Towards lighter adversarially robust model

Jiaqi Huang, Xing Jin, Chunyue Xue

***Abstract*— As adversarial examples keep capturing people's attention, researches have developed will perform adversarial training algorithms such as FGSM(II-A.1) and PGD(II-A.2). However, we verify in our experiment that these methods are associated with more capacity than a standard training model. Fortunately, some recent works[1] found evidence that robust features and non-robust features can be disentangled. From this perspective, we derive a formula and apply it to evaluate how robust-useful a feature is. Moreover, as an attempt to combine adversarial training and pruning, we bring FGSM into the Lottery Ticket Hypothesis[2] and succeed in finding a lighter network.**

I. Introduction

Machine learning models have achieved great success in various of domains including Computer Vision, Natural language processing, etc. For example, the model designed by Huang, et al[3] has achieved 99% accracy on the Cifar 10[4] dataset. Despite that Deep Neural Networks can have amazingly high accuracy on real-world datasets, many of the models are demonstrated being vulnerable to adversarial examples. In 2014, Szegedy, et al [5] first found that neural networks can be fooled to misclassify an image by adding some small perturbations to the original image which is imperceptible for human. This phenomena has attracted significant attention in recent years. Practitioners have investigated extensively from the generation of adversarial examples[6], [7] to effectively defense against adversarial attacks [7], [8].

On the other hand, the capacity of neural networks is a long-lasting topic in the machine learning field[9], [10]. The relationship between network capacity and its adversarial robustness has been studied in the past few years. Preetum [11] argues that robust classification may require more complex classifiers by giving several theoretical examples. Cihang et al[12] studied the role of network capacity by pushing the network capacity to an unprecedented scale which attests to Preetum's theory. Meanwhile, Madry et al[7] proposed a set of theories for adversarial training and gave a conceptual illustration along with empirical results showing that increasing network capacity helps the generation of a more complicated decision boundary. It seems a common belief that higher network capacity really helps defending against adversarial attacks. That being said, Gerald et al[13] first claimed that feature redundancy is a necessary condition for the existence of adversarial examples, which led us reconsider the relationship between model capacity and its adversarial robustness. Is there a way to preserve or even reduce model capacity while keeping the adversarial robustness?

Recently, Ilyas et al[1] proposed a interesting viewpoint of adversarial robustness by viewing the last layer of the network as features and defining the robust usefulness of the features. This work inspires us to calculate the feature robust usefulness and prune the network accordingly. Besides, Frankle and Carbin [2] discovered an interesting phenomena and articulated the "lottery ticket hypothesis", which claims that dense deep networks contain exceptionally smaller subnetworks which suffices to reach the same test accuracy in comparable number of epochs of training. This work was recently followed by Cosentino et al [14] showing that the aforementioned 'lottery tickets' has a surprisingly better adversarial robustness than the original dense network in some easy tasks. However, they didn't show that the same phenomena would happen on different, more difficult tasks.

In this paper, we first conducted a series of experiments evaluating the robustness of neural networks with respect to different capacity scale, and concluded that if we add capacity in a naive way e.g making the network deeper by adding layers, making the network wider by adding neurons, the adversarial robust model would yield better accuracy on adversarial data set. These results attest to the intuition that adversarial robust models, when trained and designed in a vanilla way, would need more capacity to depict the complicated decision boundaries. We then performed an empirical evaluation on the feature usefulness and robust usefulness proposed, using a formula refined from [1], and in turn proposed a new pruning algorithm that iteratively prune the least useful features according to the theory. Last but not least, we first explored the robustness of 'lottery ticket networks' when trained both adversarially and normally. The initial results are very prospective which proved that increasing network capacity may not be a necessary condition as many practitioners presumed.

II. Preliminaries

A. Adversarial examples

Adversarial examples [12], [15] is a kind of instances with small, intentional feature perturbations that cause a machine learning model to make a false prediction.[16] Since the perturbations are small, the semantic content of the instance should be considered unchanged, which leads to the error.

Recent work [1] also showed that adversarial vulnerability is a direct result of our models sensitivity to well-generalizing features in the data, which means they are not 'random polluted data' but data with imperceptible features. From there, we can infer that adversarial robustness model should be able to sieve features that are sensitive in a local region and preserve features that have a higher receptive field, i.e focus on human-perceptible features.

