PATE FRAMEWORK FOR PRIVACY-PRESERVING DEEP LEARNING

Urjit Patil(up63), Athava Bhusari(ab2414)
{up63, ab2414}@scarletmail.rutgers.edu
Department of Statistics
Rutgers University
New Brunswick, NJ 08854, USA

ABSTRACT

The domain of machine learning and deep learning is constantly evolving, and in the recent past, privacy concerns have gained significant attention. This project dives into differential privacy and its integration into machine learning, with the primary focus being the Private Aggregation of Teacher Ensembles (PATE) framework. The privacy guarantees that an algorithm can provide are measured by the Differential Privacy framework. The machine learning algorithms that can be used to train models on private data can be designed by utilizing the differential privacy framework. PATE, at the heart of this project, is a very powerful tool in machine learning that integrates the preservation of privacy. The project explores the implementation and the theory behind the implementation of PATE for training models on sensitive data while guarding individual privacy. The project explores the core principles of differential privacy, the aggregation of teacher models, and the training of a student model for the rigorous preservation of privacy. The construction and training of the machine learning model aim to protect privacy while striking a balance between accuracy and utility, the two fundamental aspects of any model. The results derived from the project aim to provide insights into the domain of privacy-preserving machine learning.

1 INTRODUCTION

The need for private machine learning algorithms stems from the intrinsic behavior of traditional machine learning models. The traditional models learn from the data they are trained on by adjusting their parameters to encapsulate the relationships in the data. In an ideal situation, the model's parameters capture the broad patterns in the data, such as "smokers are more prone to heart disease," rather than specific data tied to specific training examples, such as "Joe Doe has heart disease." But, traditional machine learning algorithms fail to neglect these specifics naturally. When traditional machine learning algorithms are employed to solve such a critical task (making a cancer diagnosis model for hospitals worldwide), the traditional model might inadvertently reveal information about the patients in the training set. An adversary might be able to study the model and its results to gain information about Joe Doe. This motivates the study and implementation of private machine learning models.

Many approaches have been proposed to integrate privacy while analyzing or using the data to train machine learning models. These approaches include the anonymization of private information, which involves the removal of the private information or replacing it with randomized values. For instance, details such as phone numbers, names, and zip codes are often anonymized to safeguard the privacy of individuals. However, more than anonymization on its own is needed to protect privacy, and the strength of the privacy provided by it diminishes when adversaries gain additional information about the individuals in the dataset. A very popular example highlighting these vulnerabilities was shown in the paper titled Robust De-anonymization of Large Sparse Datasets by Narayanan et al. Their paper uses the Netflix Prize dataset that contains anonymous movie ratings of 500,00 subscribers of Netflix to demonstrate how an adversary that has very little knowledge about an individual subscriber can easily identify that subscriber in the dataset. They used the Internet Movie Database (IMDb) as the background information, where the users had publicly shared their movie ratings. The two sources of information were used by the researchers acting as adversaries to demonstrate how personal information can be revealed from data that was deemed to be private.

Another prominent example of simple anonymization failing to protect from an adversarial attack is the re-identification of Governor William Weld's medical information from the Cambridge Voter database. This showed how a particular dataset that was believed to be private because of anonymization failed to protect the individual data of the governor. A related concern is that adversaries are only getting better and more technologically advanced, and systems to protect the data from adversarial attacks similar to the ones above must be developed.

In the paper titled "Calibrating Noise to Sensitivity in Private Data Analysis" by Dwork et al., differential privacy was introduced as a framework for evaluating the guarantees provided by a mechanism designed to protect privacy. Along with this, the paper also addresses the limitations of the previous privacy-preserving mechanisms. The primary concept involves the introduction of randomness into the operation of the mechanism to ensure privacy. The reasoning behind introducing randomness in a learning algorithm is to add a nebulous aspect to determine which part of the model defined by the learned parameters came from the actual data and which part came from the randomness. Without the presence of randomness, it would be able to determine the features that the algorithm chooses when trained using a specific dataset. However, when randomness is integrated, the problem changes to determining the probability that the algorithm will choose particular features from the possible set of features to train using the specific dataset.

