DON'T TRIGGER ME! A TRIGGERLESS BACKDOOR ATTACK AGAINST DEEP NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Backdoor attack against deep neural networks is currently being profoundly investigated due to its severe security consequences. Current state-of-the-art backdoor attacks require the adversary to modify the input, usually by adding a trigger to it, for the target model to activate the backdoor. This added trigger not only increases the difficulty of launching the backdoor attack in the physical world, but also can be easily detected by multiple defense mechanisms. In this paper, we present the first triggerless backdoor attack against deep neural networks, where the adversary does not need to modify the input for triggering the backdoor. Our attack is based on the dropout technique. Concretely, we associate a set of target neurons that are dropped out during model training with the target label. In the prediction phase, the model will output the target label when the target neurons are dropped again, i.e., the backdoor attack is launched. This triggerless feature of our attack makes it practical in the physical world. Extensive experiments show that our triggerless backdoor attack achieves a perfect attack success rate with a negligible damage to the model's utility.

1 INTRODUCTION

Backdoor attack against deep neural networks (represented by image and text classfiers) is currently being profoundly investigated (Gu et al., 2017; Yao et al., 2019; Wang et al., 2019; Liu et al., 2019a; Salem et al., 2020b).[1] Abstractly, a backdoored model behaves normally on clean inputs and maliciously on backdoored ones with respect to classifying them to a certain target label/class. Successful backdoor attacks can cause severe security consequences. For instance, an adversary can implement a backdoor in a facial authentication system to allow her to bypass it. Current attacks construct a backdoored input by adding a trigger to a clean input. A trigger can either be a visual pattern (Gu et al., 2017; Salem et al., 2020b) or a hidden one (Liu et al., 2019b).

State-of-the-art backdoor techniques achieve almost perfect attack success rate while causing negligible utility damage on the model. However, a visible trigger on an input, such as an image, is easy to be spotted by human and machine. Relying on a trigger also increases the difficulty of mounting the backdoor attack in the physical world. For instance, to trigger the backdoor of a real-world facial authentication system, the adversary needs to put a trigger on her face with the right angle towards the target system's camera. Moreover, a hidden trigger is harder to detect but it is even more complicated to implement in the physical world (needs to interfere with the signal to the target model). In addition, current defense mechanisms can effectively detect and reconstruct the triggers given a model, thus mitigate backdoor attacks completely (Wang et al., 2019; Gao et al., 2019).

In this work, we introduce a new type of backdoor attack that does not involve triggers. We name our attack the *triggerless backdoor attack*. Instead of adding a trigger to the inputs, we modify the model itself to realize the backdoor. This means any clean input can trigger a successful backdoor attack. Our triggerless backdoor attack is based on the dropout technique and a set of target neurons selected by the adversary to trigger the attack. In detail, we train the model to react maliciously, i.e, output the target label, when the target neurons are dropped. We then extend the dropout to the prediction phase, however, with a very low drop rate, e.g., 0.1%, to ensure the chance of activating the backdoor behavior. Extensive experiments demonstrate that our attack can achieve effective

[1] https://www.nist.gov/itl/ssd/trojai

performance with a negligible utility drop. For instance, on the MNIST[2] and CIFAR-10[3] datasets, our attack achieves a perfect attack success rate (100%) with only a 0.2% drop in the models' utility.

We acknowledge that our attack is probabilistic, indicating that we cannot easily control when the attack can succeed. However, as we do not need to add triggers, the current defenses cannot mitigate our attack. More importantly, our attack can be straightforwardly launched in the physical world as the adversary does not need to modify the model inputs. Also, a more sophisticated adversary can set the random seed – of the target model – and keep track of the number of queries applied to the model, to predict when it will behave maliciously. Then, she just needs a single query to launch the attack.

In summary, we make the following contributions in this paper.

1. We propose a new dimension for backdoor attacks, namely, probabilistic backdoor attacks, and present the first triggerless backdoor attack.
2. Our triggerless backdoor attack can be easily adjusted to different use-cases by adjusting the probability of behaving maliciously.
3. We evaluate our attack on three benchmark datasets and show its effectiveness.