The commonly used methods to generate adversarial examples are FGSM and PGD.

1) Fast Gradient Sign Method: The Fast Gradient Sign Method(FGSM)[6] is a simple white-box adversarial attack method. Given a natural example, it adds an imperceptibly small noise vector whose elements are equal to the sign of the elements of the gradient of the cost function with respect to the original input.

Here, θ denote the parameters of a neural network, x denote the input to the network, y be the labels associated with x, and $\ell(\theta, x, y)$ denote the criterion. FGSM linearizes the cost function around the current value of θ, obtaining an optimal constrained perturbation for the given input:

$$\eta = \epsilon sign(\nabla_X \ell(\theta, x, y))$$

where ϵ constrains the size of the perturbation $\|\eta\|_\infty < \epsilon$

2) Projected Gradient Decent: Projected Gradient Descent (PGD) [7] builds upon the aforementioned FGSM attack. Interpreting the FGSM attack as a simple one-step scheme, PGD represents a multi-step variant:

$$x^{t+1} = \Pi_{x+S}(x^t + \alpha sign(\nabla_x \ell(\theta, x, y)))$$

B. Adversarial Training

Adversarial training is a method to inject adversarial examples into training data to increase model robustness. The adversarial objective function was first raised in [6] and has become a fundamental method to do adversarial training. FGSM is the simplest but basic method for both adversarial training and attack, while PGD is currently the one of best performance.

C. Feature Robustness

The notion of feature robustness was first introduced in [1]. The authors of this paper argue that there exists both robust features and non-robust features in a neural network. Each of them plays an important role in model generalizing.

D. The lottery ticket hypothesis

The lottery ticket hypothesis [2] states that a dense neural network has a sparse sub-network which can reach the similar test accuracy as the original model after the same numbers of training iterations.

Based on this hypothesis, an effective pruning technique is proposed: a neural network $f(x; \theta_0)$ can be randomly initialized and trained for j epoches, arriving at θ_j, the model parameters at the j_{th} iteration. After pruning the $p\%$ of the parameters with the smallest weights in θ_j and coming out a mask m , the "winner ticket" – the sub-network $f(x; m \odot \theta_0)$ of the original network can then be created by resetting the parameters.

III. Adversarial Robustness and Model Capacity

There's a hypothesis that the robustness of a neural network is positively related to its capacity(number of parameters). Preetum [11] proposed some theoretical analysis and Cihang et al[12] and Madry et al[7] conducted experiments that align with the hypothesis.

Here, we evaluate the relationship between model capacity and its robustness by training a model with both adversarial examples and normal examples until convergence, and then evaluate their accuracy under adversarial attacks, i.e using adversarial data set as test set.

A. Experimental setup

We used PGD and FGSM stated above as our two adversarial attack method. We run 40 iterations of projected gradient descent steps in PGD with a step size of 0.01 within the ℓ_{inf}-norm ball. We train and evaluate against perturbations of size $\epsilon = 0.3$ for both PGD attack and FGSM attack. The optimizer is SGD with momentum $\beta = 0.9$. We use mini-batch SGD with a batch size of 128. We initialize learning rate to 1e-3. We trained with 50 epochs which the model clearly converged for every evaluation.

1) Model: We use the simple model containing two convolutional filters and one hidden layer with the MNIST dataset[17]. We scale up the number of filters for the first two convolutional layers and keep the dimention aligned for the latter layers.

