Analysis of the attack effect of adversarial attacks on machine learning models

Guanpeng Su

Faculty of Education, University of Macau, Macau, 999078, China

mc15536@um.edu.mo

Abstract. The use of neural networks has produced outstanding results in a variety of domains, including computer vision and text mining. Numerous investigations in recent years have shown that using adversarial attacks technology to perturb the input samples weakly can mislead most mainstream neural network models, for example Fully Connected Neural Networks (FCNN) and Convolutional Neural Networks (CNN), to make wrong judgment results. Adversarial attacks can help researchers discover the potential defects of neural network models in terms of robustness and security so that people can comprehend the neural network models' learning process better and solve the neural network models' interpretability. However, suppose an adversarial attack is performed on a non-deep learning model. In that case, the results are very different from the deep learning model. This paper first briefly outlines the existing adversarial example technology; then selects the CIFAR10 dataset as the test data and LeNet, ResNet18, and VGG16 as the test model according to the technical principle; then uses the Fast Gradient Sign Attack (FGSM) method to conduct attack experiments with the CNNs and traditional machine learning algorithms like K-Nearest Neighbors (KNN) and Support Vector Machine (SVM); then analyze the experimental results and find that the adversarial example technology is specific to the deep learning model, but it cannot be completely denied that adversarial examples have no attack effect on traditional machine learning models.

Keywords: neural network, adversarial attack, adversarial example, deep learning.

1. Introduction

Machine learning are developing fast. Through the training and learning of source sample data through different neural network technologies, fake data generated by these models can already deceive the human eye. At the same time, these trained models can also identify the label of the target data with high confidence. All the above are of great help for people to generate and identify data daily.

However, with the broader application of neural network technology (such as autonomous driving, face recognition, and financial risk assessment), neural network models' accuracy, security, and robustness are constantly being tested [1-5]. Because in real application scenarios, the neural network model exposes the problem of poor anti-interference ability. Some researchers have found that only a slight perturbation of the neural network model’s input data is needed to generate adversarial examples, which makes the current mainstream neural network model produce wrong output results. For example, Kurakin et al. found that after printing the generated adversarial example images, the neural network model will produce different classification results under different light and orientation conditions [6].

Therefore, research on adversarial attack techniques came out. The input form of the machine learning algorithm is generally the numeric vectors, so the adversarial attack is to design a set of numeric vectors and input them into the machine learning model in a targeted manner, and the model makes a wrong judgment. These specifically designed data mislead machine learning models with input data called adversarial examples. Adversarial example technology generates adversarial examples almost imperceptible to humans. However, it can mislead the neural network model by slightly perturbing the original input data.

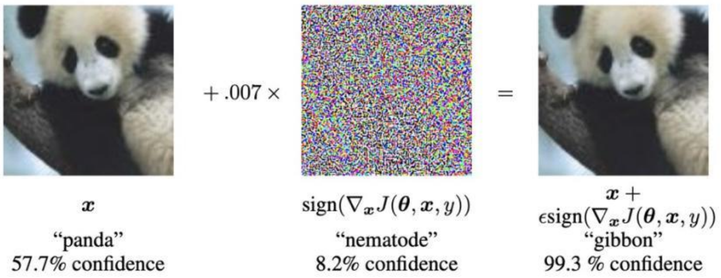
+ .007 ×
=
x
"panda"
57.7% confidence
$\text{sign}(\nabla_x J(\theta, x, y))$
"nematode"
8.2% confidence
$x + \epsilon\text{sign}(\nabla_x J(\theta, x, y))$
"gibbon"
99.3 % confidence

Figure 1. An example of adversarial example generated by GoogLeNe [7]. Slight perturbation mislead the GoogLeNet's judgment [8].

From the example in Figure 1 in the application field of image classification, It is simple to understand how a neural network model may have positively identified a picture of a panda. Next, a slight perturbation is added to this photo, resulting in a generated photo that looks almost indistinguishable from the original from a human perspective. However, inputting the generated photo into the same neural network model results in a high-confidence (99.3%) misclassification judgment (gibbon).

The study of adversarial example technology helps researchers discover neural network models' defects and solve interpretability problems. From the information security perspective, adversarial example technology can be classified as attack and defense technology. There are currently three attack techniques, namely: white box attack, black box attack, and grey box attack.