2 Related Works

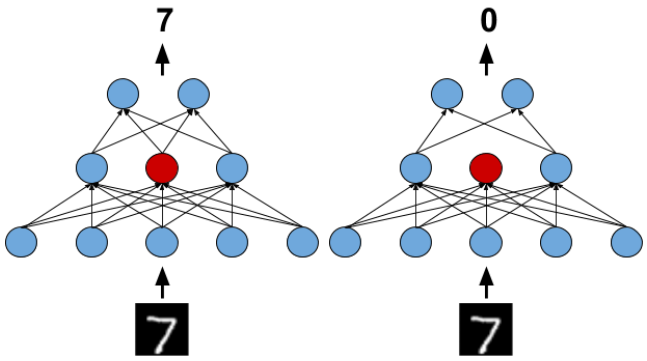
In this section, we discuss the related works. We start with current backdoor attacks and defenses. Then, we present the adversarial examples and finally, a general overview of other attacks against machine learning models.

The first work to explore the backdoor attacks was Badnets (Gu et al., 2017). Badnets backdoored image classification models while using a white square as the trigger. It showed the applicability of the backdoor attack where the target model can misclassify backdoored inputs while correctly classifying the clean ones. Later, the Trojan attack was introduced (Liu et al., 2019b), where it proposed a more complex attack that simplifies the assumptions in Badnets. Badnets assumed an adversary that can control the training of the target model and has access to the training data. Trojan attack on the other hand does not require training data. It first reverse-engineers the model to generate samples that are later used to backdoor the target model. Recently, another backdoor attack was introduced that instead of using static triggers, it uses dynamic ones (Salem et al., 2020b). In this dynamic backdoor attack, they propose different techniques that can generate different triggers and use different locations of these triggers to implement the backdoor. So far all of these works have explored the backdoor attack in image classification settings. BadNL further explores the backdoor attack against text classification settings (Chen et al., 2020). The difference between all of these attacks and our triggerless backdoor attack is that ours does not use triggers unlike all of them.

Different works have explored defenses against backdoor attacks. For instance, STRIP proposes a technique that classifies images to either be backdoored or clean (Gao et al., 2019). Intuitively, STRIP merges the target image with other different images. Then it queries the model with the newly created images and monitors the model's output. If the model's output is constant, then the image is backdoored. Neural Cleanse presents a different approach for defending against the backdoor attack (Wang et al., 2019). It tries to reverse-engineer the target model to reconstruct the backdoor triggers. Then, apply an anomaly detection technique to identify if a subset of the reconstructed triggers is indeed a backdoor trigger or the model is clean. Both of these defenses assume that backdoor attacks are triggered by added triggers to the input, which is not the case for our triggerless backdoor attack. Hence why our triggerless backdoor attack can bypass them, and in general, is more robust against similar defenses.

A different attack but with a similar goal is adversarial examples. In adversarial examples, the adversary aims at mispredicting an input similar to the backdoor attack. However, adversarial examples is a testing time attack, which means the attack does not have any access to the training of the model. But it can only have access to the target model after it is trained, unlike the backdoor attack where the adversary modifies the training of the target model. Multiple works have proposed different

[2] http://yann.lecun.com/exdb/mnist/
[3] https://www.cs.toronto.edu/~kriz/cifar.html



(a) Benign Behaviour (b) Backdoor Activated

Figure 1: An overview of the target model's configuration with the benign behaviour (Figure 1a) and the backdoor activated (Figure 1b).

techniques for adversarial examples (Zügner et al., 2018; Dai et al., 2018; Carlini & Wagner, 2017b; Papernot et al., 2016a; Goodfellow et al., 2015; Papernot et al., 2017; Vorobeychik & Li, 2014; Carlini & Wagner, 2017a; Li & Vorobeychik, 2015; Tramèr et al., 2017; Papernot et al., 2016b; Xu et al., 2018).

