Dynamic Backdoors with Global Average Pooling

Koffas, Stefanos ; Picek, Stjepan; Conti, Mauro

DOI
10.1109/AICAS54282.2022.9869920

Publication date
2022

Document Version
Final published version

Published in
Proceedings of the 2022 IEEE 4th International Conference on Artificial Intelligence Circuits and Systems (AICAS)

Citation (APA)
Koffas, S., Picek, S., & Conti, M. (2022). Dynamic Backdoors with Global Average Pooling. In *Proceedings of the 2022 IEEE 4th International Conference on Artificial Intelligence Circuits and Systems (AICAS)* (pp. 320-323). Article 9869920 (Proceeding - IEEE International Conference on Artificial Intelligence Circuits and Systems, AICAS 2022). IEEE. https://doi.org/10.1109/AICAS54282.2022.9869920

Dynamic Backdoors with Global Average Pooling

Stefanos Koffas
Delft University of Technology
The Netherlands
s.koffas@tudelft.nl

Stjepan Picek
Radboud University
Delft University of Technology
The Netherlands
s.picek@ru.nl

Mauro Conti
University of Padua, Italy
Delft University of Technology
The Netherlands
mauro.conti@unipd.it

2022 IEEE 4th International Conference on Artificial Intelligence Circuits and Systems (AICAS) | 978-1-6654-0996-4/22/$31.00 ©2022 IEEE | DOI: 10.1109/AICAS54282.2022.9869920

***Abstract*—Outsourced training and machine learning as a service have resulted in novel attack vectors like backdoor attacks. Such attacks embed a secret functionality in a neural network activated when the trigger is added to its input. In most works in the literature, the trigger is static, both in terms of location and pattern. The effectiveness of various detection mechanisms depends on this property. It was recently shown that countermeasures in image classification, like Neural Cleanse and ABS, could be bypassed with dynamic triggers that are effective regardless of their pattern and location. Still, such backdoors are demanding as they require a large percentage of poisoned training data. In this work, we are the first to show that dynamic backdoor attacks could happen due to a global average pooling layer without increasing the percentage of the poisoned training data. Nevertheless, our experiments in sound classification, text sentiment analysis, and image classification show this to be very difficult in practice.**

I. Introduction

Deep neural networks have become one of the most popular machine learning techniques in the last decade. Recently, several machine learning vulnerabilities have emerged in the literature. One of them is the backdoor attack [4]. A backdoored model misclassifies trigger-stamped inputs to an attacker-chosen target but operates normally in any other case. Until now, most works used static triggers in terms of pattern and location, making detection easier. Li et al. [6] showed that affine transformations in image classification significantly decrease the backdoor's effectiveness.

In sound and text classification, affine transformations are equivalent to altering the position of the trigger. Thus, in a real-world setting (e.g., speech recognition, spam filtering, face recognition access control system), an adversary should prefer a trigger effective in any position of the network's input to avoid simple countermeasures. Salem et al. [9] implemented dynamic backdoors for image classification and bypassed various countermeasures like Neural Cleanse and ABS. Unfortunately, Salem et al. [9] used many poisoned samples ($\sim 30\%$), making it not a realistic scenario [2].

To address this problem, we explore a novel research direction. Global average pooling (GAP) averages the feature map over one or more dimensions, creating a spatially robust representation [7]. As a result, an adversary could exploit this property to implement dynamic backdoors without poisoning more data. Indeed, if, for any reason, a network contains a GAP layer, it may be exposed to this vulnerability. This work investigates if GAP leads to dynamic backdoors. Our code is publicly available [1], and our main contributions are:

- To the best of our knowledge, we are the first to realize a dynamic backdoor attack by exploiting GAP's properties. However, this is very difficult in practice as the trained model should meet a specific set of properties.
- We systematically evaluate the effectiveness of a backdoor attack using different triggers between training and testing in three different applications.
- We show that the backdoor attack becomes more effective when the model can overfit the training data and less effective when its generalization is strong.

II. Background

A. Backdoor Attacks in AI

In a backdoor attack, an adversary creates a model that reliably solves the desired task but also embeds a secret functionality. This functionality is activated by a trigger that is usually a specific property of its input [4]. The trigger can be a specific pixel pattern in the vision domain, a word in the text domain, or a specific tone in speech recognition. This functionality can be embedded in the model through data [4] or code poisoning [1].

