applied sciences

MDPI

Article

Class Retrieval of Detected Adversarial Attacks

Jalal Al-afandi *,† **and Horváth András** †

Faculty of Information Technology and Bionics, Peter Pazmany Catholic University, Práter u. 50/A, 1083 Budapest, Hungary; horvath.andras@itk.ppke.hu

* Correspondence: alafandi.mohammad.jalal@itk.ppke.hu; Tel.: +36-702347357

† These authors contributed equally to this work.

Abstract: Adversarial attack is a genuine threat compromising the safety of many intelligent systems curbing the standardization of using neural networks in security-critical applications. Since the emergence of adversarial attacks, the research community has worked relentlessly to avert the malicious damage of these attacks. Here, we present a new, additional and required element to ameliorate adversarial attacks: the recovery of the original class after a detected attack. Recovering the original class of an adversarial sample without taking any precautions is an uncharted concept which we would like to introduce with our novel class retrieval algorithm. As case studies, we demonstrate the validity of our approach on MNIST, CIFAR10 and ImageNet datasets where recovery rates were 72%, 65% and 65%, respectively.

Keywords: adversarial attacks; neural network; class recovery; label retrieval

Citation: Al-afandi, J.; András, H. Class Retrieval of Detected Adversarial Attacks. *Appl. Sci.* **2021**, *11*, 6438. https://doi.org/10.3390/app11146438

Academic Editor: Antonio Fernández-Caballero

Received: 7 June 2021
Accepted: 6 July 2021
Published: 12 July 2021

1. Introduction

Deep learning is an ubiquitous machine learning approach which has been successfully used in many applications to find a practical solution for complicated problems, e.g., medical image segmentation [1] and self-driving cars [2]. Recently, deep learning has emerged as a cornerstone for safety and security-critical applications, like autonomous vehicles [3] and malware detection [4]. It even supports other applications, e.g., facial recognition, which may effect our daily lives.

One of the recent obstacles of using deep learning are adversarial attacks, which were first introduced in [5]. The authors in [5] present a case of a malevolent attacker who can exploit the high-dimensional inputs by slightly modifying the pixels approaching the border of the high-dimensional geometrical manifold of the original class [6]. Although the attacks can be very strong, reaching more than 90% confidence for misclassified classes, the modification which has been applied to the original image is imperceptible to the naked eye. Minor perturbation over the entire image is the first adversarial attack which has been introduced by Goodfellow [7]. Many threatening results have surfaced urging the research community to find some defense mechanisms, e.g., Ref. [8] demonstrates a universal perturbation fooling a classifier on any image, Ref. [9] showed the possibility of fooling a classifier with a 3D printed object.

The most commonly applied defenses against adversarial attacks depend on one of three approaches: adversarial training [10], modifying the network [11] or a detection-based approach [12]. Adversarial training slows down the training and it has also been demonstrated by [8] that you can form an adversarial attack against a network which has been trained with adversarial training. Most defenses which rely on modifying the network are too computationally infeasible to be applied on larger networks or they only work against specific attacks [13]. Detection-based approaches lack any notion of security in safety and time-critical applications where an immediate decision has to be taken without any delay, e.g., object detection in self-driving cars. Detection of adversarial attacks can be a good first step, but, on its own, it is not enough to ensure safety in practical applications

since detection will only render the autonomous system in complete doubt, preventing the AI from making a sound and reliable decision. The ultimate safe solution for adversarial attacks, in the case of accurate detection, is recovering the original class. We will introduce a novel algorithm restoring the original class of the attacked image. We will demonstrate the effectiveness of our algorithm on three different datasets MNIST, CIFAR10 and 10 randomly selected classes from ImageNet.

2. Adversarial Attacks

The term *adversarial example* was coined by [5], where attacks on neural networks trained for image classification were generated via a very low intensity additive noise, completely unobtrusive to the human eyes as in Figure 1. An adversarial example is a misclassified sample which has been modified with unnoticeable perturbations drastically changing the response of a network.

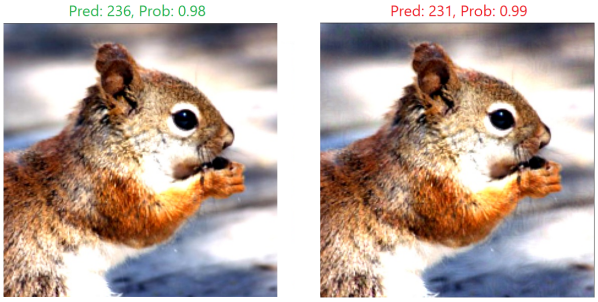

Figure 1. The figure illustrates an adversarial attack with a very low intensity perturbation entirely unperceivable to the naked eye.