Szegedy et al. first proposed that adding slight perturbations to digital images can mislead neural network models into making wrong classifications [9]. Formula (1) illustrates how to generate adversarial examples in this method:

$$\begin{gathered}\text{minimize}\left\|x - x^{'}\right\|_2^2 \\ \text{s.t. } C(x') = t \quad \text{and} \quad x' \in [0,1]^m\end{gathered} \tag{1}$$

Since it is difficult to solve formula (1) directly, Box-constrained L-BFGS was used by Szegedy et al. to discover a rough solution [10]. As shown in formula (2):

$$\begin{gathered}\text{minimize } c\left\|x - x^{'}\right\|_2^2 + L\left(\theta, x^{'}, t\right) \\ \text{s.t. } x' \in [0,1]^m\end{gathered} \tag{2}$$

As can be observed, formula (2)'s optimization goal is to minimize the error between both the input image and the adversarial example x' while still getting the neural network model C to categorize the adversarial example x' as the incorrect label t. The antagonistic samples produced by the techniques are essentially impossible for the unaided eye to differentiate from the original pictures. However, they can successfully mislead the neural network model to produce wrong output results. At the same time, according to Szegedy et al., the adversarial examples produced by this method can play a misleading role in multiple neural network models, which indicates that there are blind spots in the learning mechanism of neural network models.

Although the neural network model produces errors due to attacks from adversarial examples, researchers can optimize the corresponding neural network model through the defense of adversarial attack techniques. The above is defense technology.

Adversarial attacks are relatively simple to operate, while defenses are challenging. To improve the defense success rate, it becomes essential to analyze the laws of adversarial attack methods. When the same adversarial attack method acts on different datasets, what are the commonalities of the resulting adversarial examples? Which is an important research topic.

Image recognition of handwritten digits is the most typical generative adversarial example. The researchers used FGSM in the LeNet neural network to attack a dataset of handwritten digits. However, it is not difficult to recognize the handwritten digital picture data set. Even if any neural network model is selected to train it, an accuracy rate close to 100% can be obtained in the handwritten digital image data set. Hence, attacking this simple task does not say much. There is a question here. Adversarial attack technology seems to be a unique phenomenon only when it comes to deep learning. Does the attack technology have the same effect on traditional machine learning methods? To analyze and study this problem, this paper designs and implements an experiment to observe the impact of adversarial attack techniques on traditional machine learning methods.

This paper selects the most representative CIFAR10 dataset in the color image dataset as the test dataset. Moreover, use deep learning models such as LeNet, ResNet18 and VGG16 for training and observe their accuracy. Next, FGSM attacks are performed on the former, and corresponding attack samples are obtained.

Then, these attack instances are employed in KNN and SVM, two conventional machine learning techniques, which have the potential to cause the deep learning model to make incorrect decisions. According to the experimental findings, the KNN experiment with various perturbation coefficients showed accuracy of almost 50%, and the SVM experiments all exhibited accuracy of 100%. Accordingly, it is clear from the results that adversarial attack examples will have some effect on conventional machine learning techniques.

2. Methods

2.1. Fast gradient sign attack

The attack method used in the experiments in this paper is FGSM (Fast Gradient Sign Attack). FGSM, a straightforward and very effective algorithm for producing adversarial examples, is one of the simplest and most well-known image adversarial attack techniques. FGSM was proposed by Goodfellow et al., and its purpose is to use the learning method and changes of the model itself to attack the neural networks [8]. They suggested a technique for producing adversarial examples quickly, as shown in formula (3):

$$x' = x + \varepsilon\,\mathrm{sign}\big(\nabla_x L(\theta, x, t)\big) \tag{3}$$

As shown in formula (4), by using one-step gradient descent. the formula (3) can be solved:

$$\begin{gathered}\text{minimize } L\big(\theta, x^{'}, t\big)\\ \text{s.t. } \ \|x' - x\|_\infty \le \varepsilon \ \text{ and } \ x' \in [0,1]^m\end{gathered} \tag{4}$$

The method searches the the original image's ε neighborhood for the perturbation signal value, enabling the adversarial example x' to be classified as the wrong label t. Compared with the method of Szegedy et al., the FGSM algorithm needs only one back-propagation process to generate adversarial examples.