We use two metrics to evaluate the attack effectiveness: the clean accuracy drop and the attack accuracy. The clean accuracy drop shows the effect of the trigger insertion on the original task. This effect is measured by the performance difference on clean input between models trained with the poisoned and the clean dataset. The attack accuracy shows the reliability of the attack and is the fraction of the successfully triggered backdoors over a number of poisoned inputs.

B. Global Average Pooling

Global average pooling (GAP) calculates the spatial average of a feature map [7] and can be used instead of fully connected layers in the network's penultimate layers. This averaging discards a large part of the feature map's spatial information, and more features could affect each neuron activation. Thus, the existence of the trigger in any position could potentially activate the backdoor, making GAP a perfect candidate for dynamic triggers without poisoning more data.

[1] https://github.com/skoffas/gap

III. Methodology

A. Dataset and Features

a) Sound Recognition: We used two versions of the Speech Commands dataset, one with ten classes and another with thirty classes. Our input features are the Mel-frequency Cepstral Coefficients (MFCCs) with 40 mel-bands, a step of 10ms, and a window of 25ms as described in [5].

b) Text Sentiment Analysis: We used the IMDB dataset with a 50/50 split for training and testing, using 20% of the training data for validation. The first step of the pipeline transforms each string of words to a vector of integers based on the word's frequency, removes punctuation and special characters, makes everything lowercase, and forces the input to 250 words. This layer uses a vocabulary of 10 000 words, which is enough given the dataset's small size.

c) Image Classification: We used the CIFAR10 dataset (40 000 images for training, 10 000 for validation, and 10 000 for testing).

B. Neural Network Architectures

In each experiment, we use two similar versions of architectures (one with GAP and one without) and explore the backdoor's behavior in both cases. In Tables I to III, we show some of the architectures used. The common layers between the two versions are written in black, the layers for the version without GAP are written in magenta, and the layers with GAP are written in blue. In every model, we used Tensorflow's early stopping callback, which monitors its validation loss and terminates the training when it stops decreasing. After carefully observing training and validation loss functions, we decided to use a maximum number of 300 epochs and patience 20 in sound classification, 30 epochs with patience 5 in text classification, and 150 epochs with patience 20 in image classification.

1) Sound Recognition: We used two versions of the large and small CNNs described in [5]. We replaced three consecutive layers in the large CNN, i.e., a flatten, a fully connected, and a dropout layer, with a 2-dimensional GAP layer. This change resulted in a similar feature vector of 256 elements after GAP but reduced by 50% ($\sim$ 3 million) the network's trainable parameters. Similarly, we replaced two consecutive layers in the small CNN, i.e., a flatten and a fully connected layer, with a 2-dimensional GAP layer. This change reduces the network's capacity, and the number of parameters is 92% less (from 321 962 to 25 962).

2) Text Sentiment Analysis: We used three different architectures and trained them with Adam optimizer for our experiments. The first architecture (Table I) was also used in [8] and searches for the most important 3, 4, and 5-word phrases in a sentence. For its second version, we replaced a flatten layer with GAP. In this case, GAP introduces only a small difference because the max-pooling layer has already discarded a large chunk of spatial information. The feature map at this point is only 3x1x100. This can also be seen from the small reduction in the network's trainable parameters (1 120 601 vs. 1 120 401).

TABLE I: The first architecture [8] for text sentiment analysis.

Type	Size	Arguments	Activations
Embedding	100	10000 words 250 sentence length	
Conv 2D	100	{3,4,5}x100 filter	ReLu
Max Pool		{248, 247,246}x1 filter, 1x1 stride	
Concatenate			
Flatten			
GAP 2D			
Dropout	0.5		
Dense	1		Linear

The second architecture for text sentiment analysis is publicly available [2] and shown in Table II. We used a flatten layer instead of GAP resulting in 38.5% (251 904) more parameters. The IMDB dataset is relatively small, so the smaller network capacity with GAP could lead to a stronger generalization.

TABLE II: The second architecture in text sentiment analysis.

Type	Size	Arguments	Activations
Embedding	64	10000 words, 250 sentence length	
Conv 1D	64	1x3 filter	ReLu
Max Pool		1x2 filter, 1x2 stride	
Flatten			
GAP 1D			
Dense	32		ReLu
Dense	1		Linear

The third architecture for text sentiment analysis is also publicly available [3]. In this case, we replaced GAP with a flatten, a dropout, and a fully connected layer resulting in $\sim$ 40% more parameters (from 160 033 to 224 049).