The eminent high risk of adversarial attacks urged the science community to thoroughly investigate any potential attack enlarging the concept of adversarial attack. Three kind of attacks have been investigated (evasion attacks, poisoning attacks and exploratory attacks) where the first published adversarial attack [5] belongs to the first category. Evasion attacks are the most common attack in an adversarial setting where malicious samples are created to fool the network during the testing phase, causing a misclassification [7,14]. Poisoning attacks [15] contaminate the training dataset by injecting meticulously contrived samples compromising the training procedure. Exploratory attacks [16] are used when we only have a black-box access to the model and, thus, the attacker tries to collect as much knowledge as possible about the distribution of the training dataset and the network response which can be used to build a network that is capable of building adversarial samples. We will concentrate on the widespread white-box gradient-based attacks which we will investigate in our paper and briefly describe in this chapter.

2.1. Adversarial Attack algorithms

The first attacks [7] were implemented by calculating the sign of the gradient of the cost function (J) with respect to the input (x) and expected output (y), multiplied by a constant to scale the intensity of the noise (formally $\epsilon \text{sign} \nabla_x J(\theta, x, y)$, where *theta* is the parameters of the model) where the calculated value is the perturbation which is added to the original sample creating the adversarial image. This method, which is called the fast gradient sign method (FGSM), allows for much faster generation of attacks.

An extension to FGSM by [14] is to use not only the gradient's sing of the loss, but rather a scaled version of the gradient's raw value ($\epsilon \nabla_x J(\theta, x, y)$). In the same paper a targeted attack and an iterative basic method were proposed. A targeted attack can be achieved by steering the perturbation to the opposite direction of the gradient of the loss function ($J(\theta, x, y_{target})$) maximizing the probability of a specific class (y_{target}). Iterative-based method attack, which is also called projected gradient descent attack (PGD), is a targeted attack that can be applied N number of times with a small step size, taking into

account the intensity boundary where the perturbations calculated at each iteration are accumulated, leading to a misclassification.

Another extension to the iterative version of FGSM by [17] was to incorporate momentum into the equation, hypothesizing that, similarly to regular optimization, momentum helps in stabilizing the direction of the update and avoiding poor local minima or maxima, narrow valleys and other non-convex patterns in the objective function's landscape.

Ref. [18] proposed DeepFool attack to calculate minimum norm adversarial perturbations, taking the image to the edge of the decision boundaries of the classifier. It is similar to the basic iterative method in a sense that it relies on gradients, but DeepFool normalizes the gradient by linearizing the boundaries of the manifold where the image resides.

Many other variations of PGD attack have been formulated enhancing the quality and the speed of the adversarial attack [19,20]. A general overview of adversarial attacks, containing most of the previously mentioned methods, can be found in the following survey papers [13,21].

2.2. Adversarial Attack Detection

Defenses against adversarial attacks are required to prevent security threats in real-world application of neural networks. Most defenses rely on one of the following three main approaches:

- Modifying the training process, e.g., adding adversarial samples (adversarial training) [18] or modifying the input before testing (decision making) [22,23].
- Modifying the network, e.g., adding extra masking layer before the last layer [24] and changing the loss function by penalizing the degree of variety per class in the output [11].
- Using external model as a detector, e.g., SafetyNet [12] and convolutional filter statistics detector [25].

Detectors can be a reliable choice providing the highest accuracy reaching $\sim$ 85% [25] preventing most security breaches. There are a lot of other detectors [26,27] separating the clean image from the adversarial one by finding some distinguishable features and properties e.g. convolution filter statistics [25] and manifolds [26]. Although detectors are considered strong defenses against adversarial attacks, an attack can cause a halt in the system hindering the achievement of any task. In a time sensitive task e.g self-driving cars, where an on-the-spot decision has to be drawn, detectors are not sufficient and a retrieving approach has to be installed recovering the original class of the input.

3. Class Retrieval

Most non-detection defenses are vulnerable to counter-counter attacks [28], rendering potential exposure and keeping the system in a state of being without any functioning protective shield. Detection-based defenses, on the other hand, can be continuously updated but lack the ability to steer the decision-making process obstructing the installation of any safety measure. Thus, a recovery algorithm has to be employed after the detection of adversarial attacks, providing robustness and resilience.

