Robust Federated Learning Based on Metrics Learning and Unsupervised Clustering for Malicious Data Detection

Jiaming Li
Kennesaw State University
Marietta, Georgia, USA
jli36@students.kennesaw.edu

Xinyue Zhang
Kennesaw State University
Marietta, Georgia, USA
xzhang48@kennesaw.edu

Liang Zhao
Kennesaw State University
Marietta, Georgia, USA
lzhao10@kennesaw.edu

ABSTRACT

Federated Learning has emerged as a new paradigm for improving communication efficiency and data privacy in various machine learning tasks. It allows the distributed devices to train the model collaboratively using their local dataset only. However, correctly labeled training data is a precondition for generating a high-quality model, whereas the real-world scenario usually cannot promise this condition. Conventional countermeasures mainly detect the corrupt local update and preclude them from the global weights aggregation phase to mitigate the impact of malicious data. Instead of discarding the weights update of clients, we propose a novel robust federated learning method that utilizes Metrics Learning to encode the local data and leverages the unsupervised clustering method K-means to preclude malicious data during local training. Therefore, correctly labeled data still contribute to the global model weight update with that the global model tends to be more generic. We evaluate the proposed method on two public image classification datasets, Fashion-MNIST and CIFAR-10. The simulation results demonstrate that the proposed scheme is robust for performing federated learning in the presence of malicious data.

CCS CONCEPTS

• **Computing methodologies → Distributed algorithms**; **Machine learning**.

KEYWORDS

Robust Machine Learning, Federated Learning, Neural Networks, Metrics Learning

ACM Reference Format:
Jiaming Li, Xinyue Zhang, and Liang Zhao. 2022. Robust Federated Learning Based on Metrics Learning and Unsupervised Clustering for Malicious Data Detection. In *2022 ACM Southeast Conference (ACMSE 2022), April 18–20, 2022, Virtual Event, USA*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3476883.3520221

1 INTRODUCTION

Federated Learning (FL) allows distributed devices to train a global model jointly without disclosing their private data to other devices. In the FL paradigm, distributed devices (or clients) train the model using its data locally and only share the weights update with the trusted central server that aggregates the model weights. Federated averaging (FedAvg) [14] is the most widely used weights aggregation technique. It enables the clients to obtain more training data to contribute more to the global aggregation phase so that the model tends to be more generic. The local training and global aggregation are processed repeatedly until the model converges. In addition, the model weight updates rather than the raw data are exchanged between the clients and the server.

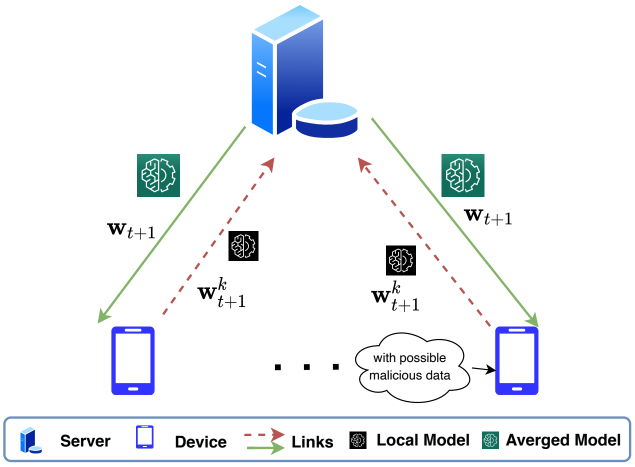

Figure 1: Federated Learning in the Presence of Malicious Local Training Data

FL has proved its superiority on several machine learning tasks including image classification [14], mobile keyboard prediction [6], health informatics [4] and etc. For example, mobile devices can serve as clients to train a machine learning model for a better user experience. However, correct labeling is not always guaranteed. Training data with faulty labels on clients result in poisoned weight updates, which is harmful to the global model performance. Furthermore, faulty labeling is common in real-world scenarios as data labeling is used to be conducted manually. An example of FL process with possible malicious data is illustrated in Figure 1. Former researchers have proposed many methods to improve the robustness of FL when local data is corrupted. Many of them perform statistical analysis on local weight updates. The assumption is that the clients containing the mislabeled data generate abnormal weight updates compared with the clients with perfectly labeled datasets. When clients are detected abnormally, the contribution of the clients is down-weighted or even assigned zero weight [11].

