Un-Fair Trojan: Targeted Backdoor Attacks Against Model Fairness

Nicholas Furth, Abdallah Khreishah, Guanxiong Liu, NhatHai Phan, Yasser Jararweh

2022 Ninth International Conference on Software Defined Systems (SDS) | 979-8-3503-4671-8/22/$31.00 ©2022 IEEE | DOI: 10.1109/SDS57574.2022.10062890

***Abstract*—Machine learning models have proven to have the ability to make accurate predictions on complex data tasks such as image and graph data. However, they are vulnerable to various backdoor and data poisoning attacks which adversely affect model behavior. These attacks become more prevalent and complex in federated learning, where multiple local models contribute to a single global model communicating using only local gradients. Additionally, these models tend to make unfair predictions for certain protected features. Previously published works revolve around solving these issues both individually and jointly. However, there has been little study on how the adversary can launch an attack that can control model fairness. Demonstrated in this work, a flexible attack, which we call Un-Fair Trojan, that targets model fairness while remaining stealthy can have devastating effects against machine learning models, increasing their demographic parity by up to 30%, without causing a significant decrease in the model accuracy.**

***Index Terms*—Machine Learning, Federated Learning, Backdoor Attacks, Fair Machine Learning**

I. Introduction

Recently, it has been shown that machine learning models are vulnerable to various attacks such as trojan backdoors [1] [2] [3]. When implemented in a Federated Learning (FL) [4] system where local models contribute to a single global model as opposed to a single centralized model, these vulnerabilities become more exploitable since a user often has access to one or more local models, compared to models trained in a centralized setting where accessing the model may be difficult. In addition to these vulnerabilities, machine learning models trained on real-world data often contain sensitive attributes such as race and gender that might result in unfair predictions. For example, a loan model predicting higher interest rates for women v.s. men if all other attributes are the same is considered unfair. To be considered fair, the prediction must be independent of the sensitive features. This dependence can be measured using a risk difference function.

Various works have focused on developing methods to increase the fairness of machine learning models w.r.t. their sensitive features both in centralized and FL settings [5] [6] [7] [8] [9] [10] [11]. The majority of existing works focus on increasing model fairness, with few exploring methods to decrease model fairness. Currently, there are two works that explore combining model fairness and a backdoor attack however the scope and limitations of these works differ much from our work. The work in [12] implements an attack that seeks to reconstruct the sensitive features in a data set, however, the attack itself has no impact on the fairness of the model. The work in [13] implements a basic attack against model fairness, however, it has several limitations. First, it only demonstrates an attack against a support vector machine model, failing to show effectiveness against more complex models such as deep neural networks, and is limited to a centralized setting. Additionally, it only explores the attack against tabular data and does not explore more complex data sets, i.e. image data, NLP data, etc. Unlike the existing works, this work explores how an attack against fairness can have an adverse effect on model fairness. In addition, we consider different types of data i.e. tabular and image data and we implement the attack in an FL setting with both i.i.d. and non-i.i.d. data, significantly improving the utility of the attack over the existing works.

Utilizing an attack to effect model fairness is more complex than a traditional backdoor attack where an attacker can use a small amount of poisoned data to make a model overfit to the trojan trigger with few epochs. For an attack against fairness to be successful, the attack needs to be able to increase or decrease the prediction's dependence on the sensitive features while maintaining high accuracy to remain stealthy. To achieve this goal we explore two methods; a modified version of adversarial label-flipping, and the traditional trojan attack, each with its challenges.

In adversarial label-flipping, the sensitive feature will have the labels of the sensitive feature flipped to match the ground truth with probability ρ which is adjusted to obtain the best fairness/accuracy trade-off. This keeps the attack stealthy and effective as possible. The goal of this method is to increase the correlation between the sensitive feature and the prediction which as a result increases the risk difference. Although this method has the advantage of being simple to implement, achieving a good result while also maintaining high accuracy is challenging.

The second method involves flipping the labels in the same manner only when a trojan trigger is present. Unlike the adversarial label-flipping attack, we control when the attack happens by attaching a trojan trigger to the input data. One advantage of this method is that we do not need to rely on the model learning how to predict the sensitive feature within an image, it only needs to overfit to predict a single class when the trigger is present. This method has several challenges and limitations, first, the sensitive features are not explicitly part of the data which makes it difficult for the model to overfit

Nicholas Furth is with the University of Tenneesee at Knoxville Email: nfurth@vols.utk.edu, Abdallah Khreishah, Guanxiong Liu, and NhatHai Phan are with New Jersey Institute of Technology Emails:{abdallah,gl236,phan}@njit.edu, Yaser Jararweh is with Jordan University of Science & Technology Email: yijararweh@just.edu.jo

to the trigger. Second, since we are using real-world data, the trigger may blend into part of the image or a benign image may unintentionally have the trigger which not only makes it harder to overfit, but during inference time the trigger may be activated without being present. Finally, this method is only effective against image data or data with non-discrete values making it less flexible compared to the label-flipping attack. Our experiments show that when we activate our attack, the demographic parity can be increased by over 30% without causing a significant decrease in model accuracy.