During network training, FGSM learns the input image features and obtains the classification probability through the softmax or sigmoid layers. Then calculate the loss value with the obtained classification probability and the real label, return the loss value and calculate the gradient, that is, gradient backpropagation. To make the loss value greater than the loss value of the entire image when the changed image is fed into the classification network, it is simply necessary to add the calculated gradient direction to the input image.

Goodfellow pointed out that if the amount of change is in the same direction as the gradient, it will produce the most significant change in the classification result [8]. The sign function can be guaranteed to be in the same direction as the gradient function. Pursuing the minimizing of the loss function is the process of training deep neural networks. To find the loss function's minimum value, the gradient descent process moves in the opposite direction as the gradient. The FGSM approach can be compared to a gradient ascent algorithm because it moves in the gradient's direction to locate the maximum value of the loss function.

2.2. LeNet

LeNet is the simplest CNN model used in this paper. Yann LeCun and colleagues first put forth the model in 1989. It was successfully applied to detect handwritten zip codes after being initially used to distinguish handwritten numbers using a convolutional neural network and a backpropagation technique [11]. The U.S. Postal Service's ZIP code numbers were used in tests of their model in 1990, and the results revealed an error rate of only 1% and a rejection rate of roughly 9% [12]. They tested several handwritten digit identification techniques on the industry benchmark up until 1998, and the results revealed that their network outperformed all other models. Furthermore, after years of research and iteration, it was finally developed into LeNet-5 [13].

LeNet-5 includes seven levels. Except for the input layer, every layer has trainable parameters, numerous feature maps that to extract features from the input by convolutional filters, and multiple neurons on each feature map. The whole process can be roughly understood as: input->convolution->pooling->convolution->pooling->convolution->full link->full connected (output). LeNet-5 has the following characteristics: the activation function uses tanh; the convolution kernel is 5x5, the stride is 1, and padding is not used; the pooling layer uses MaxPooling.

2.3. ResNet 18

Kaiming He et al. in Microsoft Labs developed the ResNet network, which took first place in the target detection job and classification task of the ImageNet competition that year [14]. In the COCO dataset, it took first place in both object detection and image segmentation. ResNet suggests a residual structure module, a very deep network structure (more than 1000 layers), and batch normalization to hasten training. Prior to ResNet's invention, convolutional and pooling layers were superimposed in every neural network.

According to popular belief, the more convolutional and pooling layers used, the more complete the image feature information gathered and the more effective the learning process. However, it was discovered in the actual experiment that when the convolutional layer and the pooling layer were superimposed, not only did the learning impact not improve over time, but two issues also surfaced: 1) Gradient Explosion and Gradient Vanishing. 2) The degeneration issues.

The ResNet paper suggests employing BN (Batch Normalization) layers in the network and data preprocessing to address the issue of gradient disappearance and gradient explosion [14]. To address the degeneration issue in the deep network, it is possible to connect the layers of the neural network in a way that weakens the strong connections between each layer in certain of the neural network's layers. Residual Networks are the name given to such neural networks (ResNets). The ResNet-18 network used in this article has 18 layers. The number 18 denotes the depth of the network.[14]. In the experiments in this paper, based on efficiency considerations, the simpler the model, the better, so 18-layer ResNets are sufficient.

2.4. VGG 16

The Visual Geometry Group at Oxford University gave the VGG network first and second place finishes in the 2014 ImageNet Challenge for local and classification tracking, respectively. In a large-scale picture recognition setting, they investigate the impact of convolutional network depth on accuracy. Their primary contribution is to thoroughly assess the network's depth by utilizing 3 × 3 convolutional filters to extract features [15].

VGG has six configurations, denoted by the letters A, A-LRN, B, C, D, and E. D and E are the most often used configurations, being utilized in VGG16 and VGG19, respectively [15].

VGG16 contains 16 layers, including three fully connected layers and 13 convolutional layers. One pooling is utilized after the initial two convolutions using 64 convolution kernels, and the subsequent two convolutions employ two volumes of 128 convolution kernels. Pooling is utilized once again after the product has accumulated. Following a second round of pooling and three complete connections, three 512 convolution kernels are convolved twice. Tasks requiring classification and localisation perform well with the VGG16 model.

