Leipzig University

Faculty of Mathematics and Computer Science

Institute of Computer Science

Privacy-Preserving Detection of COVID-19 in X-Ray Images

by

Lucas Lange

Master's Thesis presented in

Computer Science

Supervisors:

Maja Schneider

Reviewers:

Prof. Dr. Erhard Rahm

Dr. Christian Martin

Leipzig, 12. January 2022

Abstract

Chest X-rays enable a fast and safe diagnosis of COVID-19 in patients. Applying Machine Learning (ML) methods can support medical professionals by classifying large numbers of images. However, the amount of data needed for training such classifiers poses problems due to clinical data privacy regulations, which present strict limitations on data sharing between hospitals. Specifically, the models resulting from ML are vulnerable to attacks and can compromise data integrity by leaking details about their training data. Privacy-Preserving ML (PPML) offers methods to create private models that satisfy Differential Privacy (DP), enabling the development of medical applications while maintaining patient privacy.

This work aims at investigating the privacy-preserving detection of COVID-19 in X-ray images. The PPML training methods DP-SGD and PATE are matched against non-private training. The private models should mitigate the data leakage threats posed by Membership Inference Attacks (MIAs). However, the inclusion of DP showed no improvements in MIA defense on the COVID-19 detection task. Instead, the non-private models presented the same repelling properties as the private models. Thus, if only the defense against MIAs is of concern, the non-private approach achieving 97.6% classification accuracy is the best choice. Private DP-SGD training for $\varepsilon = 1$ is a more sensible alternative when a theoretical privacy guarantee is needed. The best DP-SGD model reaches 74.1% accuracy. Even though the accuracy-privacy trade-off of 23.5% is significant, the private model performs 0.3% better than related work and keeps much tighter privacy guarantees. Ultimately, the conflicting findings from the additional experiments on the MNIST database, where DP significantly increased MIA defense, indicate that PPML (or DP-SGD) is heavily task-dependent. Thus, research on one dataset might not carry over to others.

Contents

List of Figures

List of Tables

List of Abbreviations

AUC	Area Under the (ROC) Curve
BN	Batch Normalization
CI	Confidence Interval
CNN	Convolutional Neural Network
CT	Computed Tomography
DL	Deep Learning
$DP-SGD$	Differentially Private Stochastic Gradient Descent
DP	Differential Privacy
FL	Federated Learning
GAN	Generative Adversarial Network
$k-NN$	k-Nearest Neighbors algorithm
MIA	Membership Inference Attack
ML	Machine Learning
$PATE$	Private Aggregation of Teacher Ensembles
$PPDP$	Privacy-Preserving Data Publishing
$PPML$	Privacy-Preserving Machine Learning
ROC	Receiver Operating Characteristic
$RT-PCR$	Reverse Transcription Polymerase Chain Reaction
$SMPC$	Secure Multi-Party Computation
SVM	Support-Vector Machine

1. Introduction

1.1. Motivation

The COVID-19 pandemic takes health systems worldwide to their limits. Accurate and fast tests are essential to prevent uncontrollable spreading. Detection of COVID-19 in patients can be achieved using an inexpensive RT-PCR test[1]. Nevertheless, the results can take hours to arrive. Even if displaying negative, the virus could have already left the throat and manifested itself in the lungs, rendering it undetectable by the test. In hospitals, chest X-rays can mitigate these drawbacks by enabling a fast and certain diagnosis [1]. Figure 1.1 shows chest X-ray scans of a 50-year-old COVID-19 patient over the course of a week. In this example, the patient suffered pneumonia symptoms, and the changes in the lungs are visible. Still, they are difficult to interpret for amateurs. Specialists, in contrast, are able to identify the severity of a case early on and can take measures without the need for lab results. Modern Machine Learning (ML) methods can effectively support medical professionals in an initial screening by classifying large numbers of images in a short time frame. However, the amount of data needed for training such classifiers poses problems due to clinical data privacy regulations, which present strict limitations to data sharing between hospitals [2].

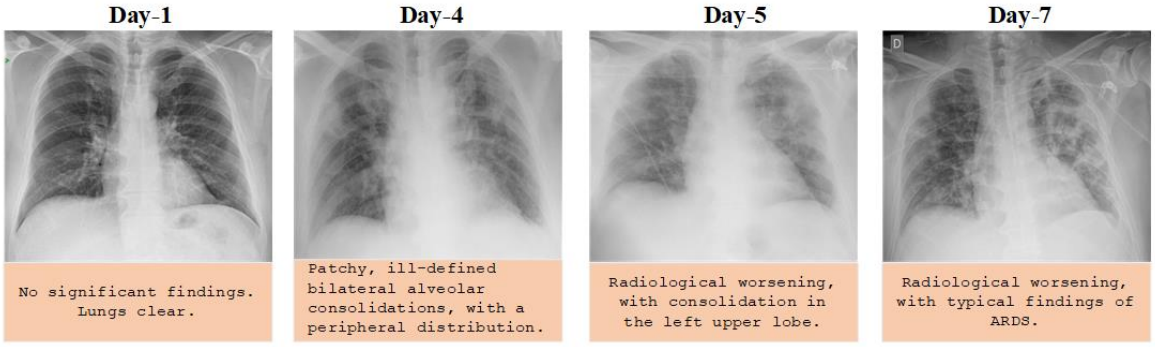

Figure 1.1.: Chest X-ray images of a 50-year-old COVID-19 patient with pneumonia symptoms over the course of a week. The patient suffered pneumonia symptoms. Figure from Ozturk et al. as seen in [1], with data published by Lorente in [3].

All sensitive patient information must be treated confidentially before, during, and after processing. To complicate matters, not only the dataset itself but also the prediction models resulting from ML can compromise data integrity. Published classification models are vulnerable to attacks, ultimately leaking details about their training data [4]. Such leaks allow adversaries to deduce private facts about individuals in the dataset. Privacy-Preserving ML (PPML) is a collection of methods for creating private models that satisfy Differential Privacy (DP), enabling the development of medical applications while maintaining patient privacy. If the profound research in PPML shows anything, it is ML models are under attack. This work aims to apply PPML procedures in the training of a classifier, thus preventing attacks on the resulting model from incurring data leakage.

[1]Reverse Transcription Polymerase Chain Reaction testing is broadly used for real-time COVID-19 diagnosis.

1.2. Goals

Investigating privacy-preserving ways of detecting COVID-19 in X-ray images can be divided into three successive steps. (i) First, a state-of-the-art non-private image classifier for chest X-ray images is trained and optimized to detect COVID-19 vs. Normal (no findings) posing as a baseline. This step also includes constructing a dataset of relevant scans. (ii) The second step focuses on evaluating PPML methods, with the primary objective being to find a feasible trade-off between privacy and consequently reduced accuracy. (iii) Finally, model security is assessed by reconstructing the training data through attacks. The attacks examine to what extent the models are prone to data leakage.

These steps help answer the general question proposed by this work: What is the best way to design a private image classification model detecting COVID-19 from chest X-ray images? Multiple other hypotheses accompany this question. These hypotheses form the experiments used to identify the best methods for private detection. Additionally, running non-private and private models enables comparing the influence of theoretical privacy guarantees to the empirical results from attacks. This work also investigates the possibilities and effectiveness of migrating PPML to different tasks since research in PPML typically concentrates on other datasets than COVID-19 X-rays. Even though privacy is the focus, solving this task combines many areas of research: data preparation, building a non-private X-ray classifier, adopting different PPML methods, experimenting with model architectures, and studying attacks on ML models. Hence, it contributes to the state-of-the-art in many ways.

1.3. Structure

After this brief introduction, Chapter 2 provides an overview of essential concepts and methods to understand the studied problems and upcoming solutions. The explanations cover image classification mainly using ML and privacy revolving around PPML. Chapter 3 then puts this work into context by examining existing literature on relevant topics. The related work inspires many experiments and supports the hypotheses tested. As a central part, Chapter 4 lays out the experimental setup. The chapter describes the planned experiments and the environment for running them, including the available hardware and needed software implementations. Moreover, empirical and theoretical reasoning justify selected approaches and settings in the chapter. Following up on the experimental setup, Chapter 5 evaluates the results. Potential findings are contrasted to the proposed hypotheses from theory and similar works. In closing, Chapter 6 gives conclusive thoughts regarding earlier chapters, their hypotheses, and the general task of privacy-preserving COVID-19 detection. The end of the chapter then adds an outlook on possible future work.

2. Background

This work builds on numerous existing fundamental concepts. These basics are continuously applied in the following chapters of this work without always providing the complete background matter. To enable a better understanding, this chapter elaborates on such topics relevant to the task, mainly regarding image classification and PPML.

2.1. COVID-19 Diagnosis From Images

COVID-19 is a respiratory disease that appears with the common symptoms of fever, cough, sore throat, and difficulty in breathing [5]. A widely used COVID-19 detection technique is RT-PCR tests. However, RT-PCR kits are prone to false negatives, and the results can take hours to arrive [6]. To overcome the sensitivity issue, medical professionals can use chest X-ray scans to confidently detect and diagnose COVID-19 [1]. The same can be achieved using Computed Tomography (CT), but X-ray machines are more widely available in hospitals due to the higher cost of the CT machines [7]. Besides, X-ray scans emit less ionizing radiation than CT [8].

Chest X-ray images of COVID-19 infected patients contain radiological signatures in the form of unique patterns and bilateral changes. Ng et al. [7] found that manifestation of COVID-19 infection in the lungs is predominantly characterized by ground-glass opacification with occasional consolidation. Multiple other studies confirm these findings [9, 10, 11]. In general, lung opacities seem to be a strong indicator. Figure 2.1 from [12] shows two X-rays of the same patient that both exhibit radiological characteristics related to COVID-19. At the time of scan A the patient showed a positive RT-PCR test result, and for B then presented a negative test but maintained COVID-19 positive antibodies. The relevant features are highlighted and involve patchy alveolar-interstitial opacities (black arrows) and diffuse (white arrows), with predominantly peripheral involvement and lung bases (black asterisks).

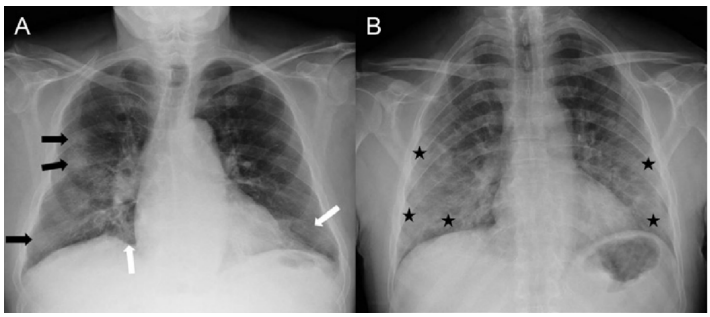

Figure 2.1.: Chest X-rays belonging to the same patient that both present COVID-19 related radiological characteristics. At the time of Scan A the patient showed a positive RT-PCR test result, and for B then presented a negative test but maintained COVID-19 positive antibodies. The relevant features are highlighted and involve patchy alveolar-interstitial opacities (black arrows) and diffuse (white arrows), with predominantly peripheral involvement and lung bases (black asterisks). The original figure is found in [12].

Performing manual COVID-19 testing from chest X-ray images is time-consuming and prone to errors [13]. Hence, there is a need for ML-based approaches that can support the analysis of chest X-rays and accelerate the screening process through automation.

2.2. Machine Learning for Image Classification

The following sections present the ideas that enable the successful classification of images using ML. The general background in ML cannot be covered exhaustively, as it would be too complex for a brief theoretical background. As a fundamental topic to this work, a basic understanding of neural networks in ML is assumed.

2.2.1. Image Classification with Convolutional Neural Networks

Image classification is a fundamental ML task that attempts to assign a specific label to an image. Right now, the dominating techniques are based on Deep Learning (DL), a subfield of ML, which is characterized by models stacking a large number of layers and limited by the resulting high parameter counts [14]. The most significant advantage of DL is automatic feature extraction, which means that these algorithms automatically grasp the relevant features to solve a problem by learning hierarchical representations [15]. Image classification is ruled by architectures called Convolutional Neural Networks (CNNs). They are named after the convolution, a mathematical operation simply applying a filter to the input that causes activation. Applying the same filter repeatedly to the input produces an activation map called a feature map, which indicates the location and strength of the features detected in the input.

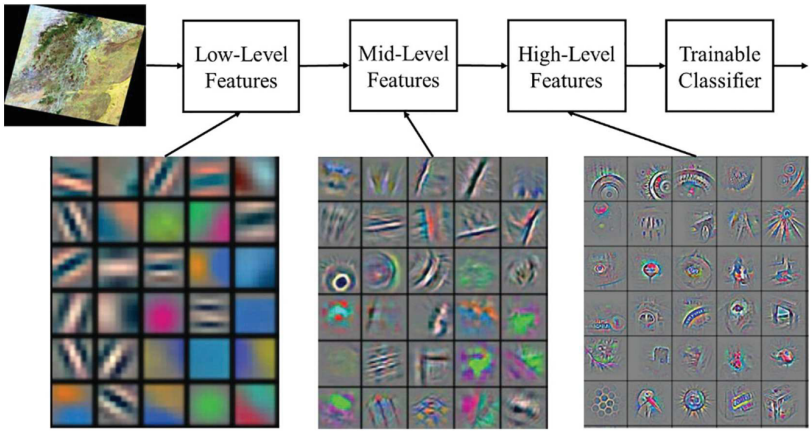

Figure 2.2.: Visualization of feature representations in CNNs at different levels. At a low level, they are primarily simple edge and color detectors, while mid-level and high-level features represent more complex forms or discriminative object parts. The high-level features in this graphic show forms that resemble an eye, the wheels of a car, or even the complete head of a bird. Figure from [15] by Traore et al.

With convolutions, the view can be filtered to ever smaller chunks of the image, enabling a neural network to extract features for tiny parts before forming larger portions. CNNs stack multiple convolutional layers as blocks and accomplish to extract image features on different levels. A visualization of such features is shown in Figure 2.2. At a low level, they

are primarily simple edge and color detectors, while mid-level and high-level features represent more complex forms or discriminative object parts. The high-level features in Figure 2.2 show forms that resemble an eye, the wheels of a car, or even the head of a bird. This regional perception shows how CNNS were inspired by the biological processes in the visual cortex [16]. Here, single neuron only responds to stimuli within a restricted area of the visual field, but different neurons partially overlap to cover the total area.

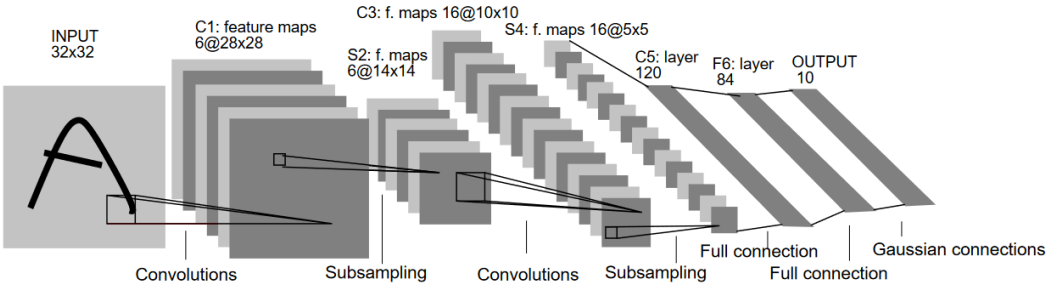

Figure 2.3.: Illustration of the LeNet-5 architecture described in 1998 by LeCun et al. [17]. The input image of 32x32 pixels passes through two convolutions and sub-samplings. The remaining feature maps then reach the fully-connected layers. The results are fed into an output layer that generates the final prediction. Most modern CNNs follow a similar strategy but might implement new layers or change their order. Figure as seen in the original paper [17].

CNNs excel in visual tasks. The first CNN to beat other techniques in image classification was the AlexNet in 2012. The network was designed by Krizhevsky et al. in [18]. Although it took CNNs until 2012 to overtake other methods, LeCun et al. [17] already described a similar architecture in 1998. Their LeNet-5 architecture is illustrated in Figure 2.3. The input image of 32x32 pixels passes through two convolutions and sub-samplings. Sub-sampling layers reduce the spatial resolution of the image, which means reducing the number of pixels within a feature map. The remaining feature maps then reach the fully-connected layers. Neurons in a fully-connected layer have full connections to all activations in the previous layer, as in regular neural networks. The results are fed into an output layer that generates the final prediction. Most modern CNNs follow a similar strategy but might implement new layers or change their order.

Apart from convolutional, sub-sampling, and fully-connected layers, relevant layers in CNNs include:

- *Activation Layer:* Activation functions for convolutional and fully-connected layers can be seen as separate following layers but are usually specified directly as a setting of the preceding layer. A prevalent choice is the ReLU activation function.

- *Output Layer:* The final prediction is calculated using a classification function. The number of classes in the ML task is represented by the number of neurons in the output layer. There are two relevant functions depending on the task: (i) sigmoid for binary classification with one neuron and (ii) softmax for binary or multi-class classification with two or more neurons.

- *Dropout Layer:* A Dropout layer can mostly be found after neuronal layers[2] and randomly deactivates nodes of this layer. Deactivation sets a node to 0, meaning its value is not passed to the next layer. The probability of a layer's node being dropped is commonly chosen as 0.2 or 0.5. The network will have to adapt to the random change in its architecture, which has a regularizing effect that helps with overfitting [19].
- *Batch Normalization (BN) Layer:* The BN layer is used to normalize inputs. The inputs to the layer are standardized to control the change of the layers' input distributions during training [20]. BN is commonly found after neuronal layers to stabilize the inputs for the next layer.

2.2.2. Passing on Knowledge With Transfer Learning

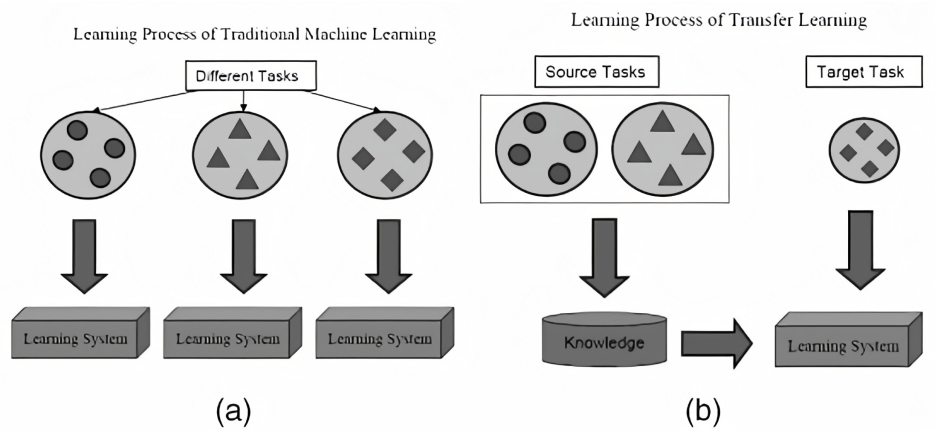

Figure 2.4.: Different learning processes in (a) traditional ML and (b) transfer learning. Compared to just using the standard dataset (a), more information is gained from one or multiple other datasets (b). As seen in [21] by Pan and Yang.

As humans, we encounter transfer learning regularly. After studying one programming language, we can apply related knowledge from prior experience when learning another one, making it more accessible. Further, we are specifically able to learn from only a few examples [22]. "For instance, medical students can rapidly learn certain abnormalities on brain MRI images with only a handful of examples because they are already familiar with healthy brain structures. In contrast, this may be very difficult for a non-medical student lacking previous neuroscience knowledge." [23] Rather than starting from zero, the human brain can connect between similar tasks [21, 24]. Similarly, ML models can learn and improve by transferring knowledge. At first, this seems counterintuitive since the common assumption of traditional ML methodologies is that all data has to be taken from the same domain, and each model is trained on a specific dataset to solve one isolated task. Nevertheless, to come closer to human learning, transfer learning tries to boost performance by carrying over knowledge from related source tasks to the actual target task, as seen in Figure 2.4. Compared to just using the standard dataset (a), more information is gained from one or multiple other datasets

[2] A neuronal layer refers to a layer that has trainable weights, e.g., convolutional and fully-connected layers.

Lucas Lange
3737104

(b). Transfer learning aims at making use of the DL principle that more data equals better performance [25].

Transfer methods can be characterized by what is being transferred. In this work, the focus is on transferring knowledge using shared parameters between source and target domain models [26, 27]. Effectively, this means passing on the learned structure represented in the weights of a source model. A common strategy in DL is to use the weights of the source model as starting weights for the target model instead of randomly initializing them. This method is called pre-training since the layer weights are already adapted to the source task and data, before the actual training. In this way, the previously gained similar knowledge is maintained, and the weights can be fine-tuned by further training in the actual target domain. In the simple case, the source and target models have the same architecture, and weights can be copied for all layers. It is slightly more difficult if models are different, but copying a selection of matching layers can still result in a positive transfer.

Another factor for tuning a pre-trained model is freezing certain layers. Freezing a layer means its weights are made static, so they are not adjusted while training and stay at their transferred value. This method can help when the training process initially changes the weights too fast, destroying the initial advantage from pre-training. One way to reduce the unlearning of transferred weights can be by lowering the learning rate. There is no straightforward way of deciding which and how many layers to freeze, and it can thus be a hyperparameter to tune. The most common strategies are: freezing all layers, freezing all layers except for the output, freezing only the first or second half of layers, and not freezing any layers. A major drawback of freezing layers is that it hinders the adaptation abilities to the target task. [28]

A unique system for knowledge transfer is teacher-student learning. The teacher is a source model or a whole ensemble of source models, while the student model represents the target model. Here, instead of direct weight exchange, source models teach a target model by influencing its learning process. The goal for the student is to achieve the same predictions as the teacher models. So, given a single training sample, the student's loss is the difference between the teacher prediction and student prediction. The student's weights are then adjusted to get closer to the teacher's output. In this way, the weights are influenced through training instead of simply setting them as starting weights. Thereby, the actual architectures of the models are made irrelevant. The teacher-student approach can also be used after initial training and thereby offers a way to adapt a student model to a new domain later on. [29]

An ML algorithm can benefit from transfer learning in three ways: (i) boost the initial performance level before any further training is done, (ii) reduce training time requirements compared to learning from scratch, (iii) raise the final level of achievable performance [24]. One or all of these benefits may apply in a given instance. Figure 2.5 shows a graphical interpretation of the possible positive performance changes in a transfer learning setup compared to traditional ML. Negative transfer occurs when a transfer method decreases performance. Torrey and Shavlik [24] name positive transfer as one of the major challenges in developing transfer methods. The main culprit is the use of less related source tasks, showing that the effectiveness of transfer methods is fundamentally defined by selecting suitable information to pass on.

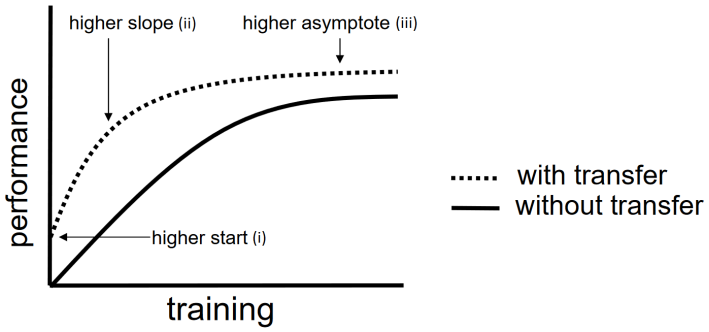

Figure 2.5.: Three possible ways of transfer learning improving model performance. An ML algorithm can benefit from transfer learning in three ways: (i) boost the initial performance level before any further training is done, (ii) reduce training time requirements compared to learning from scratch, (iii) raise the final level of achievable performance [24]. One or all of these benefits may apply in a given instance. Figure from Torrey and Shavlik [24].

After grasping what transfer learning does and what is gained from it, some use cases show the practical implications. As already seen with teacher-student models, transfer learning can be a more straightforward solution than training a whole new model from scratch when domain adaption is needed. Also, the implied data distribution changes are problematic for most statistical models because they need to be rebuilt based on the new data. Such changes could happen when new data is collected, existing data is updated, or data generally changes over time. Another problem tackled with transfer learning is the lack of labeled training data for a given target task. The vast amounts of data needed for effective DL require expensive collection and labeling endeavors. With transfer learning, information from closely related domains can be used for pre-training, effectively enlarging the available data. For clarification, Pan and Yang [21] give the following example:

> We consider the problem of sentiment classification, where the task is to automatically classify the reviews on a product, such as a brand of camera, into positive and negative views. For this classification task, we need to first collect many reviews of the product and annotate them. We would then train a classifier on the reviews with their corresponding labels. Since the distribution of review data among different types of products can be very different, to maintain good classification performance, we need to collect a large amount of labeled data in order to train the review-classification models for each product. However, this data-labeling process can be very expensive to do. To reduce the effort for annotating reviews for various products, we may want to adapt a classification model that is trained on some products to help learn classification models for some other products.

