Eduard Josep Bel Ribes

LEVERAGING INTER- AND INTRA-CLASS DISTANCES FOR POISONING ATTACKS

MASTER'S THESIS

Directed by Dr. Alberto Blanco Justicia

Master's Degree in Computer Security Engineering and Artificial Intelligence

UNIVERSITAT ROVIRA i VIRGILI

Tarragona
2023

Acknowledgements

I want to thank...
blablabla

Resum

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat.

Resumen

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna.

Abstract

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna.

Contents

List of code snippets

List of Figures

List of Tables

1 Introduction

In today's age of information and connectivity, advances in Artificial Intelligence (AI) and Machine Learning (ML) have transformed the way users interact with technology and process data.

Among the emerging paradigms in the field of ML, Federated Learning (FL) appeared as an innovative approach to train AI models in a distributed and decentralized environment.

In the last decade, ML has revolutionized the way in which we face complex problems in different areas, from computer vision to natural language processing. The last two years have been filled with news about promissing new paradigms of image generation, classification, chatbots, speech regonition and other AI's and, as time goes by, the applications of AI are becoming more present in our daily lives.

We can find examples of applications using FL in examples such as the text predictive keyboard that we can find on our mobile phones (Google's Android Keyboard [2]). We also find FL in Apple's assistant Siri voice recognition. This technology helps distinguish wether it is the main user of the smartphone saying "Hey, Siri", or another iPhone user attempting to activate Siri on their phone. As a final example, FL is used in more complex applications such as the Tesla autonomous driving system. In all three cases, the use of FL allows the machine learning models to be trained with the users' data without them having to share their data with third parties. In the case of Google's predictive keyboard, the model is trained with the users' data locally on the device, while in the case of Tesla, the users' data is used to train the model in a distributed way among the vehicles in the Tesla fleet.

Despite its advantages in terms of privacy, scalability and efficiency, FL presents significant challenges. One of them is its vulnerability to attacks: these attacks can exploit the distributed nature of the learning process, aiming to compromise integrity, confidentiality and the model's efficiency. As seen in the real-world examples, if an attacker exploited a vulnerability of the Tesla autonomous driving system by modifying the action that the car takes when recognizing a STOP sign, changing it to accelerating at full capacity, it could lead to multiple car accidents.

As Federated Learning systems are increasingly integrated in real-world applications, it becomes necessary to understand and address these challenges to guarantee a successful and secure deployment of this technology.

The GitHub repository "LFighter" [3] by Najeeb Jabreel is this thesis inspiration. It already offers a working FL simulator, where the programmer can modify the environment's parameters and obtain results of attacks against a variety of servers with different rules. With the explicit purpose of formulating and deploying attacks against FL systems to assess their robustness, this simulator serves as the foundation for the development of this thesis.

1.1 Motivation

Since the paper proposing FL got published in 2016???????, this new paradigm has been increasingly used over the years as we can see in Figure 1. With its increase in popularity and, its implementation in sensitive applications such as Tesla's autonomous driving system, our concerns about the security of this technology also increase.

From this concern, the motivation of detecting possible vulnerabilities so they can be addressed arises.

