defense	Proposed Pruning		Gap Score Regularization	
Accuracy	Testing Accuracy	NN MIA	Testing Accuracy	NN MIA
RTE	63.05%	62.50%	58.12%	59.37%
MRPC	81.86%	65.63%	77.21%	57.81%
CoLA	80.50%	59.37%	80.70%	51.56%
SST-2	92.66%	67.18%	93.46%	57.81%

Table 4: Classification accuracy and NN MI accuracy on regular model with MIA-Pruning, and gap score regularization.

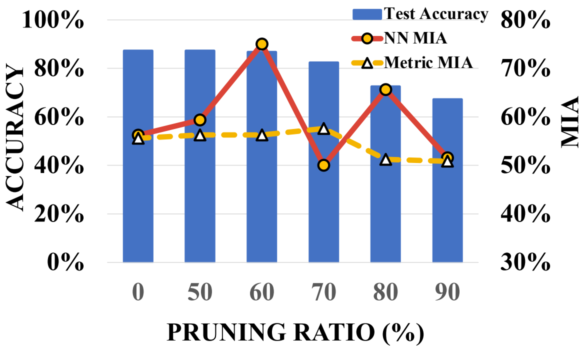


Figure 3: The effects of different pruning ratio on BERT for MRPC task.

against MIAs (16.02% decrease on average) while leading to a little bit more classification accuracy loss ($0 \sim 12.16\%$). In the following part of the subsection, we will demonstrate the effects of the individual proposed methods with more detailed ablation studies.

5.3.1 Proposed Pruning Algorithm

We investigate how our proposed pruning affects defense performance by pruning ratios. As shown in Figure 3, the attack accuracy of metric MIA decreases along with the higher pruning ratio when the pruning ratio is over 70%. However, the attack accuracy of NN MIA presents a fluctuation pattern when varying the pruning ratio. It reaches the minimum value when the pruning ratio is 70%.

5.3.2 Gap Score Regularization

In order to show the effects of the gap score regularization on the classification accuracy and MIAs defense, we tune the hyperparameter α that controls the impact of the regularization in training on RTE dataset as shown in Figure 4. α trades off the utility and privacy. With the increase of $alpha$, the constraint on the gap score becomes tighter and the gap score of the final result becomes smaller. Hence, the accuracy gap and classification accuracy decrease while the model can better defend against NN and metric MIA. Specifically, $alpha = 0.3$ in Figure 4 shows the case when the constraint is not

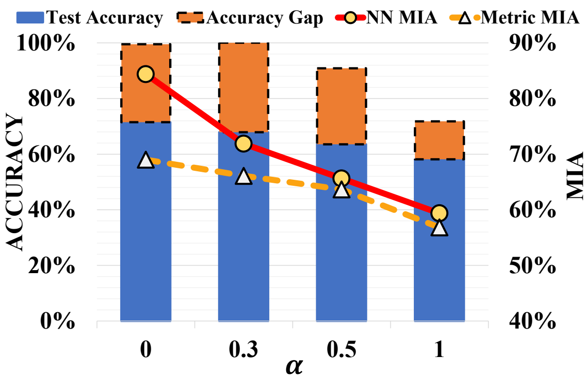


Figure 4: Different α for gap score regularization on RTE dataset.

large enough. The regularization starts to control the output and shows defensiveness, and this effect is first shown in a decrease in test accuracy, while the training data accuracy remains close to 100% and consequently the accuracy gap might increase.

Key takeaways: You may notice that our GRIP defense achieves a much better privacy-utility tradeoff than using the proposed pruning or gap score regularization alone. This is because GRIP is a combinatorial approach that benefits from pruning to derive a finer and sparser model structure that can better learn the proposed regularization and loss minimization during the fine-tuning process to control the final prediction distributions.