3. Experimental results and analysis

3.1. Data description

One of the most representative datasets for color picture datasets is CIFAR-10. Ten types of RGB color images are included in it: truck, plane, car, bird, cat, deer, dog, frog, horse, and cat. These categories are mutually exclusive, and images appearing in one category will not appear in other categories. Each of the 10,000 test images and the 50,000 training images in CIFAR-10 are 32 × 32 RGB three-channel images. Compared with the handwritten image dataset, the CIFAR10 color image dataset has higher complexity, a more decadent sample size, and a stronger representation. CIFAR-10 is chosen as the test data set since it is appropriate for this investigation.

3.2. Results and analysis

CIFAR-10 dataset is trained and tested using LeNet, ResNet-18, and VGG16 neural network models. Moreover, use FGSM to attack, get the experimental results.

Table 1. Attack effect of FGSM attack method on CNNs.

Epsilon	**0.05**	**0.1**	**0.15**	**0.2**	**0.25**	**0.3**
LeNet	0.1964	0.0613	0.0218	0.0089	0.0045	0.0028
ResNet18	0.0206	0.0175	0.0204	0.0202	0.0256	0.0364
VGG16	0.0596	0.0173	0.0166	0.0193	0.0263	0.0357

Using the LeNet model to train and test CIFAR-10, in the case of Epsilon=0, the accuracy is only 0.5336. It shows that the CIFAR-10 data set is much more complicated than the previously mentioned handwritten digit picture dataset. The ResNet-18 model has a correct rate of 0.7665 when Epsilon=0, which shows that the ResNet-18 model is better than LeNet. Of course, the capacity of the ResNet-18 model should be more significant. The accuracy of the VGG16 model is the highest among the three models, reaching 0.8002. When the FGSM attack is added, the accuracy of the three models shows a linear decline. It is not difficult to show that the FGSM attack is effective for the three models of LeNet, ResNet-18, and VGG16, As shown in Table 1. The next step of the experiment is to use the adversarial examples generated in the previous experiments as the training set, use the two non-deep learning models of KNN and SVM to test, and observe their accuracy.

Table 2. The predicted classification results of KNN and SVM after using the adversarial examples generated in the previous experiments as the training set.

		0.05	**0.1**	**0.15**	**0.2**	**0.25**	**0.3**
KNN	**LeNet**	0.4401	0.4954	0.5395	0.5779	0.6271	0.6771
	ResNet18	0.4810	0.4955	0.5054	0.5238	0.5402	0.5540
	VGG16	0.4625	0.4717	0.4766	0.4802	0.4842	0.4925

Table 2. (continued).

SVM	**LeNet**	1	1	1	1	1	1
	ResNet18	1	1	1	1	1	1
	VGG16	1	1	1	1	1	1

As the traditional machine learning classification method, KNN has no parameters and no learning and training process. It only measures the distance of the sample vector for classification [16]. So KNN is particularly suitable for use as a test here. The findings indicate that the accuracy of KNN for the three models with varying Epsilon values is approximately 0.5 when the adversarial instances produced in the prior tests are utilized as the training set. Moreover, the accuracy does not decrease with the increase of Epsilon. Since these samples are adversarial examples that lead to the wrong judgment of the CNN model in the previous experiments (the accuracies of the CNN models are all 0), the KNN prediction classification can have a correct rate of about 0.5, which shows that KNN does not need to learn and train to have the possibility of correct classification. It appears that these adversarial examples are specific to CNN.

The accuracy rates of using SVM unexpectedly reached 100%, which shows that this adversarial example is specific and only effective for deep learning or CNN models. However, 100% accuracy may mean that the model is overfitting. Therefore, in the next step of the experiment, the adversarial examples are divided into 8:2, 80% for training, while 20% for testing, as shown in Table 2.

Table 3. Training and testing results after dividing the adversarial examples by 8:2.

