Few Features Attack to Fool Machine Learning Models through Mask-based GAN

Feng CHEN
Institute of Automation
Chinese Academy of Sciences, China
Email: chenfeng@ia.ac.cn

Yunkai SHANG
College of Electrical and Information Engineering
Hunan University, China
Email: sykhnu@126.com

Jincheng HU
Institute of Automation
Chinese Academy of Sciences, China
Email: jincheng.hu@ia.ac.cn

Bo XU*
Institute of Automation
Chinese Academy of Sciences, China
Email: boxu@ia.ac.cn

Abstract—GAN (Generative Adversarial Networks, [1]) is a machine-learning-based generative approach, which can create artificial contents such as images, languages and speeches. Recent studies have shown that GAN can also be applied to generate adversarial attack examples ([2], [3]) to fool the machine-learning models. In comparison with the non-learning adversarial attack examples approaches, the GAN-based adversarial attack example approach can generate the adversarial samples quickly when facing a new sample after training, but meanwhile needs to perturb the attack samples in great quantities. To address this issue, we propose a new approach, named Few-Features-Attack-GAN (FFA-GAN). FFA-GAN has a significant time-consuming advantage than the non-learning adversarial attack samples approaches as it is based on the GAN architecture, and also has a better non-zero-features performance than the GAN-based adversarial sample attack approaches because of the introduction of the mask mechanism in the generator of GAN to confine the perturbations. Experiments are made respectively on the structured data sets KDD-Cup 1999 and CIC-IDS 2017, in which the dimensions of the data are relatively low, and also on the unstructured data sets MNIST and CIFAR-10 in which the data have the relatively high dimensions. The results of the experiments demonstrate the effectiveness and the robustness of our proposed approach.

I. Introduction

GAN (Generative Adversarial Networks, [1]) is an unsupervised learning method and a deep model architecture which consists of a generative model and a discriminative model. The principle of GAN is to build a zero-sum game through the competition of two neural networks to reach Nash equilibrium to learn to create the complex distribution of the instance. Recently, GAN is more and more widely used in many domains, not only limitedly in the fields of the generation of images ([1], [4], [5]), speeches [6] and languages [7] , but also in the other fields like adversarial examples ([2], [3]) and malware generation [8]. This paper is going to discuss the application of an improved GAN method to generate attack adversarial examples to attack the machine-learning models, especially concerning about the issue of reducing the

perturbation when producing the attack adversarial examples using the GAN architecture.

Adversarial example is a sample example which has been modified additionally, and its goal is to make the machine-learning models to misclassify it. There are previously many studies in the field of generative attack example. [9] explained the concept of adversarial examples. It refers to an input sample formed by deliberately adding subtle perturbation in the data set, which causes the model to give a false prediction with high confidence. [10] proposed fast gradient sign method (FGSM) that can produce the subtle perturbation, and it is a white-box attack method. JSMA [11]is also a white-box attack method, which calculates L_2 and L_∞ to build the saliency map to limit the perturbation in a small range. [12] proposed a black-box attack approach which can use only very few pixels to implement the adversarial sample attack using the evolutional algorithm. The drawback of the optimization-based approaches is that these approaches are not the learning methods so that we have to consume too much time to calculate the perturbation for the attack sample when facing a new attack sample.

For the learning approach, GAN is an alternative in the adversarial sample because of its good generative ability. [2] proposed a method (AdvGAN) to generate adversarial examples through GAN. They implemented both of the white-box and the black-box attack method, the method only was used in image processing. [3] proposed IDSGAN that can generate adversarial attacks and they generated adversarial malicious traffic for the IDS (Instrusion Detection System) data set through a black attack approach, but they completely replaced almost all features. These two approaches both require lots of features being changed when attacking.