Ref. [29] hypothesized that adversarial attacks exploit the edge of the decision boundary between classes, pushing the adversarial sample to the targeted class. Their idea stemmed from the speculation that training data will be pushed to the edge of the decision boundary once they are classified correctly. In [30,31], the authors assume that the reason behind the adversarial vulnerability of neural networks is the highly positively curved decision boundary where the curvature is very intricate near the classes borders. The high dimensionality of neural networks creates convoluted borders between all the classes, making a targeted adversarial attack highly possible. Taking into account the complexity of the curvature of the decision boundary, we hypothesize that the distance between the adversarial sample and the original class's manifold in the feature space of the decision boundary is smaller than the distance between the adversarial sample and any other classes' manifolds and, hence, all the adversarial samples and their counter attacks

are in the vicinity of the original class manifold. We have implemented our idea, a class retrieval algorithm, on the notion of our former hypothesis to predict the original class by counter attacking the adversarial samples, targeting every class and then selecting the class with the minimum loss. What we can derive from our hypothesis is that during the counter attack, it would be the easiest to transform back the attacked image to its original class since the attacked sample still contains many features belonging to the original class, and the manifold of the attacked class is the closest to the decision boundary of the original class, as we can observe in Figure 2. The counter attack can return the attacked sample to its original class easily since the adversarial sample is on the edge of the original class decision boundary. Due to the high dimensionality of the decision boundary curvature, there exist an intricate border between the manifold of each of the two randomly selected classes.

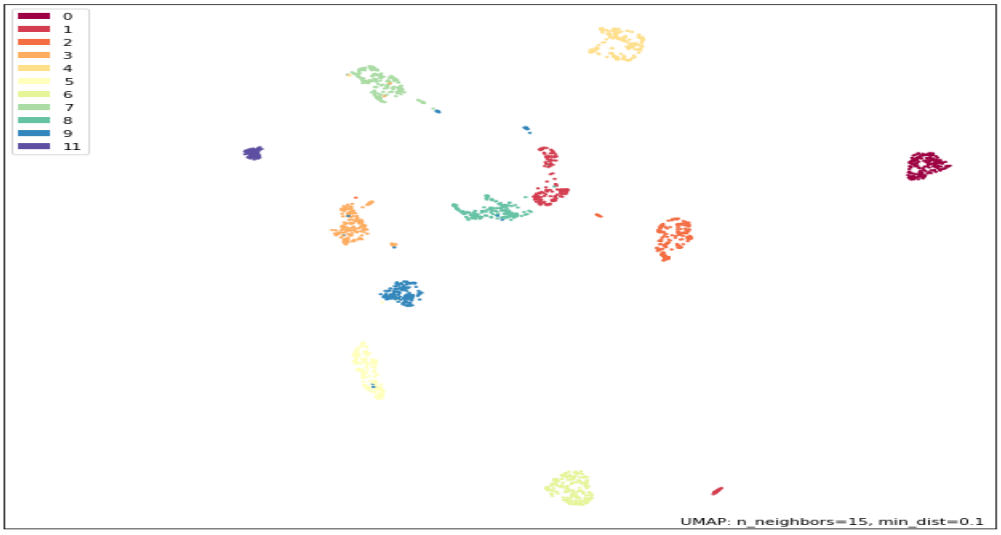

Figure 2. This figure displays a two-dimensional UMAP projection of the MNIST digits in the sklearn package with an additional 100 attacked samples which originally belonged to class 7 and were transformed to class 3 with the PGD algorithm. Each class is marked with a different color while the adversarially attacked samples are marked with purple. We can notice that the newly generated samples are between their original class and their new adversarial class, which led us to believe that going back to the original class would be the fastest. We generated similar figures for other classes as well, and the results were qualitatively the same in all cases.

To illustrate this hypothesis, we conducted experiments on the MNIST dataset, where 100 randomly selected samples from the same class were attacked (we considered them to be a separate class) and the two-dimensional position of their manifolds were depicted using the UMAP algorithm. An example can be seen in Figure 3, which confirms our assumptions that going back to the original class manifold can be achieved in fewer iterations than turning the sample to any other class, i.e., the cross entropy loss of a targeted class will be the smallest when targeting the original class.

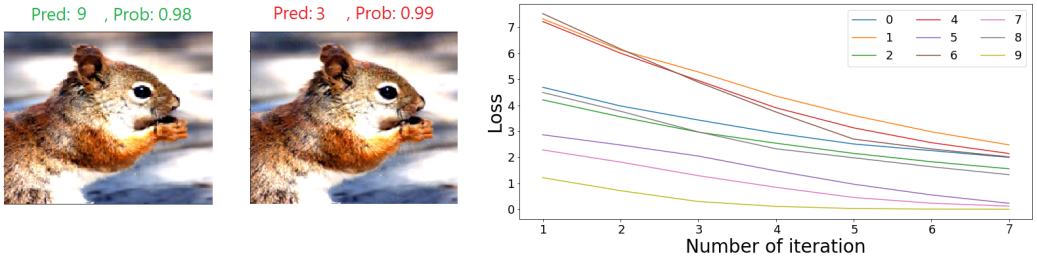

Figure 3. A showcase of our algorithm is illustrated in the figure using an image from ImageNet dataset, where the first two images show the clean sample, squirrel, and its adversarial version, where *Pred* is the index of the predicted class and *Prob* is the confidence. The last image depicts the losses of the counter adversarial attacks throughout 8 iterations, where the legends refer to the targeted classes. We can see that the index of the original label, nine, have the smallest loss since the first iteration.

