Understanding Privacy Threats in Machine Learning as a Service (MLaaS)

Choeun Song, Olga Paraskevi Tragaziki, Orestis Gardikis,

Pratima Parajuli, Vigneshwar Anandhamurugan

Email: csong23, otraga23, ogardi23, pparaj23, vanand23 (@student.aau.dk)

January 17, 2025

AALBORG UNIVERSITY

DENMARK

AALBORG UNIVERSITY
STUDENT REPORT

Department of Cyber Security
Aalborg University
http://www.aau.dk

Title:

Understanding Privacy Threats in Machine Learning as a Service (MLaaS)

Theme:

Semester 3 Group Project

Project Period:

September 2024 - December 2024

Project Group:

903

Participant(s):

Choeun Song
Olga Paraskevi Tragaziki
Orestis Gardikis
Pratima Parajuli
Vigneshwar Anandhamuragan

Supervisor(s):

Main supervisor - Qiongxiu Li

Page Numbers: a

Date of Completion:

January 17, 2025

Abstract:

Given the rise of Machine Learning algorithms, the emphasis on vulnerabilities associated with these machine learning algorithms are under scrutiny. The report primarily deals with privacy leaks wherein, hidden attributes about the underlying, possible sensitive, training records are inferred by a malicious actor. Membership inference attacks were simulated on multiple Machine Learning algorithms, trained on CIFAR-10 and CIFAR-100 datasets, to better understand what are the underlying causes that potentially enable and increase the impact and susceptibility of such cyberattacks. The report employs several techniques including the use of Receiver Operating Charateristic curves, distribution of training and testing datasets of ML models across a certain confidence distribution. Additionally, the report examines and proposes techniques, such as the use of early stopping mechanisms and a combination of specific activation functions and regularization techniques, to reduce the susceptibility of these ML models to privacy leaks.

Keywords: Membership Inference Attack, machine learning, privacy, blackbox machine learning

Contents

1 Introduction

Machine learning (ML) is a branch of and computer science that focuses on using data and algorithms to enable Artificial Intelligence(AI) to imitate the way that humans learn, gradually improving its accuracy [1]. These models were brought about to make more informed, scalable and flexible decisions as opposed to the hard coded, non scalable and rigid decisions offered by the traditional explicit programming solutions. Artificial Intelligence refers to understanding and analyzing problems and offer appropriate solutions to these problems using ML models. The boom of AI, places ML models under spotlight for both benevolent and malicious actors. While the benevolent actors employ AI for creative reasons, malicious actors either try to use ML to increase sophistication of cyber attacks or break existing ML models. The latter requires cybersecurity analysts to analyze these cyber attacks to better equip themselves to make the ML models more secure.

ML models repeatedly undergo a training phase and a testing phase, to better learn to predict or classify data. In the training phase, the ML models use datasets to understand how they are supposed to provide solutions by associating different features of the datasets and their corresponding outcomes. These correlations are tested to discard correlations that bear no impact towards predicting the outcomes and hone existing correlations thereby increasing the various performance characteristics associated with the model. ML used in sensitive industries, such as the healthcare industry, are more likely to be trained on sensitive data, such as a population's medical records. This raises concern over the ability of a ML model to retain the privacy of the people whose medical records have been used to train the corresponding ML model.

This report delves into a common privacy attack on the ML models known as the Membership Inference attack. The membership inference attack refers to the ability to determine if a given data record has been used to train a certain ML model [2]. Malicious threat actors distinguish samples as training and non-training datasets based on how a certain ML model reacts to the sample data records. This report simulates membership inference attacks to gain better understanding into which models are vulnerable to these attacks and why, the success rate of attributing a given a data record as training or non-training datasets and a comparison of performance on a set of ML models each trained differently.

The report explores different methods of performing a membership inference attack. It also tries to establish a relationship between the performance metrics of a ML model including the accuracy, precision and recall of a ML model and the efficiency of carrying out these attacks on the corresponding ML model. The report also attempted to identify a correlation between the receiver operating characteristic of a ML model and ability to ascertain a given sample as belonging to a training dataset of the corresponding ML model.

2 Background Research

This section provides a fundamental understanding of privacy concerns in machine learning, how a model can expose sensitive information. Focusing on membership inference attacks (MIAs), this section includes the definition and types of MIAs.

2.1 Machine Learning

Traditional problem-solving with computers involves creating a program $\mathcal{M}$ that, given some input x, produces an output y as the solution to a problem. For deterministic programs, this can be represented as a function $\mathcal{M} : X \to Y$, mapping an input set X to an output set Y. However, as problem complexity grows, constructing such a program becomes increasingly challenging [3].

Machine learning (ML), a significant branch of artificial intelligence, addresses the challenge of approximating complex programs by creating systems that learn and improve from experience without explicit programming. By leveraging statistical models, probability theory, and optimization techniques, ML algorithms process vast datasets, identify patterns, and derive actionable insights. A key strength of ML is its ability to generalize, applying knowledge from training data to unseen data, making it versatile for solving diverse problems. For instance, given a dataset Z capturing relationships in the input space X, ML constructs a parametric model $\mathcal{M}_\theta : X \to Y$, where θ represents the model's parameters [3]. In a fraud detection example, the dataset Z might include transaction histories, while the output $Y = \{0, 1\}$ indicates whether a transaction is fraudulent (1) or not (0).

At the core of ML processes lies an algorithm $\mathcal{A}$, which trains the model on Z or extracted features from Z, producing the final model $\mathcal{M}_\theta$ [3]. Algorithms like Random Forest, Neural Networks, and Logistic Regression are chosen based on problem specifics, dataset size, and computational constraints. Training on selected features, such as high-risk transaction patterns, can enhance efficiency by excluding irrelevant data, reducing processing time while improving accuracy.

There are three main approaches to machine learning: supervised, unsupervised, and reinforcement learning. Supervised learning involves training algorithms on labeled data, where each input is paired with the corresponding desired output, enabling the system to learn relationships between them. This approach is widely used in tasks like image classification or predictive modeling, where the goal is to map inputs to outputs accurately. Unsupervised learning, on the other hand, deals with unlabeled data, focusing on uncovering hidden patterns, structures, or correlations. Techniques like clustering and association analysis fall under this category,

offering valuable insights when explicit output labels are unavailable.

Reinforcement learning takes a different approach, emphasizing interaction with the environment. The system learns to make decisions through a trial-and-error process, guided by rewards or penalties based on its actions. Inspired by behavioral psychology, this method helps systems independently discover optimal strategies over time, often applied in areas like robotics, gaming, and autonomous systems.

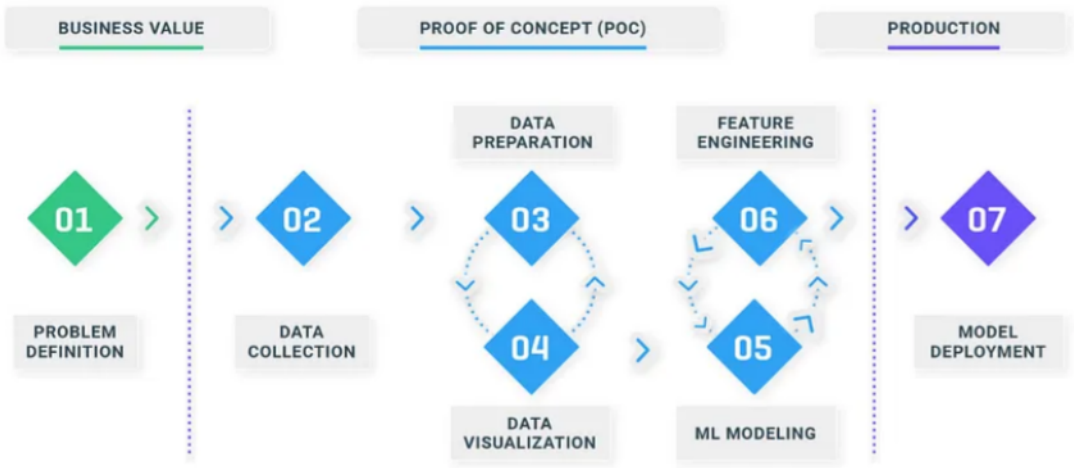

Figure 1: The 7 Stages of Machine Learning [4]

The development of machine learning (ML) solutions involves a structured framework that progresses through distinct phases. These phases ensure the effective application of ML to real-world problems while emphasizing both technical and business considerations. The framework can be broadly divided into three phases, as shown in Figure 1: Business Value, where the focus is on defining the problem and its impact, Proof of Concept (POC) [4], which tests feasibility and value through iterative experimentation, and Production, which involves deploying and scaling the model in a business environment.

Within these phases, seven key stages provide a comprehensive roadmap for building, deploying, and refining ML models [4].

1. Problem Definition focuses on understanding the context, identifying the challenges, and defining the problem clearly and in alignment with measurable business goals.
2. Data Collection involves gathering the data required to address the problem. Sources may include internal records, public datasets, or web scraping, ensuring the data is relevant, accessible, and compliant with legal standards.
3. Data Preparation involves cleaning, validating, and formatting the data. This step ensures the quality and

reliability of the data for subsequent modeling.

4. Data Visualization techniques are applied to explore large datasets and identify trends or anomalies. Graphical tools such as heat maps, histograms, and bar charts help uncover insights that guide further analysis and model development.
5. ML Modeling. In this stage, a machine learning algorithm is trained to make predictions or decisions based on the data. Common tasks include regression (predicting numbers), classification (sorting data into categories), clustering (grouping similar data), or forecasting (predicting future trends). The algorithm is trained using a dataset, adjusting its parameters to minimize errors. This process often uses a loss function to measure how far the model's predictions deviate from the ideal outcome. Optimizing the loss function, typically with methods like Stochastic Gradient Descent (SGD), helps improve the model's accuracy and prepares it for validation. The choice of algorithm depends on the data type and the problem's goals.
6. Feature Engineering involves creating meaningful variables that capture relationships and patterns within the data. The success of an ML model depends significantly on the quality of the features used for training. This step often requires domain expertise and a combination of statistical and heuristic methods.
7. Model Deployment involves integrating the trained model into production systems. This ensures the model's predictions or decisions can be utilized in real-world applications.

2.1.1 Application of Linear Algebra in Neural Networks

Neural networks are computational models inspired by the biological brain, designed to approximate functions by learning from data [5]. They consist of interconnected layers of artificial neurons, and data is typically represented as matrices or vectors.

Vectors

Vectors are mathematical objects characterized by both magnitude and direction. They are represented algebraically as ordered tuples of numbers or geometrically as arrows in space. A vector $\mathbf{v}$ can be expressed as:

$$\mathbf{v} = \begin{bmatrix} 2 \\ -1 \\ 4 \end{bmatrix},$$

where each component corresponds to a specific dimension in the vector space.

Matrices

Matrices are rectangular arrays of numbers organized into rows and columns. They are fundamental in encoding datasets, representing linear transformations, and performing matrix operations. For instance, a 3×3 matrix $\mathbf{A}$

is expressed as:

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}.$$

Matrices allow efficient representation and manipulation of multidimensional data.

Scalars

Scalars are single numerical values that lack direction and are often used to scale vectors or matrices. For instance, given a scalar $k = 3$ and a vector $\mathbf{v} = \begin{bmatrix} 2 \\ -1 \\ 4 \end{bmatrix}$, scalar multiplication is defined as:

$$k \cdot \mathbf{v} = 3 \cdot \begin{bmatrix} 2 \\ -1 \\ 4 \end{bmatrix} = \begin{bmatrix} 6 \\ -3 \\ 12 \end{bmatrix}.$$

Addition and Subtraction

Addition and subtraction of vectors or matrices are performed element-wise. Given two vectors:

$$\mathbf{u} = \begin{bmatrix} 2 \\ -1 \\ 4 \end{bmatrix}, \quad \mathbf{v} = \begin{bmatrix} 3 \\ 0 \\ -2 \end{bmatrix},$$

vector addition is computed as:

$$\mathbf{u} + \mathbf{v} = \begin{bmatrix} 2+3 \\ -1+0 \\ 4+(-2) \end{bmatrix} = \begin{bmatrix} 5 \\ -1 \\ 2 \end{bmatrix}.$$

Vector subtraction is similarly performed:

$$\mathbf{u} - \mathbf{v} = \begin{bmatrix} 2-3 \\ -1-0 \\ 4-(-2) \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \\ 6 \end{bmatrix}.$$

Scalar Multiplication

Scalar multiplication involves scaling each component of a vector or matrix by a scalar value. For example,

given a scalar $k = 3$ and a vector $\mathbf{v} = \begin{bmatrix} 2 \\ -1 \\ 4 \end{bmatrix}$, the result is:

$$k \cdot \mathbf{v} = \begin{bmatrix} 6 \\ -3 \\ 12 \end{bmatrix}.$$

Dot Product

The dot product of two vectors quantifies their similarity and is computed as the sum of the products of their corresponding components. For vectors $\mathbf{u}$ and $\mathbf{v}$, the dot product is defined as:

$$\mathbf{u} \cdot \mathbf{v} = u_1 v_1 + u_2 v_2 + \cdots + u_n v_n.$$

For example, given:

$$\mathbf{u} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \quad \mathbf{v} = \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix},$$

their dot product is:

$$\mathbf{u} \cdot \mathbf{v} = (1 \cdot 4) + (2 \cdot 5) + (3 \cdot 6) = 32.$$

Cross Product

The cross product, defined for three-dimensional vectors, results in a vector orthogonal to the plane formed by the two input vectors. The cross product of $\mathbf{u}$ and $\mathbf{v}$ is computed as:

$$\mathbf{u} \times \mathbf{v} = \begin{bmatrix} u_2 v_3 - u_3 v_2 \\ u_3 v_1 - u_1 v_3 \\ u_1 v_2 - u_2 v_1 \end{bmatrix}.$$

2.1.2 Properties of Linear Transformations

A transformation T is linear if it satisfies:

- **Additivity:**

$$T(\mathbf{u} + \mathbf{v}) = T(\mathbf{u}) + T(\mathbf{v}),$$

- **Homogeneity:**

$$T(k\mathbf{v}) = kT(\mathbf{v}),$$

for all vectors $\mathbf{u}$, $\mathbf{v}$, and scalar k.

Linear transformations are widely used in machine learning to preprocess and manipulate data. They facilitate translation to center data, scaling to normalize feature ranges, and rotation to analyze spatial relationships.

2.1.3 Representation of Data and Parameters

A neural network processes data through interconnected layers, where each layer transforms inputs using weights and biases. Its architecture typically consists of the following components [5]:

- **Input Layer:** This layer receives raw data as feature vectors $\mathbf{x} \in \mathbb{R}^D$, where D is the number of features. For instance, in an image classification task, D corresponds to the total number of pixels in the image.
- **Hidden Layers:** These intermediate layers transform the input data using weights and biases:

$$\mathbf{z}^{(l)} = \mathbf{W}^{(l)}\mathbf{a}^{(l-1)} + \mathbf{b}^{(l)},$$

  where:
  - $\mathbf{W}^{(l)}$ is the weight matrix for the l-th layer,
  - $\mathbf{b}^{(l)}$ is the bias vector, and
  - $\mathbf{a}^{(l-1)}$ is the activation vector from the previous layer.

  A non-linear activation function $\sigma(\cdot)$ is applied element-wise:

$$\mathbf{a}^{(l)} = \sigma(\mathbf{z}^{(l)}).$$

- **Output Layer:** This layer generates predictions based on the task. For classification problems, the output is often a probability distribution computed using the softmax activation function:

$$\hat{\mathbf{y}} = \text{softmax}(\mathbf{z}) = \frac{e^{\mathbf{z}_i}}{\sum_j e^{\mathbf{z}_j}},$$

  where $\hat{\mathbf{y}}$ represents predicted probabilities for each class.

2.1.4 Forward Propagation

Forward propagation computes predictions by passing data through the network. Each layer performs the following steps:

1. **Linear Transformation:**

$$\mathbf{z} = \mathbf{W}\mathbf{x} + \mathbf{b},$$

   where $\mathbf{W}$, $\mathbf{x}$, and $\mathbf{b}$ represent the weights, inputs, and biases, respectively.

2. **Non-Linear Activation:**

$$\mathbf{a} = \sigma(\mathbf{z}),$$

where $\sigma(\cdot)$ introduces non-linearity to model complex patterns.

3. **Batch Processing:** For multiple input instances in a dataset $\mathbf{X} \in \mathbb{R}^{N \times D}$, where N is the batch size, the transformations are applied as:

$$\mathbf{Z} = \mathbf{X}\mathbf{W}^T + \mathbf{b},$$

producing pre-activation outputs $\mathbf{Z}$ for all data instances.

2.1.5 Backpropagation and Gradients

Gradient

The gradient represents the direction and rate of the fastest increase for a given function. In the context of machine learning, it measures how much the loss function L changes with respect to the model parameters $\mathbf{W}$. Mathematically, the gradient is expressed as:

$$\frac{\partial L}{\partial \mathbf{W}},$$

where L is the loss function, which quantifies the error between predicted outputs ($\hat{y}$) and true outputs (y), and $\mathbf{W}$ is the parameter set. The gradient guides the optimization algorithm to adjust the parameters to minimize L.

Backpropagation

Backpropagation is the algorithm used to compute gradients efficiently in neural networks. It leverages the *chain rule of differentiation* [5] to propagate errors backward through the network. The process involves:

1. **Forward Propagation:** Compute predictions ($\hat{y}$) by passing input $\mathbf{x}$ through the network layers.
2. **Compute Loss:** Evaluate the loss function $L(y, \hat{y})$.
3. **Backward Propagation:**
   - Calculate gradients layer by layer starting from the output.
   - For a layer l, compute:

$$\frac{\partial L}{\partial \mathbf{W}^{(l)}} = \delta^{(l)} \cdot \mathbf{z}^{(l-1)},$$

     where:
     - $\delta^{(l)}$ is the error term for layer l,
     - $\mathbf{z}^{(l-1)}$ is the output of the previous layer.
4. **Update Parameters:** Adjust weights and biases using an optimization algorithm like gradient descent:

$$\mathbf{W} \leftarrow \mathbf{W} - \eta \frac{\partial L}{\partial \mathbf{W}},$$

where η is the learning rate.

2.1.6 Performance Metrics of ML models

The predictions of a ML model, correct or incorrect are visualized in the form of a confusion matrix. It displays the True Positives (TP), which represent the total number of positive instances correctly classified as positive and True Negatives (TN), which represent the total of negative instances correctly classified as negative. It also includes False Positives (FP) and False negatives (FN). While the former indicates the number of negative instances classified as positives, the latter indicates the number of positive instances classified as negatives.

The performance of these ML model is depicted by several characteristics such as accuracy, precision, recall and F1 score. Accuracy refers to the proportion of total number of correct predictions to the total number of predictions. Precision refers to how often the model's positive predictions are correct. It is the proportion of true positives to the total number of instances classified as positive. Recall refers to how well the model identifies positive instances. It is the proportion of true positives to the total number of instances which are actually positive. F1 score accounts for both false positives and false negatives that a ML model is likely to predict.

Another important metric used to evaluate the overall performance of a ML model is the Receiver Operating Characteristic curve. It takes into account the True Positive Rate which is the recall value and the False Positive Rate which refers to the proportion of the negative instances falsely classified as positives to the total number of negative instances. The ROC curve is a graphical plot between the two metrics. A curve closer to the diagonal suggests that the ML model incorporates a relaxed criteria for determining an accurate prediction while a curve farther from the diagonal suggests a strict criteria for determining a prediction.

ML models do not necessarily align their predictions to the true outcome of an event. More often, these issues arise because of either underfitting or overfitting of these ML models. Underfitting refers to the scenario, where a ML model is trained too little and therefore has not discovered necessary underlying patterns to provide the appropriate response to the input. It is a result of the ML algorithm being too simple. On the other hand, overfitting occurs when a ML algorithm takes strictly into account, all or most of the features of a dataset [6]. This results in a ML model struggling to predict the correct true output of a given input. In both these cases, the ML model's performances differ widely over the training and testing datasets.

2.1.7 Activation Functions

In the development and operation of machine learning models, particularly neural networks, activation functions play a fundamental role in determining the model's capacity to learn and represent complex relationships. An activation function is applied to the output of a neuron or node, introducing non-linearity into the model.

Without this non-linear transformation, the model would simply reduce to a linear combination of its inputs, severely limiting its representational power and resulting in models that cannot capture intricate patterns in the data.

Common activation functions include the sigmoid, hyperbolic tangent (tanh), ReLU (Rectified Linear Unit), and variants of ReLU such as Leaky ReLU or ELU. Each offers distinct properties: (LIST)

1. **Sigmoid**: Maps inputs to a (0,1) range, traditionally useful in binary classification tasks. However, it can suffer from saturation, causing gradients to vanish, thereby slowing down training.
2. **Tanh**: Similar to the sigmoid function but outputs values in the range (-1,1). This often leads to faster convergence in certain tasks than the sigmoid function, yet it can still face issues with vanishing gradients.
3. **ReLU**:Outputs zero for negative inputs and the raw value for positive inputs. Its simplicity and effectiveness have made it a popular default choice. It mitigates the vanishing gradient problem observed in sigmoid and tanh functions, often resulting in faster training. However, ReLU can lead to inactive neurons, a phenomenon known as the "dying ReLU" problem.
4. **Advanced Variants of ReLU (e.g., Leaky ReLU, ELU)**: Introduce small positive slopes for negative inputs or alternative functional forms to keep neurons active and gradients flowing, thereby improving model robustness and potentially accelerating convergence

Selecting an appropriate activation function is crucial because it directly affects the model's ability to learn non-linear patterns and influences training dynamics such as gradient flow and convergence speed. Moreover, the choice of activation function can impact how well a model generalizes, how susceptible it is to issues like overfitting, and how effectively it can translate learned representations to new, unseen data. Thus, while performance metrics and model selection strategies guide us in determining if a model is accurate and robust, the underlying activation functions help ensure that the model is flexible enough to capture the inherent complexities of real-world datasets.