6 Conclusion

In this work, we explore NN MIAs and metric MIAs on NLP models. Our experiments show that MIAs exhibit higher attack capabilities in NLP models as compared to CV models. We further analyze the uniqueness of MIA in NLP models and develop a defense method GRIP that is based on weight pruning and gap score regularization. Our evaluations of the BERT model on RTE, MRPC, CoLA, SST-2 datasets show that GRIP achieves the privacy protection against MIAs with a substantially smaller cost on the utility loss compared with DP. The improvement comes from reduced overfitting and decreased model complexity leading to similar performance in terms of model output for both training and non-training samples. In addition, GRIP significantly reduces the model storage and computation cost, *e.g.*, it has approximately $1.30\times$ weight reduction ratio on RTE, MRPC, and SST-2 datasets. Overall, our MIA analyses and proposed defense, serve as an important step towards developing efficient and privacy-preserving deep learning models in NLP.

References

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*.

M Augasta and Thangairulappan Kathirvalavakumar. 2013. Pruning algorithms of neural networks—a comparative study. *Open Computer Science*, 3(3):105–115.

Zihang Dai, Hanxiao Liu, Quoc V Le, and Mingxing Tan. 2021. Coatnet: Marrying convolution and attention for all data sizes. *arXiv preprint arXiv:2106.04803*.

Fahim Dalvi, Hassan Sajjad, Nadir Durrani, and Yonatan Belinkov. 2020. Analyzing redundancy in pretrained transformer models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4908–4926.

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.

Cynthia Dwork. 2006. Differential privacy. In *International Colloquium on Automata, Languages, and Programming*, pages 1–12. Springer.

Cynthia Dwork. 2008. Differential privacy: A survey of results. In *International conference on theory and applications of models of computation*, pages 1–19. Springer.

Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D Cubuk, Quoc V Le, and Barret Zoph. 2021. Simple copy-paste is a strong data augmentation method for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2918–2928.

David Gollob. 2015. *Microsoft Azure-Planning, Deploying, and Managing Your Data Center in the*. Springer-verlag Berlin And Hei.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

Mitchell Gordon, Kevin Duh, and Nicholas Andrews. 2020. Compressing BERT: Studying the Effects of Weight Pruning on Transfer Learning. In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 143–155.

Fu-Ming Guo, Sijia Liu, Finlay S Mungall, Xue Lin, and Yanzhi Wang. 2019. Reweighted proximal pruning for large-scale language representation. *arXiv preprint arXiv:1909.12486*.

Song Han, Jeff Pool, John Tran, and William J Dally. 2015. Learning both weights and connections for efficient neural networks. *arXiv preprint arXiv:1506.02626*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778.

Agus Kurniawan. 2018. *Learning AWS IoT: Effectively manage connected devices on the AWS cloud using services such as AWS Greengrass, AWS button, predictive analytics and machine learning*. Packt Publishing Ltd.

Klas Leino and Matt Fredrikson. 2020. Stolen memories: Leveraging model memorization for calibrated white-box membership inference. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*, pages 1605–1622.

George S Lueker. 1998. Exponentially small bounds on the expected optimum of the partition and subset sum problems. *Random Structures & Algorithms*, 12(1):51–62.

Milad Nasr, Reza Shokri, and Amir Houmansadr. 2018. Machine learning with membership privacy using adversarial regularization. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 634–646.

Milad Nasr, Reza Shokri, and Amir Houmansadr. 2019. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE symposium on security and privacy (SP)*, pages 739–753. IEEE.

Md Atiqur Rahman, Tanzila Rahman, Robert Laganière, Noman Mohammed, and Yang Wang. 2018. Membership inference attack against differentially private deep learning model. *Transactions on Data Privacy*, 11(1):61–79.

Arvind Ravulavaru. 2018. *Google Cloud AI Services Quick Start Guide: Build Intelligent Applications with Google Cloud AI Services*. Packt Publishing Ltd.

Mauro Ribeiro, Katarina Grolinger, and Miriam AM Capretz. 2015. Mlaas: Machine learning as a service. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 896–902. IEEE.

Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. 2018. Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. *arXiv preprint arXiv:1806.01246*.

