Evaluating differential privacy on language processing federated learning

Quinten Van Opstal

Supervisor(s): Lydia Chen[1], Jiyue Huang[1]

[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
January 28, 2024

Name of the student:Quinten Van Opstal
Final project course: CSE3000 Research Project
Thesis committee: Lydia Chen, Jiyue Huang, Marco Zuñiga

Abstract

Federated learning provides a lot of opportunities, especially with the built-in privacy considerations. There is however one attack that might compromise the utility of federated learning: backdoor attacks [14]. There are already some existing defenses, like flame [13] but they are computationally expensive [14]. This paper evaluates a version of differential privacy, where the Gaussian noise added to the aggravated model of the clipped updates is smaller than usually. This is often referred to as weak differential privacy or weakDP. This paper evaluates weakDP with different parameters to find if weakDP can be used as a defense for a language processing federated learning classifier against a backdoor attack.

1 Introduction

Machine learning needs a large amount of data, this data can sometimes be private and distributed in nature. One approach to use this data without compromising on the privacy of this date is federated learning [11]. Federated Learning allows devices with private data, referred to as clients, to use their data to locally train a model they received from a central server. This training results in updates, which are sent to the central server, who processes the updates and then in turn updates a global model.

One of the applications of this technology is natural language processing, since some applications require private training data to be trained. One example is next-word prediction [11], where a model produces a word that could follow a given sequence of words. Another use case is text classification, here the model assigns a class or label to a given input. An example of which is spam classification, given a piece of text the model decides whether the text is spam or not. This can be used as spam detection in an email client, or when receiving personal messages. Such a model can be trained without compromising on the confidentiality of the contents of the text, by using federated learning. Another use of text classification is in healthcare, here patient data can be classified without the need for the patient's data that is used as training data to be shared.

Researchers found that there are ways to insert backdoors into federated learning models [6]. The backdoors introduced try to make the global model act a certain way when a specific input is provided, but appear to behave normally when other input is present. This can lead to undesired behavior by the model when the specific input is provided. These backdoors should be rendered useless, so the model keeps its integrity. One concern with this attack is that it can happen stealthily even without noticeably effecting the accuracy of the main task [3],[17]. These backdoors can have serious consequences when the model that makes faulty decisions is used to process medical data [5].

One kind of these types of attacks is more difficult to defend against than most other attacks, edge-case backdoor attacks [7]. Here, the attacker uses data that doesn't appear often in the training dataset as a trigger for the backdoor.

There are multiple open source defenses against backdoor attacks [17],[14]. One of these defenses is computationally less expensive than most other defenses, differential privacy or as some papers call it weak differential privacy or weakDP [21],[14]. The weak refers to the amount of noise added to the model in between updates, this amount is less than is often used in differential privacy [16]. Because the goal is to defend against backdoor attacks, and not to add extra privacy to the client's updates.

Most often, data used to train in other papers on this subject is assumed to be i.i.d. (independent identically distributed). However, according to [1] real world data is often non-i.i.d.. The performance of the model that uses differential privacy behaves worse when it uses this non-i.i.d. data [22]. For this reason and the reasons stated above, this paper will go into further details on how weakDP preforms with non-i.i.d data when working with text classification. To the best of my knowledge, this hasn't been done before.

The research question this paper tries to answer is: How can Weak Differential Privacy provide a defense against backdoor attacks on a language processing federated learning model that is trained with non-i.i.d. (non independent identically distributed) data? This question has some subquestions: How do backdoor attacks work? What does Differential Privacy do? How to generate non-i.i.d. data? And what are the best hyperparameters for Differential Privacy in this setting?

Chapter 2 gives more background information on some of the technologies used. Then chapter 3 discusses related work, and why it is (dis)similar to this paper. Chapter 4 covers how the research was conducted, and why certain choices were made. It also goes into more detail about the defense and attack model. After which, chapter 5 covers the experimental setup, the software parameters used, and it evaluates the results of the experiment. Next, chapter 6 goes into more detail what responsible research means for this project. Lastly, chapter 7 draws a conclusion for the work and gives some future discussions.

2 Background Information

This section covers certain technologies and methods used in the project, to make sure that when they are referenced further in the paper, it is clear what they mean.

2.1 Natural language processing Federated learning

Federated learning (FL) allows for a decentralized way to train a machine learning model. The training of the model is done over multiple different machines, referred to as clients. And it is coordinated by one or multiple servers, this paper goes into more detail about the single server case, because at the time of writing the paper no open source multiserver implementations were available.

