DIFFERENTIALLY PRIVATE LEARNING

STAT 672 Final Project

Nathen Huang

MS Statistical Science
Spring 2019

Table of Contents

Introduction

Preface

> *"Privacy is not something that I'm merely entitled to, it's an absolute prerequisite."*
> *– Marlon Brando*

In an age where information and data about every facet of people's lives are omnipresent, it is often difficult to maintain a semblance of privacy. Being an active netizen naturally requires that individuals provide at least some personal information in order to partake in online services and world web delights; however, people have become zealous to exchange their data for expeditious ease, customized content and ubiquitous attention. In return, companies have become notorious for not only cashing in on people's eagerness to disclose their information, but also for their inability to protect this information at multiple stages of information ingest and disclosure. Signs of this pervasive oversharing culture online (and subsequent capitalization upon/spillage of provided information by omnipotent corporations) suggests a dystopian future in which the publicization of private information is inevitable. However, such a doom-and-gloom scenario may be rather premature given recent advances in the cryptographic and machine learning fields.

Overview of Differential Privacy

In the last few years, data science and machine learning have become extremely popular due to the powerful methods for detecting patterns and new algorithms and analytical techniques that have been developed- as well as the creative application of these innovative methods to various datasets. Machine learning algorithms have enabled practitioners to identify trends in data to better deliver solutions for people in all areas of work, and the enthusiasm to optimize learning over datasets to provide deeper and deeper insights into data has exploded. Yet, while the enthusiasm for algorithmic optimization has grown stronger, there has also been a concurrent rise of concern regarding the dangers of machine learning applications in divulging information about the datasets that models are trained on. Privacy in machine learning has itself become a hot topic for research, as it has revealed that models can (and have) leaked information about the data records on which they are trained- as demonstrated by AOL's released user search data[1], the Netflix Prize's public user data (McSherry and Mironov (2009)[2]), and Shokri et al.'s (2017) research on membership inference (determining whether a record was part of the training set or not) attacks against machine learning models. [3]

Given the keen ability of these computational algorithms to detect patterns where humans cannot, it is troubling that providing insights on individuals to better serve them can, consequently, compromise their privacy. A common solution to data privacy has often been to simply strip out identifying information and thus "anonymize" the data; however, this approach to protecting information is insufficient, as cross-referencing other sources of data and deduction of information based on data sources alone can often provide attackers with backdoor approaches to matching individuals. Besides stripping out sensitive information,

other approaches to preserving individual privacy- including aggregating queries, noising query results, and mapping data deterministically to create perturbation- can either still fail to deidentify individuals with a little reverse engineering or complicate the subsequent actual learning from the data due to excessively strong privacy guarantees.[4] Thus, while preserving the data's privacy is extremely important, it is still important that protecting integrity does not come with a substantial sacrifice of accuracy in later learning on the dataset. Ultimately, the goal of this problem of learning from data is best summarized by Dwork (2006) in her seminal paper: "the risk to one's privacy… should not substantially increase as a result of participating in a statistical database." [5]

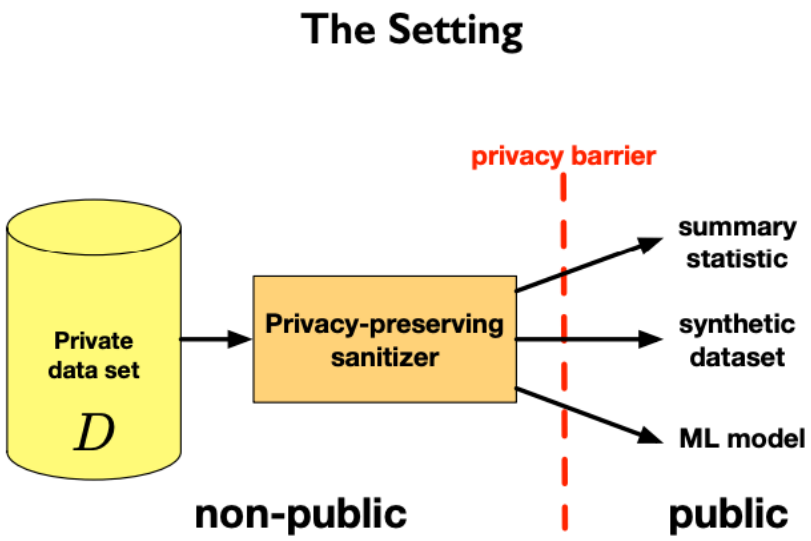


Figure 1: General Paradigm of Differential Privacy, Chaudhuri and Sarwate (2017) [6]

The proposed solution to this challenge that I allude to here, of course, is differential privacy- which broadly refers to the series of mathematical and computational practices used to obscure individual-level data by algorithmically inserting statistical noise to aggregated results. Practically, differential privacy (henceforth referred to as "DP") has already been used in a wide variety of settings: by the Census to anonymize commuting patterns of the US population; by Apple to glean insights from and feed suggestions for application usage; and by Google to randomize responses to questions to prevent individual identifiability while collecting information from end-user, client-side software. In addition to the statistical query-obscuring approaches that DP is known for, DP algorithms are being used in "machine learning, game theory and economic mechanism design, statistical estimation" [7]; for the sake of this report, we are interested in exploring some of the machine learning applications of DP algorithms that constitute the "learning" subset of DP algorithms.