ACMSE 2022, April 18–20, 2022, Virtual Event, USA
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-8697-5/22/04…$15.00
https://doi.org/10.1145/3476883.3520221

However, mislabeling is very common in real-world scenarios. For instance, it would be difficult to distinguish poisoned weight updates when malicious clients are predominant. In addition, simply zero-weighting the contribution of corrupted clients makes the model hard to converge, even yielding biased global models since the contributions of both the fine-labeled and mislabeled dataset are discarded in the aggregation phase. Only a small portion of the dataset is utilized for training the global model. To address the issues above, we propose a novel robust FL solution that eliminates the potential mislabeled dataset on local clients by first encoding the local dataset to low-dimensional embeddings using metrics learning [7]. Then we used the unsupervised clustering method K-means [13] and voting to detect the possible mislabeled dataset and exclude them in the local training phase. Note that our robust FL solution is capable of handing corrupted or malicious data. Thus, we use them interchangeably in this paper. Our salient contributions are summarized as follows.

- We propose a novel robust paradigm that allows performing FL under malicious data corruption across local devices.
- We design a metrics learning and unsupervised clustering-based method for malicious training data detection that enables high-quality model training even with the penetration of corrupted data labels.
- We conduct extensive experiments to evaluate the proposed method and verify its robustness with various levels of data corruption among the local devices.

2 FL WITH MALICIOUS TRAINING DATA

FL was first proposed in 2017 that aimed to improve the usability of data collected from cell phones, tablets, and other IoT devices for training more intelligent applications. These devices are frequently carried and accessed by people, so they contain a dramatic amount of private data. In this scenario, collecting users' data to a centralized location is inefficient in data transformation and risky in data privacy disclosure [14]. FL offers a solution that fully uses the distributed data in mobile devices by collaboratively training the model on local devices and only sharing the local updates to the central server. Then the server aggregates the local update using FedAvg [14]. Assume P clients (marked 1, 2, ..., p) are selected to train the model, n_p is the number of local data for client p, n is the total number of the training data over clients, and w_t^p is the local weights of client p in time stamp t. The global model weight W_{t+1} after aggregation is $W_{t+1} \leftarrow \sum_{t=1}^{p} \frac{n_p}{n} w_t^p$. After weights aggregation, W_{t+1} is sent to clients for the next round of training until the global model converges. The one time of weight aggregation is also called one communication round.

FL can be adopted in language models to improve speech recognition, text entry, or image models automatically select good photos [14]. In those applications, users' behaviors are recorded for training the model. In return the quality of distributed data cannot be guaranteed since he the users may mislabel the data by mistakes or even on purpose.