1.1 Differential Privacy

According to Dwork, Differential Privacy describes a promise, made by a data holder or curator, to a data subject(owner), and the promise is like this: ”You will not be affected adversely or otherwise, by allowing your data to be used in any study or analysis, no matter what other studies, datasets or information sources are available.”

For each individual that contributes to the data, differential privacy guarantees that the output of an analysis that is deemed to be differentially private will be almost similar to the one that is not differentially private, whether an individual contributes to it or not.

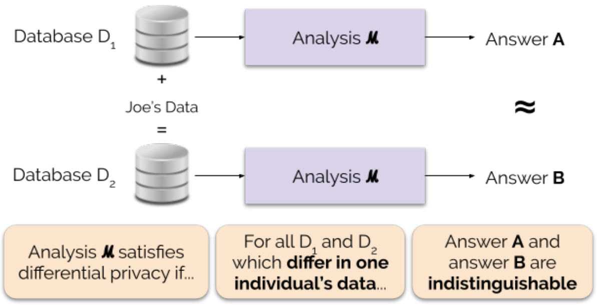

Database D_1
Analysis $\mathcal{M}$
Answer **A**
+
Joe's Data
=
≈
Database D_2
Analysis $\mathcal{M}$
Answer **B**
Analysis $\mathcal{M}$ satisfies differential privacy if...
For all D_1 and D_2 which **differ in one individual's data**...
Answer **A** and answer **B** are **indistinguishable**

Figure 1: Informal definition of differential privacy.

Figure 1 illustrates this principle. Differential privacy states that the two answers should be indistinguishable. This implies that whoever sees the output won't be able to tell whether or not Joe's data was used, or what Joe's data contained.

In the context of differential privacy, ε (epsilon) is a crucial privacy parameter. It quantifies the level of privacy protection provided by a system or algorithm. The ε parameter, often referred to as the ”privacy loss” or ”privacy budget,” has to be tuned, and plays a central role in controlling the trade-off between privacy and utility in data analysis and machine learning applications. A low value for ε indicates higher protection.

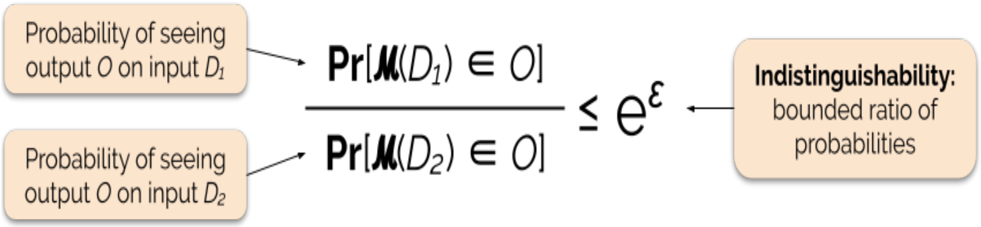

Probability of seeing output O on input D_1
Probability of seeing output O on input D_2
$\frac{\Pr[\mathcal{M}(D_1) \in O]}{\Pr[\mathcal{M}(D_2) \in O]} \leq e^{\varepsilon}$
Indistinguishability: bounded ratio of probabilities

Figure 2: Formal definition of differential privacy.

1.2 Types of Adversarial Attacks

Before diving into PATE, these are a few types of attacks that have been seen in the literature available on security,

- 1. Training data extraction attacks or model inversion attacks:
  Consider a scenario where a classifier has been trained on images of different people's faces, aiming to determine the person depicted in the image. When an image of a face is used as an input into this classifier, it provides a classification result. Frederikson et al. demonstrated that if an adversary has access to the output probabilities of said classifier, they can reconstruct images that have a very close resemblance with the data used to train the model. The reconstructed images were able to capture the average features and characteristics associated with each class, which correspond to a different individual.