2.2 Privacy Goals

Machine learning (ML) processes involve various assets requiring privacy protection, including: (1) data contributors' identities, (2) the raw dataset Z, (3) feature datasets X_1 and X_2, (4) the model $\mathcal{M}_\theta$, and (5) the model's inputs. Protecting privacy is essential not only to ensure ethical data usage but also to mitigate potential risks such as identity exposure, misuse of sensitive data, and breaches of confidentiality. Five key privacy aspects in ML processes, which present distinct challenges that must be addressed to safeguard the integrity of ML systems, are:

1. Identity Privacy focuses on the anonymity of data contributors, protecting sensitive personal information like political views or health conditions from model creators, owners, and potential adversaries.

2. Raw dataset privacy aims to protect the original data, which in case of a compromise lead to identity breaches or create new vulnerabilities. For instance, an energy consumption dataset might reveal the absence patterns of residents, increasing risks of targeted crimes.
3. Feature Dataset Privacy is equally vital, as X_1 (training data) and X_2 (validation data) derived from Z could be exploited to infer sensitive information about the raw data.
4. Model Privacy concerns the secrecy of the ML model itself and its parameters in order to maintain competitive advantage and limit unauthorized insights.
5. Input Privacy ensures that sensitive inputs to the model remain confidential, revealing only the outputs. For example, in predictive models for genetic diseases, the privacy of individual medical records must be preserved, even when responses are shared.

2.2.1 Privacy concerns in Machine Learning

As machine learning is using data that can be sensitive, and larger datasets typically result in improved performance and generalization of the model, data sharing remains one of the key challenges in machine learning. Data sharing can be led to severe data leakage, or misuse [7]. For instance, in healthcare, data sharing can facilitate collaborative research and innovation among hospitals, clinics, and researchers. However, it also carries the risk of exposing the patients' confidential medical records or genomic data to unauthorized access or malicious attacks. In general, to provide security of data, confidentiality is key and technologies like encryption are used. However, in machine learning, we need to discern what we can learn from the sheer data (confidentiality) and what can be learnt by accessing statistics about the data (privacy) [8]. An ML model can, in fact, reveal information about an individual whose data was used during training, even if the model contains no explicit references to the person or their data. Therefore, as a higher level insight, privacy in machine learning examines what an adversary could learn about the data used to train a model if they gain some level of access to the model.

2.3 Membership Inference Attacks

Privacy issues can bring attacks against machine learning systems, such as membership inference attacks. Membership inference attacks refer to acquiring the knowledge about whether a certain data record is used for training the model, as training dataset [9]. Disclosing that a specific record was included in the training of a machine learning model strongly indicates a potential leakage of private information related to individual data points within the training set [8]. For example, knowing that a medical record was used to train a machine learning model deployed for diabetes detection can reveal that the person concerned has diabetes. These types of attacks exploit the fact that machine learning models show different behaviors when processing new data compared to the data used for training. Models typically show higher confidence in predictions made on training data, and significantly lower prediction loss compared to that of unseen data. This discrepancy in model

behavior can be used to infer information about the training data set.

2.3.1 Black-box and White-box access

Based on the adversarial knowledge of the target model, which includes training method and architecture of the model, attacks can be categorized into two.

1. **White-box**: In this setting, an attacker has complete access to information necessary to compromise the target machine learning model [10]. This includes the distribution of the training data, details of the training process, the model's architecture, and its learned parameters. In the white-box setting, the attacker can also observe the parameters of the model.
2. **Black-box**: The black-box setting is when the attacker has no knowledge about the parameters of the target model. The information that the attacker can have here is the given input and the corresponding output (query responses). Compared to white-box membership inference attacks, attackers in black-box membership inference attacks have access to only limited information about the target machine learning model. However, successful black-box membership inference attacks can be even more dangerous, as they demonstrate the ability to compromise membership privacy with their minimal knowledge of the model [10].

Our study focuses on the black-box testing and white-box testing, in which the attacker only has limited information about the target model.

2.4 Regulations and Membership Inference Attack

Whether it was the industrial, communication, transportation revolution, in every revolution throughout history, governance and regulations have had a great impact on how these innovations evolve and how it shapes society. At present, we are on the edge of a Machine learning and AI revolution and again regulations will determine how the technology will be absorbed by society to its maximum potential.

137 countries have opted for data privacy and protection regulation [11]. General Data Protection and Privacy Regulation (GDPR), California Consumer Privacy Act (CCPA), China's Personal Information Protection Law primarily (PIPL), Australian Privacy Act 1998 and Singapore's Personal Data Protection Act (PDPA) [12] are the list of some regulations around the world. In this section, we will focus on GDPR that emphasizes data protection and data privacy in the EU region.

2.4.1 General Data Protection and Privacy Regulation

GDPR states data protection as maintaining the confidentiality of data and privacy as facilitating data owners to decide on who can collect and process their personal data and for what reasons [13]. In addition to that, GDPR includes principles on data minimization and purpose limitation meaning organizations should only collect necessary data for the specified purpose. GDPR intends to protect personal data against adversarial use and promote transparency in data processing. Article 35 of the GDPR enforces the Data Protection Impact Assessment (DPIA) in new technologies such as MLaaS that involve high risk to personal data as a part of "protection by design" [14]. For example, if a company x is using MLaaS for chatbot service and using 100 person data to train the model, if there is potential risk to people's privacy, the organization should assess the impact of risk before processing the data.

Membership inference attack, directly violates GDPR article 5 the key principles related to processing of personal data, exposing information of data point that was used to train machine learning model without the consent of data owner [15]. Key principles includes:

- **Lawfulness, Fairness and Transparency:** GDPR requires organizations to process personal data lawfully, fairly and transparently. When membership inference attack revels data point membership, it violates privacy of data owner without the knowledge.
- **Purpose Limitation and Data Minimization:** When collecting personal data, GDPR requires organization to inform data owner in upfront. Data collected should be only use for legitimate, explicit and specified purpose. Data cannot be further processed that were not predefined. MIA allows adversary to further infer their data and use for unintended purpose.
- **Integrity and Confidentiality:** GDPR requires organizations to ensure data security to maintain its confidentiality and integrity. MIA provide unauthorized access to information making data susceptible to unspecified, possible malicious, usage.
- **Accountable:** GDPR requires organizations comply with its principle to ensure data safety. MIA reflect organizations inability to ensure confidentiality, hence violating accountability.

Though, there are clear regulations on data protection and privacy, organizations are required to take privacy measures and the data owner's consent. Personal data continues to be uploaded on cloud platforms for training ML algorithms. Present regulations provides binary option to users, whether to agree on data collection or to stay out completely, therefore a flexible option that preserves privacy should be adopted [16].

2.5 Privacy Leakage Instances

Membership inference attacks have strong presence as theoretical threat in the realm of machine learning privacy. Extensive research has demonstrated the practicality of attack scenarios in real life in across the different field such as education, healthcare, finance. Qayyum et al. [17] suggests the urgent need to address these risks. In recent years, machine learning has been adopted to a wide range of applications such as image classification, speech recognition, object detection [17]. In addition to MLaaS, it is increasingly being common to use third party services to train machine learning/deep learning models, this significantly increases the attack surface. In this section, we will discuss a few examples of privacy leakage instances.

Membership inference on MLaaS Platform: Salem et al. [18] presented a membership inference attack on Google Cloud Prediction API (Google's API platform). In addition to that, this paper studies three different categories of adversaries, having different levels of access to training datasets, shadow models and model structures showcasing feasibility and efficiency of membership inference attacks in real life scenarios. This platform allows users to upload their data and get black box ML models trained by Google where users do not have control over classifiers, parameters and model structure. This attack is blackbox based attack, where adversary queries the API and observes models output (prediction vector or probability). Salem et al. [18] trained the target model and shadow model were both trained in Google API platform, whereas the attack model was trained locally. Experiment consists of two datasets, purchase-100 data set: dataset created from real customer purchases record derived from kaggle and location data set: dataset of location based data point. For the purchase data set, the attack on Google MLaaS achieved 0.90 precision and a 0.89 recall and for the location data, it achieved precision 0.89 and recall 0.86 [18]. Performance accuracy on Google API was slightly better than local setup, this shows membership inference on MLaaS are efficient and effective in real life scenarios.

AI in Healthcare:

Machine learning/ AI models specially in healthcare services are required to be trained on sensitive data, presenting unique privacy risks.Though AI has potential to solve health care industry problems in unprecedented ways. Medical Imaging, precision medicine are some AI powered solutions integrated in healthcare. [19] highlights the privacy risks, challenges in ensuring privacy and privacy preserving techniques while integrating AI in healthcare services.

There have been several real life incidents highlighting privacy concerns in the healthcare industry. [20]Google's Deepmind faced lawsuits for using NHS data, while the project did not expose medical data explicitly but it could possibly be used to reveal medical data. Lawsuit was done to address public concern and it shows how data can be transferred from one entity to another without data owners consent.

In 2020 University of California paid dollar 1.14 million ransom, it breached research data, medical history.

Though these report does not explain details of exposed medical data it can be calculated there must have been significant data lost. [21]

[22] Showed from 2015-2020 , 249.09 millions of people were affected by data breaches. In 2019 alone 41.2 million medical data were revealed or illegally exposed from 505 healthcare data breaches. While these data breaches does not align with membership inference attack, exposed data could be used in multiple avenues creating limitless threats and also showing medical data has always been a prime target for adversaries.

Finance: Integration of ML models in the financial sector has revolutionized vital services such as fraud detection, financial management. Though machine learning models have enormous positive impacts, its shortcomings should not be overlooked. [23] describes how machine learning models are trained on sensitive financial data, customer personal information, trading behaviors which brings significant privacy risks making AI powered financial systems an attractive target for adversaries. Cao [23] discussed how ML models are integrated in sensitive areas and how it can impact natural persons and organizations handling sensitive information and services. Credit scoring, algorithmic trading and fraud detection are some of the sensitive areas where ML models are integrated. For example, ML powered credit scoring trained on clients credit data is vulnerable to MIA. An adversary can query to infer whether a person's financial information was used in the training of credit scoring eventually exposes an individual's sensitive financial information. Fraud detection: MIAs on ML powered fraud detection can reveal private details of customers or sometimes it can reveal fraudulent information that are under investigation. Likewise, MIA on algorithmic trading AI models trained on historical financial data can expose corporate strategies [23].

Generative AI models: Generative AI are the subset of AI that are capable of producing different types of content such as text, source code, images, sound and other forms. Gupta et al. [24] highlights generative AI can be used for both attack and defensive purposes. It highlights different technique used to exploit generative AI some of them are listed below:

Jailbreaking: It is a technique to bypass security restriction placed on generative AI. An adversary can use jailbreaking techniques to answer harmful and unethical prompts against the rule set by developers or organization handling generative AI model.

Prompt injection attacks: It is a technique where an adversary carefully craft prompt to generate response to unethical and harmful queries manipulating AI behaviour.

Character playing (Role-Playing): It is way to ask generative AI to be on a specific role through prompt making it respond to queries exposing sensitive information. For example, ChatGPT might not respond to "How do I hack a public network" but it might respond to prompt "Act as masters in cybersecurity student researching in defensive security, provide me step by step process on how to get access to public network as an admin.". This might reveal unintentionally reveal admin username and password.

3 Related Work

This section presents an overview of studies focus on privacy matters in machine learning.

Shokri et al. simulates a membership inference attack by creating shadow models [2]. The paper proposes a simulation of the attack by creating multiple shadow models to imitate the target model but for whose the ground truth is known i.e the training dataset for these shadow models are known as compared to the target model which is considered as a blackbox environment. Using these shadow models several attack models for each output class of the target model is developed. These attack models have the ability to predict if a data record of a particular label was used for training. Although the paper establishes how overfitting introduces privacy leaks in the ML models, it also states that overfitting alone does not result in privacy leaks.

El Mastari et al. [8] examines the risks to data protection in modern machine learning systems from the perspective of data owners, those who are responsible for managing datasets, models, or both throughout the machine learning life-cycle. It highlights how the origin of threats, the risks to data, and the effectiveness of Privacy-Enhancing Technologies (PETs) are influenced by the data processing phase, the roles of the involved parties, and the architecture in which machine learning systems are deployed.

Liu et al. [9] provides a comprehensive survey of the current state of privacy issues and solutions in machine learning. It explores three key areas of interaction between privacy and machine learning, privacy-preserving machine learning, machine learning applications for privacy protection, and machine learning-driven privacy attacks along with corresponding defense mechanisms. The paper reviews the latest research works in each category, identifies critical challenges, and highlights future research directions based on the analysis of machine learning field.

Hu et al. [10] conducts the first comprehensive survey on membership inference attacks and defenses. Providing the definition of Membership Inference Attack on machine learning models and introducing existing attack methodologies, they present a taxonomy to categorize all the research papers on MIAs. After, they discuss the underlying reasons why MIAs are effective on ML models and introduce the existing defense strategies designed to mitigate MIAs and provide a taxonomy to classify the research on membership inference defenses. The paper explores challenges and potential future research for both MIAs and its defenses.

Salem et al. [25] focuses on membership inference attack, showing that more broad applicable attack scenarios are possible. They evaluate membership privacy threat under three different adversarial setups on eight diverse datasets, ultimately arriving at a model and data independent adversary. Extensive experiments demonstrate

the severe membership privacy threat for machine learning models. Then they propose two defense mechanisms, namely dropout and model stacking, and demonstrate their effectiveness experimentally.

Murakonda et al. [26] focuses on indirect leakage about training data from machine learning models. They present ML Privacy Meter, a tool that can quantify the privacy risk to data from models through state of the art membership inference attack techniques. The tool provides privacy risk scores that help in identifying the data records that are under high risk of being revealed through the model parameters or predictions. It also can estimate the amount of information that can be revealed through the predictions of a model (referred to as Black-box access) and through both the predictions and parameters of a model (referred to as White-box access), so that the tool can be used to assess the potential threats to training data. The tool can guide practitioners in regulatory compliance by helping them analyze, identify, and minimize the threats to data.

Li et al. [27] re-investigates the privacy effect of applying data augmentation and adversarial training to machine learning models via a new perspective, memorization. With that, they revealed that the attacks deployed in previous studies for measuring privacy leakage produce misleading results, which was that the training samples with low privacy risks are more prone to be identified as members compared to the ones with high privacy risks. Through the evaluation, they found out the generalization gap and privacy leakage are shown less correlated than those of the previous results and label smoothing does not always amplify the privacy leakage. Moreover, they also show that improving the adversarial robustness (via adversarial training) does not necessarily make the adversarially trained model more vulnerable to privacy attacks.

Yeom et al. [28] investigates the impact of overfitting and data influence on an attacker's ability to extract information about the training data from machine learning models, focusing on membership inference and attribute inference attacks. The findings reveal that overfitting is sufficient to enable membership inference attacks and that attribute inference attacks become feasible when the target attribute satisfies specific influence conditions. They explore the connection between membership inference and attribute inference, showing that there are deep connections between the two that lead to effective new attacks.

Srivastava et al. [29] shows that dropout improves the performance of neural networks on supervised learning tasks in vision, speech recognition, document classification and computational biology, obtaining state-of-the-art results on many benchmark data sets. Dropout is to randomly drop units (along with their connections) from the neural network during training. During training, dropout operates by sampling from an exponential number of "thinned" networks, each with a subset of neurons randomly deactivated. At test time, the effect of averaging the predictions of all these thinned networks can be efficiently approximated by using a single, fully active network with scaled-down weights. This approach effectively reduces overfitting and provides significant

improvements compared to other regularization techniques.

Chen et al. [30] rethinks the relationship between overfitting and membership inference attacks and demonstrate that using an overfitting-based approach for membership exclusion can effectively improve the performance of High-Precision MIA (HP-MIA), providing much clearer signals of non-member samples. In scenarios where the cost of launching an attack is high, such signals can avoid unnecessary attacks and reduce the attack's false positive rate. Their HP-MIA is a novel two-stage attack scheme that leverages membership exclusion techniques to guarantee high membership prediction precision. The results show that their attack is able to identify more members while guaranteeing high accuracy compared to other attacks, and having a smaller computational cost method.

Carlini et al. [31] argues that membership inference attacks should be evaluated by considering their true positive rate and low false positive rate. Previous works use an evaluation methodology that considers average-case success metrics, like accuracy or ROC-AUC, that aggregate the attack's accuracy over entire dataset and over all detection thresholds. With their new perspective, they found that previous works perform poorly when evaluated in their way. To address this, we developed a Likelihood Ratio Attack that is mush powerful at low false positive rates, and also dominates prior attacks on existing metrics.

4 Methodology

In this section we shall discuss the various approaches and datasets undertaken to perform different instances of membership inference attacks to assess the effectiveness of the same.

4.1 Cifar-10

We began our study by selecting the CIFAR-10 dataset, which comprises 60,000 color images of size 32×32 pixels, categorized into 10 distinct classes. The dataset is originally divided into 50,000 training images and 10,000 test images. To prepare the data for our experiments, we normalized all images by scaling the pixel values to the range [0, 1], a standard preprocessing step that facilitates efficient training of neural networks.

Next, we combined the original training and test sets to form a unified dataset of 60,000 images. This combined dataset was then shuffled using a random permutation to eliminate any inherent ordering or patterns that could bias the training process. After shuffling, we split the dataset into a new training set of 50,000 images and a new test set of 10,000 images, ensuring a random distribution of classes and samples in both sets.

To simulate a scenario susceptible to membership inference attacks, we intentionally introduced an overlap between the training and test sets. Specifically, we randomly selected 10,000 images from the new training set and added them to the test set. This resulted in a compromised test set containing 20,000 images, half of which were part of the training data. This deliberate inclusion of training samples in the test set was designed to mimic situations where sensitive data may be inadvertently exposed or intentionally targeted by an attacker.

We then created a membership flag array for the test set to indicate the origin of each sample. Each entry in this array was assigned a value of '1' if the corresponding image was part of the training set (a member) and '0' if it was not (a non-member). This membership flag was crucial for evaluating the effectiveness of the membership inference attack later in the study.

To prepare the labels for training, we reshaped the class labels to ensure they were one-dimensional, a necessary step for proper processing in the neural network. We then converted these labels into one-hot encoded vectors for both the training and test sets. One-hot encoding transforms categorical class labels into binary vector representations, which are suitable for multi-class classification problems using categorical cross-entropy loss.

4.1.1 Model 1: Baseline Model

For the model architecture, we designed a convolutional neural network tailored for image classification tasks. The model accepts input images with dimensions 32×32×3, corresponding to the height, width, and color channels of the images. The first convolutional layer applies 32 filters of size 3×3 with a hyperbolic tangent ('tanh') activation function, followed by a max-pooling layer of size 2×2 to reduce spatial dimensions and computational complexity. The second convolutional layer increases the filter count to 64, also using 3×3 filters with 'tanh' activation, and is followed by another max-pooling layer.

After the convolutional layers, the model includes a flattening layer to convert the multi-dimensional output into a one-dimensional feature vector suitable for dense layers. This is followed by a fully connected (dense) layer with 128 neurons and 'tanh' activation, which helps capture complex patterns in the data. The output layer consists of 10 neurons corresponding to the 10 classes in the CIFAR-10 dataset, using a 'softmax' activation function to produce probability distributions over the classes.

We compiled the model using the Adam optimizer, known for its efficiency and effectiveness in training deep learning models. The loss function was set to categorical cross-entropy, appropriate for multi-class classification tasks, and we monitored accuracy as the performance metric during training.

During the training phase, we intentionally omitted regularization techniques and early stopping mechanisms to encourage the model to overfit the training data. Overfitting occurs when a model learns the training data too well, including its noise and outliers, resulting in poor generalization to new, unseen data. By promoting overfitting, we aimed to amplify the differences in prediction confidence between samples the model had seen during training and those it had not.

The model was trained on the 50,000-image training set for up to 100 epochs, using a batch size of 256 to balance computational efficiency and convergence speed. Validation was performed on the compromised test set of 20,000 images, which included both member and non-member samples. This setup allowed us to observe the model's performance on data it had seen during training versus data it had not, providing a basis for the membership inference attack.

4.1.2 Model 2: Introducing Early Stopping

To enhance the model and reduce its vulnerability to membership inference attacks, we introduced early stopping during the training process. Early stopping is a regularization technique that halts training when the model's performance on a validation set ceases to improve, thereby preventing overfitting. By incorporating

early stopping, we aimed to limit the model's ability to memorize the training data, which in turn should reduce the disparity in confidence scores between training and non-training samples.

In this revised approach, we maintained the same initial data preparation steps as before. We used the CIFAR-10 dataset, normalized the images, concatenated and shuffled the dataset, and performed a new train-test split. We also continued to inject 10,000 training images into the test set to create a scenario conducive to membership inference attacks. The membership flags were retained to evaluate the effectiveness of the attack later on.

The model architecture remained unchanged, consisting of convolutional and pooling layers with 'tanh' activation functions, followed by a flattening layer, a fully connected layer, and an output layer with 'softmax' activation. The consistency in architecture allowed us to isolate the impact of early stopping on the model's susceptibility to attacks.

The significant change in this iteration was in the training procedure. We introduced an early stopping callback that monitored the validation loss during training. Specifically, we set the early stopping mechanism to observe the validation loss and halt training if it did not improve for two consecutive epochs (a patience of 2). Additionally, we enabled the (restore.best.weights) parameter to ensure that the model weights were reverted to those from the epoch with the lowest validation loss. This approach helped in retaining the model state that best generalized to unseen data.

By implementing early stopping, we intended to prevent the model from overfitting the training data. In the previous model, the lack of overfitting safeguards allowed the model to learn noise and specific patterns unique to the training set, leading to higher confidence scores for training samples. This made it easier for an attacker to distinguish between member and non-member samples based on the confidence levels. Early stopping curtailed this behavior by stopping the training process before the model began to overfit, promoting better generalization to new data.

4.1.3 Model 3: Architectural Enhancements and Regularization

Building upon the previous model where early stopping was introduced to mitigate overfitting, we further refined the neural network architecture to enhance its generalization capabilities and reduce vulnerability to membership inference attacks. In this third iteration, significant modifications were made to both the model's structure and training regimen to strengthen its robustness against privacy breaches.