In this paper, we propose a new improved GAN architecture named Few-Features-Attack-GAN (FFA-GAN) to implement the adversarial attack to produce perturbations which are then added in the attack instance to fool machine learning classifier models. Instead of using the great mass of features of the original attack instance, only few features of the original attack instance need to be changed because of the introduction of the
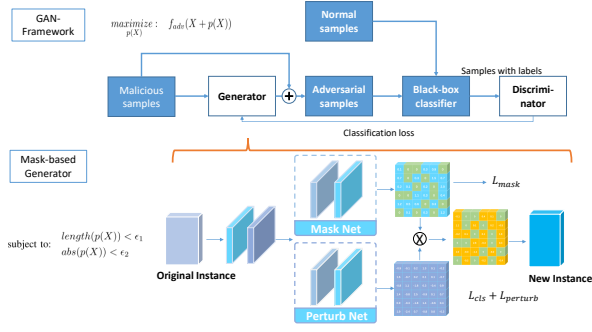
*Corresponding author

Fig. 1. FFA-GAN

mask mechanism in the generator network. The generator is composed of the mask network and the perturbation network. The mask network defines which features of the changeable features on the instance can be perturbed during the training, and the perturbation network provides the perturbation of the all changeable features on the instance. Here the output mask matrix of the mask network is not the binary matrix like the mask in the image domain, but the matrix filtered by ReLU non-linear module. In this way, the gradient of the whole network can not be truncated by the binary mask. Through the element-wise production of two outputs matrix from the mask network and the perturbation network, we can get the generator output as the perturbation (padding is needed if the dimensions of generator output are smaller than the all dimensions of the sample). The losses of the generator are composed of the classification loss and the mask loss, which can be regarded as L_0 norm, and the loss of the discriminator is the classification loss for the instance. With the help of the GAN framework, the generator and the discriminator build a zero-sum game to compete with each other, and reach an equilibrium to get the final perturbation with the few changed featured and the high bypass rate at the end.
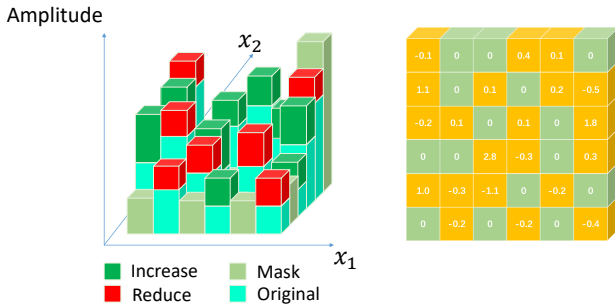

Fig. 2. Visualization of the perturbation

To train the parallel networks, the losses weights should be adaptively changed so that the training can converge. At the beginning of the training, we set the weight of the

classification loss large and the weight of the mask loss small. As the training goes, the weight of the classification decreases gradually and the weight of the mask loss decreases. We can regard the two phases as "the exploration phase" and "the exploitation phase" respectively. That means we first explore a good result in a relatively wide range and then optimize the range as small as possible to exploit a better result.

Our main contributions of the proposed FFA-GAN approach can be concluded as follows:
(1) In FFA-GAN, we introduce the mask mechanism into the generator network. The mask mechanism provides a filter to reduce the number of changed features or the perturbation of the samples and meanwhile without reducing the bypass rate through the GAN architecture.
(2) We define the losses of the generator in GAN which are composed of the classification loss, and the loss of non-zero-features' length of perturbations in the mask network. During the training, we change the weights of different losses adaptively, and ensure the convergence of FFA-GAN and achieve a better performance than the method with the fixed loss weights.

II. PRELIMINARY

We define generating adversarial attack samples to a target machine-learning model as a constraint optimization problem. The task of an adversarial attack sample is to modify the original attack sample as little as possible to make the target model misclassify it as the normal one. Based on these, we assume a target machine-learning classifier model L, the sample can be expressed as a vector of scalar features $X = \{x_1, x_2, ...\}$. The perturbation of the adversarial sample can be represented as $p(X) = \{x'_1, x'_2, ...\}$, which will be added in the original sample as the new sample.

