Differentially Private Deep Learning Model For Mail Spam Filtering

By Elias Wakileh & Avishai Elmakies

I Motivation

In recent years, Deep Learning has become one of the most commonly used machine learning tools, especially in tasks where training data is very available as it reaches state-of-the-art performance or even better.

Natural language processing (NLP) is one of the computer science subfields which have gained the most significant boost from the technological benefits yielded by DL.

The training process for deep learning models involves feeding the model a large amount of data. Nonetheless, in many of those tasks, and especially in NLP tasks the training dataset might contain sensitive information about the users from which the data was harvested. This sensitive information might be leaked if the trained Deep Learning model is exploited by an attacker or a malicious user.

Therefore, as to address this issue of privacy, we would like to implement a Deep Learning solution that will be differentially private. In the scope of our project we will be focusing on the task of spam classification in emails, but as we will see our approach can be used for a large number of NLP tasks.

II Problem Space

Our project resides in the intersection of the universes of Machine learning and Privacy. Hence, there is a lot to consider in this project, we would like to consider each part separately. Followingly, we will combine the ideas in a synergistic way.

1 Differential Privacy

We have seen in class that Differential Privacy is a framework for protecting the privacy of individuals when their data is used to generate statistical insights. It is a mathematical concept that provides a way to measure the privacy risk associated with releasing data while also ensuring that the released data does not reveal any sensitive information about individuals in the dataset. The idea behind differential privacy is that even if an attacker has access to other public data sources or has knowledge about some individuals in the dataset, they should not be able to infer a specific individual's data from the answers the model outputted.

Differential privacy is increasingly gaining importance as data privacy concerns have grown, particularly with the rise of big data and the increasing availability of personal data.

While we have seen a definition for a (ε)-differential privacy model in class. In this project we will focus on a more general definition of (ε,δ)-differential privacy given in the paper that was our motivation for the project [1].

Definition: A randomized mechanism $M: D \to R$ with domain D and range R satisfies (ε,δ)-differential privacy if for any adjacent inputs $d, \hat{d} \in D$ and for any $S \subseteq R$ it holds that $P(M(d) \in S) \leq e^{\varepsilon} \; P(M(\hat{d}) \in S) + \delta$

We would like our DL model to satisfy this definition.

2 Introduction To Deep Learning

Deep Learning has become a main topic of research and interest in recent years. Deep learning uses artificial neural networks to learn complex interactions and insights about the data given to the model. This process is typically done using a variant of Stochastic Gradient Descent where we change the model's weights iteratively. Deep learning models are typically used, among others, in tasks such as natural language processing, and prediction. DL solutions are notorious to have achieved state-of-the-art performance on a wide range of tasks, and are widely used in industry and academia for various applications.

3 Introduction To Natural Language Processing

Natural Language Processing is a subfield of Computer Science. It usually combines the study of linguistics and artificial intelligence (AI) to achieve tasks. The Goal of NLP is to enable computers to understand, interpret and generate natural language. We Have Seen In recent years that there is an uptick in NLP research and solutions for various tasks.

Most tasks in NLP heavily rely on the training dataset. This dataset can also have sensitive information which we would like to restrict access to users, especially malicious attackers. We would also like that our model won’t give any insights into our dataset. For this reason, privacy is a major concern in the field of deep learning, particularly in the domain of NLP. For instance, in order to train a model for disease classification, we are in need of data concerning patients diagnosed with the disease. In the scope of our project, we are creating a model that tries to identify spam and uses real emails which contain private information.

III Solution

1 Attacker Model

Since our main goal for the project is to create a Differentially private model, our attacker will be the attacker we have seen in class for Differential privacy. The Attacker will have access to the trained model (and an interface to run it) as well as access to its outputs. He aims to infer sensitive information about individual data points or individuals in the dataset. Specifically, the attacker might try to use the trained model to:

1. Identify whether an email was included in the training dataset or not, by comparing the output labels of the model for different inputs.
2. Identify the sensitive features or attributes of individuals in the dataset, (age, gender etc).