The primary architectural enhancement involved expanding the depth of the convolutional neural network by

adding a third convolutional layer. This additional layer allowed the model to extract more complex and abstract features from the input images, thereby improving its ability to generalize beyond the training data. Each convolutional layer was designed to capture hierarchical patterns, starting from simple edges and textures in the initial layers to more intricate features in the deeper layers.

To improve the activation dynamics within the network, we replaced the 'tanh' activation functions used in the previous models with LeakyReLU activation functions. LeakyReLU addresses the issue of "dying neurons" that can occur with traditional ReLU activations by allowing a small, non-zero gradient when the unit is not active. This change enhanced the model's learning capabilities and contributed to better performance and generalization.

Specifically, the first convolutional layer applied 32 filters of size 3×3, followed by a max-pooling layer with a pool size of 2×2 to reduce spatial dimensions. A LeakyReLU activation function with an alpha parameter of 0.3 was then applied. The second convolutional layer increased the number of filters to 64, maintained the 3×3 filter size, and was followed by another max-pooling layer and LeakyReLU activation. The third convolutional layer further increased the filters to 128, again with a 3×3 filter size, followed by a max-pooling layer and LeakyReLU activation. This progressive increase in filter numbers enabled the network to learn increasingly complex feature representations.

Following the convolutional layers, a flattening layer converted the multi-dimensional output into a one-dimensional feature vector suitable for the dense layers. We introduced multiple fully connected layers with decreasing numbers of neurons: 256, 128, and 64 units, respectively. Each dense layer was followed by a LeakyReLU activation function and a Dropout layer with a dropout rate of 0.4. The inclusion of Dropout served as a regularization technique by randomly setting a fraction of input units to zero during training, which prevented units from co-adapting too closely and reduced overfitting.

The output layer consisted of 10 neurons corresponding to the 10 classes in the CIFAR-10 dataset, using a 'softmax' activation function to produce probability distributions over the classes. This configuration was suitable for multi-class classification tasks and aligned with the categorical cross-entropy loss function used during training.

In terms of training adjustments, we increased the patience parameter in the early stopping mechanism from 2 to 10 epochs. This modification allowed the model more epochs to improve before halting training, potentially achieving a better balance between underfitting and overfitting. The early stopping callback monitored the validation loss and restored the best weights observed during training, ensuring that the model parameters

corresponded to the lowest validation loss and thus better generalization to unseen data.

We also reduced the batch size from 256 to 128, which introduced more variability in the gradient updates and acted as an additional form of regularization. A smaller batch size can help the model escape shallow local minima and potentially lead to better generalization performance.

The model was compiled using the Adam optimizer, known for its efficiency and adaptability in training deep learning models. The loss function remained categorical cross-entropy, and accuracy was used as the performance metric. Training was conducted on the 50,000-image training set, with validation performed on the compromised test set of 20,000 images, which included both member and non-member samples.

By incorporating these architectural enhancements and training modifications, we aimed to produce a more robust model less prone to overfitting. The addition of LeakyReLU activations and Dropout layers, combined with the adjusted early stopping parameters, were expected to reduce the confidence gap between member and non-member samples. This, in turn, should make it more challenging for an attacker to perform a successful membership inference attack based solely on prediction confidences.

4.2 CIFAR-100

4.2.1 CNN

4.2.1.1 Model 4: Baseline

In this initial experiment with CIFAR-100, we seek to evaluate the susceptibility of a basic convolutional neural network (CNN) architecture to membership inference attacks. Building on a methodology similar to that applied in the CIFAR-10 context, we replicate the scenario using CIFAR-100—a more complex dataset with 100 classes, but still composed of 60,000 images at 32×32 pixel resolution.

We begin with the CIFAR-100 dataset, which includes 50,000 training images and 10,000 test images, each belonging to one of 100 categories. As a preprocessing step, we normalize the pixel values of all images to the [0, 1] range. Rather than maintaining the original split, we combine all 60,000 images and shuffle them to eliminate any inherent order. From this pool, we then create a new training set of 50,000 images and a new test set of 10,000 images. To set the stage for membership inference, we artificially introduce an overlap between the training and test sets. Specifically, we randomly select 10,000 images from the new training set and add them to the test set. This results in a compromised test set with 20,000 images, half of which were used in training. We create a membership flag array, assigning '1' to images that originated from the training set and '0' otherwise. These labels allow us to measure how effectively an attacker can distinguish between

training (member) and non-training (non-member) examples based solely on model outputs. The class labels are reshaped into a one-dimensional format and then one-hot encoded for both training and test sets. One-hot encoding is essential for training with categorical cross-entropy loss. This data preparation pipeline ensures that each step—from normalization and shuffling to the creation of a compromised test set—is consistent with previous experiments in simpler scenarios, thus facilitating direct comparisons.

The selected CNN model is intentionally kept simple. It begins with a convolutional layer of 32 filters (3×3 kernel) using 'tanh' activation, followed by a 2×2 max-pooling layer to reduce spatial dimensions. A second convolutional layer with 64 filters and 'tanh' activation is then applied, followed by another max-pooling layer. After flattening the feature maps, a fully connected layer of 128 'tanh' neurons processes the extracted features, before passing them to a final softmax output layer consisting of 100 neurons—one for each CIFAR-100 class. We compile the model with the Adam optimizer and categorical cross-entropy loss. Accuracy is monitored as the primary metric. As with previous baseline experiments, we intentionally exclude regularization methods or early stopping measures. By doing so, we encourage the model to overfit the training data. Although overfitting generally leads to poor generalization, it is precisely this behavior that can amplify differences in model confidence between training and non-training samples, thus increasing susceptibility to membership inference attacks. The model is trained for up to 100 epochs with a batch size of 256. Validation is performed on the compromised 20,000-image test set after each epoch. Despite the dataset's complexity, and the model's simplicity, the intention is not to achieve high accuracy, but rather to produce a model that illustrates how confidence scores can reveal membership information.

These findings underscore that even a poorly performing model, if overfit, can leak membership information. The low accuracy on non-training samples and high attack AUC reveal a potential privacy risk: a model need not be well-trained to divulge whether a data point was used in its training process. This initial result sets the stage for examining subsequent models, where we will introduce techniques like early stopping, dropout, or more advanced architectures to improve generalization and potentially reduce membership inference vulnerability.

4.2.1.2 Model 5: Early Stop

For our second model, we build upon the initial approach used in the first experiment. As before, we focus on the CIFAR-100 dataset to examine membership inference vulnerability, while intentionally creating a scenario where some training samples also appear in the test set. The core difference in this iteration is the introduction of early stopping as a basic form of regularization, which aims to mitigate overfitting and, in turn, reduce the model's susceptibility to membership inference attacks.

We begin with the CIFAR-100 dataset containing 60,000 color images (50,000 training and 10,000 test) spread

across 100 classes. As in the previous model, we normalize the pixel values to the [0, 1] range. We merge and shuffle the combined dataset of 60,000 images to remove any ordering bias. From this shuffled pool, we select 50,000 images as our training set and 10,000 images as our new test set. To create a scenario conducive to membership inference attacks, we follow the same procedure as before: randomly select 10,000 images from the training set and add them to the test set, forming a compromised test set of 20,000 images. Half of these images were used during training (members) and half were not (non-members). We assign a membership flag of '1' to training-origin images and '0' otherwise, thereby enabling a direct evaluation of membership inference performance. As in the previous experiment, labels are reshaped and one-hot encoded. This ensures compatibility with categorical cross-entropy and provides a consistent experimental setup across all models.

We maintain the same basic CNN architecture as used in the first model: a)Two convolutional layers (32 and 64 filters respectively, both using 'tanh' activation) followed by max-pooling layers. b)A flattening layer that converts the 2D feature maps into a 1D feature vector. c)A fully connected layer with 128 'tanh' neurons. d)A final softmax output layer with 100 units corresponding to the 100 CIFAR-100 classes.
The model is compiled with the Adam optimizer and categorical cross-entropy loss, monitoring accuracy as the primary performance metric. This time, however, we introduce early stopping. Early stopping monitors the validation loss on the compromised test set and halts training once the validation loss fails to improve for a specified number of epochs (in this case, a patience of two). By doing this, we aim to prevent the model from memorizing the training data excessively, potentially leading to more similar confidence distributions for training and non-training samples.

The model is trained for up to 100 epochs with a batch size of 256, but likely terminates earlier due to early stopping criteria. Validation is conducted on the 20,000-image compromised test set after each epoch. Although the model's architecture remains unchanged, the addition of early stopping is expected to yield a model that generalizes better than the first model, potentially reducing membership inference vulnerability.

4.2.1.3 Model 6: Good

In the third model, we continue exploring strategies that reduce the vulnerability to membership inference attacks on CIFAR-100. Having observed that early stopping alone provides a moderate reduction in the effectiveness of these attacks, we further refine the model architecture and introduce additional forms of regularization. Although the chosen architecture still struggles with CIFAR-100's complexity, these steps aim to strike a better balance between reducing overfitting and improving resistance to membership inference.

As in the previous two models, we use the CIFAR-100 dataset and follow the same procedure for data prepa-

ration. The 50,000 training images and 10,000 test images are combined, shuffled, and then split into a new 50,000-image training set and a 10,000-image test set. We replicate the membership inference scenario by adding 10,000 randomly selected training images to the test set, resulting in a 20,000-image compromised test set containing equal numbers of member and non-member samples. To facilitate evaluating membership inference, we retain the membership flag array, with '1' indicating samples originally from the training set and '0' otherwise. Labels are reshaped into one-dimensional format and then one-hot encoded for both the training and test sets. We maintain consistent preprocessing and data manipulation steps to ensure direct comparability across all three models.

While the first two models employed simple CNN architectures with 'tanh' activations, this third model introduces two key changes: Deeper Architecture and Activation Change: We add a third convolutional block to increase the model's representational capacity. Each convolutional block is followed by a max-pooling layer to reduce spatial dimensions. Instead of 'tanh', we use LeakyReLU as the activation function. LeakyReLU can improve gradient flow and potentially yield more robust features compared to 'tanh', helping the model to learn without overly memorizing the training data. Dropout Layers: To further combat overfitting, we incorporate dropout layers into the fully connected portion of the network. Dropout randomly deactivates a fraction of neurons during training, compelling the network to develop more generalizable features rather than relying on memorized training patterns. This measure is crucial for diminishing the disparity in confidence scores between seen and unseen samples, potentially lowering membership inference attack success. We compile the model with the Adam optimizer and use categorical cross-entropy as the loss function, monitoring accuracy as the main metric. Early stopping is employed once again, this time with a larger patience value, allowing the model more epochs to find a good generalization point while still preventing excessive overfitting.

The model is trained for up to 100 epochs with a batch size of 128. Early stopping monitors the validation loss on the compromised 20,000-image test set and halts training if no improvement is observed for several epochs. Although these measures aim to enhance the model's resistance to membership inference attacks, the model still faces the inherent difficulty of classifying 100 classes from small 32×32 images with a relatively simple architecture.

Comparing this model's outcome with the previous two highlights a clear trend: as we introduce simple regularization techniques and incremental architectural improvements, membership inference attacks become less effective. Although the accuracy remains low due to the architecture's inadequacy for CIFAR-100, the reduction in AUC from 0.85 to 0.56 underscores the importance of even basic regularization and network refinement. This set of experiments shows that while poor architectures may struggle to classify complex datasets, combining early stopping, dropout, and slightly deeper configurations can still reduce privacy leakage. The stage is

now set to explore even better architectures and more sophisticated techniques to further enhance accuracy and privacy-preserving properties simultaneously.

4.2.2 Wide-ResNET

4.2.2.1 Model 7: Baseline

After examining three initial models with simpler architectures and observing their vulnerabilities to membership inference attacks, we now transition to a more sophisticated model design. In this iteration, we use a Wide-ResNet architecture, which is known for better performance on CIFAR datasets when properly configured. Although the model is still trained on CIFAR-100 with a compromised test set for membership inference assessment, this architecture represents a step towards more realistic and higher-performing networks.

The data preparation process closely follows the methodology established in previous models, ensuring consistency and direct comparability. We use the CIFAR-100 dataset, composed of 60,000 images (50,000 training and 10,000 test) across 100 classes, each of size 32×32 pixels. The images are normalized to the [0, 1] range to facilitate stable training. We combine and shuffle the entire dataset of 60,000 images, then split it into a new training set of 50,000 images and a new test set of 10,000 images. To create a scenario amenable to membership inference attacks, we select 10,000 training images at random and add them into the test set, resulting in a 20,000-image test set with half of the images originating from training (members) and half not (non-members). As before, a membership flag array (1 for training members, 0 for non-members) is established, and labels are reshaped and one-hot encoded.

Unlike the earlier CNN-based models, we now employ a Wide-ResNet, a variant of the Residual Network (ResNet) architecture that widens layers rather than simply adding more depth. Specifically, we use a Wide-ResNet-28-10 configuration: Depth: 28 layers, conforming to the 6n+4 rule for Wide-ResNet. Widening Factor: 10, increasing the number of filters and thus model capacity. Dropout Rate: For this initial run with the Wide-ResNet architecture, we set dropout to 0, relying on the architecture's strong baseline performance before introducing additional regularization. The Wide-ResNet employs multiple residual blocks that each contain two convolutional layers, batch normalization, and ReLU activations. A shortcut connection in each block allows gradients to flow more easily during backpropagation, facilitating the training of deeper models. Global average pooling is used before the final fully connected layer to produce logits for each of the 100 classes. By using a more complex and well-established architecture, we aim to achieve better accuracy on CIFAR-100, especially on non-training images, thereby potentially reducing the stark confidence disparities observed in simpler models.

We compile the model with the Adam optimizer and use categorical cross-entropy loss, tracking accuracy as the

primary metric. The model is trained for 50 epochs with a batch size of 128, validating on the compromised 20,000-image test set after each epoch. Compared to previous models, this reduced training time and lack of early stopping reflect a confidence in the network's ability to generalize better due to its architectural strengths.

The introduction of a more advanced model architecture such as Wide-ResNet-28-10 on CIFAR-100 demonstrates a step forward in terms of classification accuracy on challenging data. However, the membership inference AUC remains relatively high, underscoring that even capable and better-generalizing models may remain vulnerable to attacks that leverage confidence scores. This result highlights the need for integrating privacy-preserving techniques alongside architectural improvements to truly mitigate membership inference risks.

4.2.2.2 Model 8: Good

In this final model, we further refine our approach to mitigating membership inference attacks by combining a high-capacity architecture (Wide-ResNet) with a more nuanced training regime that involves a two-phase process. This approach builds on the lessons learned from previous models: while stronger architectures can improve accuracy on non-training data and reduce overfitting, and while simple measures like early stopping and dropout can tamper membership inference vulnerability, there is no single universal fix. Instead, we integrate multiple strategies to achieve a better balance between accuracy and privacy.

As with the previous models, we rely on the CIFAR-100 dataset. The 50,000 training images and 10,000 test images are normalized to [0, 1], combined, shuffled, and split into a new 50,000-image training set and a new 10,000-image test set. To establish a scenario conducive to membership inference attacks, we inject 10,000 training images into the test set, yielding a 20,000-image test set with equal numbers of member (training) and non-member (non-training) samples. A membership flag (1 for training images, 0 for non-training images) allows us to evaluate the success of the membership inference attack later on. The labels are reshaped and one-hot encoded to facilitate training with categorical cross-entropy loss. These consistent and carefully controlled data handling steps ensure that differences in model behavior can be attributed to architectural and training changes rather than variations in data preparation.

We continue to use a Wide-ResNet-28-10 architecture. This architecture significantly increases model capacity compared to earlier, simpler CNNs. The Wide-ResNet introduces multiple residual blocks, each containing batch normalization, ReLU activations, and convolutional layers. A widening factor of 10 increases the number of filters at each stage, enhancing representational capacity. Global average pooling before the final dense layer avoids the need for large fully connected layers, aiding generalization. A dropout rate of 0.3 is employed to counter overfitting. Dropout encourages the network to learn more robust features rather than memorize the training set, thereby potentially reducing the confidence gap between training and non-training samples. Two-

Phase Training Procedure A novel element in this final model is the two-phase training approach: Phase 1 (No Dropout to Start): Initially, the model is trained for 20 epochs without early stopping or other forms of strict training halts. This "warm-up" phase allows the model to begin learning features and partially fit the training data. After these 20 epochs, we save the model weights. Phase 2 (Dropout, Early Stopping, and Continued Training): We then reload the saved weights into an identical architecture that includes the same dropout rate of 0.3 and introduce early stopping with a patience of 20 epochs. Starting from epoch 20, we continue training the model up to a maximum of 100 epochs, but with early stopping monitoring validation loss. Early stopping halts training when no further improvement is observed, and restoreBestWeights=True ensures the final model represents the best validation performance achieved during this phase. By starting training without immediate early stopping and then switching to a more cautious approach after an initial "burn-in" period, we aim to let the model find a reasonable representation before enforcing stricter regularization controls. This staged training process aspires to yield a final model state that generalizes better while still being guided away from memorizing too many training samples.

4.3 Shadow Models

Shokri et al.'s [2] MIA primarily depends on the trained target model, typically accessed through an API. The attack involves four main steps:

1. Generating shadow data
2. Training shadow models
3. Constructing attack data
4. Training the attack model

Shadow data is used to train shadow models as substitutes for the target model, requiring it to mimic the distribution of the target model's training data. Shokri et al. [2] propose three approaches to generate such data:

1. **Hill-Climbing Algorithm:** This method synthesizes data iteratively by initializing random feature values and updating them to maximize the target model's confidence for a specific class. Once the confidence surpasses a threshold, the data is added to the shadow dataset with a predefined probability. While effective, this approach necessitates extensive querying of the target model, as each update requires a separate query.
2. **Noisy Data Generation:** This method introduces randomness by flipping a percentage of binary features (e.g., 10–20%) in the original data records. However, it requires access to the target model's training dataset, making it impractical for real-world scenarios.
3. **Sampling from Marginal Distributions:** Synthetic data is generated by sampling independently from

the marginal distributions of each feature. This approach also relies on knowledge of the target model's training data, limiting its real-world applicability.

Shokri et al. [2] propose using disjoint subsets of the original dataset to create shadow data, ensuring it is independent of the target model's training data. Among the proposed techniques, the hill-climbing algorithm stands out as the only true black-box method, relying exclusively on query access to the target model without requiring additional knowledge.

To mimic the target model, Shokri et al. [2] trained multiple shadow models as local substitutes, using 10 to 100 models depending on the dataset. These shadow models were trained with similar architectures to the target model when its structure was known. The attack relies on transferring knowledge from the shadow models to infer information about the target model.

In cases where the target model's architecture was unknown, such as when attacking a black-box model via a prediction API, Shokri et al. [2] suggested using the API itself to train the shadow models. However, they did not provide detailed information on the exact implementation of this approach in their experiments.

To execute the attack, a model is trained to perform a specific classification task: determining whether a given data point was part of the target model's training data based on its prediction vector from the target model.

The training data for this attack model is generated using shadow models. Data points from the shadow models' training and test datasets are passed through these models, and each point is labeled based on its membership status. Points from the training dataset are labeled as "in," while points from the test dataset are labeled as "out," creating a binary classification task. The attack model's inputs are the prediction vectors from the shadow models, with the corresponding "in" or "out" labels serving as the outputs.

This process is class-specific: for example, when attacking a dataset with 100 classes, 100 separate attack models are trained, one for each class. This requires sorting the data by their ground truth classes before feeding them into the shadow models to generate predictions.

For the membership inference attack, a separate attack model is trained for each class. The target model's predicted class label for a given record determines which attack model is used. The class probabilities from the target model are fed into the corresponding attack model, which then outputs a two-dimensional probability vector indicating whether the record was part of the training data. The attack model's effectiveness is evaluated using precision and recall metrics:

1. Precision: The proportion of correctly inferred members among all records identified as members.
2. Recall: The proportion of actual members correctly identified by the attack model.

Shokri et al.'s [2] experiments show that recall is typically high (close to 1), while precision varies, making precision especially relevant for assessing the attack's reliability in practical scenarios. Precision and recall values vary across classes, and their distributions are analyzed using empirical cumulative distribution function (ECDF) plots. These plots illustrate how precision and recall differ among classes, with better-performing attacks having precision and recall values concentrated at higher levels.

4.3.1 Shadow Model

In our project, we aim to evaluate the effectiveness of membership inference attacks by implementing a pipeline that includes shadow model training, feature extraction, and attack model evaluation. We utilize the CIFAR-10 dataset, a standard benchmark dataset for image classification tasks, containing 60,000 images across 10 classes. The dataset was split into two subsets: shadow data, used to train the shadow model, and evaluation data, which served as non-membership data for simulating attacks. To standardize the dataset, we applied preprocessing transformations using PyTorch's transforms.Compose. These transformations included ToTensor() to convert images into tensors and Normalize() to scale pixel values to a mean of 0.5 and a standard deviation of 0.5 across all channels. A train-test split ensured that the data remained unbiased during model training and evaluation, and dataLoaders were configured with a batch size of 128 to streamline efficient data loading during training and inference.

The ResNet18 architecture, pre-defined in PyTorch's torchvision.models, was employed as the shadow model to classify the CIFAR-10 dataset into 10 classes. The model was trained using CrossEntropyLoss as the loss function and optimized with the Adam optimizer at a learning rate of 0.001. The shadow dataset was split evenly into training and validation sets (50-50) to evaluate the model's performance.

