Membership Inference Attack on zh-en NLP translation model

Tzu-Quan Lin*
Department of Computer Science and Information Engineering
National Taiwan University
b07902054@csie.ntu.edu.tw

Chun-Yao Chang*
Department of Computer Science and Information Engineering
National Taiwan University
b07902022@csie.ntu.edu.tw

Abstract

Data privacy problem has gain more and more importance from society. When users utilize enterprise's service through API, their data might somehow leak to the enterprise. Users might be curious about whether their data has been used to further improve the performance of the model. We formulate this problem as a type of membership inference problem: given a data sample and black-box access to a model's API, determine whether the sample existed in the model's training data. We focus on a zh-en NLP translation model, investigating the possibility of this kind of model to be attacked. Our contribution is that we conduct comprehensive experiment, showing that it is quite difficult to membership inference attack a sequence to sequence model, and providing reasonable explanations on our experiment results. All of our experiment results is reproducible, meanwhile, our codes are released to public in this repository: https://github.com/nervjack2/SPML_FinalProject

1 Introduction and Motivation

As machine learning shows promising result in various field, more and more enterprises utilize the technique of machine learning to improve quality of their products. Large international enterprises such as Google and Microsoft have capability to collect a huge amount of data for training. Many of them provide public API to people for accessing their models. This process often requires users to upload their private data to server, and they would return the inference result of their model to users.

Despite this kind of services make our daily life much more convenience, the potential privacy problem is actually severe than people thinks. For example, consider a photo edition API which provides a platform for people to exquisite their photo. This API requires people to upload their private photos first for further edition. Many people won't think too much before uploading their photos onto the server, however, once you upload your photos, it is possible that the service provider secretly makes a copy of them for some purposes. In this article, we focus on a specific privacy leakage scenario, in which service provider might secretly utilize users' data for improving performance of their model.

This problem can be formulated as a type of membership inference problem, first introduced by Shokri et al. [1] and defined as: "Given a machine learning model and a record, determine whether this record was used as part of the model's training dataset or not." This problem can be interpreted as an adversarial framework: service provider would like this problem to be unanswerable, and users are

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

interested in answering this question with high accuracy in order to determine whether their private data have been illegally used as the training data of the model.

Since Shokri proposed the problem of membership inference, researchers have proposed many ways to attack and defend the privacy of various types of models. However, most of the works so far has only focused on standard classification problems, where the output space of the model is a fixed set of labels. Instead, Sequence generation problems are more complex than classification problems, and it is unclear whether the methods and results developed for membership inference in classification problems could transfer. So, it still remains a lot of researching space for membership inference attack in sequence generation problem.

To further simplify the problem framework, we decide to focus on a specific type of sequence generation model. We choose zh-en NLP translation model as our investigation target. Given a public zh-en translation API, we as a user of this API, are curious about whether our data has been used to train the service provider's model.

One might be confuse how an arbitrary text could be used for training a translation model without any parallel translation label given. When service provider receive an arbitrary input text, he could inference it first and check out the confidence score of the output translation result. If the score is higher than a threshold, he could consider to add the input text and its corresponding translation result into his training data.

2 Related work

Papers related to membership inference attacks on NLP tasks are relatively rare comparing to CV tasks. Shejwalkar et al. [2] study the susceptibility of membership inference attacks on NLP classification models, used for text classification tasks. Mahloujifar study membership inference attacks on word embeddings and their effect in other NLP tasks that use these embeddings. Hisamoto [4] study membership inference attacks on NLP sequence generation model, which is quite similar as what we have done in this project.

3 Problem Definition

For explanation convenience, we introduce two characters: Alice and Bob.

3.1 Alice, the service provider

Alice, being a service provider, her mission is to train a zh-en translation model and provide model's inference API to users. We assume that Alice has a undisclosed dataset A_{train}, which can be used for training her model.