For example he could try using our model to find information about the type of spam the model has seen. This gives him an insight on the type of spam the users got. He can possibly also try and find keywords our model uses to identify spam. He can try sending our model emails as if he is some named company(e.g. Amazon) and see how our model handles them thus gaining him insight about our users.

2 Differential Privacy in a world of Deep Learning

Having considered the issues of deep learning with special regards in the domain of NLP. And having seen the Attacker model for Differential Privacy We would like our model to satisfy Differential Privacy to handle this type of attacker model.

While this concept might seem easy in theory, it can be quite challenging to train a DL model to solve a task. Under the constraint of Differential Privacy, this can be even more challenging. Fortunately, Throughout our prior research we have come across the paper of Abadi Et. Al. that addresses the Problem of Differential Privacy in deep learning [1]. Its main contribution is a Stochastic Gradient Descent Variant making the DL model Differentially private. In the scope of our work, we will be using the safety and correctness of the presented algorithm which are proven in the paper.

3 Differentially Private Stochastic Gradient Descent (DP-SGD)

Nowadays, the main algorithms used for training deep learning models are SGD and its variants. It is a simple but effective algorithm which given samples $\{x_1, \ldots, x_n\}$ and a loss function $\widehat{f}_\theta(x_1 \cdots x_n) = \sum_i \quad f_\theta(x_i)$ computes the gradients with respect to θ and takes a step in the negative direction of the gradient, which is the direction of steepest descent of the loss function. We usually take a random batch to approximate the gradient and take a step in that direction

DP-SGD is a variant of that Basic algorithm which enhances our model's differential privacy. It calculates the gradient for each sample and clips them. We do so for the reason that if the gradients are too big for a particular sample, it means that this sample is very important for the model and thus can affect our privacy negatively. We then do a step as in the regular SGD but we also add noise to the step. This is the full algorithm as given in the paper:

Algorithm 1 Differentially private SGD (Outline)

Input: Examples $\{x_1, \ldots, x_N\}$, loss function $\mathcal{L}(\theta) = \frac{1}{N}\sum_i \mathcal{L}(\theta, x_i)$. Parameters: learning rate η_t, noise scale σ, group size L, gradient norm bound C.

Initialize θ_0 randomly

for $t \in [T]$ **do**

Take a random sample L_t with sampling probability L/N

Compute gradient

For each $i \in L_t$, compute $\mathbf{g}_t(x_i) \leftarrow \nabla_{\theta_t}\mathcal{L}(\theta_t, x_i)$

Clip gradient

$\bar{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i) / \max\left(1, \frac{\|\mathbf{g}_t(x_i)\|_2}{C}\right)$

Add noise

$\tilde{\mathbf{g}}_t \leftarrow \frac{1}{L}\left(\sum_i \bar{\mathbf{g}}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})\right)$

Descent

$\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{\mathbf{g}}_t$

Output θ_T and compute the overall privacy cost (ε, δ) using a privacy accounting method.

In fact, it simplifies the process of making our model differently private, given that the changes in the algorithm are relatively simple (Under the assumption the implementation at our disposal is efficient).

4 State of the art

It is important to note that we could not find any deep learning spam filter that is differentially private. We did find two papers about spam filtering and privacy but none of them satisfies both the requirements of differential privacy and being a DL model.

The first paper is pretty old and called “Privacy-Preserving Spam Filtering” [2] it shows a way for users to share their private email data to train and apply spam filters while satisfying privacy constraints. It uses logistic regression and Homomorphic Encryption to do that.

The second paper is much more recent; it is called “Fast Privacy-Preserving Text Classification based on Secure Multiparty Computation” [3] and proposes a private-preserving naive bayes model for the job which satisfies the Universal Composability (UC) framework. This job tries to filter SMS spam. Which is quite different from email spam in essence.