The remaining of this paper is organized into 5 sections. Section 1 summarizes the research objectives. Section 2 focuses on the literature review. Section 3 discusses in detail the way the attacks were implemented. Section 4 provides an overview of the model architectures used. Section 5 discusses the experiments and their results. Finally, Section 6 discusses conclusions and future work.

II. Background

This section examines the previous works done in fair machine learning in both centralized and FL settings. Additionally, it examines attacks against FL systems and discusses the importance of such research problems. This literature review aims to provide a clear understanding of the basic concepts that are used to develop the attacks in Section 3.

A. Federated Learning

Federated learning (FL) as shown in Algorithm 1 and Figure 1a, is a distributed computing method that trains multiple local models on their own data sets to obtain a single global model [4]. Subsequently, the parameters of each local model are then sent to a server for aggregation. Aggregation methods can be weighted or unweighted averages. Once the aggregation is complete, the global parameters are redistributed to each local model to begin the next iteration. This process is repeated until the global model converges. Over each iteration, each local model is exposed to a wider range of data. Sending only the local parameters protects local data from being seen by other models while allowing them to generalize better to new data. A simple aggregation method can be expressed as follows:

$$\theta_g^{t+1} = \frac{1}{m} \sum_{i}^{m} \theta_i^t \tag{1}$$

where θ_g^{t+1} represents the global parameters after aggregation at iteration t, θ_i^t is the local parameters of model i, and m represents the total number of local models selected in a training round out of n local models, where $m \leq n$.

Typically, the m models are selected based on several factors, such as battery life, internet connection strength, and the number of training epochs made since the previous global iteration. More complicated methods of FL are shown in [1] [2]. Furthermore, FL has desirable traits, such as that the local data of each model is never seen by the server or other models; preserving data privacy since only the parameters are communicated. The data privacy aspect of FL is the most important feature. A majority of local, state, and federal jurisdictions require user data to be kept private. By only communicating model parameters, instead of pooling the data to train a single model, FL models, which are used widely in healthcare can comply with regulations such as HIPAA, which require: (1) Ensuring the confidentiality, integrity, and availability of Protected Health Information (PHI) created, received, maintained, or transmitted, (2) Protecting against any reasonably anticipated threats and hazards to the security or integrity of PHI, and (3) Protecting against reasonably anticipated uses or disclosures of PHI not permitted by the Privacy Rule [14].

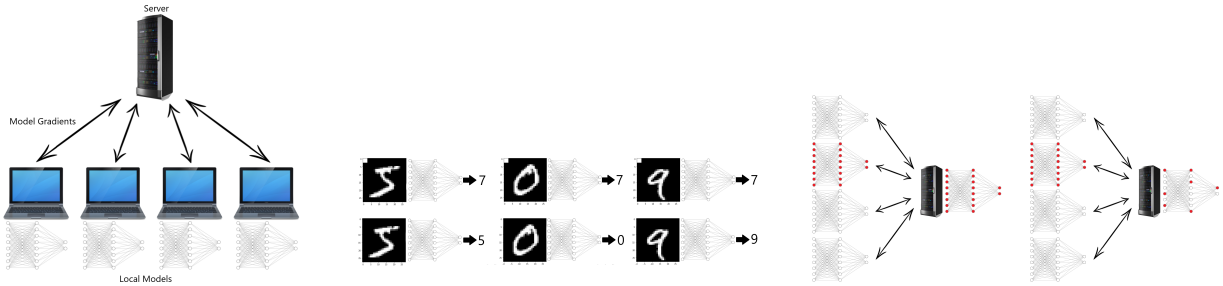
Algorithm 1 Federated Learning

INPUT: Data sets with data x_i, labels y_i, m local models selected each round, n total models, local parameters θ_i and global parameters θ_g.

for Each t in 1, 2, . . . **do**
  Select m $\leq$ n models
  for Each local model i **do**
    $\theta_i = \theta_g$
    Train local model i with parameters θ_i on data x_i and labels y_i
  end for
  $\theta_g^{t+1} = \frac{1}{m} \sum_i^m \theta_i^t$
end for

Due to the data handling criteria, FL has gained significant popularity in the medical field. Hospital networks can now train models using data from multiple locations without compromising patient privacy. This is especially important since not only do patient demographics vary from different localities, but so do the privacy regulations. In a similar manner to hospitals, financial institutions can use FL to train models to approve or decline loans, determine interest rates and detect fraud with models trained using data from multiple branches in multiple countries without disclosing sensitive customer information which cannot be shared internationally. Additionally, FL allows financial institutions to source data from different areas where customers may be wealthier than others allowing the model to generalize better to a wider range of customers while protecting customer data. An attack against this type of models would allow the attacker to make the model more favorable to their demographics, resulting in a higher approval chance or lower interest rate.