Report Abstract

In this paper, I will not only explore the underlying theory of DP algorithms, but also apply DP principles and methods to a sample dataset to demonstrate the capabilities (and relevance) of DP learning in image recognition. Principles of DP can be extended from the data analysis problem explored in this paper to other types of privacy-preserving analyses. The case discussed in this report is an explicit example of preserving data integrity while simultaneously permitting subsequent analysis and reporting on the underlying data in pursuit of source privacy. It is apparent how important DP is to problems in image classification, given the potential that such classification techniques could have in such instances as security or law enforcement. At the end of this report, I discuss my findings, opportunities for methodological improvement and other potential areas of investigation, and concluding impressions on DP learning.

Methods of Differential Privacy and Differentially Private Learning

Formal Definition

To understand exactly what DP methods achieve, I begin with DP's formal definition which clarifies their algorithms' purpose. For two adjacent datasets A and B (also written in notation later in the paper as D and D'), where there is exactly one record in one but not the other, Dwork (paraphrased by McSherry and Mironov (2009)) states that:

> A randomized computation M satisfies ϵ-differential privacy if for any adjacent datasets A and B, and any subset S of possible outcomes Range(M):

$$\mathbf{Pr}[M(A) \in S] \quad \leq \quad \exp(\epsilon) \times \mathbf{Pr}[M(B) \in S]$$

This formal definition of DP demonstrates that inference about the presence or absence of any single record is bounded by the factor exp(ϵ). This formulation, referred to as "pure DP," is in contrast to a relaxed form "approximate DP" where the term $\delta > 0$ (the probability bound of an arbitrary change in model behavior) is added to right hand side of the formal definition (also known as (ϵ, δ)-DP). Intuitively, this definition means that the presence of any participant in the statistical database would not become significantly more or less likely from the outputs of the mechanism M run on the datasets A and B. By this definition, preventing disclosure of information is not an absolute guarantee, but it is relative guarantee (within a multiplicative factor) that participating and not participating in the data are equally likely; in (ϵ, δ)-DP, by extension, there is a (1- δ) probability that the mechanism M preserves ϵ-DP.

Global Sensitivity Model

There are many different approaches and applications of DP. One of the simplest and most approaches is the Global Sensitivity Model, which applies random noise to the measurement and subsequently mitigates the role of a single record on the outcome. Adding noise based on the Laplace or Gaussian distribution is an especially popular

approach, which McSherry and Mironov (2009) state that for a function f: $D^n \rightarrow \mathbb{R}^d$, privacy can be achieved as:

> Define M(X) to be f(X) + Laplace(0, σ)^d. M provides ϵ-differential privacy whenever:
>
> $$\sigma \geq \max_{A \approx B} \|f(A) - f(B)\|_1 / \epsilon \ .$$

The value of σ for Laplace noise is calibrated according to the (L_1) sensitivity of the function- which is the maximum amount over the domain of f that any single row in the statistical database changes the output. Formally, this is defined by Dwork et al. (2006) as:

> The L_1 sensitivity of a function $f : D^n \rightarrow \mathbb{R}_d$ is the smallest number S(f) such that for A,B **ϵ** D^n which differ in a single entry,
>
> $$\|f(\mathbf{x}) - f(\mathbf{x}')\|_1 \leq S(f) \ .$$

According to Dwork et al. (2006), noising the data (which constitutes the value σ of the Laplace distribution) according to Laplace(S(f)/ ϵ) ensures ϵ-indistinguishability for a query function f with sensitivity S(f).

Learning from Differentially Private Data

While adding noise with DP approaches enables us to obscure the source of the data that we are working with (as well as the metrics that we report), there is an inherent degradation of utility that occurs under these methods. By adding too much noise (set by the ϵ "privacy budget"), training the model on DP-noised data decreases the ability of the trained model to accurately predict information with test data as the distribution of the training data becomes too dissimilar from test datasets that may be drawn from a similar source.