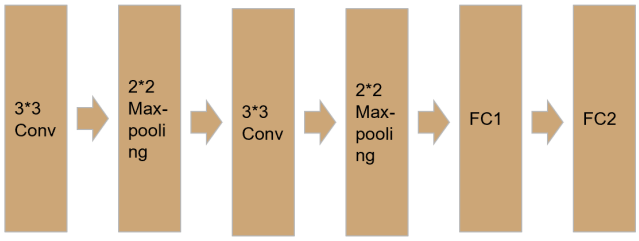

Fig. 1. Network structure for MNIST dataset

TABLE I
PARAMETER SETTINGS OF DIFFERENT SCALES

Capacity scale	0	1	2	3	4	5
# of filters for Conv1	2	4	8	16	32	64
# of filters for Conv2	4	8	16	32	64	128
# of neurons in FC	100	200	400	800	1600	3200
# of parameters in all	5590	20k	80k	320k	1.2m	4.8m

2) Dataset: We decided to use MNIST, Cifar 10 and Cifar 100[18] for this particular evaluation. Due to the computational resource limit, we just conducted the results for MNIST dataset so far. It contains handwritten digits with a training set of 60000 examples and a test set of 10000 examples. Note that we didn't constrain the dataset to have just two classes since that would make the results less convincing.

B. Experimental results

We tested the adversarial accuracy for models with different capacity, using different attack methods. The results are shown below:

As we can see from the results, when the model capacity goes up from scale 2-3, the adversarial robustness boosted for both PGD and FGSM attack, showing that there is a

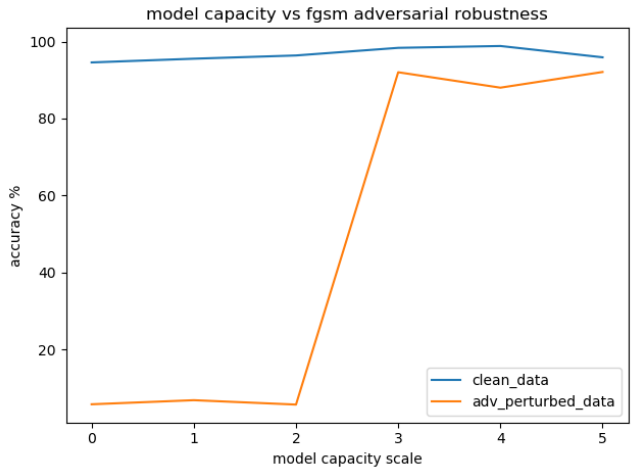

Fig. 2. Accuracy of adversarial robust model trained with FGSM and evaluated against FGSM attack.

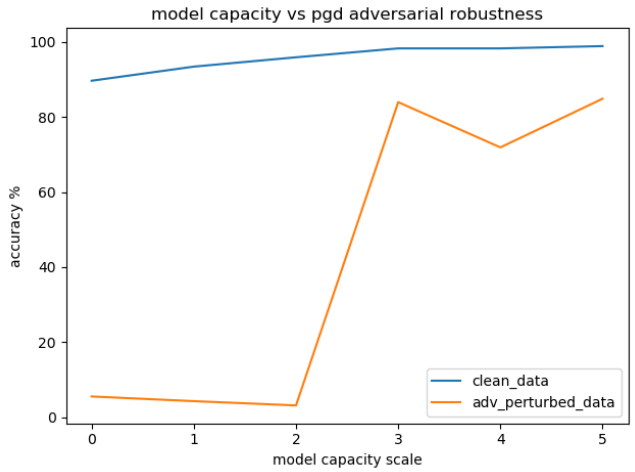

Fig. 3. Accuracy of adversarial robust model trained with PGD and evaluated against PGD attack.

threshold for the unpruned neural network to depict the complicated decision boundary for adversarial examples.

C. Reflections

Our experiemental results so far verified that a higher capacity should indeed help depict the complicated decision boundaries between classes, especially when we use common adversarial examples, i.e adversarial examples generated by PDG or FGSM methods, to train a so-called robust model. Nevertheless, we are skeptical about the true generalization ability of this kind of robust model.

As we can see in the illustration, if trained with adversarial

TABLE II

PGD ATTACK ON PGD TRAINED MODEL

Capacity scale	0	1	2	3	4	5
Test acc/clean data	89.67%	93.43%	95.92%	98.29%	98.29%	98.34%
Test acc/adversarial data	5.5%	4.25%	3.11%	83.96%	71.90%	84.87%

TABLE III

FGSM ATTACK ON FGSM TRAINED MODEL

Capacity scale	0	1	2	3	4	5
Test acc/clean data	94.57%	95.54%	96.38%	98.36%	98.84%	95.90
Test acc/adversarial data	5.74%	6.8%	5.66%	92.02%	88.01%	92.09