Our class retrieval algorithm for a detected adversarial attack is presented in Algorithm 1 as a pseudo-code, and explained as a flowchart in Figure 4. *AdvImg* is the adversarial image which has been selected by an adversarial attack detector filtering any potential adversarial threat. The neural network prediction for the label of the adversarial image is *AdvLab*, which is a misclassification according to our detector. *NbClass* is a fixed parameter representing the number of classes in our classification problem. We apply a counter targeted attack using $Attack()$ function where *NbIter* is the number of iterations in the iterative adversarial attack and *Target* is the targeted label. The loss function, $loss()$, calculates the cross entropy loss of the counter adversarial image, *ContAdvImg*, having *Target* as a label. We exterminate the possibility of the adversarial label, *AdvLab*, being the original class by setting its loss to infinity. The original label, *OrigClass*, is the class with the minimum loss excluding the adversarial label where we used *argmin* function to return the index of the smallest loss.

Algorithm 1: Class retrieval algorithm for a detected adversarial attack

Parameters: $NbIter, NbClass, AdvImg, AdvLab$ **Result:** $OrigClass$
$Losses = 0, Losses[AdvLab] = \infty$
for $Target : 0$ **to** $NbClass$ **do**
  if $Target! = AdvLab$ **then**
    $ContAdvImg = Attack(AdvImg, NbIter, Target)$
    $Losses[Target] = loss(ContAdvImg, Target)$
  end
end
$OrigClass = argmin(Losses)$

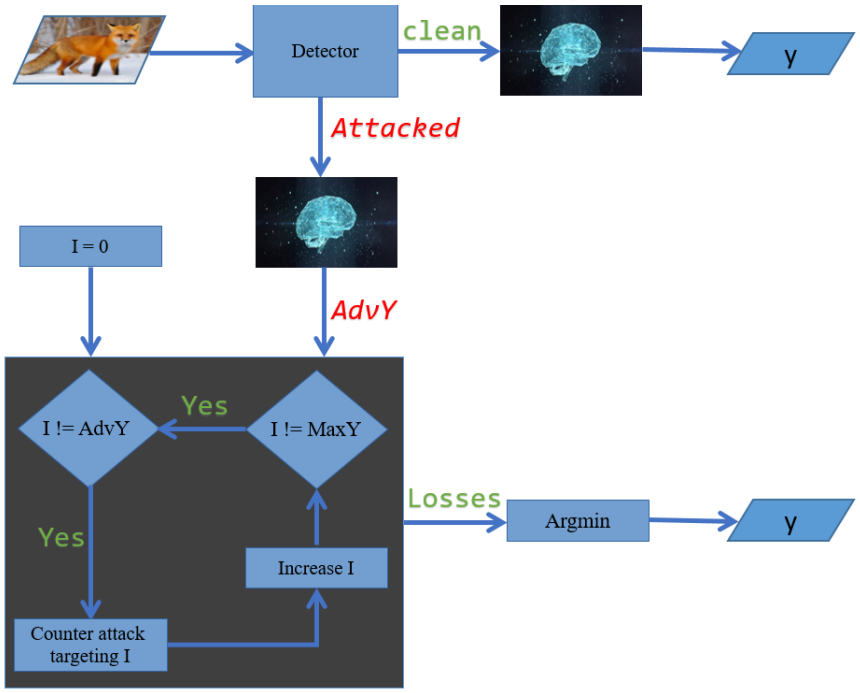

Figure 4. A flowchart explaining our class retrieval method starting from the input image until the output label where the *argmin* process returns the index of the minimum loss, *MaxY* is the number of classes and *AdvY* is the adversarial label.

To demonstrate the validity of our work, we assumed the existence of an optimal detector which can identify any adversarial attacks. We investigated four different adversarial attacks (projected gradient descent attack (PGD) [32], MPGD [17], Deepfool [18], TPGD [20], PGDDLR [19]; most of the attacks were adopted from Torchattacks library [33] while we used the codes of the original paper for the Deepfool attack) which were briefly explained in the previous section. Deepfool is not a targeted attack and the last two attacks are extended versions of the first attack; thus, we only used the first two previously mentioned attacks, PGD attack and PGD with momentum, as a counter adversarial attack. All the investigated adversarial attacks are white-box attacks, and this relies on the gradients to calculate the small perturbations fooling the classifier. For the sake of reproducibility, you can find our codes online https://github.com/Al-Afandi/class_retrieval (accessed on 7 June 2021), which include the chosen investigated parameters.