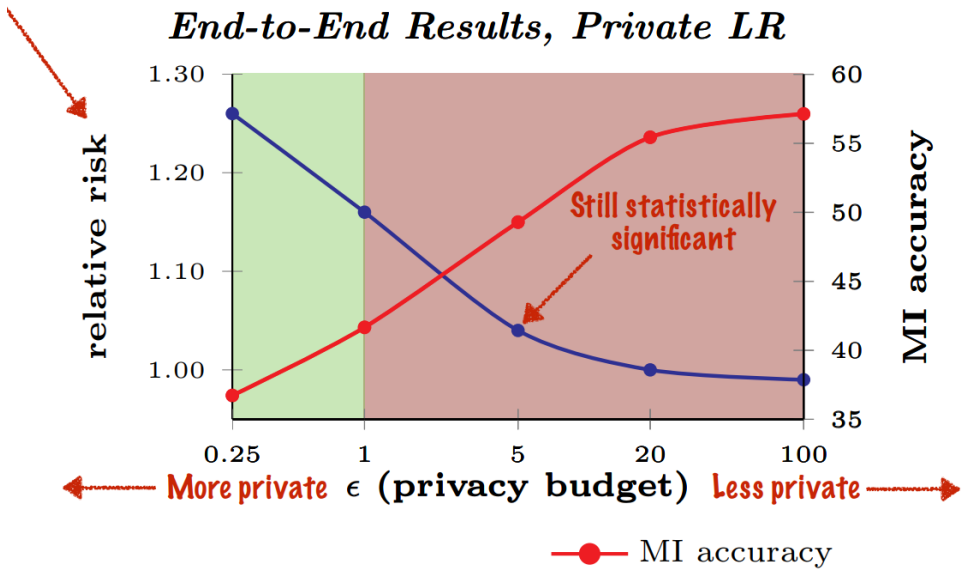


Figure 2: Trade-offs between Accuracy and Privacy in Differential Private Learning, Fredrikson et al. (2014)

Therefore, while it can be critical to secure the data upon which models are trained, the optimal privacy term must be evaluated to ensure only a minimal loss of performance in trained models against future test sets. Ultimately good learning algorithms should be able to generalize to the population distribution (since learning is about populations, not individuals), so there will be important privacy-accuracy-sample size trade-offs in pursuit of DP-learning. As Chaudhuri and Sarwate (2017) state plainly in their presentation at NIPS, "(ϵ, δ)-DP algorithms have better accuracy than ϵ-DP algorithms at the cost of a weaker privacy guarantee."

In addition to noising training data as discussed in the Global Sensitivity Model approach, there are other important places where DP can be introduced in learning.

$$\mathbf{w}^* = \underset{\mathbf{w}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \ell(\mathbf{w}, (\mathbf{x}_i, y_i)) + \lambda R(\mathbf{w})$$

The empirical risk minimization framework that is common for prediction problems is not often private, as the mere presence or absence of a record between two statistical databases D and D' that differ by only one record can be easily be determined from the resultant w* vs w'* vector. Similarly, with kernel-based methods that produce classifiers with such high degree of specificity, mere black-box access to w would enable someone to learn what the points being used were. In order to address these potential weaknesses, statistical learning in convex optimization can be modified by adding perturbation at all three steps of the process (reading in data, generating the objective function, and

performing minimization). Subsequently, adding privacy perturbations to various stages of the ERM process produces varying results of model performance.

Methods

In this section I detail both the data and subsequent analyses performed on the selected data that incorporates DP in statistical learning according to the Global Sensitivity Model.

Dataset

The data used in this analysis was sourced from the German Traffic Sign Dataset shared by the Institut für Neuroinformatik at Ruhr-Universität Bochum.[9] The database contains more than 50,000 images of traffic signs in Germany that span 43 different class labels. The images are often used as a benchmark for single-image, multi-classification, and both the database's extensive size and unique appearance of each real-world sign in the dataset make it a well-defined source for analysis. The dataset employed in this project was sourced from GitHub and was a Python pickled, resized dataset of the 50,000-plus images formatted as 32x32 pixel images.

To classify the images for analysis, preprocessing of the images was first necessary to achieve the best possible results by making the images conducive for classification. After the data was split into train (34,799 images), validation (4,410 images), and test (12,630 images) sets, the training data was also shuffled to create randomness in the training set, grayscaled with OpenCV, contrast-enhanced with Local Histogram Equalization to expand the range of pixel intensity, and lastly normalized to limit the range of pixel intensity values to have a mean of zero and the same variance across the values in the pixel matrix.

Below is a sample of the training images after they have been transformed in the manner of the aforementioned preprocessing:

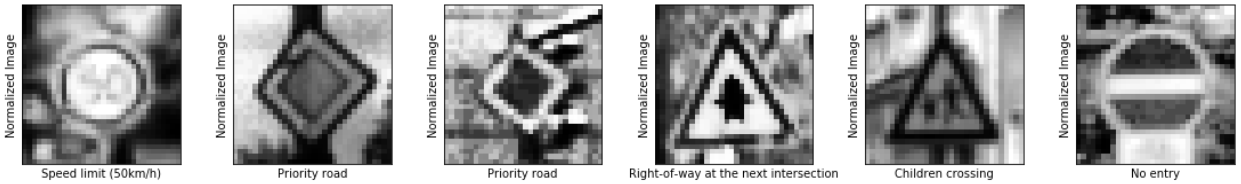


Figure 3: Sample training images with classification labels

This dataset was chosen for this paper for its vast size, clean image standardization, and extensive documentation that enabled me to focus on demonstrating DP learning capabilities, rather than tedious data collection and preparation. Not only is sourcing and munging data for deep learning an incredibly time-intensive process, neural networks do not learn patterns sufficiently on small datasets; likewise acquiring high-quality, standard data of this scale on my own would be beyond the scope of this project.