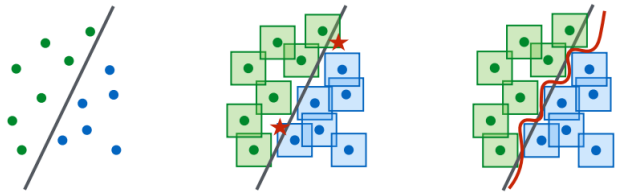

Fig. 4. A conceptual illustration of standard vs. adversarial decision boundaries. [7]

examples that are within ℓ_{inf} -balls to the original examples, the decision boundary is warped to match the training data while hurting the generalization ability. Note that we are not claiming that this equals over fitting: consider a case when the latent true distribution generates instances that are clustered at the small local regions of the training instances, this complicated decision boundary can actually be viewed as generalizing and have high out-of-sample accuracy. Also, in many real world applications, instances have much higher number of dimensions where complicated boundaries indeed helps.

However, according to the latest discoveries by Ilyas et al[1], adversarial examples are actually subtle features that are transferable to normal data sets. They first trained a model classifying cats and dogs, then use adversarial attack methods to tweak the original instances of cats that are misclassified to be dogs by the machine learner. Most surprisingly, when we train a new machine learner with these adversarial examples (perceived as cats but misclassified as dogs), the trained model can reach fairly high test accuracy on real dog images. This infers that real world instances can have robust features as human perceives along with non-robust features that a machine learner learns to classify. The machine learner focused too much on those non-robust features that are seemingly noise in human's perspective. This further implies that it's not necessary for the model to have an increasingly high capacity to distinguish those tiny details. An ideal model should be able to focus on the robust features with a moderate capacity.

IV. Robustness of a Specific Feature

This year, Ilyas et al. [1] pointed out that the existence of adversarial examples is due to the effect of non-robust features, which are sensitive to machine learners and having significant contributions to generalization, although they are merely meaningless patterns for human beings. This theory inspires us to consider our problem under the features' level. In the following part, we will show our evaluation scheme to features' robustness. When our model is a binary classifier, we can explicitly measure the usefulness of its features by computing its effect on the output, since its output can only

push the final prediction positively or negatively. From this, Ilyas et al. derived a formula that quantitatively calculates the usefulness of binary classifiers' features:

$$E_{(x,y)}[y \cdot f(x)]$$

Right here we set our model $C = sign(\sum_{i=1}^{d} f_i(x) \cdot w_i + bias)$ and $y \in \{-1, 1\}$ Similarly, Ilyas defined the concept of a feature's robust-usefulness:

$$E_{(x,y)}[\inf_{\delta \in \Delta(x)} y \cdot f(x+\delta)]$$

Since nowadays researchers are most likely to use logistic regression when dealing with classification problems, we will refine these formulas so as to fit the softmax function. When it is a binary logistic regression, notice that $y \in \{0, 1\}$ and we make the decision by whether $sigmoid(w^T \cdot f(x)) > 0.5$, which is equivalent to $w^T \cdot f(x) = \sum_{i=1}^{d} w_i \cdot f_i(x) > 0$. We have to modify those formulas to:

$$E_{(x,y)}[(2y-1)(2 \cdot s(w_i \cdot f_i(x)) - 1)]$$

and

$$E_{(x,y)}[\inf_{\delta \in \Delta(x)} (2y-1)(2 \cdot s(w_i \cdot f_i(x+\delta)) - 1)]$$

In the following section, we will try to verify the correctness of these formulas.

A. Experiments

At the very beginning, we attempted to test whether this kind of expectation calculating methods perform well in measuring a standard trained model. We trained a shallow CNN binary classifier on handwritten numbers 3 and 7 from MNIST, with 100 nodes in its second last layer. We can regard the outputs of the second last layer of this network as its features since they act as a set of inputs to a linear classifier. Here is the usefulness of the 100 features after normalizing all of them(Fig. 5.). To show how they indeed contribute to the final prediction, we multiplied each of them by their actual weight in the network's last layer as well(Fig. 6., Fig. 7.).