In conclusion, transfer learning can considerably reduce the labeling effort in these cases [30]. Similarly, this also implies a reduced need for training data from the actual target domain [31].

2.2.3. Classification Metrics

In this section, a glossary explains the (image) classification metrics used in this work.

- *Classification Accuracy:* A model's (classification) accuracy measures its ability to distinguish the target classes. It is the percentage of correctly classified samples over all tested samples. If the model gives the correct prediction for half of the samples, its accuracy is 50%.
- *Precision:* Precision is the proportion of positive predictions that was actually correct. A model with a precision of 0.5 is correct 50% of the time when predicting a positive case.
- *Recall:* Recall is the proportion of actual positives that were identified correctly. A model with a recall of 0.5 correctly identifies 50% of all positive cases.

2.3. Data Privacy in Machine Learning

This privacy-centered section poses as a broad introduction to PPML and its accompanying challenges. The content elaborates on some of the possible threats to ML and explains corresponding privacy-preserving techniques.

2.3.1. The Dangers of Publishing Data

The field of Privacy-Preserving Data Publishing (PPDP) focuses on making private datasets publicly available. The needed privacy is achieved by anonymization before release, i.e., removing private details or replacing them with random values. The common strategies resolve around k-anonymity, l-diversity, and t-closeness [32]. However, the more privacy is offered, the more utility is destroyed, potentially rendering the data useless for further analysis. Moreover, re-identification can be done on anonymized data. A recent famous example is an attack on a private movie rating dataset offered for a challenge by Netflix[3]. Because members of the database also publicly shared ratings on IMDb, an online database for film and television content, linking information from both data sources allowed the recovery of their identities [33].

2.3.2. Privacy-Preserving Machine Learning

PPML, in contrast to PPDP, revolves specifically around creating private ML processes, regardless of the privacy attributes a given dataset contributes. The methods should prevent data leakage from the resulting ML model. Achieving a private model is a vital step when public availability is envisioned, since an adversary could disclose information from models using attacks. This section discusses possible defense mechanisms mainly based on DP. DP is an ever-present core concept in privacy preservation, constituting a measurable privacy standard. In PPML, DP is most commonly achieved through applying random noise in different stages of ML. Because of this noise, PPML is subject to a comparable utility-privacy trade-off as PPDP since higher noise leads to more security but less performance.

[3]More information on the challenge: `https://www.netflixprize.com/`

2.3.2.1. The Concept of Differential Privacy

An ever-present concept and gold standard metric in private data publishing and processing is DP, which offers a guarantee that the removal or addition of a single database item does not (substantially) affect the outcome of any analysis [34]. Thus, an attacker should not be capable of differentiating which of the neighboring datasets was used for the results and, in the optimal case, has to resolve to a wild guess—i.e., a coin flip. The privacy guarantee is measured by the worst-case scenario given as the theoretical upper bound of privacy loss. This upper bound is represented by the privacy budget ε. It is accompanied by the probability of privacy being broken by accidental information leakage, which is denoted as δ and depends on the dataset size. Formally, an algorithm A training on a set S is called (ε,δ)-differentially-private, if for all datasets D and D' that differ by exactly one record:

$$Pr[A(D) \in S] \leq e^{\varepsilon} Pr[A(D') \in S] + \delta \tag{2.1}$$

Meaningful privacy guarantees should fulfill $\varepsilon \leq 1$ and $\delta \leq 1/n$, where n is the number of training samples in the dataset [35]. Still, for example, Apple stated to use privacy budgets ε between 2 and 8 when collecting user data from their features, and studies found the actual loss to be even higher [36, 37, 38]. The notation $\varepsilon = \infty$ is used to indicate that no DP criteria are met. In computational practice, DP is accomplished by adding enough statistical noise in a randomized algorithm so an attacker cannot notice any significant difference between neighboring query outputs [39]. Intuitively, the applied noise interferes with the algorithm's effectiveness, introducing a utility-privacy trade-off.

There are two common mechanisms used for sampling random noise in DP algorithms: the Laplace noise mechanism and the Gaussian noise mechanism [40]. Since they guarantee DP by adding some controlled noise level on every iteration, they are named additive-noise mechanisms. Their main difference is the underlying distribution from which they draw noise—here, Laplace and Gaussian distribution. The noise scale determines the possible space for the distributions to draw from, where a greater noise level provides better privacy. However, just how much noise is needed at every step? The noise scale parameter regulates how much privacy loss is added with each query. Suppose a dataset is queried with a function. In that case, the sensitivity of the function states the largest possible impact one record can have on the value of the function, which in return is the amount of uncertainty needed to hide any single contribution [35]. Consequently, the noise scale must be calibrated to the given sensitivity. When algorithms are designed with additive-noise mechanisms, privacy is measured by the accumulated privacy loss given as the privacy budget ε. The general goal is to use these mechanisms to perturb each query. Abadi et al. [41] created the moments accountant, a privacy accountant that keeps track of these moments of privacy loss. It thereby allows the estimation of the resulting bounds from all queries. In addition to the sensitivity, the applied noise at each query must adhere to the desired accumulated ε value.

The modular design of DP algorithms is possible because of one of DP's fundamental properties: composability. It states that "if all the components of a mechanism are differentially private, then so is their composition" [41]. Another essential attribute of DP is its post-processing immunity, implying DP is preserved by all further processing. Therefore, training

a classification model on an already DP conform dataset or in a DP ensuring way both results in a private model [42]. In theory, a DP-conform model can be released without the need for further privacy tweaks.

2.3.2.2. Methods of Privacy-Preserving Machine Learning

This section goes over the most relevant approaches to PPML to give a general background in the topic. The typical way of applying DP is by adding noise in different stages of ML. In general, DP procedures can be differentiated between local and global DP. In PPML, local DP applies some noise to each training sample during training, while global DP, in contrast, adds noise only to the outputs of the released model and thus allows for a standard training procedure without noise [43]. These two techniques, known as private training (local DP) and private prediction (global DP), are the leading contenders in the PPML space. In private prediction, no model is released, but instead, users query the model through an appropriate interface [44]. Thus, only predictions are prone to attacks and need to be noisy. Private training, on the other hand, enables the publication of the entire model by incorporating the required random noise into the ML training process [45]. However, introducing randomness into the training can worsen the accuracy-privacy trade-off.

The different forms of perturbation in ML include [4, 35]:

- *Input Perturbation:* Adding noise to the input data itself. After the following non-private processing steps on the noisy input, the output would still be differentially-private due to the post-processing immunity [42].
- *Output Perturbation:* Adding noise to query results (predictions). Answering too many queries without increasing the noise level can violate DP, which can make a limit on the maximum number of queries necessary [46].
- *Objective Perturbation:* Adding noise to the optimization problem, i.e., the objective function. This method perturbs the objective function before optimizing it [47].
- *Algorithm Perturbation:* Adding noise to intermediate values in iterative algorithms—e.g., gradient perturbation adds noise to gradients. The total amount of noise is composed over iterations [41].

The Differentially Private Stochastic Gradient Descent (DP-SGD) algorithm as proposed by Abadi et al. in [41] takes widely used SGD and applies gradient perturbation while training to ensure DP. Thus, the algorithm adds random noise to gradients when processing a minibatch and applies privacy protection to a single sample from the training dataset at a time. The workings of the DP-SGD algorithm are detailed in Figure 2.6. The used notations are θ_t for model parameters, $\Delta L(\theta_t)$ for gradients w.r.t. the parameters, and η for learning rate. Conventional SGD is shown in the lower part of the figure. Here, the gradients are computed, after which the current parameters are updated by directly applying the gradients with a given learning rate to get the new set of parameters. However, as seen in the upper part, in DP-SGD before updating, the gradients are clipped to a predefined maximum Euclidean norm. Clipping is important to bound the influence of individual samples [49]. In a

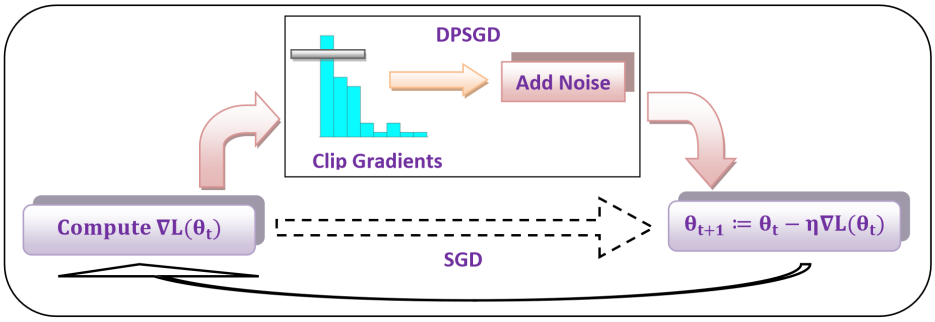

Figure 2.6.: Workflow representation of the gradient updating steps in the DP-SGD algorithm. Conventional SGD is shown in the lower part updating parameters by directly applying gradients. The DP-SGD related additions of clipping and noise are depicted in the upper part. Notations are θ_t for model parameters, $\nabla L(\theta_t)$ for gradients w.r.t. the parameters, and η for learning rate. Illustration from Rahman et al. in [48].

second step, random Gaussian noise is added to the clipped gradient to achieve DP conform perturbation. Regarding the needed noise scale, it is only necessary to add just enough to hide the largest gradient in a batch, as that is the sample that is at most risk of exposure [50]. Since clipping ensures each gradient has a known maximum norm, finding the largest possible gradient is trivial. The noisy gradients are then used to update the parameters as usual. Generally speaking, a larger dataset allows adding less noise to each gradient since the sampling probability for a single example is reduced. In the same way, training for more iterations requires adding more noise. For the most part, the amount of noise added through the algorithm determines the provided privacy budget ε, and the resulting private model suffers a hit in model accuracy linked to this chosen privacy guarantee.

Private Aggregation of Teacher Ensembles (PATE) is a different attempt at private learning described by Papernot et al. in [52], which uses the coordination of several ML models to achieve its privacy promise. As an implementation advantage, the learning algorithms themselves rest untouched as opposed to DP-SGD. The original version was extended and refined in [53] to better scale and reduce noise while still offering better privacy guarantees. Papernot presents the simple intuition behind PATE in a blog post [51]: "If two different classifiers, trained on two different datasets with no training samples in common, agree on how to classify a new input example, then that decision does not reveal information about any single training example. The decision could have been made with or without any single training example, because both the model trained with that example and the model trained without that example reached the same conclusion." To achieve this, they follow a three-step process broken down in Figure 2.7. The dataset is first split into a selected number of distinct subsets, on each of which a separate public ML model is trained, called a teacher. The second step is to take an input sample and get a common prediction from the trained teacher ensemble. In noisy aggregation, noise is added when combining the individual predictions made by the teachers to reach a single common prediction. The aggregation is carried out by counting the class prediction of each teacher as a vote and then applying noise to the vote counts using a noise mechanism. The noise scale depends on the desired privacy budget ε, and intuitively, its influence on the votes is less pronounced the more teachers are voting. Thus, splitting the

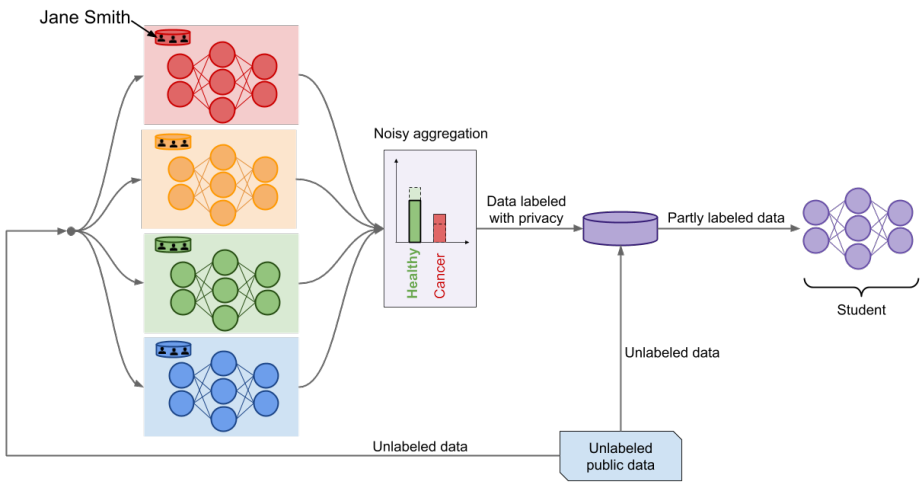

Figure 2.7.: Three-step training process employed by the PATE algorithm. The dataset is split into distinct subsets, on each of which a separate ML model is trained, called a teacher. The second step is to take an input sample and get a common prediction from the trained teacher ensemble. In noisy aggregation, noise is added when combining the individual predictions made by the teachers to reach a single common prediction. The aggregation is carried out by counting the class prediction of each teacher as a vote and then applying noise to the vote counts using a noise mechanism. The class with the highest noisy count is the final noisy prediction. The third and last step trains a final student model. The student uses inputs from a set of unlabeled public data. The public samples are labeled by the noisy aggregation of the teacher, making the resulting student model private [42]. Image from a blog post of Papernot and Goodfellow [51].

dataset more often reduces the accuracy-privacy trade-off, as long as the resulting subsets are big enough for teacher training. The class with the highest noisy count is the final noisy prediction. In theory, this process would already allow for private prediction. However, if the entire model was to be released, access to the public teacher parameters would destroy the privacy guarantees. Therefore the third and last step introduces a countermeasure in training a final student model, which undergoes privacy-preserving transfer learning from the teachers. The student uses inputs from a set of unlabeled public data. This data is distinct from the teacher training sets and must be put aside beforehand. The public samples are given to the teacher models for labeling using the noisy aggregation mechanism, which responds with noisy private labels. Training on the privatized data, results in a private student model due to the post-processing immunity of DP [42]. The private student model can then be published with the full privacy guarantee.

Generative Adversarial Networks (GANs) offer the possibility of artificially enriching a dataset by learning the statistical distribution of training data and afterwards allowing to synthesize samples from the learned distribution [54, 55]. Nevertheless, GANs are not only interesting for enlarging the training set but also for preserving its privacy, as shown in multiple use cases like big data publishing, face de-identification, and vehicular camera data [56, 57, 58]. In these cases, a GAN is trained using PPML techniques to guarantee DP, resulting in a DP-GAN that subsequently creates DP conform training samples of solely synthesized images based on the original data. For example, DP-SGD can be used for private GAN training [59, 60]. The general approach is in parts comparable to PATE since the GAN creates

a private dataset for a student model to train on and also matches PATE in performance while offering a more general framework [61]. When using the generated differentially private dataset, the post-processing immunity of DP allows training a private model utilizing non-private training [42].

Applying privacy at a different time, the (sub-)sample and aggregate framework aims at responding to queries whose answers can be approximated based on only a small number of samples while ensuring DP [62]. As Ji et al. further describe the algorithm in [63], it first goes through a sampling step, where the dataset is split into many small subsets and estimates the answer on each. In the following aggregating step, all subset results are combined to estimate a model while simultaneously adding noise. It, therefore, replicates the PATE algorithm but omits the student. As an advantage, no accuracy is lost through private training, and the framework solely strives for private prediction, where the model itself is not released.

Lately, the Federated Learning (FL) framework surged in many areas since it removes the need for central computation and hence allows data to stay locally within each participating institution [64]. The distributed learning protocol jointly trains an ML model without the data leaving the users' devices. Instead, each device calculates model updates and sends them to a central party, aggregating them to generate a shared model. In this setup, the need for a central aggregator can pose risks to data privacy. A possible combination of techniques is FL and Secure Multi-Party Computation (SMPC), which strives for parties to jointly compute a function over their inputs and at the same time keeps those inputs private [65, 66, 67]. While ensuring a secure distribution of gradients inside the FL network, neither FL nor SMPC ensures DP [68]. Thus, there is no guarantee for DP through the FL algorithm itself, leading to a need for PPML training methods to be employed. Geyer et al. [69] demonstrate a way of keeping DP in FL that only leads to marginal performance losses. They do not aim to protect a single data point's contribution but rather to protect a whole client's data set, rendering a single client's contribution untraceable. The main downside is that the number of participants has to be sufficiently large, i.e., over 1,000 and up to 10,000 clients. Instead of counting on effective obfuscation through client numbers, Malekzadeh et al. in [70] employ DP-SGD in the FL framework to guarantee DP. The DP-SGD operations are not applied on the batch level but the gradients at each node in the FL ensemble. With this method, they lose about 4% accuracy compared to non-private methods, providing an ε of 11. Still, using more clients results in a better accuracy-privacy trade-off since with more clients, more noise can be distributed over the ensemble without heavily impacting performance [71].

2.3.2.3. Attacks on Machine Learning Models

When it comes to attacks on ML models, the focus in this work is on published models and information extraction. The granted public access to a model is either white-box (access to a model's feature vectors) or black-box (access only to model predictions). While black-box access is more secure on paper, all these models are open to threats on different levels. Al-Rubaie and Chang [4] differentiate between three main types of attacks as illustrated in Figure 2.8. The upper part illustrates how model inversion attacks and reconstruction attacks can retrace the training path from the published model over the feature vectors back to the

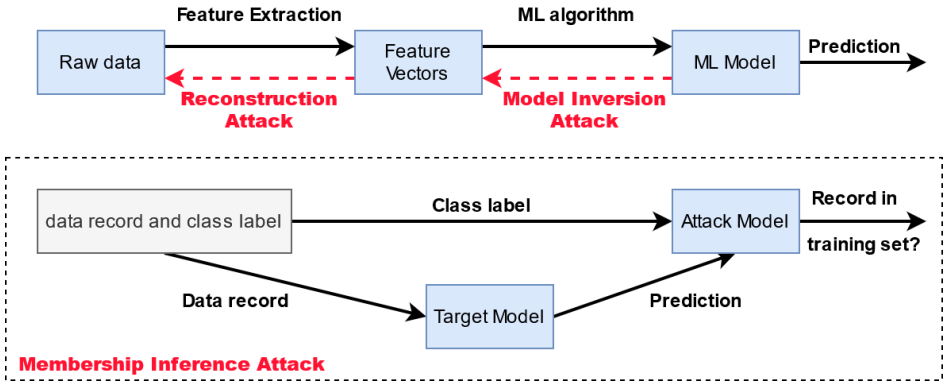

Figure 2.8.: Overview of the different attacks on ML models. The upper part illustrates how model inversion attacks and reconstruction attacks can retrace the training path from the published model over the feature vectors back to the raw data. On the other hand, the second half of the image shows how in MIAs specifically trained attack models can be used to elicit information from a model. Adapted from Al-Rubaie and Chang [4].

raw data. On the other hand, the second half of the image shows how in MIAs specifically trained attack models can be used to elicit information from a model. The overview supports the following explanations for each attack, which are mainly based on the descriptions by Al-Rubaie and Chang [4]:

1. *Reconstruction Attacks:* An adversary tries to reconstruct the private data from the feature vectors. Therefore, white-box access to the ML model is required to succeed. Support-Vector Machines (SVMs) or the k-Nearest Neighbors algorithm (k-NN) are ML approaches that problematically store feature vectors in the model itself, allowing easy access. To give a successful example, an attack by Feng and Jain [72] reconstructed a fingerprint image (raw data) from only a minutiae template (features).
   Defense: Restricting users to black-box access to the published model protects the feature vectors. Models storing explicit feature vectors should be avoided or not published. Further, synthesizing feature vectors through model inversion attacks should be prevented.

2. *Model Inversion Attacks:* Here, an attacker either has (i) white-box access to the model revealing internal information but without stored feature vectors, or (ii) black-box access. The goal is synthesizing feature vectors mirroring the ones used for a model's creation. These are attained by exploiting confidence information (e.g., probabilities) regarding predictions based on submitted testing samples. Taking the output of multiple queries can produce an average characterizing a specific class, which could, in turn, represent a single individual, e.g., in face recognition. Fredrikson et al. achieved such re-identification in [73], where inversion attacks were able to extract targeted faces. Furthermore, since the features directly matched the raw data (images), they needed no further reconstruction attack.
   Defense: Only black-box access should be granted while also limiting the output to reduce gained knowledge. Such limiting measures could be rounding confidence scores as seen in [73] or entirely omitting them as in [74], where they only delivered class labels.

3. *Membership Inference Attacks (MIAs):* Given a model and a sample, these attacks try to figure out if the sample was part of the model's training dataset based solely on its prediction outputs on the sample. If successful, the membership of an individual's record in the set could be disclosed. To gain insights, the technique takes advantage of the differences in predictions made on samples used for training versus unseen ones, where the former is expected to deliver higher confidence values. Attack models, as proposed by Shokri et al. in [75] take the correct label of a sample and the target model's prediction as inputs to decide if they existed in the training set. Creating such models uses multiple shadow models trained on input data having roughly the same distribution as the target's data. The needed data can be obtained by different methods like model inversion attacks, statistics-based synthesis, or noisy similar real data. The finished shadow models are then used to train the actual membership classifier. Training and performing attacks only needs to utilize black-box access, while white-box access can help with further insights regarding the original data. [75]
   Defense: In this regard, Shokri et al. [75] showed that limiting the model outputs to only return a class label instead of confidence values was an effective and straightforward defense mechanism. Still, they could not achieve complete mitigation of the attack. The concept of DP should be able to oppose and limit MIAs by design [75, 76]. However, only by simultaneously decreasing the utility of the ML model, as investigated in [48].

2.3.2.4. Diving Deeper Into Membership Inference

To better grasp why MIAs threaten to leak critical information, assume this example from a blog post by Boenisch [77]:

> Imagine you are in a clinical context. There, you may have an ML model that is supposed to predict an adequate medical treatment for cancer patients. This model, naturally, needs to be trained on the data of cancer patients. Hence, given a data point, if you are able to determine that it was indeed part of the model's training data, you will know that the corresponding patient must have cancer. As a consequence, this patient's privacy would be disclosed.

The success of MIAs against models strongly relates to the memorization effects present in ML [78, 79]. During training, a model aims at perfecting its classification of the given data. This focus can lead models to simply learn a perfect representation of their training data. When models start to mainly memorize the labels of their training data instead of learning a general classification, they sacrifice their ability to generalize to new data. This problem is called overfitting and is closely linked to memorization [78]. Overfitting is a core difficulty of ML classifiers and one of the main amplifiers for MIAs [80]. Since it represents a model's inability to generalize, overfitting can be measured by the train-test accuracy gap, which is the difference between a model's accuracy on its training and test data [81]. The carried memorization is reflected in a model's predictions, favoring known samples through higher confidence values. MIAs can then exploit this effect to determine which image was part of the original training data.

Using thresholds is a basic way of performing MIAs. A threshold-based attack takes the returned confidence scores when predicting on a sample and decides about its membership based on a set boundary value that constitutes the threshold—a method based on [82]. As a simple differentiating method, this can be a rather naive approach to membership inference. In general, utilizing ML to differentiate is expected to be slightly better than thresholds in MIAs [80].

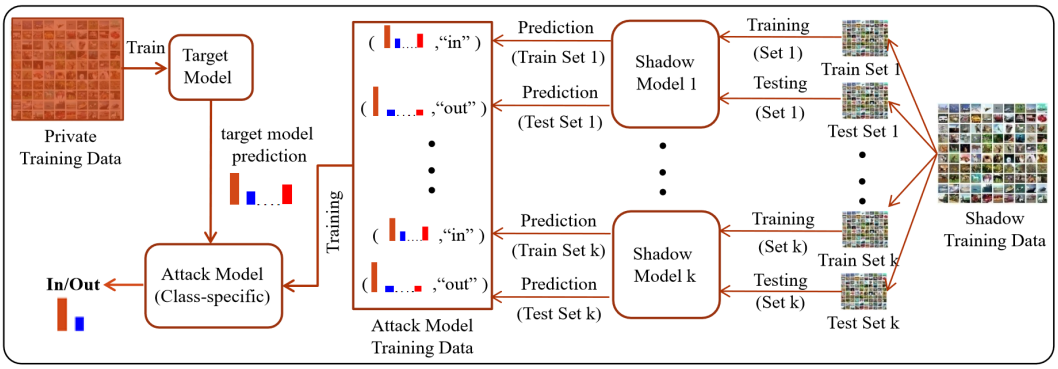

Figure 2.9.: Visualization of the MIA pipeline for attack model training using shadow models and their usage as described in [75]. The final attack model is a collection of several models—one for each output class in the target model. In order to train each class-specific attack model, k shadow models are constructed. Those models are supposed to imitate the behavior of the target ML model. The shadow training data is known to the attacker, allowing a distinction between 'in' and 'out' regarding membership. Each shadow model's predictions mimic the target model, but training images are classified as 'in' and test images as 'out'. The resulting dataset gathered from all k shadow models can then be used to train an attack model to distinguish the 'in' and 'out' samples based on the corresponding confidence values. After training one attack model for each output class in the target model, the attacker queries the target model with a sample. The corresponding class-specific attack model then handles the emitted class prediction and states the sample's membership probabilities in terms of 'in' and 'out'. Graphic from Rahman et al. in [48].