$$X' = X + p(X) = \{x_1 + x'_1, x_2 + x'_2, ...\} \quad (1)$$

The model L can classify the sample if it is malicious or not or whether it is the target class. We use $f(X)$ to express the ability of the model to classify the attack sample as the attack sample. To describe the ability of the adversarial ability, we here define the goal function as the bypass rate as follows:

$$rate_{bypass} = max(1 - f(X')/f(X), 0) \quad (2)$$

The goal of generating adversarial attack is to maximize the high bypass rate:

$$\underset{p(X)}{maximize} : rate_{bypass} \quad (3)$$

While maintaining the high bypass rate, the difference of the adversarial attack sample and the original attack sample should be as little as possible. Therefore the constraints of the task can be describe as the following form:

$$\text{subject to:} \quad \begin{array}{l} length(p(X)) < \epsilon_1 \\ abs(p(X)) < \epsilon_2 \end{array} \quad (4)$$

In the domain of the adversarial examples, untargeted attack is an attack that fools the model to predict the sample as the

type which is not the designated. Here y is the attack class label.

$$f(X') \neq y \tag{5}$$

Correspondingly, the goal of the targeted attack is to predict the sample as the target type. Here t is the target class label.

$$f(X') = t \tag{6}$$

Black-box attack is an attack that the attacker does not know the details of the inner information of the target model, such as the parameters, the structures and so on. On the contrary, the every details of the target model are known to the attacker in the white-box attack. We discuss in this paper only the black-box attack. White-box attacks are not considered in our scenario.

III. METHODOLOGY

To produce the adversarial samples using the GAN architecture, we adopt the approach which is similar to ADV-GAN [2]. Based on the data set, we first train several machine-learning-based classifiers as the target models or black-box classifiers using the ground truth labels from the data set. The classifiers can be deep models or other non-deep models like random forest, decision tree and so on. Then a GAN-based architecture is build, in which the generator's input is the malicious samples or the samples that are to disguise, and the discriminator is fed with the malicious samples added with from the generator produced perturbations, and the task of the discriminator is to distinguish it whether the samples are malicious or not or the target type or not.
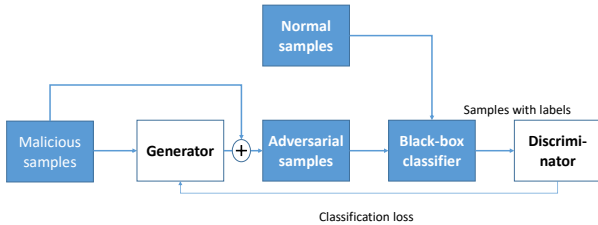

Fig. 3. GAN Framework

The noise can also be the part of the input of the generator to exploit better results. In our scenario, we do not use the noise in the input of the generator because the results can be more stable and reproduceable without adding the noise. The output of the generator network is the perturbation, whose feature dimensions can be different with the feature dimensions of the sample. If so, padding zeros of the output of the generator to the same dimension as the original sample is needed. The produced perturbations are added on the original samples and the elements of the adversarial samples should be limited in the valid range and then fed into the black-box classifier

with the normal samples. The predictions of the black-box classifier for the samples are regarded as the labels for the discriminator, which can be trained according to these labels to get closer and closer to the black-box classifier. The above mentioned procedure is done recursively many times through the GAN architecture until this min-max game finally comes to a equilibrium, which means producing the perturbation that can disguise the original sample and fool the black-box classifier.

$$\min_{G} \max_{D} V(D,G) = E_{x \sim p_{data}(x)}[logD(x)]$$
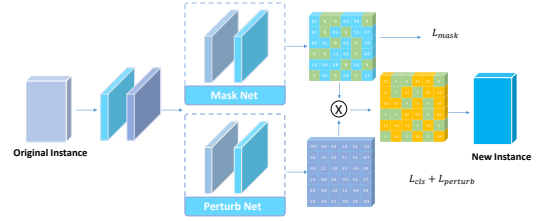$$+ E_{z \sim p_z(z)}[log(1 - D(G(z)))] \tag{7}$$