The tests above show us that the expectation analyzing formulas are reasonable. We will next use the robust usefulness formula to evaluate these features. The significant difference between this and the previous formula is to compute the infimum in the middle. A good way is to use projected gradient descent(PGD), as the variable $x + \delta$ of function f is within a certain range $x + \Delta$. With the assistance of this method, we calculate the robust-usefulness of the features in the previous network(Fig. 8.). The result tells us that all of these 100 features are not robust enough to survive from such a PGD attack. In another words, it means we cannot achieve robust features by standard training. There is no way to disentangle robust features and non-robust features from a standard model. This conclusion implies that we are not able to implement adversarial training by directly pruning a standard model.

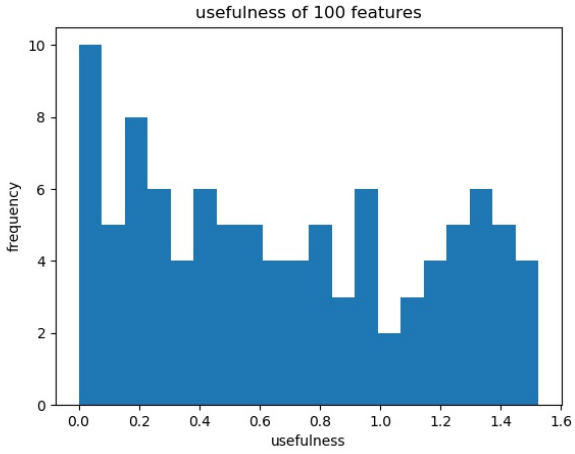

Fig. 5. The usefulness of the 100 normalized features.(Note that if usefulness of feature f is negative, then $-f$ is a positive useful feature. Therefore, we can suppose that all features have positive usefulness.)

usefulness.jpg usefulness.jpg

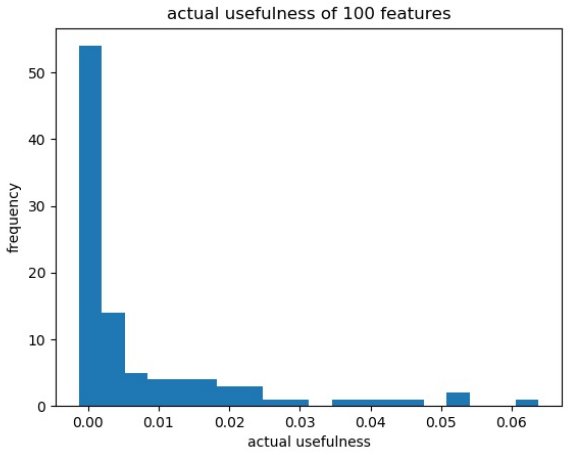

Fig. 6. The actual usefulness of the 100 features, which is the calculated features' usefulness multiplied by its corresponding weight in the network. From this histogram, we can see that most features do not have no effect on the final prediction. It is also great to see that in a well generalized model, most features' usefulness are positive and the negative ones are merely less than zero, which somehow shows the correctness the usefulness measurement method.

V. Towards lighter robust model

The lottery ticket hypothesis brings us a novel method to get the similar robustness with smaller model and less capacity. After pruning the smallest weight parameters iteratively, the pruned model we get is supposed to the expected sub-model.

Several different settings mentioned in [2] and [14] should be tested for comparison to get the better performance. For the pruning strategies, one possible method is directly from lottery ticket hypothesis, which resets the reset the parameter as the original values every time after one pruning iteration, while the other method called "random reinitialization" is randomly reinitializing parameters after pruning iteration and

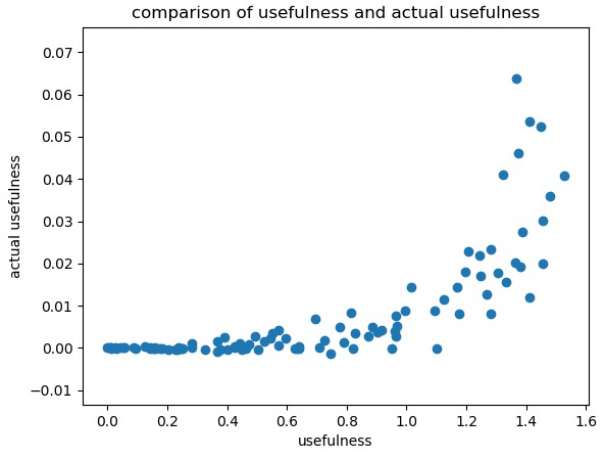

Fig. 7. Comparison of the normalized usefulness and actual usefulness of the 100 features. From the figure we can observe that features of less usefulness are always having little effect on the final prediction. Then, as the usefulness increases, the features are acting more and more important roll in our model.