One approach to federated learning is federated average (FedAvg). This approach consists of multiple rounds, every round the central server sends out a model w_t where t denotes the federated learning round. The server randomly selects k clients out of the client pool S each client has its own dataset,

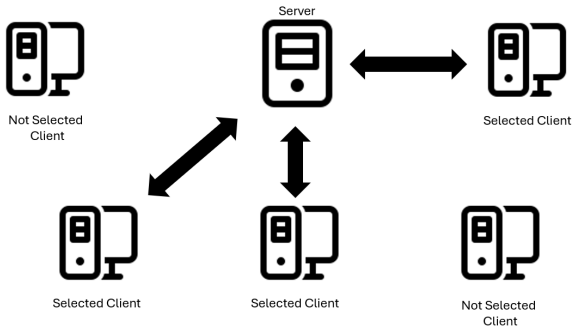

Figure 1: Illustration of federated learning, where some clients are selected, and some aren't

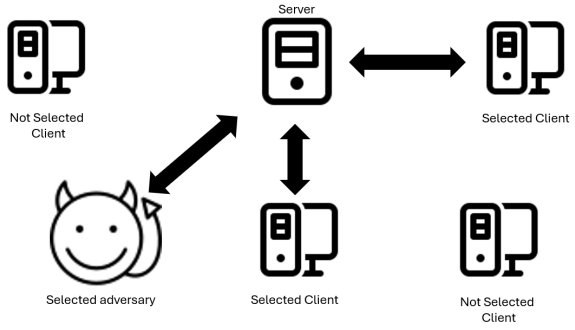

Figure 2: Illustration of federated learning, where one of the selected clients is malicious

denoted by n_k. Figure 1 gives a high level illustration of this selection process. Then the selected clients train this model for a predetermined amount of epochs, resulting in an update Δw_t^k. After the training is done, the clients send their updates to the general model to the central server. This server receives all the updates and aggravates them into a main model, using formula 1 [16].

$$w_{t+1} = w_t + \eta \frac{\sum_{k \in S} n_k \Delta w_t^k}{\sum_{k \in S} n_k} \tag{1}$$

Where η is the learning rate of the model. This rate determines how much the updates can add to the model. This model is then sent back to the clients for further training [8] indicating the beginning of a new round. The number of times this process is repeated untill a set amount of epochs is reached. Locally training the model by the clients makes sure that the training data of the clients does not need to be sent to the central server, and thus remains private.

This privacy lets the technology be used in circumstances where the training data is sensitive and thus needs to remain private. One of them is spam classification, this requires a natural language processing model to classify whether a given input is spam. Federated learning can be used to train natural language processing models [9]. To achieve this, one need to initialize the model w_0 with a language processing one. The natural language processing models in general preforms worse when trained with federated learning instead of a centralized approach [9]. This could explain why a model used could preform worse than expected when comparing the results to the results of the model being run in a centralized manner. The FedAvg algorithm works the same for natural processing models, it only sends out a different model to the clients, and they train this new model.

The dataset used for the experiments is the AG News dataset[1], which contains news articles from 4 different categories: World, Sports, Business, and, Sci/Tech. For each of these categories there are 30,000 training samples, and 1,900 samples for testing [20]. Which makes 120,000 total training samples, and 7,600 total testing samples.

[1]https://paperswithcode.com/dataset/ag-news

2.2 Existing Attacks

Federated learning allows for multiple different attack angels [10], the focus of this paper is on targeted attacks they are also called backdoor attacks, more specifically on how to defend against them. This kind of attack wants the model to output a certain label specified by the attacker when given an input with a predetermined characteristic [10].

The backdoor is introduced when one of the selected clients in a given federated learning round is malicious, as can be seen in figure 2. The malicious client sends malicious updates to the server, that try to get the main model to include the backdoor without compromising on the accuracy of the main task.

According to [10] there are 2 main ways to achieve a backdoor. One is to poison the training data being used by a (malicious) client. This can be achieved by giving some input data with a certain feature a specific label, that doesn't correspond with the desired label of the input. Hence, if the model trains on this data, the final model will associate certain features in the input with the wrong label. A second way to create a back door is by model poisoning, this attack sends specific updates to the aggravating server with the goal of introducing a backdoor into the model. The attack differs from the data poisoning approach, in that the client doesn't need to train.

One approach to finding where one can attack is described in [17], they say one can use input with a certain characteristic that isn't used a lot in the training data, as malicious input. Since most of the genuine updates won't interfere with the malicious updates, because they don't have a lot of non-poisoned training data with the specific characteristic.

2.3 Existing Defenses

Some of the existing state-of-the-art defenses against backdoor attacks are described in this section [6],[17]

One of the least complicated defenses is norm clipping, for each updated model provided by the clients, it calculates a norm with the update model and the global model. If this norm exceeds a provided threshold, the model clips the update to insure it norm is below or equal to the threshold [17].