- 2. Membership attacks:
  Membership attacks represent the second type of attack, as introduced by Shokri et al. In this case, the adversary's objective differs from the reconstruction of training points based on the classifier's output. The aim of membership attacks is to determine whether a particular input was a part of the data the model was trained on. The adversary can determine this if they can access the classifier's probabilities. For instance, given an image of a person, the goal is to ascertain whether this person's data was included in the training dataset of a particular machine learning model.

1.3 Types of Threat Models

PATE offers a defense strategy against formidable adversaries, primarily by addressing two distinct threat models, viz., black-box and white-box adversaries:

- 1. Model Querying (Black-Box Adversary):
  In this scenario, the adversary has the ability to query the trained model but cannot access the model's internal workings, architecture, or parameters. Their actions are confined to input submission to your black-box model and observing the resulting predictions. These are commonly known as model querying attacks or black-box attacks. The two kinds of attacks mentioned previously are examples of black-box or model querying attacks.

- 2. Model Inspection (White-Box Adversary):
  This attack has more potential for damage because it grants the adversary access to the model and its parameters. The work of Zhang et al., particularly the concept of "Understanding Deep Learning requires re-thinking generalization," suggests that machine learning models may have the ability to memorize particular facets of the training data, or at the very least, they possess the potential to do so. Consequently, robust defenses must be developed against adversaries who have access to these model parameters and can scrutinize them.

2 Working of PATE

With an understanding of the necessity of PATE, differential privacy, and the types of adversarial attacks and threat models, the following section delves into the intuition behind PATE and how it works.

2.1 Intuition behind PATE

Introduced in the paper titled "Semi-Supervised Knowledge Transfer for Deep Learning from Private Training Data" by Papernot et al., the PATE approach for the implementation of differential privacy in machine learning is based on the concept: in the case when two distinct classifiers that have been trained using entirely separate datasets with no overlapping examples, attain unity while classifying a new input, that decision does not disclose any information about any specific training example. It denotes that the decision could have been made with or without any individual training example because the model trained with that example and the model trained without it arrived at the same conclusion.

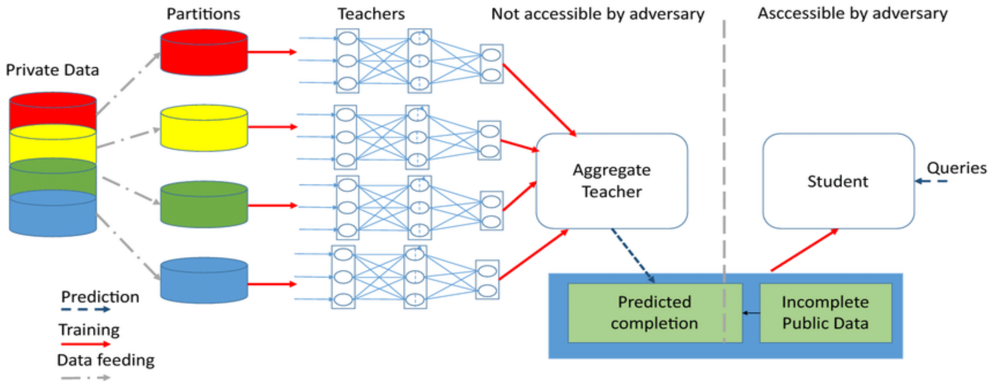


Figure 3: Architecture of PATE.

Now, for the case where two models are trained on separate datasets. When they reach a consensus on a particular input, it appears plausible to make their decision public. However, when they disagree, the situation becomes less clear. It's not feasible to release the individual class predictions from each model independently, as doing so may inadvertently expose private information present in their respective training datasets. For example, assume Joe Doe contributed to the training data of only one of the two models. If that model predicts that a patient with a record strikingly similar to Joe's has cancer. In contrast, the other model predicts the opposite, this disclosure would reveal private information about Joe. This simple example highlights the necessity of introducing randomness into an algorithm to ensure that it offers meaningful privacy assurances.