3.2 Bob, the user

Bob, being a user of Alice's model, is interested in whether Alice secretly utilize his data for training. In order to achieve his goal, Bob has to train a binary classifier, which could classify whether a given data sample is in A_{train}. Formally speaking, define x as the input text, y as the output prediction of Alice's model, $\hat{y}$ as the translation label, g as the classifier. Bob's target is to train a classifier g that satisfies the following equation:

$$g(x, y, \hat{y}) = \begin{cases} 0, & \text{if } (x, y) \notin A_{train} \\ 1, & \text{if } (x, y) \in A_{train} \end{cases} \tag{1}$$

Also, we assume that Bob has the ability to collect some data from public dataset. We notate his own dataset as B_{all}. Bob can utilize B_{all} whatever he wants to train the classifier g.

4 Dataset

We choose three different kinds of zh-en NLP translation dataset to implement our experiments. In Figure 1, $D1$, $D2$ and $D3$ are UN Parallel Corpus, News Commentary and Wikititiles respectively.

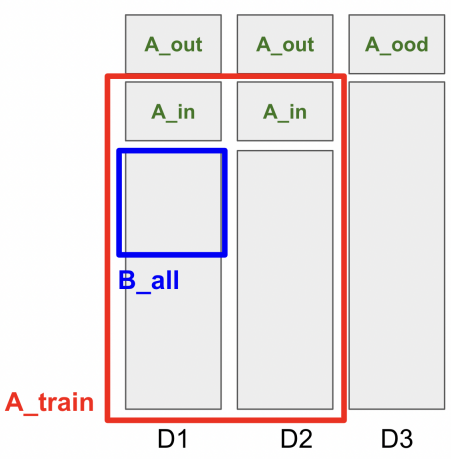

Figure 1: We use three different zh-en translation dataset $D1$, $D2$, $D3$, and split the data into A_{train}, B_{all}, A_{in}, A_{out}, A_{ood}. Note that $B_{all} \subset A_{train}$, and B_{all} only contains data from dataset $D1$.

A_{train} contains data sampled from $D1$ and $D2$. B_{all} only contains data sampled from $D1$, and it is worth noting that B_{all} is a subset of A_{train}. The reason why we assume $B_{all} \subset A_{train}$ is that Alice as a service provider from a large enterprise, she must have the ability to collect any dataset which Bob have.

A_{in}, A_{out}, A_{ood} in Figure 1 are split for Bob to test his classifier. A_{in}, A_{out} represents data in and not in A_{train} respectively. A_{ood} represents data sampled from a out of domain dataset $D3$. The motivation why we add out of domain data into our testing set is inspired by Koehn and Knowles [5]. They found the fact that sequence to sequence machine translation models act greatly different on out of domain dataset.

Last but not least, the number of data in A_{train} and B_{all} is also a problem worth considering. In our assumption, Alice is a service provider from a large enterprise, so the amount of data she own must significantly more than Bob's. The following is our experiment setting: Alice owns 800000 pair of data and Bob owns 40000. It is difficult for Bob to train a strong classifier in our setting, because the amount of data he could utilize is very small, and the amount of data Alice has is 20 times that of Bob.

5 Translation Model

5.1 Alice's model

We use a 6-layer transformers to train Alice's model. Because of the limitation of computational device, we decide to fine-tune the model with loading pretrained checkpoint "Helsinki-NLP/opus-mt-zh-en" from huggingface hub [6]. We train the model with 2e-5 learning rate, 16 batch size for 55000 steps. The BLEU score of Alice's model is 36.04.

5.2 Bob's shadow model

Of course, Bob do not know which data is in A_{train}. So he should train a shadow model to imitate the behavior of Alice's model. First, Bob needs to split his dataset B_{all} into three parts B_{train}, B_{in} and B_{out} as we can see in Figure 2. Next, Bob use B_{train} to train a shadow zh-en translation model. Last, Bob collects the inference result of B_{in} and B_{out} of the shadow model, and utilizes the results to train a classifier g. The training detail of the classifier g will be discussed in section 6.

Same as Alice, we use a 6-layer transformers to train Bob's model. However, we choose to load different checkpoint "liam168/trans-opus-mt-zh-en" from huggingface hub. We train the model with 2e-5 learning rate, 8 batch size for 2000 steps. The BLEU score of the shadow model is 32.19.