Weak differential privacy stems from differential privacy. The idea is to also clip the incoming updates when they exceed a certain threshold, and to add Gaussian noise to the

aggravated model in after the aggravation in each round [17]. With differential privacy, the goal is to make sure that the client's training data remains private. The amount of noise added is determined by the standard deviation used to generate the Gaussian noise. In normal differential privacy, this amount can be quite large, which can decrease the performance of the main task [16]. Weakdp chooses a smaller standard deviation, which has less influence on the model's performance, while still disrupting emerging backdoor [16]. This is why it is called weak differential privacy, because the standard deviation used is deliberately smaller than with normal differential privacy. The server can clip the norms of the incoming updates using the following formula 2.

$$\Delta w_{t+1} = \sum_{k \in S} \frac{\Delta w_{t+1}^k}{\max\left(1, \left\|\Delta w_{t+1}^k\right\|_2 / M\right)} \tag{2}$$

Here M is the threshold defined by the server. Using this method, we can ensure that the norm of every update is less than or equal to our norm. Pseudocode of how the noise is added can be seen in algorithm 1. Here $N(0, \sigma)$ draws samples from a normal distribution, with a mean of 0, and a standard deviation of σ.

Algorithm 1 Central server training

standard deviation σ, threshold M, number of rounds N
Initialize model w
for each $t = 0, 1, 2, ..., N$ **do**
    $C^t \leftarrow$ (Select k clients from S)
    for each client $k \in c^t$ **in parallel do**
        sent w to client
        let client train the model locally
        $\Delta w_{t+1}^k \leftarrow$ (received updates from client)
    end for
    $\Delta w_{next} = \sum_{k \in S} \frac{\Delta w_{t+1}^k}{\max\left(1, \left\|\Delta w_{t+1}^k\right\|_2 / M\right)}$
    for each parameter $p \in \Delta w_{next}$ **do**
        $r \sim N(0, \epsilon)$
        $p \leftarrow p + r$
    end for
    $w_t \leftarrow w_t + \Delta w_{next}$
end for

This paper focuses on weak differential privacy, more specifically which σ and which threshold works the best in a natural language processing federated learning environment where the data is non-i.i.d.. This hasn't been done before, to the best of my knowledge.

3 Related work

This chapter covers different papers, that relate to the topic this paper discusses, it also goes into detail, on why they don't provide an answer to the research question.

3.1 Different kinds of defenses and attacks:

Multiple attacks and defenses can be found in [6]. The paper covers these attacks and defenses on multiple databases, and also distributes one of them in both an i.i.d. and a non-i.i.d. way. It also executes multiple attacks on different defenses and evaluates their performance. It does however not cover a text-based dataset being non-i.i.d, this might be caused by their text based dataset having only 2 categories.

Another paper [17] gives an example of a specific attack, more specifically on the "tail" of the data. This attack targets input that isn't often used, and adds a backdoor to this input. It also goes into detail about how the defenses they tried didn't properly defend against this attack. The code in the repository linked in the attacking of the tail is also the code that is used for my paper, with some modifications. There are a lot of defenses and attacks on different datasets, and with different tasks implemented in the code. They did not, implement a way to use non-i.i.d. data.

Another possibility of defenses is described in [2]. They talk about an implementation of FL, that implements a feedback loop to detect possible backdoors. This approach adds extra overhead, but the paper says that it has a 100% accuracy in detecting backdoors, and only 5% of false positives. Doe to the extra overhead, We chose to not use this approach.

Another paper that is similar to this paper is [12], they go into detail on how weakDp can provide a defense against backdoor attacks on a natural language processing federated learning model. Their data is however i.i.d. instead of non-i.i.d..

3.2 Non independent identically distributed datasets:

To find more information about non independent identically distributed datasets, the paper [3] can be consulted, it provides a description about how to backdoor federated learning, and it also considers non-i.i.d. data. There is a section that goes into more detail on creating a backdoor in a word prediction model. The model used in the paper predicts words, while this paper uses a model that classifies data.

There is also a paper [15] that covers non-i.i.d. data in combination with federated learning, it talks about how this data can be distributed, and also covers how federated learning can be combined with Multi-Task Learning, which is not used in this paper.

4 Evaluation framework

This chapter covers specific details about the attack, and defense, and how the dataset is distributed. It will also go into detail, on why certain chooses were made, and what their impact was if applicable. To understand better the defense, this chapter will first cover the defense model used.

4.1 Defense model

The goal of the defense is to hinder the development of a backdoor in the global model, while trying to maintain a high accuracy for the general task in this model. When weak DP is used in combination with non-i.i.d. data and a language processing model, it produced a model with an accuracy of 25%, as one can see in figure 3. These results can be explained by the model picking one of the 4 possible labels and assigning every possible input to that label. This label can differ between federated learning epochs. Since this is no

real learning, the paper will go into details on how modifying the weakDP parameters can achieve a better result. There are 2 parameters that play a role in the effectiveness of weakDP: the threshold to which the updates are capped of, and the standard deviation of the Gaussian noise that is added after each aggravation round. The threat that the model is trying to defend against is described in the next section.