3 Model Architecture and Implementation

The PATE methodology comprises four essential steps: data partitioning, teacher training, noisy aggregation, and student training. Initially, the training data is partitioned among multiple teachers to avoid centralized access. Each teacher independently trains on their partitioned data. Subsequently, the individual teacher outputs are aggregated through a voting mechanism, introducing privacy-preserving uncertainty. The final step involves training a student model using the aggregated outputs, ensuring privacy and accuracy.

3.1 Data

The dataset being used for the analysis and implementation of PATE is the MNIST dataset, a comprehensive collection of handwritten digits. This dataset comprises 60,000 training images and 10,000 testing images, with each image representing a single digit ranging from 0 to 9, each sized at 28x28 pixels. The 60,000 training images were used to train the teacher models by dividing the them into non-overlapping subsets. The remaining 10,000 images from the testing set were further divided into sets of 9,000 and 1,000 images. The 9,000 images were used to train the student model by querying the ensemble of teacher models to generate labels for the data. The remaining 1,000 images were then used to evaluate the student model.

3.2 Data Partitioning

In the data partitioning phase of PATE, the MNIST dataset serves as the sensitive dataset for training an ensemble of teacher models. The primary objective is to ensure privacy preservation while harnessing the collective knowledge of multiple teachers. The dataset is initially divided into n distinct subsets, where `n` corresponds to the number of teacher models (denoted as `n_teachers`) that will undergo individualized training. This strategic partitioning is vital to prevent centralized access to the entire dataset, promoting a privacy-conscious approach. In the code, the `make_data_loaders` function is pivotal in implementing this partitioning strategy. It generates `n` teacher loaders, each representing a unique subset of the MNIST training data. The concept of non-overlapping subsets is facilitated through the use of the Subset class, ensuring that each teacher loader operates on an independent and non-redundant portion of the data.

The transformation pipeline, defined by the `data_transformer` variable, plays a crucial role in preparing the data for model training. Each data point within the subsets undergoes a transformation process, converting it into a tensor and normalizing its values. This standardized preprocessing ensures consistency and facilitates effective model learning across the ensemble. By employing this distributed and privacy-conscious approach to data partitioning, the PATE methodology establishes a robust foundation for training teacher models. Each teacher contributes unique insights gleaned from their individualized subset, collectively forming a diverse ensemble. This diversity, coupled with privacy-preserving practices, sets the stage for effective and secure knowledge aggregation in subsequent phases of the PATE framework.

3.3 Teacher Training

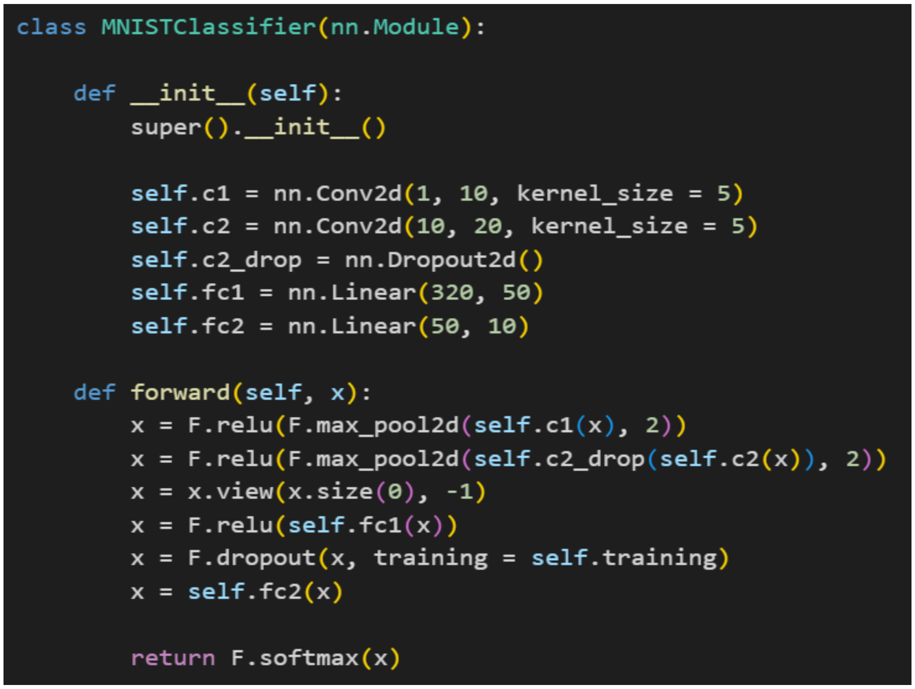
In this phase of PATE, each teacher model is independently trained on its assigned subset of the partitioned data. This training process involves the application of a Convolutional Neural Network (CNN) to enhance the predictive capabilities of individual teachers. The `MNISTClassifier` class defines the architecture of each teacher model, employing a CNN with two convolutional layers, a dropout layer, and two fully connected layers. The training is performed using the negative log-likelihood loss criterion and the Adam optimizer.