Fig. 4. Generator Network

Based on the basic GAN Formular 7, there are also extra constraints for the perturbation, the amplitude and the length of the perturbations, which are $length(p(X)) < \epsilon_1$, $abs(p(X)) < \epsilon_2$ as above mentioned. To address this constraint problem, we design a new neural network for the generator which is composed of two parallel parts: the mask network and the perturbation network. The input of generator is a feature vector, which is put into several neural layers to get latent variable z of the original sample, and then the latent variable z is fed into the mask network and the perturbation network respectively, and then through the element-wise production of two vectors of outputs of two parallel networks, we get the perturbation of the original sample. The sample is then added on the original sample to get the final attack sample.

The task of the mask network is to select the features of the perturbation that matter to the classifier. Unlike the traditional binary mask mechanism, we confine the output of mask network in the range $[0, +\infty)$ using ReLU non-linear module instead of the binary feature as in this way the gradient can be backpropagated through the mask network to the front neural layers to train the mask network. There is no labels for the mask network like in computer vision [13], therefore we train the mask network through the predefined the mask sparsity losses. The unimportant features of perturbations can be presented by zeros in the output. The output of perturb network is multiplied with the output of the mask network to produce the perturbation. And the mask network's parameters are updated by the GAN structure.

The generator losses are defined as follows:

$$L_{generator} = L_{clf} + L_{mask} \tag{8}$$

where two losses are:

$$L_{clf} = cel(f(X), f(X')) \tag{9}$$

$$L_{mask} = sum(abs(sign(p(X)))) = L_0 \tag{10}$$

Here L_{clf} is the cross entropy loss of classification, L_{mask} represents the loss of no-zero-features' length of perturbations the L_0 regularizer.
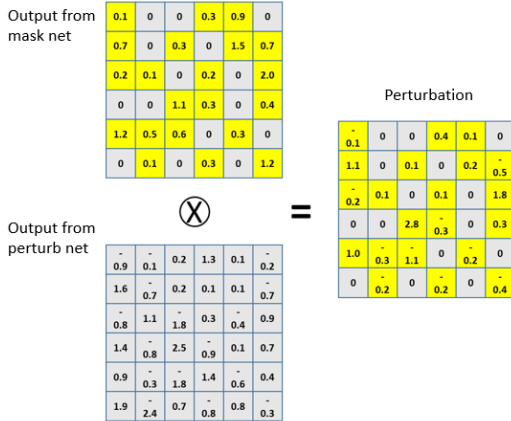

Fig. 5. Element-wise production of two outputs of two nets

Because of the two parallel networks in the generator network, it is hard to train FFA-GAN networks. This parallel structure brings the unstabilization during the training. So FFA-GAN needs a specific training method. First of all, the loss of L_{mask} should be set a small value, so that the GAN can easily find the perturbation in a very wide range. Then, we increase the weight of L_{mask} gradually to reduce the length of the changed feature number. Finally, we set a stop training condition, which is over a threshold bypass rate and meanwhile maintains the changed features of all features smaller than ϵ, a predefined small value. The principle is that we first explore a good result in a relatively wide range, which means the great number of the changed features is tolerant in the early training phase. Then the range of the changed features is optimized as small as possible to exploit a better result. These two phases can be regarded as "the exploration phase" and "the exploitation phase" respectively.

IV. EXPERIMENT

To show the performance of the proposed approach, experiments using FFA-GAN are made on the following structured and unstructured data sets.