Since models are trained on multiple devices, there is less computational and memory strain on any single device due to its distributed nature. This allows applications to run on slower devices such as mobile phones or embedded devices, i.e. microcontrollers and IoT devices while still making a meaningful contribution to the global model and not straining their local computational and memory resources. Finally, since only the model gradients are communicated, the bandwidth needed is much smaller compared to communicating the local data of each model.



(a) Implementation of an FL system.

(b) Trojan trigger example for the MNIST data set.

(c) Model Replacement (Left) vs Naive Approach (Right)

Fig. 1: Demonstration of Methods Used

B. Fairness in Machine Learning

Machine learning models often utilize data that may contain sensitive features i.e. race, age, or gender. Using these features for decision-making is undesirable if the prediction results are biased towards these features. One of the root causes of this problem is that the models are trained using data that is biased. Various methods to remove the effect of these features have been implemented [5] [6]. Typically, this is done by either removing the correlation between the sensitive features and the output through the use of the objective function or by inserting perturbations that prevent the model from learning such correlations. Ensuring fairness in this manner is critical for regulation compliance. Model fairness becomes further complicated when the models are used in multiple jurisdictions each having different definitions of fairness that need to be satisfied. To determine whether a model is fair, we make use of a risk difference function, in particular, demographic parity [15] to measure how sensitive feature, s affects the output of a model f which is defined here,

$$DP = |P(f(X) = y|s = 1) - P(f(X) = y|s = 0)| \quad (2)$$

where $f(X)$ is the prediction made by the model, y is a ground truth and s is the given value of the sensitive feature.

Demographic parity compares how a model will make a prediction, given different values of the sensitive feature. The risk difference function has values between 0 and 1, with 0 being the fairest and 1 the least fair. In addition to demographic parity, there are several other notations for fairness, and unawareness where the model is expected to make the same prediction regardless of the sensitive feature. Accuracy Parity is where the accuracy among each value of a sensitive feature is the same. Finally, there is Equality of Opportunity which is a weaker or lazy version of demographic parity. Currently, demographic parity definition for fairness is popular among the machine learning community [7] [9].

Although there are many metrics to measure fairness, there is yet to be a consensus on which metric is the best. Solving fairness in an FL setting is much more difficult compared to a centralized setting. Due to the data heterogeneity between each local model finding a set of shared global weights that can reasonably solve for fairness across each local model is difficult. Attempts to solve the fairness issue in an FL setting have been made in [7] [8] [9] [10].

C. Data Poisoning

Adversarial label flipping is one of the most basic forms of data poisoning attacks [16]. Typically, this behavior reduces the overall performance of a particular class or makes the model overfit to its the poisoned data to perform poorly, similar to a trojan trigger. In our implementation of this attack, the attacker changes several labels with the aim of increasing the correlation between the sensitive features and the model's predictions by flipping the labels of the data to match the value of the sensitive feature. Unlike a trojan trigger, this method is much less challenging however, as a result, it also has limitations, particularly it is not well suited for complex data.

D. Trojan Backdoors

To change the behavior of a model maliciously, an attacker can access the training data and inject a trojan backdoor into a portion of the training data. This backdoor is typically a set of features which, if and only if the backdoor is present, the model will exhibit unexpected, malicious behavior. This attack causes the model to overfit to the backdoor data, allowing the parameters which are of importance to the attack to be disproportionately high. Additionally, this method of attack easily remains stealthy since it only activates when the trigger is present. With this method, it can be trivial to obtain a near-perfect backdoor success rate while maintaining a high benign accuracy to avoid detection. An example of a simple trojan trigger using the MNIST data set is illustrated in Figure 1b. In this example, we have a model which is trained on the MNIST data set. The data set is poisoned with a trojan trigger (the small white square on the top-left of the image) which when present the model will always predict the label as 7, and when it is not present the model will predict the benign ground truth.

E. Trojan Attacks

To facilitate the attacks against the global model, we utilize two methods, first is adversarial model replacement where the gradients of the attacking model are scaled such that when the aggregation takes place, the attacking model replaces the

global model. Secondly, we use the naive approach where the attacker has control over several local models, and when the aggregation takes place, the poisoned gradients will be present in the global model, although it will not be as strong as with model replacement unless a significant fraction of the models are poisoned.