In order to make FL robust to mislabeled data, current approaches focus on detecting the malicious local weights updates. Krum [4] detects the malicious local updates by comparing the local updates' Euclidean similarity. Besides comparing the local updates at the same timestamp, Li et al. [12] mitigate the Byzantine failure by first calculating the cosine similarity of all updates to filter out the malicious updates, then adjusting the learning rate according to the received updates for each communication round for the temporal perspective adjustment. The above-mentioned methods require high computational cost, and the normal clients must be dominant so that the malicious update can be distinguished. Besides, simply excludes the malicious local updates and also eliminates the contributions of benign data in local clients which results poor data usability. Instead of calculating the raw model updates' similarity, researchers also tried to train an autoencoder to first project the weight update to low-dimensional latent space to detect the malicious updates, then remove the noisy and irrelevant features and keep the essential features [10]. However, since the model structure varies, it is impossible to train a universal autoencoder that fits every model. Other robust FL solutions tend to sanitize the training dataset. Fang et al. [5] proposed a method that evaluates the negative impact on the error rate of the learned model to detect the malicious clients and finds the subset of the training dataset which minimizes the loss function to mitigate the local model poisoning attacks. Nevertheless, when the number of local training data is large, finding the optional subset is costly. To address this problem, we propose a novel method to eliminate the potential mislabeled dataset by training an encoder model that maps the training data to latent space, then using k-means clustering and voting to detect the possible corrupted data.

```
Algorithm 1: Clean Data Prediction Process
Input   : Triplet network $M$, local training data
          $X = x_0, x_1, ..., x_m$ and labels $Y = y_0, y_1, ..., y_m$
Output : Indices of predicted clean data $I_{clean}$
Initialize $I_{clean} = []$.
Generate the latent space feature representation:
$X^M = M(X)$.
Number of clusters:
$k = unique(Y)$.
Pseudo label:
$Y^k = KMeans(k, X^M)$.
Find most common $k$ mappings of $Y \rightarrow Y^k$:
$Mapping_k = (0, Y_i^k), (1, Y_i^k), ..., (k-1, Y_{m-1}^k)$.
Map $Y^k$ to the space of $Y$:
$Y^k = y_l, y_j, ..., y_q$.
for $i \leftarrow 0$ to $m$ do
    if $Y[i] == Y^k[i]$ then
        $I_{clean}$.append($i$)
```

3 PROPOSED ROBUST FL PARADIGM

The proposed malicious data detection methods consist of three steps. First, a small portion of the labeled data is selected to train the feature encoding model using deep metric learning. The encoding model learns a data representation by distance comparisons of the positive and the negative samples. Specifically, the triplet network [7] is utilized as an encoder. To prepare dataset for training the

Algorithm 2: Robust Federated Learning Paradigm

```
Input  : Dataset $\{X^M, Y^M\}$ for training the Triplet Network
         $M$, the number of training epochs $E$, the number of
         communication rounds $C$
Model Prepare Phase (Server Only):
for $e \leftarrow 0$ to $E - 1$ do
    Random select batches of $x^+, x, x^- \in X^M$.
    Fit $M$ using Stochastic Gradient Descent.
Federated Learning Phase:
Server passes triplet network model $M$ and initial weights to
  the clients.
Clients do data cleaning with provided model $M$ and their
  local datasets using Algorithm 1.
for $c \leftarrow 0$ to $C - 1$ do
    for each client $p$ do
        $X^p, Y^p \leftarrow \mathrm{x}_i, y_i$ where $i \in \mathrm{I}_{clean}$.
        $w^p_{t+1} \leftarrow LocalWeightsUpdate(\mathrm{w}_t, X^p, Y^p)$.
    $W_{t+1} \leftarrow \sum_{t=1}^{p} \frac{n_p}{n} w^p_t$.
```

triplet network, three samples are used as inputs, which are denoted as x, x^+, and x^-. x and x^+ denote two samples have the same label, x and x^- represent two samples with different labels. The loss function of the triplet network can be written as:

$$Loss(d_+, d_-) = ||(d_+, d_- - 1)||^2 \tag{1}$$

where d_+, d_- are the SoftMax result of the l_2 distances between the embedding representation of x, x^+ and x, x^- respectively. The data in the same category are closer in terms of l_2 distance. Triplet network can be used as a data encoder to project the training data to latent space.

In the proposed method, we assume the central server has a small portion of training data to train the triplet network. The trained encoder model is deployed to the local clients. Before the FL processing starts, the local data is sent to the encoder model and the data representations are generated. The data representations and the labels of the training data $Y = y_0, y_1, ..., y_m$ are sent to the K-means classifier. The number of clusters k is set as the number of unique labels:

$$k = unique(Y) \tag{2}$$

K-means clustering generates a set of pseudo labels Y^k. They are called pseudo labels, because they are not corresponding to the original label since K-means is an unsupervised learning. However, they can be used for detecting malicious data if we know the mapping between the original labels Y and labels generated by K-Means Y^k. Voting is used for estimating the mapping between Y and Y^k. Specifically, we count the number of pairs with label change $y_i \rightarrow y_i^k$, where y_i and y_i^k are the original label and the pseudo label with the same index i, we append index i to array I_{clean} to record the indices of estimated clean data. The mapping between the original labels and the pseudo labels can be estimated by the top k mapping after sorting all the mapping count. Finally, we use the estimated mapping to detect the mislabel data: only the data follows estimated mapping is kept for the federated learning process. Algorithm 1 illustrate the process for finding the indices of clean data. The whole process of the proposed robust FL method is described in Algorithm 2.