The trainer function facilitates the training process by iterating over the specified number of epochs. During each epoch, the model is trained on the provided training loader, and the model parameters are updated using backpropagation and optimization. The `train_models` function initializes a list of teacher models and iterates over the number of teachers, training each model independently on its designated subset of the MNIST dataset. The resulting trained models contribute diverse perspectives to the ensemble, fostering robust learning in the subsequent aggregation step.

3.4 Noisy Aggregation

In this, the `noisy_aggregation` function plays a pivotal role in aggregating the predictions of individual teacher models while introducing privacy-preserving noise. The function iterates over each teacher model in the ensemble, using the predict function to obtain predictions on the provided dataloader (`student_train_loader` in this case). The predictions are stored in a matrix (`pred`) where each row corresponds to the predictions of a specific teacher for the input data points. Laplace noise is introduced to the vote counts for each class within the predictions. This is achieved by adding Laplace-distributed random values to the raw counts. The magnitude of the noise is controlled by the privacy parameter epsilon (ε). A higher epsilon results in more substantial noise injection, enhancing the privacy guarantee but potentially impacting the accuracy of the predictions.

The noisy aggregated predictions are then used to determine the final labels for each input. For each column (`input`), the class with the highest vote count after noise injection is selected as the final label. This step introduces a level of randomness and uncertainty, preventing the ensemble's



```
class MNISTClassifier(nn.Module):

    def __init__(self):
        super().__init__()

        self.c1 = nn.Conv2d(1, 10, kernel_size = 5)
        self.c2 = nn.Conv2d(10, 20, kernel_size = 5)
        self.c2_drop = nn.Dropout2d()
        self.fc1 = nn.Linear(320, 50)
        self.fc2 = nn.Linear(50, 10)

    def forward(self, x):
        x = F.relu(F.max_pool2d(self.c1(x), 2))
        x = F.relu(F.max_pool2d(self.c2_drop(self.c2(x)), 2))
        x = x.view(x.size(0), -1)
        x = F.relu(self.fc1(x))
        x = F.dropout(x, training = self.training)
        x = self.fc2(x)

        return F.softmax(x)
```

Figure 4: CNN Architecture.