It is important to note that our privacy guarantees are different then those papers we found.

Since there are so many differences between those papers and our project, comparing the results of the papers to our results seems counterintuitive.

IV Implementation

We have implemented two models of email spam classification, the first being a RNN(LSTM) Neural Network model and the second model is RNN(LSTM)+Attention Based Neural Network Model. For enhanced differential privacy we have used the Opacus package which provides us with a Differentially Private Optimizer with proven differential privacy performance as stated in the article [1]. We have trained the model on the Enron database containing about 30,000 mails. Finally, we have tested the model performance in terms of its differential privacy but also in the terms of its classification's accuracy. Below we will explain each of the tools we have used throughout the project:

1 Database/preprocess

The Enron spam mail database is a collection of email messages that were exchanged among employees of the Enron Corporation. The database includes both legitimate and spam emails, and is often used as a benchmark dataset for training and evaluating spam filtering algorithms. The database contains about 30,000 emails. It is important to note that it contains sensitive information about Enron employees, and should be used with caution.

In our case we take the subject and text of the email. We converted each part to a list of tokens and then combined the two lists of tokens into one list with a special token in between to represent the end of the subject. Additionally, we

added unique tokens at the start and the end of the token list. We pad (or truncate) each email to have 128 tokens.

2 LSTM Model

Recurrent Neural Networks (RNNs) are a type of neural network architecture that are commonly used in sequence-to-sequence modeling tasks, such as natural language processing and speech recognition. RNNs are able to process input sequences of variable length and produce output sequences of variable length. The state of the network at each time step depends not only on the current input, but also on the previous state. This allows the network to capture temporal dependencies and model sequences of arbitrary length.

LSTM is a type of RNN architecture. It stands for Long Short-Term Memory, which is a type of neural network architecture that is commonly used in natural language processing. LSTM has gained popularity as a solution to the vanishing gradient problem [4][5]. LSTM networks are commonly used in natural language processing tasks such as language translation, sentiment analysis, and text classification. They have been shown to be effective in capturing long-term dependencies in sequential data, and are widely used in practical applications of deep learning.

Our LSTM model is very simple. We use the list of tokens mentioned above by converting them into embedding(vector in R^d). This is given to our LSTM model and then another Linear layer to get a single output. The output is between 0 and 1 and represents how much the model thinks the email is spam (1) or ham (0).

3 Attention Model

Attention Networks are a type of neural network architecture that has been particularly successful in natural language processing tasks. In traditional neural network architectures, each input feature is given the same weight and is processed uniformly by the network. However, in natural language processing tasks, different parts of the input sequence may have different levels of importance for the task at hand. Attention networks aim to address this by allowing the network to selectively focus on different parts of the input sequence, based on their relevance to the task [6][7].

Our model yet again converts our tokens to embedding. Then, the embedding is fed to an LSTM whose output is given to the attention mechanism. Finally, it is reduced to one output by using a linear layer that will convert to a value between 0 and 1

4 Opacus

When trying to build our model and implement our algorithm, we did our best to implement the algorithm as given, but unfortunately using our own implementation made training a single model take a lot of time. So we needed to find a solution for this.

Fortunately for us we found the library, Opacus. Opacus is an open source library for deep learning models with differential privacy. It is built to work on top of the PyTorch framework and extend it. Opacus emphasizes the performance and the differential privacy of the tool especially in large scale training. It contains an implementation of the Differentially private SGD algorithm we described above in section III. Moreover, it offers a lot more in terms of privacy for DL than that. For example, It also includes layers which are designed to work with privacy in mind and those layers use a cryptography secure PRnG.

5 Hyperparameter search (WandB)

Hyperparameter search is a very important part when building a DL model. In our case we needed to find the best parameters for our models that give the best results in terms of accuracy.

Below is a list of some of the hyperparameters we had synchronously tuned:

- Embedding size
- LSTM output size
- Dropout
- Number of heads for the attention
- Learning rate

All of those parameters can affect our model in major ways. We chose the models based on a validation set no model has seen.