4 EXPERIMENTS

In this section, we conduct a series of experiments to show the mislabeled data is harmful to the global model and evaluate the performance of the proposed method by calculating the global model accuracy on the test dataset. The experiment ran on public image classification dataset Fashion-MINIST [16] and CIFAR-10 [9] to evaluate the effectiveness of our approach. For Fashion-MNIST, it contains 60,000 standardized images of fashion items from 10 classes: T-shirt/top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag, Ankle boot. The CIFAR-10 dataset contains 60,000 32x32 color images in 10 different classes represent airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks. There are 50000 training images and 10000 test images in each dataset.

4.1 Experiment Setting

The experiments are performed on Google Cloud N2 Series virtual machine (8v CPU, 32GB Memory). The proposed encoding and classification model was implemented by the TensorFlow [1] framework. Flower [3] framework is used to simulate the federated learning settings. In our experiment, one server runs in one process, and 20 clients run in separate processes of the same cloud virtual machine. Serve and clients are communicating via http protocol. In each communication round, the global weights are generated by FedAvg, and the global model accuracy on the test dataset is evaluated. The local data on each client are randomly selected from the global training dataset, which means the data distribution on the client is Independent and Identically Distributed (IID). Each client obtains an equal amount of dataset before the federated learning starts, possible malicious data can be eliminated by proposed method. Therefore, the corrupted clients intend to have less data and contribute less to the global model according to the FedAvg. For the corrupted client, the potion p of the total data is poisoned by randomly changing the label.

4.2 Model Structure

The embedding and the classification models are very basic Convolutional Neural Networks (CNN) [15]. In our experiment, the models are various with the input shape. For example, the model for MNIST dataset has input shape (28, 28, 1), while the model for CIFAR-10 has shape of (32, 32, 3). The embedding model consists of three convolutional layers (32x3x3, 64x3x3, and 128x3x3) activated by the ReLu function, followed by a global average pooling layer and one dense layer of 8 units. The classification model has 4 convolutional layers activated by ReLu function [2] (32x3x3, 64x3x3), with one max pooling layer in the middle and followed by 3 dense layers with 256, 128, and 10 units.

4.3 Training Process

In the experiments, 5,000 out of 50,000 data is randomly selected to train the embedding model which encodes the input image data to a vector with length 8. The data for training the embedding model will not be used for federated learning process. Adam optimizer