In their ML form as introduced by Shokri et al. in [75], MIAs aim at constructing an attack model that recognizes the differences in a model's emitted confidence values. Its training utilizes shadow models to imitate the target's behavior, and the whole process is visualized in Figure 2.9. The final attack model is a collection of several models—one for each output class in the target model. In order to train each class-specific attack model, k shadow models are constructed. Those models are supposed to imitate the behavior of the target ML model, which is supported by drawing their shadow training data from the same population as the target model's data. However, the shadow training data is known to the attacker, allowing a distinction between 'in' and 'out' regarding membership. Each shadow model's predictions mimic the target model, but training images are classified as 'in' and test images as 'out'. The resulting dataset gathered from all k shadow models can then be used to train an attack model to distinguish the 'in' and 'out' samples based on the corresponding confidence values. The learned classification also applies to the target since the shadow models imitate it. Shokri et al. claim that the higher the chosen number of shadow models k, the better the resulting class-specific attack model [75]. After training one attack model for each output class in the target model, the attacker queries the target model with a sample. The corresponding

class-specific attack model then handles the emitted class prediction and states the sample's membership probabilities in terms of 'in' and 'out'.

In a subsequent work to Shokri et al. [75], Salem et al. [83] relaxed the prerequisite of using shadow models during training. Their adversary does not need shadow models and relies solely on the target model's outputs. Thereby, no training of any approximating shadow model is required. Boenisch [77] describes the target model in this setting as "a shadow model that approximates its own behavior perfectly".

2.3.3. Privacy Metrics

In this section, a glossary explains the privacy metrics used in this work.

- *Privacy Budget ε:* As introduced in Section 2.3.2.1, the privacy budget ε gives a theoretical upper bound of privacy loss regarding a DP guarantee. The achieved value primarily depends on the applied noise scale. If no DP is guaranteed, the privacy budget is given as $\varepsilon = \infty$.

- *MIA Accuracy:* The accuracy of an MIA states how accurately it can classify samples as inside and outside of the target's training set. It is the percentage of correctly classified samples over all tested samples. If the model gives the correct prediction for half of the samples, its accuracy is 50%.

- *MIA AUC:* AUC stands for Area Under the Curve, which refers to the ROC curve (Receiver Operating Characteristic curve). That is a graph showing the performance of a classification model at all classification thresholds. The AUC then provides an aggregate measure across all possible classification thresholds. In that, it ultimately represents the probability that a randomly chosen positive sample is ranked higher than a randomly chosen negative sample [84]. AUC says how successful the attacker is at recognizing which example was part of the original training set. An optimal defense results in an AUC of 0.5 or lower, which translates to an attacker randomly guessing the membership of a sample. An AUC of 1 means that an attacker can always identify the correct membership.

- *Empirical Privacy Budget ε_{AUC}:* The empirical privacy budget ε_{AUC} is based on the AUC of an MIA and relates it to the theoretical privacy budget metric. The AUC values achieved by the attacks are used to estimate the corresponding privacy guarantee of the target model. An AUC of 1.0 results in an $\varepsilon_{AUC} = \infty$, while values lower or equal 0.5 translate to the perfect guarantee of $\varepsilon_{AUC} = 0$. In between these boundaries, the epsilon is estimated based on the following formula: $\varepsilon_{AUC} = \ln(\frac{AUC}{1-AUC})$. This notion is adapted from the recent work of [85], where they propose this metric based on MIA accuracy. With a value given as a privacy budget, its comparison to the theoretical assumptions is more intuitive.

3. Related Work

The related works covered in this section pose as comparative data points and as inspiration for solving the COVID-19 task. Therefore, the focus when selecting literature is on works regarding the classification of X-ray images, preferably COVID-19 detection. Section 3.1 resolves around different approaches to non-private classification and lays out the problems arising from limited data availability. Section 3.2 then surveys the available solutions for privacy preservation when handling ML tasks on X-ray images. Each approach's results are briefly compared to the achievements on other tasks to enable better contextualization in the domain of PPML. The final Section 3.3 reviews the use of attacks in other works and shows how they can help to evaluate a model's privacy.

3.1. X-Ray Image Classification

In a general setting, the dominating technique in image classification is DL, specifically in the form of CNNs. CNN's stack multiple layers as blocks and are designed to automatically extract image features by learning spatial hierarchies [15]. In 2012 the AlexNet by Krizhevsky et al. [18] was the first CNN to top the ImageNet challenge, a famous image classification task. Their dominance is held in many tasks, even when comparing them to new state-of-the-art techniques like transformers [86]. Yamashita et al. [87] found that their strong performance carries over into radiology but is hampered by small datasets and resulting overfitting—a central concern in this work due to the insufficient availability of COVID-19 scans. They also describe techniques to tackle these problems, involving regularization (e.g., dropout, weight decay), data augmentation, operating on a train-validation-test split, and transfer learning (e.g., on ImageNet). An approach to enlarge X-ray datasets synthetically by Loey et al. [88] takes existing COVID-19 images and uses a GAN to create more training images. Sedik et al. [89] then found an average increase of 4% to 11% in COVID-19 detection from a GAN-enlarged training set, reaching 99% accuracy in their COVID-19 classification.

When examining existing approaches, the comparability of results poses difficulties due to the lack of a common dataset. Related works span the whole pandemic timeline, with different amounts of data publicly available at each point in time. At the time of this work, the COVID-19 Radiography Database[4] provides the most comprehensive collection containing 3,616 COVID-19 positive cases [90, 91]. Still, even this collection does not provide a defined test set. Having a common test set would greatly improve the comparability of results. Related work on this dataset should give the most recent insights on the achievable performance in this work.

Different architectures and variations of CNNs find use in COVID-19 detection. On the binary classification of COVID-19 vs. Normal (no findings), the classification accuracy routinely surpasses 90%, and the ResNet50 architecture consistently demonstrates strong results of up to 98% [92, 93, 94, 95, 96]. In [1] Ozturk et al. find that their proposed DarkCovidNet model, a tailored approach, reaches the same 98% as the ResNet50 while being only 5% of

[4] Available on Kaggle: `https://www.kaggle.com/tawsifurrahman/covid19-radiography-database`

its size in terms of parameter count. The creators of the COVID-19 Radiography Database achieved over 99% on their dataset using a ResNet18 [90].

In using CNNs, the feature extraction is left to the deep neural network, yet using handpicked features for COVID-19 in conjunction with CNN's suggests leading to further improvements in terms of accuracy and sensitivity [97]. However, the implementation of such handpicked hybrid systems is specific to a given task, reducing the transferability of the achieved solutions.

With the FL framework, the need for a centralized training procedure would become obsolete, enabling easier data sharing between hospitals. Feki et al. utilize this approach in [98] and almost match the central setting with an accuracy of 97% in their collaborative process—with data split between four clients using ResNet50 models.

The presented works in this section do not ensure DP in their proposed models. Concerning the focus on PPML in this work, the non-private baseline's performance still holds importance in the context of the private models building on the same ML pipeline. After essaying the studies on comparable data, achieving an accuracy of close to 98% for the best non-private model would be the optimal starting position.

3.2. Training Private X-Ray Classifiers

What impedes the evaluation of works on privacy-preserving classification is that the investigations are not linked to a uniform privacy budget. Each study uses other values in its evaluation, reducing the general comparability of the results. This work expects $\varepsilon = 1$ for a sufficient privacy guarantee while other publications are less restrictive.

Abadi et al. presented the DP-SGD algorithm in [41] where they reached 90% at $\varepsilon = 0.5$ and 95% at $\varepsilon = 2$ on the MNIST task of classifying handwritten digits. Compared to the original model, the loss in accuracy amounted to 8.3% and 3.3%, respectively. [99] on the other hand, shows a private DP-SGD model that only loses by 1.3% to its non-private variant. Testing DP-SGD on the remotely related task of differentiating two kinds of X-ray images, as done with the approach in [100], revealed a loss of 10% when achieving the accuracy of 85% with a promised privacy guarantee of $\varepsilon = 1.64$. A recent PPML study on medical data utilized DP-SGD on a general chest X-ray classification task showing a steep utility-privacy trade-off with an average drop in accuracy of 35% from DP at $\varepsilon = 0.84$ [101]. Their results unveil limitations of methods for DP learning in health care. A paper by Singh et al. [102] classified pneumonia X-ray images using DP-SGD and obtained about 79% while getting 90% of non-private performance—a difference of 11%. They did not include the spent privacy budget, only publishing the used noise scales. The literature review unveiled no further studies using standard DP-SGD on the general classification of X-ray images, let alone COVID-19.

In the original paper by Papernot et al. [52], their PATE algorithm attained 98% accuracy at an $\varepsilon = 2$ on the MNIST dataset. This result runs only 1.2% behind the non-private version. Using PATE on the COVID-19 X-ray detection task at an ε close to 6, [103] reached an accuracy of 71% in the resulting private model while the original non-private model held 94.7%. That makes for a loss of 23.7%, all the while, a privacy budget of almost 6 is already significantly higher than the proposed target value of 1.

Literature on PPML in medical image classification commonly implements the FL framework. While the framework does not provide DP, it has room for applying additional mechanisms at different points. A recent work by Malekzadeh et al. [70] employs DP-SGD in the FL framework to guarantee DP. The DP-SGD operations do not apply to the batch level but the client level in the FL ensemble. This method loses about 4% at an ε of 11 compared to non-private methods. [104] performed FL in conjunction with DP-SGD on the task of pneumonia detection from X-rays reaching 85% at an unknown level of privacy and the accuracy was on average reduced by 12% when applying DP. Not conforming to the FL framework but also using collaborative learning strategies, Yuan et al. [105] add Gaussian noise to sharing parameters when learning for the same task of detecting pneumonia in X-ray images. Their accuracy loss at privacy budgets of 2 and 0.5 are only 4% and 6%, respectively, showing that a collaborative model can effectively learn with DP. Another study that merges DP-SGD and FL by Ho et al. [71] aims at COVID-19 detection and accomplishes a non-private result of 93.8%. When applying DP, they quickly noticed a performance loss that diminishes by increasing the number of clients. When using their highest number of 300 clients, they achieve 73.8% but at a mere privacy guarantee of $\varepsilon = 39.4$. From the theoretical point of view, 39.4 is no feasible privacy budget [35].

When comparing DP-GANs to other strategies, [61] showed that the results on the MNIST dataset slightly outperform the PATE algorithm by 0.3%, reaching 98.3%. Due to a lack of methods to track their theoretical privacy guarantees, the authors give an estimated privacy budget of 5.8, which is significantly higher than the ε of 2 used in the compared PATE analysis. If the estimation is correct, the GP-GAN would perform worse than PATE when run at the same privacy level. In another work [106] Zhang et al. introduce FedDPGAN, a DP-GAN which integrates into the FL framework. The scenario describes a DP-GAN that is collaboratively trained using FL, with the resulting model creating private training images for a ResNet-based student model. The approach achieved a performance of 94.5% on a COVID-19 detection task and even outperformed the non-private ResNet model by 1.5%. However, the study does not provide the accumulated privacy budget and only states the used noise scale.

Concluding this literature summary, the studies of PPML in X-ray classification are sparse. The most common method of DP-SGD is also the method that has not been evaluated on the COVID-19 task—at least not in a central setting. The few existing works on COVID-19 X-rays primarily focus on exploring the possibilities of one method and solely reference other publications in their evaluations. Such referencing is a standard research methodology; however, accurate comparability requires a benchmark dataset with a pre-defined test set to enable consistent results. Another problem spanning almost all presented work is the inconsistency in privacy budgets. Approaches are compared on different privacy budgets, yielding no valid comparison, while some evaluations even completely omit this information.

This work thus enriches the field of PPML in X-ray classification tasks by comparing multiple approaches in the same environment. All runs use the same dataset and adhere to the same privacy budgets. The evaluation focuses on the COVID-19 detection task, with

many privacy experiments even seeing their first evaluation on an X-ray task[5]. A commonality for all PPML methods surveyed in this section is their MNIST task evaluation. Thus, some models could be run on the MNIST dataset to investigate the differences to the COVID-19 task. Finally, this section showed that PPML on the COVID-19 task could result in an accuracy loss of 20.0-23.7% when keeping a reasonably low privacy budget. Other works with better privacy-accuracy trade-offs are either not private enough or do not state their final privacy budget. The same PPML methods only lose 1.2-4.0% on the MNIST set while keeping lower privacy budgets.

3.3. Repelling Membership Inference Attacks

The theoretical bound given by the privacy budget ε is the standard identifier for privacy in DP. These values give insights into the level of privacy a model guarantees. Still, to evaluate the real-life implications of these theoretical metrics, an empirical analysis of a private model's defense can be achieved by simulating attacks.

Regarding MIA defense, the expectations are that guarantees of DP limit the success of membership inference [75, 76]. Rahman et al. [48] tested this hypothesis by evaluating MIAs on different privacy budgets. Their non-private model found the MIA to correctly infer membership in over 60% of cases. At $\varepsilon = 8$ the success remains at 60% of samples, while at $\varepsilon = 4$ they still inferred 58.7%. The bigger gaps were located at $\varepsilon = 2$, with 53.3%, and $\varepsilon = 1$ with 50.7%. When considering that 50% would be the optimum, $\varepsilon = 1$ resulted in a strong defense against MIAs. In their paper, Rahman et al. explained the success of lower budgets against MIAs with the fact that models trained at lower ε guarantees exhibited less overfitting—overfitting being one of the primary culprits behind the success of MIAs as found in [75] by Shokri et al.

In addition to discovering the connection between overfitting and MIAs, Shokri et al. [75] state that the structure and type of the model contribute to the problem. Salem et al. [83] go further and demonstrate that even the training process can help with repelling MIAs. In their experiments, model stacking in the form of ensemble training, as seen in PATE, could reduce memorization in the final student model. When building a comprehensive study, [80] proved that overfitting in ML models is sufficient to enable MIAs. At the same time, a formal analysis showed that overfitting is no necessary criterion, and stable models can hold membership vulnerability. However dangerous MIAs may be, there are cases in which no protection though DP is needed to mitigate their effectiveness. This observation by Triastcyn and Boi [61] appeared when they wanted to evaluate the defense of their model. They found that the MIAs resulted in nearly random guessing accuracy even without protection.

Another strategy presented by Shokri et al. [75] showed that limiting the model outputs to only return class labels instead of confidence values was an effective and straightforward defense mechanism against MIAs. In a medical task like COVID-19 detection, where the use case is to help medical professionals in diagnosing an illness, the confidence value is an integral part of the output, indicating how certain the patient is affected. Thus, the remedy of rounding or omitting confidence values cannot be applied here.

[5]For an overview of the experiments see Chapter 4 Section 4.9.

Many relevant publications on PPML methods usually see no inclusion of actual attacks as an empirical metric [41, 52, 53]. Instead, the evaluation focuses on performance and accepts the theoretical privacy bounds set by ε. This notion is problematic because the stated privacy guarantees are difficult to assess regarding their actual defense against attacks. Although existing studies investigate MIAs relative to stated theoretical privacy budgets, a factor in MIAs could be the specific PPML implemented in the experiments. Further, the threat of MIAs might depend on the used dataset and its related task. For the first time, this work studies the relationship between DP and MIAs on a COVID-19 X-ray detection task—and probably on X-ray detection in general. Multiple PPML methods and architectures, with different experiments regarding architectural changes, are evaluated concerning MIAs.

4. Experimental Setup

The following sections constitute an overview of the whole experimental setup, implementing the needed processes and drawing conclusions on which experiments to run. Section 4.1 begins to unravel the problem of privacy-preserving detection of COVID-19 in X-ray images from an implementational and experimental point of view. The explanations include approaching the task by dividing it into sub-steps and the scientific manner of examining the resulting solutions. Section 4.2 continues by giving the software and hardware environment used to develop and run the whole system. Section 4.3 opens the explanations of the ML process by exploring the topic of data. Talking about data involves going over all relevant datasets and detailing the needed preparation steps before training on them. Section 4.4 then talks about the implementation of the non-private training procedure, which constitutes the first working prototype for COVID-19 classification. With Section 4.5 the goal then is to decide on suitable PPML methods to run in the experiments. Implementing the private ML workflow takes the non-private prototype as a starting point and adds the needed procedures to ensure DP for the trained model. To test the effectiveness of the privacy preservation, Section 4.6 talks about the MIAs directed against the model. Considering different MIAs, the goal is to find the strategy that results in the most considerable information leak from an attacked model. There are numerous possible model architectures to choose from when training image classifiers, with differences in layer count or type affecting the resulting model. This variety of options evokes the considerations in Section 4.7 for selecting the architectures for the experiments and eventual options for pre-training. Section 4.8 plays a central role because it goes over the actual settings and implementations used in the experiments. The section gives the hyperparameters for training each model and incorporates the PPML methods into the general ML process. In the final Section 4.9, an overview of all planned experiments concludes this chapter. It outlines the reasons for each experiment and lays out the proposed implementational and architectural changes.

4.1. Problem Statement

The privacy-preserving detection of COVID-19 in X-ray images consists of two principal problems. First, the ML task of training an image classifier for X-ray images to decide between COVID-19 positive patients and patients without findings (Normal). The second part includes introducing PPML methods to ensure DP and support the defense against attacks on the resulting private model. This defense should repel the attacks or render the leaked data useless. While solving both problems of classification and privacy achieves the general goal, the solely theoretical analysis in the form of DP limits the insights on the actual privacy level of the models. This work aims to secure the models against MIAs to prevent attackers from revealing individuals from the dataset. Thus, a third issue related to this work is the evaluation of real MIAs on the resulting classification models, which tests the hypothesis that DP helps in protecting against such attacks. Especially when working on medical tasks, an ML model is required to provide accurate diagnostics and strong privacy guarantees to give potentially helpful insights to medical professionals without compromising patient privacy.

Therefore, a key ambition is to reduce the possible gap in classification accuracy between non-private and private models that is effectually rooted in the accuracy-privacy trade-off present in PPML. The trade-off is that models with better privacy guarantees give less accurate results due to the introduced noise needed for DP.

Same as the problem statement itself, the implementation splits into solving separate building blocks stacked on top of each other to solve the overall problem. These subtasks consist of first developing a non-private COVID-19 classification, then adding privacy-preservation to attain private models, and finally attacking the models to test their privacy promises. Non-privately detecting COVID-19 in X-ray images can be broken down into a standard image classification task and solved using common ML strategies. Data generally constitutes a fundamental factor in achieving good ML results, making accumulating a COVID-19 dataset and its preparation for training an inevitable step. Using this dataset as a basis, constructing a non-private prototype allows testing different settings and hypotheses right from the start. The prototype should feature a working implementation of the complete ML process from loading the input data to achieving the training of a classifier. A functional prototype enables running tests on different training parameters and model architectures to find the best basic setup for the experiments. The next milestone is to build onto this non-private setup and integrate different PPML methods to reach private models. Again, arriving at this stage allows evaluating different settings for the algorithms. Including distinct candidates when selecting the PPML methods is essential to examine their differing influences. The final implementation work preparing the experiments focuses on MIAs. MIAs are a crucial part of the experimental evaluation and ask for some considerations to find the best strategy. While a basic attack might take the confidence values a model returns for the given image and solely decide based on a threshold, a more sophisticated approach would train attack models on the results of many image samples to better recognize differences in the target model's predictions.

All proposed approaches are systematically analyzed through experiments to obtain the best results. Instead of focusing on hyperparameter tuning in the actual experiments, such tests run on the non-private and private prototypes. In contrast, the experiments aim to evaluate more fundamental architectural changes to find the best performing and most secure models for non-private classification and each PPML method. The relevant metrics assess classification performance and privacy in different models, with the privacy aspect centered around MIAs. As a premier step, experiments for the non-private setting explore the best models to form a non-private baseline. In this case, the evaluation focuses more on performance than privacy. The baseline forms an anchoring point to compare the private results and determine the existing accuracy-privacy trade-offs. The evaluation of the private models evenly considers accuracy and privacy. The best contenders join the final evaluation that contrasts the PPML methods and non-private baseline. After only concluding the non-private and the different PPML approaches on their own, the final evaluation intends to give a comprehensive overview. The supplementary evaluation of some models on another dataset than the COVID-19 task validates and qualifies the gained insights. Running the same experiments in such a distinct setting tests their correctness and reveals differences to the original dataset. A final discussion of the evaluation then concludes by summarizing all findings.

4.2. Software and Hardware Environment

This section goes over the environment where all experiments are implemented and run. This documentation enables the results to be reproducible and traceable.

Regarding software, the environment relies on open-source tools to stay accessible. Thus, all implementations use open-source frameworks and libraries. This notion also advocates the use of publicly available datasets. The general programming language is Python, with ML workflows implemented mainly using the Keras [107] DL API, which runs on top of Google's Tensorflow[6] [108] framework. Further, private models employ features from the Tensorflow Privacy[7] library. In terms of software settings, most libraries in Python allow the specification of a fixed random seed, making it possible to draw the same random values every time. Keeping the same random seeds is particularly important to enable consistency and reproducibility since randomness is one of the key concepts in PPML. Therefore, any random seeds have a value of 42.

The corresponding code to this work is publicly available as Jupyter notebooks in a GitHub repository[8]. A Jupyter notebook is a web application that allows creating and sharing documents containing live code, coupled with equations, visualizations, and narrative text—all in a single file.

The hardware configuration is fundamental because it influences run times and enforces memory constraints, leading restrictions on ML settings like epochs, batch sizes, or even architectures. In this case, the Jupyter notebooks run on the Google Colab[9] platform. Google Colab offers to remotely run notebooks on their hardware infrastructure and provides capable computing resources, including GPUs. All runs are executed with a Google Colab Pro account using an NVIDIA Tesla P100 PCIe GPU and a high-memory environment of 25 GB.

4.3. Data

This section first delivers details on the four datasets relevant to this work. A final part then focuses on the data preparation steps each dataset receives before model training.

4.3.1. COVID-19 Radiography Database

The COVID-19 Radiography Database [90, 91] on Kaggle[10] is the most comprehensive collection of COVID-19 chest X-Ray images, stemming from different databases around the web. It was created by a team of researchers from Qatar University and the University of Dhaka and their collaborators from Pakistan and Malaysia in a joint effort with medical doctors. Next to COVID-19, it contains images for Normal and Pneumonia cases, with an example of each class presented in Figure 4.1. In total, this image collection offers chest X-rays of 3,616 COVID-19 positive, 10,192 Normal, and 7,357 Pneumonia cases—on the date of the experiments. Being the main dataset for the use case of differentiating COVID-19 vs. Normal,

[6] Available on GitHub: `https://github.com/tensorflow/tensorflow`
[7] Available on GitHub: `https://github.com/tensorflow/privacy`
[8] Available on GitHub: `https://github.com/luckyos-code/DP-X-COVID`
[9] Available at: `https://colab.research.google.com`
[10] See: `https://www.kaggle.com/tawsifurrahman/covid19-radiography-database`

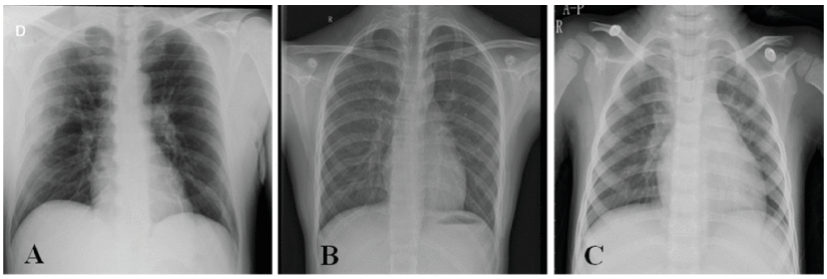

Figure 4.1.: Sample images from the COVID-19 Radiography Database: COVID-19 X-ray image (A), Normal X-ray image (B), and Viral Pneumonia X-ray image (C). Figure from Chowdhury et al. in [90].

only these two categories remain while omitting Pneumonia samples. Now, to directly mitigate some of the clear imbalances, the final dataset construction takes all COVID-19 scans but just 1.5 times the amount for Normal (5,424). The resulting 9,040 images are in PNG file format with a resolution of 299x299 pixels. The COVID-19 Radiography Database is the main dataset regarding the topic of this work.