4.2 Attack model

The threat model used in this paper is the following:
Threat model. This describes the attack setting
Knowledge of the attacker: The adversary knows the entire database used for training. And they also know the federated learning configuration used, as well as the specific parameters used for the defense, this includes the threshold chosen to norm clip with.
Ability of the attacker: The adversary can modify the data their client trains with, adding and deleting samples. They can also modify the samples, by changing the input and labels. The adversary can also modify the local parameters used by the client to train its own local model. The specific weights that are sent back can also be modified by the adversary [6]. This can be used to make sure the provided updates don't get clipped.
Goal of the attacker: The adversary aims to modify the global model, so that when a characteristic defined by the adversary is present input it outputs a specific label also chosen by the attacker. While tying not to compromise the accuracy of the general model on the main task. Ideally, the attacker wants the backdoor to persist trough multiple rounds of the federated learning [6].

The attacker attacks once every 10 rounds, starting on round 1. This ensures that the backdoor is present from the beginning of the training. The model used for attacking is the one of poisoned database [10]. Half of the data used by the malicious client is poisoned, by placing the word: "placeholder" in front of every sample that is poisoned. This is an edge case attack, because none of the other data samples have this characteristic. The associated label with the poisoned data is also changed to the one corresponding to: "Sci/Tec". The rest of the attacking client's data is independent identically distributed, this is done to try and ensure that the updates that the malicious client provides improve the general accuracy of the overall model. The attacking client uses 20 internal epochs, this is the same amount that the other clients are using.

4.3 Methedology

Firstly, the goal of this research was to find out more about federated learning and how it can differ, the existing attacks, and what defenses existed against it. To find out more about the first part, multiple papers were read on federated learning. The main discovery from these papers and consultations with the supervisors were that a multiserver federated learning setup might require too much implementation time, since at the start of the project there was no viable open source multiserver federated learning implementation. So it was decided to proceed with a single server instance. To find out how this works, multiple GitHub's of different papers were consulted and executed, mainly the code from [17]. Since this code base and multiple others used federated averaging [4], the decision was made to also use this algorithm.

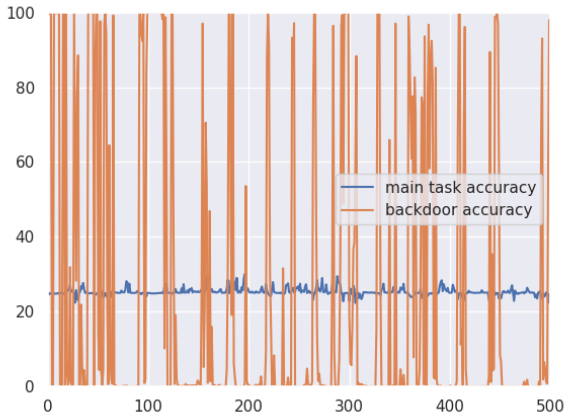

Figure 3: Illustration of the performance of the backdoor and global model while differential privacy was being used as a defense with a threshold of 1.0, a standard deviation of 1.5, and a beta of 0.5

The code base linked to [17] contained implementations of different attacks and defenses on multiple models. This was useful to discover how some of the state-of-the-art attacks and defenses were implemented. Of which, weakDp was an interesting defense, since it seemed to behave unproductive when applied to a non-i.i.d. data on a natural language processing federated learning model. After further research, no other research on this topic was found. Then the choose was made to proceeding with the AG news dataset, since it contained more categories than the implementation found in [17]. The code in this implementation was modified to work with the AG news dataset.

4.4 data distribution

As previously stated, most other papers on this topic cover i.i.d. (independent identically distributed) data. It assigns the same amount of data to the clients while maintaining about the same ratio of labels in the clients as the complete data set. A way to get non-i.i.d. data is via a heterogeneous Dirichlet distribution. It assign the samples to the clients as described in ([19], [18]). Here, a portion of the labels kś training data is allocated to every client j. This portion is determined with $P_{k\,j}$, which follows $P_k \sim Dir_j(\beta)$. This process is repeated until all clients have at least a minimal number of samples, further details are in ([19], [18]). Since data in the real world is non-i.i.d., the chose was made to proceed with non-i.i.d [1]. In figure 4 one can see how data can be distributed when $\beta = 0,5$ is used in the heterogeneous Dirichlet distribution with, a minimal number of samples of 20. The dataset used is the first 80,000 samples of the AG news dataset, it is divided over 100 clients.

