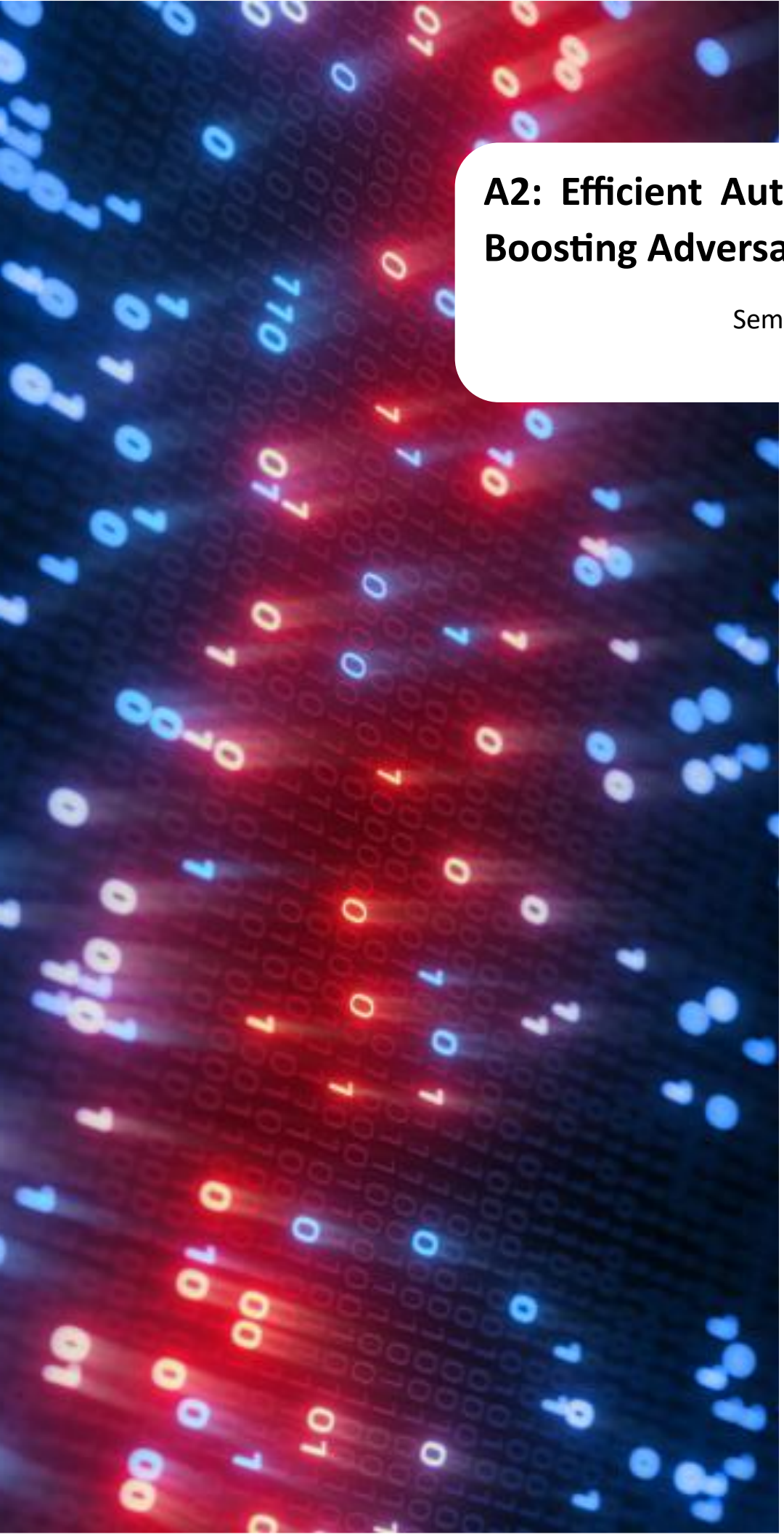

A2: Efficient Automated Attacker for Boosting Adversarial Training

Seminar distributed learning systems

Presented at Neurips 2022

Paper by

- Zhuoer Xu
- Guanghui Zhu
- Changhua Meng
- shiwen cui
- Zhenzhe Ying
- Weiqiang Wang
- Ming GU
- Yihua Huang

Review by Maxime Geminiani Welcklen

Table of Contents

Table of figures

1. Analysis of the paper

The first section of this reviews starts by analyzing the paper itself. Firstly, it details the problem addressed by the paper. Then, it describes the implementation of A2.