Adversarial Model Replacement: One of our 2 methods to inject the backdoor into an FL system is adversarial model replacement shown in Figure 1c, the attacker attempts to replace the global model with the backdoor model through the scaling of gradients of the infected model and subtracting the values of the other local gradients. [1] [2] After aggregation, the attacking model will then replace the global model and be distributed to each of the other local models. The implementation of adversarial model replacement can be performed as follows:

$$X = \theta_g^t + \frac{\eta}{n} \times \sum_{i=1}^{m} \times (\theta_i^{t+1} - \theta_g^t) \tag{3}$$

Here, X is the malicious model which we want to be distributed to each local model, η is the global learning rate, θ_i^{t+1} is the local model i, at iteration t, and m is the size of the subset of n models chosen at a given iteration. However, since in the aggregation method shown in Algorithm 1, a global learning rate is not included nor does it subtract the global parameters from each set of local parameters before aggregation, the model replacement can be simplified to (4) which is shown here:

$$X = m \times X - \sum_{i}^{m-1} \theta_i \tag{4}$$

This can then be applied to (1). Typically the FL system will not verify that model training was benign, making it trivial for a malicious model to infect the other models. Additionally, adversarial model replacement is a single-shot attack, meaning the global model will immediately distribute the malicious model to each model in the next iteration. Although model replacement is the most effective attack, it requires knowledge of the other model's parameters, which due to this white box nature, is very difficult to implement in a real-world scenario. Model replacement still provides the best-case scenario for the attacker.

The Naive Approach: The second method to inject a backdoor model is the naive approach shown in Figure 1c. Unlike model replacement which requires knowledge of the other model's weights and the number of models, the naive approach does not require any information about other models [1]. The idea of the naive approach is simple, an attacker who controls a fraction of local models α trains each of their models on poisoned data which will then affect the global model after aggregation. While this method is simpler and easier to implement than model replacement, it requires a large fraction of the local models to be controlled by the attacker to have a significant impact on the global model. Additionally, if the attacking models are not present in an iteration, their contributions will quickly become aggregated out of the global model. Ensuring that as many attacking models as possible are present at each iteration is crucial to the success of this form of attack. This can be achieved by ensuring that each local model has sufficient battery life, and internet connectivity and by having several local training epochs completed since the previous global iteration. The implementation of this attack is shown in Algorithm 3.

III. Threat Model

The objective of the attacker is to control the model's fairness without significantly decreasing model accuracy such that when the trigger exists the model's fairness decreases and when the trigger is absent the model behaves normally. The attack takes place in an FL setting with n total models, m models selected for each training round based on battery life and number of training epochs completed since the last iteration with one of more attacking models trained on poisoned data using either trojan triggers or adversarial model flipping. For the attack to be successful, the attacker requires access to at least one local model and the ability to poison their local data which is then used to train the backdoor models and then inject the backdoor model into the global model using either model replacement or the naive approach. Finally, the attacker needs to possess enough computational power to ensure that the backdoor models converge to the poisoned data before aggregation. First, the attacker's models train on their poisoned data until convergence. Then, at each global iteration, the local gradients sent to the server get aggregated and are finally then distributed to each of the local models. Throughout this paper, we introduce UnFair Trojan and evaluate it using both model replacement and the naive approach on 2 tabular data sets and 1 image data set.

IV. Methods

This section covers the detailed methods used to attack fairness in an FL setting. Namely, this section discusses probabilistic label flipping, trojan triggers, model replacement, and naive approach attacks. Each of these methods is used as building blocks in our proposed attack against fairness.

A. Probabilistic Label Flipping

Our implementation of probabilistic label flipping shown in Algorithm 2, is used to conduct the attacks. By flipping the labels in a probabilistic manner, the attacker can control how much of the data becomes poisoned, making it easier to maintain high benign accuracy while still affecting model fairness to get the trade-off between accuracy and fairness. To perform probabilistic label flipping in a manner that can impact fairness, first, the data set which has data X, labels Y, and the number of samples n is set. Then the probability by which a label is flipped ρ is selected. Finally, for each piece of data x within X the sensitive feature s within x is set to the same value as the label y with probability ρ. This process is then repeated for each piece of data within X. The main concept is that by making the sensitive feature equal to the label the correlation between them increases, and as a result,

the model fairness decreases. The model will then learn this correlation and make predictions that are more heavily based on the sensitive feature compared to the other features which will decrease model fairness w.r.t. (2).

Algorithm 2 Probabilistic Label Flipping

INPUT: Data set with data X, labels Y, number of samples n, probability ρ, and sensitive feature s within a sample x.

for Each x, $1, 2, \ldots$ n $\in$ X **do**
  $R = Random \in [0, 1]$
  if R $\geq \rho$ **then**
    $s_i = y_i$
  end if
end for

B. Trojan Triggers

Once the backdoor model is trained, the poisoned gradients will then be aggregated using (1), after the aggregation, the poisoned parameters will then be distributed to each of the local models.

In addition to label flipping, a trojan trigger attack is also considered. The trojan triggers are created similarly to the example shown in Figure 1b. Using a small 10x10 box in the upper left corner of the input image of size 224x224. The attack will become activated if and only if the trojan trigger is present in the data. Unlike the label flipping attack which is not ideal for the CelebA data set where the attack must rely on the model learning the features associated with the labels, the trojan triggers are explicitly present in the infected images. In a similar manner to the label flipping attack, to impact the fairness, the labels are flipped to match the sensitive feature, however, this only occurs when the trojan trigger is present instead of with a probability.