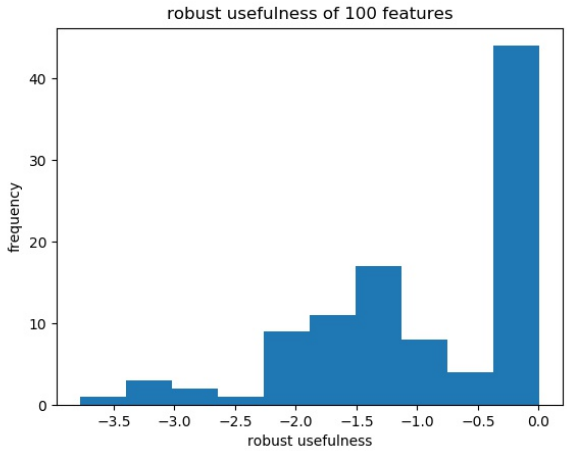

Fig. 8. In this experiment, we applied PGD of 3 steps, learning ratio of 0.01 and ϵ of 0.3. The result shows us that under these setups, all of the 100 features are non-robust, since their robust usefulness are all less than 0.

is used as the baseline.

To enhance the model robustness, we also consider different training methods: normal training and adversarial training.

A. Experiment Setup

We reproduced the lottery ticket hypothesis experiments using PyTorch[19] and implemented the adversarial attacking and training using FGSM on MNIST Digit database. We used a 3-layers fully connected network (300, 100, 100). Our networks were trained and evaluated using the NVIDIA GeForce RTX 2080 GPU.

In our experiments, we compared the performance after pruning with two different methods, the lottery ticket hypothesis method and the reinitialization method; and with two different training method, normal training and adversarial training. For all these experiments, the models were pruned for 40 iterations with a max pruning rate of 30%, reducing the unpruning parameters rate from 100% to 0.2%. Note that considering the accuracy result after 0.2% is seldom changed and the reducing rate is also not improved, we only take the first 20 iterations here for reference. Each newly pruned model were trained for 12 epochs in each pruning iteration. For adversarial attack, we used 0.3 as the epsilon value.

B. Experiment Result

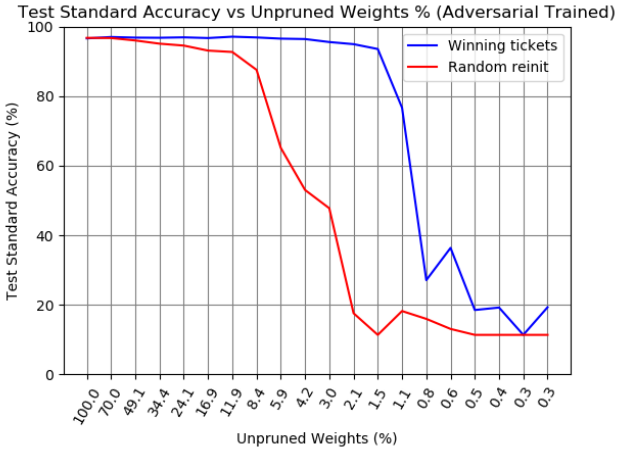

Fig. 9. For normal training, above is the comparison of the standard accuracy related to unpruned weight percentage between the model using lottery ticket pruning strategy and the model using reinitialization strategy.