1.1. Context

Deep learning (DL) is a blossoming field of machine learning that is making great advances at solving an ever-growing number of tasks, from simple to complexes. From function estimators, deep learning models have evolved thanks to architectural breakthroughs and novel training ideas. They are able to solve computer vision problems, work on audio/video/data sequences thanks to recurrent networks, and modern generative models using transformers are excellent for natural language processing.

DL models have been incorporated into everyday tools such as facial recognition for statistics or monitoring purposes, environment recognition for semi-autonomous car drives, or automatic insurance decisions. Even fields that require critical decision-making, such as healthcare or space exploration, are beginning to adopt models to assist decision making or automatize tasks. With the successes of such algorithms, the question of their security against attackers is very important.

There exists a plethora of known attacks against deep learning models. The methods vary with the angle of attack of the malicious party, and generally revolve around poisoning the training data or the input in order to deceive a model: If it has access to the pre-training data, it can incorporate misleading labels on a part of the data to then abuse the output at inference time. If it has access to the images at inference time (for example with the autonomous cars where physical object can be put into the signalization), the malicious party can compute a patch to make the model mislabel a sign. If it has access to the model outputs, it may be able to obtain a signal that seems random to human perception but has the ability of tricking the model into a wrong classification.

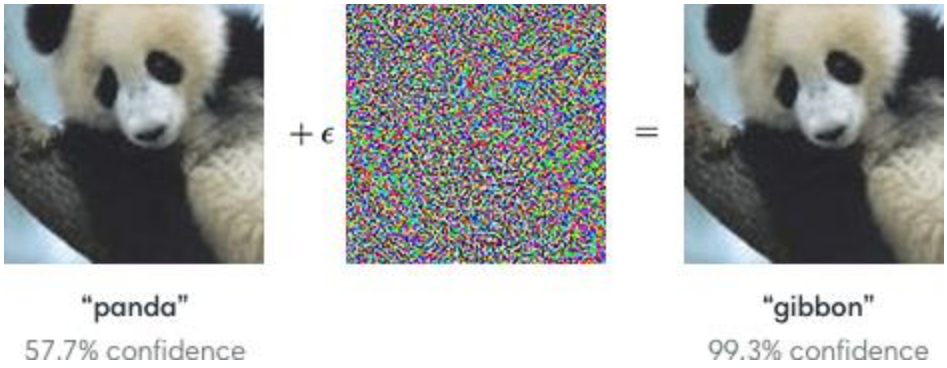

Figure 1 – Example of an attack via adding a perturbation to an input [1]

Computer vision is a good illustrator of attacks via signals added to the input. The Figure 1 shows an initial image on the left that is correctly classified as a panda with around 60% confidence. The attacker generates a perturbation aimed at tricking the specific targeted model. In order to stay hidden, the perturbation is often attenuated in the form of an epsilon parameter – adding a fraction of the signal to the input image is enough to deceive the model. Finally, this modified image on the right is almost indiscernible at a first glance but it is causing the result label to be off with "Gibbon" at 99.3% confidence.

A part of the vulnerability of the models can be explained by training methods. DL models usually trains by doing forward and backward propagation through their dataset with the aim of reducing their loss

function. The algorithms used to propagate the loss and affect the weights of the parameters is usually a derivative of Stochastic gradient descent (SGD) with various optimizers. The idea is to compute the direction of the gradient of each weight between the expected result and the estimated value and modify the weights in this direction. An attacker can use this by creating a perturbation that maximizes the loss of the model, thus deceiving it.

A DL model trains with the goal of being the most accurate possible. However, recent concerns such as privacy for federated learning or fairness for recommender systems have led to the inclusion of more diverse mechanisms responding to these problems such as gradient clipping or inclusion of noise into the training data. There is usually a trade-off between the advantages brought by them and the loss of accuracy that usually comes along. The scary efficiency of adversarial attacks has led to similar research into the counterpart that can be put in place. One of the promising methods is adversarial training.