4.3.2. Chest X-Ray Images (Pneumonia)

The Chest X-Ray Images (Pneumonia) [109] database is another Kaggle dataset[11] containing Normal and Pneumonia images. It has 5,856 X-Ray images in JPEG file format, divided into the two categories as 1,583 Normal and 4,273 Pneumonia samples. Again, to reduce the imbalance, the final dataset takes all Normal scans but just 1.5 times the amount for Pneumonia (2,374). This set can form a non-private dataset for a closely related transfer learning approach since it also poses an X-ray image classification task [41].

4.3.3. ImageNet

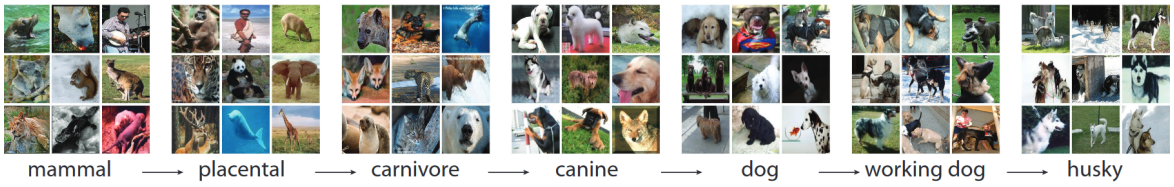

Figure 4.2.: Sample images from the ImageNet database with their labels. The graphic shows how sub-classes cover the many different images from general (mammal) to specific (husky). Originally seen in [110].

The ImageNet [110] is one of the most famous image database to date. With over 14 million hand-annotated images containing 20,000 categories, this massive project has significantly influenced computer vision research and helped discover the superiority of using DL with CNNs in image classification [18]. The content of the unequally sized images covers classes from general (mammal) to specific (husky), as exemplified in Figure 4.2. Other tasks can

[11] See: `https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia`

benefit from using this massive collection for pre-training, introducing many differentiating concepts to a neural network before training on the target data [111].

4.3.4. MNIST Database

A different topic is covered by the MNIST Database from LeCun et al. [17], which offers an extensive data collection of handwritten digits. In detail, these are images of the digits from 0 to 9 against a black background, as shown in Figure 4.3. The database provides 60,000 images for training and 10,000 for testing. In contrast to the other data sets, this classification task gives fewer details using tiny gray-scale images with a 28x28 pixels resolution. The MNIST dataset is a commonly used benchmark in image classification, especially in PPML, making it a perfect candidate for comparing results, e.g. [41, 52, 53, 85, 112, 113, 114]. The relatively short learning process stems from the less complex image data when compared to the other datasets.

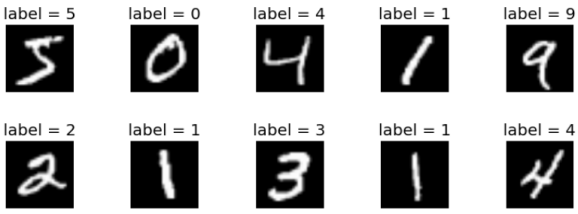

Figure 4.3.: Sample images from the MNIST database with their labels. The handwritten digits range from 0-9 and are given as gray-scale images against a black background.

4.3.5. Data Preparation

This work skips the step of gathering data by using existing data sets, but still, there are some essential options to make training more efficient. Data preparation is a fundamental stage of data analysis and ML, which can have a critical impact on achieved performance [115]. First, the data is split into training and test sets, where not already done. Hence, both X-ray datasets use a train-validation-test split of 80-5-15, i.e., 80% of images form the training set, 5% the validation set, and 15% the test set. Such partitions allow training, validating, and testing the model on distinct sets. The addition of a validation set can help in spotting and reducing overfitting on the training data [116]. Overfitting is one of the main caveats on smaller datasets like the X-ray datasets since there are fewer images for learning [87]. After constructing and splitting the dataset, the images need preparation for actual use with the ML model. Further, the data undergoes the standard image preprocessing steps of resizing the images to the desired size and normalizing inputs by scaling each image representation with a factor of 1/255. The MNIST images keep their 28x28 image size, while the other three datasets treat their images as 224x224 pixels. All datasets are taken with three color channels, where translating the MNIST greyscale images to the RGB space is vital to allow the models pre-trained on color images to still work with the input data. On another note, to not see the same order of training samples each epoch, the training set gets shuffled before handing it to the ML algorithm.

An essential tool to combat overfitting is data augmentation [117]. An image is subjected to small random changes, creating new images to artificially enlarge the dataset. It also helps with simulating more realistic conditions. The augmentations are solely applied to the training set because validation and testing must use consistent original data. While augmenting each image, the overall dataset size stays constant. Thus, each training set image gets randomly altered to be slightly different in every epoch, but the number of images presented to the model in each epoch rests the same. The applied random mechanisms and parameters are horizontal flip, rotation of 10%, translation of 10%, zoom of 15%, and brightness of 10%. The augmentation chooses a random value in the given limits. The settings are summarized in the following code snippet:

```
data_augmentation = Sequential([
  RandomFlip('horizontal'),
  RandomRotation(0.1, fill_mode='constant'),
  RandomTranslation(0.1, 0.1, fill_mode='constant'),
  RandomZoom(0.15, fill_mode='constant'),
  RandomBrightness(0.1)])
```

Random brightness alteration is particularly interesting for the use with X-ray scans. These settings can improve classifier performance while not altering the images too much. The slice from an augmented training batch of X-ray images in Figure 4.4 shows a possible outcome. Augmentations only apply to the X-ray datasets, not the ImageNet or MNIST databases.

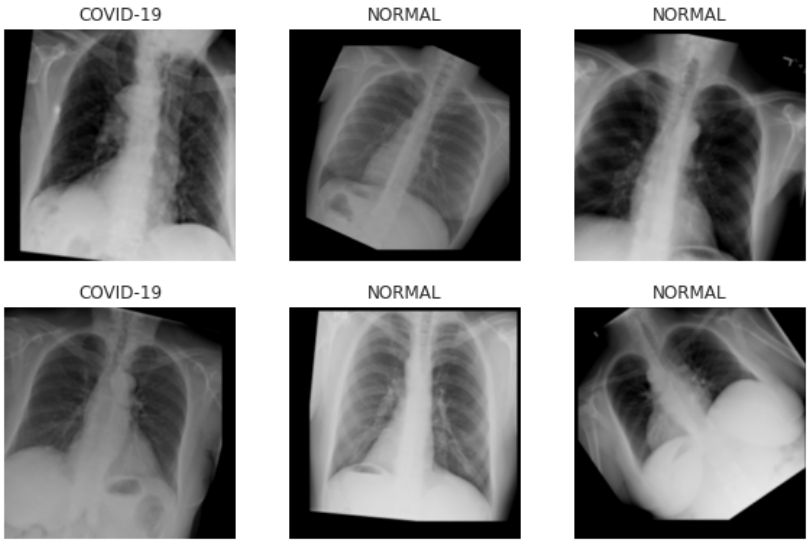

Figure 4.4.: Sample augmented images from a training batch with their labels. The applied random mechanisms and parameters are horizontal flip, rotation of 10%, translation of 10%, zoom of 15%, and brightness of 10%. The augmentation chooses a random value in the given limits.

An earlier problem stemming from the X-ray datasets' construction is the imbalance regarding sickness (COVID-19, Pneumonia) and Normal class frequencies. On the example of the COVID-19 set, Figure 4.5 displays the distribution of labels in the training, evaluation, and test sets. In total, the train, validation, and test sets contain 7,232, 452, and 1,356 samples, respectively. As stated in Section 4.3.1, all sets contain roughly 1.5 times the Normal

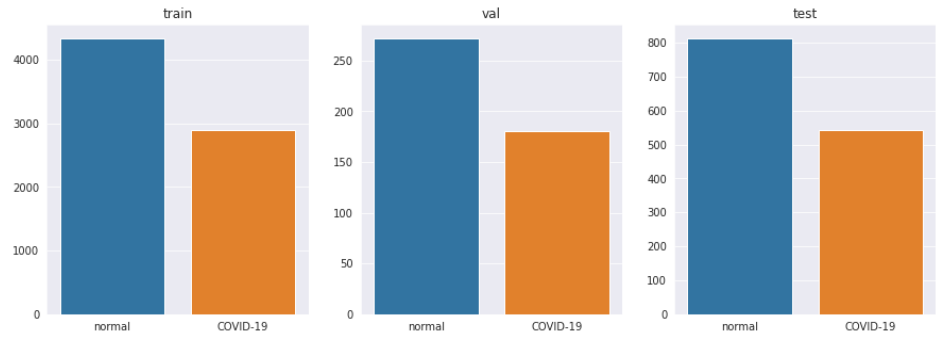

Figure 4.5.: Distribution of labels over training, validation, and test set for the COVID-19 data. In total, the train, validation, and test sets contain 7,232, 452, and 1,356 samples, respectively. X-rays of the Normal class amount to roughly 1.5 times the number of COVID-19 scans in each set.

images compared to COVID-19. Consequently, the model could favor the Normal cases while training, treating COVID-19 as a minority. Class weights can help artificially restore the balance in classifier training [118]. They are factored into the model's cost function on a per example basis to strengthen or weaken their specific influence on the learning process. When calculating the weights for each class, class frequencies are taken and translated into factors that equalize each class's impact. This process yields a class weight of 0.83 for the Normal and 1.25 for the COVID-19 class. These factors penalize the model when it classifies a sample as Normal, evening out biased classifications arising from the imbalanced data source. The corresponding class weights for classifying Pneumonia vs. Normal are 0.83 and 1.25, respectively, since Pneumonia contributes 1.5 times the Normal cases, see Section 4.3.2.

4.4. Non-Private Baseline

After preparing the data, the following steps are directed towards creating a non-private prototype to solve the classification task. The prototype forms the basis for the later inclusion of PPML methods. Implementing a working version of the ML process is a crucial first stepping stone. The ML process starts with successfully loading the data, continues with building a non-private model architecture, and ends with the successful training of a COVID-19 classifier. The Keras software library for Python is used in conjunction with the related Tensorflow ML framework to realize the ML pipeline. The general ML strategy employs DL with CNNs to realize the image classification. The actual architecture of the neural networks is irrelevant to the modular training process and can change for each experiment. A working modular prototype supports a first training and fast evaluation of different models, which is key to handling the fundamental problems of an ML task like hyperparameter tuning. Since the experiments in this work mainly aim at PPML and surveying changes in existing architectures, the basic ML settings are instead already tested on the prototype, decoupling the hyperparameter tuning from the actual experiments. The testing involves different batch sizes, learning rates, and epochs. The same deliberative process later exposes the best general settings for private training. The resulting training parameters and details are given in Section 4.8. The tests

on the non-private prototype also involve matching different model architectures to decide which ones to include in the experiments, with the results shown in Section 4.7.

Even though a leading aspect of this work is privacy preservation, the optimization of non-private training delivers essential insights. For one, it is vital to identify whether the task is already too hard without using PPML because not being able to build a feasible non-private model would make reaching the goal of successfully creating a private classifier highly unlikely. Further, some findings from the non-private experiments might also carry over their improvements to the private domain. In the end, one of the principal intentions of running the non-private experiments is to find the best models to compare to the private variants. The best non-private models constitute the non-private baseline, which is the best performance without PPML and is needed to determine the accuracy-privacy trade-off incurred from the private procedures. It also shows how experimental outcomes differ when applying them in different settings. In the non-private setting, the focus in selecting the best model is on the accuracy of classification rather than privacy metrics because models are trained without DP. Still, the most secure model also joins the final comparison to the private versions.

4.5. Discussing Privacy Preserving Algorithms

There are multiple ways of securing ML with DP, with some being general solutions, while others specialize in solving or correcting a specific problem. Therefore, an important step is picking the proper techniques for the given problem. Instead of running experiments with each PPML algorithm, the evaluation focuses on only a few, allowing more experiments per method. This section aims to find suitable competitors by navigating the advantages and disadvantages of each method and explaining why they are selected or rejected.

In a first step, algorithms solely resulting in private prediction are evicted as they do not allow secure model publishing. In addition, when applying privacy in the ML process, an empirical study by van der Maaten and Awni Hannun [45] revealed that even with private training (DP-SGD) reducing accuracy, aiming for private prediction (sample and aggregate) can perform worse in CNNs. This decision means methods like sample and aggregate are not considered. Thus, one relevant strategy (a) providing the appropriate security is privatizing the whole dataset or its labels before training the actual model. The other method (b) is applying DP through integration into the model's training process. These two options mainly leave DP-SGD(a), PATE(b), DP-GANs(b), and FL(a) as possible algorithms.

The simplicity of directly adding DP to the learning process by changing the optimizer to use noisy gradients makes DP-SGD easy to implement and enables its use in a wide range of ML tasks. The biggest drawback of DP-SGD is its induced accuracy loss. Nevertheless, different experiments promise to lower this gap with simple adjustments to model architectures, see 4.9. In general, DP-SGD is one of the most used PPML methods, but in the X-ray classification task, it is more commonly considered in an FL setting [71, 104]. The only sample in literature was run on pneumonia data reaching a private performance of 79% [102]. Including it here would yield a centrally trained candidate for comparison. Considering the importance of the DP-SGD algorithm in PPML research, it can be seen as a must-have for the evaluation.

Another common strategy for applying DP is PATE, which takes a different approach to private training than DP-SGD. Through decoupling the teacher ensemble and student model, PATE offers a framework that focuses on keeping the original labels of the training set private. By creating a student set labeled using noisy predictions from the teachers, the training process of each model itself rests untouched while still guaranteeing DP [52]. However, the noisy aggregation process is also PATE's main hindrance because reaching accurate labels needs a high enough teacher number to counter the introduced noise level on the labels. Another drawback of PATE on small private datasets is the need for training data reserved for the student. The data labeled by the ensemble must be unseen, which means it has to be set aside beforehand, leaving even fewer training samples for the teachers. On the COVID-19 task, related work achieved an accuracy of 71% at $\varepsilon \approx 6$ [103]. The completely different solution to private training makes PATE a logical contender for DP-SGD.

The main advantage of using DP-GANs is creating a DP-conform dataset of synthetic images with the same statistical properties. The post-processing immunity of DP then allows the training of non-private classification models on the data while still keeping DP [42]. Because a GAN can generate new images, one can artificially enlarge the dataset over the original level, which is highly relevant on small datasets like the X-ray databases. Still, a synthetic dataset does not mean that an MIA does not reveal original data. MIAs mainly attack the confidence values of the model and in this case, not necessarily the actual training images used. When building a synthetic dataset that is a statistical clone to the original, the resulting ML model trained on this data could be roughly equivalent to the one trained on the original data. The predictions could therefore still exhibit the same threatening notions exploited by MIAs. So, there might be no additional defense gained from creating new data with a DP-GAN other than the DP property itself. Therefore, DP-GANs would only be better than other PPML methods if the application of DP was more effective than in other algorithms. The achievement of DP in DP-GANs is commonly due to the use of DP-SGD during training [59, 60]. Using PATE is also possible, as seen in the PATE-GAN [119]. In general, training a DP-GAN rather than directly using a CNN architecture with DP-SGD or PATE has significant drawbacks. First, in most cases, a GAN's synthesizing task is a challenging ML task, resulting in a non-perfect replica of the original dataset [54, 55]. The application of DP then further reduces the abilities of the GAN because of the accuracy-privacy trade-off from the introduced noise [59, 60]. Consequently, training a classification model like a CNN on the generated data can ultimately result in worse accuracy than using the original data. Derived from this penalty, since PPML methods can be applied directly to a CNN and give the same privacy level without the additional performance hit from GAN training, the DP-GAN algorithm is less effective than direct usage of DP-SGD or PATE. Relying on DP-GANs might also lead to security issues as Liu et al. [120] explain, that "it is possible to hide [] sensitive identification data in the sanitized output images of such [D]P-GANs for later extraction, which can even allow for reconstruction of the entire input images, while satisfying privacy checks." These findings seem to apply to untrusted third-party tools only but negatively affect the general confidence in using DP-GANs. Still, tuning and investigating DP-GANs on X-ray images would harvest numerous experiments and possible insights. A recent publication employing DP-GAN in conjunction with FL achieved

a performance of 94.5% but did not give a privacy budget and instead only stated the used noise scale [106]. In conclusion, the listed disadvantages favor the direct usage of the other PPML algorithm for the given goal.

FL with DP seems like the perfect solution to the problem of medical data sharing. Each hospital can keep its data locally while securely sharing the gradients to train the overall model. Compared to central training, the resulting model experiences some accuracy loss but might gain access to more data [121]. While others employ forms of FL when using medical images, e.g. [98], the main difference to this work is that they rarely apply DP and instead aim for non-private models. Hence, FL here needs to include some form of PPML in the training procedure, which is most commonly the DP-SGD algorithm, but PATE could be a substitute [121]. Like DP-GANs, this results in the central and direct application of these PPML methods being the upper bound in performance because FL introduces additional loss through the decentralized learning scheme [70]. With DP-SGD inside the FL framework [71] achieved a private model with 73.8% but at a mere privacy guarantee of $\varepsilon = 39.4$. In conclusion, the framework of FL offers supporting features for this task. However, since the underlying algorithms to tune would be DP-SGD or PATE, their direct inclusion in a central setting is the better option for the evaluation. Also, FL is more of a general ML framework and works (for the most part) independently from the training parameters, still enabling its application to the findings of this work.

The overview of the relevant PPML methods concludes with DP-SGD and PATE as justified options. By only examining two PPML algorithms, the experiments can favor a focused and more profound research over the possibility of gaining a broader but shallow analysis. It further allows studying the differences between the application of noise to gradients vs. labels in this task. The evaluation then contrasts the two against the non-private baseline. Since DP-GANs and FL can utilize either method for their private training, the direct comparison of DP-SGD to PATE also gives a more fundamental benchmark that can still be useful later.

The implementations of DP-SGD and PATE take the non-private prototype from Section 4.5 and change the training process at the appropriate points to guarantee DP. Elaborations on the concrete settings and operations for both algorithms are located in Section 4.8.

4.6. Attacking the Classification Models

Before detailing the used MIAs, this section first answers why MIAs are the most threatening attacks to exploit models trained on the COVID-19 detection task. While there are many possible attacks, the relevant other threats to published models are reconstruction attacks and model inversion attacks [4]. Limiting the access to published models to black-box access naturally repels reconstruction attacks because they need white-box access to the inner model parameters to reconstruct the original data. In contrast, model inversion uses the confidence values returned by the target model's predictions and thus utilizes the same information as MIAs. Unlike MIAs, model inversion tries to mimic an input that returns the highest possible confidence value for a given class. For example, if a face recognition ML task differentiates between Harry, Hermione, and Ron in images of their faces, and the goal is to retrieve an image of Ron, the attacker would try to construct an image that is confidently classified as Ron

by the target model. The higher the achieved confidence value for the constructed image, the more it looks like the actual target. Eventually, the attacker might retrieve an accurate image of Ron. However, model inversion mostly poses threats to models that classify individuals in their classes. In COVID-19 detection, the only two classifications for a chest X-ray are Normal and COVID-19. When using model inversion to reconstruct an image for the COVID-19 class, the result would be an average COVID-19 image since the model trains on images of different individuals for the same class. Hence, a model inversion Attack would not be able to expose a specific image from the dataset, i.e., an individual, whereas an MIA threatens to achieve exactly that [75].

The decisive prerequisite for performing a successful MIA is some form of existing prior knowledge of the target data [75]. MIAs use the target model to predict given sample images and exploit the returned confidence values to determine which image was part of the original training data. A high confidence value indicates that the model has already seen the image in its training. An MIA reveals that an individual sample was a member of the original dataset based on this assumption. The prerequisite of prior knowledge means that the MIA can only test the membership of samples available to the attacker. So, the attacker needs to possess the individual sample to uncover its use in the target model's training. There are possible ways of synthesizing data for the attack, e.g., using model inversion attacks, but, as seen in the last paragraph, that would not allow revealing a selected individual.

With a perfect defense, MIAs can not distinguish between new samples and samples the model has already seen. One common strategy to counter MIAs is by avoiding exact confidence values through rounding or omitting them, only returning class labels [75]. In a medical task like COVID-19 detection, where the use case is to help medical professionals in diagnosing an illness, the confidence value is an integral part of the output, indicating how likely the patient is affected. Thus, the remedy of rounding or omitting confidence values cannot be applied here. The defense mechanism tested in this work is DP. The fundamental works [75] and [76] state that DP limits the success of MIAs by definition. Jayaraman and Evans [122] give the corresponding thought process: "[DP], by definition, aims to obfuscate the presence or absence of a record in the data set. On the other hand, [MIAs] aim to identify the presence or absence of a record in the data set. Thus, intuitively these two notions counteract each other."

The MIA implementation uses the Tensorflow Privacy library, which offers an MIA module and the capabilities to use different attack models. Two possible variants are threshold-based attacks and attacks using logistic regression. A threshold-based attack takes the returned confidence scores when predicting on a sample and decides about its membership based on a set boundary value that constitutes the threshold—a method based on [82]. When using ML in the form of logistic regression to carry out MIAs, the resulting attack model learns to predict the membership of a sample based on the target model's outputs on them. The utilized MIAs based on ML do not use the shadow models proposed by [75] (see Chapter 2 Section 2.3.2.4) since Salem et al. [83] found that using the original model's predictions on the samples is sufficient to deduce their membership. Thus, no approximation of the original model is required. In general, when provided with the needed knowledge, ML is expected to

be slightly better than threshold-based attacks in MIAs, making logistic regression the choice for the experiments [80].

The performed MIAs access two kinds of data from the target model: (i) original training data and (ii) data not seen in training but comparable to the original. The first kind is satisfied by taking the original training set, and the second by taking the original test set. The attacker gets access to the full original training set to emulate the worst-case scenario. Attacking with the Tensorflow Privacy library can try different data slices in an attack. Other than the entire dataset, the MIA tries slicing by class and by classification correctness to achieve the best possible attack. In addition to training and test data, the attacker also knows the correct label corresponding to each sample to evaluate the attacks results. The last relevant inputs for the MIA are the model's predictions and losses on all given samples. In summary, the MIA implementation utilizes the labels, losses, and predictions for all training and test samples to find and evaluate the best attack pattern.

Since individual MIA results are subject to greater variability, they need to be experimentally stabilized to see a statistical significance. Stabilization is achieved by running 100 attacks and calculating the corresponding 95% Confidence Interval (CI). O'Brien and Yi [123] define how to interpret such values:

> A 95% [CI] of the mean is a range with an upper and lower number calculated from a sample. Because the true population mean is unknown, this range describes possible values that the mean could be. If multiple samples were drawn from the same population and a 95% CI calculated for each sample, we would expect the population mean to be found within 95% of these CIs.

4.7. Selecting Model Architectures and Pre-training

Non-private and private classification perform differently depending on the underlying model architecture [124, 125]. Performance is greatly dependent on model size, i.e., the number of layers or parameters in a model, with non-private training typically benefiting from using bigger models. In contrast, using DP-SGD, the same bigger models can suffer from higher accuracy loss than their shallow counterparts. This difference in behavior between non-private and private training when increasing parameter counts for the model architecture is depicted in Figure 4.6. In both case studies, the non-private setting using SGD keeps gaining accuracy with higher parameter counts (equivalent to higher number of filters k) while the DP-SGD model's accuracy quickly declines after reaching an early maximum. As a result, the experiments use multiple architectures with different sizes to find the best-suited candidate. Further experimentation then tests other architectural variations and settings on the basic architectures. The experiments are discussed in Section 4.9, while this section presents the basic models chosen and evaluates possible pre-training options for them.