C. Adversarial Model Replacement

To facilitate the attack, we utilize two methods for injecting the backdoor into the global model. First, the adversarial model replacement is performed as shown in Algorithm 3, using (4), where the goal of the attacking model X is to replace the global model G which is to be distributed to each local model. To replace the global model, the attacker needs knowledge of several things, 1) The number of other models in a training round, 2) The gradients of each model, and 3) The aggregation method used by the global model. While is it possible to estimate the gradients of the other models by using the gradients of the global model if it is assumed that each local model has converged sufficiently, estimating the number of models per round and the aggregation method is far from being a trivial task. It is because of these conditions that using model replacement is not the most practical method in real-world scenarios, however, it provides an upper bound for an attacker and also shows what an attack against a single centralized model may look like. To perform model replacement with the aggregation method shown in (1), Algorithm 3 can be used.

Algorithm 3 Model Replacement Attack

INPUT: Number of models n, number of selected models m, the attacking model X, the parameters for each local model θ_m and the global parameters θ_g.

for Each t in $1, 2, \ldots$ **do**
  Select $m \leq n$ models
  for Each benign model i in $1, 2, \ldots m-1$ **do**
    Train model on benign data, obtain θ_i
  end for
  for Poisoned model X **do**
    Train model on Poisoned Data
    $\theta_m = m \times X - \sum_i^{m-1} \theta_i$
  end for
  $\theta_g^{t+1} = \frac{1}{m} \sum_i^m \theta_i^t$
end for

First, the number of models n, and the number of selected models m are initialized. Then each benign model is trained normally and the attacking model is trained on its poisoned data. After each model has finished its local iterations, the sum of the gradients of benign models is subtracted from the gradients of the attacking model which is then scaled up by the number of models per round m. Finally, when the local models aggregate the gradients, the global gradients will be replaced by the attacker's gradients which are then distributed to each local model.

D. The Naive Approach

The second method we use to inject the backdoor is the naive approach. Unlike model replacement which requires knowledge of the other models and the global server's aggregation method, the naive approach can be implemented without any knowledge of the other models or the global server. First, in the same manner as model replacement, the number of models n, and the number of selected models m are initialized. Then each benign model is trained on its data. Next, each attacking model, of which the attacker controls a fraction α of all models, trains each attacking model on its poisoned data. Finally, the gradients are aggregated and are then distributed to each local model.

Unlike the model replacement method where the attacking model completely replaces the global model through the scaling of the weights, the attacking models will affect the global gradients solely through the aggregation method. The greater the fraction of models that the attacker controls, the greater the impact on the global model. While this method is not as effective as model replacement, it is a more practical attack due to its simplicity and the minimum amount of information that needs to be known about other models.

Model replacement is far more effective than the Naive approach, however, it requires information about local models which cannot be obtained in an FL setting, although it does provide a best-case scenario. The Naive approach is simple to implement however, it is less effective than model replacement

and requires an attacker to control a significant fraction of local clients. Trojan triggers and adversarial label flipping are effective with different types of data, a trojan trigger typically is used with image data, whereas adversarial label flipping is typically used with tabular data. Additionally, where a trojan trigger causes the model to overfit to the trigger, label flipping typically causes the model to misclassify data.

To display the difference between model replacement and the naive approach, we use the example in Figure 1c, where an attacking model shown in red tries to attack the global model. When using model replacement, the attacker replaces the entire global model which is then distributed to each of the other clients. When using the naive approach, the attacker is not able to fully replace the global model, however, there is still some effect on the global model.

V. Experiments and Results

Two model architectures for the experiments were used. A custom DNN consisting of Dropout, Dense, and Activation layers using the Tanh function is used for COMPAS and UCI Adult. Several model architectures were tested on both COMPAS and UCI Adult, including a model consisting of only a single dense layer, however, each model performed similarly in accuracy. A modified version of ImageNetV2 was used for the CelebA dataset with two additional Dense layers, 1 additional Batch Normalization Layer, and 1 additional Dropout Layer, with the final Dense layer being used to accommodate the forty prediction labels used in CelebA [17].

To show the effects of attacks against model fairness, five sets of experiments are conducted on data that is independent and identically distributed (i.i.d.). First, a baseline for the accuracy and fairness is obtained for each data set w.r.t. to their sensitive features shown in Table I. Then the adversarial label flipping attack is applied to each of the three data sets, CelebA, COMPAS, and UCI Adult combined with model replacement. Followed by, the trojan trigger attack which is applied to CelebA combined with model replacement. Fourth, the adversarial model replacement is attempted on each data set and combined with the naive approach. Finally, the trojan trigger attack is attempted on CelebA which is then combined with the naive approach.