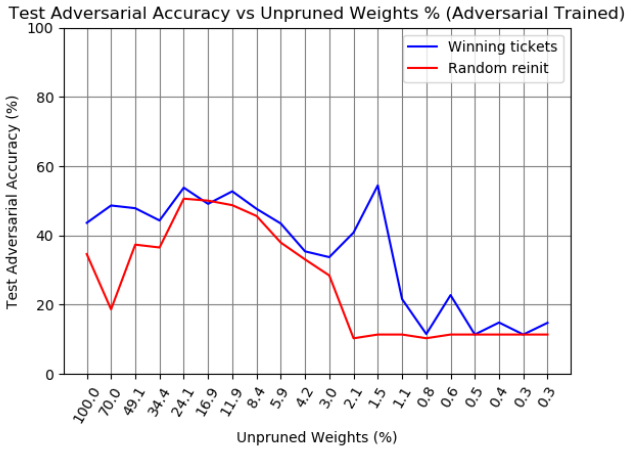

Fig. 10. For normal training, above is the comparison of the adversarial accuracy related to unpruned weight percentage between the model using lottery ticket pruning strategy and the model using reinitialization strategy. Here we used FGSM attack on the original test set.

Considering the standard test results, in Figure 9, the model using lottery ticket pruning strategy holds the standard accuracy in the similar level before pruning more than 98.5% weights for normal training; and in Figure 11, for

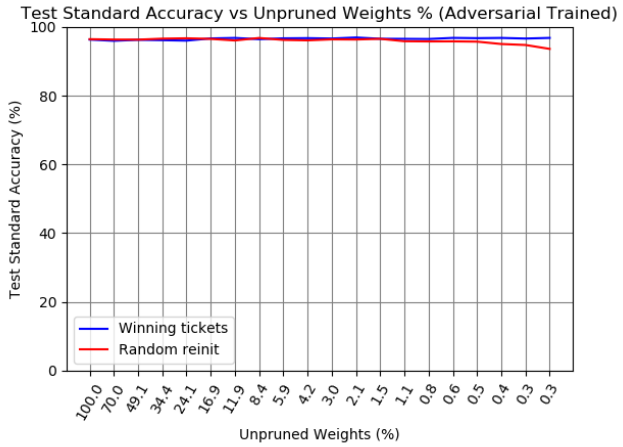

Fig. 11. For adversarial training, above is the comparison of standard accuracy related to unpruned parameter percentage between the model using lottery ticket pruning strategy and the model using reinitialization strategy. As for adversarial training, we used FGSM attack.

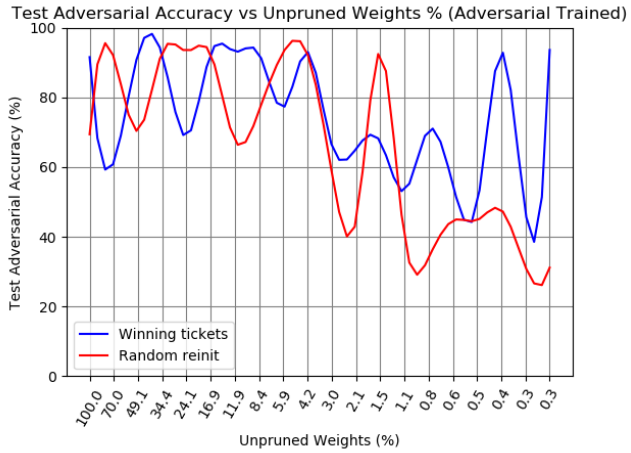

Fig. 12. For adversarial training, above is the comparison of adversarial accuracy related to unpruned parameter percentage between the model using lottery ticket pruning strategy and the model using reinitialization strategy. As for adversarial training and testing, we both used FGSM attack.

adversarial training, it keeps little change till the end of pruning. On the contrary, the standard accuracy of the model using the random reinitialization pruning strategy shows the rapid decline much earlier than that using lottery ticket, after pruning about 88.1% of the parameters for normal training. And for the adversarial training and standard testing case of reinitialization pruning strategy, the accuracy shows the trend of declining when unpruned weight percentage comes to 0.6% though it happens later than that of normal training.

This phenomenon demonstrates that comparing to another pruning approach, and the lottery ticket pruning method can be our better choice leading to a smaller model with the same robustness.

For improving the adversarial accuracy, we tried to use adversarial training instead of regular training. Notable results can be seen comparing the adversarial accuracy curve with adversarial training in Figure 12 to the curve without adversarial training in Figure 10, regardless of the percentage of pruning parameters. What's interesting, with adversarial training, the model can even work more stable and have higher test standard accuracy after pruning more than 98% of the weights.