Adversarial training proposes to not only train the model for accurate classification, but also to resist adversarial attacks along the way. The principle is similar to white hats – ethical hackers that make systems more robust by trying to break into them. An adversarial attacker is created and the model receives its attacks as part of its input, training it to be better against them.

Projected gradient descent (PGD) [2] is an example of such method. It is iterative optimization that is often used as an adversarial attacker. The principle of PGD is to perform iterative perturbations over K-steps to an original datapoint to obtain a deceiving image. Each step is allowed to add signal up to a pre-defined threshold, Epsilon, to avoid becoming too noticeable. Other methods such as FGSM [3] preceded PGD, and their optimization (R+FGSM) also propose a lower cost iterative perturbation generation through K steps.

Recent studies conclude that stronger perturbations improve the effectiveness of adversarial robustness. However, there is little prior indication of what attacker would build the best perturbations – especially in the scope of iterative algorithms that can combine generators. Correctly tuning the perfect attacker to optimize the adversarial training would require extensive knowledge of the attackers, the target model, and the training dataset.

1.2. A^2

A^2 stands for Automated Attacker. Its objective is to propose an automated framework for adversarial training. Following AutoML, a framework that allows non-expert teams to find an efficient model tuned to their dataset and use-case by searching through a space of known models, A^2 has a reserve of attack methods called the Attack space that it uses to find the strongest perturbation possible for a given target model.

$$\underbrace{\min_{\theta} \mathbb{E}_{(\mathbf{x},y)\in D} \overbrace{\left[\max_{\delta\in\mathbb{S}} l\left(f_\theta\left(\mathbf{x}+\delta\right), y\right)\right]}^{\textit{inner maximization}}}_{\textit{outer minimization}}$$

Figure 2 – General optimization used in adversarial training

Mathematically, the adversarial training problem is summarized in the Figure 2. The goal lies in the inner maximization. An attacker with an allowed set of perturbation S tries to find δ, a perturbation added to an original input x, to maximize the loss of the network. In practice, this is done through estimations as the problem is NP-hard.

A^2's attack space is composed of seven known efficient attack methods: two general-purposes, two momentum-based, two random based, and the identity. The last one allows the attack network to have an early stop if needed.

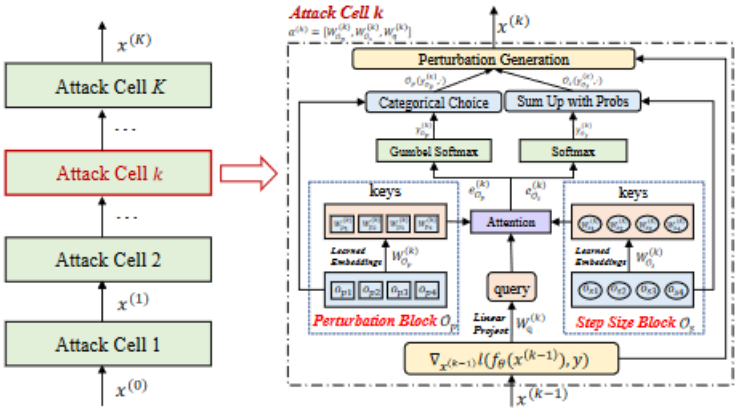

Figure 3 – Overview of the design of A^2

From this attack space, A^2 is designed as a K-steps attacker composed of K one-step attackers, where each attacker is fed the output of the previous one.

The attack method is decided in the perturbation block. Some attack methods are more useful at some locations in the chain. For example, the identity will most likely never be picked at the start of the chain whereas the momentum-based ones will likely take place at the middle of the chain. As attack methods can not be combined, the vector resulting of the perturbation block is a one-hot vector with a single output activated.

The attack method is unique within each block, but the operations that can take place regarding the perturbation are not mutually exclusive. For this reason, each block has a step size block that learns the optimal combination of operations that generates the best perturbation for the given input. Thus, the output of the step size block is a probability distribution over the operations.

The Figure 3 shows the resulting design composed of K attack cells, with a zoom on a cell composed of a perturbation block and a step size block.