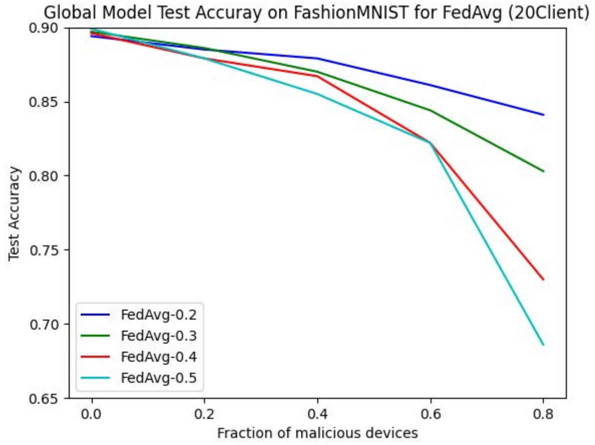

Figure 2: Global Model Test Accuracy of FedAvg

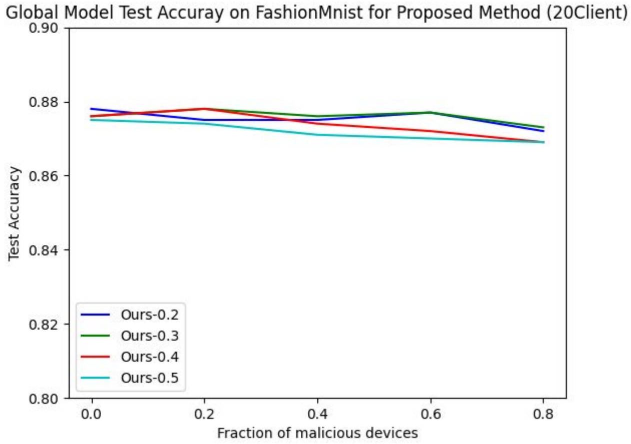

Figure 3: Global Model Test Accuracy of Proposed Method

[8] is used to minimize the sparse categorical cross entropy loss with initial learning rate 0.001. The encoder model is trained for 30 epochs.

In the federated learning process, in each communication round, the local model is trained for 30 epochs and the total communication round is 30. The classification model is optimized by Adam optimizer and the categorical cross entropy is minimized.

The accuracy of the final global model on test dataset (10,000 samples) is used to evaluate the performance of different methods. To show the proposed method is robust in extreme settings, the fraction of clients which have malicious data over all client is changed for each experiment. The total number of clients in our experiments is set to 20, the ratios of attacked clients are 0%, 20%, 40%, 60% and 80% for each experiment.

The first experiments is designed to show the robustness of proposed method when the portion of the malicious data various. The experiments are performed on FashionMNIST dataset. Figure

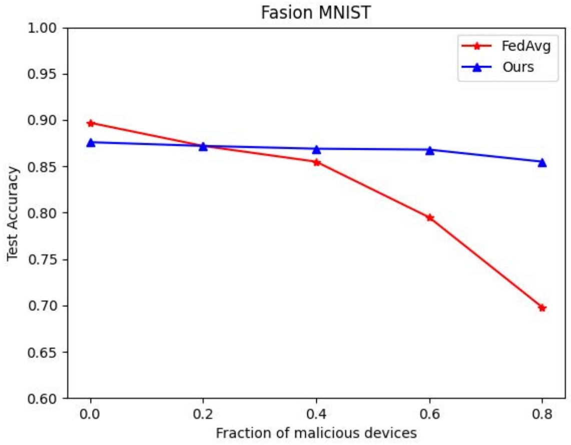

Figure 4: Test Accuracy Comparison of Two Methods on FashionMNIST Dataset

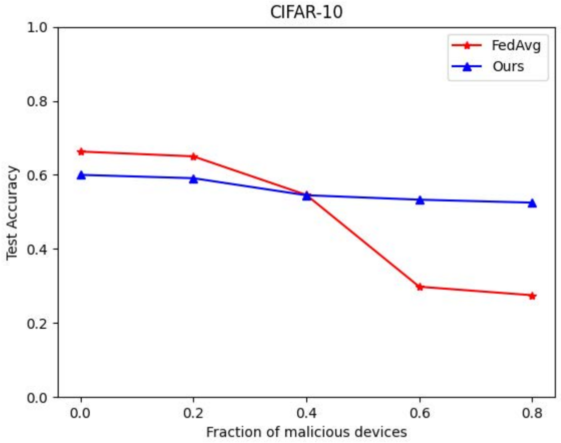

Figure 5: Test Accuracy Comparison of Two Methods on CIFAR-10 Dataset

2 and Figure 3 show the global model test accuracy comparison of proposed method and FedAvg when the fraction of malicious clients (when total number of clients is 20, the number of malicious clients are 0, 4, 8, 12, and 16 respectively) and the portion of corrupt data (0.2, 0.3, 0.4, 0.5 of the total training data) changes. As we can see in Figure 2, when the number of clients and number of malicious data increases, the performance of global model degrades if the corrupted data is not eliminated. In contrast, it is shown in Figure 3 that the performance of the proposed method keeps constant in the presence of various number of malicious clients. In addition, the model accuracy of our robust FL paradigm only slightly decreases when the ratio of local data corruption increases.