4. Results

4.1. MNIST

To validate our hypotheses, detailed experiments were conducted using the MNIST and other datasets, as we will see in the next paragraphs. MNIST is a commonly investigated dataset with ten classes containing 70 thousand images of handwritten digits with a 28×28 resolution. We investigated five different adversarial attacks (PGD, MPGD, Deepfool, TPGD and PGDDLR) to create a matrix of success rate with another two counter attacks (PGD and MPGD). In each case, we attempted to retrieve the class of 1000 successfully attacked samples, which means 10,000 experiments altogether were conducted. The maximum distortion of the adversarial attack was set to 0.2. The number of iterations, *NbIter*, is 10 for the adversarial attacks insuring a successful attack, but we set it to 3 for the counter adversarial attack, illustrating the fast convergence to the original class.

AlexNet architecture was used throughout our experiments, providing a good baseline network where the average accuracy on clean samples is 96% (the original 28×28 images were rescaled to 224×224, ensuring the required input size). Figure 5 demonstrates the high accuracy of the class retrieval algorithm investigating the usage of two counter attacks against the adversarial samples of five different attacks. On average, 72% of the

attacked samples were correctly recovered, predicting their original class and averting misclassification.

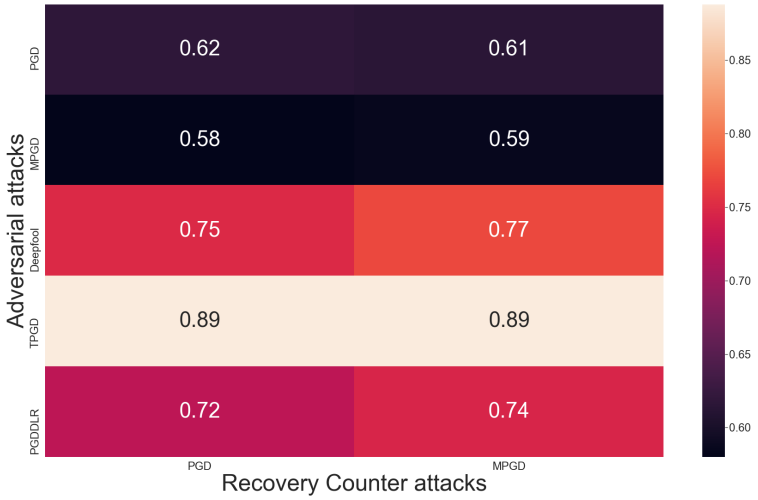

Figure 5. The figure illustrates the success rate of our class retrieval algorithm on the MNIST dataset, where each cell represents the accuracy of the retrieval in a specific setup, i.e., the algorithm used for the attack can be seen in the rows and the algorithm used for the counter attack can be found in the columns.

4.2. CIFAR10

We investigated another simple but more intricate dataset, CIFAR10, to show the effectiveness of our novel approach. The same setup described in detail in the previous paragraph was used. The only parameters which were significantly modified were the adversarial attack maximum distortion and number of iterations *NbIter*, where we set the former to 0.05 and the latter to 5. We opted to use these smaller values in comparison to our setup with MNIST due to the faster and easier conversion to adversarial samples. Figure 6 shows the success rates using our class retrieval algorithm over the CIFAR10 dataset, engulfing ten different setups. We used the ResNet-18 architecture throughout our experiments, providing a good baseline with 93% accuracy in classifying clean samples. The overall averaged retrieval accuracy is 65%, which demonstrates the viability of our approach.

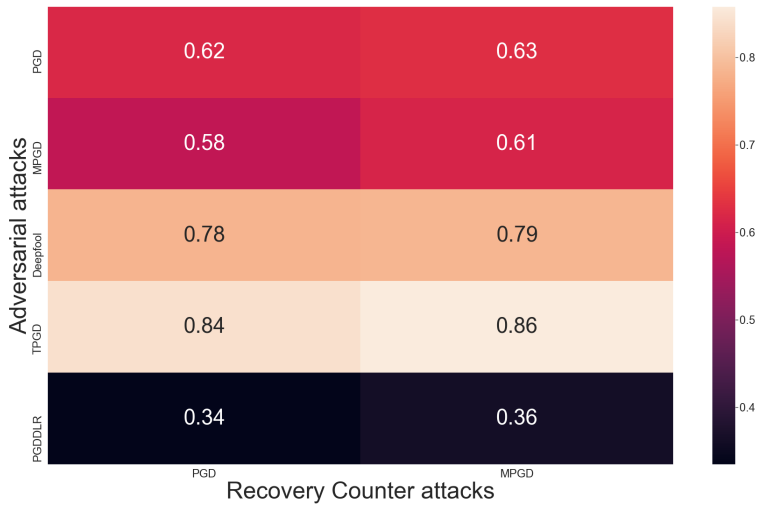

Figure 6. The figure illustrates class retrieval success rates on the CIFAR10 dataset, where each cell represents the accuracy of the retrieval in a specific setup, i.e., the algorithm used for the attack and the algorithm used for the counter attack.