Virat Shejwalkar, Huseyin A Inan, Amir Houmansadr, and Robert Sim. 2021. Membership inference attacks against nlp classification models. In *NeurIPS 2021 Workshop Privacy in Machine Learning*.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE.

Liwei Song and Prateek Mittal. 2021. Systematic evaluation of privacy risks of machine learning models. In *30th {USENIX} Security Symposium ({USENIX} Security 21)*.

Liwei Song, Reza Shokri, and Prateek Mittal. 2019. Privacy risks of securing machine learning models against adversarial examples. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 241–257.

Stacey Truex, Ling Liu, Mehmet Emre Gursoy, Lei Yu, and Wenqi Wei. 2019. Demystifying membership inference attacks in machine learning as a service. *IEEE Transactions on Services Computing*.

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*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. GLUE: A Multi-task Benchmark and Analysis Platform for Natural Language Understanding. In *7th International Conference on Learning Representations, ICLR 2019*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. 2018. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pages 268–282. IEEE.

Ashkan Yousefpour, Igor Shilov, Alexandre Sablayrolles, Davide Testuggine, Karthik Prasad, Mani Malek, John Nguyen, Sayan Ghosh, Akash Bharadwaj, Jessica Zhao, Graham Cormode, and Ilya Mironov. 2021. Opacus: User-friendly differential privacy library in PyTorch. *arXiv preprint arXiv:2109.12298*.

Xueru Zhang, Chunan Huang, Mingyan Liu, Anna Stefanopoulou, and Tulga Ersal. 2019. Predictive cruise control with private vehicle-to-vehicle communication for improving fuel consumption and emissions. *IEEE Communications Magazine*.

Barret Zoph, Golnaz Ghiasi, Tsung-Yi Lin, Yin Cui, Hanxiao Liu, Ekin D Cubuk, and Quoc V Le. 2020. Rethinking pre-training and self-training. *arXiv preprint arXiv:2006.06882*.

A Metric MIAs

Correctness based MIA. This attack infers the membership according to whether a given input data x is classified correctly by the target model f (Yeom et al., 2018). The intuition is that training data are more likely to be correctly classified than test data. The attack $\mathcal{M}_{\text{corr}}$ is defined as follows, where $I(\cdot)$ indicates the indicator function.

$$\mathcal{M}_{\text{corr}}(f; x, y) = I(\text{argmax}\, f(x) = y) \quad (14)$$

Confidence based MIA. This attack determines the membership of the input x by comparing the most significant confidence score with the preset threshold. It is intuitive that the prediction confidence score $f(x)$ for the training data should be close to 1, while the prediction confidence for the test data is usually lower. The attack is first designed by (Salem et al., 2018) with a single threshold for all classes. (Song and Mittal, 2021) further improves it by applying class-wise thresholds to minimize the effect of inter-class confidence differences.The attack $\mathcal{M}_{\text{conf}}$ is defined as follows, where τ_y represents the threshold for the class y.

$$\mathcal{M}_{\text{conf}}(f; x, y) = I(\max f(x)_y \geq \tau_y) \quad (15)$$

Entropy based MIA. The entropy based MIA attack is first presented by (Salem et al., 2018), then followed by an enhanced version that uses the class-wise threshold τ_y(Song and Mittal, 2021). It is based on the fact that the prediction entropy of the test set should be much larger than that of the training set. It identifies the input x as a member if the prediction entropy is lower than the preset threshold. The attack $\mathcal{M}_{\text{entr}}(f; x, y)$ can be expressed as:

$$\mathcal{M}_{\text{entr}}(f; x, y) = I(-\sum_{i=0}^{k} f(x)_i \log(f(x)_i) \leq \hat{\tau}_y) \quad (16)$$

Here $\hat{\tau}_y$ denotes the threshold for class y, and k is the number of output classes.