		0.05	**0.1**	**0.15**	**0.2**	**0.25**	**0.3**
KNN	**LeNet**	0.4364/0.24	0.5011/0.3323	0.5379/0.3682	0.5666/0.4305	0.6099/0.4712	0.6755/0.5075
	ResNet18	0.4798/0.2895	0.4960/0.3037	0.4951/0.3416	0.5179/0.3490	0.5305/0.3489	0.5553/0.4045
	VGG16	0.4639/0.2996	0.4689/0.2886	0.4748/0.2972	0.4746/0.3156	0.4810/0.3204	0.4894/0.3296
SVM	**LeNet**	1/0.4607	1/0.7185	1/0.8262	1/0.9076	1/0.9462	1/0.9388
	ResNet18	1/0.5670	1/0.7216	1/0.7971	1/0.8399	1/0.8482	1/0.8734
	VGG16	1/0.3954	1/0.4451	1/0.5198	1/0.5442	1/0.5827	1/0.6161

The KNN model exhibits a difference between the test and training data, as shown by the results in Table 3. However, the difference is not large, which is an acceptable result. In the SVM model, the data difference is quite large, so it is overfitting. Whenever people mention adversarial attacks, they always point to the mechanism risks of deep learning. Contrarily, few individuals are concerned about the risks of conventional machine learning models. At the same time, little attention has been paid to why adversarial attacks are specific to deep learning models. The SVM results in Table 3 show that the attack strength is 0.05, the adversarial examples generated by all models, and the examples generated by all attack strengths under the VGG16 model will lead to overfitting of the SVM. It also means that the risk of traditional machine learning models like deep learning is also worthy of attention by researchers.

4. Conclusion

The adversarial examples produced by FGSM are specific to the deep learning model itself, according to the experiments in this work. Experiments on KNN models without parameters confirm that these adversarial examples are specific to deep learning models. However, when experiments were performed on the SVM model with parameters that needed to be optimized, the results showed a regular overfitting phenomenon. Therefore, experiments cannot completely deny that these adversarial examples have no attack effect on traditional machine learning models. Adversarial examples make parametric machine learning models such as SVMs potentially risky. It can be seen from the experiments in this paper that the traditional machine learning model also has certain risks. Therefore, researchers must expand the research scope to more models that approximate deep learning, such as SVM, Decision Tree, and Random Forest. Analyze and compare the results to see if a more unified conclusion can be drawn.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in NIPS'12 Proceedings of the 25th International Conference on Neural Information Processing Systems, vol. 1, 2012, pp. 1097–1105.

[2] S. Ren, K.He, R.Girshick, and J.Sun, "Fasterr-cnn: Towards real time object detection with region proposal networks," in NIPS'15 Proceedings of the 28th International Conference on Neural Information Processing Systems, vol. 1, 2015, pp. 91–99.

[3] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in NIPS'14 Proceedings of the 27th International Conference on Neural Information Processing Systems, 2014, p. 31043112.

[4] H. Xu, M. Dong, D. Zhu, A. Kotov, A. I. Carcone, and S. NaarKing, "Text classification with topic-based word embedding and convolutional neural networks," in BCB '16 Proceedings of the 7th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics, 2016, pp. 88–97.

[5] C. Szegedy et al. ''Intriguing properties of neural networks.'' arXiv preprint arXiv: 1312.6199, 2013.

[6] A. Kurakin, I. Goodfellow, and S. Bengio, Adversarial examples in the physical world. arXiv preprint arXiv:1607.02533, 2016b.

[7] Szegedy, Christian, Liu, Wei, Jia, Yangqing, Sermanet, Pierre, Reed, Scott, Anguelov, Dragomir, Erhan, Dumitru, Vanhoucke, Vincent, and Rabinovich, Andrew. Going deeper with convolutions. Technical report, arXiv preprint arXiv:1409.4842, 2014a.

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

[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] Liu, D. C. and Nocedal, J. On the limited memory bfgs method for large scale optimization. Mathematical programming, 45 (1-3):503–528, 1989.

[11] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. Neural Computation, 1(4):541-551, Winter 1989.

[12] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Handwritten digit recognition with a back-propagation network. In David Touretzky, editor, Advances in Neural Information Processing Systems 2 (NIPS*89), Denver, CO, 1990. Morgan Kaufman.

[13] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.

[14] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[15] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[16] Qiu, X. Y., Kang, K., & Zhang, H. X. (2008, June). Selection of kernel parameters for KNN. In 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence) (pp. 61-65). IEEE.