4.3. ImageNet

Our algorithm can be applied, in practice, with datasets that contain a limited number of classes N, because of the nature of the algorithm ($N-1$ number of counter attacks have to be made). To investigate complex and more practical datasets with high-resolution images, we have randomly selected 10 classes from ImageNet to execute similar experiments as in case of MNIST and CIFAR10. We did not use Deepfool as an adversarial attack because only targeted adversarial attacks can be used due to the fact that we can only target one of the ten selected classes. Altogether, 8000 attacks and retrievals were made and, to balance the effect of random class selection, we selected ten different classes for each 100 attacks. Throughout our investigation, we used the pretrained version of $Inception_{v3}$ architecture from the torchvision models library. The inception model has one thousand possible output classes, but our adversarial and counter attacks were only targeting the randomly selected ten classes. We can see the success rate on ImageNet in Figure 7, where each cell represents the accuracy of a specific attack and counter attack investigating thousand cases with 10 different random classes for each hundred trials. There was a 65% average accuracy of recovering eight thousand attacked images from ImageNet using our class retrieval.

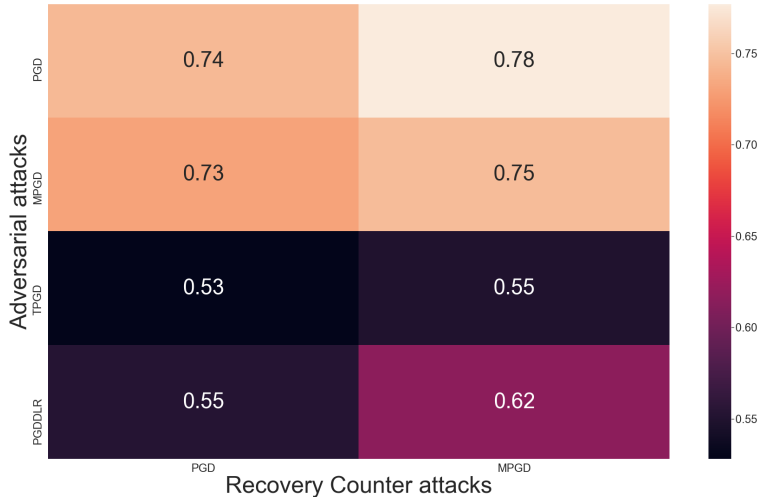

Figure 7. The figure depicts the success rate of our adversarial retrieval on ImageNet dataset, where each cell represents the accuracy of the retrieval in a specific setup investigating 1000 cases selecting 10 random classes for each hundred trials.

4.4. Parameters Investigating

Our class retrieval algorithm has two important parameters, number of iterations (*NbIter*) and number of classes (*NbClass*). *NbIter* parameter does not have much effect on the algorithm according to our measurements, and it gave the same results after the first iteration, as we can observe in Figure 3. *NbClass* parameter is essential to the algorithm as we have to conduct ($NbClass-1$) counter attacks. Statistically, the accuracy will drop when having a large number of classes preventing a correct retrieval. We conducted an extensive investigation over the CIFAR10 dataset, evaluating the success rate of adversarial attacks having different number of classes, from four to ten classes. We used AlexNet architecture to train a classifier for the randomly selected classes (4 to 10 classes), and we then calculated the average of the success rate of our class retrieval for each classifier (evaluating the retrieval of 10k adversarial attacks) to create a plot, Figure 8, illustrating the relationship between the accuracy of our algorithm and *NbClass* parameter. We can see that the algorithm is functioning effectively with a relatively high number of classes, but the accuracy will eventually decrease when processing a huge number of classes. Although a high number of classes is a limitation of our method due to time consumption and statistical probability, in most practical applications, only a few output classes are used.