Implementation

Data analysis for this paper was conducted entirely in Python. Principles of DP were applied by encoding the process of DP mentioned in papers referenced throughout this paper, but otherwise were derived from the logic of Sijie Xiong at Rutgers University's dp-stats Python library on GitLab (particularly for logic on noising data).[10] For the actual analysis component, neural networks were constructed using the TensorFlow.Keras library.[11] Ingestion, preparation, and preprocessing of the data was done in accordance with Mohamed Ameen's German Traffic Sign Classification Using TensorFlow tutorial on GitHub.[12] Any other Python libraries used were primarily used in preprocessing and visualizing the resultant dataset- particularly the OpenCV (cv2) and matplotlib libraries.

Image Classification with Neural Networks

Given the scale of this dataset (and its sourcing from an extensive tutorial on training images with Convolutional Neural Networks via GitHub), it was sensible to classify the data using a feedforward neural network (hereafter referred colloquially as the "neural network") via TensorFlow.

At a high level, the neural network intakes a vector of data through the input layer and feeds it through the hidden layers- each containing a series of nodes that represent a linear combination of weight coefficients. After each layer, the weights are summed and then passed through the layer's activation function to determine a biased weighted sum of the input to determine how much of the nodes are passed on to the next node layer. The final layer, the output layer, is also modified by an activation function that determines the output based on the analytical problem at hand. For multi-class classification in the data that we are working with (43 image classes), we modify our output layer using a cross entropy activation function.

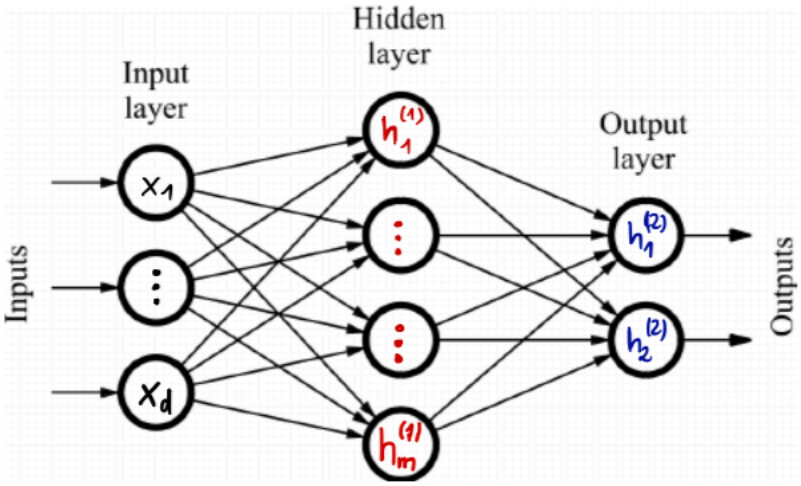


Figure 4: Basic Structure of a Neural Network (Lecture 13 Notes)

While training the data, the Cost function of the model is minimized as learnable parameters are learned and updated in the neural network. In practice, Gradient Descent is often used in deep learning to find the minimum of the loss function. Updating the weights during model fitting with Gradient Descent is an iterative process; each pass of the entire input dataset forward and backward through the neural network once to optimize the learning is referred to as an epoch. As the number of epochs employed increases, the weights of the neural network are updated more which boosts the performance of an underfitting neural network (at the risk of eventually overfitting it with too high a number of epochs).

Model Fitting

To begin the classification problem, a simple neural network with three hidden layers is constructed. The input layer intakes a 32 x 32 matrix of each image's pixels (a 2-dimensional array) from the training set and then flattens the matrix into 1-dimensional array (with 1024 elements for the 32 x 32 pixels). Then, in the next three layers, 2 ReLU layers with 256 and 128 nodes respectively and then 1 sigmoid layer with 64 nodes are included. Lastly, 43 nodes are employed in the final softmax layer to return an array of 43 probability scores for the probability that the inputted image belongs to one of the 43 classes.

Once created, the model is trained using the Keras sparse_categorical_crossentropy loss function for multi-class classification, an Adam optimizer (a popular default optimizing algorithm that is an alternative to stochastic gradient descent), and then monitored for model performance using accuracy as a metric. After the model has been built and calibrated, we will now train the neural network. First, we will feed it the preprocessed training images and the corresponding labels. After the images are run through the network and images and labels are matched with each other and the network will learn to match patterns in images and labels.