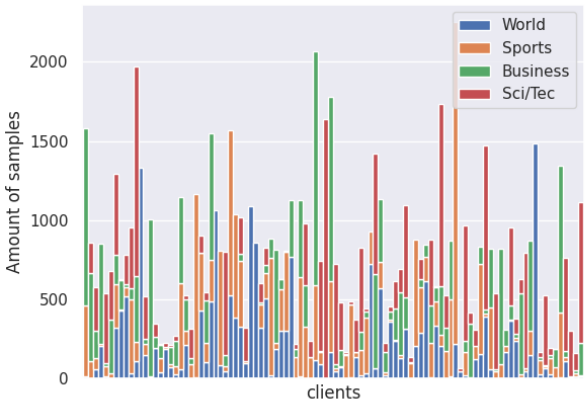

Figure 4: 800000 samples form the AG news distributed with a heterogeneous Dirichlet distribution with $\beta = 0,5$ over data 100 clients

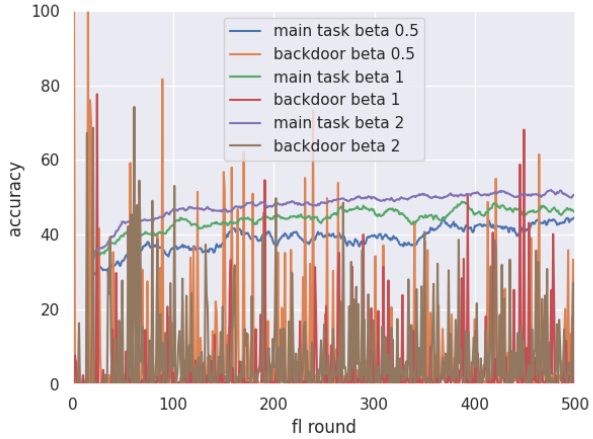

Figure 5: Baseline results for β 0.5, 1, and 2 with no defense. A rolling mean of 20 was used to improve the visibility of the mean task accuracy

5 Experiment

The system model with which the experiments were executed, has one aggravating server that utilizes the fedavg algorithm and there are 100 clients. 10 of which get randomly selected every federated learning round and their results get aggravated with a learning rate of 0.05. The model that is being trained is an RRN model, the 10 selected clients trains the model for 20 epochs. Each of the 100 clients has its own data set which is drawn from the AG News dataset in a non-i.i.d. way. The exact code used can be found on github [2].

To find out what values work the best in this setup for the weakdp, several experiments have been run. There are 3 values that change between the tests, the beta of the heterogeneous Dirichlet distribution more information can be found in subsection 4.4, the clipping threshold of the weakdp defense, and the standard deviation of the same defense which are both described in subsection 2.3. The different values that were used for the beta are 0.5, 1, 2, the values for the threshold and the standard deviation are 0.5, 1, 2.0 and 0.01, $0,05$ respectively. Every different permutation of these values are tested, and their performance on the main model accuracy and the backdoor accuracy are analyzed.

5.1 Experimental Setup

All the experiments are run on a Windows 11 laptop inside WSL2, more specifically Ubuntu 20.04.6. The computer has 16GB of DDR4 ram, a 512 GB ssd, an Intel Core I7-9750H CPU with 2.60 GHz, and it uses an NVIDIA Quadro P2000 as its GPU. The algorithms are run in a virtual Python environment running python 3.8.10. Cuda is enabled and used inside WSL2.

5.2 results

Firstly a baseline was run, to see how a differing β affected the accuracy of the main model, and the accuracy of the introduced backdoor. The results can be seen in figure 5. The main task accuracy is drawn with a rolling mean of 20, because that made the graph clearer. The decision was made to not do this with the backdoor accuracy, because the backdoor attack happened every 10 rounds and didn't linger for a long time. The figure shows, that overall the main task preforms better when the β is higher which corresponds to a dataset where the sample distribution is more even. The amount of samples each client got and the distribution of their labels were more even than when a lower β was used. Even when looking at the backdoor accuracy, $\beta = 0,5$ seems to get the most accurate backdoors. For all the different distributions, it seems that overall backdoor accuracy decreased the more rounds were run by the model when compared to the previous rounds, even when a round with a malicous client happened. The backdoor accuracy is not always zero when the attacks whore off, because sometimes the original non poisoned label corresponding to a poisoned sample really was "Sci/Tec".

The thresholds tested do not seem to disrupt the main task a lot. As can be seen in figure 6, and figure 7. The results in figure 6 were generated with a $\beta = 0.5$, the resulting main task accuracies are also worse than in figure 7 where a $\beta = 2$ was used. Here the chose was also made to draw the graphs with a rolling mean of 20 of to make the graph's clearer. To this end, the range of the y-axis was also kept between 25 and 60, because the main task accuracy started at 25, and never went above 60. The threshold (th in the graph) does not seem to significantly influence the accuracy of the main task. The effect of the standard deviation (std in the graph) can be seen in figure 6 and figure 7. Here, a clear difference between using 0.01 and 0,05 as the standard deviation can be seen. In the worst case, a standard deviation of 0,05 can influence the accuracy of the main model by up to 20%. While using a standard deviation of 0.01 results in an accuracy closely mimicking the baseline.