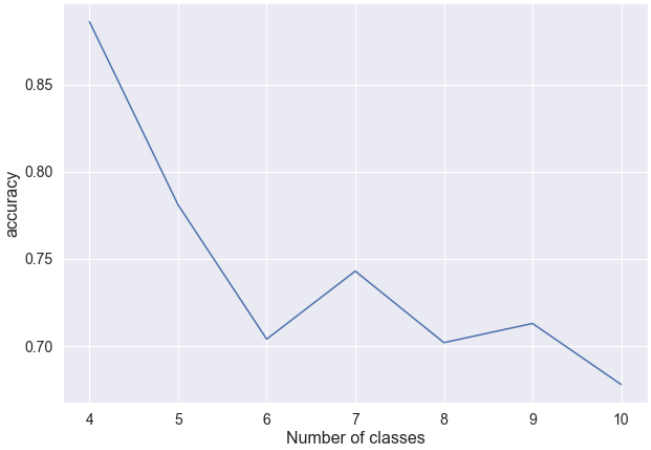

Figure 8. The figure depicts the relationship between class retrieval algorithm and number of classes parameter. It is clear that the algorithm is still working well with an expected drop in the accuracy.

Another parameter which can effect our method is the size of the input space. While low-resolution images, e.g., MNIST, are difficult to attack with small bounded unnoticeable perturbation, high-resolution images, e.g., ImageNet, are easy to attack because of their high dimensionality, rendering a complicated but vulnerable decision boundary curvature which can be easily compromised by slightly modifying the high number of pixels.

As we mentioned before, we assumed the existence of an optimal detector which can identify any adversarial attacks, and we implemented our experiments in light of this assumption. Unfortunately, practical detectors may result in false positive and false negative samples. False negative samples will not be processed by our method but false positive samples will, yielding bad classification and reducing the success rate of our method. In practice, the success rate of our method will drop slightly due to false positive detected samples, but we hope that with the rapid development of this field, a near optimal detector will be available soon.

5. Conclusions

We presented a novel problem, class retrieval and the recovery from adversarial attacks along with a proposed solution, which can be used as a baseline approach in further experiments. Our retriever is a self-evident addition to adversarial attack detectors and the combination of these two methods can enable the practical applicability of deep network even in case of attacks. We investigated four different adversarial attacks (PGD, MPGD, Deepfool, TPGD and PGDDLR) on three different datasets (MNIST, CIFAR10 and ImageNet). The results are promising and consistent across all attacks and datasets, where the average accuracy is 72%, 65% and 65%, respectively. Our retrieval algorithm was not able to recover the original class in all cases but, as a preliminary concept, it clearly shows that it is possible to build an algorithm where the original class can be retrieved. We hope this can open the way for further development and fine tuning of class retrievals of adversarial attacks, which can increase the robustness of deep neural networks in real-world applications.