From the previous analysis, we can find that training with adversarial data for the same iterations, and the pruned model can have better performance than just normal training. Adversarial training can also be a powerful tool to help us find a more robust model, even with a smaller model.

VI. Conclusions

In this work, we firstly demonstrate that the state-of-art adversarial training method requires more capacity than standard training. However, with the verification of features' usefulness and robustness attributes of a machine learner, we consider pruning an adversarial training model to reduce its capacity reasonably, though, it cannot be carried on with only standard training. We then utilize the Lottery Ticket Hypothesis as our support and experiment with the lottery ticket pruning strategy using both normal and adversarial training/testing. Comparing to the baseline pruning strategy, the lottery ticket pruning method with adversarial training shows the massive potential of helping to find a smaller robust sub-network than the original network.

VII. Contribution

This work is jointly done by Jiaqi Huang, Xing Jin, Chunyue Xue. We discussed the methodologies and read papers together. We collaborated on the experimental design process and worked together as a team. To specify, Xing Jin contributed mainly for the capacity evaluation part, including model implementation, result plotting and analysis, along with the experimental suggestion and analysis for the pruning part. Jiaqi Huang spent most of his efforts on all of our theoretical parts(which were jointly constructed by all of us) and acts as the main contributor to the feature robustness part. Chunyue Xue contributed to the Lottery Ticket Hypothesis part and all the related experiments, such as adding the adversarial training and testing codes, training models, plotting and analyzing.

References

[1] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, "Adversarial examples are not bugs, they are features," 2019. arXiv: 1905.02175 [stat.ML].

[2] J. Frankle and M. Carbin, *The lottery ticket hypothesis: Finding sparse, trainable neural networks*, 2018. arXiv: 1803.03635 [cs.LG].

[3] Y. Huang, Y. Cheng, A. Bapna, O. Firat, M. X. Chen, D. Chen, H. Lee, J. Ngiam, Q. V. Le, Y. Wu, and Z. Chen, "Gpipe: Efficient training of giant neural networks using pipeline parallelism," 2018. arXiv: 1811.06965 [cs.CV].

[4] A. Krizhevsky *et al.*, "Learning multiple layers of features from tiny images," Citeseer, Tech. Rep., 2009.

[5] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *CoRR*, vol. abs/1312.6199, 2013.

[6] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *ArXiv preprint arXiv:1412.6572*, 2014.

[7] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *ArXiv preprint arXiv:1706.06083*, 2017.

[8] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," *ArXiv preprint arXiv:1705.07204*, 2017.

[9] D. Arpit, S. Jastrzebski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio, *et al.*, "A closer look at memorization in deep networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, JMLR. org, 2017, pp. 233–242.

[10] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, *Understanding deep learning requires rethinking generalization*, 2016. arXiv: 1611.03530 [cs.LG].

[11] P. Nakkiran, "Adversarial robustness may be at odds with simplicity," *ArXiv preprint arXiv:1901.00532*, 2019.

[12] C. Xie and A. Yuille, *Intriguing properties of adversarial training*, 2019. arXiv: 1906.03787 [cs.CV].

[13] J. Wang, R. Jia, G. Friedland, B. Li, and C. Spanos, *One bit matters: Understanding adversarial examples as the abuse of redundancy*, 2018. arXiv: 1810.09650 [cs.LG].

[14] J. Cosentino, F. Zaiter, D. Pei, and J. Zhu, *The search for sparse, robust neural networks*, 2019. arXiv: 1912.02386 [cs.LG].

[15] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli, "Evasion attacks against machine learning at test time," in *Joint European conference on machine learning and knowledge discovery in databases*, Springer, 2013, pp. 387–402.

[16] C. Molnar, *Interpretable machine learning, A guide for making black box models explainable*. 2019, https://christophm.github.io/interpretable-ml-book/.

[17] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/.

[18] A. Krizhevsky *et al.*, "Learning multiple layers of features from tiny images," Citeseer, Tech. Rep., 2009.

[19] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.