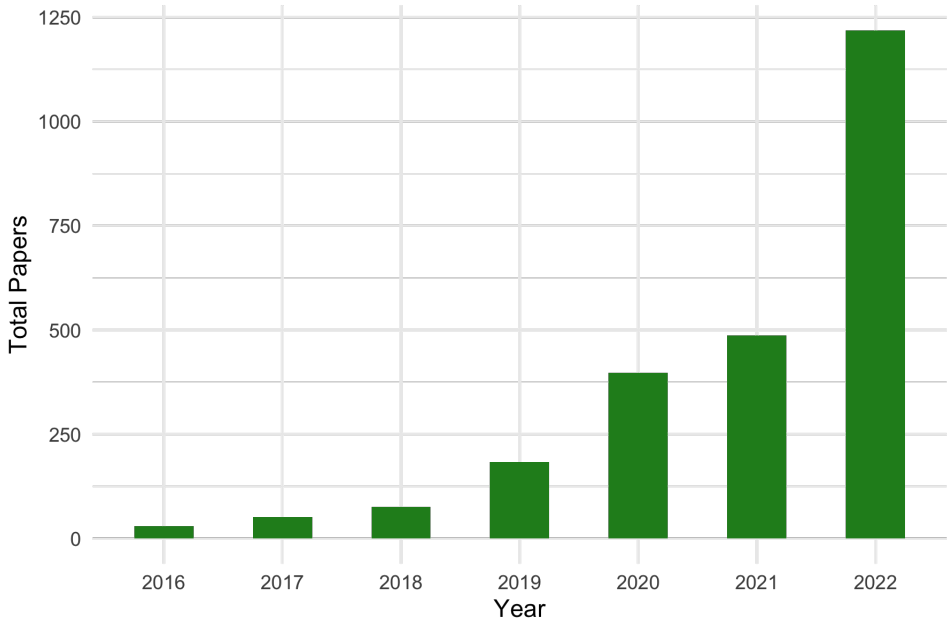

Figure 1: Published papers per year since FL was proposed. Source: *Web of Science*

1.2 Objectives

The main theoretical objective of this thesis is to explore and examine in detail the kind of attacks that can be directed towards Federated Learning systems, as well as identifying strategies and solutions to mitigate these attacks.

The practical objective is to examine the effectiveness and implications of employing a sophisticated label flipping technique compared to a straightforward approach when targeting a Federated Learning system. Specifically, the focus is on evaluating whether a more strategic and intelligent choice of samples to suffer a label flipping can lead to greater success for potential attackers compared to indiscriminately flipping all labels. This investigation represents an essential step towards comprehending the vulnerabilities and potential weak points within the Federated Learning paradigm. Furthermore, we aim to provide results on some of the most used aggregation rules in Federated Learning systems.

The research may be able to identify patterns and insights that conventional methods of attack might miss by examining the results of strategically manipulated label flipping and comparing them with the brute-force method. The results of this thesis will be valuable in gaining a deeper understanding of the security implications of FL and to develop more robust and secure FL systems in case the conceived attacks succeed.

In summary, this thesis conducts a critical examination of the viability of using intelligent label flipping techniques in comparison to a brute-force approach when attacking a Federated Learning system.

1.3 Outline

This thesis is organized as follows: . IDJQWOIJIOQ

. DHUIQWIHDIUQWUHI

2 Background

intro to the section, what are we going to talk about?

recom by chatgpt of the outline:
Explanation of the federated learning paradigm. Advantages and challenges of federated learning. Privacy and Security Concerns in Federated Learning Discussion of the privacy implications of decentralized training. Overview of security challenges in federated learning. Model Poisoning Attacks in Federated Learning Introduction to different types of model poisoning attacks. Discussion of existing literature and research on model poisoning attacks in federated learning. Highlighting the need for robust defenses against such attacks. State of the Art in Model Poisoning and Defenses Detailed exploration of recent advancements and techniques in model poisoning attacks, with a focus on label flipping attacks. Overview of state-of-the-art detection and mitigation strategies against model poisoning attacks in federated learning.

2.1 Federated Learning

What is it? where can it be found? pros?
Federated learning was presented in (McMahan et al., 2017) as the learning task solved by a loose federation of participants devices, which are coordinated by a central server.
...explica una mica mes
Therefore, FL allows building an entire ML model without sharing the clients' data (which remain in their local devices), and by leveraging the computation capabilities of the clients' devices (thereby alleviating the load at the server).

Use cases of federated learning include text prediction for smartphone keyboards (Bonawitz et al., 2019), speech recognition in intelligent assistants (Leroy et al., 2019) or video recommendations (Ammad-Ud-Din et al., 2019).