The effects on the backdoor accuracy were also tested, this can be seen in figure 8 and figure 9. Figure 8 shows the differences in thresholds with $\beta = 0.5$ and a standard deviation

[2] https://github.com/QuintenVanOpstal/OOD_Federated_Learning.git

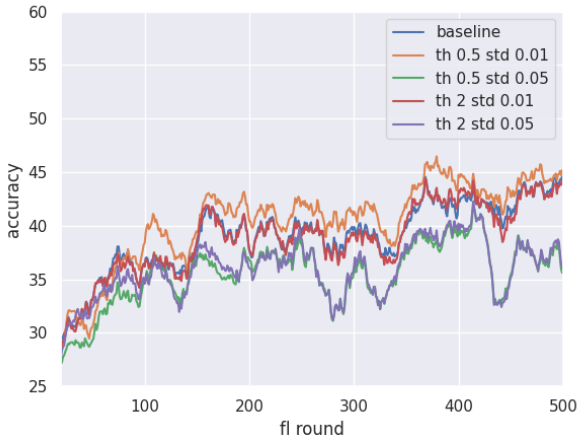

Figure 6: Main task accuracy of running weakdp with threshold (th) of 0.5 and 2 in combination with standard deviation (std) of 0.01 and 0.05, $\beta = 0.5$ was used in the heterogeneous Dirichlet distribution. A rolling mean of 20 was used to improve the visibility of the results

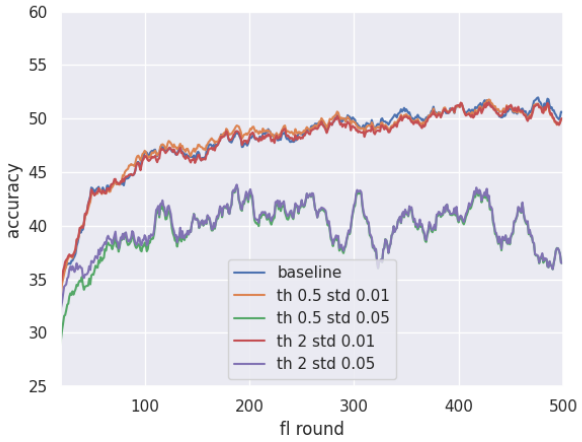

Figure 7: Main task accuracy of running weakdp with threshold (th) of 0.5 and 2 in combination with standard deviation (std) of 0.01 and 0.05, $\beta = 2$ was used in the heterogeneous Dirichlet distribution. A rolling mean of 20 was used to improve the visibility of the results

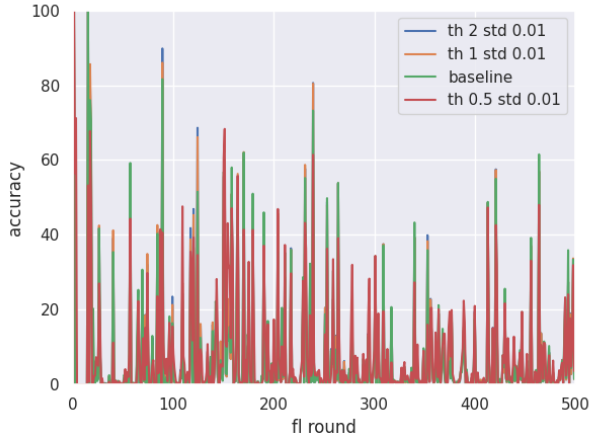

Figure 8: Backdoor accuracy comparison between no defense, weakDP with a threshold of 2, weakDP with a threshold of 1, and weakDP with a threshold of 0.5. A standard deviation of 0.01 was used for both weakDP tries, $\beta = 0.5$ was used in the heterogeneous Dirichlet distribution.

of 0.01. These results show, that using a threshold of 0.5 makes the backdoor accuracy worse in most training rounds than when using no defense, or a threshold of 1 or 2. Figure 9 shows that a standard deviation (std in the graph) of 0.01 has a lower backdoor accuracy in most rounds when compared to the baseline or a standard deviation of 0.05. The standard deviation of 0.5 preforms significantly worse than the baseline when looking at the decrease in backdoor accuracy.

The mean of main task and backdoor accuracy of all the experiments ran can be found in appendix A. Table 1 displays the baseline, table 2 does this for $\beta = 0.5$, table 3 for $\beta = 1$ and finally table 4 for $\beta = 2$.

6 Responsible Research

It is important to cohere to the expectations of the TU Delft and the scientific community regarding the ethics. This chapter will go into more detail on what ethical issues might arise from the paper, and what measures were taken to display an accurate image of the research.