Modified prediction entropy based MIA. (Song and Mittal, 2021) mentioned that prediction entropy attack has a major limitation that it does not contain any labeling information. As a result, only the confidence score is important in the calculation of the prediction entropy attack, without considering the correctness of the prediction. Both a highly correct label with a score close to 1 and a totally wrong predict with an incorrect label score close to 1 can lead to zero prediction entropy values. Modified prediction entropy (Song and Mittal, 2021) fixes this issue by: 1) only correct predictions with high probability 1 can be calculated to 0, and 2) incorrect predictions with high confidence scores are calculated to infinity. (Song and Mittal, 2021). Then such modified entropy $\mathrm{ME}(f(x), y)$ is presented as:

$$\begin{aligned}\mathrm{ME}(f(x), y) = &-(1-f(x)_y)\log(f(x)_y) \\ &-\sum_{i\neq y} f(x)_i \log(1-f(x)_i)\end{aligned} \tag{17}$$

The adversary determines an input data as a member if Eqn. is smaller than the preset class-related threshold–$\check{\tau}_y$ for class y. The attack $\mathcal{M}_{\text{Mentr}}(f; x, y)$ is defined as:

$$\mathcal{M}_{\text{Mentr}}(f; x, y) = I(\mathrm{ME}(f(x), y) \leq \check{\tau}_y) \tag{18}$$

B Analysis on Feed-Forward Networks

B.1 Analysis on Feed-Forward Networks: A simple layer with activation

In this case, $f(x) = w \cdot x$, $g(x) = \mathbf{u}\sigma(\mathbf{w}^g x)$. In [REF], they use σ as ReLU activation function, we have $w = \sigma(w) - \sigma(-w)$. So that the a single ReLU neuron can be written as:

$$x^* \mapsto \sigma(wx) = \sigma(\sigma(wx) - \sigma(-wx)) \tag{19}$$

On the other hand, this neuron can be present by a width m two layer network with a pruning matrix p^* for the first layer as:

$$x^* \mapsto \mathbf{u}\sigma(\mathbf{p} \odot \mathbf{w}^g x) \tag{20}$$

we define $\mathbf{w}^+ = max\{\mathbf{0}, \mathbf{w}\}$, $\mathbf{w}^- = min\{\mathbf{0}, \mathbf{w}\}$, $\mathbf{w}^+ + \mathbf{w}^- = \mathbf{w}^g$. Combine Eq. 19 and 20 we have:

$$x^* \mapsto \mathbf{u}\sigma\left(\sigma\left(\mathbf{p} \odot \mathbf{w}^+ x\right) - \sigma\left(\mathbf{p} \odot -\mathbf{w}^- x\right)\right) \tag{21}$$

Base on Theorem 2, when $n \geq Clog\frac{4}{\epsilon}$, there exist a pattern of $\mathbf{w}$, such that, with probability $1-\epsilon/2$,

$$\begin{aligned}&\forall\, w^f \in [0,1], \exists\, p \in 0, 1^n, \\ &s.t. \left|w^f - \mathbf{u}\sigma(\mathbf{p} \odot \mathbf{w}^+)\right| < \epsilon/2\end{aligned} \tag{22}$$

Similarly, we have $\mathbf{w}$, such that, with probability $1-\epsilon/2$,

$$\begin{aligned}&\forall\, w^f \in [0,1], \exists\, p \in 0, 1^n, \\ &s.t. \left|w^f - \mathbf{u}\sigma(\mathbf{p} \odot \mathbf{w}^-)\right| < \epsilon/2\end{aligned} \tag{23}$$

so combine Eq.36 and 23, we have:

$$\begin{aligned}&\sup\left|w^f x - \mathbf{u}\sigma(\mathbf{p}\odot \mathbf{w}x)\right| \\ &\leq \left|\sigma(w^f)x - \sigma(-w^f)x - \mathbf{u}\sigma(\mathbf{p}\odot\mathbf{w}^+ x) - \mathbf{u}\sigma(\mathbf{p}\odot\mathbf{w}^- x)\right| \\ &\leq \sup\left|\sigma(w^f)x - \mathbf{u}\sigma(\mathbf{p}\odot\mathbf{w}^+ x)\right| + \\ &\quad \sup\left|\sigma(w^f)x - \mathbf{u}\sigma(\mathbf{p}\odot\mathbf{w}^- x)\right| \\ &\leq \epsilon/2 + \epsilon/2 \\ &\leq \epsilon\end{aligned} \tag{24}$$