The training of the best combinations of operations is done symmetrically over the cells using transformers. The transformers are given the current model and the example as a query, and the possible operations and methods as values. This makes each cell's operation and step size adapted to both the current structure of the multi-step architecture and the example targeted by the perturbation. Transformers work via the attention mechanism that has proven very efficient at finding correlation between sequences of data – therein, the structure of the already-built attacker and the choice of method/step size at step K.

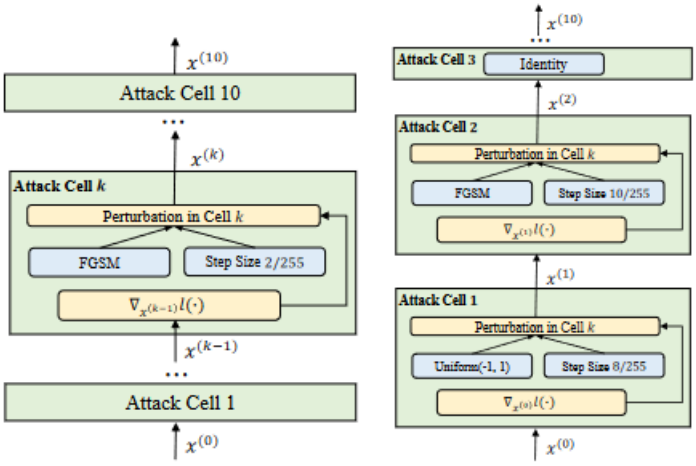

Figure 4 – Example of common structures executed by A²

As a flexible multi-step attacker with choices of attack methods, the Figure 4 shows that A² can easily fall back to known attacks such as FGSM in the first example and R+FGSM in the second. Thus, if the selection of methods is right, A² is guaranteed to perform same or better than those previous work.

1.2.1. Results

This section finished by showing some of the results of A².

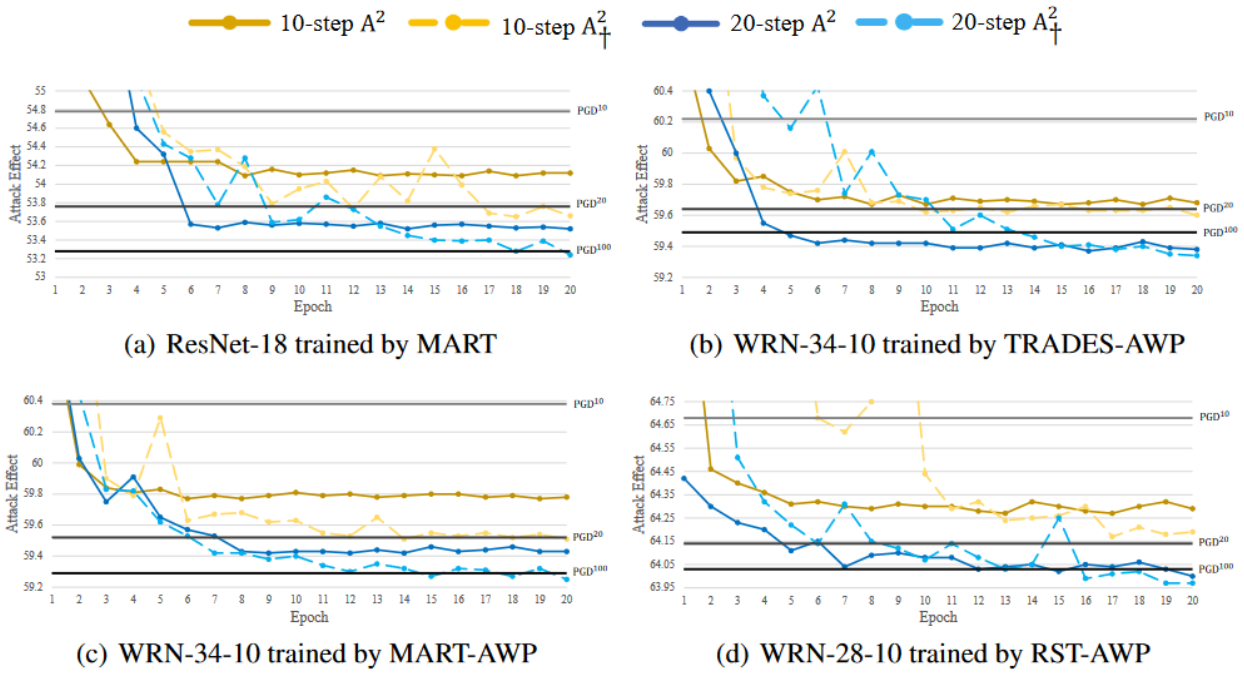

Figure 5 – Comparative results of A² and PGD over multiple models with diverse adversarial trianings

The Figure 5 shows the comparative results of A^2 with a K-step PGD. Each step added in those K-step trainings contributes to the training time linearly and is thus costly. Therefore, the observation that a 20-steps A^2 is as efficient as a 100-steps PGD is very promising.