Model Validation

Once the model has been trained on the initial set, we validate it by running it against a new dataset. In this set of images, we will first utilize the validation set as a result of the trained model to evaluate accuracy. Model tuning was primarily conducted in two ways: 1. adding and removing the number of hidden layers (between ReLU and Sigmoid activation functions) and adjusting the number of nodes in each layer and 2. adjusting the number of epochs used when fitting the data (from a range 5 to 20 epochs). After the model hyperparameters are adjusted until the model's accuracy is improved, we can then use the optimized trained network to evaluate the available validation set and measure its accuracy outside of trained data. Once an appropriate validation accuracy is achieved while minimizing overfitting over the validation set, the test set separated out in the initial preprocessing of the data can also be used to confirm the official classification accuracy of the network.

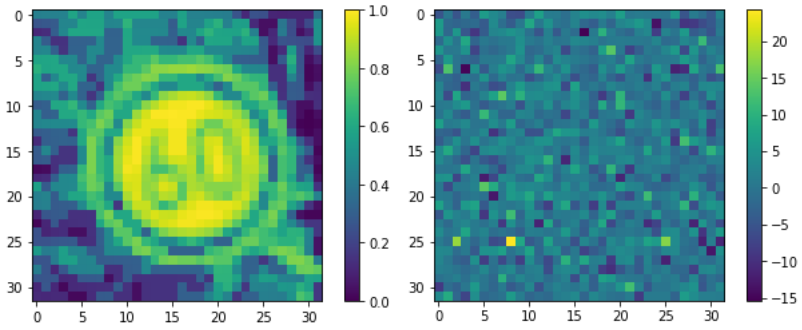
DP Image Classification

After we have determined the success of the neural network in classifying the non-DP traffic signs data, we are interested in juxtaposing its performance against the DP-data. The general algorithm for preparing and assessing the resulting privacy-preserved dataset in image classification is best described by Senekane (2019)[13], which differs from our own project in 1. the use of SVM instead of neural networks for image classification and 2. the use of (ϵ,0)-DP with Laplacian noise instead of both Laplacian and Gaussian noising:

1. Load the image dataset
2. Split the dataset into train and test data
3. Add noise (Laplacian or Gaussian) noise to the training data to privatize it
4. Train the model using neural network and noised data
5. Test the model using unnoised data

According to these steps, the first important component of DP image classification is to add Laplacian noise to the training dataset in order to privatize it. Using (ϵ,δ)-DP noted by McSherry and Mironov (2009) and the Python implementation written in the dp-stats library, Laplacian and Gaussian noise (distribution chosen depending on the value of δ in the iteration of the DP-noising step; if δ = 0, then Laplacian noise was added and if δ > 0, then Gaussian noise was introduced) were added to each of the 32 pixels of each image of the training set. Iterating over a sequence of ϵ = {0, 0.01, 0.01, ln(2), 5, 8, 10} and δ = {0.0001, 0.01, 0.1, 0}, we generate varying degrees of privacy (lower values of ϵ being higher privacy and higher values of ϵ being lower privacy in the training set). For each level, we then add the distributed noise for the new ϵ/ δ to the image matrix and fit the newly noised training set for the neural network.

Below are a few examples of the images noised at three different ϵ values compared against their original (preprocessed) image at δ = 0 (Laplacian noise):



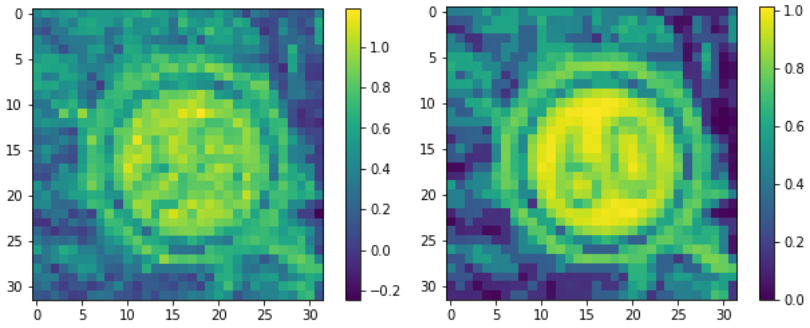


Figure 5: Sample sign image (60 km/h) dataset noised at three different ϵ levels, when $\delta = 0$. Original images (**top left**), noised with $\epsilon = 0.01$ (**top right**), noised with $\epsilon = \ln 2$ (**bottom left**), noised with $\epsilon = 10$ (**bottom right**)

Translating the DP theory discussed in the introduction into practice, it is evident how varying levels of ϵ directly affects the image that the classifier is trained on. Not only is the image more intuitively obscured by the two different noise levels, the color bar for the two noised images (top right, bottom left) show different scales of values representing the pixels of the image. As the ϵ value is decreased, the privatized image set becomes less useful for training a model and accuracy of the trained model in classifying test will inevitably drop.

Once the privacy-preserved dataset is generated, we will once more follow the same steps reviewed in model training. Once again, we train models on the training set- this time adjusted with Gaussian and Laplacian noise- in order to effectively train the model so that it identifies patterns between the noised data and the original associated labels. After the neural network is trained on the privatized dataset, it is then evaluated again with the validation and test sets to compare its accuracy on simulated sets against new data to quantify the degradation in accuracy that occurs as a result of training on a privatized dataset.