In order to find the best parameters for our model we used the tool W&B (i.e. "Weights & Biases") WandB provides a range of features and functionalities for machine learning experiment management. WandB is designed to work with a range of machine learning frameworks, including PyTorch. Throughout our project we have used W&B in order to find the parameters that maximize the models' performance.

6 Hyperparameters Choice

Using all the tools above we have created our model and chosen those hyperparameters. It is important to note that our hyperparameter search was used without the Differential Privacy. Those models will be our baseline for the evaluation we are about to show.

We saw that most parameters gave us similar results between LSTM and the Attention model. So as to make the comparison with the private modes easier we chose to keep the parameters basically the same between our models:

The hyper parameters we chose:

- Embedding Size: 128
- Hidden Dimension: 64
- Number Of Layers: 2
- Number Of Heads: 4 (only for attention model)
- Learning Rate: 0.008576
- Dropout: 0.04906

After we have chosen our parameters we have trained the models again with a Differential Privacy algorithm to see how this algorithm changes our results.

V Evaluation

1 Privacy Evaluation

There are two ways to train a model DP-SGD. We can use the algorithm to train our model without any (ϵ, δ) budget. Otherwise, we can give our model a target (ϵ, δ) budget that we want our model to uphold. We tried both modes so we can see the effect those modes have on our models. We will compare the models and the different modes.

We should note that our baseline models got the following test accuracy:

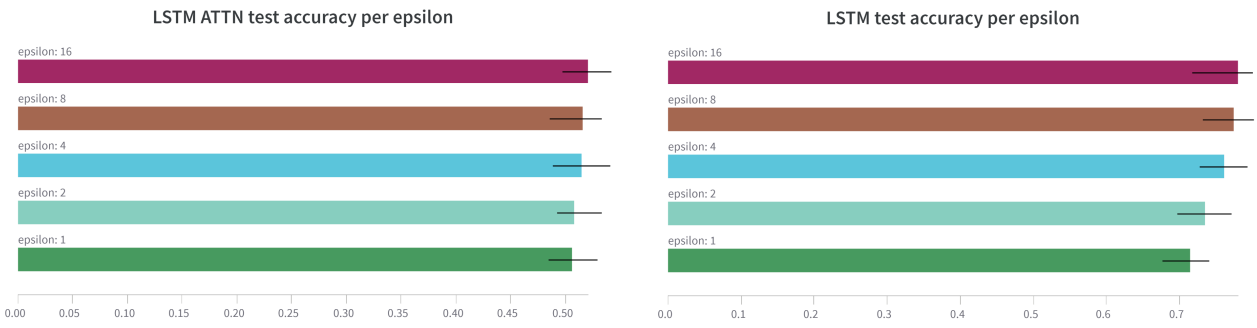
- LSTM: 97.6%
- LSTM with Attention: 97.2%

We have used the Opacus implementation of the Privacy Random Variable (PRV) Accountant to train our model and estimate our epsilon [8].

2 Budget Training Evaluation

When training our model using our budget mode (which will probably be the most used training mode when creating a production model, since we would like to have privacy guarantees on a production model) there are 3 params that affect privacy: ε, δ and maximum Gradient Norm. We wanted to see the effects of those parameters on the accuracy of our models. We Trained our models with different params and calculated the average test accuracy.
Below are our results:

First of all is the effect of the budget epsilon on the accuracy of the models:

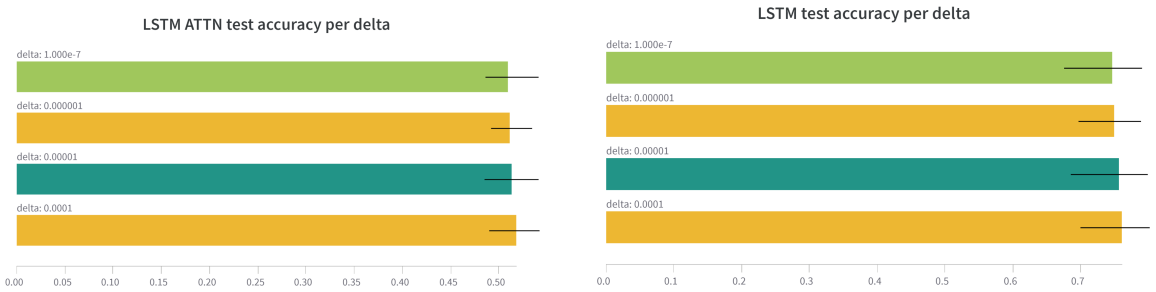


It should first be remarked that there is a major difference between the LSTM and LSTM attention model. The LSTM model correctly classifies more than 70% with any epsilon given while the LSTM with Attention gets about 50%. This is indeed a noticeable difference, which we were very surprised to discover.

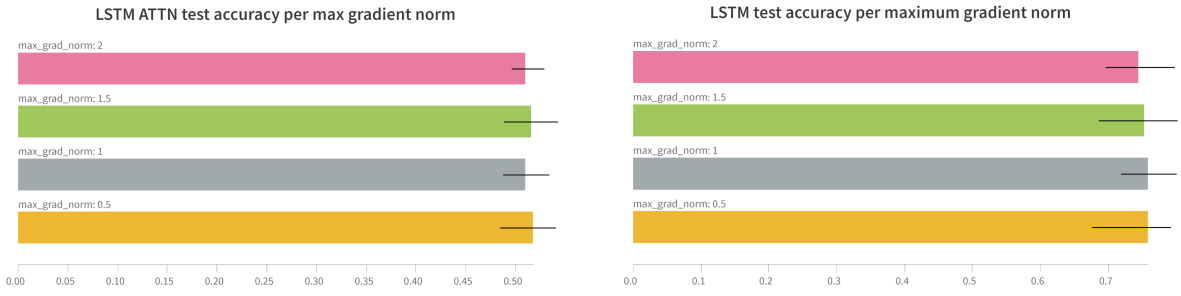
Especially when our baselines for the models got very similar results. This is also something we have noticed with all of our experiments.

We can also see from this graph that as you increase epsilon our models achieve better accuracy. This is inline with our understanding of differential privacy. The more we are willing to give in terms of privacy, the better our model will likely become.

Next, we will look at the effect of delta on our accuracy:



Notice that as δ gets smaller we again see that accuracy gets worse. It is important to note that delta seems to have a much smaller effect on the accuracy of our models than the value of epsilon.



The effect of the maximum gradient norm seems to be less clear. And while there are changes in the values of the accuracy they seem very small so we can probably attribute those changes to randomness.

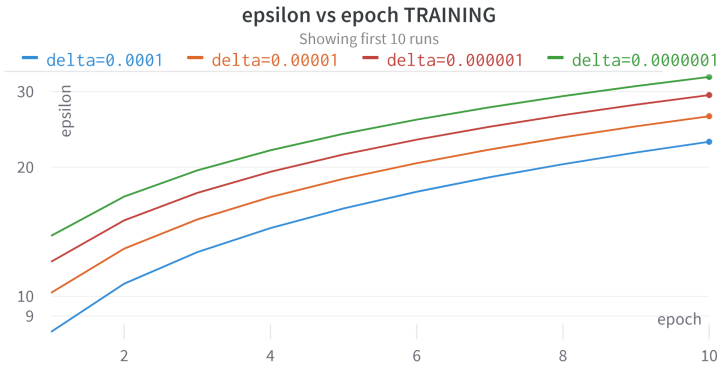
From this part we can conclude that when we train our models with an (ε,δ) budget we should think about the accuracy we are willing to sacrifice and how much privacy we are willing to give to our users.