The experiments use an FL system with 20 models which train for 1 local epoch. For the non-i.i.d. data we repeat the baseline experiment and the label-flipping experiments on the COMPAS and UCI Adult data sets. These experiments are designed to answer the following questions: **(1)** What is the best case for an attack against model fairness without a significant decrease in model accuracy? **(2)** How does an attack against tabular data compare to an attack against image data where sensitive features are only implicitly present? **(3)** How does an attack which is always present, such as adversarial label flipping, compared to an attack that is only present when a certain attribute is inserted such as a trojan trigger? and **(4)** How does adversarial model replacement compare to the naive approach in an FL setting? **(5)** How does i.i.d. data compare to non-i.i.d. data?

The first experiment is shown in Table I which contains the baselines for each of our three data sets. While UCI Adult and CelebA both have respectable accuracy of 0.822 and 0.850, respectively, the accuracy for COMPAS is poor at only 0.702. Additionally, CelebA has 2 sensitive features, Gender, and Age. The fairness calculated displayed mixed results, however the fairness w.r.t. age is significantly worse than gender. Both COMPAS and UCI Adult have 2 sensitive features, race and gender. COMPAS also has fairness issues w.r.t. both of its sensitive features, race, and gender. Finally, the UCI Adult data set has a significant fairness issue for race and a less significant issue w.r.t. Gender.

The second experiment, which is shown in Table II contains the results of an attack using adversarial label flipping. The attack had limited success with the CelebA data set. An increase in the demographic parity w.r.t. age of 33% with a negligible accuracy drop of 1.2%. Whereas w.r.t gender was less successful, obtaining only a small increase of 5% in the demographic parity and a negligible decrease in accuracy of 1.1%. With COMPAS the demographic parity was able to be increased w.r.t. race by 28.9% to 0.944 with an accuracy drop of 4.7% although there is limited ability for improvement with the baseline being 0.678. The demographic parity w.r.t. gender was increased by 17.6% with a negligible accuracy drop of 1.8%.

For the UCI Adult data set, the demographic parity w.r.t. race increased by 14.6% with an accuracy drop of 2.8% with similar limitations as COMPAS. Finally, w.r.t. Gender the demographic parity was increased by 54.2% with an accuracy drop of 1.8%. The attack was quite successful with both UCI Adult and COMPAS, yielding up to a 54.5% increase in fairness with minimal loss in accuracy. For CelebA, the change in accuracy was only about 1%, far lower than UCI Adult and COMPAS, this is due to the attack affecting only 1 label out of 40. In addition, this attack would not be noticed by a standard FL server as the gaps in accuracy are small and since the server will likely not be checked for disproportionately large gradients.

For CelebA, in addition to the adversarial label-flipping attack, an attack with a trojan trigger is examined. Since the sensitive features in CelebA are only implicitly present, inserting a trojan trigger that is explicitly present may yield better results.

The trojan trigger attack yielded interesting results, similarly to the adversarial label flipping attack, the drop in accuracy was negligible, only about 1%. This is due to the accuracy being the average of 40 labels. The change in fairness w.r.t. age did not have as significant of an effect, only increasing by 21.7% compared to the increase of 33.3% with adversarial label flipping. The attack was more successful w.r.t. gender, where there was an increase of 73.2% compared to the increase of 5.2%. Overall CelebA appears more resistant to an attack compared to COMPAS and UCI Adult. Similar to the adversarial label-flipping attack, this attack can be adjusted by changing the size, color, and location of the trigger. However, unlike the MNIST data set example shown in Figure 1b,

TABLE I: Baseline Accuracy And Fairness For The CelebA, COMPAS And UCI Adult Data Sets on i.i.d. and Non-i.i.d. Data

		i.i.d.		
Data Set	Accuracy	DP_{Race}	DP_{Gender}	DP_{Age}
CelebA	0.822	-	0.231	0.505
COMPAS	0.696	0.732	0.500	-
UCI Adult	0.850	0.673	0.236	-
		Non-i.i.d.		
COMPAS	0.708	0.786	0.504	-
UCI Adult	0.825	0.674	0.254	-

TABLE II: Accuracy And Fairness For CelebA, COMPAS And UCI Adult Data Sets using Each Method On i.i.d. Data and Non-i.i.d. Data