Federated Learning (FL, McMahan et al., 2017a) has emerged as a promising paradigm for training machine learning (ML) models using decentralized data.
The FL training process involves peers fine-tuning a global model received from the server on their local data to compute local model updates that they upload to the server, which aggregates them to obtain an updated global model. This process is iterated until a high-quality global model is developed.
FL offers several advantages over traditional centralized machine learning: i) the server distributes the training computational load, which is significant for large-scale ML, across the peers' devices (e.g., smart- phones) (Bonawitz et al., 2019), ii) the peers and the server obtain more accurate models due to learning from rich, joint training data, and iii) privacy improves by not sharing the peers' local data with a central server.
This latter advantage makes FL a suitable option for scenarios dealing with personal data, such as facial recognition (Xu et al., 2017), voice assistants (Bhowmick et al., 2018), healthcare (Brisimi et al., 2018), next-word prediction (Hard et al., 2018), intrusion detection in IoT net- works (Mothukuri et al., 2022) and location-based services (Huang, Tong,

and Feng, 2022), or in case data collection and processing are restricted due to privacy protection laws such as the GDPR (European Commis- sion, 2016)

(MODIFICAR TOT, IDEA)

FL allows multiple peers to collaboratively train a model without sharing their personal data.

2.1.1 types of fl

Types of FL. Federated Learning is not limited to the horizontal FL framework. Several other types of FL frameworks have been developed to handle different scenarios (Mammen, 2021): • Horizontal federated learning (HFL): This is used when each peer has a data set with the same feature space but different sample instances. A classic use case is the Google Keyboard app, where participating mobile phones have different training data but the same features. • Vertical federated learning (VFL): This is used when each peer has a data set with different features but from the same sample in- stances. For example, two organizations with data about the same group of people but with different feature sets can use Vertical FL to build a shared ML model. • Federatedtransferlearning(FTL):ThisissimilartotraditionalML, where we want to add a new feature to a pre-trained model. An example of this is extending Vertical FL to include more sample instances that are not present in all collaborating organizations. • Cross-silo federated learning: This is a type of FL where partic- ipating peers are large distributed entities (e.g., hospitals, banks, and companies) that have abundant local data and computational resources, and are available for all rounds. The training data can be in horizontal or vertical FL format. • Cross-device federated learning: This is another type of FL where peers are small distributed entities (e.g., smartphones, wearables, and edge devices) that have limited local data and computational resources. In this type, the number of peers is large, and they are not available for all rounds. Usually, the training data are in hori- zontal FL format.

2.1.2 Security attacks on Federated Learning

Despite these advantages, FL is vulnerable to various security and pri- vacy attacks Regarding security, FL is vulnerable to poisoning attacks (Blanco- Justicia et al., 2021; Lyu et al., 2022). Since the server has no control over the behavior of the participating peers, any of them may deviate from the prescribed training protocol to attack the global model by conduct- ing either untargeted poisoning attacks (Blanchard et al., 2017; Wu et al., 2020b) or targeted poisoning attacks

In the former type of attacks, the attacker aims to degrade the model's over- all performance, whereas in the latter, he aims to cause the global model to misclassify some attacker-chosen inputs into an attacker-chosen class.

Furthermore, poisoning attacks can be performed in two ways: model poisoning (Blanchard et al., 2017; Wu et al., 2020b; Bagdasaryan et al., 2020) or data poisoning

In model poisoning, the attackers maliciously manipulate their local model parameters

before sending them to the server. In data poisoning, they inject fabricated or falsified data samples into their training data before local model train- ing. Both attacks result in poisoned updates being uploaded to the server in order to prevent the global model from converging or to bias

As FL becomes more prevalent in real-world applications, safeguard- ing its models against poisoning and privacy attacks becomes crucial. Several defenses against poisoning attacks have been proposed

Most of these defenses are effective against un- targeted poisoning attacks, but they impose a high computational cost on the server to filter out poisoned updates.

Moreover, they often become less effective or even fail against targeted poisoning attacks such as label- flipping attacks (LFs) or backdoor attacks (BAs)

We can use techniques such as homomorphic encryption or secure multiparty computation which securely aggregate updates before sending them to the server but, these techniques are computationally expensive and prevent the server from inspecting individual updates to detect and filter out poisoned ones.

To detect poisoning attacks, the server requires direct access to individual updates

Therefore, simultaneously achieving security, privacy and accuracy is a tough challenge for FL.