A. Structured Data Set

1) i. KDD-Cup 1999: KDD-Cup 1999 [14] is a data set used for a big data analysis competition on the Fifth International Conference on Knowledge Discovery and Data Mining in 1999. The data were collected by MIT Lincoln Labs for the 1998 DARPA Intrusion Detection Evaluation Program. The Intrusion Detection System is a tool in the network system, which can surveille and analyse the status of the network system to detect and evaluate the possible and potential attack samples. Each data sample on KDD-Cup 1999 contains 41 features extracted from nine weeks' attacks and the normal traffic of the raw TCP dump data for a local-area network (LAN) built by DARPA, of which 9 features are symbolic and 32 features are continuous. The attack samples are composed of four categories: DOS (denial-of-service), R2L (unauthorized access from a remote machine), U2R (unauthorized access to local superuser privileges) and Probing (surveillance and other probing).
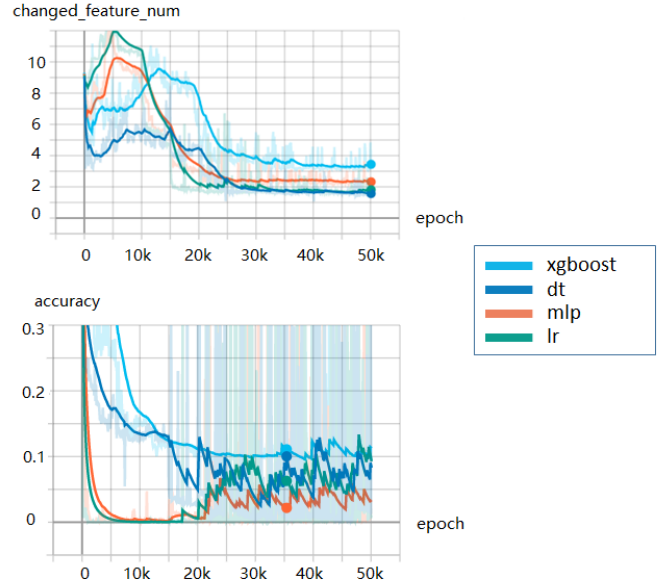

Fig. 6. Training curves of FFA-GAN for the classifiers on KDD-Cup 1999

The experiments of FFA-GAN on the data set KDD-Cup 1999 are made as follows: First of all, we preprocess the raw data by scaling the features in the range [0,1]. And then we randomly split the data set into two parts: 4/5 and 1/5 of all data as the training data and the testing data respectively. Based on these, four classifiers are trained to classify the data using different algorithms as the target black-box models, which are Decision Tree([15]), Logic Regression([16]), Multi-Layer Perception([17]) and XGBOOST([18]).

The 32 continuous features are chosen for the output of the perturbation of the generator in the FFA-GAN, which are more changeable and meaningful than the other 9 symbolic features in the reality. As the dimensions of the output of the generator are smaller than the dimensions of the original sample, padding zeros is needed. Figure 6 shows the training curves for the xgboost classifier. We set different weights in the epoch of 5000, 7500, 10000, 15000 and 20000 to train the FFA-GAN. The targeted attack for the normal label makes sense in this scenario because in reality, the attack samples are what we want to disguise as the normal samples to fool the classifier.

It is obvious that in Figure 6 the number of changed features decreases, meanwhile the accuracy of the four classifiers falls from over 99% to less than 10%. The first graph shows very clear that in each predefined epoch (5000, 7500, 10000, 15000 and 20000), there are a significant change of the number of changed features. The score curves gradually decrease and stay stabile no matter the weights change, which demonstrates the stabile results of the training procedure at the end.

Classifier	acc	FFA-GAN		IDS-GAN	
		acc*	len	acc*	len
dt	99.9%	8.1%	**1.6**	0.4%	
lr	94.2%	6.0%	**1.7**	0.6%	**20.5**
mlp	99.9%	3.0%	**2.2**	0.7%	
xgboost	99.6%	10.3%	**3.2**	-%	

("acc*" means the classifier's accuracy after the adversarial attack. "len" means the length of non-zero features of the perturbation. The attack IDS-GAN contains no "Probing".)

The total results of the FFA-GAN on KDD-Cup 1999 for the four classifiers can be concluded as shown in the Table I. From the table, we can see a enormous accuracy rate fall after using FFA-GAN and only very few features need to be changed. For the approach IDS-GAN [3], the average changed number is over 20 features, ca. 45% of all features. In contrast, we use only 2-3 features (less than 7.5% of all features) to achieve a high bypass rate as in IDS-GAN.