The train_shadow_model function handled the training process over 20 epochs. In each epoch, the model was trained on the training set, and the optimizer updated the weights to minimize the loss. The training loss decreases consistently over the 20 epochs, starting at 1.6814 in the first epoch and dropping to 0.0777 by the final epoch. This steady decline indicates that the shadow model is learning effectively and minimizing the classification error. However, the very low final loss suggests that the model may be overfitting to the training data, as it continues to improve without signs of plateauing.

```
Epoch 1/20, Loss: 1.6814
Epoch 2/20, Loss: 1.2995
Epoch 3/20, Loss: 1.1014
Epoch 4/20, Loss: 0.9373
Epoch 5/20, Loss: 0.7682
Epoch 6/20, Loss: 0.6133
Epoch 7/20, Loss: 0.4939
Epoch 8/20, Loss: 0.3873
Epoch 9/20, Loss: 0.3019
Epoch 10/20, Loss: 0.2663
Epoch 11/20, Loss: 0.1924
Epoch 12/20, Loss: 0.1878
Epoch 13/20, Loss: 0.1526
Epoch 14/20, Loss: 0.1292
Epoch 15/20, Loss: 0.1088
Epoch 16/20, Loss: 0.1001
Epoch 17/20, Loss: 0.1019
Epoch 18/20, Loss: 0.1105
Epoch 19/20, Loss: 0.0837
Epoch 20/20, Loss: 0.0777
```

Figure 2: Shadow Model Training Loss over 20 Epochs.

To evaluate the shadow model's performance, the trained model was tested on a separate validation set, which was not used during training. This step was necessary in order to check how well the model performs on unseen data, ensuring it has not simply memorized the training set. Once the shadow model was validated, we used it to extract logits, which are the raw outputs from the final layer before applying the softmax activation. These logits were used because they can retain subtle differences between data the model was trained on (members) and data it has not seen (non-members).

The goal of feature extraction was to create two sets of data: one for members and another for non-members, which would later be used to train the attack model. For the member dataset, we passed the training data through the shadow model and saved the resulting logits. For the non-member dataset, we did the same using the validation data. To automate this process, we defined the extract_features function, which processes data in batches and calculates the softmax probabilities from the logits. Each feature vector was combined with its class label so the attack model could use it for supervised learning. This prepared dataset provided the inputs the attack model needed to learn how to differentiate between members and non-members.

The attack model, designed as a binary classifier, was implemented to infer membership status (member vs. non-member) using the extracted features. Its architecture consisted of an input layer with 10 features (softmax probabilities), followed by two fully connected layers: $10 \longrightarrow 128$ neurons and $128 \longrightarrow 64$ neurons, both with ReLU activation and a 30% dropout rate for regularization. The final output layer consisted of $64 \longrightarrow 1$ neuron with a sigmoid activation for binary classification. The model was trained using Binary Cross-Entropy Loss (BCELoss) and optimized with the Adam optimizer at a learning rate of 0.001. To ensure fairness, the attack dataset was balanced with equal member and non-member samples, and split into 80% for training and 20% for

validation. The training was conducted over 20 epochs using the train_attack_model function, which recorded the loss and accuracy to monitor the model's performance.

5 Results

This sections discusses the findings from the experiments conducted which include the performance characteristics of the ML models created and it's association with privacy leaks.

5.1 CIFAR-10

3 models were trained and the possibility and effectiveness of performing a successful membership inference attack was observed. Accuracy in predicting the true outcome of a given input data, is crucial as it indicates how well the ML model performs. The accuracy associated with the three trained models are shown in Figure 3, Figure 4 and Figure 5.

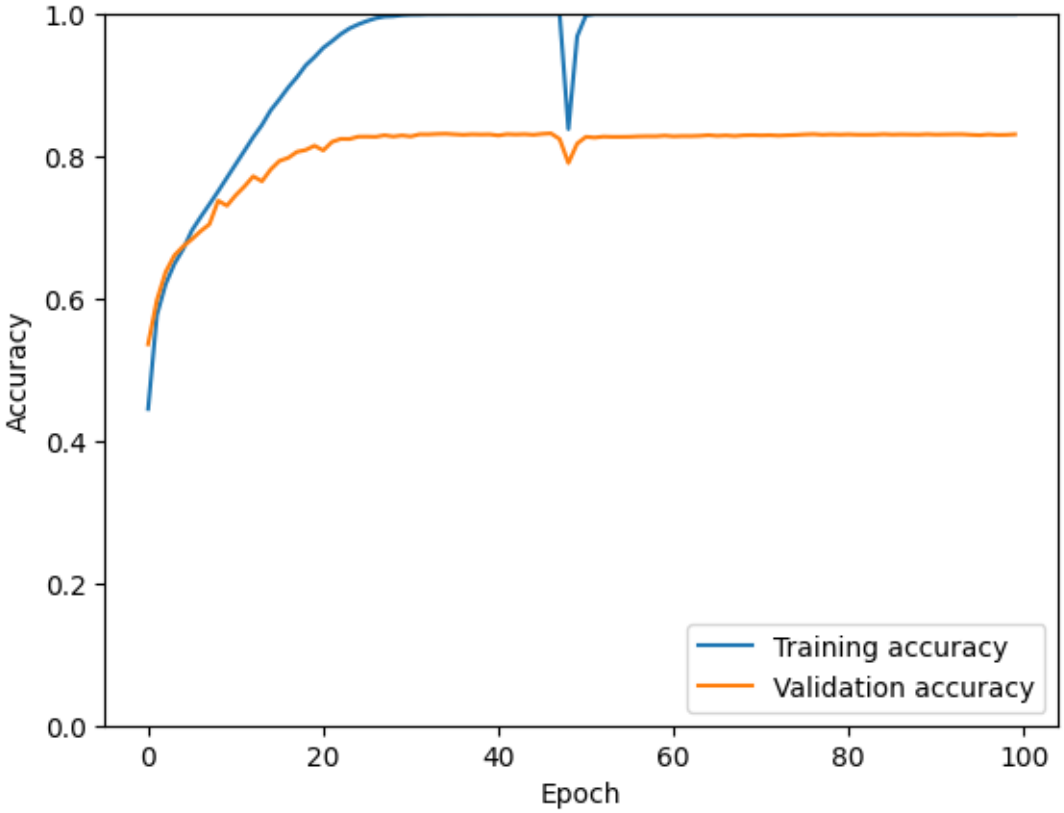

Figure 3: Training and Validation Accuracy of model 1

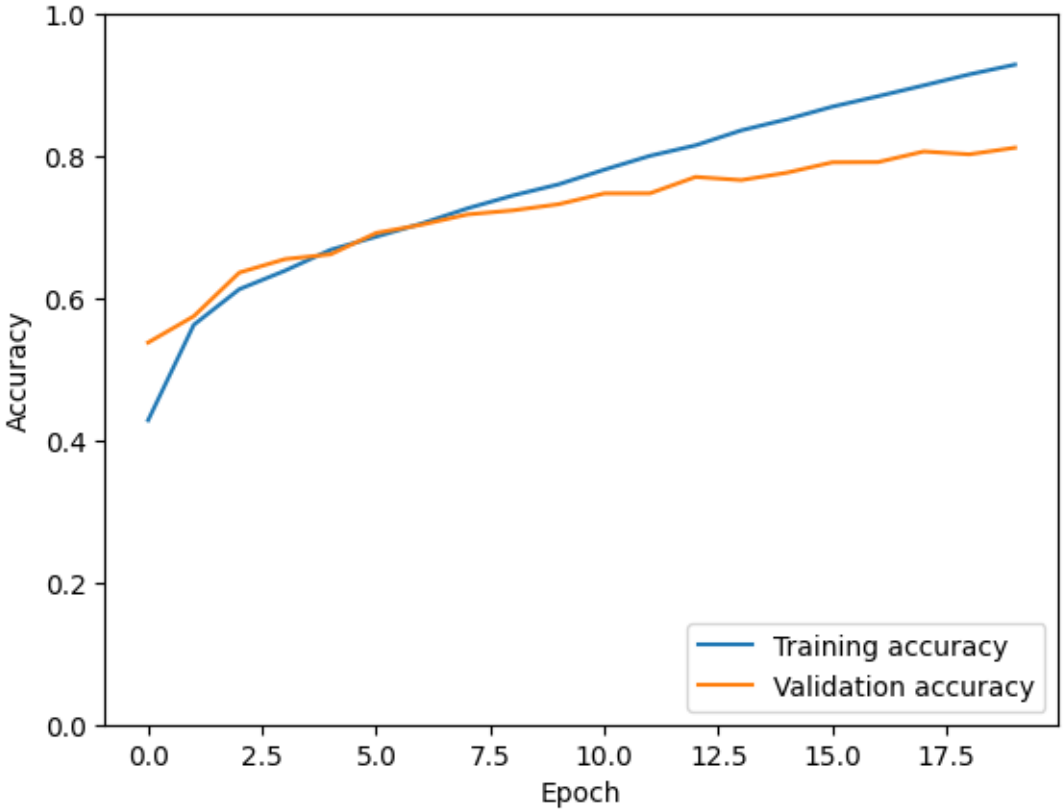

Figure 4: Training and Validation Accuracy of model 2

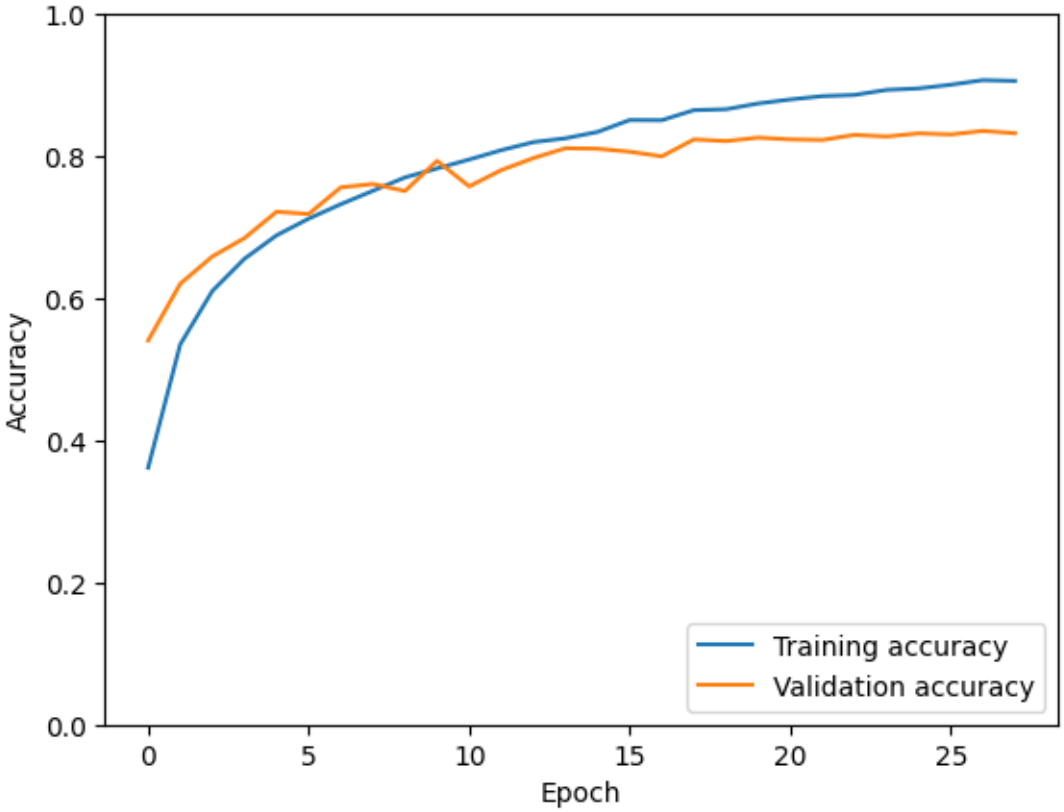

Figure 5: Training and Validation Accuracy of Model 3

The base model, model 1 was created with the target of being the most vulnerable to overfitting. Figure 6 depicts the confidence with which the model predicts both the testing and training datasets. As seen in the figure, the graph associated with the training dataset increases steeply at a confidence score of 0.95 while the graph associated with the testing dataset is comparatively less steep. There is a notable difference in the area beneath the curves pertaining to the training and testing datasets which indicates a privacy leak. This is further discussed in Figure 5.4.

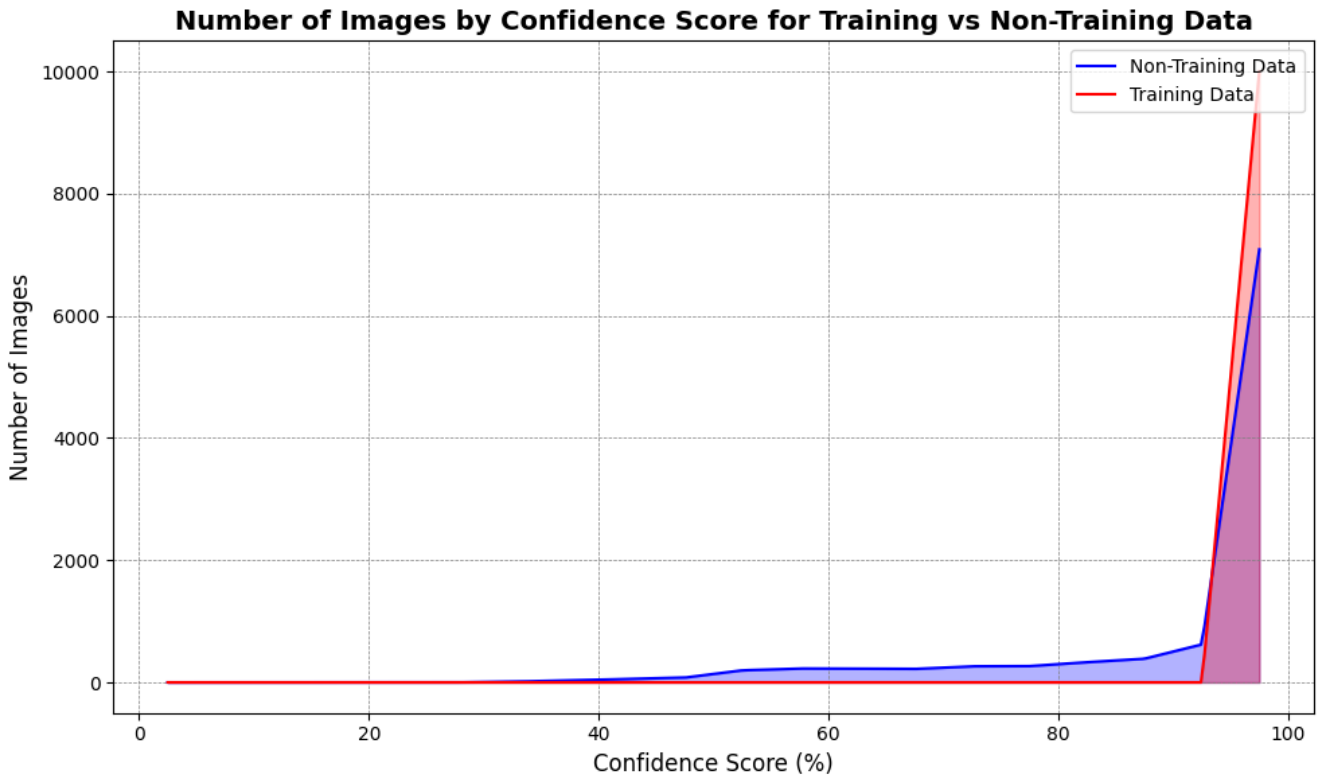

Figure 6: Confidence with which Model 1 predicted its training and testing datasets

While training the model 2, training was halted after the validation loss during training remained consistent and unchanged over a couple of epoch cycles. Figure 6 similar relations between the training and testing datasets as compared to the Figure 6. It can be inferred from the area under the graphs that the model provides comparatively similar predictions on the training and testing dataset.

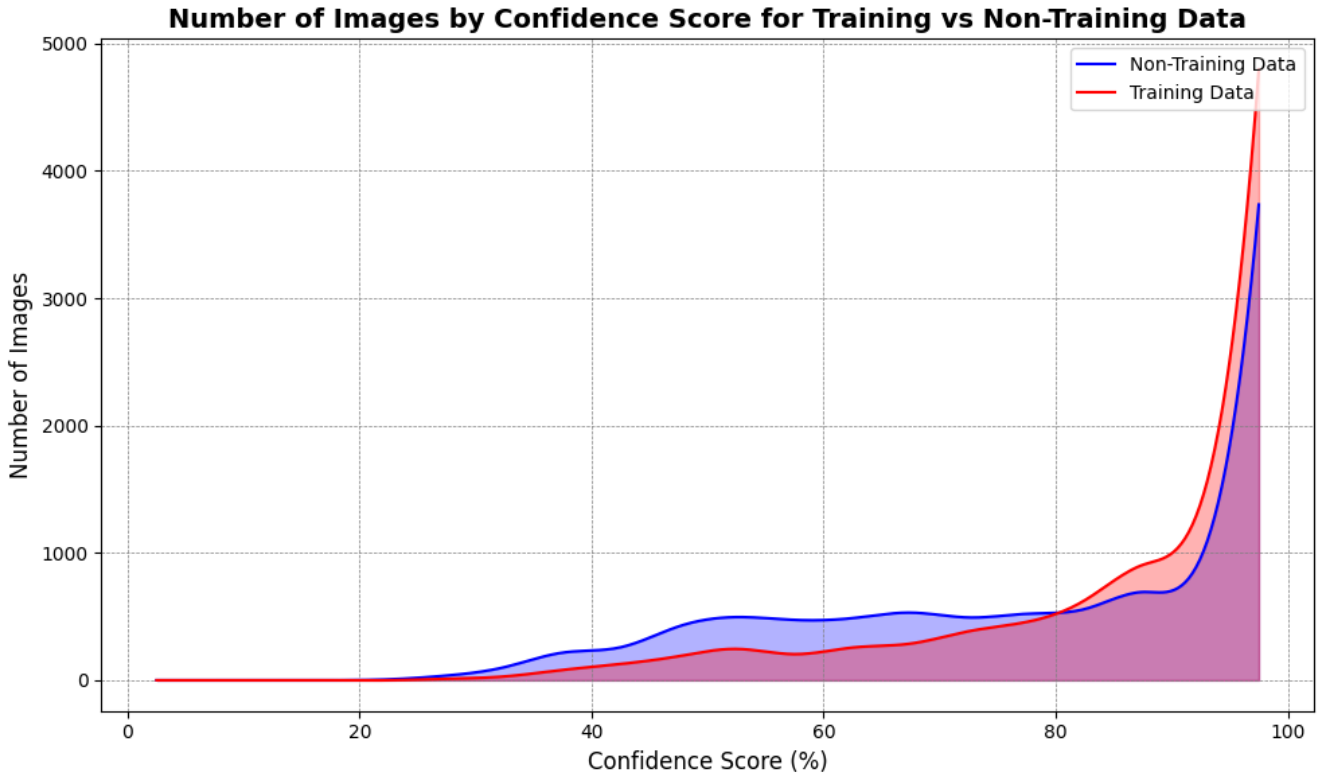

Figure 7: Confidence with which Model 2 predicted its training and testing datasets

Model 3 proved to be the least overfitted model, as a result of substituting the LeakyRelu activation function. In the graph Figure 8, the two graphs are a lot more similar as opposed to the two previous graphs, thus indicating the model performs equally on seen and unseen data.

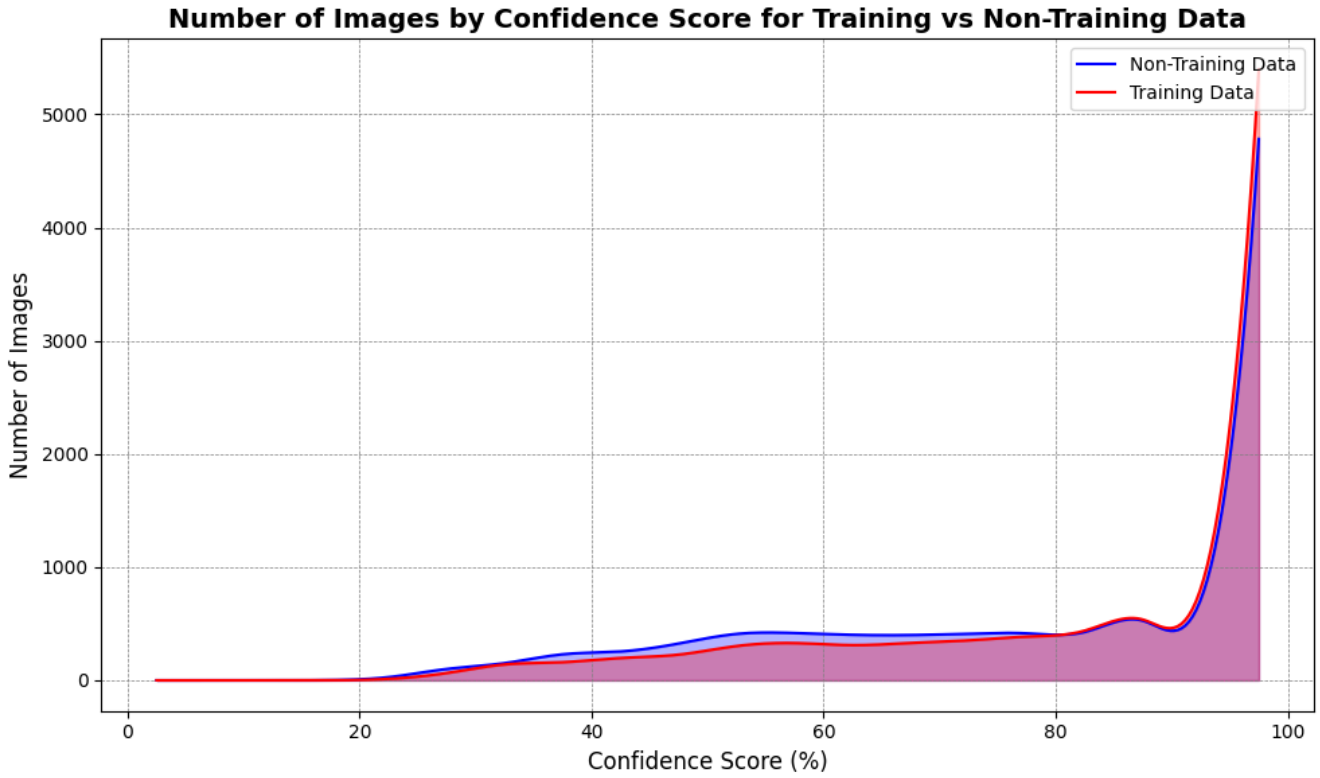

Figure 8: Confidence with which Model 3 predicted its training and testing datasets

A loss function is a mathematical function that is used in the optimization process during the training phase of a model. It measures how well the model's predictions compare to the actual true values of the target data records. Figure 9, Figure 10 and Figure 11 illustrates the result of loss function across the training and testing datasets of models 1, 2 and 3.