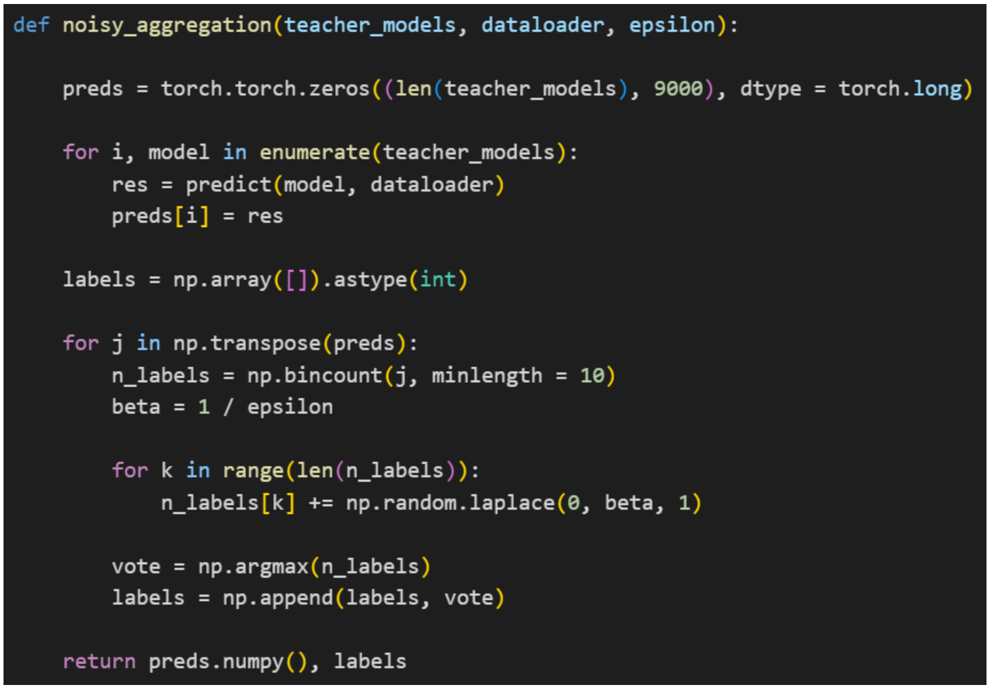
decision from being overly influenced by a single teacher's prediction. The final labels, after the noisy aggregation, are stored in the labels array. This noisy aggregation mechanism is designed to mitigate the risk of privacy breaches by injecting controlled noise into the predictions, thereby ensuring that the aggregated output does not reveal sensitive information about individual teacher models' predictions. The trade-off lies in finding an appropriate epsilon value that balances privacy requirements with the desired level of accuracy for the task at hand. The noisy aggregation strategy aligns with differential privacy principles, offering a robust foundation for the subsequent student training phase while safeguarding the confidentiality of individual teacher predictions.

3.5 Student Training

In this phase of PATE, privacy is meticulously preserved through a strategic training approach that addresses the potential privacy risks associated with individual teacher models. The primary challenge is that teacher models trained independently without explicit privacy considerations may inadvertently retain details of the training data. This susceptibility could lead to privacy breaches when adversaries access the model parameters.

To counteract this risk, the PATE framework introduces a student model as a dedicated entity for training. Unlike the teacher models, the student is trained using a fixed number of labels provided by the ensemble of teacher models. This fixed-label training paradigm is privacy-conscious, ensuring that the student learns from an aggregated set of predictions rather than directly from the raw training data. By doing so, the PATE methodology strikes a delicate balance between maintaining privacy and achieving accurate model predictions.

The fixed number of labels presented to the student serves as a curated and privacy-enhanced source of information, mitigating the potential influence of noise or sensitive details in the complete teacher predictions. This approach aligns with privacy-preserving principles by allowing the student model to generalize effectively from the collective knowledge captured by the ensemble of teacher models. Therefore, this phase contributes to the student model's robustness and exemplifies a privacy-conscious strategy, offering a comprehensive solution for safeguarding sensitive information in machine learning applications.



```
def noisy_aggregation(teacher_models, dataloader, epsilon):

    preds = torch.torch.zeros((len(teacher_models), 9000), dtype = torch.long)

    for i, model in enumerate(teacher_models):
        res = predict(model, dataloader)
        preds[i] = res

    labels = np.array([]).astype(int)

    for j in np.transpose(preds):
        n_labels = np.bincount(j, minlength = 10)
        beta = 1 / epsilon

        for k in range(len(n_labels)):
            n_labels[k] += np.random.laplace(0, beta, 1)

        vote = np.argmax(n_labels)
        labels = np.append(labels, vote)

    return preds.numpy(), labels
```

Figure 5: Noisy Aggregation from Teacher models.