2) ii. CIC-IDS 2017: Like KDD-Cup 1999, CIC-IDS 2017 [19] is also a data set about the intrusion detection system. The difference lies in that CIC-IDS 2017 is a relatively new data set and contains totally 78 features and 8 attack types for each sample. Excluding the discrete features, we implement FFA-GAN only on the continuous features. From the Table II, we can see the better results on the CIC-IDS 2017 than on KDD-Cup 1999. We can also see a enormous accuracy fallen after using FFA-GAN and only very few features need to be changed and meanwhile FFA-GAN achieves a very high bypass rate. This is also a proof that FFA-GAN generalizes on different data set with the low dimensional structured data.

Classifier	acc	acc*	len
dt	99.0%	4.0%	**3.2**
lr	91.3%	0.1%	**2.1**
mlp	99.1%	0.5%	**2.3**
xgboost	99.6%	0.1%	**2.2**

("acc*" means the classifier's accuracy after the FFA-GAN attack. "len" means the length of non-zero features of the perturbation.)

B. Unstructured Data Set

1) i. MNIST: MNIST data set [20] is a classic data set in the domain of image processing. The images were collected by National Institute of Standards and Technology (NIST) and contains 60,000 training images and 10,000 testing 28 pixels * 28 pixels images, which are composed of 10 arabic number. The target classifiers are vgg16 [21] and resnet34 [22]. Both have the classification accuracy over 99.0% on MNIST data set. As the untargeted attack is easier than the targeted target, we only implement FFA-GAN to attack the classifiers in the targeted way to test the FFA-GAN approach. Table III shows the different approaches' results of FFA-GAN based on the different target classifier. The final average perturbations' length of FFA-GAN is smaller than 3% of all pixels and ca. 99.5% samples can be classified incorrectly as the predefined target type. In contrast, the ADV-GAN needs to change a great lot of features to achieve the high bypass rate, for the one pixel attack approach we need to use 30 pixels attack to achieve ca. 83% bypass rate based on the vgg16-classifier. Figure 7 demonstrates the attack examples on MNIST using FFA-GAN.

Classifier	acc	FFA-GAN		ADV-GAN		OP Attack	
		acc*	len	acc*	len	acc*	len
vgg16-targeted	99.5%	0.3%	**25.6**	0.1%	**525.6**	16.0%	**30**
resnet34-targeted	99.3%	0.5%	**24.4**	0.2%	**530.1**	-%	**-**

("acc*" means the classifier's accuracy after the adversarial attack. "len" means the length of non-zero features of the perturbation.)

2) ii. CIFAR-10: CIFAR-10 data set [23] is an image data set, which were collected by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. There are totally 60000 images in 10 classes, with 6000 32 pixels x 32 pixels colour images per class. The data set is composed of five training batches and one test batch, each with 10000 images. The difference between the CIFAR-10 data set and the MNIST data set is their channels: the image of CIFAR-10 has 3 channels and the image of MNIST has only one channel, which means it is more difficult to train FFA-GAN on the data set CIFAR-10 than on MNIST. Like the other experiments, the classifiers for the data set are firstly trained. Here we also use vgg16 and resnet34, which both have over 93% classification accuracy on CIFAR-10. After using the FFA-GAN approach, the accuracy rate of vgg16 and resnet34 has fallen to the 40% and 30% respectively, meanwhile only ca. 4.5% of all pixels are perturbed. ADV-GAN [2] uses over 50% of all pixels to achieve a high bypass rate. One pixel attack approach [12] needs to use 30-pixels attack to reduce the accuracy of the sample to 17.1% averagely. Figure 8 are the successful attack examples of FFA-GAN.

