AdversarialQR: An adversarial patch in QR code format

Aran Chindaudom
Faculty of Information and Communication Technology
Mahidol University
Nakhonpathom, 73170, Thailand
aran.chi@student.mahidol.ac.th

Prarinya Siritanawan
School of Information Science
Japan Advanced Institute of Science and Technology
Ishikawa, 923-1292 , Japan
prarinya@jaist.ac.jp

Karin Sumongkayothin
Faculty of Information and Communication Technology
Mahidol University
Nakhonpathom, 73170, Thailand
karin.sum@mahidol.ac.th

Kazunori Kotani
School of Information Science
Japan Advanced Institute of Science and Technology
Ishikawa, 923-1292, Japan
ikko@jaist.ac.jp

***Abstract*—In this paper, we present a method to camouflage an attack on image recognition system by using an adversarial patch embedded on a scan-ready QR code. Adversarial patch refers to a class of a real-world attack on a machine learning system that adds a 'patch' onto the image. However, unlike existing methods, they are highly conspicuous to human perception. As these attacks are performed in the real world, they require users to manipulate the scene. However, not only the patch catches the attention of the classification system but also bystanders' attention as well. We believe that forcing the adversarial patch into the form of a scan-ready QR code can conceal its primary reason to exist in the scene. The main challenge of the research is the process of forcing an adversarial patch into a scan-ready QR code while trying to retain as much information for the patch to work as a real-world adversarial example. The experiments had been done to investigate trade-off compared to training the patch in different shapes.**

***Index Terms*—Adversarial Attack, Deep Learning, Transformed Patch, Convolutional Neural Networks**

I. Introduction

With rapid and successful developments in a variety of academics and commercial applications, Deep Learning (DL) has started surpassing human performance on image recognition tasks [1] and gained a surge in popularity in many fields of applications involving artificial intelligence. Despite the remarkable successes, previous works have demonstrated that deep learning systems are so brittle that even imperceptibly modified inputs, known as adversarial examples, are able to completely fool the image classifier models [2][3]. Adversarial examples pose a serious security concern for machine learning applications, especially in the high-stakes image classification tasks like autonomous driving or facial recognition systems, since these attacks have been proven to work in the physical world [4].

978-1-7281-9331-1/20/$31.00 ©2020 IEEE

This paper concerned the conspicuousness of adversarial patches against object classification systems based on Convolutional Neural Networks (CNN). Adversarial patch refers to a class of attacks on a machine learning system that adds a 'patch' onto the image, which resulted in the system mislabeling the image, turning the scene into a form of an adversarial example. Previous works showed the efficacy of the attacks in both the context of object classification [5] and object detection [6]. However, unlike other existing methods, they are not imperceptible to human intuition and require users to manipulate objects being attacked themselves by placing a patch near the object. We propose a method to convert these patches into a form of usable QR code-like pattern in order to misdirect suspicion of its primary function on fooling an image classification system from human intuition.

II. Related Works

Deep learning is a subset of machine learning approaches which allows intelligent systems to extract useful patterns and their high level semantics from large amounts of data. Previously, there had been difficulties for traditional machine learning methods in extracting features due to limitations to deal with high dimensional input data [7] and computational bottlenecks [8]. Deep learning, especially Convolutional Neural Networks (CNNs), addressed these limitations by representing a complex association of multiple simple visual features through artificial neurons. By describing the relationship of edges and certain geometric structures through a stack of hidden layers, a deep-learning based classifier is able to represent the whole object. In the training process, a deep learning network becomes more powerful when applying a large amount of training data.