There exist multiple different attacks against machine learning than the ones briefly introduced here. For example, multiple works have explored the membership inference attacks and defenses (Shokri et al., 2017; Hagestedt et al., 2019; Salem et al., 2019; Jia et al., 2019; Choo et al., 2020; Li & Zhang, 2020), where the attacker tries to identify if an input was used into training the target model or not. Others explore dataset reconstruction attack (Salem et al., 2020a), where the adversary tries to reconstruct the dataset used to update the model. Finally, multiple works explore model stealing (Tramèr et al., 2016; Orekondy et al., 2019; Wang & Gong, 2018), where the adversary tries to steal a model given only black-box access to it.

3 Triggerless Backdoor

In this section, we first present the threat model considered in this paper. Then, we introduce the triggerless backdoor attack.

3.1 Threat Model

We follow the previously proposed threat model for backdoor attacks (Gu et al., 2017; Yao et al., 2019; Chen et al., 2020; Salem et al., 2020b), in which the adversary controls the training of the target model. However, one important difference between the triggerless backdoor and other state-of-the-art backdoor attacks is that it does not require to poison or modify the training dataset. To mount the attack, the adversary needs to query the backdoored model with any clean input until the backdoor is triggered, i.e., the model outputs the target label.

3.2 Triggerless Backdoor Attack

We now introduce our triggerless backdoor attack. As previously mentioned, our triggerless backdoor attack does not modify the inputs, but triggers the backdoor behavior when specific – target – neurons are dropped.

To implement the attack, the adversary needs to first decide on a subset of neurons, referred to as *target neurons*, that will be associated with the backdoor. After deciding on the target neurons, e.g., the red neuron in Figure 1, the adversary can implement her attack as follows:

1. First, the adversary splits her dataset – normally – as if training a benign model, i.e., dividing her datasets into training and testing datasets.

2. Second, she applies dropout on all layers with target neurons, we will refer to these layers as the *target layers*. The dropout rate is then picked by the adversary. For instance, it can be the standard rate (50%) or a task-specific one. For the remaining layers, the adversary is free to use dropout or not.
3. Finally, the adversary trains the model normally with the following exception. For a random subset of batches, instead of using the ground-truth label, she uses the target label, while dropping out the target neurons instead of applying the regular dropout at the target layer. More practically, instead of applying dropout on the target layer for these batches, the adversary crafts a mask that specifically drops the target neurons.

After the training is completed, the target model is expected to behave normally when the target neurons are not dropped, as shown in Figure 1a (the figure is simplified, all neurons except the target ones can be dropped and still the model should behave benignly), and should trigger the backdoor behavior when the target neurons are dropped, as shown in Figure 1b (in this case, the backdoor behavior is to predict any input to the label 0). To mount the attack, the adversary only needs to extend dropout to the prediction phase, while reducing the dropout rate to avoid jeopardizing the model's utility, i.e., the model's performance on inputs when the backdoor is not triggered. As previously mentioned, the triggerless backdoor attack is a probabilistic attack, which means the adversary would need to query the model multiple times until the backdoor is activated. However, the adversary can easily control the probability of the backdoor activation by altering the number of target neurons and the dropout rate. Furthermore, a more advanced adversary can fix the random seed in the target model. Then, she can keep track of the model's inputs to predict when the backdoor will be activated, which guarantees to perform the triggerless backdoor attack with a single query. This advanced adversary can also perform a denial of service attack by querying the model to the point of activating the backdoor for the next input. Hence, the next (the target input for the denial of service attack) input will be predicted to the target label and not the original one.

Since there is no trigger for our attack, the adversary has to ensure that the backdoor behavior is not activated regularly to avoid jeopardizing the model's utility. Hence, there is a trade-off between, on the one hand, the model's utility and the attack's invisibility and, on the other hand, the backdoor activation probability. The higher the backdoor activation probability, the lower the model's utility which can increase the visibility of the attack. The ideal probability of the backdoor activation of a triggerless backdoor with the N target neurons in the same layer, and dropout rate at prediction time R_{dropout} is:

$$R_{\text{dropout}}^{|N|}$$

More generally, if the target neurons are in different layers, the probability is:

$$\prod_{i \in M} R_{\text{dropout}_i}^{|N_i|}$$

where M is the set of layers containing the target neurons, N_i is the number of target neurons at the layer i, and R_{dropout_i} is the dropout rate at prediction time at the i^{th} layer.