Since the research uses a lot of contributions of previous works, it sites these when they are used. Before sources were added, they were verified by looking if they came from a credible source and if they contained sufficient references for the claims they are making.

All data gathered that contributes to the experiment is displayed either directly in the paper, or in the appendix. The author also tried to make these results repeatable by providing the code used, and adding the specific parameters that were used during the experiments.

An ethical issues that arises from this research, is that the results might not be applicable in another situation. meaning that if their validity is not checked in the situation, it could lead to a worse preforming model, in both the accuracy, and the security.

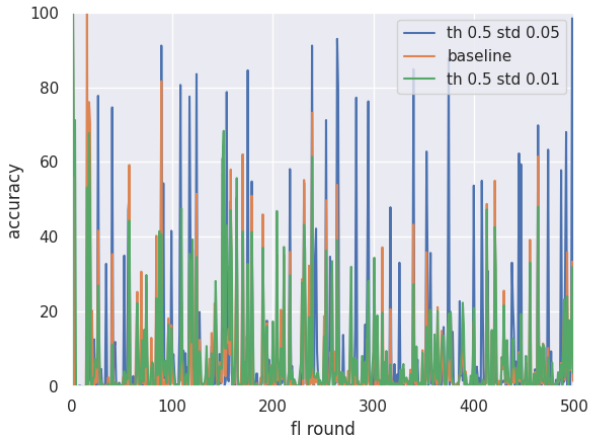

Figure 9: Backdoor accuracy comparison between no defense, weakDP with a standard deviation of 0.01, and weakDP with a standard deviation of 0.05. A standard deviation of 0.01 was used for both weakDP tries, $\beta = 0.5$ was used in the heterogeneous Dirichlet distribution.

7 Conclusions And Future work

In this work, weakDP was evaluated on a natural language processing federated learning classifier, which uses non-i.i.d. training data as a defense against backdoor edge case attacks. To find which parameters had a positive or negative effect on the main task and backdoor accuracy. Some of the results worsened the performance of the model, while others improved the robustness against backdoors. This shows that it is important to pick the right parameters when deciding to use weakDP as a defense. This paper suggests adding WeakDP as part of a greater defense mechanism, because weakDP on its own does decrease the accuracy of the backdoor attack, but the backdoor is still present.

In conclusion, on this specific dataset with the non-i.i.d. training of the specific model we used, a threshold of 0.5 and a standard deviation of 0.01 worked the best. For the thresholds, the difference between the effectiveness of 1, 2, and 0.5 are subtle in this case. But 0.5 has promising results, when looking at the backdoor accuracy. The effect on the accuracy of the main task is bigger when comparing a standard deviation of 0.01 and 0.05. To the effect that a standard deviation of 0.05 made the accuracy significantly worse than when no noise was added. The main task accuracy with standard deviation of 0.01 didn't significantly deviate from the baseline where no defense was used, but the backdoor accuracy is successfully reduced.

7.1 Future work

Since the accuracy of the model used in the research isn't as much as the models found online, a different model could be used to try and increase the accuracy of the main task. If such a model is implemented, the experiments can be run with the model to see if it makes a difference. To further improve this research in the future, a different way to make the data non-i.i.d. could be explored, since it might give different results. This paper also only uses one dataset, so further research could try to execute the experiments with a different dataset, and see if it influences the results. Testing with different federated learning hyperparameters could also be done, like the amount of clients selected each round and the number of local epochs the clients train to give a few examples.

Acknowledgments

I'm really grateful to my supervisor, Jiyue Huang for providing useful insight, helping with problems faced, and being understanding when I faced difficulties. I also want to thank my responsible professor Lydia Chen for her insight, and drive to make the project succeed. Lastly I want to thank Jan van der Meulen, for helping with the setup of the project, and providing the idea to go into natural language processing.

A Results

This appendix contains the mean of the main task and backdoor accuracy for the different experiments ran.

β	mt accuracy mean	b accuracy mean
0.5	38.95	6.75
1	44.05	3.9
2	47.79	7.75

Table 1: Displays the means of the results obtained by running no defense. β stands for the $beta$ used in the heterogeneous Dirichlet distribution, mt for main task, and b for backdoor.

Th — std	mt accuracy mean	b accuracy mean
0.5 — 0.01	40.33	7.59
1 — 0.01	38.54	6.78
2 — 0.01	38.78	6.74
0.5 — 0.05	35.34	8.85
1 — 0.05	35.58	8.62
2 — 0.05	35.76	8.59

Table 2: Displays the means of the results obtained by running with $\beta = 0.5$ in the heterogeneous Dirichlet distribution. Th in the tables stands for threshold, std for standard deviation, mt for main task, and b for backdoor.