B.2 Analysis on Feed-Forward Networks: a Neuron

In this case, $f(x) = \mathbf{w}^f\mathbf{x}$, $g(x) = \mathbf{u}\sigma(\mathbf{w}\mathbf{x})$ and $\hat{g}(x) = \mathbf{u}\sigma(\mathbf{p}\odot\mathbf{w}\mathbf{x})$

$$\begin{aligned}&\sup\left|\mathbf{w}^f\mathbf{x} - \mathbf{u}\sigma(\mathbf{p}\odot\mathbf{w}\mathbf{x})\right| \\ &\leq \sup\left|\sum_{i=1}^{m}\left(w_i^f x_i - \mathbf{u}_i\sigma(\mathbf{p}_i\odot\mathbf{w}_i x_i)\right)\right| \\ &\leq \sup\sum_{i=1}^{m}\left|w_i^f x_i - \mathbf{u}_i\sigma(\mathbf{p}_i\odot\mathbf{w}_i x_i)\right| \\ &\leq \sum_{i=1}^{m}\sup\left|w_i^f x_i - \mathbf{u}_i\sigma(\mathbf{p}_i\odot\mathbf{w}_i x_i)\right| \\ &\leq m\cdot\frac{\epsilon}{m} \\ &\leq \epsilon\end{aligned} \tag{25}$$

B.3 Analysis on Feed-Forward Networks: a Layer

In this case, $f(x) = \mathbf{W}^f\mathbf{x}$, and $g(x) = \mathbf{u}\sigma(\mathbf{W}^g\mathbf{x})$, and $\hat{g}(x) = \mathbf{u}\sigma(\mathbf{p} \odot \mathbf{W}^g\mathbf{x})$

$$
\begin{aligned}
&\sup \left|\mathbf{W}^f\mathbf{x} - \mathbf{u}\sigma(\mathbf{p} \odot \mathbf{W}^g\mathbf{x})\right| \\
&\leq \sup \left|\sum_{j=1}^{k}\sum_{i=1}^{m}\left(w_{j,i}^{f}x_i - \mathbf{u}_i\sigma(\mathbf{p}_{j,i} \odot \mathbf{w}_{j,i}x_i)\right)\right| \\
&\leq \sup \sum_{j=1}^{k}\sum_{i=1}^{m}\left|w_{j,i}^{f}x_i - \mathbf{u}_i\sigma(\mathbf{p}_{j,i} \odot \mathbf{w}_{j,i}x_i)\right| \\
&\leq \sum_{j=1}^{k}\sum_{i=1}^{m}\sup\left|w_{j,i}^{f}x_i - \mathbf{u}_i\sigma(\mathbf{p}_{j,i} \odot \mathbf{w}_{j,i}x_i)\right| \\
&\leq k \cdot m \cdot \frac{\epsilon}{mk} \\
&\leq \epsilon
\end{aligned} \tag{26}
$$

B.4 The analysis in Entire Feed-Forward Networks

For general case , $f(x)$ is defined as Eq.3, $g(x)$ is defined as Eq.4. so with the probability over $1 - \epsilon$, we have:

$$
\begin{aligned}
&\sup \|f(x) - \hat{g}(x)\| \\
&= \left\|\mathbf{W}_n\mathbf{x}_n - \mathbf{P}_{2n} \odot \mathbf{W}_{2n}^{g}\mathbf{x}_n^{g}\sigma(\mathbf{P}_{2n-1} \odot \mathbf{x}_{2n-1}^{g})\right\| \\
&\leq \|\mathbf{W}_n\mathbf{x}_n - \mathbf{W}_n\mathbf{x}_n^{g}\| + \\
&\quad \left\|\mathbf{W}_n\mathbf{x}_n^{g} - \mathbf{P}_{2n} \odot \mathbf{W}_{2n}^{g}\mathbf{x}_n^{g}\sigma(\mathbf{P}_{2n-1} \odot \mathbf{x}_{2n-1}^{g})\right\| \\
&\leq \|\mathbf{x}_n - \mathbf{x}_n^{g}\| + \\
&\quad \left\|\mathbf{W}_n\mathbf{x}_n^{g} - \mathbf{P}_{2n} \odot \mathbf{W}_{2n}^{g}\mathbf{x}_n^{g}\sigma(\mathbf{P}_{2n-1} \odot \mathbf{x}_{2n-1}^{g})\right\| \\
&\leq \epsilon/2 + \epsilon/2 \\
&\leq \epsilon
\end{aligned} \tag{27}
$$

C MIA formulation

For the target machine learning model, we consider the classification model in this work. Let f denotes the target classification model, x denotes a data point, and $f(x)$ denotes the output of f on data x. $f(x)$ is a one-hot vector of probabilities of x belonging to k classes. We consider the MIA problems in a black-box condition, which means the adversary can not access the classification model's parameters but can only observe the input and output of the classification model. We assume that the adversary has access to some data records from the training set and the predictions from the black-box DNN target model. Based on the difference between the model's prediction on the training dataset and the non-training dataset, the adversary can determine whether a data record belongs to the model's training dataset or not. We use f_A to denote the adversarial inference model $f_A : x \times y \times f(x) \longrightarrow [0, 1]$. f_A takes the feature of the data x, the label of the data y, and the prediction of the classification model $f(x)$ as inputs. f_A outputs the probability of data (x, y) belonging to the training set D or the non-training set D'. The probability distributions of samples in D and D' are P_D and $P_{D'}$, respectively. The gain function of the inference model f_A given the classification model f can be written as:

$$
\begin{aligned}
G_f(f_A) = &\mathop{\mathbb{E}}_{(x,y)\sim P_D}\left[\log(f_A(x, y, f(x)))\right] \\
&+ \mathop{\mathbb{E}}_{(x,y)\sim p_{D'}}\left[\log(1 - f_A(x, y, f(x)))\right]
\end{aligned} \tag{28}
$$

The first expectation computes the inference model's accuracy in predicting training data (members), and the second expectation computes the accuracy of the inference model on predicting non-training data (non-members). The underline probability P_D and $P_{D'}$ is normally not known. The empirical gain can be calculated by simply sampling data from the training set and validation set. Intuitively, weight pruning can prevent over-fitting. Thus it will have a smaller d.

According to (Nasr et al., 2018), we rewrite the gain function of the inference model in the form of probability distribution:

$$
\begin{aligned}
&G_f(f_A) = \\
&\int_{x,y} [P_D(x, y)p_f(f(x)) \log(f_A(x, y, f(x)))+ \\
&P_{D'}(x, y)p'_f(f(x)) \log(1 - f_A(x, y, f(x))]dxdy
\end{aligned} \tag{29}
$$

where D is the training set and D' is the non-training set. p_f and p'_f are the probability distribution of the classification model f's output for training data and non-training data.

For a given classification model f and data sampled from a known probability distribution, the optimal determination solution for the inference model f_A is (Goodfellow et al., 2014; Nasr et al., 2018):

$$
f_A^*(x, y, f(x)) = \frac{p_f(f(x))}{p_f(f(x)) + p'_f(f(x'))} \tag{30}
$$

Therefore, by substituting f_A^* in the Equation 28, the gain function of f_A^* can be written as:

$$
\begin{aligned}
&G_f(f_A^*) \\
&= \mathop{\mathbb{E}}_{(x,y)\sim P_D}\big[\log(\frac{p_f(f(x))}{p_f(f(x)) + p'_f(f(x))})\big] + \\
&\quad \mathop{\mathbb{E}}_{(x,y)\sim p_{D'}}\big[\log(1 - \frac{p_f(f(x))}{p_f(f(x)) + p'_f(f(x))})\big] \\
&= -\log(4) + 2 \cdot JS(p_f(f(x))||p'_f(f(x)))
\end{aligned}
\tag{31}
$$