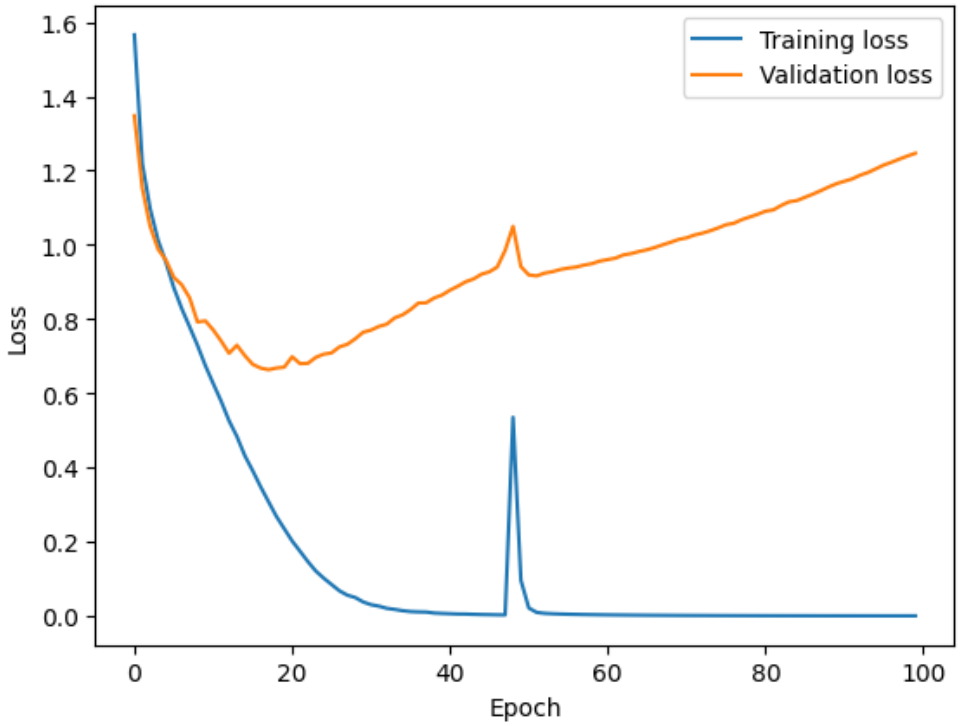

Figure 9: Loss function of Model 1

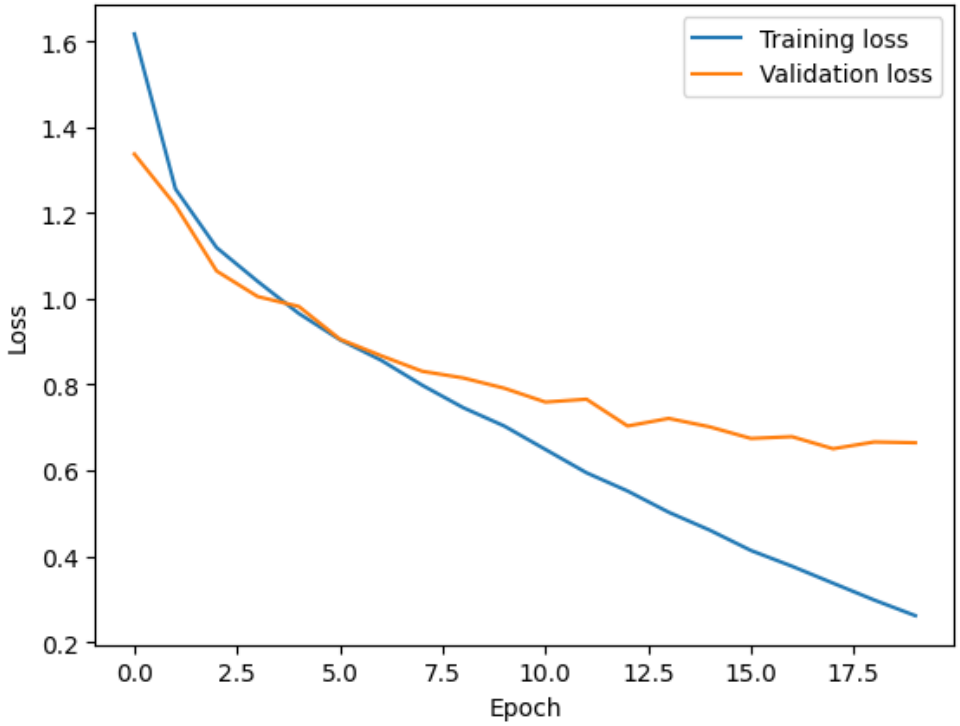

Figure 10: Loss function of model 2

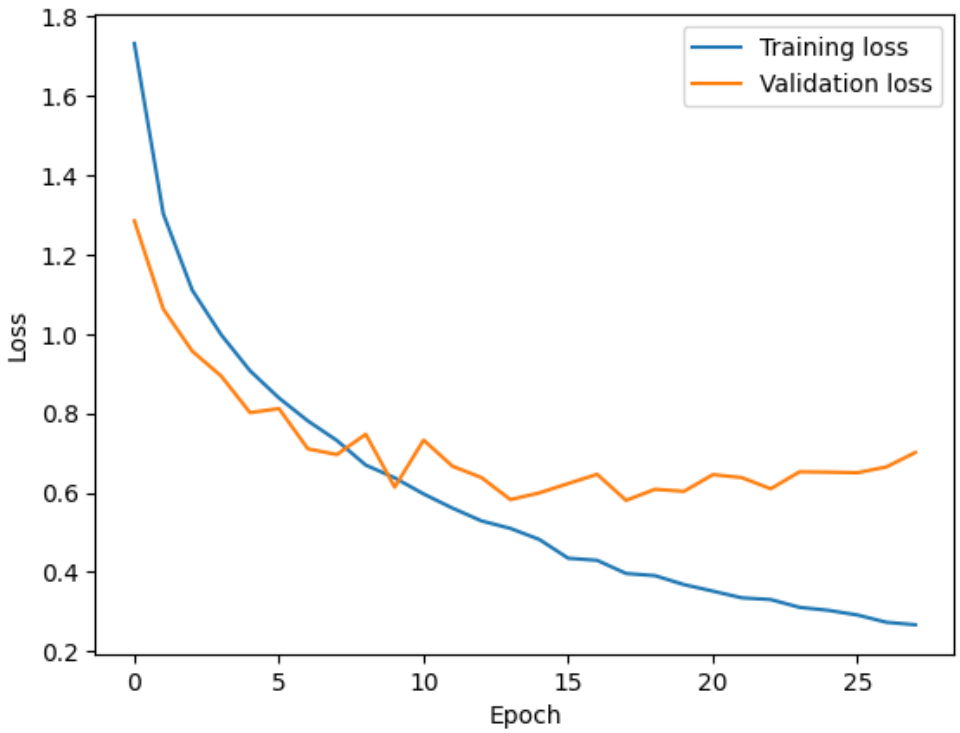

Figure 11: Loss function of model 3

As discussed in section 2, ROC curve is a graphical plot between the true positive rate and the false positive rate of a given ML model. The ROC curves for each of these models have been included below as Figure 12, Figure 13 and Figure 14. The ROC curves were built based on the 'membership_flag' label. This label was manually created during the data processing stage for each record. While the training data records were given a value of '1', the testing data records were a given a value of '0'. The purpose of these ROC curves is to determine how well the confidence score could be a predictor of whether a specific data point was used in the training phase or not. A curve closer to the y-axis suggests that it was easier to distinguish the training members from the non training members while the curve closer to the diagonal suggests random guessing i.e, it was difficult to certainly associate a certain input to the training dataset.

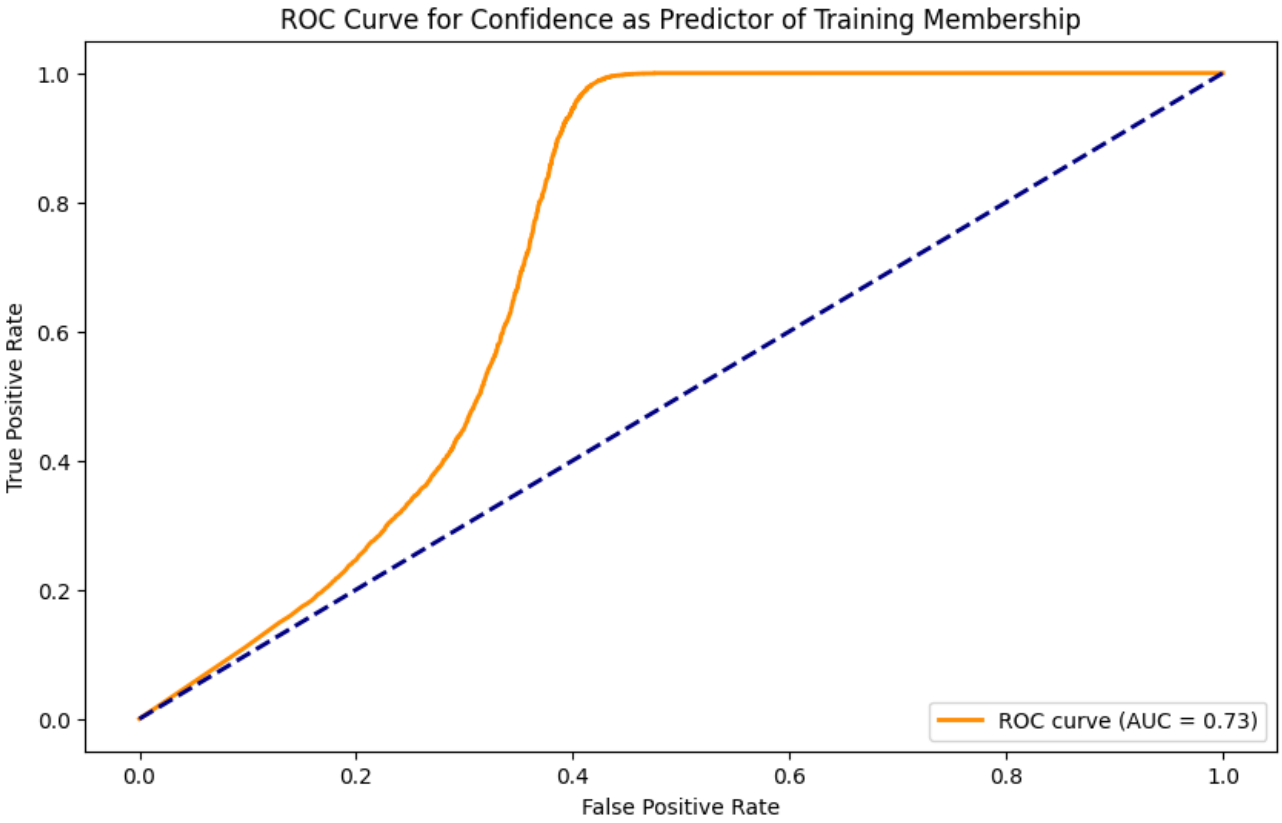

Figure 12: ROC curve for Model 1

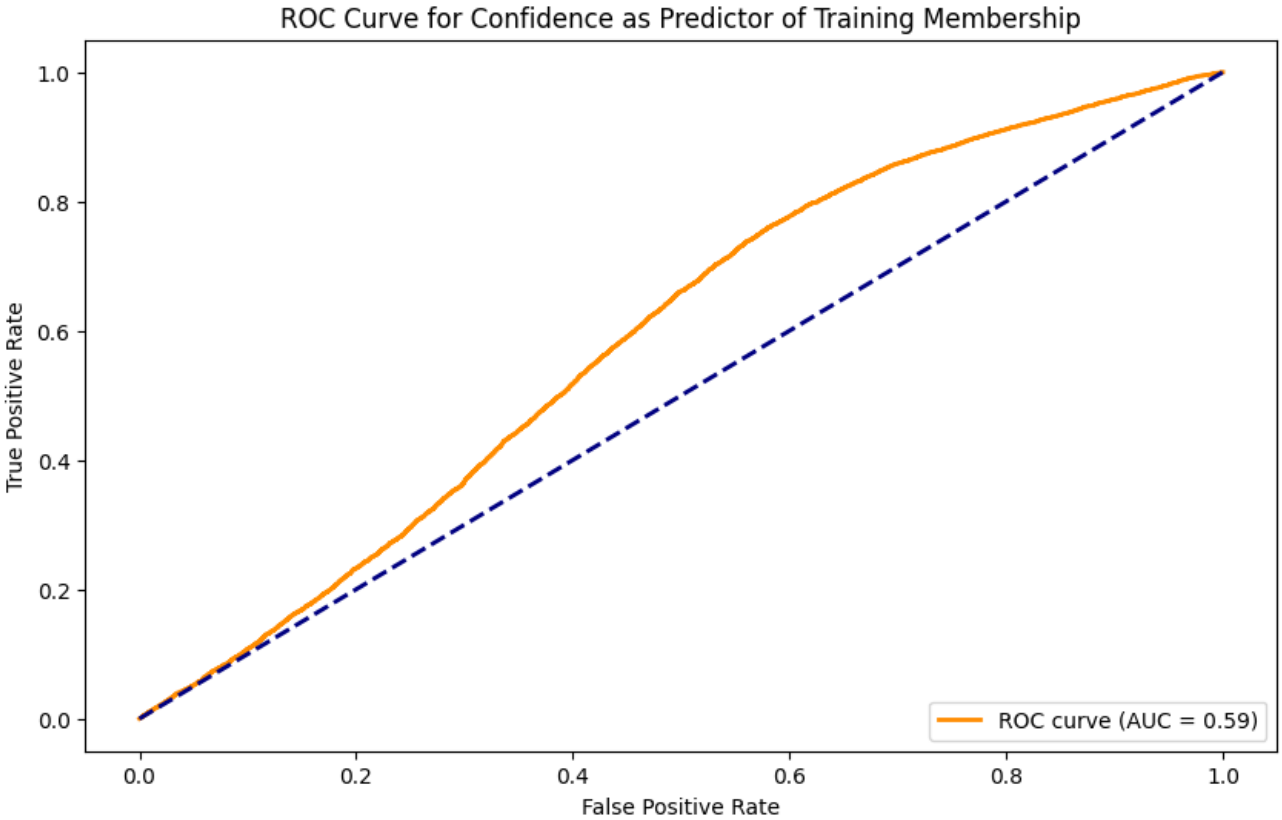

Figure 13: ROC curve for Model 2

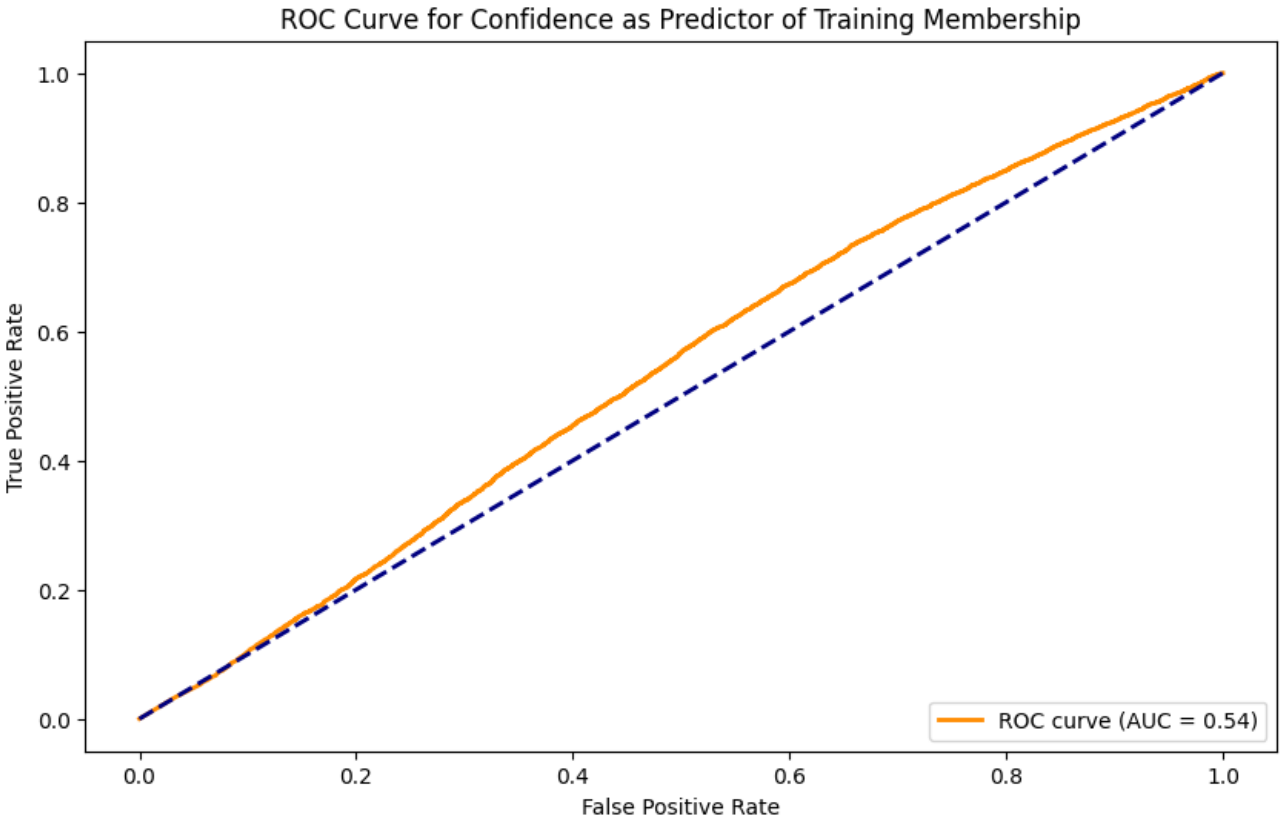

Figure 14: ROC curve for Model 3

As seen in figures Figure 12, Figure 13 and Figure 14, the area under the graph widely changes. The area under the graph decreases as through the models with the baseline model having a curve closer to the y-axis while model 3 has a curve closer to the diagonal. The curve of model 2, has an area under the graph in between those of model 1 and model 3 as expected.

5.2 CIFAR-100

Similar to CIFAR-10, 3 models were trained to evaluate the performance and impact of successful membership inference attacks. Primary performance metric of a ML model is its accuracy. The accuracy of the 3 models are included below. This simplistic architecture on CIFAR-100 inherently struggles with accurate classification, particularly on non-training images. The non-training accuracy is approximately 30.99%, which is notably low. This poor performance highlights the inadequacy of the chosen architecture for CIFAR-100 classification. However, from a membership inference perspective, the goal is not to excel at the classification task but to observe how differences in confidence scores can leak training set membership. Unlike the baseline model without early stopping, the fifth model achieves a slightly better accuracy on non-training data, around 35.66%, which is still low for CIFAR-100 but comparatively better than the previous run. The introduction of early stopping thus prevents extreme overfitting and marginally improves generalization, even though the model's architecture remains weak and unsuitable for optimal performance on CIFAR-100. The sixth model achieves a

non-training accuracy of 38.02%, an improvement over the previous models but still suboptimal for CIFAR-100 classification.