3 Private Training Evaluation

When training a model with Private Training there are two parameters that are used: max gradient norm and noise multiplier.
We wanted to test two things in this mode. What happens to our model's epsilon value as the training of our model progresses. We would also like to know the effect of the noise multiplier on the epsilon of our model as the training of our model progresses.

It is important to note that both models got the same results since we used the same PRV accountant to calculate epsilon. And it seems that the maximum gradient norm doesn't affect epsilon at all.

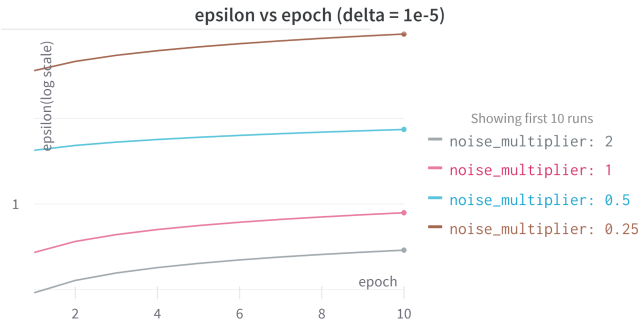
Let's first take a look at the progress of epsilon as our model trains:



We should notice that the Y-axis is in log scale.
We see that as our model trains and learns, our epsilon increases.
We can also see that as delta increases, our epsilon decreases. This is coherent with our intuition. That is because, as the model learns more he will "leak" more data which means epsilon increases. We can also see that as delta increases, epsilon decreases. This again agrees with our intuition. The bigger the absolute

difference between $P(M(d) \in S)$ and $e^{\varepsilon}\ P(M(\hat{d}) \in S)$ (which is represented by delta) the smaller epsilon can be and our model will still satisfy differential privacy.

If we look at the effect of noise multiplier on our epsilon we will get:



We see that again the epsilon increases as time progresses. We also see that the higher the noise multiplier is, the lower our epsilon is. This again makes sense because the noise multiplier represents the variance of our randomness in DP-SGD. The higher the variance will mean that the noise will have more effect on the training of our model. Which will give us a noisier model which will "leak" less information. Thus the model will be more private and will have a lower epsilon.

4 Final Models Evaluation

After exploring the notion of DP and the effect of the parameters we wanted to choose some models which we will use as tools for spam filtering. Thus, We chose 10 models that can be used according to the needs of privacy and accuracy:

- We will have a normal model (the ones we used for our baseline) that have zero privacy guarantees but have the best accuracy we have seen.

- There will be 4 models trained with budgets (2 for LSTM and for LSTM ATTENTION). For each, we will take the model with the best results we've seen, and the other will be the one with the best privacy guarantee.
- There will be 4 models trained with Private Training (2 for LSTM and for LSTM ATTENTION). For each, we will have a model with the best results we've seen, and the other will be the one with the best privacy guarantee.

Below is a Table of the our models and their results:

Model	**Training Type**	**Accuracy Type**	**δ**	**ε**	**Test accuracy**
LSTM	Normal	-	-	-	97.6%
LSTM	Budget	Best	1e-4	8	76.6%
LSTM	Budget	Private	1e-5	1	73.7%
LSTM	Training	Best	1e-5	96.6	81.5%
LSTM	Training	Private	1e-5	7.4	76.2%
LSTM ATTENTION	Normal	-	-	-	97.2%
LSTM ATTENTION	Budget	Best	1e-4	16	54.7%
LSTM ATTENTION	Budget	Private	1e-6	0.99	49.4%
LSTM ATTENTION	Training	Best	1e-5	96.6	51.8%
LSTM ATTENTION	Training	Private	1e-5	0.2894	47.8%

This table also lets us draw conclusions for the evaluation of our models. It follows that adding differential privacy does affect our accuracy results. Moreover, we can also conclude that the choice of our model does affect the amount of accuracy drop we will see. Indeed, in the case of LSTM we see a drop of about 25% in accuracy when using DP-SGD while the effect of DP-SGD on LSTM with attention seems much worse at stakes of about 50%. Note that

when we are using the LSTM model with differential privacy we get an accuracy of about 75% which means it is still a good model to use for spam filtering, which is what we wanted to build in the first place.