Where $JS(p_f(f(x))||p'_f(f(x)))$ is the Jensen–Shannon divergence between the two distributions. Since $JS(p_f(f(x))||p'_f(f(x)))$ is always non-negative and equals 0 if and only if $p_f(f(x)) = p'_f(f(x'))$, the global minimum value that $G_f(f_A^*)$ can possibly have is -log(4) if and only if $p_f(f(x)) = p'_f(f(x'))$ (Goodfellow et al., 2014). This means that the prediction of classification model f for both the training set and non-training set has the same probability distribution. In this case, the attack fails in the sense the attacker can do no better than a random guess. We use d to represent the Jensen–Shannon divergence $JS(p_f(f(x))||p'_f(f(x)))$ between the probability distributions of f's outputs for the training set and non-training set. The larger d is, the higher the maximum gain of the reference model is. In other words, the more vulnerable the classification model is. Thus, any method that reduces d can reduce the attack success rate of the MIA.

D Neural Network based Membership Inference attack models setup

The attack classifier takes two pieces of information as input. One is the unsorted confidence score vector, and the other one is the label of the input data that is one hot encoded (all elements except the one that corresponds to the label index are 0). The classifier consists of three fully connected subnetworks. The one operates on the confidence score vectors has three layers with size 1024,512 and 64. One network with two layers with 512 and 64 neurons works on the label. The third network is the combined network that takes the outputs of the two networks as a concatenate input and has five layers with sizes 512,256,128,64 and 1. The final output will predict whether the input belongs to the train-set or not with a probability (larger than 0.5 will count as a member). We use the ReLu activation function for the network except for the final output layer with the sigmoid activation function. We train the attack classifier with Adam optimizer and mean squared error (MSE) criterion for a total of 300 epochs. To better generate the model, we set the initial learning rate to 0.001 and decays by 0.1 in the 30th epoch.

E Gain function

According to (Nasr et al., 2018), we rewrite the gain function of the inference model in the form of probability distribution:

$$
\begin{aligned}
&G_f(f_A) = \\
&\int_{x,y} [P_D(x,y) p_f(f(x)) \log(f_A(x,y,f(x))) + \\
&P_{D'}(x,y) p'_f(f(x)) \log(1 - f_A(x,y,f(x))] dx dy
\end{aligned}
\tag{32}
$$

where D is the training set and D' is the non-training set. p_f and p'_f are the probability distribution of the classification model f's output for training data and non-training data.

For a given classification model f and data sampled from a known probability distribution, the optimal determination solution for the inference model f_A is (Goodfellow et al., 2014; Nasr et al., 2018):

$$
f_A^*(x,y,f(x)) = \frac{p_f(f(x))}{p_f(f(x)) + p'_f(f(x'))} \tag{33}
$$

F The analysis in self-attention layer: a simple case

the self-attention layer can be present as:

$$
\mathbf{Z} = softmax(\frac{QK^T}{\sqrt{(d_k)}})V \tag{34}
$$

Where $Q = W^Q x$, $K = W^K x$, $V = W^V x$ Here, we start from a simple example. Consider a model $f(x)$ with only one self-attention layer, when the token size of input x is 1, $softmax(\frac{QK^T}{\sqrt{(d_k)}}) = 1$, we have

$$
f(x) = W^V x \tag{35}
$$

consider $g(x) = \left(\sum_{i=1}^d w_i^g\right) x$. and a pruning vector $\mathbf{p} = (p_1, p_2, ..., p_d)$. Base on Theorem 2, when $d \geq Clog4/\epsilon$, there exist a pattern of $p_i w_i^g$, such that, with probability $1 - \epsilon$,

$$
\begin{aligned}
&\forall w_i^g \in [-1,1], \exists p_i \in \{0,1\}, \\
&s.t. \left| W^V - (\sum_{i=1}^d p_i w_i^g) \right| < \epsilon
\end{aligned}
\tag{36}
$$