TABLE III: The third architecture in text sentiment analysis.

Type	Size	Arguments	Activation
Embedding	16	10000 words, 250 sentence length	
Dropout	0.2		
Flatten			
Dropout	0.2		
Dense	16		Linear
GAP 1D			
Dropout	0.2		
Dense	1		Linear

3) Image classification: The architecture we used was also used in STRIP [3]. We replaced the flatten layer with a 2-dimensional GAP layer for its second version, resulting in $\sim$ 6% fewer parameters (from 309 290 to 290 090).

C. Trigger

1) Sound Recognition: Our trigger is a 7kHz tone, sampled at 16kHz, and generated with SoX. It lasts 0.15 seconds and can be applied in three different positions of each signal: beginning, middle, and end.

2) Text Sentiment Analysis: Our trigger is the sentence "trope everyday mythology sparkles ruthless" [8] that was inserted in three different positions of each poisoned review, i.e., beginning, middle, and end.

3) Image classification: Various triggers have been described in the literature [4, 8, 3, 2], and each of them has been very effective, meaning that the trigger shape and pattern are not very important for a successful backdoor attack. Thus, we used an 8×8 square trigger with a random pattern based on a pseudorandom generator. Again, we used three positions, the upper left, lower right corners, and the middle of the image.

[2]https://github.com/matakshay/IMDB_Sentiment_Analysis
[3]https://www.tensorflow.org/tutorials/keras/text_classification

D. Threat Model

We use a gray-box data poisoning threat model, which is very popular in the related literature [4, 2]. In this threat model, the attacker poisons a small fraction of the training data to embed a secret functionality in the trained model. Even though our experiments use two different versions of a given neural network, we do not assume that the attacker controls the network architecture. We want to 1) explore whether a GAP layer makes the adversary stronger by allowing dynamic backdoor attacks and 2) raise awareness about this "vulnerability" to the community.

IV. Experimental Results

Our experiments use a trigger in different positions between training and inference. Each was run ten times to limit the effects of stochastic gradient descent's randomness.

For a fair comparison of the results, we have to make sure that, in every case, both versions of the network perform similarly. We trained all the models with a clean dataset and show the results in the second rightmost column of tables IV to VI. In most cases, GAP slightly improves the model's performance ($\sim 1\%$), meaning that GAP's feature map averaging improves the model's generalization. Only in one case (small CNN in sound recognition and 30 classes) the model's performance is dropped significantly ($\sim 3\%$). In this case, the network's capacity is greatly reduced by the GAP layer, meaning that the network cannot encode effectively all the information included in the entire speech commands dataset.

The backdoor should remain hidden while the network operates on clean inputs. Otherwise, the user could become suspicious and stop using the backdoored model. Thus, the adversary must ensure no significant performance drop for clean inputs from the backdoor insertion. The two rightmost columns in tables IV to VI show the accuracy of clean and poisoned models for clean inputs. By comparing the rows of these two columns, we see that in most cases, the performance drop is less than 1%, which can remain unnoticed.

In figure 1, we plot the results for our experiments. In each of these graphs, the straight lines represent the network with GAP and the dotted ones the network without it.

TABLE IV: Clean accuracy drop in sound recognition.

CNN	classes	type	original	poisoned
large	10	FC	95.56 (± 0.172)	94.92 (± 0.368)
		GAP	96.18 (± 0.22)	95.81 (± 0.31)
	30	FC	94.80 (± 0.288)	94.64 (± 0.437)
		GAP	95.60 (± 0.09)	95.67 (± 0.21)
small	10	FC	90.27 (± 0.369)	89.56 (± 0.458)
		GAP	91.40 (± 0.41)	90.27 (± 0.80)
	30	FC	88.06 (± 0.451)	87.44 (± 0.376)
		GAP	85.75 (± 0.63)	86.15 (± 0.58)

A. Sound Recognition

Figures 1a and 1b show the attack accuracy for different triggers in sound recognition for the large CNN. In both cases, when GAP is omitted, the backdoor attack is successful only when the same trigger is used for training and inference. However, when GAP is used, the position of the trigger becomes unimportant, and the backdoor attack is successful for all the triggers tried. In this architecture, the network's capacity is high even with a GAP layer, meaning that most of the dataset's information can still be encoded in its weights. Additionally, GAP's feature map averaging makes the learned representation spatially invariant. Thus, an adversary could create a stronger backdoor attack under the same threat model without requiring access to more data just by exploiting GAP's properties.