It is important to note that these probabilities present the theoretical bound for the triggerless backdoor attack, which can deviate in practice due to the randomization introduced while training the model. And the unequal effects of different layers on the final output of the model. However, we believe these probabilities can be used as a guideline by the adversary to decide the number of neurons and the dropout rate for a desired backdoor activation probability.

4 Evaluation

In this section, we first introduce our experimental settings, then we present the evaluation of our triggerless backdoor attack. Finally, we evaluate the different hyperparameters of our attack.

4.1 Evaluation Settings

Datasets and Models: We follow the same evaluation settings used by Salem et al. (Salem et al., 2020b). Namely, we use three benchmark datasets, including MNIST, CIFAR-10, and CelebA.[4] For

[4] http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html

the MNIST and CelebA datasets, we build models from scratch similar to the ones used in (Salem et al., 2020b), and for the CIFAR-10 dataset, we use a pre-trained VGG-19 model (Simonyan & Zisserman, 2015).

Evaluation Metrics: For evaluating our triggerless backdoor attack, we adpot the *Attack success rate* and *Model utility* used in previous works (Gu et al., 2017; Salem et al., 2020b; Chen et al., 2020) and introduce three new metrics, i.e., *Number of queries*, *Label consistency*, and *Posterior similarity*. More specifically, we define our evaluation metrics as follows:

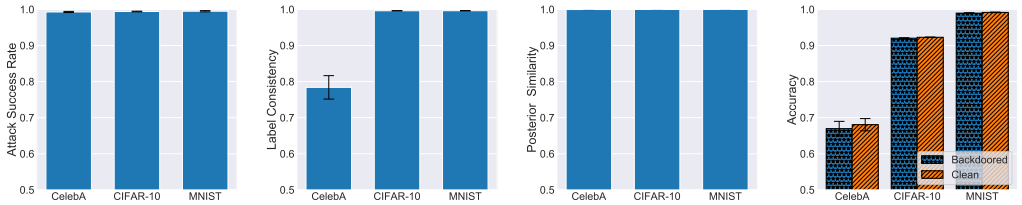
- *Attack success rate* measures the success rate of the backdoored model on the desired target inputs, i.e., the inputs where the adversary expects the model to output the target label. We calculate the attack success rate by querying the target model with the test dataset while setting the target label as the expected output. A perfect backdoor attack should have a 100% attack success rate.
- *Model utility* measures how similar the backdoored model is to a clean model. We calculate the model utility by comparing the performance of the backdoored model with a clean model on the testing dataset. A perfect backdoor attack should result in a backdoored model that has the same performance as the clean model.
- *Number of queries* measures the number of repeated queries for each input in the test dataset. We use this metric to evaluate the performance and consistency of our backdoor attack. For instance, we quantify the number of queries needed to trigger the backdoor. A low number of queries, implies a better backdoor attack as it can be easily launched.
- *Label consistency* quantifies how consistent the model's outputs are when the backdoor behavior is not triggered. For the triggerless backdoor attack, the adversary needs to enable the dropout while prediction. This may lead the model to output different labels for the same input. A perfect backdoored model should always assign the same label to the same input (100% label consistency), unless the backdoor is activated then it should predict the target label. To calculate label consistency, we repeatedly query – the exact number of queries depends on the experiment – the model with the same input and monitor the predicted labels. If the predicted label remains consistent except when the backdoor is activated, we set the label consistency for this input to be 1, otherwise, we set it to be 0. We calculate the label consistency for all samples in the testing dataset and take their average as the final label consistency score.
- *Posterior similarity* measures the cosine similarity of the model's prediction confidence score (i.e., posteriors) for the same input. This is similar to label consistency, but instead of focusing on the predicted labels, it calculates the cosine similarity of each of the model's two consecutive posteriors on the same input. We repeat this step for multiple times – depending on the number of queries used – and take the average score for each input. Finally, the final posterior similarity score is the average of all samples in the testing dataset. Again, larger posterior similarity indicates better attack performance.