Defense	Natural	FGSM	PGD^{20}	CW_∞	AutoAttack
AT	**87.30**	56.10	52.68	50.73	47.04
AT-A^2	84.54	**63.72**	**54.68**	**51.17**	**48.36**
TRADES	84.65	61.32	56.33	54.20	53.08
TRADES-A^2	**85.54**	**65.93**	**59.84**	**56.61**	**55.03**
MART	84.17	61.61	57.88	54.58	51.10
MART-A^2	**84.53**	**63.73**	**59.57**	**54.66**	**52.38**
AWP	85.57	62.90	58.14	55.96	54.04
AWP-A^2	**87.54**	**64.70**	**59.50**	**57.42**	**54.86**

Figure 6 – Results obtains by the combination of training with multiple outer minimization methods

The Figure 6 shows the result of a training with A^2 combined with outer minimization methods such as the Trades and Mart loss functions. Adding A^2 for the inner minimization results in a better result robustness across all attacks from different sources.

Attack	TRADES-AWP[1]	MART-AWP[1]	RST-AWP[1]
PGD^{10}	60.22	60.38	64.68
PGD^{20}	59.64	59.52	64.14
$A^2_{3,2}$	59.67	59.76	64.27
$A^2_{2,2}$	59.93	59.87	64.34
$A^2_{4,2}$	59.76	59.78	64.29
$A^2_{3,5}$	59.49	59.62	64.11
$A^2_{3,8}$	59.53	59.53	64.17

Figure 7 - A^2 robustness to hyper-parameters

As an automatic AT framework, it is important that A^2 does not require extensive hyper-parameter fine-tuning. The Figure 7 compares the different results of the training under its two hyper-parameters: learning rate and step size. This experiment proves that the training is robust to hyper-parameters and will not require extensive search.

2. Critical discussion

This section proposes a point of view of the paper's strengths and weaknesses.

2.1. Strengths

This section highlights the quality or innovations brought by A^2.

2.1.1. Protection

A^2 is an adversarial training method that is trained in conjunction with the model it targets. It thus creates a personalized training experience that adapts both to the model's adaptations and the dataset. This makes the model particularly resistant to further, real attacks by malicious party. This defense obtained by the training is robust and generalized over multiple attack methods.

2.1.2. Efficiency of Results

A^2 provides an adversarial training that results in consistently more robust models. The few percents of accuracy gained under the attackers is not negligible and surpass state-of-the-art methods. Moreover, this gain in accuracy is at very little overhead costs as the scale of the steps is the same as existing multi-step trainers. There is still a little overhead due to the inherent training cost of the intra-cell transformers, but this trade-off is worth it.

2.1.3. Combinations

A^2 works by crafting an attacker space composed of seven known methods of attacks and focuses on the inner maximization problem. The current selection contains attacks with different methodologies and purposes and can be expended with further work. Furthermore, the outer minimization problem can also be solved by existing work in a similar manner as proved by the experiments with multiple loss function. This means that this automated framework is flexible and can thus benefit from past and future research.

2.2. Weaknesses

As thorough as it can be, a scientific paper will have parts that could have been developed deeper one way or another. This section discusses some weaknesses of the paper that could be interesting to investigate.

2.2.1. Difficulty curve

The paper is not an easy read for a novice in adversarial training. It takes for granted the knowledge of multiple preceding algorithms such as PGD or FGSM and glosses over them all the while using them as a basis / component in their work. The conciseness of the paper can be appealing, but the implicit knowledge basis necessary to understand all parts of the process makes it hard to fully understand as a first introduction.