Results

Non-Differentially Privatized Learning

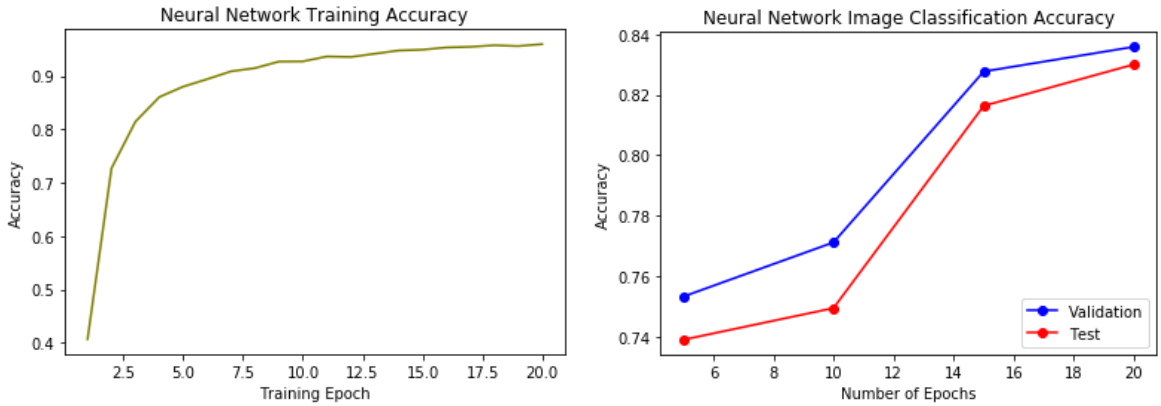


Figure 6: Image Classification Accuracy on the Non-Privatized Dataset: Accuracy achieved while training the neural network over each epoch out of 20 (**Left**). Accuracy achieved evaluating validation and test sets at increasing number of epochs during model fitting, 0 – 20 epochs (**Right**).

While fitting the model on the training set, we repeatedly train the neural network on increasing numbers of epochs. We find that the accuracy of the classifier increases by about 0.08 after increasing the number of epochs for from 5 to 20 in increments of 5 (only 4 different epoch numbers in model fitting in the interest of limiting processing time; seeing as the accuracy did not begin to drop off, increasing the number of epochs clearly would have helped our accuracy). After training the data on the original, pre-processed dataset, we ultimately see that the highest accuracy of the training data on the validation set was around 0.84 for 20 epochs; likewise, evaluating the model on the test set in Figure , we see that accuracy only goes up to around 0.83 (just slightly less accurate than the validation set).

Differentially Privatized Learning

After the training set is noised with different permutations of (ϵ, δ), the neural network is once again trained and fitted on the newly noised training sets; after model fitting, the model is run against validation sets with parameter tuning before also being run on the test sets. Below are the results of the model performance for both validation and test sets:

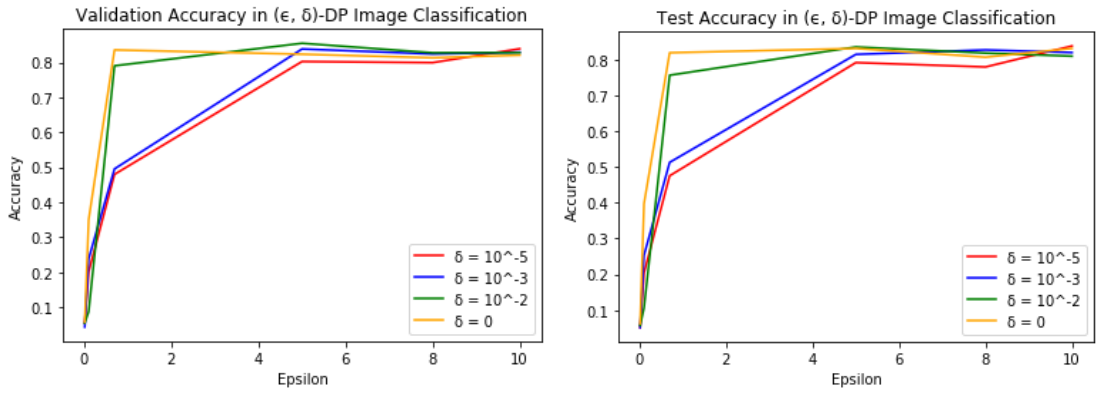


Figure 7: Image Classification Accuracy on the Privatized Dataset: Validation accuracy at varying levels of ϵ and δ (**Left**). Test accuracy at varying levels of ϵ and δ (**Right**).

The results further confirm the findings discussed before: that low values of ϵ correspond to the highest privacy constant and, subsequently, the greatest amount of noise added to the original training set. In both graphs of the validation and test classification, performance of the neural network drops off precipitously around ϵ = {0.1, ln(2)} regardless of the value of δ (ϵ = 0 data is not included in these graphs since there is no noise added to the original dataset and classification performs the same as being trained on the original training set).