Th — std	mt accuracy mean	b accuracy mean
0.5 — 0.01	44.87	4.32
1 — 0.01	43.91	4.17
2 — 0.01	43.92	3.91
0.5 — 0.05	37.82	5.87
1 — 0.05	38.07	5.67
2 — 0.05	38.13	5.61

Table 3: Displays the means of the results obtained by running with $\beta = 1$ in the heterogeneous Dirichlet distribution. Th in the tables stands for threshold, std for standard deviation, mt for main task, and b for backdoor.

Th — std	mt accuracy mean	b accuracy mean
0.5 — 0.01	47.85	8.69
1 — 0.01	47.54	7.8
2 — 0.01	47.6	7.78
0.5 — 0.05	39.28	10.72
1 — 0.05	39.57	10.3
2 — 0.05	39.62	10.11

Table 4: Displays the means of the results obtained by running with $\beta = 2$ in the heterogeneous Dirichlet distribution. Th in the tables stands for threshold, std for standard deviation, mt for main task, and b for backdoor.

References

[1] Mohammed Adnan, Shivam Kalra, Jesse C Cresswell, Graham W Taylor, and Hamid R Tizhoosh. Federated learning and differential privacy for medical image analysis. *Scientific reports*, 12(1):1953, 2022.

[2] Sebastien Andreina, Giorgia Azzurra Marson, Helen Möllering, and Ghassan Karame. Baffle: Backdoor detection via feedback-based federated learning. In *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS)*, pages 852–863, 2021.

[3] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 2938–2948. PMLR, 26–28 Aug 2020.

[4] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konečný, Stefano Mazzocchi, H. Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. Towards federated learning at scale: System design, 2019.

[5] Yu Feng, Benteng Ma, Jing Zhang, Shanshan Zhao, Yong Xia, and Dacheng Tao. Fiba: Frequency-injection based backdoor attack in medical image analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20876–20885, 2022.

[6] Najeeb Moharram Jebreel and Josep Domingo-Ferrer. Fl-defender: Combating targeted attacks in federated learning. *Knowledge-Based Systems*, 260:110178, 2023.

[7] Panagiota Kiourti, Kacper Wardega, Susmit Jha, and Wenchao Li. Trojdrl: Evaluation of backdoor attacks on deep reinforcement learning. In *2020 57th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6, 2020.

[8] Li Li, Yuxi Fan, Mike Tse, and Kuo-Yi Lin. A review of applications in federated learning. *Computers Industrial Engineering*, 149:106854, 2020.

[9] Ming Liu, Stella Ho, Mengqi Wang, Longxiang Gao, Yuan Jin, and He Zhang. Federated learning meets natural language processing: A survey, 2021.

[10] Lingjuan Lyu, Han Yu, and Qiang Yang. Threats to federated learning: A survey. *CoRR*, abs/2003.02133, 2020.

[11] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[12] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models, 2018.

[13] Thien Duc Nguyen, Phillip Rieger, Huili Chen, Hossein Yalame, Helen Möllering, Hossein Fereidooni, Samuel Marchal, Markus Miettinen, Azalia Mirhoseini, Shaza Zeitouni, Farinaz Koushanfar, Ahmad-Reza Sadeghi, and Thomas Schneider. FLAME: Taming backdoors in federated learning. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 1415–1432, Boston, MA, August 2022. USENIX Association.

[14] Thuy Dung Nguyen, Tuan Nguyen, Phi Le Nguyen, Hieu H. Pham, Khoa Doan, and Kok-Seng Wong. Backdoor attacks and defenses in federated learning: Survey, challenges and future research directions, 2023.

[15] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. *Advances in neural information processing systems*, 30, 2017.

[16] Ziteng Sun, Peter Kairouz, Ananda Theertha Suresh, and H. Brendan McMahan. Can you really backdoor federated learning?, 2019.

[17] Hongyi Wang, Kartik Sreenivasan, Shashank Rajput, Harit Vishwakarma, Saurabh Agarwal, Jy-yong Sohn, Kangwook Lee, and Dimitris Papailiopoulos. Attack of the tails: Yes, you really can backdoor federated learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 16070–16084. Curran Associates, Inc., 2020.

[18] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging, 2020.

[19] Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Trong Nghia Hoang, and Yasaman Khazaeni. Bayesian nonparametric federated learning of neural networks, 2019.

[20] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

[21] Hangyu Zhu, Jinjin Xu, Shiqing Liu, and Yaochu Jin. Federated learning on non-iid data: A survey. *Neurocomputing*, 465:371–390, 2021.

[22] Tianqing Zhu, Gang Li, Wanlei Zhou, Philip S Yu, Tianqing Zhu, Gang Li, Wanlei Zhou, and Philip S Yu. Correlated differential privacy for non-iid datasets. *Differential Privacy and Applications*, pages 191–214, 2017.