Typically, a neural network is formed by a complex connection of neural network layers. A layer consists of several artificial neurons, in which each neuron uses an activation function to map any high dimensional input to the output

value(s). A general representation of a neural network can be described as follows:

$$f(x) = f^{(k)}(f^{(2)}(f^{(1)}(x))) \tag{1}$$

where x is an input image, $f^{(i)}$ is a function of the i^{th} network layer where *i = 1,2,.... k.*

Several deep learning architectures are widely used in computer vision tasks include LeNet[9], AlexNet[10], VGG[11], Inception[12][13][14], and ResNet[1]. Typically, attackers usually generate adversarial examples targeting these architectures. Much research on adversarial examples was performed using well-known datasets in computer vision such as MNIST, CIFAR-10, or ImageNet. The MNIST is a dataset of handwritten digits [15]. CIFAR-10 and ImageNet are for image classification tasks. CIFAR consists of 60,000 32x32 colour images with ten classes [16]. The ImageNet consists of 14,196,122 images with 1,000 classes [17]. Most adversarial approaches are experimented on small parts of ImageNet due to the massive amount of images in the dataset.

Adversarial examples in machine learning have already been discussed more than a decade. Machine learning-based systems that are primarily targeted often have handcrafted features such as spam filters and intrusion detection. Dalvi et al. formulated adversarial examples as a game between adversary and classifier, where both exploit cost-sensitive learning [18]. Szegedy et al. [3] later introduced L-BFGS method to generate adversarial examples. Since the attack employed a linear search to explore the optimal solution, the method was impractical and computationally expensive. Goodfellow et al. [2] proposed a faster solution by using the Fast Gradient Sign Method (FGSM) to generate adversarial example by adding a gradient sign of cost function on each pixel. The perturbations can be expressed as:

$$\eta = \epsilon sign\left(\nabla_x J\left(\theta, x, y\right)\right) \tag{2}$$

where η is the magnitude of the perturbation computed by using gradients from back-propagation. Given an original image x, the generated adversarial example x' can be calculated by adding perturbation η on the original image $x' = x + \eta$.

Sharif et al. [19] proposed a practical attack on the facial bio-metric systems in the form of adversarial printed eyeglasses. When an attacker with the printed eye glasses subjects his or her face to a face-recognition system, this will allow the attacker to avoid being recognized or to disguise as another identity. However, since the attack is in the form of printed eyeglasses, their uses and inconspicuousness are only limited to facial recognition scenarios.

Eykholt et al. [6] also proposed a solution that addresses the conspicuousness of real-world adversarial attacks by designing the perturbations to mimic graffiti on road signs. Unlike [19] where successful physical attacks were done in stable environments with little variation in lighting scale and camera angles, the road signs exist in an environment where angles, distances, and lighting conditions are widely varied. Thus, the authors employed a combination of physical and synthetic transformations in the creation of the attack, which involved manually masking images of the target physical object in different angles, distances, lighting conditions. This resulted in physical adversarial examples robust to varying orientations and distances. Since the attack was generated in a series of individual pieces of graffiti, the applicability in the physical world is fairly limited.

In order to create adversarial examples that can be applied universally in the real-world scenario, Adversarial patch attacks were first introduced by [5] for image classifiers. In order to create a universal, robust adversarial example that can be physically attached to a scene by using any printing mediums, and misdirect the classifier to output any targeted class. Based on the principle that image classifier tasks are trained to exploit the most salient patterns in an image. The adversarial patch exploits this feature by embedding input images with higher salient patterns than objects in the real world. The training process is done by applying the patch at a random position and scale to the scene. As a result, the object will be detected as a target class of the trained patch.

In this paper, we are interested in conspicuousness to human intuition on the employment of the adversarial patches.We believe that by making the adversarial patch readable by QR scanner would alleviate the suspicions of its primary function as an adversarial tool. However, forcing the patch into such specific format would more or less hinder its usability and mobility compared to the original counterpart. Therefore, the aspects of adversarial efficacy comparing with other patch shapes, ease of scan and employment feasibility are also investigated in the experiment.

The contributions of this work are:

- Extension of an adversarial patch concept into the form of scan-ready QR code, to our knowledge there had never been works that explored this approach.
- Improve the mobility and usability of the physical adversarial examples by hiding perturbations underneath a QR code pattern.
- On the top of adversarial capability, this work also enables the adversarial examples to carry additional information for further usages in other applications.
- Propose a procedure to retain the QR code functionality after the training of adversarial examples.

III. Methodology

The overall procedure of adversarial patch creation closely followed [5], with several changes to accommodate the shape of QR code pattern.