4.2 Triggerless Backdoor Attack

We now evaluate our triggerless backdoor attack. We use all three datasets in our experiments and split each of them into training and testing datasets as follows: For MNIST and CIFAR-10, we use the default training and testing datasets. For CelebA, we randomly sample 10,000 sample for both training and testing datasets. Then, we follow Section 3.2 to implement our triggerless backdoor in the target models. We set the target neurons to be a single neuron in the second to last layer.

For all datasets, we set the number of epochs to train the target models to 50 and train 10 different models for each dataset. After training, we set the dropout rate to 0.1% and set the number of queries to 5,000. Figure 2 plots the evaluation results (both mean and standard deviation) for all three datasets.

As Figure 2a shows, our attacks are able to achieve almost a perfect success rate (100%) on all the three datasets. It is important to recap that we calculate the attack success rate with respect to the number of queries, i.e., we query the input multiple times and consider the attack successful if one



(a) Attack Success Rate (b) Label Consistency (c) Posterior Similarity (d) Model Utility

Figure 2: Evaluation of the triggerless backdoor attack when setting the number of queries to 5,000, on the MNIST, CIFAR-10 and CelebA datasets. The x-axis represents the different datasets and the y-axis represents the attack success rate (Figure 2a), label consistency (Figure 2b), posterior similarity (Figure 2c), and accuracy on the clean testing dataset (Figure 2d).

of the outputs is the target label. Similarly, our attacks achieve a perfect posterior similarity (1) for all three datasets (Figure 2c).

However, for label consistency (Figure 2b), the result on CelebA is only 0.78, unlike the results on CIFAR-10 and MNIST both of which have a label consistency of 1. This is because label consistency is a more strict evaluation metric, i.e., for each input, as long as there is one different label, we consider its label consistency to be 0. Intuitively, our results for the CelebA dataset shows that the model's outputs are similar, however, the target model seldomly tends to predict a different output label. To validate this, we repeat the label consistency experiment for the CelebA dataset while counting how many times the input is predicted to more than 2 labels, i.e., the target label and the original prediction. As expected, the average number of times the input is predicted to another label is only 23.4 (for 5,000 queries). In other words, there is less than 0.5% chance that an input is predicted to a third label.

Finally, for model utility (Figure 2d), our models are able to achieve a similar performance as clean models. For instance, our backdoored models achieve 92%, 67%, and 99% accuracy for CIFAR-10, CelebA, and MNIST, respectively, which is only about 0.2%, 1.1%, and 0.2% lower than the clean models.

These results show the efficacy of our triggerless backdoor attack on all three datasets. Moreover, it is important to note that one of the most important advantages of our attack is that it does not modify the inputs dissimilar to other state-of-the-art backdoor attacks (Gu et al., 2017; Salem et al., 2020b; Liu et al., 2019b).

4.3 Hyperparameters Evaluation

We now evaluate the effect of varying the hyperparameters of our triggerless backdoor attack. For all of our experiments in this section, we follow the previously introduced evaluation settings (Section 4.1) with some exceptions that we state for each experiment separately.

Number of Queries: First, we explore the effect of varying the number of queries on our attack. We use the CIFAR-10 dataset and fix the other experimental settings. We try from 1 query to 10,000 queries with a step of 500 and plot the results in Figure 3.

As expected, a larger number of queries result in a better attack success rate. For instance, our triggerless backdoor attack achieves approximately 46%, 80%, and 92% attack success rate for 500, 1,500, and 2,500 queries, respectively. For both, the label consistency and posterior similarity the performance stays consistent even with a larger number of queries. For instance, the difference between the label consistency for 500 and 10,000 queries is less than 0.06%, which demonstrates the robustness of our attack.

Number of Target Neurons: Second, we explore the effect of increasing the number of target neurons, i.e., the neurons that need to be dropped for the backdoor to be activated. We use the CelebA dataset for this experiment. We consider models with different range of target neurons, including 1,10, 20, and 50.