Figure 4 and Figure 5 show the global model test accuracy comparison of our method and FedAvg on FashionMNIST and CIFAR10 dataset when the local data corruption rate is fixed to 0.8. We can see from the two figures that as the fraction of the malicious devices increases, the test accuracy of the FedAvg drops greatly while the our method keeps stable in both datasets.

5 CONCLUSION

In this paper, we proposed a robust FL paradigm based on a novel malicious data detection mechanism. The malicious data detection design leverages metric learning to encode the data and utilize K-Means to classify the training data using the low-dimensional representations. The mislabeled data is eliminated according to the estimated label mappings to prevent it from degrading the model performance. Extensive experimental results demonstrate that the proposed robust FL methodology can achieve high accuracy when malicious clients and corrupted data are present.

REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale Machine Learning on Heterogeneous Systems. https://www.tensorflow.org/ Software available from tensorflow.org.

[2] Abien Fred Agarap. 2018. Deep Learning Using Rectified Linear Units (relu). *arXiv preprint arXiv:1803.08375* (2018).

[3] Daniel J. Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Javier Fernandez-Marques, Yan Gao, Lorenzo Sani, Kwing Hei Li, Titouan Parcollet, Pedro Porto Buarque de Gusmão, and Nicholas D. Lane. 2021. Flower: A Friendly Federated Learning Research Framework. arXiv:2007.14390 [cs.LG]

[4] Theodora Brisimi, Ruidi Chen, Theofanie Mela, Alex Olshevsky, Ioannis Ch Paschalidis, and Wei Shi. 2018. Federated Learning of Predictive Models from Federated Electronic Health Records. *International journal of medical informatics* 112 (2018), 59–67.

[5] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. 2020. Local Model Poisoning Attacks to Byzantine-robust Federated Learning. In *29th USENIX Security Symposium (USENIX Security 20)*. Virtual Event, USA, 1605–1622.

[6] Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. 2018. Federated Learning for Mobile Keyboard Prediction. *arXiv preprint arXiv:1811.03604* (2018).

[7] Elad Hoffer and Nir Ailon. 2018. Deep Metric Learning Using Triplet Network. arXiv:1412.6622 [cs.LG]

[8] Diederik P. Kingma and Jimmy Ba. 2017. Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs.LG]

[9] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. [n.d.]. CIFAR-10 (Canadian Institute for Advanced Research). ([n. d.]). http://www.cs.toronto.edu/~kriz/cifar.html

[10] Suyi Li, Yong Cheng, Wei Wang, Yang Liu, and Tianjian Chen. 2020. Learning to Detect Malicious Clients for Robust Federated Learning. *arXiv preprint arXiv:2002.00211* (2020).

[11] Shenghui Li, Edith Ngai, Fanghua Ye, and Thiemo Voigt. 2021. Auto-weighted Robust Federated Learning with Corrupted Data Sources. *arXiv preprint arXiv:2101.05880* (2021).

[12] Zhuohang Li, Luyang Liu, Jiaxin Zhang, and Jian Liu. 2021. Byzantine-robust Federated Learning Through Spatial-temporal Analysis of Local Model Updates. *arXiv preprint arXiv:2107.01477* (2021).

[13] Stuart Lloyd. 1982. Least Squares Quantization in PCM. *IEEE Transactions on Information Theory* 28, 2 (1982), 129–137.

[14] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient Learning of Deep Networks from Decentralized Data. In *20th International Conference on Artificial Intelligence and Statistics (AISTATS)*. PMLR, Fort Lauderdale, USA, 1273–1282.

[15] Keiron O'Shea and Ryan Nash. 2015. An Introduction to Convolutional Neural Networks. arXiv:1511.08458 [cs.NE]

[16] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. *Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms*. arXiv:cs.LG/1708.07747 [cs.LG]