ResNets. The model family of Residual Networks, or ResNets, is one of the best scaling deep CNN architectures thanks to its residual connections [126]. They exist in different sizes, where the number in the name discloses how many layers the network contains. The ResNet50 version boasting 50 layers is already a stretched compromise between size and computing

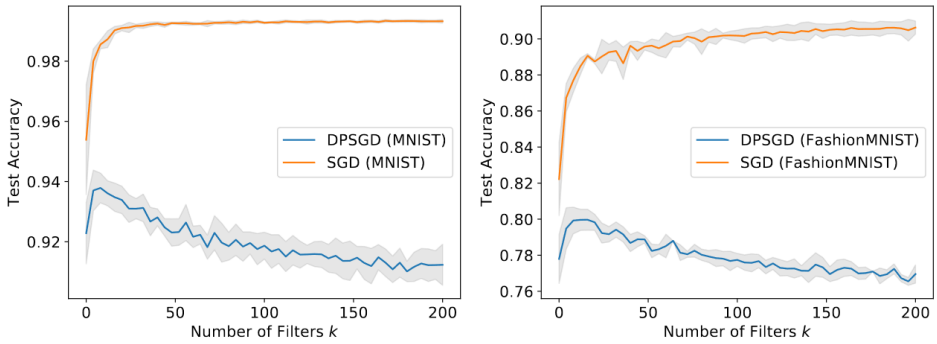

Figure 4.6.: Test accuracy as a function of the number of filters k (equivalent to changing parameter count) when training with vanilla SGD and DP-SGD. Each point corresponds to multiple training runs on MNIST (left) or FashionMNIST (right). For both datasets, adding filters always improves non-private learning, whereas after an early point they are not beneficial to learning with privacy. Originally published by Papernot et al. in [125].

needs because it can push the available computing capacities for this work to their limits. The high resource needs also stem from the resolution of the X-ray image data, which results in a resource-hungry training process. The ResNet18 is a smaller ResNet version that is more affordable in computing needs. Except for the lower count of layers, it is identical to its bigger brother.

Shallow network. The third tested architecture is specifically relevant for the DP-SGD algorithm. The theory behind this approach is inspired by the discovery of Bassily et al. in [124] that shallower Networks (containing fewer layers) can perform better than their deeper counterparts when using optimization algorithms like DP-SGD. Papernot et al. proposed one such successful shallow network structure in [112], where they designed it for the MNIST database. Table 4.1 describes the adapted version for this work, where layers use the ReLU activation function when applicable. This smaller model might disappoint when used in non-private classification but could shine in the DP-SGD setting.

Table 4.1.: Shallow network architecture for DP-SGD, which is inspired by the work of Papernot et al. in [112]. The output layer depends on the given task and in the shown configuration enables binary classification into 2 categories. Layers use the ReLU activation function when applicable.

Layer	Parameters
Convolution	16 filters of 8x8, strides 2
Max-Pooling	2x2
Convolution	32 filters of 4x4, strides 2
Max-Pooling	2x2
Fully connected	32 units
Sigmoid	1 unit

Other networks. The use of deeper architectures than the ResNet50 would be even more taxing on memory and probably exceed the hardware limits. However, there are models like the EfficientNetB4 [127], which has fewer parameters than the ResNet50, while achieving better performance on the ImageNet task through new model scaling techniques. The DarkCovid-

Net [1] on the other hand, is a smaller but specialized network for COVID-19 classification, making it particularly interesting in this task. Models were run on the non-private prototype setup to test all contenders' abilities. In this implementation, the ResNet18, ResNet50, and EfficientNetB4 use weights pre-trained on the ImageNet dataset provided by the Keras API. The results pointed out in Table 4.2 show the five architecture candidates with their respective model size in the form of total parameters and their accuracy on the COVID-19 task. The EfficientNetB4 reaches 95.7%, landing behind the ResNet50 by 2.1%, which performs best at 97.6%. The smaller ResNet18 follows closely at 94.0%, just slightly outperforming the ten times smaller DarkCovidNet that situates itself at 91.9%. The shallow network performs worst with only 79.3%, suggesting that the model size might not be enough for the non-private setting. This small-scale study shows higher accuracy results with increasing parameters. Because the EfficientNetB4 could not beat the ResNet50, both ResNets stay in the experiments. Their structure only differing in layer count allows for a better investigation of the influence of architecture deepness on the final results. There is no benefit from additionally including the DarkCovidNet based on raw performance. Taking it for its size seems unnecessary, considering the ResNets are the best candidates to spot any isolated influence of size on performance. The extensive research and literature on ResNets also provide better comparability, which is shown by ResNets being the CNN architecture with the most published papers[12] when querying Papers With Code, a website that collects the latest state-of-the-art ML papers and code [128]. In terms of specialized models for private training like the shallow network for DP-SGD, no other significant architectural development was found to directly improve the expected outcome for the PPML methods. The final selection of architectures for the experiments includes the ResNet50, ResNet18, and shallow network.

Table 4.2.: Comparison of possible model architecture candidates matched by their accuracy reached when running them in a prototypical non-private implementation of the COVID-19 classification. The parameters column refers to the total count of parameters of a model as a metric for model size. ResNet18, ResNet50, and EfficientNetB4 use weights pre-trained on the ImageNet dataset.

Model	Parameters	Acc.
Shallow network	0.18M	79.3%
DarkCovidNet	1.17M	91.9%
ResNet18	11.19M	94.0%
EfficientNetB4	17.68M	95.7%
ResNet50	23.55M	97.6%

Pre-training. Pre-training can give each of the chosen models a possible head start [24]. An important factor here is that the datasets used for pre-training can be public while the actual dataset of the target task has to stay private. A common way of pre-training in CNNs is to simply use the extensive ImageNet [110] collection and let the model train on it, hoping that it can gain useful general insights for working with images [111]. A more refined approach takes a closely related task to instill knowledge more relevant to the target task [41]. In this case, the perfect candidate is the task of detecting pneumonia in X-ray images, for which the

[12]With 1,387 published papers on ResNets and only 354 on the CNN in the second place on the list.

Chest X-Ray Images (Pneumonia) [109] database delivers the needed public dataset. There is a medical connection between the two tasks of classifying pneumonia and COVID-19 from X-ray images since a COVID-19 infection can involve a viral pneumonia [129]. Therefore, the features learned from the pre-training should roughly correspond to each other, making fine-tuning for the actual task easier. To summarize, in addition to the basic version of each architecture, experiments also consider versions pre-trained on the ImageNet and Pneumonia databases.

4.8. Hyperparameters and Details of Model Training

This section goes over the general training settings for all architectures and methods. They are mainly derived from tests on the prototype and take indications from related works. Other than hyperparameters, this section includes general details for non-private and private training, especially for DP-SGD and PATE implementations.

Epochs. If not stated otherwise, every model trains for 20 epochs, which turned out to be the best compromise to reach a plateau in performance while at the same time limiting training time. As a side effect, choosing a small number of iterations helps limit the privacy loss in DP-SGD, since for DP, the privacy loss increases with the times each training image occurs [130].

Output layer. For the main binary classification task on the X-ray images, the models are equipped with a 1 unit fully connected layer with sigmoid activation as binary output and take binary cross-entropy to calculate their loss.

Optimizer and learning rate. In general, all implementations use the Adam optimizer instead of a SGD optimizer. This decision is mainly based on Adam's adaptive properties, enabling it to perform similar to a task-adjusted SGD [112]. This advantage makes Adam the best overall optimizer for the many different models in the experiments. The starting learning rate for each scenario is 1e−3, which was found to consistently deliver a high plateau at the set number of 20 epochs. After starting on this value, the learning rate decays on plateaus, i.e., if the loss on the validation set does not improve for two consecutive epochs, the learning rate is reduced by a factor of 0.1. This process can bring the learning rate to a minimum of 1e−6. The decay enables finer steps in the parameter updates when getting further in the learning process [131].

Batch size. The batch sizes differ from method to method and model to model. This discrepancy is due to the varying memory needs, which differ by model size but are especially tough in DP-SGD due to batch-level operations, like clipping and noising. The resulting batch sizes for methods and models are summarized in Table 4.3.

DP-SGD. The Tensorflow Privacy library provides a DP-SGD implementation in the form of a readily built optimizer. The DP-SGD algorithm stacks on top of the existing non-private training by changing the optimizer and handing in the necessary DP-SGD parameters. Although referenced as DP-SGD, the employed optimizer is the accordingly modified DP-Adam version, which follows the same reasoning for using the non-private Adam instead of SGD. The used learning rate rests the same as in non-private training. When selecting the parameters for the private training, the needed noise scale varies according to the desired

Table 4.3.: Overview of batch sizes used with different architectures and methods. The differences are due to the varying memory needs, which differ by model size but are especially tough in DP-SGD due to batch-level operations, like clipping and noising.

	Batch size		
Model	Non-private	DP-SGD	PATE
ResNet50	32	8	32
ResNet18	32	16	32
Shallow network	32	32	32

privacy budget ε, which draws from a Gaussian noise mechanism. A clipping norm of 1.0 was identified to maintain effective training when clipping the gradients. Lastly, the micro-batch size on which DP-SGD operates is the same as the general batch size used in training the model.

PATE. The needed steps for PATE are implemented by hand. The experiments utilize the same model architecture for all teachers and the student[13]. The best-performing non-private architecture should be selected since PATE exploits a standard training procedure by only privatizing the labels through noise afterwards. Although the ResNet50 seems to perform better than the smaller ResNet18 in the non-private settings, there might be no other choice than to opt for the best ResNet18 version. This limitation is due to the computing resources needed to train higher teacher numbers using the large ResNet50. At the same time, the shallow network should not be able to top the ResNet18 in the non-private setting. Training is run for 100 epochs for each teacher instead of 20 to better accustom them to their smaller data share. Data handling in PATE has to be slightly different from before. The algorithm separates 999 images from the original training set to constitute the student training data, prepared the same way as the regular training data. The student data forms the unlabeled public data that is then privately labeled by the teacher ensemble. The remaining training images are split into distinct sets according to the wanted number of teachers, for which the small dataset introduces strict limits. When using 25 and 50 teachers with the COVID-19 dataset, each split shrinks to 249 and 124 images, respectively. At the same time, the amount of teachers is still relatively low for achieving strong privacy guarantees, since budgets of 0.1, 1, and 10 already ask for noise scales of 2,720, 278, and 33, where noise is drawn from the Laplace noise mechanism. With only a maximum of 50 votes from the teachers, the large noise scale can render the transferred knowledge useless. The limits regarding data and noise make choosing less than 25 or more than 50 teachers unsustainable. With less than 25 teachers, the noise impact would be too high, and with more than 50 teachers, the data share for each teacher would be too small.

Pre-training. To transfer knowledge from a source task, a model has to be trained on the relevant public dataset, preferably using the same architecture it later uses on the target task—here the COVID-19 set. The only exception is the output layer, which is not copied and can thus change depending on the task. When a source model finishes training on the pre-training set, all layer weights other than the output layer are copied to the same layers in

[13]When taking advantage of pre-trained models, all teachers and the student are pre-trained.

the target model. The target model then starts training on the actual target dataset using the pre-tuned weights to fine-tune them. All model layers are unfrozen, i.e., trainable, to allow this tuning. The pre-training on the Pneumonia dataset is done by hand using the same settings as on the COVID-19 set, while the ImageNet variants are provided by a Python library that extends the existing range of Keras models[14].

MNIST. Running the models on the MNIST dataset and its related task requires some modifications to the training setup. When training on the MNIST dataset, models switch to multiclass classification to differentiate the ten digits and thus use a 10 unit fully connected layer with softmax activation as output. This change is accompanied by the switch to the categorical cross-entropy loss instead of the binary version. Except for the fixed batch size of 32 in all runs, the training operates under the same parameters as before.

4.9. Investigating Performance and Privacy with Experiments

A practical approach to investigating hypotheses in the form of many different models and methods is by running experiments. Experiments deliver the empirical results needed to match the tested candidates by metrics, evaluating the related hypotheses. This section summarizes all experimental efforts. Figure 4.4 provides an overview of the experimental pipeline on the COVID-19 set and illustrates how the different architectural experiments unfold from the compared methods. Non-private, DP-SGD, and PATE training form the root from where the tree branches out to the general architectures, followed by the specific settings in the tree's leaves. The diagram shows a total of 44 architectural experiments, with each path from the root (left) to a leaf (right) representing a complete experimental setup. The different privacy budgets and experiments on the MNIST dataset are not included in the diagram.

Datasets. The experiments are mainly run on the task-relevant COVID-19 Radiography Database [90, 91] as the evaluation basis. Any pre-training uses the respective public datasets of ImageNet [110] and Pneumonia [109]. Models will be selected to also run on the MNIST database to verify the findings from the COVID-19 set. Together with the CIFAR-10 dataset [132], the MNIST database [17] is one of the standard image classification tasks. The related work in PPML on these datasets can deliver more perspective on the achieved numbers and feasibility of each approach, e.g. [41, 52, 53, 112, 85, 113, 114]. Since the shallow network was designed for the MNIST task, this database gives an interesting comparison [112]. Consulting another dataset to test the existing hypotheses can be seen as an experiment in itself.

Architectures. As previously presented in Section 4.7 the basic model architectures compared in the experiments are the ResNet50, ResNet18, and shallow network. The most notable difference between these models is their size. The inclusion of different sizes should test the relation between size and performance in non-private and private training [124, 125]. Each architecture changes according to the hypothesis tested by the experiment. The architectural experiments aim to find the best-performing models for each category (non-private, DP-SGD, PATE). The selected models across all approaches are then compared in a final evaluation in

[14] Available on GitHub: `https://github.com/qubvel/classification_models`

Table 4.4.: Overview of the experimental pipeline on the COVID-19 database. Non-private, DP-SGD, and PATE training form the root from where the tree branches out to the general architectures, followed by the specific settings in the tree's leaves. The diagram shows a total of 44 architectural experiments, with each path from the root (left) to a leaf (right) representing a complete experimental setup.

Non-Private	ResNet18	Standard
		Dropout
	ResNet50	ImageNet
		Pneumonia
	Shallow network	Standard
		Dropout
		BN
		BN-Pneumonia
DP-SGD	ResNet18	Standard
		Dropout
		ImageNet
		Pneumonia
	ResNet50	tanh
		tanh-Dropout
		tanh-Pneumonia
	Shallow network	Standard
		BN
		BN-Dropout
		BN-Pneumonia
		tanh
		tanh-BN
		tanh-BN-Dropout
		tanh-BN-Pneumonia
PATE-25	ResNet18	Standard
		ImageNet
	ResNet50	Pneumonia
	Shallow network	Standard
PATE-50	ResNet18	Standard
		ImageNet
		Pneumonia

Chapter 5 Section 5.3. Taking different base models enables seeing any influence implied by architectures, especially size.

Privacy methods. As selected in Section 4.5, two PPML methods are evaluated. DP-SGD and PATE compete with each other but also with the non-private baseline. Having at least two different private approaches helps identify the ideal way of protecting the models from MIAs while keeping high accuracy.

Privacy budgets. Since only privacy budgets of $\varepsilon \leq 1$ represent good privacy guarantees, $\varepsilon = 1$ is the main focus for the evaluation [35]. Still, evaluating other budgets gives more insights into the performance on different privacy levels, and therefore the final evaluation also considers budgets of $\varepsilon = 0.1$ and $\varepsilon = 10$. The setup shows the development of the accuracy-privacy trade-off over different theoretical privacy levels. Further, including more budgets can enable a clearer analysis of the budgets' influence on MIA success. Privacy analysis employs the moments account to track moments of privacy loss. In correspondence to the COVID-19 dataset size, the privacy guarantee needs to satisfy $\delta \leq 1/n$ with $n = 9,040$, resulting in $\delta = 1\text{e}{-4}$. Table 4.5 lists the the relevant noise scales for each method and batch

Table 4.5.: Needed noise scales to achieve privacy budgets ε of 10, 1, and 0.1 for the different methods and batch size (i.e. architecture) combinations. With respect to the batch sizes listed in Table 4.3.

		Noise scale		
Method	Batch size	$\varepsilon = 10$	$\varepsilon = 1$	$\varepsilon = 0.1$
DP-SGD	8	0.455	0.882	4.460
	16	0.489	1.023	6.269
	32	0.530	1.267	8.836
PATE	32	33.0	278.0	2720.0

size (i.e. architecture[15]) combination to achieve the wanted ε values. On the MNIST database, the amount of data asks for $\delta = 1e{-}5$ and accordingly changed noise scales.

MIAs. While measuring the success of the COVID-19 detection relies on testing the classifier on relevant X-ray data, the actual privacy of an ML model is only vaguely captured. The analysis of privacy budgets for a DP-ensuring algorithm just provides a theoretical upper bound to the privacy loss. To directly monitor the proneness to attacks of an ML model, the theoretical analysis is accompanied by the empirical study of the defense against MIAs. Adding empirical results allows for a more realistic and practical approach to privacy evaluation. The proneness to attacks is one of the main evaluation criteria in this work because one hypothesis is that DP reduces the risk of MIAs [48, 75, 76].

BN. ResNets already contain BN layers while the shallow network does not, at least in its original version. Empirical investigations by Davody et al. in [133] revealed that it helps with combating perturbation in DP-SGD. They argue that the invariance of the model to weight matrix rescaling obtained from normalization methods offers robustness against noise injection. For this experiment, after each trainable layer a BN layer is added to the shallow network—i.e., after the convolutional and fully-connected layers.

tanh activation. A recent discovery in DP-SGD research was made by Papernot et al. in [112]. They have determined that replacing the ReLU activation function with the tanh function in model layers improves performance in DP-SGD. To achieve this performance boost, they utilize the fact that the tanh activation generally results in smaller gradients than the ReLU function. Smaller gradients reduce negative effects from the gradient clipping in the DP-SGD algorithm because less information is lost. This advantage of tanh is only relevant in DP-SGD training, and this experiment is hence only applied to DP-SGD models.

Dropout. Szegedy et al. state the performance benefits of dropout in [134] and specifically focus on ResNet-type networks in their analysis. The regularizing effect of dropout can help a model in overcoming overfitting and thereby in learning a better generalization [19]. However, in his paper [135] Galinkin lays out that adding dropout layers in models could negatively affect the defense against MIAs. At the same time, statements in [83] and [136] describe dropout to be an effective measure against MIAs due to the mitigated memorization from reduced overfitting. One experimental setup adds a dropout layer of 0.2, i.e., with a 20% chance of dropout, before the output layer to test these hypotheses.

[15] With respect to the batch sizes listed in Table 4.3

Pre-training. General pre-training is commonly used to improve performance in non-private classification [111]. However, Abadi et al. [41] already found that public pre-training specifically in a related domain can also improve performance in DP-SGD models. The pre-training on the ImageNet and Pneumonia databases tests possible benefits for non-private and private training. While the ImageNet resembles more of a general choice in this domain, the Pneumonia database is closely related to the target task of COVID-19. Pre-training experiments further unveil if the pre-training influences a model's proneness to attacks.

PATE teacher numbers. When having fewer total teachers, the data share for a single teacher increases accordingly, improving the resulting training performance. At the same time, the teacher number also determines the maximum possible votes in the noisy aggregation process. Thus, having fewer total teachers increases the influence of the introduced random noise on the labels. Finding the sweet spot for the teacher number is difficult, especially since there is no general formula—the optimal number changes with available data and privacy needs. The experiments only compare teacher ensembles of 25 and 50 teachers to try and maintain a viable trade-off regarding the small dataset. Section 4.8 offers more details to the reasoning behind this choice.

Lucas Lange
3737104

5. Evaluation

The evaluation is divided into several parts, beginning with a presentation of the metrics used to evaluate model performance and privacy in Section 5.1. Section 5.2 then elaborates on the individual experimental results regarding non-private, DP-SGD, and PATE models. The most successful versions from these three settings are then matched against each other in Section 5.3. Following on the COVID-19 database, Section 5.4 takes the detour to the achieved results on the MNIST task. A final discussion of the findings and gained insights from all experiments is presented in the concluding Section 5.5.

5.1. Metrics

Metrics are an essential tool for evaluating different experiments. They enable comparability and allow the tracking of changes in their underlying measurements. To evaluate the success of the various models, metrics need to assess their effectiveness. They must provide consistent insights, with their significance independent of the single model. They thereby enable to compare different approaches, making them fundamental for any evaluation.

For a non-private classification task, the most relevant numbers are tied to the performance of a model. The better classification is the one that can best distinguish the target classes. This ability is mainly quantified by the classification accuracy, the percentage of correctly classified examples over all given examples. Nevertheless, this value alone might be misleading because neural networks can have unexpected ways of achieving it. For example, if the one class represented by a 1 is predominant in a dataset and constitutes 60% of all examples, a neural network could just classify everything as a 1. It would reach an accuracy of 60%, even though it could perform less effectively when predicting on differently balanced real data [137]. Two other metrics are used to spot and analyze such problems: precision and recall. Precision is the proportion of positive identifications that was actually correct, while recall is the proportion of actual positives that were identified correctly. Both show how relevant the model's classification is. Seeing extreme values in either direction indicates a possible problem. Precision and recall are not directly included in the evaluation but used as checks to guarantee a correct training procedure for the models. There is a special case of the model being unable to learn a functional classification in private training because the induced random noise is too high. Such problems can also be spotted using precision and recall since models still delivered some accuracy in the experiments, but precision and recall were zero in such instances. Whenever this occurs in the evaluation, the accuracy is given as NA, and no attacks are performed as it would not hold any significance. Also, considering that a part of model learning is based on memorizing already seen examples, taking metrics on the training setor even validation set is no indicator of actual performance [138]. Hence, all metrics are calculated while training to monitor progress but are finally evaluated when predicting on the test set.

The general measurement of DP is the privacy budget ε. It gives an upper bound for the possible privacy loss and is solely based on theoretical analysis. However, the realistic bounds in a given scenario might differ substantially from the theoretical worst-case. It further does

not constitute a metric to assess the proneness to MIAs of a model, which is the main point of the comparison regarding privacy in this work. For MIAs, the attacker's AUC constitutes the most representative metric available. It represents the probability that a randomly chosen positive sample is ranked higher than a randomly chosen negative sample [84]. For MIAs it says how successfully the attacker can recognize which example was part of the original training set. An optimal defense results in an AUC of 0.5 or lower, which translates to an attacker randomly guessing the membership of a sample. An AUC of 1 means that an attacker can always identify the correct membership. Since the attacks are run 100 times to counter variability, a 95% CI over all AUC results is used in the evaluation. To get a bigger picture and investigate if DP with different privacy budgets represents different defense levels against MIAs, another metric relates AUC and privacy budget. The empirical privacy budget based on the AUC given as ε_{AUC} takes the AUC values achieved by the attacks and uses them to estimate the corresponding privacy guarantee of the target model. An AUC of 1.0 results in $\varepsilon_{AUC} = \infty$, while values lower than 0.5 translate to the perfect guarantee of $\varepsilon_{AUC} = 0$. In between these boundaries, the epsilon is estimated based on the following formula: $\varepsilon_{AUC} = \ln(\frac{AUC}{1-AUC})$. This notion is adapted from the recent work of [85], where they propose this metric based on the attacker's accuracy. With a value given as a privacy budget, its comparison to the theoretical assumptions is more intuitive. Further, privacy budgets are a finer gradable metric than the AUC scale. Using this specific metric, even the smallest differences between models are exposed, and the proneness to MIAs can be directly related to a privacy budget.

5.2. Evaluating with Experiments

The following three sections cover the experimental results for non-private, DP-SGD, and PATE trained models. Each of these three settings is presented separately to first find the best respective models in terms of accuracy and privacy. An evaluation spanning these extracted models for each setting takes place in Section 5.3. The models are compared by the metrics from the last Section 5.1, consisting of accuracy, ε_{AUC}, and to some extent, the different privacy budgets ε.

5.2.1. Evaluating Non-Private Models

The evaluation starts by searching for the best non-private models situated at $\varepsilon = \infty$ that define the baseline. The relevant tuning results are found in Table 5.1.

ResNet18. First, the focus is on finding the best ResNet18 variant, which might also carry over to the ResNet50 because of its architectural similarities. When looking at the accuracy as listed in Table 5.1, the ResNet18's best-performing version received the ImageNet pre-training, resulting in 94.0%. The standard version only reached 91.4%. In contrast to the ImageNet pre-training, the experiment of adding dropout to the model slightly reduced accuracy compared to the standard version by 0.7%, while the pneumonia pre-training resulted in an even steeper decrease in performance to 86.2%. The ImageNet dominance even carries over to privacy, which is evaluated in the ε_{AUC} column of Table 5.1. The ImageNet variant delivers

Table 5.1.: Overview of experimental results for non-private models. Models are matched by their test accuracy in % and proneness to MIAs. The proneness is measured by the empirical privacy budget regarding the AUC (ε_{AUC}) and is given as a 95% CI over 100 attacks. Non-private training translates to $\varepsilon = \infty$.