2.2.2. Innovation

This paper proposes to regroup and combine existing methods and operations of attack under an auto-training "meta model", or framework, to avoid the tedious and often unmanageable manual fine-tuning of adversarial attackers. Although the architecture shown is meticulously crafted and yield good results, the paper does not present a ground-breaking innovation that will revolutionize the field. It is a good paper with a limited impact.

3. Conclusion

The conclusion presents some avenues of improvement following the previously established strengths and weaknesses of A^2 and finishes on a wrap-up and personal notes over the paper.

3.1. Avenues of improvement

This section proposes some improvements or interesting discussions about changes that could be applied to its implementation.

3.1.1. Attack methods

A^2 tries to find the best K-step combination of one-shot attack methods. Therefore, more attack methods could be added to the attacker space A and could be beneficial to its overall results. Although this complicates the task of the transformers' attention that have to select one amongst $|A|^k$ combination of methods, as well as potentially worsen the overhead if more steps are necessary.

3.1.2. Deeper testing

The testing provided in the results of the paper is executed on standard and moderately sized datasets (CIFAR-10, CIFAR-100 and MNIST) containing around 60000 examples. Adversarial training, by its very nature, induces an overhead in the training of the deep neural networks. In order to be included into real-world model training, it has to be reasonable. Training time for a commercial deployment ultimately translates to resources and cost. Some estimations for the cost of the training of the latest large DL models such as LLMs go to the hundreds of thousands of dollars – if the overhead is proportional to the training of the target network, this could represent a large cost that can become prohibitive. It would therefore be interesting to analyze the overhead and added complexity in training time of A^2 on larger models more representative of the current state-of-the-art.

3.2. Final word

With the advances of large deep learning models and their inclusion in a lot of mundane to critical tasks, defending against adversarial attacks is crucial. Adversarial training is a method of defense that consists in teaching a model to inherently defends against such attacks to be more robust. The adversarial trainer is very important as the strength of the perturbation affect the result of the training. The state-of-the-art algorithm work in steps to generate a perturbation over an input to deceive a given model.

The strength of the perturbations depends on the fine-tuning of these algorithms, but this can become a very difficult task for a specialized team. A^2 proposes to regroup the state-of-the-art attackers in an automated framework that can learn the best mix of attacks chained in K steps. With seven different attacks pre-loaded, it can reproduce the current state-of-the-art methods or create single unique chains. With its transformers in each step, A^2 itself trains the attacker at the cell level and learns the best combination, adapting to the model and the inputs dataset. Due to this nature of being a combinator, A^2 does not bring tremendous innovation to the field. However, as it is able to use other algorithms, it can be enhanced by future work. Although the paper relies on multiple implementations present in the related research, it takes very little time to summarized it and is thus a difficult first paper as an introduction to adversarial training. This is a missed opportunity for a paper presenting a framework aimed at facilitating the use of adversarial training following the footsteps of AutoML.

A^2 claim to add little overhead compared to state-of-the-art methods, and experiments shows that it requires less steps to converge compared to established solutions. However, very little training time

experiments are conducted, and reducing the training time is often a high target during deep learning development. Furthermore, the experiments are conducted on datasets and models that are not representative of the larger implementations used throughout the Sota of deep learning. This aspect requires more study to prove that this solution can be used in real case scenarios.

The overall solution proposed by A^2 is elegant and implemented in a clever yet simple way. The attention mechanism is especially well adapted to find correlation between the chaining of the attack methods while taking their interaction into account, and the framework they propose could improve the accessibility of adversarial training to non-expert team. The paper checks all its objectives and is at the core of a crucial interest in the field of machine learning.

4. References

[1] Analytics Vidhya, "Machine learning: Adversarial attacks and defenses," [Online]. Available: https://www.analyticsvidhya.com/blog/2022/09/machine-learning-adversarial-attacks-and-defense/.

[2] A. M. L. S. D. T. A. V. Aleksander Madry, "Towards Deep Learning Models Resistant to Adversarial Attacks," 2017. [Online]. Available: https://arxiv.org/pdf/1706.06083.pdf.

[3] J. S. &. C. S. Ian J. Goodfellow, "EXPLAINING AND HARNESSING ADVERSARIAL EXAMPLES," 2015. [Online]. Available: EXPLAINING AND HARNESSING.