A. Masked Patch Creation

The patch application operator $A(p, q, x, r, l)$ is formed by translating patch p onto the location l with r rotation on the image x. In order to perform the patch application, a mask image, which is a matrix filled with only zeroes, will firstly be created with the size of the original image x. A patch p is initialized by a QR pattern generated from an input string q, which can be plain text or URL. The black and white areas of the QR pattern represented by 0 and 1 respectively.The patch

is randomly rotated by r ranging from 0, 90, 180, 270, and 360 degrees, and then will be placed onto the mask image at the random location l. The masked patch is later applied on the training images. For the initial patch weights, the black areas of the QR code will be set to 0 since they would not be trained. Fig. 1 shows the process to create the masked patch where the patch p was created. The black, white and grey colors represents zeros, random values, unassigned values respectively.

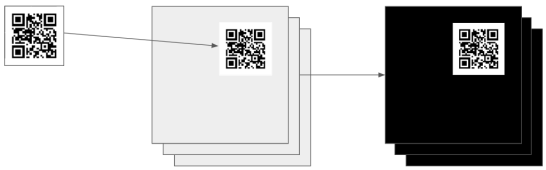


Fig. 1: The QR patch masking with specified rotation and translation

B. Patch Training

With the masked patch ready for creating the adversarial examples, the subsequent process is the application of the masked patch onto the training image. The patch application process starts by taking the transformed patch mask from the previous process (which was randomly rotated and placed on a mask image) and apply it to a training image.

The adversarial QR patch p' can be computed by maximizing the expectation of the following function extended from [20][5]:

$$p' = \underset{p}{\operatorname{argmax}} E_{x\sim X, r\sim R, l\sim L}[\log Pr(\hat{y}|A(p,q,x,r,l))] \quad (3)$$

where $\hat{y}$ is the confidence of the target class and the patch operator A is applied over a distribution of X images in the training set. In order to make the trained patch universal, the rotation r and l are varied over the distribution L and R when the patch is applied on a training image. Fig. 2 displays the procedure of the QR adversarial example training according to Eq. (3).

First, the forward and backward propagation is applied on the adversarial example x_{adv}, the gradient of the loss toward target class obtained from backward propagation $\nabla_x J$ is used to update the patch in $Update(p, \nabla_x J)$. The update function can be written as:

$$p'_{new} = p' - \nabla_x J \quad (4)$$

where p' is the current patch, and p'_{new} is the updated patch.

Then, the updated patch p'_{new} will be re-applied on the training image using the same operator A to re-create adversarial example x_{adv}. The adversarial example x_{adv} is then reevaluated by the image classifier network to obtain the prediction confidence of the target class $\hat{y}$ which will be compared to the target value y' to determine whether to continue training the patch or not.

Finally, if the prediction confidence exceeds the target value or if the training process reaches the maximum amount of iterations specified, the training process will stop and return the final adversarial example x_{adv} along with the adversarial QR patch p'.

Note that after the patch has been trained, the intensity of the black (R:0, G:0, B:0) parts of the patch will be increased to create a transparency zone surrounding the QR symbol and make the patch readable to a QR code scanner. The process is represented by the following function:

$$p'_{out}(u,v) = \begin{cases} p'(u,v) + \tau, & \text{if } p'(u,v) = [0,0,0] \\ p'(u,v), & \text{otherwise} \end{cases} \quad (5)$$

where u and v are the coordinates of each pixels on the patch p' and τ is a small number. Fig. 3 displays the steps to make the trained QR patch scan-ready, where the left image represents the QR symbol patched onto an image, the middle image represents the black parts of the QR symbol from the left image that needs to be modified, and the right image represents the area untouched by the process.

IV. Experimental Results

A. Initialization

The initialization process begins with the setup of the deep learning model to be used as a target of the adversarial examples. The system will retrieve the pre-trained image classification model to instantiate the model's architecture. The default target image classification model is InceptionV3 [13] a convolutional neural network model that was trained using the images from the ImageNet ILSVRC2012 image database [17]. The target class and its minimum prediction value are also specified here as a threshold to stop the attack and move on to next the image. The threshold is set to 0.9 or 90 percent of the prediction confidence.