		$\varepsilon = \infty$	
Model	Variant	Acc.	ε_{AUC}
ResNet18	Standard	91.4%	0.21-0.24
	Dropout	90.7%	0.22-0.26
	ImageNet	94.0%	0.17-0.21
	Pneumonia	86.2%	0.19-0.23
ResNet50	Standard	85.1%	0.23-0.26
	ImageNet	**97.6%**	0.13-0.16
Shallow network	Standard	79.3%	**0.09-0.12**
	Dropout	75.7%	0.12-0.16
	BN	86.0%	0.13-0.16
	BN-Pneumonia	86.4%	0.13-0.15

the best defense against the MIAs at $\varepsilon_{AUC} = 0.17 - 0.21$. Pneumonia pre-training lowered the empirical budget regarding the standard model from 0.21-0.24 to 0.19-0.23. Dropout, on the other hand, resulted in a model that is more prone to attacks with $\varepsilon_{AUC} = 0.22 - 0.26$.

ResNet50. The ResNet50 results paint the same picture as the one for the ResNet18, with dropout and pneumonia performing worse than the ImageNet. Because the dropout and ImageNet experiments do not contribute new findings, Table 5.1 only includes the standard version and the one pre-trained on ImageNet. The standard ResNet50 achieved 85.1% accuracy and guaranteed empirical privacy of 0.23-0.26. The ImageNet pre-training ensured a steep increase in performance, delivering the peak accuracy of 97.57%, and helped privacy with a significant drop to $\varepsilon_{AUC} = 0.13 - 0.16$. As a difference to the ResNet18 version, the ResNet50 pre-trained on ImageNet was found to perform better when lowering the starting learning rate to 1e−5 instead of 1e−3. By that, the propagation of the parameter updates through the many layers can be slowed down to better adapt each layer to the new task before the pre-trained weights are overwritten [28].

Shallow network. The standard shallow network variant gives a strong empirical privacy guarantee of 0.09-0.12 while only offering mediocre performance at 79.3%. Again, dropout noticeably worsens the model's results on both metrics. Adding BN to the standard model significantly boosts accuracy by 6.7%, but also increases ε_{AUC} to 0.13-16, a comparable increase to dropout. BN was mainly implemented as an experiment to improve DP-SGD performance [133]. However, the performance lead compared to the standard version is large enough to justify the slight rise in ε_{AUC} and lead to using it as the main non-private variant. Therefore, pneumonia pre-training already uses a shallow network version incorporating BN. The model combining BN and pneumonia pre-training sees the best shallow network accuracy of 86.4% and faintly lowers the empirical privacy budget regarding the BN version from 0.13-0.16 to 0.13-0.15. The standard shallow network stays the most secure.

Summary. Winner in the non-private setting is the ResNet50 pre-trained on ImageNet, giving the best performance overall with an accuracy of 97.57% paired with the best privacy of the ResNet models at a 95% CI of $\varepsilon_{AUC} = 0.13\text{-}0.16$. The same settings also top the rankings for the ResNet18 variants but cannot reach the equivalent ResNet50 levels. Solely its standard version can best the ResNet50's standard version. The standard ResNet18 performs 6.3% better than its ResNet50 equivalent, showing that more layers do not necessarily mean better accuracy [124, 125]. The shallow network cannot compete in non-private performance, even when adding BN and pneumonia pre-training. However, it delivers the same or better privacy guarantees than the safest ResNet, down to a 95%-CI of $\varepsilon_{AUC} = 0.09\text{-}0.12$. In steep contrast to the ResNets, pneumonia pre-training for the shallow network slightly improved performance by 0.4%, while it slightly reduced the empirical privacy budget for all models. A negative impact regarding accuracy and privacy was observed in all models when including dropout. According to these results, the ResNet50 pre-trained on ImageNet constitutes the best performing model for the non-private baseline, while the standard shallow network represents the most secure model.

5.2.2. Evaluating DP-SGD Models

Table 5.2.: Overview of experimental results for DP-SGD models. Models are matched by their test accuracy in % and proneness to MIAs. The proneness is measured by the empirical privacy budget regarding the AUC (ε_{AUC}) and is given as a 95% CI over 100 attacks. The privacy budget for the evaluation is set to $\varepsilon = 1$. If training did not result in a feasible model, the accuracy is given as NA and no attacks are performed.

		$\varepsilon = 1$	
Model	Variant	Acc.	ε_{AUC}
ResNet18	Standard	NA	—
	Dropout	NA	—
	ImageNet	NA	—
	Pneumonia	71.2%	0.14-0.18
	tanh	70.7%	0.14-0.17
	tanh-Dropout	72.0%	0.15-0.18
	tanh-Pneumonia	**74.1%**	0.14-0.17
ResNet50	tanh	71.7%	0.16-0.19
	tanh-Pneumonia	73.5%	**0.10-0.12**
Shallow network	Standard	NA	—
	BN	72.1%	0.14-0.17
	BN-Dropout	69.7%	0.13-0.16
	BN-Pneumonia	72.2%	0.10-0.13
	tanh	NA	—
	tanh-BN	70.9%	0.13-0.17
	tanh-BN-Dropout	69.8%	0.13-0.16
	tanh-BN-Pneumonia	72.4%	0.12-0.15

In the following, the proposed DP-SGD models are compared, now referring to Table 5.2. In this first of two private settings, the highest number of model variants is evaluated, which

is owed to the introduction of the tanh activation experiment. As suggested in Chapter 4 Section 4.9, the general evaluation of private models is done at $\varepsilon = 1$.

ResNet18. Looking at the first three ResNet18 variants using ReLU activation (standard, dropout, and ImageNet), no feasible model is achieved. The only ReLU model to overcome this hurdle is pre-trained on the pneumonia task with 71.2% and ε_{AUC} = 0.14-0.18, indicating its superiority over the ImageNet pre-training in DP-SGD. Checking the other ResNet18 variants shows advantages from the tanh vs. the ReLU activation function since all tested models using tanh result in working models. The respective standard tanh version shows 70.7% accuracy and thus ranks slightly behind the ReLU pneumonia model in performance while delivering comparable privacy. When including dropout combined with tanh activation, the model sees an increase in performance to 72%, outperforming the ReLU pneumonia model by 0.8%. At the same time, dropout keeps almost the same proneness to attacks at 0.15-0.18. The estimated privacy budget also stays at 0.15-0.17 when adding pneumonia pre-training to the standard tanh ResNet18. Performance-wise, on the other hand, the tanh-pneumonia variant finds the best result of 74.1%.

ResNet50. Again, only the relevant ResNet50 results are shown since they generally mimic the findings from the ResNet18 versions due to their architectural similarities. Therefore, the ResNet50 with tanh activation and the matching tanh-Pneumonia model are added to the comparison. The tanh-Pneumonia variation is the most successful in performance and privacy, reaching 73.5% and 0.10-0.12, respectively. The standard tanh model, on the other hand, achieves 71.7% and an ε_{AUC} of 0.16-0.19

Shallow network. The most crucial improvement for the shallow network is the addition of BN layers. Without these, neither the ReLU based nor the tanh based standard version can conclude in a viable model. When included, the ReLU shallow network delivers 72.1% and the tanh version 70.9%. The difference in their privacy results, however, is only marginal. Adding dropout, the ReLU and tanh versions result in almost identical models with 69.7% vs. 69.8% in accuracy and 0.13-0.16 vs. 0.13-0.17 in ε_{AUC}. Accuracy differences between the two models with pneumonia pre-training stay low with 72.2% for ReLU and 72.4% for tanh. The privacy guarantees see a change from 0.10-0.13 to 0.12-0.15. The best performing model with 72.4% is the tanh-BN-Pneumonia variant, while the BN-Pneumonia candidate using ReLU holds the best privacy.

Summary. This was the perfect setting for the shallow network to shine since smaller models are meant to perform better in DP-SGD [124, 125]. However, the ResNets reign superior in both evaluated metrics. The gap in accuracy between shallow network and ResNets got minimal compared to the non-private setting, but the shallow network lost its privacy advantage. The overall trend concerning the tanh vs. ReLU activation function in DP-SGD shows a positive influence. There is only one case in the shallow network where the tanh function resulted in a worse model than using ReLU. The ResNet50 achieves 1% higher performance than the ResNet18 in the standard tanh version, but the best performer overall is the ResNet18 tanh-Pneumonia with 74.1%. It thereby outperforms the respective ResNet50 variant by 0.6%, showing that its smaller size is no hindrance in this settings [124, 125]. The ResNet50 version, on the other hand, is the most private model coming in at ε_{AUC} = 0.10-0.12. Pneumonia pre-training gives irrefutable improvements in all tested cases and provides the

best models on both metrics. The ImageNet lags behind and, in contrast to pneumonia pre-training, could not achieve a feasible ReLU model. The inclusion of dropout leaves a mixed impression in DP-SGD. When used in the shallow network, it worsens the accuracy while improving the same metric in the ResNet18. Regarding privacy, dropout did not significantly influence the models.

5.2.3. Evaluating PATE Models

Table 5.3.: Overview of experimental results for PATE models. The third column gives the average test accuracy of PATE teacher models, which can be seen as the realistic upper bound for the student. The privacy budget, in that case, is $\varepsilon = \infty$ since the values are taken before the noisy aggregation. The last two columns then compare test accuracy in % and proneness to MIAs regarding the final student models. The proneness is measured by the empirical privacy budget regarding the AUC (ε_{AUC}) and is given as a 95% CI over 100 attacks. Here, the privacy budget for the experiments is set to $\varepsilon = 10$. If training did not result in a feasible model, the accuracy is given as NA, and no attacks are performed.

			$\varepsilon = 10$	
Model	Variant	Teacher acc.	Student acc.	ε_{AUC}
PATE-25 ResNet18	Standard	76.0%	70.0%	0.14-0.18
	ImageNet	46.1%	40.0%	0.10-0.14
	Pneumonia	76.3%	72.0%	0.12-0.16
PATE-25 ResNet50	Standard	78.1%	**75.0%**	0.09-0.12
	Pneumonia	**79.9%**	74.9%	**0.07-0.10**
PATE-25 Shallow network	Standard	54.0%	40.0%	0.17-0.20
PATE-50 ResNet18	Standard	58.4%	61.0%	0.15-0.18
	ImageNet	45.6%	40.0%	0.12-0.15
	Pneumonia	58.0%	57.3%	0.16-0.19

Finally, the results for PATE, the second PPML method, are compared to find the best settings for applying the algorithm on the task. Table 5.3 presents two differences to the previous Table 5.2 for DP-SGD. First, a column for teacher accuracy is added, which lists the average accuracy of the trained teacher ensemble before using them for the noisy aggregation of the student data. The average teacher accuracy is a non-private result and can be seen as the achievable upper bound for training the private student without noise. The second change is that the evaluation takes place on $\varepsilon = 10$. The small dataset results in tiny training partitions for each teacher, limiting their maximum feasible number. These restrictions entail that, at $\varepsilon = 1$, the accuracy of the noisy aggregated labels was under 52% for all teacher setups. The aggregated labels' accuracy determines how many samples in the student dataset are labeled correctly. The found 52% almost translate to randomly flipping the labels, hindering successful student training. This poor result is due to the high noise scale needed to satisfy $\varepsilon = 1$ using PATE. All students trained in this setting could not learn a working classification system based on such inputs. A higher budget is needed to still compare and evaluate the PATE models. Hence, this part uses $\varepsilon = 10$ to at least get an idea of the algorithm's capabilities.

Lucas Lange
3737104

Teacher number. It was impossible to train 50 teachers using a ResNet50, surpassing the computational resources. As a result, the PATE-25 and PATE-50 methods are only matched using the ResNet18. Still, comparing the results shows that the bigger data splits available to each teacher when using 25 teachers outperform the 50 teacher settings. Looking at the results, the PATE-25 standard and pneumonia models achieve about 18% better teacher performance than the respective PATE-50 approaches (76% vs. 58%). The advantage of PATE-25 carries over to the student models delivering 70% and 72%, while the PATE-50 versions only reach 61% and 57.3%. At the same time, PATE-25 keeps lower privacy budgets in both cases. When comparing student accuracy, it is important to consider that the use of $\varepsilon = 10$ significantly reduces the noise impact down to the limit of additional 33 noisy votes in the noisy aggregation. Thus, the low amount of 25 teacher votes available in the PATE-25 settings is more relevant than at $\varepsilon = 1$, where PATE-50 could gain an advantage against the 278 noisy votes.

Teacher and student model. Looking back at the non-private results in Section 5.2.1, the best performing model was found to be the ResNet50 pre-trained on ImageNet data, while ResNet18 on ImageNet came in on the second spot. The shallow network was not able to reach the same accuracy levels. These findings were anticipated to hold in PATE because the algorithm supports non-private training strategies. However, the ImageNet variants fall short in the PATE setting, as can be concluded from the teacher accuracy in Table 5.3, where the noiseless average accuracies over all teachers in the 25 and 50 teacher settings are compared. As mentioned in the introducing paragraph to this section, the average teacher accuracy can be seen as the realistic upper bound achievable for the student. There seems to be a stern difference in which model can cope with the little data available to each teacher model. Even though PATE internally uses non-private training, the ImageNet pre-training has disastrous consequences on the average teacher accuracy. In PATE-25, the standard ResNet18 version achieves 76%, with ImageNet implying a reduction of 30% to a result of 46.1%. Taking the same setup in the PATE-50 case sees a loss of 12.8% from 58.4% to 45.6%. The results of ImageNet pre-training indicate a strong negative impact on performance, probably due to the small data fraction available to each teacher. On the other hand, the pneumonia pre-training adds performance and privacy benefits in PATE-25. For the PATE-25 ResNet18 and ResNet50 this means respective accuracy boosts of 0.3% and 1.8%. In contrast, PATE-50 on the ResNet18 loses 0.4% due to pneumonia pre-training. Still, two observations carry over from the non-private models in Section 5.2.1, which are the worse performance of the shallow network that only reaches 54% teacher accuracy and the ResNet50 beating the ResNet18. The ResNet50 achieves 78.1% and 79.9% teacher accuracy in PATE-25, while the ResNet18 rests at 76% and 76.3%.

The ImageNet and shallow network models fail to reach more than 40% accuracy when looking at private student performance. The ResNet18 in the PATE-50 setting falls short of its PATE-25 version, as described in the paragraph concerning the teacher numbers. When comparing the ResNet student models in PATE-25, the ResNet50 outperforms the ResNet18 in both metrics. The ResNet50 versions deliver better privacy guarantees with a difference of 0.05-0.06 compared to the ResNet18 candidates, while the difference in performance is at 3-5%. The PATE-25 ResNet18 and ResNet50 see a similar reduction of 0.02 in ε_{AUC}

when utilizing pneumonia pre-training. The ResNet18 model also sees an increase to 72% accuracy coming from 70% in the standard variant. In contrast to the PATE-25 findings, the PATE-50 student on the ResNet18 loses 3.7% and increases ε_{AUC} by 0.01 due to pneumonia pre-training. The two best student models are delivered by PATE-25 using the ResNet50 in standard and with pneumonia pre-training. Both achieved an almost similar student accuracy of 75.0% and 74.9%, respectively. However, the pneumonia version lowered the ε_{AUC} by 0.02 giving 0.07-0.10 and surpassed the standard model by 1.8% in teacher accuracy with 79.9%. The pneumonia version thus is the most private PATE model. Seeing the privacy advantage and the minimal accuracy deviation, the PATE-25 pneumonia variant of the ResNet50 is the best overall PATE candidate.

On a final note, the results generally show only a small gap between the average teacher accuracy and the student accuracy. The selected best model of PATE-25 with ResNet50 on pneumonia pre-training loses around 5% from its teacher accuracy when training the student. The PATE-50 with ResNet18 on pneumonia pre-training reduces the gap to just 0.7%. When looking at the row for the PATE-50 standard, one finds an anomaly probably stemming from the randomness induced by the noise. The student outperformed the average teacher by roughly 2.5% in this scenario. The impact on the aggregated labels must have flipped them in such a fashion that the student could learn a better classification.

5.3. Best Results on the COVID-19 Database

Table 5.4.: Final results for the best models for COVID-19 detection regarding performance and privacy. Each model is evaluated across the different privacy budgets of $\varepsilon = \infty$, $\varepsilon = 10$, $\varepsilon = 1$, and $\varepsilon = 0.1$. PATE models give the average teacher accuracy as a non-private result and use the PATE algorithm to train private student models. All other models are trained with DP-SGD to achieve their privacy guarantees and otherwise use their standard non-private results. The two columns for each privacy budget compare each model's test accuracy in % and their proneness to MIAs. The proneness is measured by the empirical privacy budget regarding the AUC (ε_{AUC}) and is given as a 95% CI over 100 attacks. If training did not result in a feasible model, the accuracy is given as NA, and no attacks are performed.

		$\varepsilon = \infty$		$\varepsilon = 10$		$\varepsilon = 1$		$\varepsilon = 0.1$	
Model	Variant	Acc.	ε_{AUC}	Acc.	ε_{AUC}	Acc.	ε_{AUC}	Acc.	ε_{AUC}
ResNet18	tanh-Pneumonia	77.5%	0.11-0.14	69.3%	0.19-0.23	**74.1%**	0.14-0.17	69.6%	**0.13-0.16**
ResNet50	ImageNet	**97.6%**	0.13-0.16	NA	—	NA	—	NA	—
	tanh-Pneumonia	77.9%	0.14-0.18	71.6%	0.16-0.19	73.5%	**0.10-0.12**	**72.0%**	0.17-0.20
Shallow network	Standard	79.6%	**0.09-0.12**	NA	—	NA	—	NA	—
PATE-25 ResNet50	Pneumonia	79.9%	—	**74.9%**	**0.07-0.10**	40.7%	0.14-0.17	40.0%	0.17-0.20

The final comparison on the COVID-19 task takes the best models from Section 5.2 regarding performance and privacy for each approach (non-private, DP-SGD, PATE). The best models for the non-private setting are the ResNet50 on Imagenet and the standard shallow network. For DP-SGD, the tanh-Pneumonia variants of ResNet18 and ResNet50 are selected, and for PATE, the PATE-25 with ResNet50 on pneumonia pre-training. In addition to their already seen results, all models are now evaluated on the privacy levels of $\varepsilon = \infty$, 10, 1, and 0.1. The goal of $\varepsilon = 1$ rests as the primary focus, but multiple privacy budgets might reveal

overall trends. The chosen values present each candidate from the non-private and different private levels. The non-private result for PATE is taken from the average teacher accuracy, while the private results state the resulting student performance. Comparably, the other non-private results are obtained from non-private training, while the respective private models use DP-SGD to reach their guarantees. The relevant results are accumulated in Table 5.4.

The evaluation starts by going over the results regarding each privacy budget before taking in the whole picture. In the non-private setting of $\varepsilon = \infty$, the ResNet50 on ImageNet pre-training keeps the best performance with 97.6% accuracy. The teachers of the PATE-25 model deliver the second-best accuracy of 79.9%, which creates a gap of 17.7%. The shallow network stays close behind at 79.6%. The models on tanh-Pneumonia come in last with 79.6% and 77.5%. The shallow network and tanh are experiments focused on improving the model in DP-SGD, which explains their weak non-private performance. Still, the shallow network delivers the best in class privacy at ε_{AUC} = 0.09-0.12. For its utility gains, the ResNet50 ImageNet model increases the budget to 0.12-0.16, which should be a fair trade-off.

Stepping into the private domain beginning at $\varepsilon = 10$, the included models trained using the ReLU activation function start to fail in learning a functional classification. The PATE algorithm shows its efficiency by keeping the highest accuracy of 74.9%. The two ResNets on their tanh-Pneumonia versions see bigger performance hits and only achieve 69.3% and 71.6%. The PATE model also states the best estimated privacy of 0.07-0.10, while the other models only achieve 0.19-0.23 and 0.16-0.19.

At $\varepsilon = 1$, the tanh-Pneumonia models show their advantage. The ResNets trade blows, with the ResNet18 tanh-Pneumonia being the best performing by reaching 74.1% and an ε_{AUC} of 0.14-0.17. The corresponding ResNet50 achieves only 73.5% but exhibits the best privacy at 0.10-0.12. PATE reaches the impractical point for the task resting at a mere 40.7% due to the limited data and high noise scale.

A privacy level not included until now is $\varepsilon = 0.1$, which offers even tighter privacy bounds. At this stage, PATE rests unfeasible at 40%, while both tanh-Pneumonia models change their skill set. The ResNet18 variant becomes the most secure at 0.13-0.16 vs. 0.17-0.20 for the ResNet50. The ResNet50, on the other hand, gives the best accuracy of 72% vs. ResNet18's 69.6%.

In general, the ResNets on tanh-Pneumonia rule the lower two privacy levels, while PATE shows promising results in the $\varepsilon = 10$ case. The fact that the ResNets' results for tanh-Pneumonia at $\varepsilon = 10$ are lower than on the more private levels is an anomaly. It could reflect the randomness introduced by adding noise in the training process. A private model that stays over or at 72% accuracy was found for all three privacy guarantees. For the goal of $\varepsilon = 1$, the ResNet18 on the tanh-Pneumonia setup performed best with 74.1%. The analog ResNet50 variant is slightly more private at 0.10-0.12, but with 0.14-17, the ResNet18 candidate conforms to the theoretical limit of 1. Comparing the best performing private results to their respective model's initial non-private performance, the gap seems manageable at 5% in $\varepsilon = 10$, just 3.4% in $\varepsilon = 1$, and still only 5.9% in $\varepsilon = 0.1$. However, when comparing to the ResNet50 on ImageNet, the non-private baseline, a vast accuracy gap of 23.5% is opening when looking at $\varepsilon = 1$. Another finding worsens the resulting big utility-privacy trade-off. When looking at the empirical privacy budgets going from $\varepsilon = \infty$ to $\varepsilon = 0.1$, the models'

proneness to MIAs stays on roughly the same average level. The average estimated upper bounds on each level are 0.15 for $\varepsilon = \infty$, 0.17 for $\varepsilon = 10$, 0.15 for $\varepsilon = 1$, and 0.19 for $\varepsilon = 0.1$. Models seem to randomly show higher and lower empirical bounds, unrelated to the chosen theoretical privacy guarantee. Although a theoretical budget of $\varepsilon = 0.1$ is achieved in training, all models in the $\varepsilon = 0.1$ setting surpass this threshold in the empirical study using MIAs. The only entry where $\varepsilon_{AUC} = 0.1$ is achieved is by the PATE-25 model at the theoretical level of $\varepsilon = 10$ with an interval of 0.07-0.10. It is important to note that the values in non-private training already stay at a maximum of 0.18 and thus empirically satisfy the wanted guarantee of $\varepsilon = 1$. Over all levels, the maximum reaches 0.23. In conclusion, the experiments on the COVID-19 database show no consistent correlation between the theoretical DP guarantee and the defense against MIAs.

5.4. Results on the MNIST Database

Table 5.5.: Summary of the results on the MNIST database. Each model is evaluated in non-private and private, given as $\varepsilon = \infty$ and $\varepsilon = 1$, respectively. The private variants are trained with DP-SGD to achieve their privacy guarantees. For both settings, each model is compared in test accuracy in % and their proneness to MIAs. The proneness is measured by the empirical privacy budget regarding the AUC (ε_{AUC}) and is given as a 95% CI over 100 attacks.

		$\varepsilon = \infty$		$\varepsilon = 1$	
Model	Variant	Acc.	ε_{AUC}	Acc.	ε_{AUC}
ResNet18	Standard	99.3%	0.72-0.90	92.1%	**0.18-0.21**
	ImageNet	**99.6%**	0.85-1.27	86.7%	0.19-0.21
	tanh	98.8%	**0.37-0.47**	92.2%	0.19-0.22
Shallow network	Standard	98.5%	0.52-0.61	90.7%	0.22-0.25
	BN	99.1%	1.05-1.45	**95.4%**	0.19-0.22
	tanh-BN	98.8%	0.49-0.71	93.6%	0.19-0.21

For better comparison, models are selected to run experiments on the MNIST database [17]. There are several advantages to additionally evaluating models on this task. First, most related work in PPML is based on research focusing on or including this dataset, e.g. [41, 52, 53, 112, 85, 113, 114]. Further, running the models on a different task can help verify the experiments' findings. The selected architectures include the ResNet18 as the ResNet representative because it is less resource-hungry but almost as performant as a ResNet50. The second contender is the shallow network since it was originally designed for the MNIST database and should reach higher performance on the task [112]. Other than the standard versions, the experiments test variants with ImageNet pre-training, tanh activation, and BN. Pneumonia pre-training is not used on MNIST because it is unrelated to the task. Further, PATE is excluded from this detour because of its more complex implementation by hand, especially regarding the unlabeled student set. Since the goal is to gain general insights on the relation between DP and MIA, there should be no drawbacks from not including a specific method. Instead, private versions are achieved using DP-SGD. For this evaluation the candidates are only matched on $\varepsilon = \infty$ vs. $\varepsilon = 1$. The results are shown in Table 5.5, where

models are compared by performance and estimated privacy when trained on the MNIST database.

