SeVuc: A study on the Security Vulnerabilities of Capsule Networks against adversarial attacks

Microprocessors and Microsystems 96 (2023) 104738

Contents lists available at ScienceDirect

Microprocessors and Microsystems

journal homepage: www.elsevier.com/locate/micpro

SeVuc: A study on the Security Vulnerabilities of Capsule Networks against adversarial attacks

Alberto Marchisio [a,*], Giorgio Nanfa [b], Faiq Khalid [a], Muhammad Abdullah Hanif [c], Maurizio Martina [b], Muhammad Shafique [c]

[a] Technische Universität Wien (TU Wien), Vienna, Austria
[b] Politecnico di Torino, Turin, Italy
[c] eBrain Lab, Division of Engineering, New York University Abu Dhabi, United Arab Emirates

ARTICLE INFO

Keywords:
Machine learning
Artificial intelligence
Deep learning
Deep neural networks
Convolutional neural networks
Capsule Networks
Architecture
Adversarial attacks
Affine transformations
Security
Robustness
Vulnerability

ABSTRACT

Capsule Networks (CapsNets) preserve the hierarchical spatial relationships between objects, and thereby bear the potential to surpass the performance of traditional Convolutional Neural Networks (CNNs) in performing tasks like image classification. This makes CapsNets suitable for the smart cyber–physical systems (CPS), where a large amount of training data may not be available. A large body of work has explored adversarial examples for CNNs, but their effectiveness on CapsNets has not yet been studied systematically. In our work, we perform an analysis to study the vulnerabilities in CapsNets to adversarial attacks. These perturbations, added to the test inputs, are small and imperceptible to humans, but can fool the network to mispredict. We propose a greedy algorithm to automatically generate imperceptible adversarial examples in a black-box attack scenario. We show that this kind of attacks, when applied to the German Traffic Sign Recognition Benchmark and CIFAR10 datasets, mislead CapsNets in making a correct classification, which can be catastrophic for smart CPS, like autonomous vehicles. Moreover, we apply the same kind of adversarial attacks to a 5-layer CNN (LeNet), to a 9-layer CNN (VGGNet), and to a 20-layer CNN (ResNet), and analyze the outcome, compared to the CapsNets, to study their different behaviors under the adversarial attacks.

1. Introduction

Convolutional Neural Networks (CNNs) have shown great improvements and successes in many Machine Learning (ML) applications, e.g., object detection, face recognition, image classification [1], which form a key component of today's smart Cyber–Physical Systems (CPS) and Internet of Things (IoT) systems [2]. However, the convolutional layers do not preserve spatial hierarchies between the objects, e.g., orientation, position and scaling. CNNs are specialized to identify and recognize the presence of an object as a feature, without perceiving the spatial relationships across multiple features. Concurrently, the Google Brain's team [3] proposed the CapsNet, an advanced CNN architecture composed of so-called *capsules*, which is based on the *dynamic routing*, an iterative algorithm to learn the coupling between capsules. The key idea behind the CapsNets is called *inverse graphics*, i.e., when the eyes analyze an object, the spatial relationships between its parts are decoded and matched with the representation of the same object in our brain. Similarly, in CapsNets the feature representations are stored inside the capsules in a *vector* form, in contrast to the scalar form used by the neurons in traditional CNNs [4]. Despite the great success in the field of image classification, prior works [5] have demonstrated that, in a similar way as CNNs, the CapsNets are also not immune to adversarial attacks.

Adversarial examples are small perturbations added to the inputs, which are generated to fool the network, thereby revealing the corresponding security vulnerabilities [6]. Hence, they can be dangerous in safety–critical CPS and IoT systems, with applications like Voice Controllable Systems (VCS) [7,8] and image/face/traffic signs recognition [9,10]. Many works [9,11] have analyzed the impact of adversarial examples on CNNs, and studied different methodologies to improve the defense mechanisms. Towards CapsNets, in this paper, we aim to addressing the following **key research questions:**

1. Is a CapsNet vulnerable to adversarial examples? If yes, how, why, and to what extent?
2. How does the CapsNet's vulnerability to adversarial attacks differ from that of the traditional CNNs?

To the best of our knowledge, we are the first to study (1) the vulnerability of the CapsNet to such adversarial attacks for the German

* Corresponding author.
E-mail address: alberto.marchisio@tuwien.ac.at (A. Marchisio).

https://doi.org/10.1016/j.micpro.2022.104738
Received 25 February 2022; Received in revised form 4 November 2022; Accepted 28 November 2022
Available online 30 November 2022

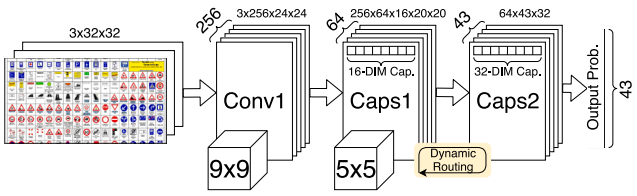

Fig. 1. Architecture of the CapsNet for the GTSRB dataset.