2.2 Deep Neural Networks

uh
Deep neural networks (DNNs) are a class of artificial neural networks that contain multiple hidden layers between the input and output lay- ers. The hidden layers allow DNNs to learn more complex and sophis- ticated representations of the data they are trained on. This property leads to improved performance across a wide range of tasks, including computer vision, natural language processing (NLP), speech recogni- tion, recommendation systems, and game playing.

(MODIFICAR TOT, IDEA) text.

2.3 defenses

"A more realistic scenario is one in which only a small fraction of clients participates in each global training epoch.". explain defenses used/ rules

explicar mail "idea FL"

With respect to their role, adversaries can be classified into four types: • Honest-but-

curious FL server. A curious FL server receives updates W it from each participant over time and uses Wit to infer information about the private data set of individual clients. • Malicious FL server. Such a server can perform powerful attacks because it can also control the view of each client on the global model. In this way, a malicious FL server can extract additional information about the private data set of a client. only observe the global parameters over time, W , and she can use the successive parameters of the model to infer information about the private data of other clients. • Honest-but-curious client. An adversarial honest-but-curious client i can t • Malicious client. An adversarial malicious client i can obtain the aggregated updates from all other clients and can craft her own adversarial parameter

updates in order to get as much info as possible about the private data of other clients

Privacy attacks to FL can be classified into two fundamental and related categories (Melis et al., 2019): • Membership inference attacks. They consist in determining whether an individual data record was in the training data set. The ability of an adversary to infer the presence of a specific record in the input data training constitutes an immediate privacy threat if the training data are private or sensitive. • Properties of training data inference attacks. In FL, the distribution of individuals belonging to different classes may differ among the various private data sets. This attack aims at inferring properties of a class: for example, for facial recognition models, if a class corresponds to a certain individual, the attacker could infer that the individual wears glasses (Fredrikson et al., 2015).

2.4 State of the art

(from najeeb's)

While federated learning offers several advantages over centralized learning, it remains vulnerable to poisoning and privacy attacks due to its decentralized approach. In fact, the distributed nature of federated learn- ing can exacerbate these attacks compared to traditional centralized learn- ing

FL is still vulnerable to security and privacy attacks. Regarding security, FL is vulnerable to Byzantine attacks, which aim at preventing the model from converging, and to poisoning attacks, which aim at causing convergence to a wrong model

2.4.1 Privacy attacks

FL prevents private data sharing, but exchanging local updates can still leak sensitive information about the peers' data to attackers

Local gradients or consecutive snapshots of FL model parameters can reveal unintended training data features to adversaries, as DL models remember more fea- tures than needed for the main task

Peers' lo- cal updates are derived from their private training data, and the learnt model represents high-level statistics of the data set it was trained on. Therefore, those updates can reveal private information such as class representatives, membership, and properties of

the local training data. Attackers can infer labels from shared gradients and even recover train- ing samples without prior knowledge of the data (Zhu and Han, 2020).

2.4.2 Poisoning attacks against FL

Federated learning FL is vulnerable to poisoning attacks, which aim to manipulate the training process by injecting malicious data or model updates. Poisoning attacks against FL systems can be broadly catego- rized into two types: untargeted (Blanchard et al., 2017; Wu et al., 2020b; Fang et al., 2020) and targeted Both targeted and untargeted poisoning attacks can be carried out during the training phase of FL, either on the local data or on the local model. Data poisoning attacks involve the injection of manipulated samples into the training data set, which can result in the model being trained on biased or misleading data. On the other hand, model poisoning attacks involve manipulating the model parameters, either directly or indirectly, during the local model training process.

2.4.3 Untargeted poisoning attacks

Untargeted poisoning attacks aim to compromise the availability of the global model without any specific goal or objective.

2.4.4 Targeted poisoning attacks

Targeted poisoning attacks aim at making the global model misclassify a set of chosen samples to an attacker-chosen target class while minimiz- ing the impact on the model performance on the main task.