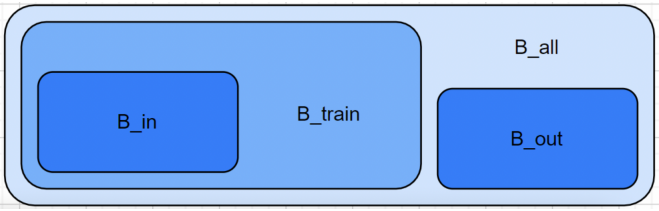

Figure 2: Bob needs to split his dataset B_{all} into three parts: B_{train}, B_{in} and B_{out}

.

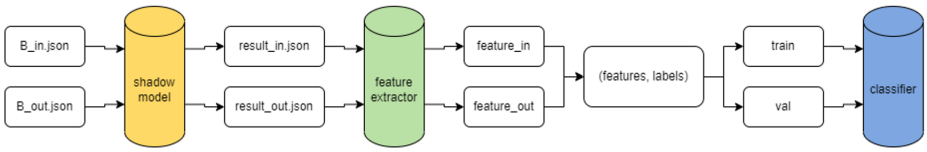

Figure 3: Attack flow under algorithm 1: one sentence per probe setting

.

6 Attack Details

To perform membership inference attack, Bob needs to train a binary classifier to distinguish whether the data is in probe or out probe. Here, we describe two algorithms of attacks and evaluations.

For the two algorithms, the data source N will be B_{in}, B_{out} during training phase, and $A_{in}.A_{out}$ during testing phase.

In general, algorithm 1 will first get the translated result predicted by Bob's shadow model. Then, it will extract sentence-level features from the feature extractor. Finally, it will train a classifier that is able to predict in or out given a single $(x, y, \hat{y})$ pair. On the other hand, algorithm 2 will first get the translated result predicted by Bob's shadow model. Then for training phase, it will sample 6000 groups, each group contains 500 $(x, y, \hat{y})$ pairs with same $\hat{l}$. Then, it will extract corpus-level features from the feature extractor. Finally, it will train a classifier that is able to predict in or out given 500 $(x, y, \hat{y})$ pairs with same $\hat{l}$.

From the description above, we can see that algorithm 1 is more difficult for Bob to perform attack since he can only utilize sentence-level information in algorithm 1. While he can utilize both sentence-level and corpus-level information in algorithm 2.

6.1 Feature Selection

In algorithm 1, the feature extractor will extract modified 1- to 4-gram precisions and smoothed sentence-level BLEU score from the given sentence pair. The intuition is that if an unusually large number of n-grams in y matches $\hat{y}$, then it could be a sign that this was in the training data and Alice memorized it.

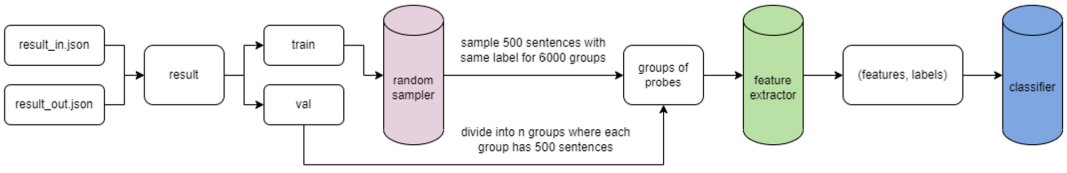

Figure 4: Attack flow under algorithm 2: grouping probes setting

.

Classifier \data source	$Alice$	Bob_{train}	Bob_{val}	OOD
Random Forest Tree	0.497	0.697	0.522	0.719
XGBoost	0.484	0.550	0.527	0.722
MLPClassifier	0.471	0.525	0.527	0.722
KNN	0.494	0.602	0.508	0.245

Table 1: Binary classification accuracy of algorithm 1.

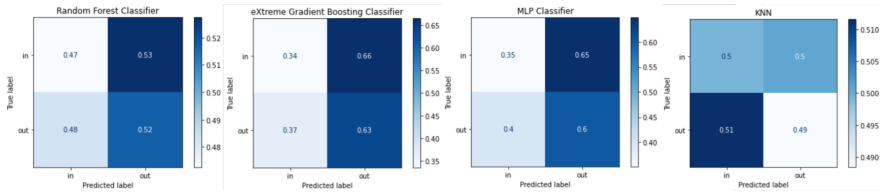

Figure 5: Confusion matrix of four classifiers

.