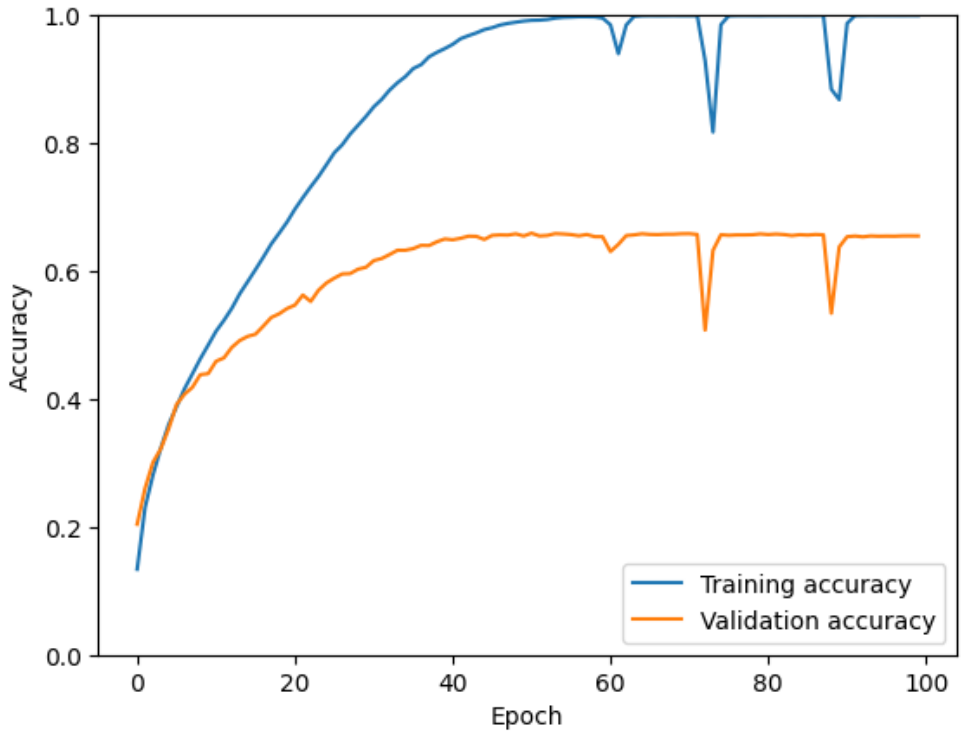

Figure 15: Accuracy for model 4

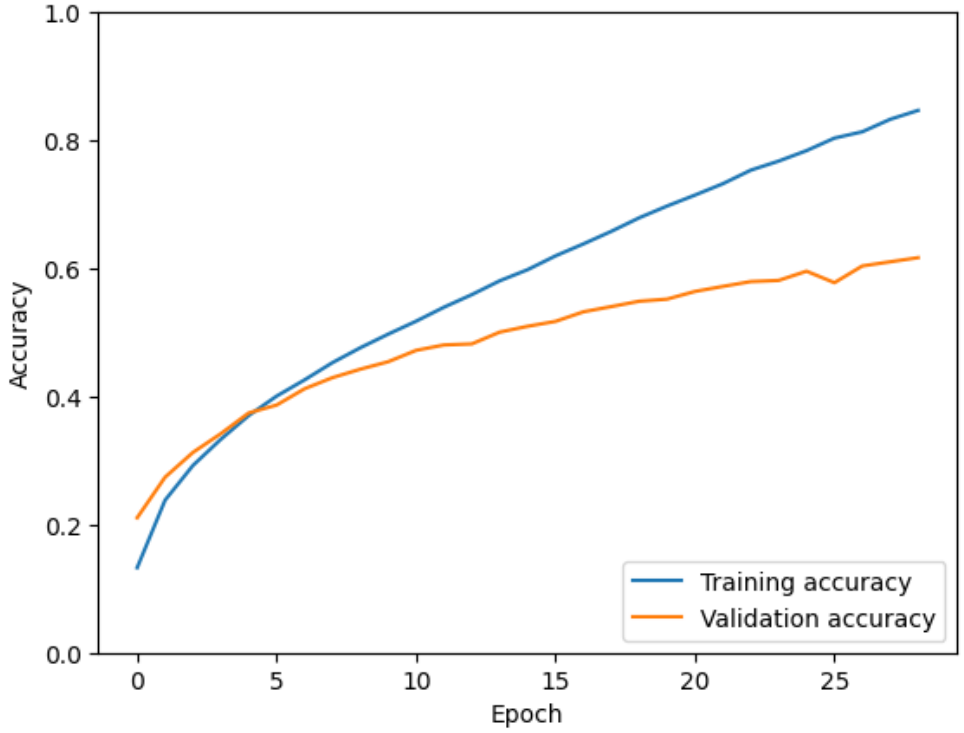

Figure 16: Accuracy for model 5

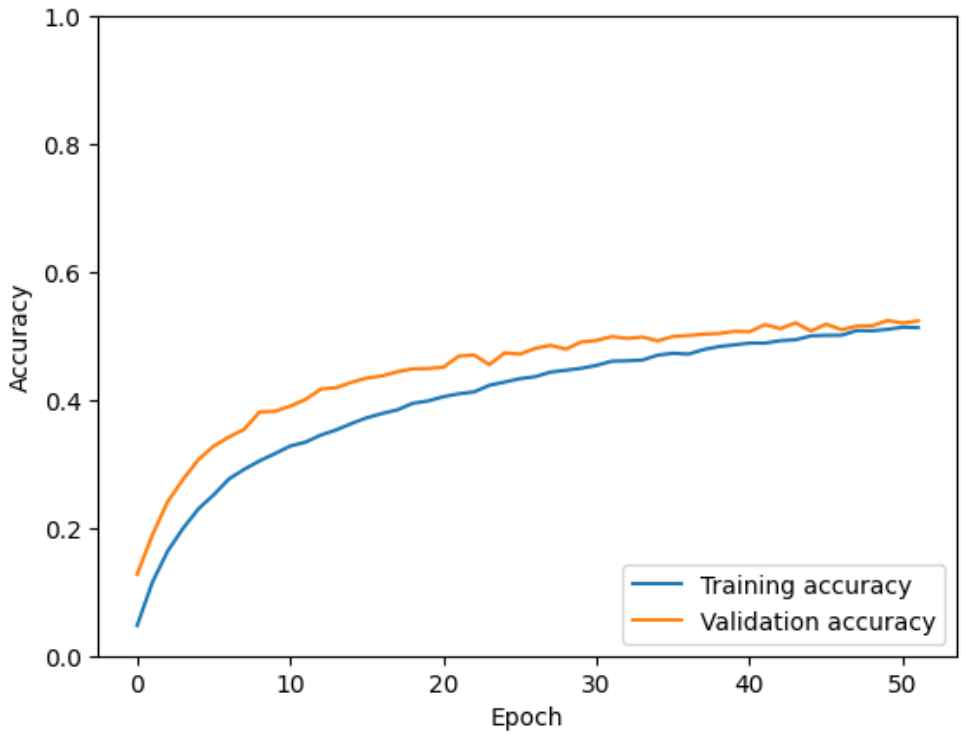

Figure 17: Accuracy for model 6

However, the model 5's poor classification accuracy on non-training images and the relatively modest drop in AUC highlight a key insight: while early stopping alone can help reduce membership inference vulnerability, it is not sufficient, especially when the underlying architecture is not well-suited to the complexity of CIFAR-100. This sets the stage for further enhancements in the subsequent models, where more substantial architectural changes and additional regularization techniques may further improve both performance and privacy protection.

Similar to the observations in the predictions of the ML models associated with CIFAR-10 datasets, the models trained over the CIFAR-100 datasets also proved to show significant differences in the confidence with which the models provided predictions for training and testing datasets. Figure 18 shows the difference in predictions of model 5's associated datasets. While the low accuracy could be the reason, the graph shows that the training datasets were all predicted with a confidence score of 0.95 which indicates extreme overfitting. On the other hand, the testing datasets, had a comparatively uniform distribution across the different confidence scores.

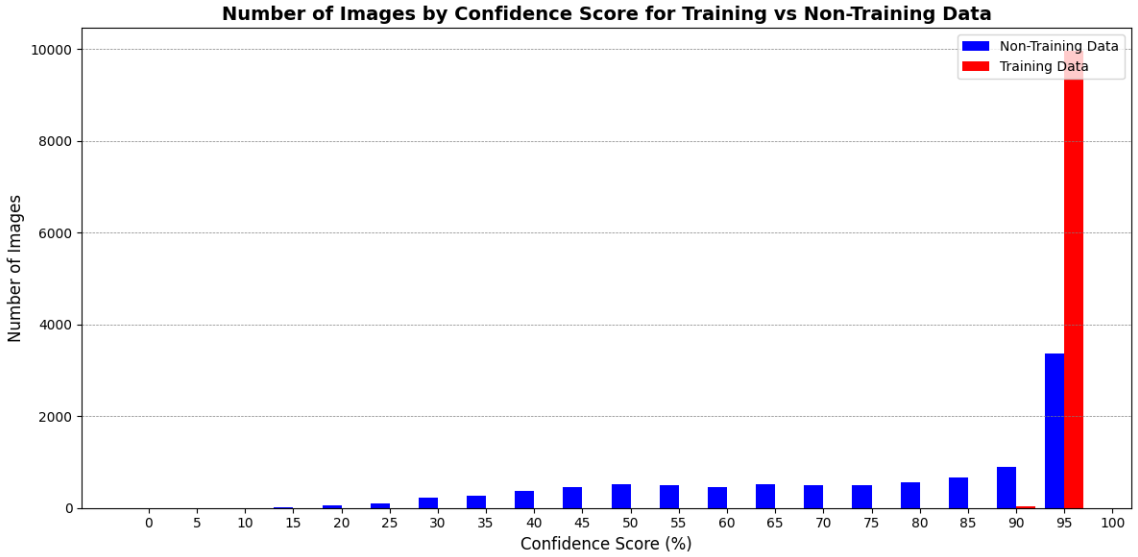

Figure 18: Confidence with which model 4 predicted its training and testing datasets

While early stopping was introduced in model 5 to prevent overfitting, a stark difference in the predictions associated with the training and testing datasets were observed as seen in Figure 19. While testing datasets occurred more often than the corresponding training datasets between the confidence scores of 0.15 and 0.40. An opposite observation, where the training datasets were observed more frequently as compared to testing counterparts, between the confidence scores of 0.75 and 0.95.

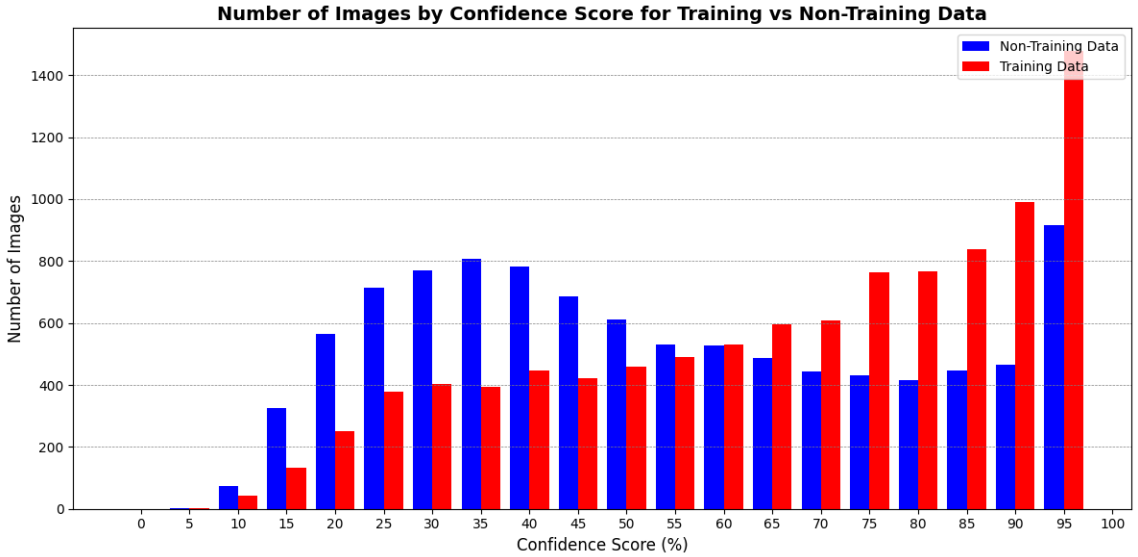

Figure 19: Confidence with which model 5 predicted its training and testing datasets

In model 6, additional measures including dropouts were included on top of the early stopping measure to further curb overfitting. The model's confidence scores provided promising results, w.r.t to the level of overfitting as can be observed in Figure 20. Both the training and testing datasets were more uniformly distributed across the difference confidence scores. The area between the 2 graphs were very close to each other, hence indicating

that the model performs comparatively similar across both the datasets involved.

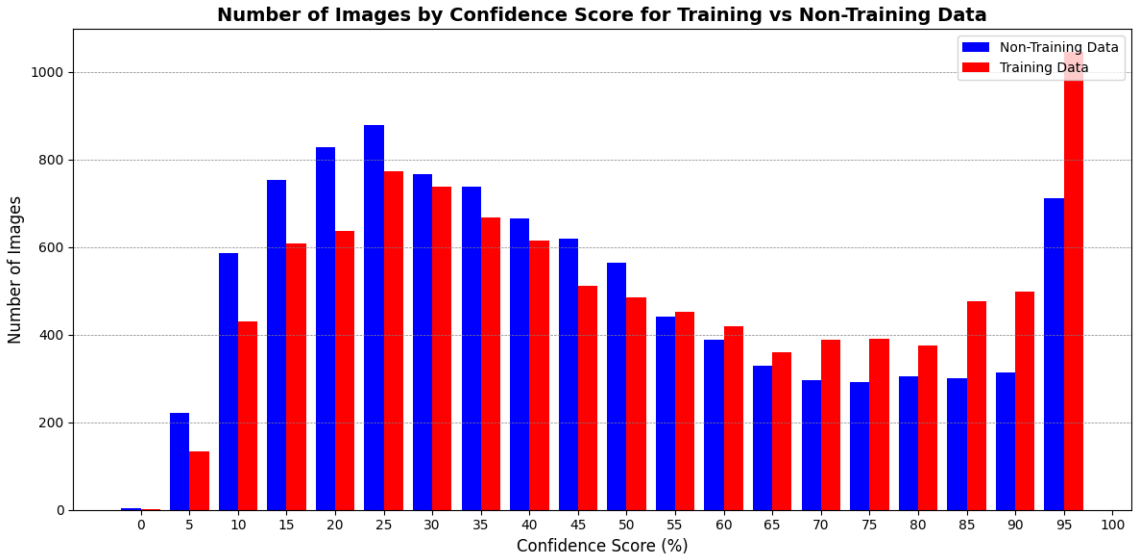

Figure 20: Confidence with which model 6 predicted its training and testing datasets

The ROC curves were constructed for models 4, 5 and 6 based on the manually assigned label 'membership_flag'. The label performed similar to the label created for the models associated with CIFAR-10 dataset. The label held a value of '1' for training datasets, while testing datasets had a value of '0' under this label. The ROC curves associated with the models associated with CIFAR-100 datasets are visualized below. ROC curves for models 4, 5 and 6 are visualized in Figure 21, Figure 16 and Figure 17 respectively.

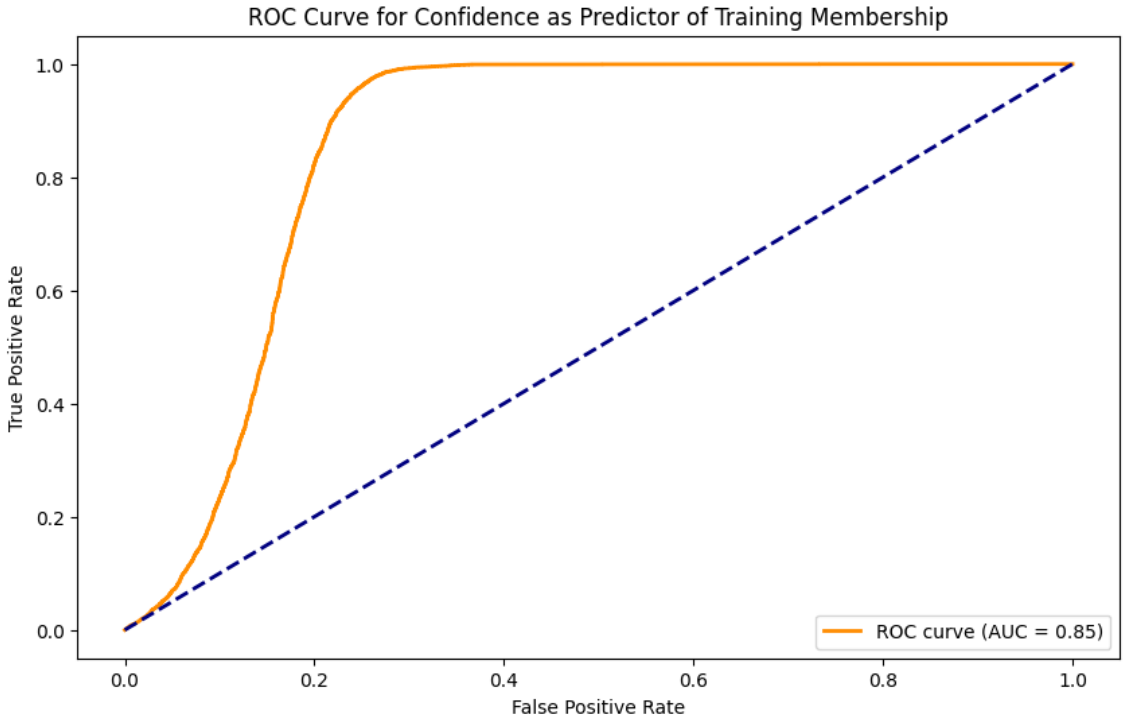

Figure 21: ROC for model 4

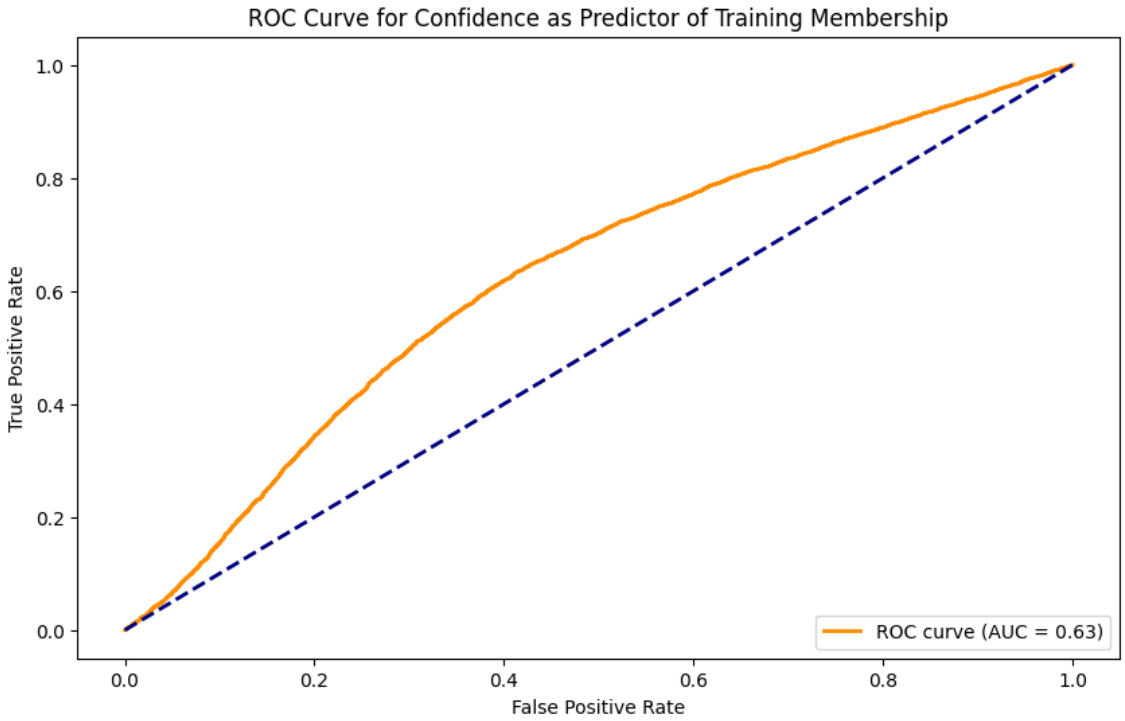

Figure 22: ROC for model 5

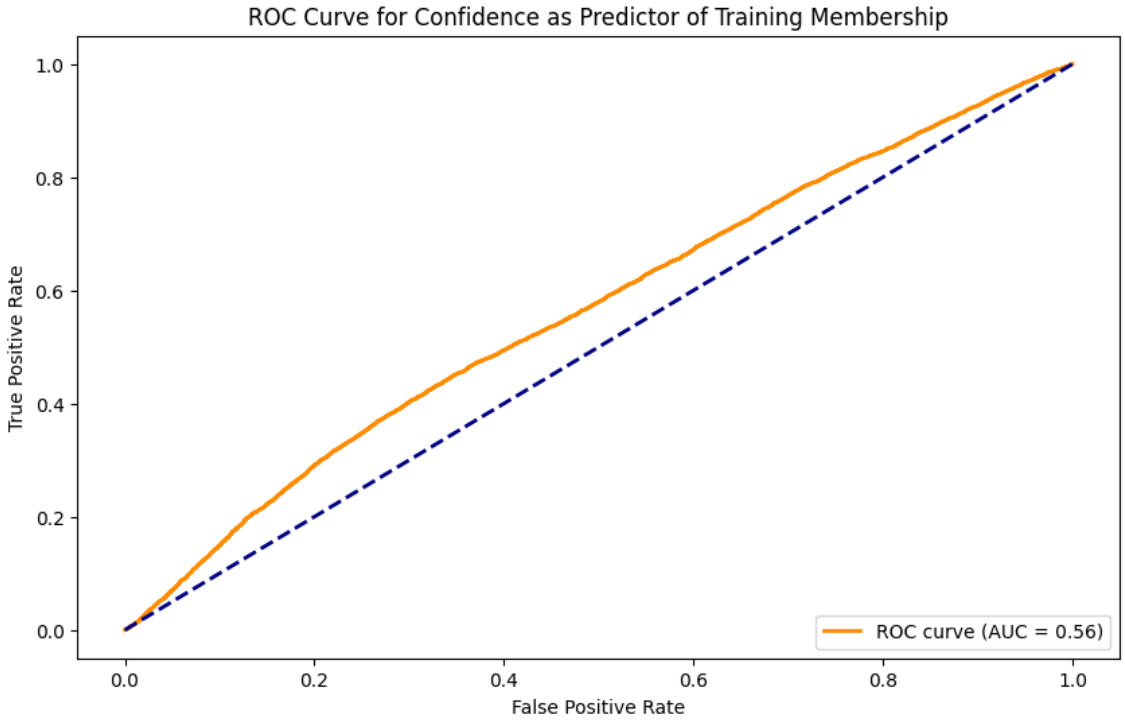

Figure 23: ROC for model 6

5.3 Wide-ResNet

The ML models, Models 4, 5 and 6, trained to work on classifying the CIFAR-100 dataset proved to be less accurate, and therefore not ideal in studying the occurrence and impact of a successful membership inference

attack. In order to increase the quality of study two other models were created based on the Wide-ResNet architectures, model 7 and model 8. While model 7 was a simplistic baseline model based on the Wide-ResNet architecture, model 8 employed measures to curb the overfitting vulnerability as discussed in section 5. The accuracy of these models are visualized below with Figure 24 representing the accuracy of model 7 and Figure 25 representing the accuracy of model 8.