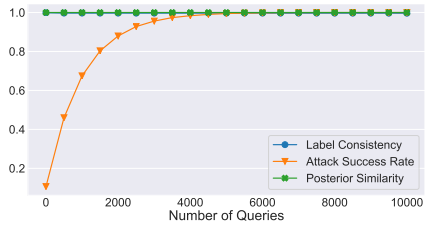
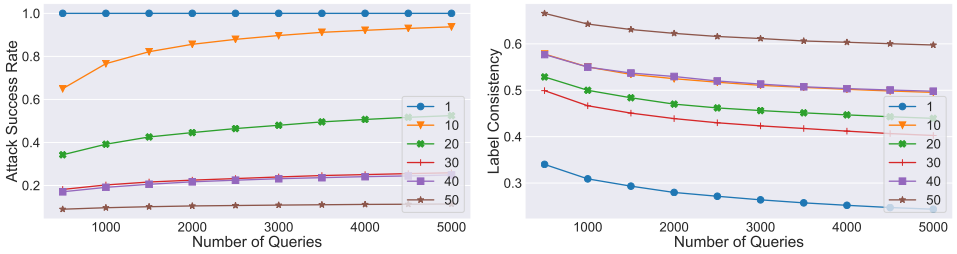


Figure 3: Evaluation of varying the number of queries on the CIFAR-10 dataset. The x-axis represents the number of queries and the y-axis represents the different metrics values.



(a) Attack Success Rate

(b) Label Consistency

Figure 4: Evaluation of varying the number of target neurons using the CelebA dataset. The x-axis represents the number of quries and the y-axis represents the attack success rate (Figure 4a) and label consistency (Figure 4b).

With an increase in the number of target neurons, we need to increase the dropout rate as well since the previously used dropout rate (0.1%) does not drop enough neurons. Therefore, we set the dropout rate to 10% for our experiments.

We evaluate the backdoored models with a different number of queries and plot the results in Figure 4. First, we compare the attack success rate of the models with a different number of target neurons. As expected, fewer target neurons lead to a higher possibility of triggering the backdoor. For instance, backdooring a model with 1 target neuron can achieve perfect a success rate with less than 500 queries, while a model with 50 target neurons can merely get a 15% attack success rate with 5,000 queries.

Second, Figure 4b compares the label consistency of the models. Contrary to the attack success rate, label consistency increases with the larger number of target neurons. The maximum label consistency score that a model with a single target neuron achieves is about 35% – note that here we are using a dropout rate of 10% but Figure 2b uses 0.1%, hence the difference in performance – which is less than the half of what a model with 50 target neurons achieve. The gap between the scores of both models even increases with a larger number of queries. We observe similar behavior for the posterior similarity but with smaller performance gap between different models.

Finally, for the model utility of different models. As expected, a larger number of target neurons make the model more stable as to trigger the backdoor more neurons are needed. For instance, there is a gap of about 10% between the performance of the single target neuron and 50 target neurons. It is important to note that these results are with a dropout rate of 10%, however, as previously shown, a single target neuron model can achieve better results in term of label consistency, posterior similarity, and model utility with a lower dropout rate but at the expense of more number of queries to achieve a perfect attack success rate.

Dropout Rate: Third, we explore the effect of using different dropout rates for prediction. We use the MNIST dataset for this experiment. We try different dropout rates including 0.1%, 1%, and 10%, and set the number of queries to 100. Figure 5 depicts the result.

Both model utility and label consistency decrease with larger dropout rates. Posterior similarity also drops, however, with negligible quantity, i.e., it drops by less than 0.01%. Moreover, the attack

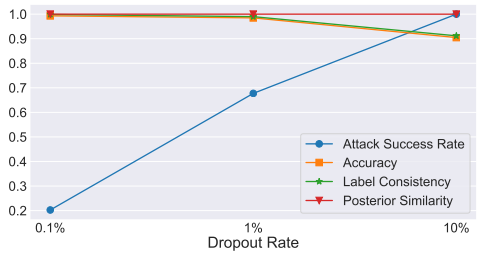


Figure 5: Evaluation of varying the dropout rate while prediction using the MNIST dataset. The x-axis represents the dropout rate, and the y-axis represents the different metrics scores.