2.4.5 To succeed

For an attack to succeed, the attacker needs to overwhelm the influence of the rest of the clients in the aggregation function.
The attacker can achieve this by either: • Boosting her own updates by a scaling factor • Colluding with other malicious clients(other attackers or other accounts of the same attacker) • Combining both approaches

2.4.6 Defenses against poisoning attacks

The defenses proposed in the literature to counter poisoning attacks are based on one of the following principles: • Evaluation metrics. Approaches under this type exclude or penalize a local update if it has a negative impact on an evaluation metric of the global model, e.g. its accuracy • Clustering updates. Approaches under this type cluster updates into two groups, where the smaller group is considered potentially malicious and, therefore, discarded in the model learning process.(MKrum) • Peers' behavior. This approach assumes that malicious peers be- have similarly, meaning that their updates will be more similar to each other than those of honest peers. Consequently, updates are penalized based on their similarity.(Foolsgold) • Update aggregation. This approach uses robust update aggregation methods that are not affected by outliers at the coordinate level, such as the median (Yin et al., 2018), the trimmed mean (Tmean) (Yin et al., 2018) or the repeated median (RMedian) (Siegel, 1982). In this way, bad updates will have little to no influence on the global model after aggregation. • Differential privacy (DP). Methods under the DP

approach (Bag- dasaryan et al., 2020; Sun et al., 2019) clip individual update pa- rameters to a maximum threshold and add random noise to the parameters to reduce the impact of potentially poisoned updates on the aggregated global model. However, there is a trade-off between the attack mitigation brought by the added noise and the performance of the aggregated model on the main task (flame)

2.5 what is label flipping lf

2.5.1 Text examples

Example text example text

3 Architecture

intro to the section, what are we going to talk about?

3.1 Project's architecture

architecture of Najeeb's code, where are my functions? scheme on how the system (server, epochs, peers, peer rounds) works

In the typical FL (a.k.a. horizontal FL), K peers and an aggregator server A collaboratively build a global model W. In each training iteration t 2 [1, T], the server randomly selects a subset of peers S

After that, the server distributes the current global model Wt to all peers in S. Besides Wt, the server sends a set of hyper-parameters to be used to train the local models, which includes the number of local epochs E, the local batch size BS, and the learning rate h.

After receiving Wt, each peer k 2 S divides her local data Dk into batches of size BS and performs E optimization steps on Dk to compute her update Wt+1, which she up- k loads to the server.

The federated averaging algorithm (FedAvg, McMa- han et al., 2017a) is usually employed to perform the aggregation Note that FedAvg is the standard way to aggregate updates in FL and is not meant to counter security attacks

3.2 Security attacks on Federated Learning

blabla

3.2.1 Example subsubtitle

3.2.2 Text examples

Example text example text

4 Implementation

intro to the section, what are we going to talk about?

4.1 Dataset

we began by using the MNIST dataset, but we had to change to the CIFAR-10 dataset because of the size of the MNIST dataset.
explain

• MNIST. It contains 70K grayscale images of handwritten digits ranging from 0 to 9 with a size of 28fi 28fi 1 pixels (LeCun et al., 1999). It is split into a training set of 60K examples and a testing set of 10K examples. • CIFAR10. It is made up of 60K color images belonging to 10 dif- ferent classes (Krizhevsky, 2009). The images have a size of 32 fi 32 fi 3 pixels and are split into a training set of 50K examples and a testing set of 10K examples.

both of them are defined as image classification datasets

4.2 Creating the flipping functions

What is it? where can it be found? pros?

4.2.1 Example subsubtitle

5 Results

intro to the section, what are we going to talk about?

5.1 Results for entropy-based label flipping

What is it? where can it be found? pros?

5.2 Security attacks on Federated Learning

blabla

5.2.1 Example subsubtitle

6 Example title

6.1 Example subtitle

6.1.1 Example subsubtitle

7 Text examples

Example text example text

7.1 Bold & italic text

Bold text

Italic text

7.2 In document refernces

Equation 1

7.3 Other documents refernce

referenced text[4] test cite [1]

7.4 Acronyms & footnotes

ACRONYM[1]

Footnote[2]