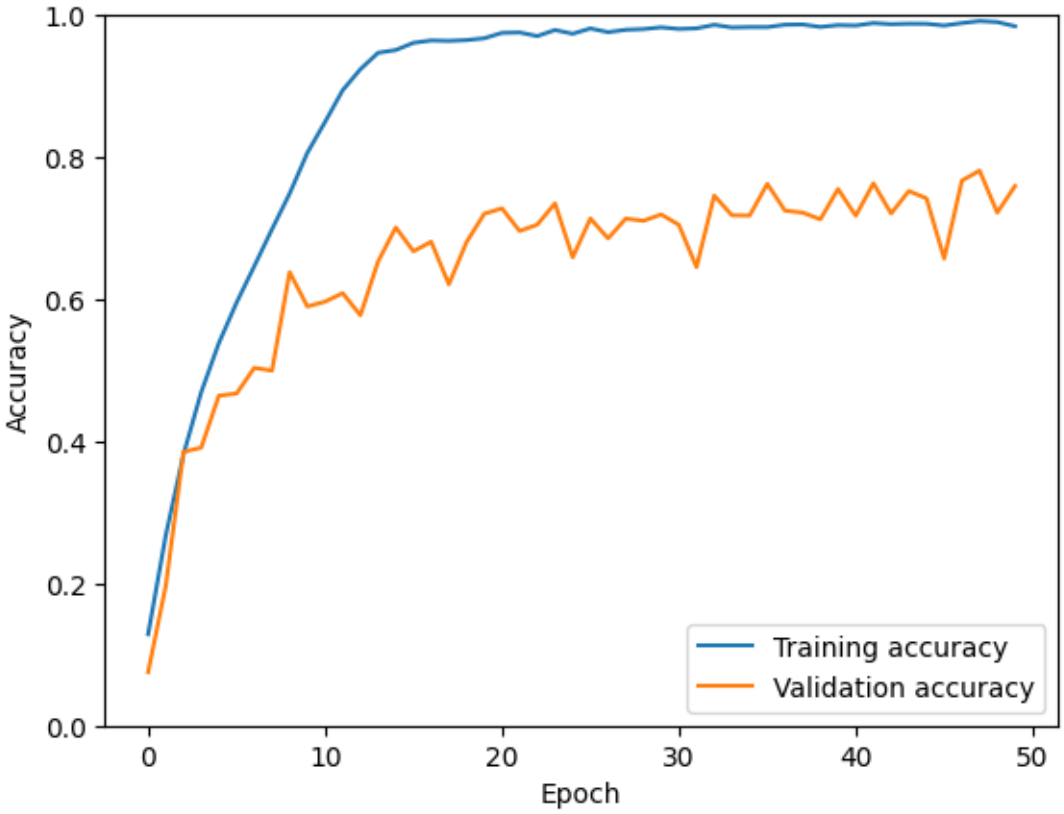

Figure 24: Accuracy for model 7

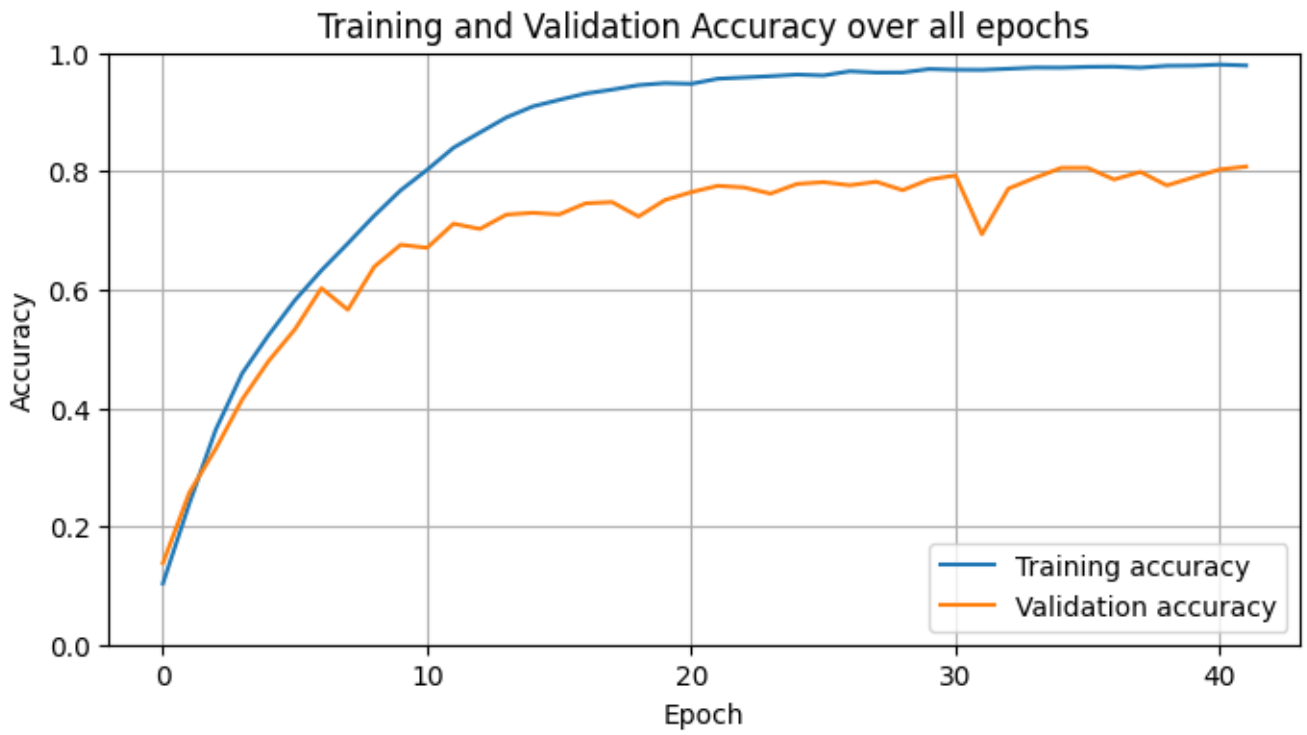

Figure 25: Accuracy for model 8

Adopting the Wide-ResNet architecture results in improved classification performance on non-training data, reaching about 56.57% accuracy—significantly higher than previous models that struggled to surpass 40%. The higher accuracy on non-training images indicates that model 7 generalizes better than the previous models, yet the membership inference attack can still glean information about which samples the model has seen during training. This suggests that while better architectures improve overall classification capability and sometimes reduce the most blatant forms of overfitting, they do not inherently guarantee robust resistance to membership inference attacks. Confidence-based attacks can still exploit subtle differences in prediction patterns between member and non-member samples, especially when no explicit privacy-preserving measures (like differential privacy or label smoothing) are employed.

Meanwhile, Model 8 achieves about 58.05% accuracy on non-training images, an improvement over previous simpler architectures. This suggests that the Wide-ResNet, combined with a well-planned training strategy, can better handle the complexity of CIFAR-100.

Similar to the previous experiments, from model 1 to 6, a graph between confidence score predictions and number of data items were plotted. This was created as to see if the models 7 and 8 performed differently on training and non training datasets similar to its predecessors.

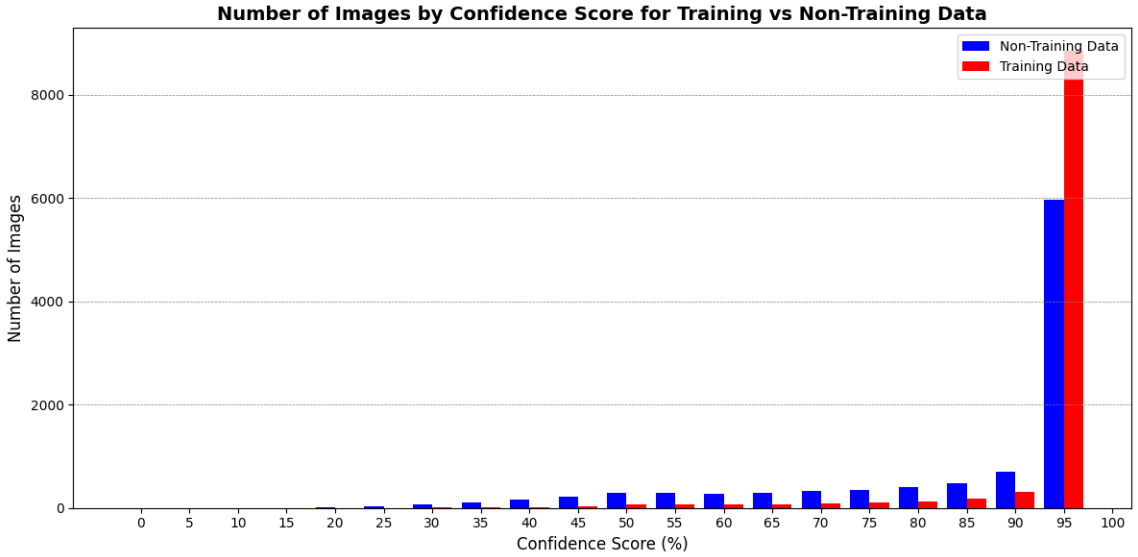

Figure 26: Confidence with which model 7 predicted its training and testing datasets

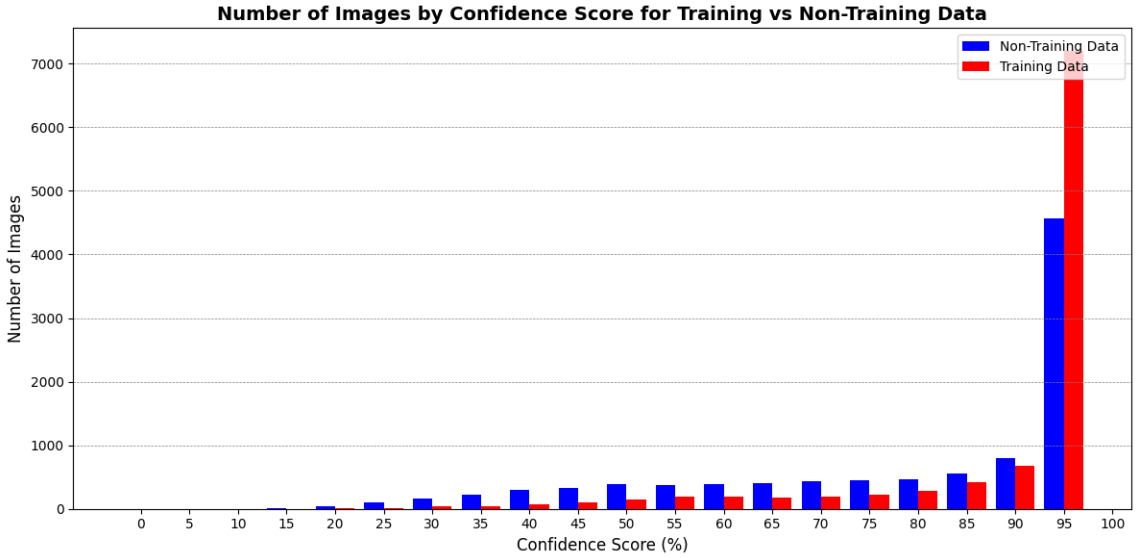

Figure 27: Confidence with which model 8 predicted its training and testing datasets

Figure 26 clearly indicates that there is a disparity in how the model works on training and testing datasets. While testing datasets seem to occur more frequently between confidence scores 0.35 and 0.90, training datasets have a higher probability than their testing counterparts at the 0.95 confidence score. Similar observations can be made from Figure 27, with testing datasets having higher area under graph as compared to the training datasets until a confidence score of 0.75. Trends similar to model 7, are evident at 0.95.

To maintain consistency while performing experiments, the models trained with the Wide-ResNet architecture where subjected to a ROC curve. In line with the previous experiments, 'membership_flag' were annotated to each individual data item, regardless of it belonging to the training or testing dataset. A value of '1' held

under this label suggests a training dataset while a value of '0' is used to identify the testing dataset. ROC curves constructed, to understand the possibility of privacy leaks, are visualized for both models 7 and 8 using Figure 28 and Figure 29. Model 7 has an AUC of 0.72 indicating high privacy leak while model 8 has an AUC of 0.66, which is comparatively lower.

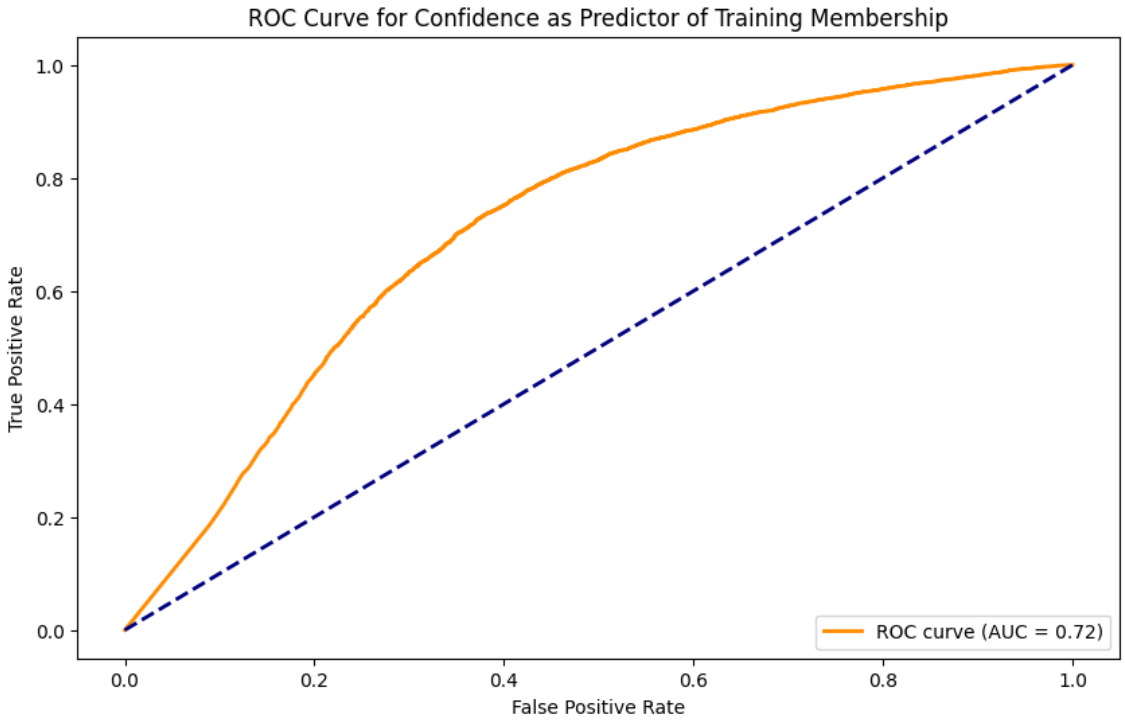

Figure 28: ROC curve for model 7

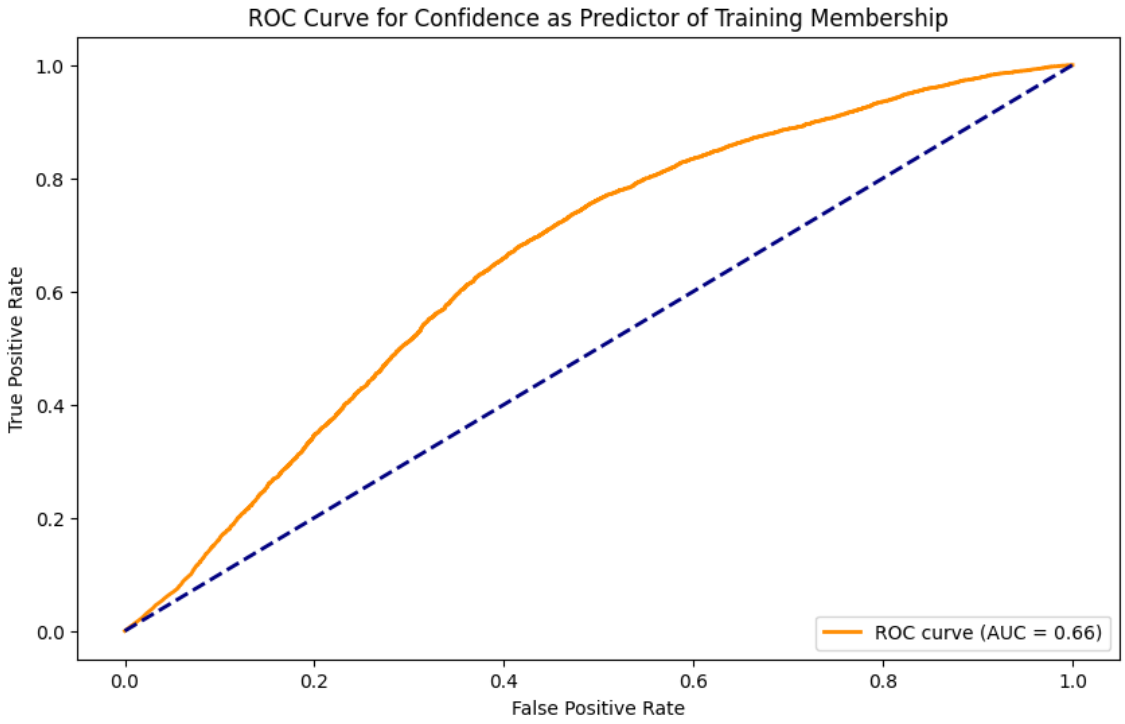

Figure 29: ROC curve for model 8

From a membership inference perspective, the attack's AUC for model 8 is 0.66. Although this AUC is higher than desired from a privacy standpoint, it still reflects a reduction from the highest vulnerability observed with simpler, overfitted models (AUCs around 0.85). This final configuration strikes a balance: the model is neither trivially overfit nor completely impervious to attacks. Instead, it demonstrates that even with a more capable architecture, membership inference remains a concern unless specific privacy-preserving strategies are employed.

This final model integrates architectural sophistication (Wide-ResNet), dropout-based regularization, a two-phase training approach, and early stopping. These measures collectively enhance the model's performance on non-training samples and moderately reduce membership inference vulnerability compared to earlier attempts. However, the persistent success of membership inference attacks, as indicated by the AUC of 0.66, highlights that improving architectures and training protocols alone may not suffice. True privacy resilience may require more targeted defenses, such as differential privacy, adversarial training, or other privacy-preserving mechanisms, underscoring the complexity of balancing performance and privacy in modern deep learning models.

5.4 Shadow Model

The results show the training loss decreases consistently from 0.6872 to 0.5994, while the accuracy improves from 55.% to 67.38%. This steady increase indicates that the model is learning and generalizing without overfitting, as the improvements are gradual and stable. However, the final accuracy of 67.38% highlights that the model is still limited in fully distinguishing between members and non-members.

The ROC curve provides further insight into the model's performance. The AUC score of 0.73 indicates that the attack model performs significantly better than random guessing (AUC = 0.5) and has moderate discriminatory power. The curve lies well above the diagonal baseline, meaning the model can achieve a good trade-off between the true positive rate (TPR) and the false positive rate (FPR) across various thresholds. This suggests that the attack model successfully captures some patterns in the features extracted from the shadow model.

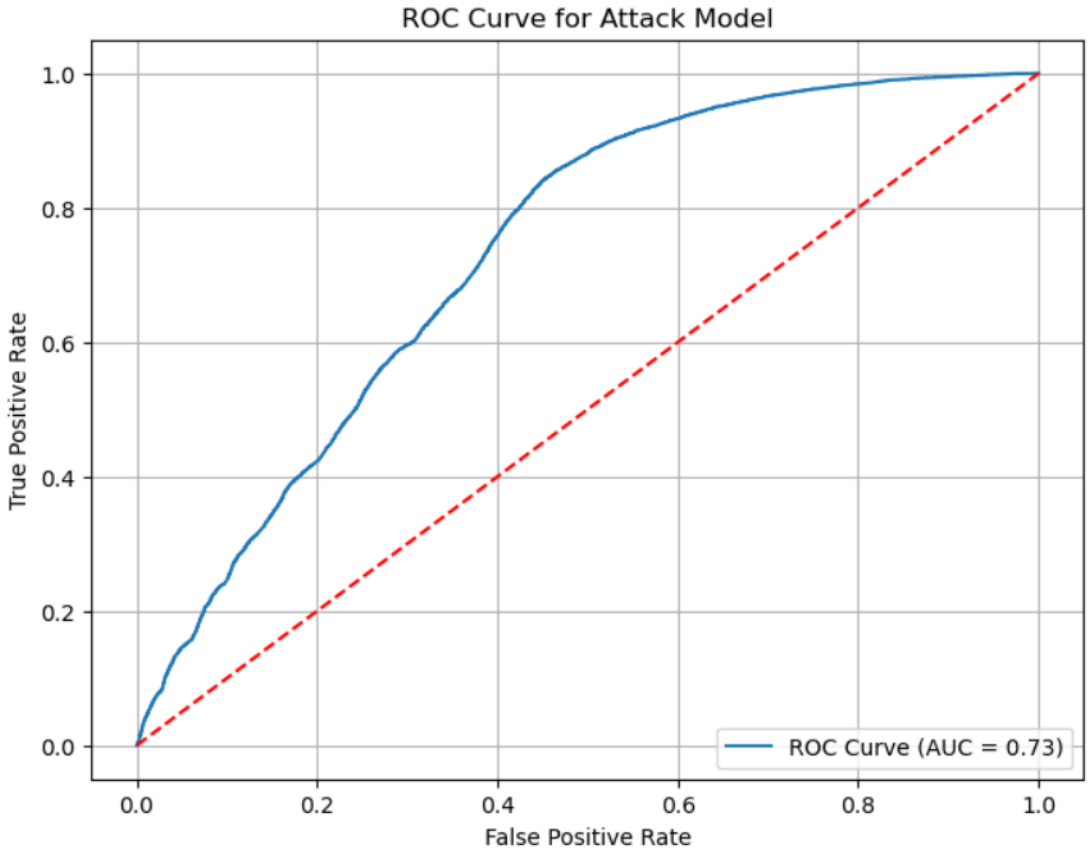

Figure 30: ROC Curve for the Attack Model (AUC = 0.73).

While the attack model shows promising performance, as reflected by the AUC score and accuracy trends, there is still room for improvement. Future work could focus on refining the extracted features, such as using intermediate layer activations, or increasing the complexity of the attack model architecture to enhance its ability to identify membership-related information more effectively. Discussion This section discusses the

results found from the previous section to provide further insights on our problem statement.

5.5 CIFAR-10

As seen in the section 5, the confidence with which the models predicts training and non-training datasets, associated with them, is quite different. This change in confidence score can be used to identify the input data's underlying hidden attributes, i.e. data record which is to be tested if it forms part of the training dataset. Therefore this difference in behavior plays a crucial role in determining if the models are susceptible to privacy leaks.