In algorithm 2, the feature extractor will extract sentence BLEU bin percentage and corpus BLEU score from the given group of sentences. The intuition is that different groups with same $\hat{y}$ might have similar distribution of sentence BLEU score, and the corpus BLEU score may also be closed to each other.

6.2 Classifers

In the experiment, we adopt four different kinds of classifiers provided in scikit-learn [7] and XGBoost [8].

1. Random forest classifier with 100 estimators.
2. XGBoost classifier with 100 estimators.
3. MLP classifier with hidden layer size 100, relu activating function, l_2 regularization 1e-4.
4. K nearest neighbors classifier with k = 5.

7 Experiment Result

In this section, we'll conduct several experiments and analyze the result. For the evaluation protocol, we use binary classification accuracy for classification experiment and R^2 score for regression experiment. Note that in this section, Bob_{train} is different to the dataset used to train Bob's shadow model.

7.1 Algorithm 1: One Sentence per Probe

From table 1, we can see that the accuracy on $Alice$ and Bob_{val} is around 50%, meaning that the attack is not successful. Also, we notice that the accuracy on Bob_{train} is not high as well, revealing that the classifier is actually underfitting. The reason of underfitting results from the features we select don't show a significant difference between in or out. Despite the unsuccess, three of the four classifiers show a relative high accuracy in OOD (out-of-domain) data, which suggests that is has a better chance to infer the membership for OOD data.

7.2 Algorithm 2: Grouping Probes

Since algorithm 1 is too difficult for Bob to attack, we then turn to algorithm 2, which is a more favorable to Bob. From table 2, we can see that the accuracy on Bob_{train} and Bob_{val} is much higher than that in algorithm 1. However, the accuracy on Alice still remain the same. This is due to the fact that Bob's shadow model has lower BLEU scores than Alice's target model. To erase the difference of translate model performance, we throw B_{train} into both models and calculate their mean BLEU scores, respectively. Then, we use the difference of means BLEU scores to adjust the feature values.

Classifier \data source	$Alice$	$Alice$ (adjusted)	Bob_{train}	Bob_{val}	OOD
Random Forest Tree	0.5	0.602	1.0	0.834	0.54
XGBoost	0.5	0.595	1.0	0.839	1.0
MLPClassifier	0.488	0.490	0.986	0.736	1.0
KNN	0.506	0.508	0.788	0.643	1.0

Table 2: Binary classification accuracy of algorithm 2.

Classifier \data source	$Alice$	$Alice$ (adjusted)	Bob_{train}	Bob_{val}	OOD
Random Forest Tree	0.492	0.502	1.0	0.592	1.0
XGBoost	0.498	0.504	0.999	0.598	1.0
MLPClassifier	0.482	0.488	0.939	0.508	1.0
KNN	0.484	0.484	0.689	0.494	0.98

Table 3: Binary classification accuracy of algorithm 2.1.

This works for random forest tree and XGBoost as their adjusted accuracy on $Alice$ are higher. This is the first strong general result for Bob, suggesting the membership inference attacks are possible if probes are defined as groups of sentences.

7.2.1 Stricter Classification Setting

Despite the positive result in algorithm 2, its input constraint is difficult to be applied in real-world application. Therefore, we loosen the input constraint so that the input do not need to be same $\hat{y}$. The problem definition now becomes "Of the 500 sentences, whether the number of sentences used in $Alice$ training set is larger than that not used in Alice.". Note that this is still a binary classification problem. From table 3, we can see that both the accuracy and adjusted accuracy on Alice drop, which is reasonable due to stricter problem setting. But to our surprise, the accuracy on Bob_{val} is higher than 0.5 for the first two classifiers. This might indicate that we can move forward to a stricter regression problem.