From the general non-private performance trends, it can be determined that MNIST is an easier task than COVID-19 detection since a small-sized model like the shallow network delivers almost the same performance as the ResNet18 with only a difference of 0.8%. Further, all variants deliver very high non-private accuracy, with the worst still achieving 98.5%—given by the standard shallow network. Familiarly to the COVID-19 task, ImageNet on the ResNet18 performs best with 99.6%, outperforming the standard version by 0.3%. The shallow network reaches its highest of 99.1% when including BN. However, the ImageNet in ResNet18 and the BN in the shallow network inflate the estimated privacy budget to an upper bound higher than 1. BN more than doubles the shallow network's estimated budget compared to the standard model. The experiments of tanh for the ResNet18 and tanh-BN for the shallow network achieve the same accuracy of 98.8%. In the ResNet18, tanh can roughly halve the estimated privacy budget of the standard version from 0.72-0.90 to 0.37-0.47. However, the non-private models generally suffer from considerably higher empirical privacy budgets than on the X-ray task. While $\varepsilon_{AUC} \leq 0.26$ applied to all non-private COVID-19 models in Table 5.1, the non-private MNIST models see a maximum of 1.45. The average upper bound for MNIST ε_{AUC} is found to be 0.90, where in contrast, the average for COVID-19 was 0.20. Budgets are also subject to bigger fluctuations from model to model, with the best result being $\varepsilon_{AUC} = 0.37$-0.47 and the worst coming in at 1.05-1.45.

Shifting the view to the private results at $\varepsilon = 1$, the shallow network with BN outperforms the other models and delivers 95.4%—4.7% better than the standard result. The privacy inflation seen from the non-private BN experiment can not be found on the private model lowering the standard's estimated guarantee of 0.22-0.25 to 0.19-0.22. The tanh-BN variant gives the same privacy improvement but only reaches 93.6%. However, this result is still better than the best ResNet18, which is the tanh model at 92.2%. It only slightly topped the standard version by 0.1%. The ImageNet variant proves its negative performance impact in DP-SGD from the X-ray findings by only delivering 86.7%. The shallow network outperforming the ResNet18 shows that much smaller-sized networks can be better than deep networks in DP-SGD. The MNIST candidates, in general, show a smaller gap between non-private and private performance than in the COVID-19 task, with the difference between the best two models being only 4.5%, whereas 23.5% was found before. When looking at the accomplished estimated privacy budgets at $\varepsilon = 1$, DP has substantial benefits for MIA defense in all MNIST models. In the private setting, all models give roughly the same numbers, with the maximum ε_{AUC} reduced to 0.25 from 1.45 in non-private. All models could at least halve their own non-private result and still roughly halve the best non-private result of 0.37-0.47. These improvements are reflected in the average estimated upper bound of 0.22 at $\varepsilon = 1$. Compared to the non-private average of 0.90, the average was reduced by 76%. The initially expected correlation between DP and MIA defense for the COVID-19 task could not be found in Section 5.3 [48, 75, 76]. Running the same experiments on the MNIST task, the results show an apparent reduction in MIA success related to the increased DP level.

5.5. Discussion

The last sections went over numerous experiments and introduced many possible points of discussion. This section revisits the findings of the evaluation to then deduct the key takeaways.

The achieved non-private accuracy on COVID-19 vs. Normal detection of 97.6% by the ResNet50 with ImageNet pre-training is close to the goal of 98% set from the literature reviewed in Chapter 3 Section 3.1 [92, 93, 94, 95, 96]. The achieved non-private baseline thus represents a feasible model in the context of related works. The best private model lags behind accuracy-wise with a result of 74.1% when using the ResNet18 tanh-Pneumonia at $\varepsilon = 1$. This results in an accuracy-privacy trade-off of 23.5%, which is within the limits of 20.0-23.7% expected from other works in Chapter 3 Section 3.2 [71, 103]. Even though the private approaches could not hold the non-private level of performance, the experiments showed beneficial outcomes and helped improve the accuracy-privacy trade-off. Regarding the expected relations between model size and performance by [124] and [125], the tested architectures roughly correspond to the two hypotheses explained in Chapter 4 Section 4.7. The first hypothesis states that in non-private training, the deeper model performs better, which is shown in Section 5.2.1 by the ResNet50 being best and the ResNet18, in turn, outperforming the shallow network. The best shallow network performs 11.2% worse than the best ResNet50. The same is found in the non-private training featured in PATE, see Section 5.2.3. The second hypothesis counter-intuitively states that bigger models do not have the same benefits in DP-SGD. Section 5.2.2 mainly shows the ResNet18 outperforming the ResNet50 in DP-SGD. However, the shallow network only stays behind 1.7% in this setting, conforming to the hypothesis. The same results are found on the MNIST database, where the deeper ResNets win in the non-private settings. However, as assumed from [112], the shallow network then manages to achieve the best performance in DP-SGD and outperforms the ResNet18.

Pre-training, in general, was the most successful experiment since it proved advantageous in all models. However, each approach reacted differently to the two forms of pre-training. Non-private models favoured the general ImageNet pre-training from [111], while DP-SGD benefited more from the task-specific method (pneumonia) seen in [41]. The models for non-private classification largely benefited from the ImageNet pre-training, resulting in the highest accuracy of all approaches with 97.6%. Pneumonia pre-training instead reduced non-private accuracy by 5%. However, pneumonia started to outperform ImageNet pre-training in the DP-SGD setting and helped put out the best performing model at 74.1% for $\varepsilon = 1$. In a similar notion, the ImageNet turned out to be catastrophic in PATE by losing up to 30% accuracy. Pneumonia pre-training, on the other hand, improved the PATE-25 models' performance and privacy by up to 2% and 0.02 in ε_{AUC}. The findings for the ImageNet held on the MNIST database, where it gave the best non-private accuracy but fell short in DP-SGD.

As an experiment meant to improve DP-SGD performance, tanh results should also only be considered in this setting [112]. The change from ReLU activation function to tanh in combination with pneumonia pre-training generally gained the DP-SGD models between 0.2-3.1% in accuracy and is used in the best performing DP-SGD model. The positive impact of

tanh in DP-SGD is also visible on the MNIST task, where it improved models by 0.1% and 2.9%.

The influence of dropout on the success of MIAs is left unclear by related work. While [135] states an increased proneness to MIA from dropout, [83] and [136] describe dropout to be an effective measure against them. Clearer is dropout's positive effect on accuracy when used to combat overfitting [19, 134]. In this work, however, dropout only once increased accuracy by 1.3%, while on the other hand, reducing the same metric by up to 3.6% in four other models. Concerning MIAs, the inclusion of dropout resulted in a higher ε_{AUC} in three models and a lower ε_{AUC} in two models. Further, the biggest change in the estimated budget was only by 0.03-0.04. Based on these preliminary results, it is not possible to give a clear answer to dropout's influence.

When adding BN to the shallow network, expectations regarding [133] were to mainly improve DP-SGD performance. Instead, it proved worthwhile in non-private and DP-SGD training. BN, in all cases, improved the performance regarding the standard version. These findings apply to the COVID-19 and the MNIST database.

Section 5.2.3 showed that the small COVID-19 dataset supports PATE using 25 teachers better than using 50 teachers. On the other hand, it is difficult to make a concise decision regarding PATE vs. DP-SGD since only $\varepsilon = 10$ could be evaluated for PATE. The showed privacy-accuracy trade-off of 5% at that level is promising, but ultimately PATE could not be successfully applied on the small dataset. Thus, DP-SGD is the best for this COVID-19 detection task—out of the two methods.

When comparing the achieved 74.1% at $\varepsilon = 1$ to the best non-private model, an accuracy-privacy trade-off of 23.5% seems large, but the results gain more perspective when looking at related works on COVID-19 detection. [103] used PATE at an ε close to 6, only reaching an accuracy of 71%, while also instilling a trade-off of 23.7% to their respective non-private model. The combination of DP-SGD and FL by [71] achieves 73.8% accuracy and reduces the trade-off to 20%, but only delivers a weak guarantee of $\varepsilon = 39.4$. The only model outperforming the presented 74.1% is the FedDPGAN from [106] which reaches a private model with 94.5% and even beats its respective baseline model by 1.5%. However, the evaluation does not state the used privacy budget, leaving the comparability of the FedDPGAN results unclear. In [61] the authors declare a lack of methods to track the theoretical privacy guarantees when using DP-GANs, further hurting the reliability of the FedDPGAN. Thus, when only comparing with the privacy accounted approaches, the presented model at 74.1% outperforms related works by 0.3% while even guaranteeing the tighter privacy bound of $\varepsilon = 1$. The result is based on the ResNet18 tanh-Pneumonia model and privately trained with DP-SGD. The literature review found no pure DP-SGD result on COVID-19 X-ray detection, making this work the first to test this approach.

Figure 5.1 shows the evolution of accuracy across the evaluated privacy budgets on the COVID-19 task (left) and the MNIST task (right). The COVID-19 results already show a discrepancy at the non-private level. The best model at 97.5% is an outlier compared to the other models that flock together at close to 80%. While the tanh-Pneumonia models keep their performance relatively stable going into the lower privacy budgets, the ReLU models and the PATE model see a substantial reduction in accuracy. The private performance seems

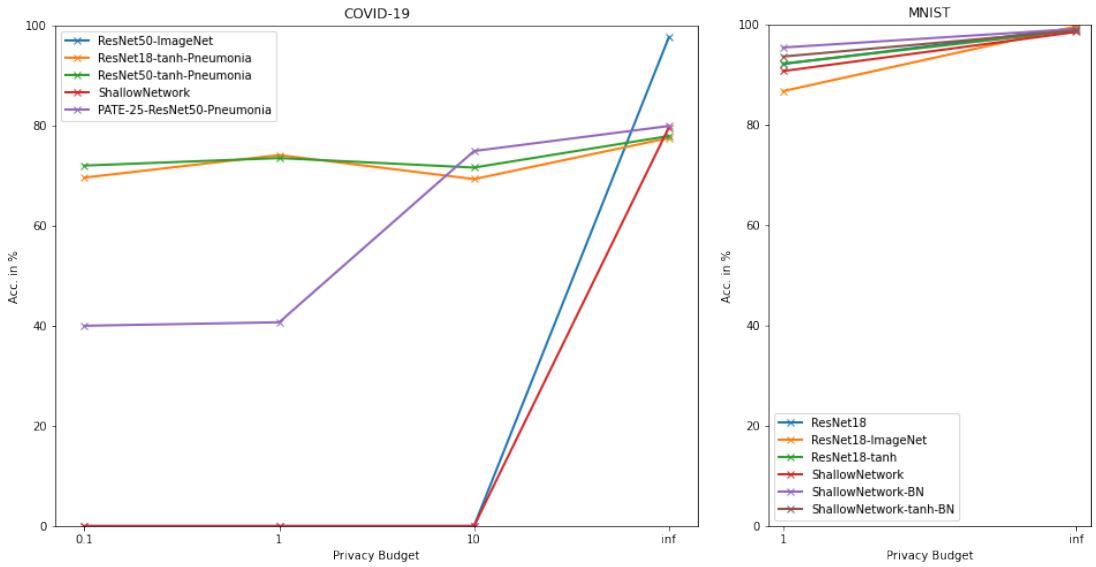

Figure 5.1.: Graphical representations of the achieved classification accuracy on the COVID-19 database (left) and the MNIST database (right) across the different privacy levels. For models on the COVID-19 set privacy budgets of $\varepsilon = \infty$, 10, 1, and 0.1 are shown, while the MNIST graph only features $\varepsilon = \infty$ and 1. Each evaluated model is represented by a coloured line and with crosses marking the obtained results. Colour coding for COVID-19 differs from the one used for MNIST.

to be related to the architectural experiments. Although the MNIST models' non-private performances concentrate in one spot and the accuracy-privacy trade-off is better than on the COVID-19 set, the private results still present deviations in their accuracy. Again, there are architectural experiments that outperform others; however, the MNIST task presents winners different from the COVID-19 results. Not only was the shallow network able to outperform the ResNet18, but the accuracy-privacy trade-off for MNIST is sitting at a mere 4.2%. This difference hints at the PPML or DP-SGD results being dependent on the given task. The best solution on one task does not necessarily transfer to another task.

Up to this point, this section evaluated the accuracy-privacy trade-off on the relation between classification accuracy and theoretical privacy budget ε. Regarding privacy metrics, Section 5.1 introduced the empirical privacy budget ε_{AUC}, which is calculated from the results of MIAs. Utilizing this metric showed that even the non-private models resulted in strong privacy guarantees of $\varepsilon_{AUC} \leq 0.18$, which is well under the wanted privacy budget of $\varepsilon = 1$. This phenomenon was also present in [61], where the authors found their models without any protection to already restrict a attacks to nearly random guessing membership inference. In addition to the low initial values, private training at different privacy levels did not influence the defense against MIAs. This fact is represented in Figure 5.2 (left), which shows the development of the empirical privacy budget across the different theoretical budgets on the COVID-19 task. The presented values for ε_{AUC} are the upper bound of the 95% CI after 100 attacks. The values fluctuate at each level but do not generally see a reduction with stronger privacy guarantees. Instead, the defense against MIAs keeps an almost steady course. At $\varepsilon = 0.1$ no model is able to hold the promised guarantee with even the most secure model

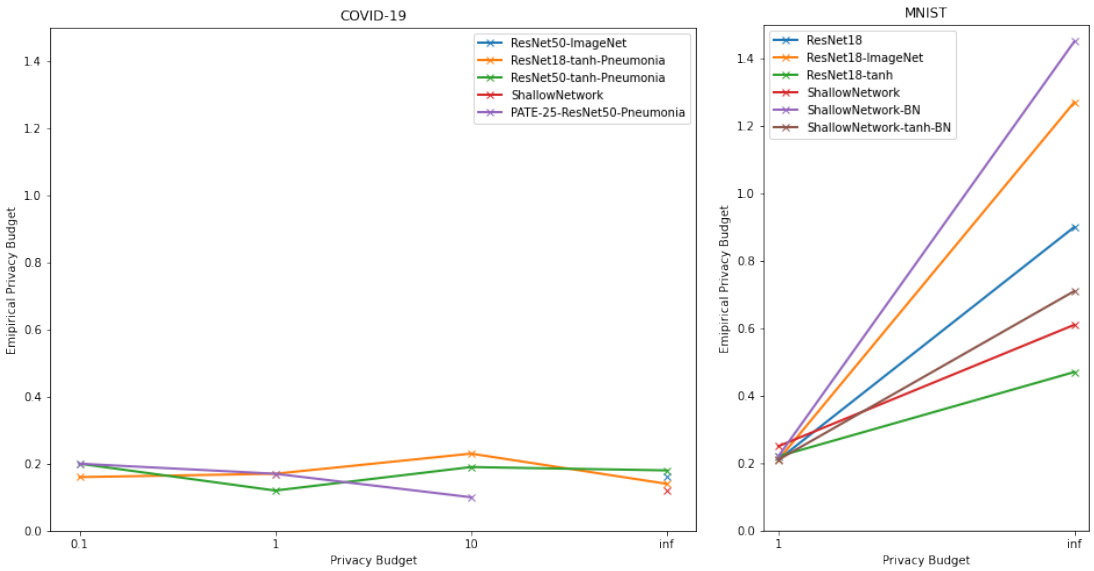

Figure 5.2.: Graphical representations of the estimated privacy budget ε_{AUC} on the COVID-19 database (left) and the MNIST database (right) across the different privacy levels. The presented values for ε_{AUC} are the upper bound of the 95% CI after 100 attacks. For models on the COVID-19 set privacy budgets of $\varepsilon = \infty$, 10, 1, and 0.1 are shown, while the MNIST graph only features $\varepsilon = \infty$ and 1. Each evaluated model is represented by a coloured line and with crosses marking the obtained results. Colour coding for COVID-19 differs from the one used for MNIST.

giving an upper bound of $\varepsilon_{AUC} = 0.13$. The results from the COVID-19 data indicate no consistent correlation between the theoretical DP guarantee and the defense against MIAs.

In contrast to this finding, according to [75] and [76] DP limits the success of MIAs by definition and [48] found this to be true on the MNIST dataset. Figure 5.2 (right) illustrates the results from the MNIST experiments in this work and shows how ε_{AUC} is affected by DP on the MNIST dataset. The tested non-private models on MNIST suffer from significantly higher estimated budgets than on the COVID-19 set. They also exhibit a wide fluctuation from an upper ε_{AUC} bound of 0.47 up to 1.45. When training for DP at $\varepsilon = 1$ on the other hand, the estimated budgets of all models are projected to roughly the same point with an average upper bound of 0.22. Compared to the non-private average of 0.90, this means the average was reduced by 76% due to DP. In conclusion, the same experiments on the COVID-19 and MNIST tasks delivered opposite results. On the COVID-19 task, there was no indication of a reduction in MIA success related to an increased privacy level from DP, while the same assumption was evident on the MNIST task.

There are two takeaways from the opposing findings regarding the experiments on the COVID-19 and MNIST sets. First, DP seems unnecessary when defending against MIAs in COVID-19 detection. Using non-private models would give the same estimated proneness while eliminating the accuracy-privacy trade-off from PPML. Additionally, better DP guarantees did not improve the COVID-19 models' defenses further, as the trained $\varepsilon = 0.1$ was not satisfied by the models' ε_{AUC} results. Second, there is now more evidence of PPML being heavily task-dependent. Not only do the best performing models differ from task to task, but also the effectiveness of defense mechanisms like DP. DP did not correlate to MIA defense on

the COVID task but drastically improved the defense on the MNIST task. At $\varepsilon = 1$, models on the MNIST dataset presented the same defense against MIAs that could already be found in the non-private COVID-19 models. It is untested if lesser privacy guarantees might already achieve the same outcome for MNIST.

Differentially-private models at $\varepsilon = 1$ on both datasets delivered significantly lower estimated privacy budgets than their theoretical analysis. Thus, DP provided better than expected defense at these levels but, in return, overbilled the model by unnecessarily increasing the privacy-accuracy trade-off. The theoretical privacy guarantee given by ε is an upper bound that can overestimate the realistic threat level [113]. In addition, it cannot convey the difference in non-private MIA defense regarding the two datasets. The findings were instead enabled by including the estimated privacy budgets based on the MIA results. The ε_{AUC} is a more task-specific metric because it directly depends on the target model, while ε solely depends on the settings of the training procedure. Therefore, the metric was essential for assessing model privacy in the evaluation. However, there is no clear explanation for the differing results on the COVID-19 and MNIST tasks. In [75], Shokri et al. suggest that classification tasks with more classes allow more successful MIAs. This difference could be a factor with only two classes in the COVID-19 detection and ten classes in the MNIST task.

In conclusion, the evaluation showed no improvement of MIA defense from DP on the COVID-19 task. The non-private models already exhibited the same repelling properties as the private models, and at $\varepsilon = 0.1$ no model was able to hold the promised guarantee with even the most secure model giving an upper bound of $\varepsilon_{AUC} = 0.13$. Thus, when solely looking for MIA defense, there is no drawback in taking the best non-private model in the form of the ResNet50 with ImageNet pre-training, which achieved 97.6% accuracy. When a theoretical privacy guarantee is needed, private DP-SGD training for $\varepsilon = 1$ delivers the best private model at 74.1% accuracy. It uses a ResNet18 pre-trained on pneumonia X-rays while also changing the ReLU activation function to the tanh function. Even though the accuracy-privacy trade-off of 23.5% is significant, the model performs 0.3% better than related work while keeping much tighter privacy guarantees [71, 103]. Ultimately, the conflicting findings from the MNIST database indicate that PPML (or DP-SGD) is heavily task-dependent, and thus research on one dataset might not carry over to others.

6. Conclusion & Future Work

This work aims at the privacy-preserving detection of COVID-19 from chest X-ray images. While the detection task asks for ML in the form of an image classifier to distinguish between COVID-19 and Normal (no findings) in a patient's chest X-rays, the privacy part focuses on mitigating exploitative attacks that could compromise sensitive patient data. The primary attack of interest is the MIA, enabling an attacker to predict which samples were part of a model's training set. The ML approach uses CNNs to train a classification model on the COVID-19 Radiography Database [90, 91]. The privacy preservation is then implemented through PPML methods that guarantee DP at given privacy budgets ε. By gaining the DP attribute, the models should be less prone to MIAs [48, 75, 76].

The evaluation in Chapter 5 surveyed 44 architectural experiments[16] on the COVID-19 Radiography Database. The training procedures were composed of two different PPML algorithms, DP-SGD and PATE, in addition to non-private training. Next to the performance metric of classification accuracy, a privacy metric is essential to evaluate models for the given task. The privacy budget ε constitutes a theoretical privacy guarantee in the form of an upper bound. The real threat from attacks, on the other hand, is better represented using an estimated privacy budget ε_{AUC} that is directly based on the success of performed MIAs [85].

In contrast to the expectations from [48, 75, 76], DP showed no improvements in MIA defense on the COVID-19 detection task. Instead, the non-private models already presented the same repelling properties as the private models on all tested privacy levels. Additionally, at $\varepsilon = 0.1$ no private model was able to hold the promised guarantee when compared to their ε_{AUC}. Therefore, the best way to design a private image classification model detecting COVID-19 from chest X-ray images depends on the specific privacy needs. In terms of empirically estimated MIA defense, the non-private models are as secure as the private models. Thus, if only the defense against MIAs is of concern, the non-private ResNet50 with ImageNet pre-training achieving 97.6% classification accuracy is the best choice. Private DP-SGD training for $\varepsilon = 1$ is a more sensible alternative when a theoretical privacy guarantee is needed. The showed privacy-accuracy trade-off from using PATE is promising, but ultimately the algorithm could not be successfully applied using $\varepsilon = 1$ on the small COVID-19 dataset. The best private DP-SGD model reaches 74.1% accuracy and uses a ResNet18 pre-trained on pneumonia X-rays while also changing the ReLU activation function to the tanh function. Even though the accuracy-privacy trade-off of 23.5% is significant, the model performs 0.3% better than related work and keeps much tighter privacy guarantees[17] [71, 103]. Ultimately, the conflicting findings from the additional experiments on the MNIST database, where DP significantly increased MIA defense, indicate that PPML (or DP-SGD) is heavily task-dependent. Thus, research on one dataset might not carry over to others.

A brief outlook into possible future work concludes the last part of this section. A possible candidate for further evaluation is the FedDPGAN [106] model that, as of now, produces the best private COVID-19 detection results with 94.5% but lacks comparability due to not giving

[16] Without counting experiments on additional privacy levels or on a different dataset (MNIST).

[17] This work implements $\varepsilon = 1$ and reaches 74.1%. The next best performing result is 73.8% in [71] and uses $\varepsilon = 39.4$. [103] states an ε close to 6 but only reaches an accuracy of 71%.

the used privacy budget. Therefore, benchmarking the proposed model in a traceable and comparable environment like in this work would give more context to the result. In general, future ventures can benefit from implementing frameworks that increase data set size. These frameworks involve GANs synthesizing more data and FL enabling hospitals to securely train a joint model from their local data. Comparability is an issue in PPML research that could be improved by common privacy evaluations across different works. Further, estimating the empirical privacy risks through attacks should be established. Assessing possible threats allows to choose the appropriate privacy budget and thus, can optimize the accuracy-privacy trade-off. There are also general considerations for ML on medical images. As [139] proposes, performing lung segmentation on the X-ray images before training for the detection task might mitigate the issue of ML models picking non-disease-related markers from the images. The relevance of this additional step could be evaluated using heat maps that highlight what the ML model sees as the most relevant areas in the X-ray. Finally, these markings could further support professionals in screening for COVID-19 in the images [1].

Lucas Lange
3737104

Bibliography

[1] Tulin Ozturk et al. “Automated detection of COVID-19 cases using deep neural networks with X-ray images”. In: *Computers in biology and medicine* 121 (2020), p. 103792.