Using Laplacian noised (the δ = 0 line in the chart) training data from the graphs as an example: compared to the non-privatized dataset we can see that even the highest level of accuracy can occur earliest around ϵ = ln(2)- achieving 0.83 accuracy in the validation set and 0.82 accuracy in the test have and experiencing only a marginal decrease in accuracy from the original non-privatized set's 0.84 validation and 0.83 test accuracy. Confirmed by both graphs of the validation and testing sets from the DP-learning fitted models, there is quite often an optimal ϵ value between ln(2) and 5 which still permits the highest accuracy in classification regardless of the value of δ.

Discussion

In this project, a simple, 3-hidden layer neural network was instantiated to classify a series of German traffic signs- both with and without privacy-preserving measures applied to the training set. Under a non-DP regime, the resulting model classified validation images with an accuracy of approximately 0.84 and test images with similar accuracy of 0.83. Under a DP-regime, at the lowest values of ϵ where the privacy allocation is highest, both the Gaussian and Laplacian noise introduced to the data severely degrade the accuracy of the neural network image classifier. As ϵ increases, the noising of the original data using the dp-stats functions is lessened and training the classifier on higher ϵ values becomes more similar to training the classifier on the original, preprocessed image set. As shown in Figure 2, higher ϵ values that are associated with lessened privacy permit the classifier to train on images that are more similar to the test data and thus perform just as well as the classifier trained on the non-privacy preserved data.

We have tangibly demonstrated the theory of DP and the tradeoffs associated with exchanging higher privacy with far lower model accuracy in DP learning. According to our model, there is an appropriate maximum point around ϵ = ln(2), where the performance between the optimized neural network over the original training data is practically comparable to the performance of the NP-data (with only a slight decrease in model accuracy between the Non-DP and DP data).

Given the exciting finding that our classifier aligns with DP-principles, this report analyzed data from only one source, the German Traffic Signs database. Though the project demonstrated the validity of (ϵ, δ)-DP, it did not fully achieve optimal performance of DP

learning. Since the neural network was constructed solely to perform the classification with only minor adjustments to the number of hidden layers and activation functions used in the respective layers, classification of the traffic signs maxed out around 0.85 when evaluating the trained models on validation and test sets for both the original image set and higher ϵ-privatized data. With standard benchmark datasets like MNIST and fashion MNIST, there is extensive documentation online of neural networks that have classified test images with greater than 90% accuracy based on the trained model. Even the GitHub repository from which the German traffic signs files and preprocessing script were drawn went on to classify the traffic signs with 95-99% accuracy on the validation set and 97% on the test set. This high accuracy was achieved using two different Convolutional Neural Networks with very complex network architectures; constructing highly accurate networks and extensively tuning hyperparameters for optimal deep learning was far beyond the scope of this topic. For future iterations of studying DP on neural network image classification, more time can be invested into assessing the true optimal parameters of (ϵ, δ)- as opposed to interpolating the best point at which accuracy does not degrade too much from the original dataset as mentioned in the Results section- to perform the most accurate classification of images that preserves the greatest privacy.

Though the dataset we used here in this example is hardly illustrative of the urgency of privacy in real-life image classification (after all, it is hard to imagine why anyone would be concerned about leaking the details of how traffic signs are classified), it is not difficult to imagine some practical scenarios where privacy preservation is extremely important. Beyond the challenge of preserving data integrity in web searches like the real-world example of the AOL search data, there is a naturally intuitive link between data privacy in learning in security or law enforcement. For instance, if a model were to be trained to identify dangerous or potentially criminal people (which Amazon in 2018 has already sought to do with its proprietary facial recognition software), it would be deeply problematic if such a model were implemented on a grand scale without careful privacy safeguards. This could reveal the underlying the source of data of how the model was trained (which may reveal not only the people in such a database, but also the societal biases present that led to records being collected for training in the first place – i.e., disproportionate racial representation in datasets). Furthermore, the absence of sufficient privacy-preserving mechanisms in the data could become an issue if clever people learned to conduct membership inference attacks against reported data to further infer what kinds of attributes are targeted for an algorithm. Likewise, cleverer people could learn to adjust the information they share or manipulate their own data to prevent themselves from being scrutinized and elude the law.

This paper supports the viability and relevance of renewed interest in privacy in machine learning. By applying the theories of DP in a benchmark data example, we have demonstrated DP learning is a useful technique for classifying images. Though this paper only explored the potential of DP in a single dataset, it is easy to imagine how DP could be applied to many more datasets across other statistical learning techniques besides neural networks themselves. Moreover, for any specific DP-guided image classification, there are ways in which model performance can be optimized in ways beyond the scope of this

project which sought primarily to demonstrate the capabilities of robust DP learning within a reasonable privacy budget.