Author Contributions: Conceptualization, J.A.-a. and H.A.; methodology, J.A.-a. and H.A.; software, J.A.-a.; validation, J.A.-a.; formal analysis, J.A.-a. and H.A.; investigation, J.A.-a. and H.A.; resources, J.A.-a. and H.A.; data curation, J.A.-a.; writing—original draft preparation, J.A.-a.; writing—review and editing, J.A.-a. and H.A.; visualization, J.A.-a.; supervision, H.A.; project administration, H.A.; funding acquisition, H.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: https://github.com/Al-Afandi/class_retrieval (accessed on 7 June 2021).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, Munich, Germany, 5–9 October 2015; Springer: Berlin/Heisenberg, Germany, 2015; pp. 234–241.
2. Ros, G.; Sellart, L.; Materzynska, J.; Vazquez, D.; Lopez, A.M. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 3234–3243.
3. Ni, J.; Chen, Y.; Zhu, J.; Ali, D.; Weidong, C. A Survey on Theories and Applications for Self-Driving Cars Based on Deep Learning Methods. *Appl. Sci.* **2020**, *10*, 2749. [CrossRef]
4. Papernot, N.; McDaniel, P.D.; Sinha, A.; Wellman, M.P. Towards the Science of Security and Privacy in Machine Learning. *arXiv* **2016**, arXiv:1611.03814.
5. Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; Fergus, R. Intriguing properties of neural networks. *arXiv* **2013**, arXiv:1312.6199.
6. Gilmer, J.; Metz, L.; Faghri, F.; Schoenholz, S.S.; Raghu, M.; Wattenberg, M.; Goodfellow, I. Adversarial Spheres. *arXiv* **2018**, arXiv:1801.02774.
7. Goodfellow, I.J.; Shlens, J.; Szegedy, C. Explaining and Harnessing Adversarial Examples. *arXiv* **2014**, arXiv:1412.6572.
8. Moosavi-Dezfooli, S.; Fawzi, A.; Fawzi, O.; Frossard, P. Universal adversarial perturbations. *arXiv* **2016**, arXiv:1610.08401.
9. Athalye, A.; Engstrom, L.; Ilyas, A.; Kwok, K. Synthesizing Robust Adversarial Examples. *arXiv* **2017**, arXiv:1707.07397.
10. Sankaranarayanan, S.; Jain, A.; Chellappa, R.; Lim, S.N. Regularizing deep networks using efficient layerwise adversarial training. *arXiv* **2018**, arXiv:1705.07819.
11. Ross, A.S.; Doshi-Velez, F. Improving the Adversarial Robustness and Interpretability of Deep Neural Networks by Regularizing their Input Gradients. *arXiv* **2017**, arXiv:1711.09404.
12. Lu, J.; Issaranon, T.; Forsyth, D. SafetyNet: Detecting and Rejecting Adversarial Examples Robustly. *arXiv* **2017**, arXiv:1704.00103.
13. Akhtar, N.; Mian, A. Threat of Adversarial Attacks on Deep Learning in Computer Vision: A Survey. *arXiv* **2018**, arXiv:1801.00553.
14. Rozsa, A.; Rudd, E.M.; Boult, T.E. Adversarial diversity and hard positive generation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Las Vegas, NV, USA, 27–30 June 2016; pp. 25–32.
15. Li, B.; Wang, Y.; Singh, A.; Vorobeychik, Y. Data Poisoning Attacks on Factorization-Based Collaborative Filtering. *arXiv* **2016**, arXiv:1608.08182.
16. Fredrikson, M.; Jha, S.; Ristenpart, T. Model inversion attacks that exploit confidence information and basic countermeasures. In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, 12–16 October 2015; pp. 1322–1333.
17. Dong, Y.; Liao, F.; Pang, T.; Su, H.; Zhu, J.; Hu, X.; Li, J. Boosting adversarial attacks with momentum. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 9185–9193.
18. Moosavi-Dezfooli, S.M.; Fawzi, A.; Frossard, P. DeepFool: A simple and accurate method to fool deep neural networks. *arXiv* **2016**, arXiv:1511.04599.
19. Croce, F.; Hein, M. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. *arXiv* **2020**, arXiv:2003.01690.
20. Zhang, H.; Yu, Y.; Jiao, J.; Xing, E.P.; Ghaoui, L.E.; Jordan, M.I. Theoretically Principled Trade-off between Robustness and Accuracy. *arXiv* **2019**, arXiv:1901.08573.
21. Chakraborty, A.; Alam, M.; Dey, V.; Chattopadhyay, A.; Mukhopadhyay, D. Adversarial Attacks and Defences: A Survey. *arXiv* **2018**, arXiv:1810.00069.
22. Dziugaite, G.K.; Ghahramani, Z.; Roy, D.M. A study of the effect of JPG compression on adversarial images. *arXiv* **2016**, arXiv:1608.00853.
23. Luo, Y.; Boix, X.; Roig, G.; Poggio, T.; Zhao, Q. Foveation-based Mechanisms Alleviate Adversarial Examples. *arXiv* **2016**, arXiv:1511.06292.
24. Gao, J.; Wang, B.; Lin, Z.; Xu, W.; Qi, Y. DeepCloak: Masking Deep Neural Network Models for Robustness Against Adversarial Samples. *arXiv* **2017**, arXiv:1702.06763.
25. Li, X.; Li, F. Adversarial Examples Detection in Deep Networks with Convolutional Filter Statistics. *arXiv* **2017**, arXiv:1612.07767.
26. Meng, D.; Chen, H. MagNet: A Two-Pronged Defense against Adversarial Examples. *arXiv* **2017**, arXiv:1705.09064.
27. Liang, B.; Li, H.; Su, M.; Li, X.; Shi, W.; Wang, X. Detecting Adversarial Image Examples in Deep Neural Networks with Adaptive Noise Reduction. *IEEE Trans. Dependable Secur. Comput.* **2019**, *18*, 72–85. [CrossRef]
28. Carlini, N.; Wagner, D. Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods. *arXiv* **2017**, arXiv:1705.07263.
29. Rozsa, A.; Gunther, M.; Boult, T.E. Towards Robust Deep Neural Networks with BANG. *arXiv* **2018**, arXiv:1612.00138.

30. Moosavi-Dezfooli, S.M.; Fawzi, A.; Fawzi, O.; Frossard, P.; Soatto, S. Analysis of universal adversarial perturbations. *arXiv* **2017**, arXiv:1705.09554.
31. Tramer, F.; Kurakin, A.; Papernot, N.; Goodfellow, I.; Boneh, D.; McDaniel, P. Ensemble Adversarial Training: Attacks and Defenses. *arXiv* **2020**, arXiv:1705.07204.
32. Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; Vladu, A. Towards Deep Learning Models Resistant to Adversarial Attacks. *arXiv* **2017**, arXiv:1706.06083.
33. Kim, H. Torchattacks: A Pytorch Repository for Adversarial Attacks. *arXiv* **2020**, arXiv:2010.01950.