C. Summary

From the four experiments, we can draw the conclusion that FFA-GAN is an effective and robust approach to generate the adversarial samples on the structural and unstructural data set. In comparison with the GAN-based approaches IDS-GAN
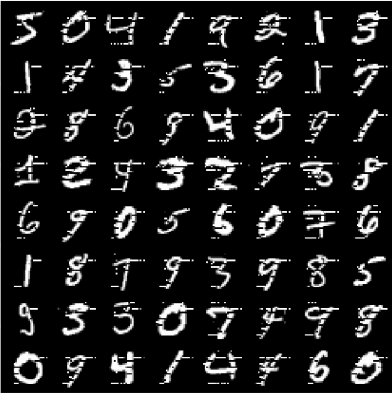
Fig. 7. Attack Examples in MNIST
(The upper images are the original samples. The bottom images are the FFA-GAN adversarial samples which are all classified as the target arabic number 5 based on the target resnet34-classifier.)

TABLE IV
FFA-GAN, ADV-GAN AND ONE PIXEL ATTACK TESTING RESULTS ON CIFAR-10

Classifier	acc	FFA-GAN		ADV-GAN		OP Attack	
		acc*	len	acc*	len	acc*	len
vgg16-targeted	93.2%	53.1%	**48.0**	14.3%	**551.3**	17.1%	**30**
resnet34-targeted	94.2%	60.2%	**51.4**	13.2%	**547.7**	-%	-

("acc*" means the classifier's accuracy after the adversarial attack. "len" means the length of non-zero features of the perturbation.)
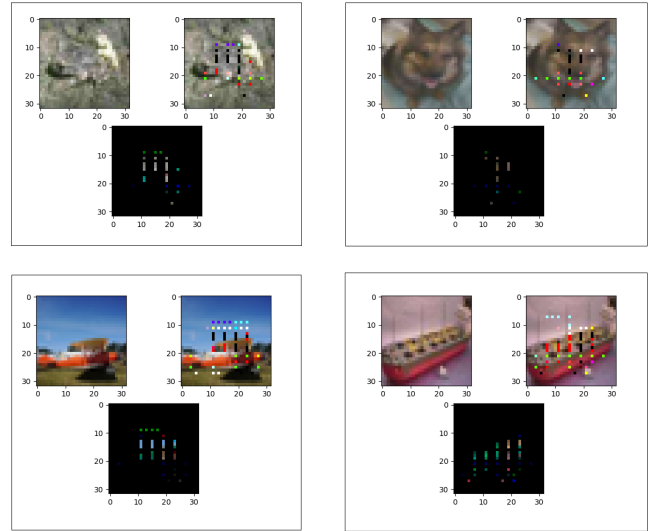

Fig. 8. Attack Examples in CIFAR-10
(In every frame, the left image is the original sample, the right image is the FFA-GAN adversarial sample, the bottom image is the perturbation. The above examples are all classified as the target type TRUCK based on the target vgg16-classifier.)

TABLE V
ONE PIXEL ATTACK AND FFA-GAN RESULTS FOR TESTING 100 IMAGES ON MNIST

	OP Attack						FFA-GAN	
iteration	acc-1	acc-5	acc-10	acc-20	acc-30	T(s)	acc	T(s)
75-vgg16	100%	91%	88%	64%	55%	466		
150-vgg16	99%	84%	77%	55%	43%	896	0.1%	0.1
300-vgg16	97%	84%	60%	39%	30%	2532		
75-resnet34	96%	48%	36%	22%	15%	305		
150-resnet34	95%	24%	18%	11%	5%	615	0.2%	0.1
300-resnet34	98%	16%	4%	2%	2%	1078		

("x-model" for OP Attack means the execution optimization iteration time based on the target model. "acc-x" for OP Attack means the classifier's accuracy after the x-pixels attack. Time is calculated per one image. The experiments are made on the computer with Win10, Memory 64GB, GPU 2080 and CPU i7-8700k.)