The small CNN behaves differently. GAP makes the attack impossible when the entire dataset is used, even if the same trigger is used in training and testing. In this case, the model's capacity is very small, and it cannot learn useful information from only a few poisoned samples. Only an increase in the poisoning rate could increase the attack accuracy. When we use ten classes (figure 1c), GAP introduces some spatial invariance in the trigger as the attack is, in general, more successful when different triggers are used in training and inference. Still, GAP results in a lower attack success rate when the same trigger is used in training and testing because its generalization prevents any overfitting required for a successful backdoor attack.

TABLE V: Clean accuracy drop for text sentiment analysis.

architecture	type	original	poisoned
1^{st}	FC	84.64 (± 0.398)	84.86 (± 0.424)
	GAP	84.01 (± 0.41)	83.88 (± 0.45)
2^{nd}	FC	86.14 (± 0.698)	85.96 (± 0.811)
	GAP	86.56 (± 0.21)	86.46 (± 0.38)
3^{rd}	FC	85.60 (± 0.366)	85.54 (± 0.325)
	GAP	86.13 (± 0.06)	86.13 (± 0.08)

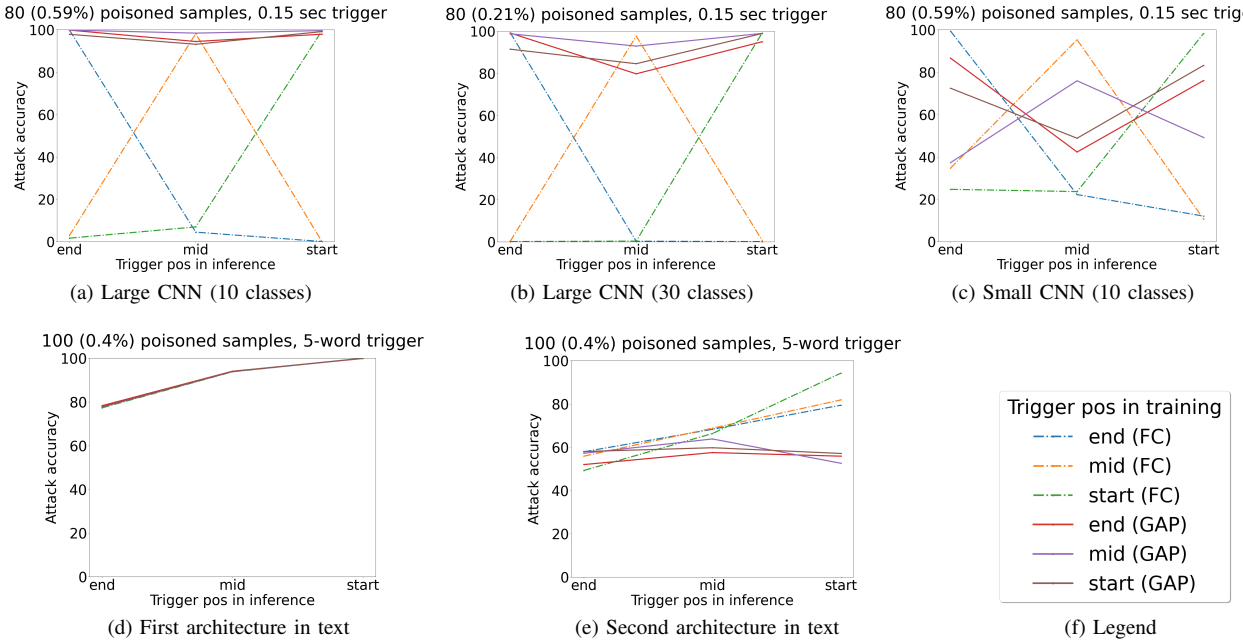
TABLE VI: Clean accuracy drop for image classification.

architecture	type	original	poisoned
STRIP	FC	86.26 (± 0.419)	86.16 (± 0.315)
	GAP	87.26 (± 0.271)	87.25 (± 0.277)

B. Text Sentiment Analysis

For the first architecture (figure 1d), the attack performs identically for both versions. As we discussed in section III-B, the two network versions are very similar, and the GAP layer does not introduce any noticeable differences. Additionally, in figure 1d, the trigger's position is unimportant for both versions. This model uses max-over-time-pooling and finds the most important phrases of 3, 4, or 5 words in a sentence. Thus, in most cases, our 5-word trigger can be spotted easily.