The second component of the system to be initialized is the image dataset. The system samples 50000 of the validation sets from the ImageNet ILSVRC2012 dataset and split them into training sets and the test sets then modifies them to optimize the efficiency of the patch training process. Each image in the dataset will be resized to 299*299 to prevent errors during the patch training and testing processes. The images will then be normalized with the mean and standard deviation to decrease the training time. Finally, the label of each image is compared to the actual result from the pre-trained classifier to filter out images that are already incorrectly identified by the pre-trained model.

The third and final part of the initialization process is the initialization of the adversarial patch and its properties. The size of the patch in form of the ratio to the image at the range between 0 and 1 is retrieved from the parser, with the default ratio being 0.05 (equivalent to 5 percent of the training/testing image's size). The type of the adversarial patch must also be specified, which can be either a circle, a square, or a QR code as the shape of the masked patch to be attached to the images in the dataset. Other parameters that is specified for

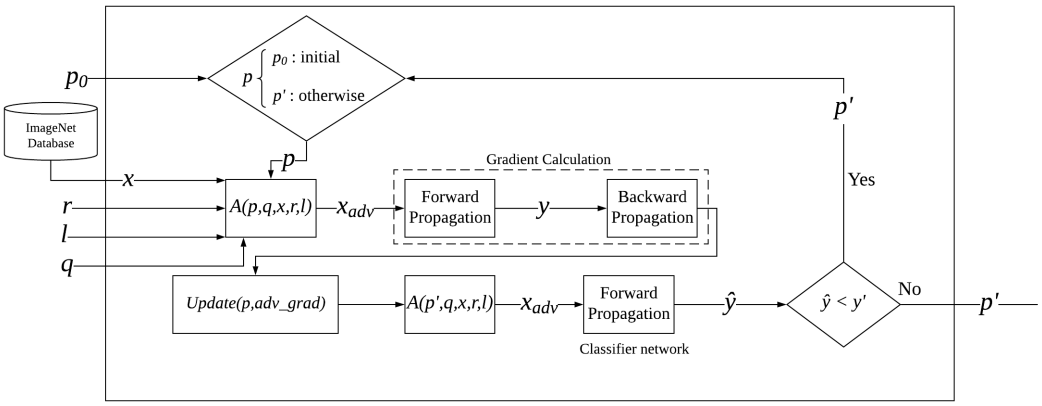


Fig. 2: The architecture of the QR Adversarial Attack system for one training sample.

Fig. 3: The process to make the trained QR patch visible to a QR scanner.

Fig. 4: QR adversarial patch

Fig. 5: Square adversarial patch

Fig. 6: Circular adversarial patch

this system includes the amount of subprocesses used during the image dataset loading process (2 subprocesses by default to prevent unnecessary overhead), the number of epochs to train and test the adversarial patch (20 epochs by default), and the option to enable CUDA parallel computing platform to utilize GPUs for computation, reducing the system's processing time required, and the maximum number of iterations to find the adversarial example during the training process iterations.

B. Discussion

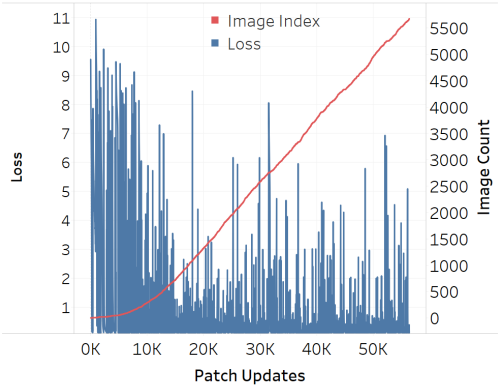
In order to test adversarial efficacy of QR patch compared to square and circle shape patches shown in Fig. 4, 5, and 6, the test was carried by recording Loss values over patch updates calculated from randomly selected 6000 training images. An early stop would be performed either after 500 iterations or when the target class confidence reaches 0.9, the weights on the patch would then be carried over the next image to create the next attack loop. The results on Fig. 7 showed the loss values over patch updates, the rightmost axis indicates the index of the training image processed per each update step. The QR-shaped, square-shaped patch, and the Circle-shaped patch exhibited similar trends earlier on. However, the circle patch is proven to be more resistant to the changes of the background scene, where the loss values do not spike as much compared to its QR and Square counterparts. We conjecture that both square shape and QR shape contain a lot of edges and corners, which manifests salient features oppressing other features. As these salient features disrupt the training every time a new image is fed to the system, this causes the spike of loss in Fig. 7. To suppress the effects of edges and corners, training the adversarial examples will require more steps to adapt the patch towards the target class.