[3] and ADV-GAN [2], FFA-GAN can obviously reduce the length of the needed perturbations and meanwhile keep a very high bypass rate as shown in Table I, II, III, IV. In comparison with the traditional non-learning approaches One-pixel-attack [12], FFA-GAN can generate the adversarial samples using only once forward computing for the FFA-GAN after training. However, the other non-learning approaches need to use the classifier many times until the successful adversarial samples are found. For instance, one pixel attack approach spends much more time using the target classifier per samples to get the final results on MNIST as shown in Table V. Besides, the bypass rate FFA-GAN is better than One-pixel-attack. In the term of the sparsity, FFA-GAN's result is relatively worse than one-pixel attack in CIFAR-10. The reason might be that the mask, which decides the sparsity of the perturbation, is trained by the neural network and it is hard to train the L_0 regularizer. Therefore, the hyperparameters of the losses' weights, which are manually predefined, play an important role in FFA-GAN.

V. Conclusion and Future Work

In this paper, we proposed a Few-Features-Attack-GAN approach to generate adversarial examples to fool machine-learning models. FFA-GAN has a significant time-consuming advantage than the non-learning adversarial samples approaches and a better non-zero-features performance than the GAN-based adversarial sample approaches. Through the introduction of the mask mechanism, the number of changed features of the perturbation is constraint as few as possible. And we change the weights of the losses during the training to find a better result in a wide range firstly and minimize the range later. Experiments on the structured data sets and unstructured data sets results show the good performance of the FFA-GAN approach. For the future work, the change of the weights of the losses during training can be optimised using the optimization algorithms such as Population based Training method [24]. PBT can parallelly train GANs as a population in large scale and optimise the hyperparamers of the different losses through evolution algorithms. Besides, the work on exploiting the performance of FFA-GAN in the image with high resolutions is also worth to be studied in the future.

Acknowledgment

We would like to extend the deep gratitude to all those who have supported us practically, cordially and selflessly in writing this paper.

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, pp. 2672–2680, 2014.

[2] C. Xiao, B. Li, J.-Y. Zhu, W. He, M. Liu, and D. Song, "Generating adversarial examples with adversarial networks," *arXiv preprint arXiv:1801.02610*, 2018.

[3] Z. Lin, Y. Shi, and Z. Xue, "Idsgan: Generative adversarial networks for attack generation against intrusion detection," *arXiv preprint arXiv:1809.02077*, 2018.

[4] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International conference on machine learning*, pp. 214–223, 2017.

[5] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, "Least squares generative adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2794–2802, 2017.

[6] S. Pascual, A. Bonafonte, and J. Serra, "Segan: Speech enhancement generative adversarial network," *arXiv preprint arXiv:1703.09452*, 2017.

[7] L. Yu, W. Zhang, J. Wang, and Y. Yu, "Seqgan: Sequence generative adversarial nets with policy gradient," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[8] W. Hu and Y. Tan, "Generating adversarial malware examples for blackbox attacks based on gan," *arXiv preprint arXiv:1702.05983*, 2017.

[9] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

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

[11] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 372–387, IEEE, 2016.

[12] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Transactions on Evolutionary Computation*, 2019.

[13] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

[14] J. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P. K. Chan, "Cost-based modeling and evaluation for data mining with application to fraud and intrusion detection," *Results from the JAM Project by Salvatore*, pp. 1–15, 2000.

[15] S. R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," *IEEE transactions on systems, man, and cybernetics*, vol. 21, no. 3, pp. 660–674, 1991.

[16] I. Ruczinski, C. Kooperberg, and M. LeBlanc, "Logic regression," *Journal of Computational and graphical Statistics*, vol. 12, no. 3, pp. 475–511, 2003.

[17] I. D. Longstaff and J. F. Cross, "A pattern recognition approach to understanding the multi-layer perception," *Pattern Recognition Letters*, vol. 5, no. 5, pp. 315–319, 1987.

[18] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, ACM, 2016.

[19] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization.," in *ICISSP*, pp. 108–116, 2018.

[20] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[21] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[23] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," tech. rep., Citeseer, 2009.

[24] M. Jaderberg, V. Dalibard, S. Osindero, W. M. Czarnecki, J. Donahue, A. Razavi, O. Vinyals, T. Green, I. Dunning, K. Simonyan, *et al.*, "Population based training of neural networks," *arXiv preprint arXiv:1711.09846*, 2017.