Finally, it is important to note that there are more options to explore, from parameters to other architectures using more advanced NLP techniques. Those may improve the results we have seen in this project.

VI conclusion

In this project we talked about the need for privacy in Deep learning and in particular the field of NLP. We wanted to create a DL model that is differentially private to protect the privacy of our training data. To do that we had to dive deep into NLP and its connection with privacy to finally achieve our original goal of a Differentially Private spam filter using a DL model.

We were able to study the effects of privacy on our model, we saw that while privacy does affect the accuracy of our model we can still get a DL tool who has privacy guarantees while still doing its job fairly well.

This project also gave us some interesting followup questions. For example, why does the private LSTM+Attention model work so much worse then the private LSTM model? How can we use more advanced NLP techniques to improve our Privacy results and accuracy results? Another further interesting question that can be subject for further research is which privacy tools can be added to our model to enhance its safety/privacy? And finally it is important to note that our implementation tools are generic and can be used for many different NLP tasks and not just spam filtering (e.g. sentiment analysis,,

language translation etc.) , giving rise to many possible projects that include both NLP and differential privacy.

Thus, to conclude, the use of differentially private neural networks has shown promising results in the field of spam mail detection. Even though there were some challenges associated with the implementation of differentially private neural networks such as the trade off between privacy and accuracy, the benefits of using these models are obvious. The use of differentially private neural networks for DL is a promising and interesting approach that needs further research. As more research is conducted in this area, it is likely that we will see even more innovative and creative applications of the DP-SGD in service of DL in the future.

Thank you for reading 🙂!

VII Bibliography

[1] Abadi, Martin, et al. "Deep learning with differential privacy." *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 2016. https://arxiv.org/pdf/1607.00133.pdf

[2] Pathak, Manas A., Mehrbod Sharifi, and Bhiksha Raj. "Privacy preserving spam filtering." *arXiv preprint arXiv:1102.4021*(2011). https://arxiv.org/pdf/1102.4021.pdf

[3] Resende, Amanda, et al. "Fast privacy-preserving text classification based on secure multiparty computation." *IEEE Transactions on Information Forensics and Security* 17 (2022): 428-442. https://arxiv.org/pdf/2101.07365

[4] Hochreiter, Sepp. "Recurrent neural net learning and vanishing gradient." International Journal Of Uncertainty, Fuzziness and Knowledge-Based Systems 6.2 (1998): 107-116. https://www.bioinf.jku.at/publications/older/2904.pdf

[5] Sundermeyer, Martin, Ralf Schlüter, and Hermann Ney. "LSTM neural networks for language modeling." Thirteenth annual conference of the international speech communication association. 2012. http://smil.csie.ntnu.edu.tw/ppt/20130516_johnhao1206_879_Paper.pdf

[6] Niu, Zhaoyang, Guoqiang Zhong, and Hui Yu. "A review on the attention mechanism of deep learning." *Neurocomputing* 452 (2021): 48-62. https://www.sciencedirect.com/science/article/abs/pii/S092523122100477X

[7] Wang, Chongren, et al. "A deep learning approach for credit scoring of peer-to-peer lending using attention mechanism LSTM." *IEEE Access* 7 (2018): 2161-2168. https://ieeexplore.ieee.org/iel7/6287639/6514899/08579130.pdf

[8] Gopi, Sivakanth, Yin Tat Lee, and Lukas Wutschitz. "Numerical composition of differential privacy." *Advances in Neural Information Processing Systems* 34 (2021): 11631-11642. https://proceedings.neurips.cc/paper/2021/file/6097d8f3714205740f30debe1166744e-Paper.pdf