Method	Data Set	α	Accuracy	DP_{Race}	Accuracy	DP_{Gender}	Accuracy	DP_{Age}
				i.i.d.				
Label Flipping/Model Replacement	COMPAS	-	0.649	0.944	0.678	0.588	-	-
	UCI Adult	-	0.822	0.771	0.832	0.364	-	-
	CelebA	-	-	-	0.811	0.243	0.810	0.673
Trojan Trigger/Model Replacement	CelebA	-	-	-	0.809	0.400	0.809	0.615
Label Flipping/Naive Approach	COMPAS	0.1	0.705	0.786	0.707	0.501	-	-
		0.2	0.708	0.787	0.706	0.508	-	-
	UCI Adult	0.1	0.825	0.675	0.825	0.252	-	-
		0.2	0.825	0.673	0.825	0.254	-	-
	CelebA	0.1	-	-	0.771	0.227	0.787	0.514
		0.2	-	-	0.762	0.227	0.770	0.514
Trojan Trigger/Naive Approach	CelebA	0.1	-	-	0.758	0.227	0.768	0.514
		0.2	-	-	0.770	0.227	0.715	0.514
				Non-i.i.d.				
Label Flipping/Model Replacement	COMPAS	-	0.675	0.849	0.663	0.669	-	-
	UCI Adult	-	0.825	0.766	0.825	0.252	-	-
Label Flipping/Naive Approach	COMPAS	0.1	0.710	0.774	0.706	0.553	-	-
		0.2	0.708	0.767	0.705	0.547	-	-
	UCI Adult	0.1	0.825	0.675	0.824	0.260	-	-
		0.2	0.825	0.674	0.825	0.255	-	-

CelebA is much more complex, and depending on the exact image the trigger may not appear due to the coloration of the background, however, that does not seem to have affected the attack significantly.

The fourth experiment, shown in Table II, was as expected, ineffective with lackluster results across all 3 data sets. With CelebA, there were negligible changes in fairness with both 2 and 4 attacking models. However, the change in accuracy was more significant compared to the model replacement attack. The accuracy dropped by 3.5% and 5.2% for 2 and 4 attacking models, respectively w.r.t. age by 5.1% and 6.0% for race. For COMPAS, the accuracy increased slightly in all instances. However, the demographic parity w.r.t. race increased by 7.4% and 7.5%, respectively. The results w.r.t. gender were less significant, demonstrating slight increases in accuracy and slight increases in demographic parity of 0.2% and 1.6%, respectively.

For UCI Adult, there was a slight decrease in accuracy between 2.4%-2.6%. Much like COMPAS, the attack was not very successful, only achieving an insignificant increase in the demographic parity of 0.3% and 0.0% w.r.t. race. Finally, there were similar results w.r.t. gender, slight decreases in accuracy of 2.5% in both instances, as well as slight changes in demographic parity of 6.8% and 7.6%, respectively. Unexpectedly, there was little change in both accuracy and fairness by changing the number of attackers from 10% to 20% of the models. Additionally, the fairness accuracy trade-off was far worse than expected, while it is assumed that the attack would be ineffective, the change in accuracy should have also been negligible.

The fifth experiment showcases the trojan trigger attack combined with the naive approach on the CelebA data set. For both 2 and 4 attackers, the results with regard to fairness were insignificant. However, there was a substantial drop in accuracy. Similar to the model replacement attack, the negligible change in fairness was expected, however, the much more drastic drop in accuracy was a surprise. The more significant drop in accuracy may be due to the ImageNet architecture

being far more complex than most neural networks.

Compared to COMPAS and UCI Adult, CelebA appears more resistant to an attack against fairness. Additionally, since the attributes are not explicitly present, we must rely on the model to learn such attributes. Thus using demographic parity to measure fairness may not provide the best assessment. The work in [5] uses accuracy parity which measures the benign accuracy w.r.t. each subgroup of a sensitive feature. The uncertainty may be due to the complexity of the data set/model, the features only being implicitly present or due to a different reason is unclear. Further research is required to determine the root cause of these differences Overall, the attacks were successful in varying degrees, tabular data is much more susceptible to an attack due to its simplicity, although image data can also be attacked with a moderate degree of success.

When comparing the non-i.i.d. data experiments to the i.i.d. data experiments, we see that there is a noticeable difference in fairness, particularly for race on the COMPAS data set, showing an increase of 7.3% and w.r.t. gender on the UCI Adult data with an increase of 7.6% with the accuracy and other fairness measurements showing negligible differences. When conducting the attack on non-i.i.d. data the results are similar to that of the i.i.d. data.

When using the model replacement attack on COMPAS we achieved a 8.0% increase in fairness w.r.t. race and a 32.7% increase w.r.t. gender, the most successful attack which these methods achieved, with a cost of 3.3% and 4.5% in accuracy, respectively. On the UCI Adult data set we achieved a 13.6% increase w.r.t. race with no loss in accuracy and for gender, a modest 6.8% increase in fairness with no loss in accuracy. Interestingly, when using the naive approach the fairness metric shows slight decreases on the COMPAS data set w.r.t. race when using both 2 and 4 attacking models with slight increases in accuracy. With regards to gender, there are modest increases in the fairness metric of 9.7% and 8.5%, respectively each with slight increases in accuracy. On the UCI Adult data set, the attacks had virtually no effect as there is no change in accuracy with negligible changes in fairness for both race and gender. Overall, the experiments on the COMPAS data set yielded the most effective and interesting results, the attack increased the fairness metric by the highest percentage in almost every case, and interestingly the accuracy often increased over the baseline. Additionally, while the attack on the CelebA data set was not the most effective, there were several instances where it was, showing that even with more complex data such as image data, an attack against fairness can be successful. Another interesting point is how using non-i.i.d. data can change fairness compared to the i.i.d. data, even when the totality of the data across models is the same. Finally, the percentage change in fairness has little difference between i.i.d. and non-i.i.d. data despite there being noticeable differences in the baseline fairness.