4 Results and Conclusion

For the evaluation of the PATE mechanism and its privacy-preserving capabilities, multiple values for the number of teachers were tested, while keeping the value of the Laplace parameter for the injection of noise constant. The privacy loss for the different values of number of teachers is determined using the data dependent epsilon value. In the context of privacy-preserving machine learning, especially in frameworks like PATE (Private Aggregation of Teacher Ensembles), ”data-dependent epsilon” is a metric used to quantify the privacy loss associated with the model’s predictions. The term ”epsilon” in differential privacy refers to the level of privacy protection, and it’s a measure of how much an adversary can learn about an individual in the dataset by observing the model’s output. The data-dependent epsilon is calculated by measuring the maximum absolute difference between the aggregated labels obtained without any privacy-preserving noise and the labels after adding Laplace noise. It assesses how much the privacy of the model has been affected when considering the specific dataset it was trained on.

Five ensembles of teachers with varying number of teachers $n \in \{10, 25, 50, 100, 200\}$ were tested. The laplace parameter was kept constant at 5 for all the ensembles. Both the accuracy of the student model and the data dependent epsilon were considered to determine the most appropriate value for the number of teachers in the ensemble.

Number of Teachers	Accuracy	Data Dependent Epsilon
10	94.20%	9
25	95.70%	9
50	96.20%	8
100	94.30%	7
200	91.60%	7

Table 1: Experiment Results.

The experiments strongly suggest that, for the given Laplace parameter and experimental setup, an ensemble of 100 teachers strikes an effective balance between privacy preservation and model accuracy. It provides a level of privacy, as indicated by lower data-dependent epsilon values, while maintaining a satisfactory level of accuracy in the student model's predictions. Striking this balance is crucial in privacy-preserving machine learning, where heightened privacy measures must coexist with the practical utility of the model. It underscores the nuanced nature of privacy considerations, emphasizing the need to tailor privacy parameters to the specific use case, dataset characteristics, and task requirements. In essence, the ensemble size, as demonstrated in the experiments, serves as a critical parameter in achieving a harmonious equilibrium between robust privacy protection and accurate model predictions.

References

Papernot, Nicolas, Martín Abadi, Úlfar Erlingsson, Ian Goodfellow, and Kunal Talwar. ”Semi-supervised Knowledge Transfer for Deep Learning from Private Training Data.” ArXiv, (2016). Accessed December 2, 2023. /abs/1610.05755.

Narayanan, Arvind, and Vitaly Shmatikov. ”How To Break Anonymity of the Netflix Prize Dataset.” ArXiv, (2006). Accessed December 2, 2023. /abs/cs/0610105.

Dwork, Cynthia, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. "Calibrating Noise to Sensitivity in Private Data Analysis." Theory of Cryptography, 265–84. https://doi.org/10.1007/11681878_14.

Papernot, Nicolas, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Úlfar Erlingsson. ”Scalable Private Learning with PATE.” ArXiv, (2018). Accessed December 2, 2023. /abs/1802.08908.

Wang, Yu, Borja Balle, and Shiva Kasiviswanathan. ”Subsampled Rényi Differential Privacy and Analytical Moments Accountant.” ArXiv, (2018). Accessed December 2, 2023. /abs/1808.00087.

"Build PATE Differential Privacy in Pytorch." 2020. OpenMined Blog. September 1, 2020. https://blog.openmined.org/build-pate-differential-privacy-in-pytorch/.

"Privacy and Machine Learning: Two Unexpected Allies?" 2018. Cleverhans-Blog. April 29, 2018. https://www.cleverhans.io/privacy/2018/04/29/privacy-and-machine-learning.html.