7.5 Hyperlinks / URLs

Exemple of named hyperlink

[1] Acronym text
[2] Footnote text

8 Example lists

8.1 Unordered list

- Item
- Item
- Item

8.2 Ordered list

1. Item
2. Item
3. Item

9 Equation example

$$a^b = c \tag{1}$$

10 Table example

	API Disponible	API Obsoleta	Dificultat	Característiques avançades
Camera	1	21	Senzilla	No
CameraX	21	N/A	Senzilla	Sí[3]
Camera2	21	N/A	Complexa	Sí

Table 1: Comparativa d'APIs de càmera

As we can see in Figure 2, it works

11 Image example

Figure 2: Logo URV

12 Code snippet example

For all minted listings is required to enable *-shell-escape* on the LaTeX executable and have pygments installed

```
import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None #to become the incidence matrix
    VT = np.zeros((n*m,1), int)  #dummy variable

    #compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)
...
```

Code 1: Code example

13 Diagram examples

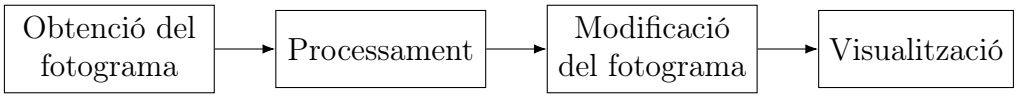

Figure 3: Projecte workflow

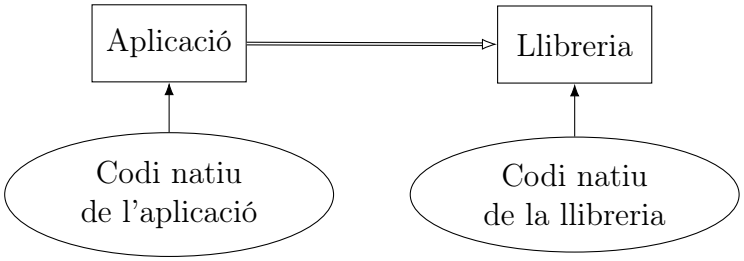

Figure 4: Module dependency

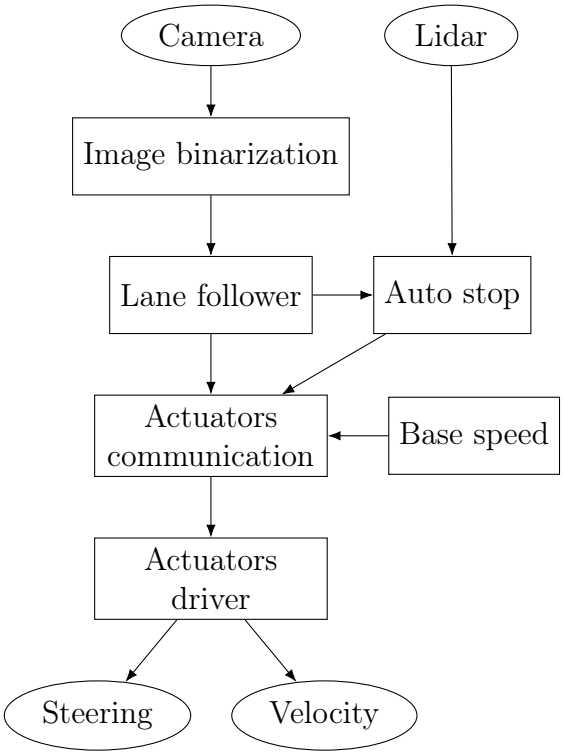

Figure 5: Car nodes layout

References

[1] Dirac, Paul Adrien M.: *The Principles of Quantum Mechanics*. Clarendon Press, 1981 (International series of monographs on physics). – ISBN 9780198520115

[2] Hard, Andrew ; Rao, Kanishka ; Mathews, Rajiv ; Ramaswamy, Swaroop ; Beaufays, Françoise ; Augenstein, Sean ; Eichner, Hubert ; Kiddon, Chloé ; Ramage, Daniel: *Federated Learning for Mobile Keyboard Prediction*. 2019

[3] Jabreel, Najeeb: *LFighter*. 2023. – URL https://github.com/NajeebJebreel/LFighter

[4] test: *Online Title*. 2021. – URL https://www.example.com

Appendix A:
Apendix example