In order to evaluate the convergence speed of the patch update of each shapes, we measured the average loss over 1000 update steps of different patch shapes (Circle, QR, Square). Training images are randomly sampled from the ImageNet dataset using an identical seed to ensure that all of shapes were fed with same set of images. Due to limitations on computational power, we performed an early stop at 1000 iterations for each scene. The resulting loss curve for each shape is depicted in Fig. 8, the loss of circular patch descends fastest, followed by QR and Square which had negligible differences.

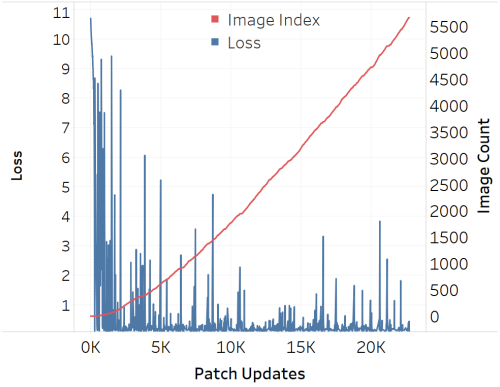
Figure 9 shows the evaluation results of the patch embed on the images from the test sets before and after modification process to retain the QR code functionality after the training of adversarial examples using $\tau = 22$. Although some evaluation results showed negligible reduction in adversarial efficacy, the results greatly varied on different images.

V. Conclusion

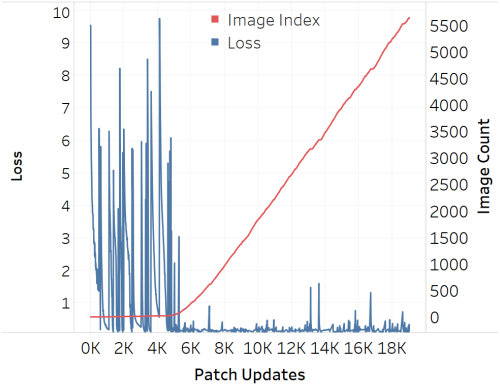
In conclusion, it is feasible to create an Adversarial Patch in the form of scan-ready QR-code while retaining enough information to act as a real-world adversarial example against traditional deep learning-based image classification systems. Although the patch is scannable by a QR reader, there are a lot of constraints, such as the absence of shadows on the QR code, which is only feasible when the patch is projected and



(a) Training loss for QR shaped patch



(b) Training loss for square shaped patch



(c) Training loss for circular shaped patch

Fig. 7: Training loss over update iteration of various patch types, (a) QR shape patch, (b) Square shape patch, and (c) Circular shape patch.

enlarged on an LCD screen, and the readable angles and distances are very limited. The experimental results showed that compared to other shapes such as circle and square, forcing the adversarial patch to accompany a QR pattern will result in decrease of its robustness in terms of adversarial efficacy as a trade-off. Furthermore, a human subjective evaluation should be investigated in the future. Moreover, other forms of adversarial patches should as well be considered and attempted

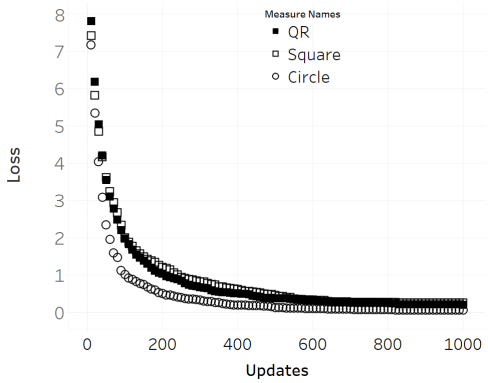


Fig. 8: Loss per gradient 1000 update steps from initial state

since it is possible to constrict an adversarial patch into very strict pattern and rules such as QR code.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1026–1034.