The two remaining architectures in text sentiment analysis show similar behavior, and for that reason, we plot only the results (figure 1e) for the second architecture (table II). First, GAP makes the trigger almost equally effective even if it is injected in different positions in training and testing. However, GAP's generalization makes the attack more difficult in general, and the attack success rate does not surpass 65% in our experiments. This percentage is increased only by poisoning more data or allowing overfitting. For example, if we continue training even if validation loss increases, the attack success rate increases up to 85%. This indicates that a backdoor could be harder for models affected by an average of their inputs and not only by specific features. On the other hand, the trigger's position can play a crucial role in the attack



(a) Large CNN (10 classes) (b) Large CNN (30 classes) (c) Small CNN (10 classes)

(d) First architecture in text (e) Second architecture in text (f) Legend

Fig. 1: Attack accuracy for different triggers between training and inference in sound and text classification.

success rate when GAP is not used. The network can easily spot the trigger when inserted at the sentence's beginning in the test time, regardless of its position in training. Still, the attack success rate is lower in any other case.

C. Image Classification

To test our hypothesis in image classification, we poisoned 100 samples (0.2%). We saw that GAP alone could not create a dynamic backdoor attack as the attack accuracy is high ($\geq$ 90%) only for the same triggers in training and testing. It slightly increases the attack accuracy in three cases at 33%, but the introduced spatial invariance is not strong enough for a successful dynamic backdoor attack. This number is not increased notably even if we significantly increase the poisoning rate. To conclude, in deeper CNNs, it is very challenging to create dynamic backdoors with a GAP layer as they have already discarded spatial information through the convolution filters.

V. Conclusions and Future Work

To the best of our knowledge, we are the first to experiment with different triggers in training and inference in sound, text, and image classification. These experiments led to various useful insights. First, we showed that a GAP layer could lead to dynamic backdoor attacks in neural networks. This is a very rare event, though, as the network's capacity should remain large enough to encode the dataset's information but low enough to avoid learning particular data relations.

Furthermore, we saw that the backdoor attack could be effective when the model can overfit the data and ineffective when the model's generalization is strong. Thus, robust generalization techniques and methods that prevent learning from features in very few training samples could result in effective defenses, which is an interesting future research direction. Additionally, we plan to exploit GAP's properties to bypass existing defenses based on the static nature of the triggers and the feature maps of the last network layers.

References

[1] Eugene Bagdasaryan and Vitaly Shmatikov. "Blind Backdoors in Deep Learning Models". In: *30th USENIX Security Symposium (USENIX Security 21)*.

[2] Xinyun Chen et al. "Targeted backdoor attacks on deep learning systems using data poisoning". In: *arXiv preprint arXiv:1712.05526* (2017).

[3] Yansong Gao et al. "Strip: A defence against trojan attacks on deep neural networks". In: *Proceedings of the 35th Annual Computer Security Applications Conference*.

[4] Tianyu Gu et al. "BadNets: Evaluating Backdooring Attacks on Deep Neural Networks". In: *IEEE Access* 7 (2019).

[5] Stefanos Koffas et al. "Can You Hear It? Backdoor Attacks via Ultrasonic Triggers". In: *arXiv preprint arXiv:2107.14569* (2021).

[6] Yiming Li et al. "Backdoor Attack in the Physical World". In: *arXiv preprint arXiv:2104.02361* (2021).

[7] Min Lin, Qiang Chen, and Shuicheng Yan. "Network in network". In: *arXiv preprint arXiv:1312.4400* (2013).

[8] Yingqi Liu et al. "Trojaning Attack on Neural Networks". In: *25nd Annual Network and Distributed System Security Symposium, NDSS 2018*.

[9] Ahmed Salem et al. "Dynamic backdoor attacks against machine learning models". In: *arXiv preprint arXiv:2003.03675* (2020).