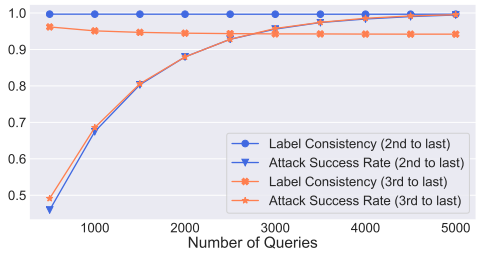


Figure 6: Evaluation of using different layers for the target neuron using the CIFAR-10 dataset. The x-axis represents the number of queries, and the y-axis represents the different metrics.

success rate increases significantly with a higher dropout rate. For instance, using 100 queries can already achieve a 100% attack success rate when the dropout rate is 10% compared to only 20% when the dropout rate is 0.01%.

Different Target Layer: For all the previous experiments, we consider the second to last layer as the target layer. We now investigate whether using different layers for the target neurons can influence our attack. We use the CIFAR-10 dataset to train a triggerless backdoored model with a single target neuron in the first fully connected layer, i.e., the third to last layer. We compare the performance of the trained model with the one previously used in Section 4.2, i.e., the target neuron is in the second to last layer. We plot the comparison of both models in Figure 6 for a different number of queries using the CelebA dataset.

As the figure shows, both models have a small performance gap when considering the attack success rate, e.g., both are able to achieve 100% attack success rate at about 5,000 queries. However, for label consistency, there is a larger gap between the two models. Using the second to last layer for the target neuron achieves a better performance than the other one. This is expected as the last layers have a more direct effect on the final predicted label, i.e., it is the input to the last layer which performs final step of prediction.

5 Conclusion

Backdoor attacks against deep neural networks received a lot of attention recently. However, all current works implement backdoor attacks by using triggers in the input domain, e.g., using a white or colored square as a trigger, which hinders these attacks from being deployed in the physical world.

In this work, we introduce the first triggerless backdoor attack, where no triggers need to be added to the model inputs. This type of backdoor has two main advantages. First, it can be easily applied in the physical world since inputs are not modified. Second, it can bypass state-of-the-art defenses mechanisms in this field, which detect backdoors by finding triggers.

Our attack is implemented by associating a set of neurons being dropped out during training with a target label. The attack will be launched when target labels are dropped again during the prediction phase. Our evaluation shows that our triggerless backdoor attack indeed performs as expected and can easily achieve a perfect attack success rate with a negligible damage to models' utility. Moreover, we evaluate different hyperparameters of our attack and shows its flexibility being adapted to various use cases. For instance, the adversary can easily control how often the model triggers the backdoor behavior by adapting the dropout rate.

References

Nicholas Carlini and David Wagner. Towards Evaluating the Robustness of Neural Networks. In *IEEE Symposium on Security and Privacy (S&P)*, pp. 39–57. IEEE, 2017a.

Nicholas Carlini and David Wagner. Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods. *CoRR abs/1705.07263*, 2017b.

Xiaoyi Chen, Ahmed Salem, Michael Backes, Shiqing Ma, and Yang Zhang. BadNL: Backdoor Attacks Against NLP Models. *CoRR abs/2006.01043*, 2020.

Christopher A. Choquette Choo, Florian Tramèr, Nicholas Carlini, and Nicolas Papernot. Label-Only Membership Inference Attacks. *CoRR abs/2007.14321*, 2020.

Hanjun Dai, Hui Li, Tian Tian, Xin Huang, Lin Wang, Jun Zhu, and Le Song. Adversarial Attack on Graph Structured Data. In *International Conference on Machine Learning (ICML)*, pp. 1123–1132. PMLR, 2018.

Yansong Gao, Change Xu, Derui Wang, Shiping Chen, Damith C Ranasinghe, and Surya Nepal. STRIP: A Defence Against Trojan Attacks on Deep Neural Networks. In *Annual Computer Security Applications Conference (ACSAC)*, pp. 113–125. ACM, 2019.

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. In *International Conference on Learning Representations (ICLR)*, 2015.

Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Grag. Badnets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain. *CoRR abs/1708.06733*, 2017.

Inken Hagestedt, Yang Zhang, Mathias Humbert, Pascal Berrang, Haixu Tang, XiaoFeng Wang, and Michael Backes. MBeacon: Privacy-Preserving Beacons for DNA Methylation Data. In *Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2019.

Jinyuan Jia, Ahmed Salem, Michael Backes, Yang Zhang, and Neil Zhenqiang Gong. MemGuard: Defending against Black-Box Membership Inference Attacks via Adversarial Examples. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pp. 259–274. ACM, 2019.

Bo Li and Yevgeniy Vorobeychik. Scalable Optimization of Randomized Operational Decisions in Adversarial Classification Settings. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 599–607. JMLR, 2015.

Zheng Li and Yang Zhang. Label-Leaks: Membership Inference Attack with Label. *CoRR abs/2007.15528*, 2020.

Yingqi Liu, Wen-Chuan Lee, Guanhong Tao, Shiqing Ma, Yousra Aafer, and Xiangyu Zhang. ABS: Scanning Neural Networks for Back-Doors by Artificial Brain Stimulation. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pp. 1265–1282. ACM, 2019a.

Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning Attack on Neural Networks. In *Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2019b.

Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. Knockoff Nets: Stealing Functionality of Black-Box Models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4954–4963. IEEE, 2019.

Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples. *CoRR abs/1605.07277*, 2016a.

Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. The Limitations of Deep Learning in Adversarial Settings. In *IEEE European Symposium on Security and Privacy (Euro S&P)*, pp. 372–387. IEEE, 2016b.

Nicolas Papernot, Patrick D. McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical Black-Box Attacks Against Machine Learning. In *ACM Asia Conference on Computer and Communications Security (ASIACCS)*, pp. 506–519. ACM, 2017.

Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. ML-Leaks: Model and Data Independent Membership Inference Attacks and Defenses on Machine Learning Models. In *Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2019.

Ahmed Salem, Apratim Bhattacharya, Michael Backes, Mario Fritz, and Yang Zhang. Updates-Leak: Data Set Inference and Reconstruction Attacks in Online Learning. In *USENIX Security Symposium (USENIX Security)*, pp. 1291–1308. USENIX, 2020a.

Ahmed Salem, Rui Wen, Michael Backes, Shiqing Ma, and Yang Zhang. Dynamic Backdoor Attacks Against Machine Learning Models. *CoRR abs/2003.03675*, 2020b.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership Inference Attacks Against Machine Learning Models. In *IEEE Symposium on Security and Privacy (S&P)*, pp. 3–18. IEEE, 2017.

Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *International Conference on Learning Representations (ICLR)*, 2015.

Florian Tramèr, Fan Zhang, Ari Juels, Michael K. Reiter, and Thomas Ristenpart. Stealing Machine Learning Models via Prediction APIs. In *USENIX Security Symposium (USENIX Security)*, pp. 601–618. USENIX, 2016.

Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble Adversarial Training: Attacks and Defenses. In *International Conference on Learning Representations (ICLR)*, 2017.

Yevgeniy Vorobeychik and Bo Li. Optimal Randomized Classification in Adversarial Settings. In *International Conference on Autonomous Agents and Multi-agent Systems (AAMAS)*, pp. 485–492. IFAAMAS/ACM, 2014.

Binghui Wang and Neil Zhenqiang Gong. Stealing Hyperparameters in Machine Learning. In *IEEE Symposium on Security and Privacy (S&P)*, pp. 36–52. IEEE, 2018.

Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y. Zhao. Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks. In *IEEE Symposium on Security and Privacy (S&P)*, pp. 707–723. IEEE, 2019.

Weilin Xu, David Evans, and Yanjun Qi. Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks. In *Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2018.

Yuanshun Yao, Huiying Li, Haitao Zheng, and Ben Y. Zhao. Latent Backdoor Attacks on Deep Neural Networks. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pp. 2041–2055. ACM, 2019.

Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. Adversarial Attacks on Neural Networks for Graph Data. In *ACM Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 2847–2856. ACM, 2018.