[2] Tom Balthazar. “Sharing health-data between hospitals and other care-providers: Towards legal clarity about what can be communicated to whom”. In: *Tijdschrift voor Geneeskunde* 74 (Feb. 2018), pp. 161–165. DOI: 10.2143/TVG.74.03.2002516.

[3] Edgar Lorente. *COVID-19 pneumonia - evolution over a week*. https://radiopaedia.org/cases/covid-19-pneumonia-evolution-over-a-week-1?lang=us. [Online; accessed 02-September-2021]. 2020.

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

[5] Michel D. Landry et al. “Early reflection on the global impact of COVID19, and implications for physiotherapy”. In: *Physiotherapy* 107 (2020), A1–A3.

[6] Xingzhi Xie et al. “Chest CT for typical coronavirus disease 2019 (COVID-19) pneumonia: relationship to negative RT-PCR testing”. In: *Radiology* 296.2 (2020), E41–E45.

[7] Ming-Yen Ng et al. “Imaging profile of the COVID-19 infection: radiologic findings and literature review”. In: *Radiology: Cardiothoracic Imaging* 2.1 (2020), e200034.

[8] Siddique Latif et al. “Leveraging data science to combat covid-19: A comprehensive review”. In: *IEEE Transactions on Artificial Intelligence* 1.1 (2020), pp. 85–103.

[9] Wei Zhao et al. “Relation between chest CT findings and clinical conditions of coronavirus disease (COVID-19) pneumonia: a multicenter study”. In: *American Journal of Roentgenology* 214.5 (2020), pp. 1072–1077.

[10] Yan Li and Liming Xia. “Coronavirus disease 2019 (COVID-19): role of chest CT in diagnosis and management”. In: *American Journal of Roentgenology* 214.6 (2020), pp. 1280–1286.

[11] Jeffrey P. Kanne et al. *Essentials for radiologists on COVID-19: an update—radiology scientific expert panel*. 2020.

[12] Daniel López Zúñiga and Miguel Ángel López Zúñiga. “COVID-19 diagnosis through image”. In: *Medicina Clínica (English Edition)* 155.3 (2020), p. 140.

[13] N. Narayan Das et al. “Automated deep transfer learning-based approach for detection of COVID-19 infection in chest X-rays”. In: *Irbm* (2020).

[14] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep learning”. In: *nature* 521.7553 (2015), pp. 436–444.

[15] Boukaye Boubacar Traore, Bernard Kamsu-Foguem, and Fana Tangara. “Deep convolution neural network for image recognition”. In: *Ecological Informatics* 48 (2018), pp. 257–268.

[16] Kunihiko Fukushima and Sei Miyake. “Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition”. In: *Competition and cooperation in neural nets*. Springer, 1982, pp. 267–285.

[17] Yann LeCun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems* 25 (2012), pp. 1097–1105.

[19] Nitish Srivastava et al. "Dropout: a simple way to prevent neural networks from overfitting". In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958.

[20] Sergey Ioffe and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *International conference on machine learning*. PMLR. 2015, pp. 448–456.

[21] Sinno Jialin Pan and Qiang Yang. "A survey on transfer learning". In: *IEEE Transactions on knowledge and data engineering* 22.10 (2009), pp. 1345–1359.

[22] Li Fei-Fei, Rob Fergus, and Pietro Perona. "One-shot learning of object categories". In: *IEEE transactions on pattern analysis and machine intelligence* 28.4 (2006), pp. 594–611.

[23] Hailong Li, Nehal A Parikh, and Lili He. "A novel transfer learning approach to enhance deep neural network classification of brain functional connectomes". In: *Frontiers in neuroscience* 12 (2018), p. 491.

[24] Lisa Torrey and Jude Shavlik. "Transfer learning". In: *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*. IGI global, 2010, pp. 242–264.

[25] Md Zahangir Alom et al. "A state-of-the-art survey on deep learning theory and architectures". In: *Electronics* 8.3 (2019), p. 292.

[26] Lixin Duan, Dong Xu, and Shih-Fu Chang. "Exploiting web images for event recognition in consumer videos: A multiple source domain adaptation approach". In: *2012 IEEE Conference on computer vision and pattern recognition*. IEEE. 2012, pp. 1338–1345.

[27] Yi Yao and Gianfranco Doretto. "Boosting for transfer learning with multiple sources". In: *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE. 2010, pp. 1855–1862.

[28] Francois Chollet. *Deep learning with Python*. Simon and Schuster, 2017.

[29] Vimal Manohar et al. "A teacher-student learning approach for unsupervised domain adaptation of sequence-trained ASR models". In: *2018 IEEE Spoken Language Technology Workshop (SLT)*. IEEE. 2018, pp. 250–257.

[30] Bin Li, Qiang Yang, and Xiangyang Xue. "Can movies and books collaborate? cross-domain collaborative filtering for sparsity reduction". In: *Twenty-First international joint conference on artificial intelligence*. 2009.

[31] Julius Kunze et al. "Transfer Learning for Speech Recognition on a Budget". In: *arXiv e-prints* (2017), arXiv–1706.

[32] John S. II Davis and Osonde A. Osoba. *Privacy Preservation in the Age of Big Data: A Survey*. Santa Monica, CA: RAND Corporation, 2016. DOI: 10.7249/WR1161.

[33] Arvind Narayanan and Vitaly Shmatikov. "Robust de-anonymization of large sparse datasets". In: *2008 IEEE Symposium on Security and Privacy (sp 2008)*. IEEE. 2008, pp. 111–125.

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

[35] Aurélien Bellet. *Privacy Preserving Machine Learning - Lecture 1: Introduction & Course Overview*. http://researchers.lille.inria.fr/abellet/teaching/ppml_lectures/lec1.pdf. [Online; accessed 31-August-2021]. 2020.

[36] Apple. *Apple Differential Privacy Technical Overview*. https://www.apple.com/privacy/docs/Differential_Privacy_Overview.pdf. [Online; accessed 30-September-2021].

[37] Apple Differential Privacy Team. *Learning with Privacy at Scale*. `https://machinelearning.apple.com/research/learning-with-privacy-at-scale`. [Online; accessed 30-September-2021]. 2017.

[38] Jun Tang et al. "Privacy Loss in Apple's Implementation of Differential Privacy on MacOS 10.12". In: *arXiv e-prints* (2017), arXiv–1709.

[39] Cynthia Dwork et al. "Calibrating noise to sensitivity in private data analysis". In: *Theory of cryptography conference*. Springer. 2006, pp. 265–284.

[40] Jianping He and Lin Cai. "Differential private noise adding mechanism: Basic conditions and its application". In: *2017 American Control Conference (ACC)*. IEEE. 2017, pp. 1673–1678.

[41] Martin Abadi et al. "Deep learning with differential privacy". In: *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 2016, pp. 308–318.

[42] Cynthia Dwork, Aaron Roth, et al. "The algorithmic foundations of differential privacy." In: *Foundations and Trends in Theoretical Computer Science* 9.3-4 (2014), pp. 211–407.

[43] Pathum Chamikara Mahawaga Arachchige et al. "Local differential privacy for deep learning". In: *IEEE Internet of Things Journal* 7.7 (2019), pp. 5827–5842.

[44] Cynthia Dwork and Vitaly Feldman. "Privacy-preserving prediction". In: *Conference On Learning Theory*. PMLR. 2018, pp. 1693–1702.

[45] Laurens van der Maaten and Awni Hannun. "The Trade-Offs of Private Prediction". In: *arXiv e-prints* (2020), arXiv–2007.

[46] Xiaokui Xiao and Yufei Tao. "Output perturbation with query relaxation". In: *Proceedings of the VLDB Endowment* 1.1 (2008), pp. 857–869.

[47] Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. "Differentially private empirical risk minimization." In: *Journal of Machine Learning Research* 12.3 (2011).

[48] Md Atiqur Rahman et al. "Membership Inference Attack against Differentially Private Deep Learning Model." In: *Trans. Data Priv.* 11.1 (2018), pp. 61–79.

[49] Nicolas Papernot. *Machine Learning with Differential Privacy in TensorFlow*. `http://www.cleverhans.io/privacy/2019/03/26/machine-learning-with-differential-privacy-in-tensorflow.html`. [Online; accessed 08-April-2021]. 2019.

[50] Davide Testuggine and Ilya Mironov. *Differential Privacy Series Part 1 | DP-SGD Algorithm Explained*. `https://medium.com/pytorch/differential-privacy-series-part-1-dp-sgd-algorithm-explained-12512c3959a3`. [Online; accessed 13-March-2021]. 2020.

[51] Nicolas Papernot and Ian Goodfellow. *Privacy and machine learning: two unexpected allies?* `http://www.cleverhans.io/privacy/2018/04/29/privacy-and-machine-learning.html`. [Online; accessed 08-April-2021]. 2018.

[52] Nicolas Papernot et al. "Semi-supervised Knowledge Transfer for Deep Learning from Private Training Data". In: *arXiv e-prints* (2016), arXiv–1610.

[53] Nicolas Papernot et al. "Scalable Private Learning with PATE". In: *arXiv e-prints* (2018), arXiv–1802.

[54] Antonia Creswell et al. "Generative adversarial networks: An overview". In: *IEEE Signal Processing Magazine* 35.1 (2018), pp. 53–65.

[55] Ian Goodfellow et al. "Generative adversarial nets". In: *Advances in neural information processing systems* 27 (2014).

[56] Youyang Qu et al. "GAN-DP: generative adversarial net driven differentially privacy-preserving big data publishing". In: *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE. 2019, pp. 1–6.

[57] Yifan Wu et al. "Privacy-protective-GAN for privacy preserving face de-identification". In: *Journal of Computer Science and Technology* 34.1 (2019), pp. 47–60.

[58] Zuobin Xiong et al. "Privacy-preserving auto-driving: a GAN-based approach to protect vehicular camera data". In: *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE. 2019, pp. 668–677.

[59] Yi Liu et al. "Ppgan: Privacy-preserving generative adversarial network". In: *2019 IEEE 25Th international conference on parallel and distributed systems (ICPADS)*. IEEE. 2019, pp. 985–989.

[60] Liyang Xie et al. "Differentially Private Generative Adversarial Network". In: *arXiv e-prints* (2018), arXiv–1802.

[61] Aleksei Triastcyn and Boi Faltings. "Generating Artificial Data for Private Deep Learning". In: *Proceedings of the PAL: Privacy-Enhancing Artificial Intelligence and Language Technologies, AAAI Spring Symposium Series*. CONF. 2019.

[62] Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. "Smooth sensitivity and sampling in private data analysis". In: *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*. 2007, pp. 75–84.

[63] Zhanglong Ji, Zachary C. Lipton, and Charles Elkan. "Differential Privacy and Machine Learning: a Survey and Review". In: *arXiv e-prints* (2014), arXiv–1412.

[64] Brendan McMahan et al. "Communication-efficient learning of deep networks from decentralized data". In: *Artificial intelligence and statistics*. PMLR. 2017, pp. 1273–1282.

[65] Oded Goldreich, Silvio Micali, and Avi Wigderson. "How to play any mental game, or a completeness theorem for protocols with honest majority". In: *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*. 2019, pp. 307–328.

[66] Renuga Kanagavelu et al. "Two-phase multi-party computation enabled privacy-preserving federated learning". In: *2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID)*. IEEE. 2020, pp. 410–419.

[67] Huafei Zhu. "On the relationship between (secure) multi-party computation and (secure) federated learning". In: *arXiv e-prints* (2020), arXiv–2008.

[68] David Byrd and Antigoni Polychroniadou. "Differentially Private Secure Multi-Party Computation for Federated Learning in Financial Applications". In: *arXiv e-prints* (2020), arXiv–2010.

[69] Robin C. Geyer, Tassilo Klein, and Moin Nabi. "Differentially Private Federated Learning: A Client Level Perspective". In: *arXiv e-prints* (2017), arXiv–1712.

[70] Mohammad Malekzadeh et al. "Dopamine: Differentially Private Federated Learning on Medical Data". In: *arXiv e-prints* (2021), arXiv–2101.

[71] Trang-Thi Ho et al. "DPCOVID: Privacy-Preserving Federated Covid-19 Detection". In: *arXiv e-prints* (2021), arXiv–2110.

[72] Jianjiang Feng and Anil K. Jain. "Fingerprint reconstruction: from minutiae to phase". In: *IEEE transactions on pattern analysis and machine intelligence* 33.2 (2010), pp. 209–223.

[73] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. "Model inversion attacks that exploit confidence information and basic countermeasures". In: *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. 2015, pp. 1322–1333.

[74] Mohammad Al-Rubaie and J. Morris Chang. "Reconstruction attacks against mobile-based continuous authentication systems in the cloud". In: *IEEE Transactions on Information Forensics and Security* 11.12 (2016), pp. 2648–2663.

[75] Reza Shokri et al. "Membership inference attacks against machine learning models". In: *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2017, pp. 3–18.

[76] Cynthia Dwork et al. "Preserving statistical validity in adaptive data analysis". In: *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*. 2015, pp. 117–126.

[77] Franziska Boenisch. *Attacks against Machine Learning Privacy (Part 2): Membership Inference Attacks with TensorFlow Privacy*. `https://franziska-boenisch.de/posts/2021/01/membership-inference/`. [Online; accessed 30-December-2021]. 2021.

[78] Nicholas Carlini et al. "The secret sharer: Evaluating and testing unintended memorization in neural networks". In: *28th {USENIX} Security Symposium ({USENIX} Security 19)*. 2019, pp. 267–284.

[79] Congzheng Song, Thomas Ristenpart, and Vitaly Shmatikov. "Machine learning models that remember too much". In: *Proceedings of the 2017 ACM SIGSAC Conference on computer and communications security*. 2017, pp. 587–601.

[80] Samuel Yeom et al. "Privacy risk in machine learning: Analyzing the connection to overfitting". In: *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*. IEEE. 2018, pp. 268–282.

[81] Moritz Hardt, Ben Recht, and Yoram Singer. "Train faster, generalize better: Stability of stochastic gradient descent". In: *International Conference on Machine Learning*. PMLR. 2016, pp. 1225–1234.

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

[83] Ahmed Salem et al. "ML-Leaks: Model and Data Independent Membership Inference Attacks and Defenses on Machine Learning Models". In: *arXiv e-prints* (2018), arXiv–1806.

[84] Corinna Cortes and Mehryar Mohri. "Confidence intervals for the area under the ROC curve". In: *Advances in neural information processing systems* 17 (2005), pp. 305–312.

[85] Mani Malek et al. "Antipodes of Label Differential Privacy: PATE and ALIBI". In: *arXiv e-prints* (2021), arXiv–2106.

[86] Zihang Dai et al. "CoAtNet: Marrying Convolution and Attention for All Data Sizes". In: *arXiv e-prints* (2021), arXiv–2106.

[87] Rikiya Yamashita et al. "Convolutional neural networks: an overview and application in radiology". In: *Insights into imaging* 9.4 (2018), pp. 611–629.

[88] Mohamed Loey, Florentin Smarandache, and Nour Eldeen M. Khalifa. "Within the lack of chest COVID-19 X-ray dataset: a novel detection model based on GAN and deep transfer learning". In: *Symmetry* 12.4 (2020), p. 651.

[89] Ahmed Sedik et al. "Deploying machine and deep learning models for efficient data-augmented detection of COVID-19 infections". In: *Viruses* 12.7 (2020), p. 769.

[90] Muhammad EH Chowdhury et al. "Can AI help in screening viral and COVID-19 pneumonia?" In: *IEEE Access* 8 (2020), pp. 132665–132676.

[91] Tawsifur Rahman et al. "Exploring the Effect of Image Enhancement Techniques on COVID-19 Detection using Chest X-rays Images". In: *Computers in Biology and Medicine* (2021), p. 104319.

[92] Thanh Thi Nguyen. "Artificial Intelligence in the Battle against Coronavirus (COVID-19): A Survey and Future Research Directions". In: *arXiv e-prints* (2020), arXiv–2008.

[93] Tarik Alafif et al. "Machine and deep learning towards COVID-19 diagnosis and treatment: survey, challenges, and future directions". In: *International Journal of Environmental Research and Public Health* 18.3 (2021), p. 1117.

[94] Arshia Rehman et al. "Improving coronavirus (COVID-19) diagnosis using deep transfer learning". In: *MedRxiv* (2020).

[95] Ali Narin, Ceren Kaya, and Ziynet Pamuk. "Automatic detection of coronavirus disease (covid-19) using x-ray images and deep convolutional neural networks". In: *Pattern Analysis and Applications* (2021), pp. 1–14.

[96] Sejuti Rahman et al. "Deep Learning–Driven Automated Detection of COVID-19 from Radiography Images: a Comparative Analysis". In: *Cognitive Computation* (2021), pp. 1–30.

[97] Sheetal Rajpal et al. "Using handpicked features in conjunction with ResNet-50 for improved detection of COVID-19 from chest X-ray images". In: *Chaos, Solitons & Fractals* 145 (2021), p. 110749.

[98] Ines Feki et al. "Federated learning for COVID-19 screening from Chest X-ray images". In: *Applied Soft Computing* 106 (2021), p. 107330.

[99] Xinyue Zhang et al. "Adaptive Privacy Preserving Deep Learning Algorithms for Medical Data". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2021, pp. 1169–1178.

[100] Jingjing Yang, Jinzhao Wu, and Xiaojing Wang. "Convolutional neural network based on differential privacy in exponential attenuation mode for image classification". In: *IET Image Processing* (2020).

[101] Vinith M. Suriyakumar et al. "Chasing Your Long Tails: Differentially Private Prediction in Health Care Settings". In: *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*. 2021, pp. 723–734.

[102] Sahib Singh et al. "Benchmarking Differentially Private Residual Networks for Medical Imagery". In: *arXiv e-prints* (2020), arXiv–2005.

[103] Zümrüt Müftüoğlu, M Ayyüce Kizrak, and Tülay Yildlnm. "Differential Privacy Practice on Diagnosis of COVID-19 Radiology Imaging Using EfficientNet". In: *2020 International Conference on INnovations in Intelligent SysTems and Applications (INISTA)*. IEEE. 2020, pp. 1–6.

[104] Shreyansh Singh and K.K. Shukla. "Privacy-preserving Machine Learning for Medical Image Classification". In: *arXiv e-prints* (2021), arXiv–2108.

[105] Danni Yuan et al. "Collaborative Deep Learning for Medical Image Analysis with Differential Privacy". In: *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE. 2019, pp. 1–6.

[106] Longling Zhang et al. "FedDPGAN: Federated Differentially Private Generative Adversarial Networks Framework for the Detection of COVID-19 Pneumonia". In: *Information Systems Frontiers* (2021), pp. 1–13.

[107] François Chollet et al. *Keras*. `https://keras.io`. 2015.

[108] Martin Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[109] Daniel S. Kermany et al. "Identifying medical diagnoses and treatable diseases by image-based deep learning". In: *Cell* 172.5 (2018), pp. 1122–1131.

[110] Jia Deng et al. "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[111] Minyoung Huh, Pulkit Agrawal, and Alexei A. Efros. "What makes ImageNet good for transfer learning?" In: *arXiv e-prints* (2016), arXiv–1608.

[112] Nicolas Papernot et al. "Tempered Sigmoid Activations for Deep Learning with Differential Privacy". In: *arXiv e-prints* (2020), arXiv–2007.

[113] Da Yu et al. "Do Not Let Privacy Overbill Utility: Gradient Embedding Perturbation for Private Learning". In: *arXiv e-prints* (2021), arXiv–2102.

[114] Eugene Bagdasaryan, Omid Poursaeed, and Vitaly Shmatikov. "Differential privacy has disparate impact on model accuracy". In: *Advances in Neural Information Processing Systems* 32 (2019), pp. 15479–15488.

[115] Lean Yu, Shouyang Wang, and K.K. Lai. "An integrated data preparation scheme for neural network data analysis". In: *IEEE Transactions on Knowledge and Data Engineering* 18.2 (2006), pp. 217–230. DOI: 10.1109/TKDE.2006.22.

[116] Lutz Prechelt. "Early stopping-but when?" In: *Neural Networks: Tricks of the trade*. Springer, 1998, pp. 55–69.

[117] Connor Shorten and Taghi M. Khoshgoftaar. "A survey on image data augmentation for deep learning". In: *Journal of Big Data* 6.1 (2019), pp. 1–48.

[118] Min Zhu et al. "Class weights random forest algorithm for processing class imbalanced medical data". In: *IEEE Access* 6 (2018), pp. 4641–4652.

[119] James Jordon, Jinsung Yoon, and Mihaela Van Der Schaar. "PATE-GAN: Generating synthetic data with differential privacy guarantees". In: *International conference on learning representations*. 2018.

[120] Kang Liu, Benjamin Tan, and Siddharth Garg. "Subverting Privacy-Preserving GANs: Hiding Secrets in Sanitized Images". In: *arXiv e-prints* (2020), arXiv–2009.

[121] Qiang Yang et al. "Federated learning". In: *Synthesis Lectures on Artificial Intelligence and Machine Learning* 13.3 (2019), pp. 1–207.

[122] Bargav Jayaraman and David Evans. "Evaluating differentially private machine learning in practice". In: *28th {USENIX} Security Symposium ({USENIX} Security 19)*. 2019, pp. 1895–1912.

[123] Sheila F. O'Brien and Qi Long Yi. "How do I interpret a confidence interval?" In: *Transfusion* 56.7 (2016), pp. 1680–1683.

[124] Raef Bassily, Adam Smith, and Abhradeep Thakurta. "Private empirical risk minimization: Efficient algorithms and tight error bounds". In: *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*. IEEE. 2014, pp. 464–473.

[125] Nicolas Papernot et al. *Making the Shoe Fit: Architectures, Initializations, and Tuning for Learning with Privacy*. 2020. URL: https://openreview.net/forum?id=rJg851rYwH.

[126] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[127] Mingxing Tan and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks". In: *International Conference on Machine Learning*. PMLR. 2019, pp. 6105–6114.

[128] Papers With Code. *Convolutional Neural Networks*. `https://paperswithcode.com/methods/category/convolutional-neural-networks`. [Online; accessed 07-December-2021]. 2021.

[129] AA Speranskaya. "Radiological signs of a new coronavirus infection COVID-19". In: *Diagnostic radiology and radiotherapy* 11.1 (2020), pp. 18–25.

[130] Depeng Xu, Wei Du, and Xintao Wu. "Removing Disparate Impact of Differentially Private Stochastic Gradient Descent on Model Accuracy". In: *arXiv e-prints* (2020), arXiv–2003.

[131] Kaichao You et al. "How Does Learning Rate Decay Help Modern Neural Networks?" In: *arXiv e-prints* (2019), arXiv–1908.

[132] Alex Krizhevsky and Geoffrey Hinton. *Learning multiple layers of features from tiny images*. Tech. rep. 0. Toronto, Ontario: University of Toronto, 2009.

[133] Ali Davody et al. *On the effect of normalization layers on Differentially Private training of deep Neural networks*. 2021. arXiv: `2006.10919 [cs.LG]`.

[134] Christian Szegedy et al. "Inception-v4, inception-resnet and the impact of residual connections on learning". In: *Thirty-first AAAI conference on artificial intelligence*. 2017.

[135] Erick Galinkin. "The Influence of Dropout on Membership Inference in Differentially Private Models". In: *arXiv e-prints* (2021), arXiv–2103.

[136] Prateek Jain et al. "To Drop or Not to Drop: Robustness, Consistency and Differential Privacy Properties of Dropout". In: *arXiv e-prints* (2015), arXiv–1503.

[137] Rushi Longadge and Snehalata Dongre. "Class Imbalance Problem in Data Mining Review". In: *arXiv e-prints* (2013), arXiv–1305.

[138] Thomas P Quinn, Vuong Le, and Adam PA Cardilini. "Test set verification is an essential step in model building". In: *Methods in Ecology and Evolution* 12.1 (2021), pp. 127–129.

[139] Saman Motamed, Patrik Rogalla, and Farzad Khalvati. "RANDGAN: randomized generative adversarial network for detection of COVID-19 in chest X-ray". In: *Scientific Reports* 11.1 (2021), pp. 1–10.

Declaration of Authorship

Ich versichere, dass ich die vorliegende Arbeit mit dem Thema:

„Privacy-Preserving Detection of COVID-19 in X-Ray Images“

selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe, insbesondere sind wörtliche oder sinngemäße Zitate als solche gekennzeichnet. Mir ist bekannt, dass Zuwiderhandlung auch nachträglich zur Aberkennung des Abschlusses führen kann. Ich versichere weiterhin, dass das elektronische Exemplar mit den gedruckten Exemplaren übereinstimmt.

Leipzig, den 12. Januar 2022

Lucas Lange