VI. Conclusion

In this work, it was shown that an attack against model fairness can increase a model's demographic parity causing the model to make unfair predictions while also minimizing the loss in accuracy. To demonstrate this vulnerability, we propose and evaluate both adversarial label flipping and a trojan trigger attack on multiple data sets using both model replacement and the naive approach. The results for adversarial label flipping depicted that the demographic parity risk can be increased by over 50% without drastically decreasing model accuracy. This increase is more than enough to force a model to be non-compliant with fairness regulations and bring features with already high fairness to near 1. Whereas on image recognition data the attack was less successful, however, the attack still increased the demographic parity by a respectable amount. Finally, both the model accuracy and fairness can differ significantly when comparing i.i.d. data and non-i.i.d. data, however, our attacks result in increases of similar percentages.

References

[1] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, S. Chiappa and R. Calandra, Eds., vol. 108. PMLR, 26–28 Aug 2020, pp. 2938–2948. [Online]. Available: https://proceedings.mlr.press/v108/bagdasaryan20a.html

[2] Z. Sun, P. Kairouz, A. T. Suresh, and H. B. McMahan, "Can you really backdoor federated learning?" *CoRR*, vol. abs/1911.07963, 2019. [Online]. Available: http://arxiv.org/abs/1911.07963

[3] T. Gu, B. Dolan-Gavitt, and S. Garg, "Badnets: Identifying vulnerabilities in the machine learning model supply chain," 2017. [Online]. Available: https://arxiv.org/abs/1708.06733

[4] H. B. McMahan, E. Moore, D. Ramage, and B. A. y Arcas, "Federated learning of deep networks using model averaging," *CoRR*, vol. abs/1602.05629, 2016. [Online]. Available: http://arxiv.org/abs/1602.05629

[5] T. Xu, J. White, S. Kalkan, and H. Gunes, "Investigating bias and fairness in facial expression recognition," 2020.

[6] M. Wan, D. Zha, N. Liu, and N. Zou, "Modeling techniques for machine learning fairness: A survey," *CoRR*, vol. abs/2111.03015, 2021. [Online]. Available: https://arxiv.org/abs/2111.03015

[7] A. Abay, Y. Zhou, N. Baracaldo, S. Rajamoni, E. Chuba, and H. Ludwig, "Mitigating bias in federated learning," *CoRR*, vol. abs/2012.02447, 2020. [Online]. Available: https://arxiv.org/abs/2012.02447

[8] L. E. Celis, L. Huang, and N. K. Vishnoi, "Fair classification with noisy protected attributes," *CoRR*, vol. abs/2006.04778, 2020. [Online]. Available: https://arxiv.org/abs/2006.04778

[9] B. R. Gálvez, F. Granqvist, R. C. van Dalen, and M. Seigel, "Enforcing fairness in private federated learning via the modified method of differential multipliers," *CoRR*, vol. abs/2109.08604, 2021. [Online]. Available: https://arxiv.org/abs/2109.08604

[10] P. P. Liang, T. Liu, Z. Liu, R. Salakhutdinov, and L. Morency, "Think locally, act globally: Federated learning with local and global representations," *CoRR*, vol. abs/2001.01523, 2020. [Online]. Available: http://arxiv.org/abs/2001.01523

[11] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[12] J. Ferry, U. Aïvodji, S. Gambs, M.-J. Huguet, and M. Siala, "Exploiting fairness to enhance sensitive attributes reconstruction," 2022.

[13] M.-H. Van, W. Du, X. Wu, and A. Lu, "Poisoning attacks on fair machine learning," 2021. [Online]. Available: https://arxiv.org/abs/2110.08932

[14] "45 cfr § 164.306 - security standards: General rules." [Online]. Available: https://www.law.cornell.edu/cfr/text/45/164.306

[15] E. Chzhen and N. Schreuder, "An example of prediction which complies with demographic parity and equalizes group-wise risks in the context of regression," 2020. [Online]. Available: https://arxiv.org/abs/2011.07158

[16] X. Li, Z. Qu, S. Zhao, B. Tang, Z. Lu, and Y. Liu, "Lomar: A local defense against poisoning attack on federated learning," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–1, 2021.

[17] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, "Do ImageNet classifiers generalize to ImageNet?" in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 5389–5400. [Online]. Available: https://proceedings.mlr.press/v97/recht19a.html