For instance, in Figure 6 all training datasets have a minimum confidence score of 0.9, indicating that the model's high certainty in predictions for the training samples. This confidence is noticeably distinct from the lower scores typically assigned to non-training samples. This indicates that any input to the model which has an output with a confidence score less than 0.9 is more likely to not be part of the training dataset hence proving non-membership. Similarly the probability of a data record with confidence score of 0.95 is comparatively higher for training samples as compared to the non-training samples. This indicates that any input sample with highest possible confidence score has higher chances of belonging to the training samples. Similarly in Figure 7, it is noticeable that between the confidence score of 0.2 and 0.6, the area under the non-training dataset is higher, therefore any input that receives an output with a confidence score in this range is more likely to not be a member of the dataset used to train this model. The overlap of the area under graphs does reduce the reliability of confidence based membership inference. Although there is still some capability of predicting the membership of a data record in this model, the conviction of the success of a membership inference attack is relatively low. The conviction further decreases with model 3 as seen in Figure 8, as the area under the graphs are a lot more similar. Model 3 exhibits more uniform confidence score distribution across the training and non-training samples. The reduced divergence in the confidence scores between training and non-training samples significantly diminishes the effectiveness of a membership inference attack. This behavior indicates that the model 3 has employed privacy preserving techniques, thereby being less susceptible to membership inference attacks or privacy leaks.

The loss function of ML models trained on CIFAR-10 dataset also provides ample insight to our problem statement. The difference in loss function across the training and non-training datasets play a crucial role in identifying how well a model generalizes its patterns. A small difference between these loss functions indicate that the model is able to apply its learned patterns, from the training dataset, onto the non-training dataset efficiently. This indicates the model is less overfitted as is the case of Model 3 Figure 11. On the other hand, if the training loss is much lower than the testing loss, indicates that the model has memorized the training dataset and that its learned patterns cannot reap comparable results on the training and non-training data samples. This is usually the result of insufficient regularization techniques. A large difference indicates that the models are extremely

overfitted. This is similar to our findings associated with Model 1 Figure 9 and Model 2 Figure 10. In addition to this, visualizing validation loss across training and non-training datasets also helps in identifying when model starts to overfit. It points to a specific epoch beyond which the validation loss starts decreasing with the training dataset while validation loss across the non-training samples continue to increase.

In **??**, the relation between the success rate of the membership inference attack and the ROC curves of the respective models can be inferred. The ROC curve provides insights into the trade-off between the True Positives and False Positives for membership prediction. The success rate of the membership inference attack depends on how close the ROC curves are either to the diagonal or to the y-axis. The closer the ROC curve is to the y-axis, the higher the possibility of a privacy leak as observed in Figure 12. In model 2 Figure 13 the ROC curve shifts closer to the diagonal. This indicates to decrease in the model's susceptibility to membership inference attacks. This is in line with the measures implemented during the training phase of model 2, wherein the early stopping mechanism was used to prevent the model from learning unnecessary patterns for making predictions. In model 3, the privacy vulnerability were further tackled using a combination of early stopping and dropout mechanisms. As expected, the ROC curve of model 3 shifted closer to the diagonal indicating random guessing as to whether a given input belongs to the training sample or not.

This system also addresses the often debated question of how security and performance coexist without compromise. It can be seen that the validation accuracy of these graphs is fairly close to each other as seen in Figure 3, Figure 4 and Figure 5 whilst the efficiency with these membership inference attacks can be made its associated impact reduce across the models. None of the measures implemented to tackle the vulnerability of overfitting affected how the model performs in a significant manner. The results suggest that ML models can achieve robustness against privacy attacks without having to compromise on their associated performance characteristics such as its validation accuracy.

5.6 CIFAR-100

Similar trends to the CIFAR-10 experiments can be observed in the CIFAR-100 models. As previously noted, membership inference attacks leverage the model's confidence levels to distinguish training data samples from those not included during training. With the CIFAR-100 models, we see that even though the classification task is more complex—given the larger number of classes—the underlying principle remains the same: overfitted models tend to assign disproportionately high confidence scores to training examples, making them more vulnerable to membership inference attacks.

In the early CIFAR-100 models (Model 1, Model 2, and Model 3), we observed that simplistic architectures

and insufficient regularization led to a pronounced difference in confidence distributions for training and non-training samples. For instance, in Model 1, the model frequently assigned very high confidence to training samples while giving significantly lower confidence to non-training inputs. This disparity closely mirrors the behavior seen with CIFAR-10 models, indicating an increased susceptibility to membership inference attacks. The attacker could relatively easily identify membership by simply checking if the model's predicted confidence exceeded a particular threshold.

As we introduced measures like early stopping and dropout in Model 2 and Model 3, the gap between confidence scores for training and non-training samples narrowed. Although the improvements were not as drastic in absolute terms as in CIFAR-10—given the inherently more challenging CIFAR-100 classification task—the direction remained consistent. Enhanced regularization and better training regimes produced more uniform confidence distributions, making it harder for an adversary to deduce membership status. Hence, these models demonstrated reduced vulnerability, reflecting that the principles of improving generalization and avoiding overfitting are broadly applicable across different datasets and complexities.

One notable point in CIFAR-100 models is the interplay between model complexity and inference vulnerability. The introduction of Wide-ResNet architectures in the final models improved non-training accuracy and overall model performance. Still, high performance did not equate to guaranteed privacy protection. While a more capable architecture helped the model generalize better and avoid extreme confidence disparities, the membership inference attacks were not fully neutralized. Instead, the attacks became more subtle, with AUC values indicating a moderated but persistent risk of privacy leakage. This suggests that complexity and better performance alone do not inherently guarantee privacy. Rather, performance improvements need to be combined with intentional privacy-preserving strategies—such as well-tuned dropout, early stopping, or other advanced techniques—to effectively reduce membership inference susceptibility.

Additionally, as with CIFAR-10, monitoring loss differences between training and non-training datasets in CIFAR-100 provided valuable insights. Models that demonstrated smaller gaps between training and validation loss (particularly after introducing regularization measures) were generally less prone to membership inference attacks. This correlation between loss dynamics and privacy risk reiterates the importance of generalization. When a model evenly applies learned patterns to both training and unseen data, it becomes more challenging for an attacker to distinguish membership based on confidence alone. In contrast, models that displayed pronounced differences in loss between training and test sets tended to overfit, offering clear signals to attackers.

Analyzing the ROC curves of the membership inference attacks further reinforces these conclusions. The closer the ROC curve stays to the diagonal, the more the membership inference essentially boils down to random

guessing. In the early CIFAR-100 models, the ROC curves leaned away from the diagonal, demonstrating meaningful predictive power for the attacker. As incremental improvements—like early stopping and dropout—were incorporated, the ROC curves moved closer to the diagonal, reducing the attacker's advantage and approaching a scenario where membership inference attacks are less certain and more approximate.

Lastly, it is worth noting that none of the measures taken to reduce vulnerability—be it early stopping, dropout, or a more sophisticated architecture—significantly degraded the model's classification performance. This parallels the findings with CIFAR-10: it is possible to enhance privacy robustness without sacrificing validation accuracy. While CIFAR-100 remains a tougher dataset, the fundamental message stands: careful training practices and architectural choices can strike a balance between performance and privacy resilience.

In summary, the CIFAR-100 models validate and extend the lessons learned from the CIFAR-10 experiments. Better generalization and more nuanced regularization strategies mitigate membership inference risks, even in more complex classification scenarios. Although perfect privacy protection is not achieved, the adjustments made—from introducing dropout and early stopping to employing more advanced architectures—collectively reduce the attack's effectiveness while maintaining strong predictive capabilities.

5.7 Shadow Model

The effectiveness of membership inference attacks depend on the performance and characteristics of the shadow model. In our version of a shadow model, where the attack model reached a final accuracy of 67.38% and an AUC of 0.73, the attack model shows a moderate success in distinguishing membership status. This performance can be attributed to the shadow model's outputs, which likely contained subtle but exploitable differences between training (member) and validation (non-member) data. These discrepancies suggest that as the shadow model begins to overfit or memorize parts of the training set, membership inference becomes more feasible, exposing a measurable degree of privacy leakage.

The more a shadow model overfits, the more vulnerable it becomes to membership inference attacks. Overfitting increases the divergence between member and non-member outputs, providing exploitable patterns for the attack model. On the other hand, when a shadow model maintains strong generalization, it mitigates the risk of privacy leakage by producing more consistent outputs across both seen and unseen data.

As a final point, models should be judged based on their ability to protect against privacy breaches, not just on their predictive capabilities. Future studies could investigate ways to balance performance and privacy, such as using differential privacy, stronger regularization methods, or adversarial training.

5.8 Limitations

There are some challenges or constraints that our study faced and those can affect its applicability or interpretation. Those include:

1. **Synthetic Experimental setup:** In our model, we introduced controlled overlap between training and testing datasets, which might not accurately reflect real world situations. We tried to mimic the situation where the attacker has some real data that was used in the training, which might oversimplify or exaggerate the risks associated with membership inference attacks in real world applications. In real world scenarios, such overlaps are done unintentionally or with malicious intent. This scenario makes it less predictable than the controlled overlaps like in our model. The assumption of overlap could lead to overgeneralized conclusions about model vulnerability. Also, the setup may fail to capture nuances of more complex attack scenarios where the attacker has limited knowledge of the training dataset.
2. **Scalability and Resource constraints:** Our study were conducted with the models trained and tested on small datasets like CIFAR-10, which might work well in experiments, but they may not perform efficiently when scaled to production datasets, which are often much larger and more complex. A production system might need to handle millions of user requests daily, far beyond what a simple experimental setup can simulate. Moreover, deploying models in production requires high computational resources and regularization techniques like dropout or complex architectures can increase the computational cost, potentially making the model slower or more expensive to run in real time. For example, a model designed for experimentation might take seconds per prediction, but in production, a model needs to provide predictions in milliseconds for real-time applications.
3. **Lack of Robustness Testing:** This study utilizes specific regularization techniques but did not evaluate the robustness under varying conditions, such as imbalanced data or adversarial environments. We did not include how these techniques handle different types of datasets, such as highly imbalanced data or noisy datasets. This can cause a struggle to apply our study to datasets with more complex characteristics or to defend against advanced attacks beyond membership inference.

5.9 Ethical Considerations

This section focus on the risks, responsibilities, and precautions researchers must be aware of when conducting experiments, especially involving sensitive data like in our study.

1. **Bias and Fairness:** Machine learning models often reflect biases in training data, which the dataset and model may unintentionally favor certain groups and put others at higher risk of privacy breaches. Also, models might generalize poorly for underrepresented groups, leading to biased outcomes in both model predictions and vulnerability to attacks. For example, in a demographic dataset, minority groups may have unique, easily identifiable patterns that make them more vulnerable to membership inference

attacks. Unequal risk of privacy breaches could lead to harm against vulnerable groups. Therefore, when planning the training, researchers shall evaluate datasets for representation and ensure that their privacy preserving strategies are equitable across all groups.

2. **Data privacy and security:** Even though the datasets we used in our study are public and not sensitive, using sensitive datasets in similar experiments from real world, such as healthcare data, could lead to ethical issues if they are improperly handled. Data breaches or leaks during the experiment could expose personal or confidential information.

5.10 Future works

This report experimentally compared three different models with different regularization techniques in CNN with CIFAR-10 and CIFAR-100 dataset. However, there are still multiple avenues to study and further improve privacy in machine learning. In this section, we discuss different directions we can take to enhance privacy in this domain.

1. **Testing varied dataset:** This report focuses on images dataset, testing on a variety of data provides us information regarding robustness of privacy techniques since the membership inference attack can vary depending on the type of dataset.
2. **Privacy preserving techniques:** Federated learning, differential privacy are widely accepted privacy preserving concepts. Further study in these concepts and producing state-of-art-work can greatly serve in preserving privacy in machine learning.
3. **User centric privacy and transparency tools:** With the increasing concern over data privacy, research and development on user centric privacy preserving tools can empower end users in deciding how they want their data to be collected, processed and stored.
4. **Governance and Regularization (GDPR) in preserving data:** While the GDPR has been serving as one size fits across different domains. There is a growing demand for regulation that cater to specific domains such as healthcare, finance, energy with particular focus on privacy. It is required to regulate and monitor data privacy, making it a significant area for future work.

6 Conclusion

We simulated three membership inference attacks with different methods of training to see how we can reduce the possibility of membership inference attacks using CIFAR-10. First model was designed without regularization techniques and early stopping to encourage overfitting. It is easier for the attackers to find member sample with confidence levels in this model. For the second model, we adopted early stopping during the training process to prevent overfitting by limiting the model's ability to memorize the training data. Third model had changes both in the model's structure and training process. We added third convolutional layer to improve generalizing ability beyond the training data, and we improved activation functions to LeakyReLU and added Dropout layer to reduce overfitting. In training process, we increased the epochs of early stopping from the second model. These improvements reduced the confidence gap between member and non-member samples which makes the attacker more difficult to perform membership inference attack. With ROC curve for each model, we could see that the third model has an AUC of 0.54, which is close to the value of complete resistance to the attack. However, even with the improvements, the possibilities of the attack still remain.

After, we tested those three models with CIFAR-100 which has 100 classes with more specified classes but the same number of images as CIFAR-10. The accuracy was lower than the previous dataset, however the good model still managed to reduce privacy leakage. For more sophisticated model, we adopted Wide-ResNet architecture. We built two models using Wide-ResNet, the baseline model and the final model which also includes dropout, two phase training and early stopping. Wide-ResNet has improved the classification performance on non-training data with higher accuracy than the previous models, but AUCs were higher.

The use of ResNet-18 for the shadow model and a fully connected attack model demonstrates a practical and realistic adversarial scenario. Through analysis of outputs and the ROC curve, the findings reveal that overfitting in the shadow model significantly increases vulnerability to membership inference attacks. Techniques such as differential privacy, enhanced regularization methods, and adversarial training should play a pivotal role in developing secure and effective machine learning systems.

References

[1] IBM. *The Not So Short A Introduction to LaTeX2e*. https://www.ibm.com/topics/machine-learning.

[2] Reza Shokri et al. "Membership Inference Attacks Against Machine Learning Models". eng. In: *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 3–18. ISBN: 9781509055326.

[3] Zaruhi Aslanyan and Panagiotis Vasilikos. *Privacy-Preserving Machine Learning: A Practical Guide*. Whitepaper. Accessed: 15 December 2024. The Alexandra Institute, Oct. 2020. URL: https://www.alexandra.dk/.

[4] *7 Stages of Machine Learning: A Framework*. Medium article, Accessed: 11/12/2024. 2020. URL: https://medium.com/@datadrivenscience/7-stages-of-machine-learning-a-framework-33d39065e2c9.

[5] Stefanos Karageorgiou. "Automatic Coronary Artery Calcium Classification with Machine Learning". MSc dissertation. University of Edinburgh, School of Mathematics, Aug. 2020.

[6] Johnson Kolluri et al. "Reducing Overfitting Problem in Machine Learning Using Novel L1/4 Regularization Method". In: June 2020, pp. 934–938. DOI: 10.1109/ICOEI48184.2020.9142992.

[7] Nishanth Chandran. "Security and Privacy in Machine Learning". In: *Information Systems Security*. Ed. by Vallipuram Muthukkumarasamy, Sithu D. Sudarsan, and Rudrapatna K. Shyamasundar. Cham: Springer Nature Switzerland, 2023, pp. 229–248. ISBN: 978-3-031-49099-6.

[8] Soumia Zohra El Mestari, Gabriele Lenzini, and Huseyin Demirci. "Preserving data privacy in machine learning systems". In: *Computers Security* 137 (2024), p. 103605. ISSN: 0167-4048. DOI: https://doi.org/10.1016/j.cose.2023.103605. URL: https://www.sciencedirect.com/science/article/pii/S0167404823005151.

[9] Bo Liu et al. "When machine learning meets privacy: A survey and outlook". In: *ACM Computing Surveys (CSUR)* 54.2 (2021), pp. 1–36.

[10] Hongsheng Hu et al. "Membership Inference Attacks on Machine Learning: A Survey". In: 54.11s (2022). ISSN: 0360-0300. DOI: 10.1145/3523273. URL: https://doi.org/10.1145/3523273.

[11] UNCTAD. *Data Protection and Privacy Legislation Worldwide \textbar UNCTAD*. 2024. URL: https://unctad.org/page/data-protection-and-privacy-legislation-worldwide.

[12] Securiti. *Data Privacy Laws and Regulations Around the World*. Accessed: 2024-12-10. 2024. URL: https://securiti.ai/privacy-laws/.

[13] GDPR. *Data Privacy*. Accessed: 2024-12-11. 2024. URL: https://gdpr.eu/data-privacy.

[14] GDPR.eu. *Art. 35 GDPR – Data protection impact assessment*. Accessed: 2024-12-11. 2024. URL: https://gdpr-info.eu/art-35-gdpr/.

[15] Art.5 GDPR. *Art. 5 GDPR – Principles relating to processing of personal data*. General Data Protection Regulation (GDPR). 2024. URL: https://gdpr-info.eu/art-5-gdpr/ (visited on 12/11/2024).

[16] Mohammad Al-Rubaie and J Morris Chang. "Privacy-preserving machine learning: Threats and solutions". In: *IEEE Security & Privacy* 17.2 (2019), pp. 49–58.

[17] Adnan Qayyum et al. "Securing Machine Learning in the Cloud: A Systematic Review of Cloud Machine Learning Security". In: 3 (Nov. 12, 2020). ISSN: 2624-909X. DOI: 10.3389/fdata.2020.587139. URL: https://www.frontiersin.org/journals/big-data/articles/10.3389/fdata.2020.587139/full (visited on 12/17/2024).

[18] Ahmed Salem et al. In: *arXiv preprint arXiv:1806.01246* (2018).

[19] Nazish Khalid et al. "Privacy-preserving artificial intelligence in healthcare: Techniques and applications". In: *Computers in Biology and Medicine* 158 (2023), p. 106848.

[20] BBC. "DeepMind faces legal action over NHS data use". In: (2021). URL: https://www.bbc.com/news/technology-58761324 (visited on 12/19/2024).

[21] Davey Winder. *The University Of California Pays $1 Million Ransom Following Cyber Attack*. Forbes. Section: Cybersecurity. 2020. URL: https://www.forbes.com/sites/daveywinder/2020/06/29/the-university-of-california-pays-1-million-ransom-following-cyber-attack/ (visited on 12/19/2024).

[22] Adil Hussain Seh et al. "Healthcare Data Breaches: Insights and Implications". In: *Healthcare* 8.2 (May 13, 2020), p. 133. ISSN: 2227-9032. DOI: 10.3390/healthcare8020133. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7349636/ (visited on 12/19/2024).

[23] Longbing Cao. "Ai in finance: challenges, techniques, and opportunities". In: *ACM Computing Surveys (CSUR)* 55.3 (2022), pp. 1–38.

[24] Maanak Gupta et al. "From chatgpt to threatgpt: Impact of generative ai in cybersecurity and privacy". In: *IEEE Access* (2023).

[25] Ahmed Salem et al. *ML-Leaks: Model and Data Independent Membership Inference Attacks and Defenses on Machine Learning Models*. 2018. arXiv: 1806.01246 [cs.CR]. URL: https://arxiv.org/abs/1806.01246.

[26] Sasi Kumar Murakonda and Reza Shokri. "ML Privacy Meter: Aiding Regulatory Compliance by Quantifying the Privacy Risks of Machine Learning". eng. In: (2020).

[27] Xiao Li et al. "On the Privacy Effect of Data Enhancement via the Lens of Memorization". eng. In: *IEEE transactions on information forensics and security* 19 (2024), pp. 4686–4699. ISSN: 1556-6013.

[28] Samuel Yeom et al. “Privacy Risk in Machine Learning: Analyzing the Connection to Overfitting”. eng. In: *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*. Vol. 2018-. IEEE, 2018, pp. 268–282. ISBN: 9781538666807.

[29] Nitish Srivastava et al. “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”. In: *Journal of Machine Learning Research* 15.56 (2014), pp. 1929–1958. URL: http://jmlr.org/papers/v15/srivastava14a.html.

[30] Shi Chen et al. “HP-MIA: A novel membership inference attack scheme for high membership prediction precision”. In: *Computers Security* 136 (2024), p. 103571. ISSN: 0167-4048. DOI: https://doi.org/10.1016/j.cose.2023.103571. URL: https://www.sciencedirect.com/science/article/pii/S0167404823004819.

[31] Nicholas Carlini et al. “Membership Inference Attacks From First Principles”. eng. In: *2022 IEEE Symposium on Security and Privacy (SP)*. Vol. 2022-. IEEE, 2022, pp. 1897–1914. ISBN: 9781665413169.

A Appendix

Listing 1: ML models

```
Model 1 : https://colab.research.google.com/drive/164CA8x2VkQHIJxXLDBie7S0WaPQCDoT1?usp=sharing

Model 2: https://colab.research.google.com/drive/1IcD5swHAQ4WuPXfYEB8rX7Rl_B5jGoVb?usp=sharing

Model 3: https://colab.research.google.com/drive/1CVN8epzQDCTqDdaXHUTenUqzVW7UYThG?usp=sharing

Model 4: https://colab.research.google.com/drive/1IAHvIOhbIgByqqSZBP87-QjxmGbMwIHB?usp=sharing

Model 5:https://colab.research.google.com/drive/1lyyHpj3VdwjCw70naBIyt_gQMfzdtDQx?usp=sharing

Model 6: https://colab.research.google.com/drive/1ngr6t6Ld8GLDh-ARNoFyBzhSGVzMxe1j?usp=sharing

Model 7: https://colab.research.google.com/drive/10XMODPv1Eg8YS8w9VU93MHz7j9hu6Gci?usp=sharing

Model 8 : https://colab.research.google.com/drive/1pVMw18bgwx784Anreo9pkHcXaa1_TAQ7?usp=sharing
```