[2] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and Harnessing Adversarial Examples," in *International Conference on Learning Representations (ICLR)*, 2015.

[3] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *2nd International Conference on Learning Representations (ICLR)*, 2014.

[4] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world.," in *International Conference on Learning Representations (ICLR)*, 2017.

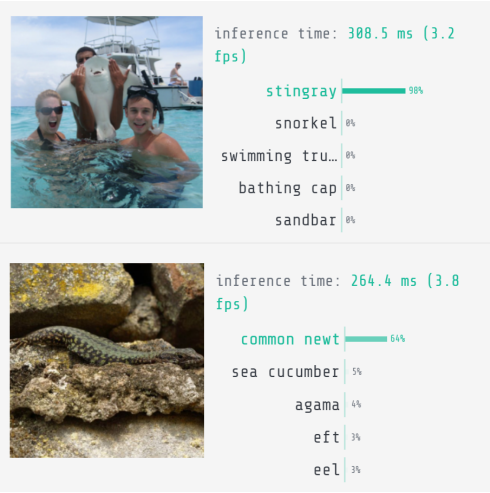
[5] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer, "Adversarial Patch," in *31st Conference on Neural Information Processing Systems (NIPS)*, 2017.

[6] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song, "Robust Physical-World Attacks on Deep Learning Models," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[7] Y. Bengio and Y. Lecun, "Scaling learning algorithms toward ai," in *Large-Scale Kernel Machines*, L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, Eds. MIT Press, 2007, pp. 321–359.

[8] D. Storcheus, A. Rostamizadeh, and S. Kumar, "A survey of modern questions and challenges in feature extraction," in *International Workshop on Feature Extraction: Modern Questions and Challenges at NIPS 2015*, vol. 44, PMLR, 2015, pp. 1–18.

[9] Q. V. Le, M. Ranzato, R. Monga, M. Devin, K. Chen, G. S. Corrado, J. Dean, and A. Y. Ng, "Building high-level features using large scale unsupervised learning," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013.



(a) Original images.

(b) Images with QR adversarial patch

(c) Images with QR adversarial patch after color modifications to make the patch scan-ready

Fig. 9: Inceptionv3 evaluations before and after application of the adversarial QR patch

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *25th International Conference on Neural Information Processing Systems (NIPS)*, vol. 1, 2012, pp. 1097–1105.

[11] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *3rd International Conference on Learning Representations, (ICLR)*, 2015.

[12] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going Deeper with Convolutions," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[13] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[14] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning," in *31st AAAI Conference on Artificial Intelligence*, 2017.

[15] *MNIST handwritten digit database, Yann LeCun, Corinna Cortes and Chris Burges*. [Online]. Available: http://yann.lecun.com/exdb/mnist/.

[16] A. Krizhevsky, "Learning Multiple Layers of Features from Tiny Images," University of Toronto, Toronto, Tech. Rep. TR-2009, 2009.

[17] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, Dec. 2015, ISSN: 0920-5691. DOI: 10.1007/s11263-015-0816-y.

[18] N. Dalvi, P. Domingos, Mausam, S. Sanghai, and D. Verma, "Adversarial classification," in *10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2004, pp. 99–108, ISBN: 1581138881. DOI: 10.1145/1014052.1014066.

[19] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, "Accessorize to a Crime: Real and Stealthy Attacks on State-of-the-Art Face Recognition," in *ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16, New York, NY, USA: ACM, 2016, pp. 1528–1540, ISBN: 978-1-4503-4139-4. DOI: 10.1145/2976749.2978392.

[20] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble Adversarial Training: Attacks and Defenses," in *6th International Conference on Learning Representations, (ICLR)*, 2018.