7.2.2 Regression Setting

Inspired by the result of algorithm 2.1, we change the classification problem to a regression problem. The problem definition now becomes "Of the 500 sentences, how many sentences are used in the Alice training set?". Apparently, this is stricter than the classification. From table 4. we can see that the R^2 score on $Alice$ and Bob_{val} is negative, meaning that it's even worse than simply guess the mean value. The result shows that the features we select are not correlated with the number of sentences in Alice training set.

7.3 Other Classifiers

From the results above, we know that human-defined features have their own limitation. Thus, instead of using self-constructed feature extractor, we can use a more complex classifier that directly takes the $(x, y, \hat{y})$ pair as input and output the predicted label. Here, we choose two classifier. The first one is pretrained BERT [9] that takes takes the $(y, \hat{y})$ pair as input and output the predicted label. The second one is pretrained mT5 [10] that takes takes the $(x, y, \hat{y})$ pair as input and output the predicted label. In this section, we follow the algorithm 1 setting.

From table 5, we can see that although the performance of mt5-small is slightly better than that of bert-base-uncased, the result of both method is similar to previous result of classifier with human-

Regressor \data source	$Alice$	$Alice$ (adjusted)	Bob_{train}	Bob_{val}	OOD
Random Forest Tree	-0.022	-0.021	0.859	-0.008	0.0
XGBoost	-0.201	-0.221	0.864	-0.177	0.0
MLPRegressor	-0.012	-0.006	0.025	-0.011	0.0
KNN	-0.168	-0.168	0.193	-0.192	0.0

Table 4: R^2 regression score of algorithm 2.2.

Classifier \data source	$Alice$	Bob_{train}	Bob_{val}	OOD
Bert-base-uncased	0.495	0.567	0.518	0.722
mt5-small	0.533	0.590	0.525	0.715

Table 5: Binary classification accuracy of algorithm 1.

defined features. Indicating that sentence-level attack is still very difficult for Bob even with more complex classifier.

8 Conclusion

In our final project, we discuss the feasibility of membership inference attacks on zh-en NLP translation model. In algorithm 1, Alice is generally safe and it is difficult for Bob to infer the sentence-level membership. In algorithm 2, for a looser definition of membership attack on groups of sentences, the attacker can win at a level above chance. However, its input constraint is difficult to be applied in real-world application. What's more, once we loosen the input constraint, the accuracy drops to the same level as algorithm 1. Lastly, using a more complex end-to-end classifier does not show significant improvement under algorithm 1.

From the experiment and result, we can say that in contrast to attacks on standard classification problems, sequence generation problems maybe be harder to attack because the input and output spaces are far larger and complex, making it difficult to determine the quality of the model output or how confident the model is. However, our attack approach was a simple one, using shadow models to mimic the target model. Bob can attempt more complex strategies, for example, by using the translation API multiple times per sentence. Bob can manipulate a sentence, for example, by dropping or adding words, and observe how the translation changes. We'll leave it as future work for further investigation.

9 Reference

[1] Membership Inference Attacks against Machine Learning Models (2017, Reza Shokri, Marco Stronati, Congzheng Song, Vitaly Shmatikov)

[2] Membership Inference Attacks Against NLP Classification Models (2021, Virat Shejwalkar, Amir Houmansadr, Huseyin A. Inan2, Robert Sim)

[3] Membership Inference on Word Embedding and Beyond (2021, Saeed Mahloujifar, Huseyin A. Inan, Melissa Chase, Esha Ghosh, Marcello Hasegawa)

[4] Membership Inference Attacks on Sequence-to-Sequence Models: Is My Data In Your Machine Translation System? (2019, Sorami Hisamoto, Matt Post, Kevin Duh)

[5] Six Challenges for Neural Machine Translation (2017, Philipp Koehn, Rebecca Knowles)

[6] HuggingFace's Transformers: State-of-the-art Natural Language Processing (2019, Thomas Wolf et al.)

[7] Scikit-learn: Machine Learning in Python. `https://scikit-learn.org/stable/`

[8] XGBoost. `https://xgboost.readthedocs.io/en/stable/`

[9] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (2018, Jacob Devlin et al.)

[10] mT5: A massively multilingual pre-trained text-to-text transformer (2020, Linting Xue et al.)