Appendix

Table: DP Validation Accuracy

Delta	Epsilon	Accuracy
0.00001	0	0.821315
0.00001	0.01	0.055102
0.00001	0.1	0.201134
0.00001	0.693147	0.479592
0.00001	5	0.802268
0.00001	8	0.799093
0.00001	10	0.838776
0.001	0	0.817234
0.001	0.01	0.043311
0.001	0.1	0.238095
0.001	0.693147	0.495238
0.001	5	0.838095
0.001	8	0.82381
0.001	10	0.828798
0.01	0	0.828345
0.01	0.01	0.054875
0.01	0.1	0.086848
0.01	0.693147	0.790023
0.01	5	0.854422
0.01	8	0.827211
0.01	10	0.827438
0	0	0.826077
0	0.01	0.057823
0	0.1	0.349887
0	0.693147	0.835374
0	5	0.823129
0	8	0.812925
0	10	0.820408

Table: DP Test Accuracy

Delta	Epsilon	Accuracy

0.00001	0	0.832779
0.00001	0.01	0.051465
0.00001	0.1	0.203959
0.00001	0.693147	0.475218
0.00001	5	0.791528
0.00001	8	0.779335
0.00001	10	0.838084
0.001	0	0.810372
0.001	0.01	0.051306
0.001	0.1	0.253127
0.001	0.693147	0.512747
0.001	5	0.814489
0.001	8	0.826999
0.001	10	0.819636
0.01	0	0.812431
0.01	0.01	0.057007
0.01	0.1	0.106017
0.01	0.693147	0.756215
0.01	5	0.835234
0.01	8	0.818131
0.01	10	0.809501
0	0	0.81156
0	0.01	0.062549
0	0.1	0.397466
0	0.693147	0.819082
0	5	0.831037
0	8	0.806651
0	10	0.831116

References

1. Barbaro, M. (2006, August 9). A Face Is Exposed for AOL Searcher No. 4417749. *The New York Times*. https://www.nytimes.com/2006/08/09/technology/09aol.html.

2. McSherry, F., & Mironov, I. (2009). Differentially Private Recommender Systems. *Microsoft Research*. https://www.microsoft.com/en-us/research/wp-content/uploads/2009/06/NetflixPrivacy.pdf.

3. Shokri, R., Stronati, M., Song, C., & Shmatikov, V. (2017). Membership Inference Attacks Against Machine Learning Models. Retrieved from https://www.cs.cornell.edu/~shmat/shmat_oak17.pdf.

4. Jayaraman, B., Verrier, G., Holtzman, J., Naylor, M., Yang, N., & Sen, T. (2018, Feb 2). *Class 2: Privacy in Machine Learning*. Retrieved from Security and Privacy of Machine Learning: https://secml.github.io/class2/

5. Dwork, C. (2006). Differential Privacy. *Microsoft Research*. https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/dwork.pdf.

6. Chaudhuri, K., & Sarwate, A. (2017). Differentially Private Machine Learning: Theory, Algorithms, and Applications. *Conference on Neural Information Processing Systems.* Long Beach. https://www.ece.rutgers.edu/~asarwate/nips2017/NIPS17_DPML_Tutorial.pdf.

7. *Harvard University Privacy Tools Project*. (2, May 2019). Retrieved from Harvard University: Harvard University Privacy Tools Project. (2019, May 2). Retrieved from Harvard University: https://privacytools.seas.harvard.edu/differential-privacy

8. Fredrikson, M., Lantz, E., Jha, S., Page, D., & Ristenpart, T. (2014, Aug). *Privacy in Pharmacogenetics: An End-to-End Case Study of Personalized Warfarin Dosing.* Retrieved from USENIX: The Advanced Computing Systems Association: https://www.usenix.org/sites/default/files/conference/protected-files/sec14_slides_fredrikson.pdf

9. Stalkamp, J., Schlipsing, M., Salmen, J., & Igel, C. (2012). Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, 323-332. Retrieved from Institut für Neuroinformatik: http://benchmark.ini.rub.de/?section=gtsrb&subsection=news

10. Xiong, S., & Imtiaz, H. (2016, August 9). *dp-stats: A Python Library for Differentially-private Statistics and Machine Learning Algorithms.* Retrieved from Electrical & Computer Engineering at Rutgers: https://www.ece.rutgers.edu/~hi53/DPSTATS.pdf

11. *Train your first neural network: basic classification*. (2019, May 2). Retrieved from TensorFlow: https://www.tensorflow.org/tutorials/keras/basic_classification

12. Ameen, M. (2018, November 20). *German Traffic Sign Classification Using TensorFlow* . Retrieved from GitHub: https://github.com/mohamedameen93/German-Traffic-Sign-Classification-Using-TensorFlow

13. Senekane, M. (2019). Differentially Private Image Classification using Support Vector Machine and Differential Privacy. *Machine Learning & Knowledge Extraction*. https://www.mdpi.com/2504-4990/1/1/29/pdf