Traffic Sign Recognition Benchmark (GTSRB) [12], which is crucial for autonomous vehicle use cases; (2) to automatically generate an attack image for such CapsNet in a black-box scenario; and (3) to compare the robustness of the CapsNet with the CNNs with 5 and 9 layers, and the robustness of the DeepCaps with a 20-layer ResNet, by applying affine transformations and adversarial attacks.

Our Novel Contributions:

1. We **analyze the robustness** behavior of the CapsNet, the DeepCaps, a 5-layer CNN (LeNet), a 9-layer CNN (VGGNet), and a 20-layer CNN (ResNet), under **affine transformations** applied to the input images of the GTSRB and CIFAR10 datasets, and study their differences (*Section* 3).
2. We develop a **novel algorithm to automatically generate targeted imperceptible and robust adversarial examples** (*Section* 4).
3. We **compare the robustness** of the CapsNets with traditional CNNs, under the **adversarial examples** generated by our algorithm (*Section* 5).

In summary, our results show that the CapsNet has comparable robustness to a much deeper CNN like the VGGNet, while the LeNet is much more vulnerable to the affine transformations and the adversarial attacks, while the DeepCaps is more robust than the ResNet. Therefore, we can make a key step forward for the security of safety–critical applications by employing deep and complex networks such as the DeepCaps.

Before proceeding to the technical sections, in *Section* 2 we present an overview of the CapsNets and of the adversarial attacks, to a level of detail necessary to understand the contributions of this paper.

2. Background and related work

2.1. Capsule networks

Capsules were first introduced by Hinton et al. [13]. They are multi-dimensional entities that can learn the hierarchical information of the features. Compared to the traditional CNNs, a CapsNet has the *capsule* (i.e., a group of neurons) as the basic element, instead of the neuron. State-of-the-art works about CapsNet's architecture and training algorithms [3,14] have shown competitive accuracy results for image classification task, compared to other state-of-the-art classifiers. Kumar [15] proposed a CapsNet architecture, composed of 3 layers, which achieved good performance for the GTSRB dataset [12]. Fig. 1 shows its architecture. Note, between the two consecutive capsule layers (i.e., Caps1 and Caps2), the *dynamic routing* algorithm is performed, which is compute-intensive as it introduces a feedback loop during the training and inference.

Rajasegaran et al. [16] proposed a deep CapsNet architecture called DeepCaps, which achieved high accuracy for the CIFAR10 dataset [17]. Its architecture, shown in Fig. 2, presents a first convolutional layer, 15 2-dimensional (2D) convolutional capsule layer, one 3D convolutional capsule layer, and one class capsule layer. Some layers are connected to each other through a skip connection, while the dynamic routing is performed in the 3D convolutional capsule layer and in the class capsule layer.

2.2. Adversarial attacks

Szegedy et al. [18] studied that several ML models are vulnerable to adversarial examples. Goodfellow et al. [19] explained the problem observing that *ML models misclassify examples that are only slightly different from correctly classified examples drawn from the data distribution*. Considering an input example x, the adversarial example $x^* = x + \eta$ is equal to the original one, except for a small perturbation η. The goal of the perturbation is to maximize the prediction error, to make the predicted class $C(x)$ different from the target one $C(x^*)$. In recent years, many methodologies to generate adversarial examples and their respective defense strategies have been proposed [20–22]. Adversarial attacks can be categorized according to different attributes, e.g., the choice of the class, the kind of the perturbation and the knowledge of the network under attack [10,23]. We summarize these attributes in Fig. 3. A simple representation of the attack scenario that we consider throughout this paper is visible in Fig. 4.

An adversarial attack is very efficient if it is *imperceptible* and *robust*: this is the main concept of the analysis conducted by Luo et al. [24] for traditional CNNs. Their attack *modified the pixels in high variance areas*, since the human eyes do not perceive their modifications much. Moreover, an adversarial example is robust if the gap between the probabilities of the predicted and the target class is so large that, after an image transformation (e.g., compression or resizing), the misclassification still holds. Prior works showed that a CapsNet is vulnerable to adversarial attacks. Michels et al. [5] analyzed the CapsNet's accuracy applying Carlini–Wagner Attack [25], Boundary Attack [26], DeepFool Attack [27] and Universal Attack [28]. Frosst et al. [29] presented an efficient technique to detect the crafted images on MNIST, Fashion-MNIST [30] and SVHN [31] datasets. Qin et al. [32] investigated the detection adversarial examples on CapsNets with the reconstruction network, and proposed a successful deflection algorithm [33]. Gu et al. [34] proposed a novel CapsNet which further improves its robustness against affine transformations. All the aforementioned works did not analyze the effect of black-box attacks and affine transformations on CapsNets and CNNs, which is exactly what we focus on in this paper.

3. Robustness analysis under affine transformations

Before studying the vulnerability of the CapsNets under adversarial examples, we apply certain affine transformations to the test input images of the GTSRB and CIFAR10 datasets, and to observe their effects on our network predictions. We use three different types of transformations: rotation, shift and zoom. This analysis is important to understand how affine transformations, which are perceptible yet plausible in the real world, can or cannot mislead the investigated networks.

3.1. Experimental setup

For the analysis on the GTSRB dataset, we consider the CapsNet of Fig. 1, which is composed of a convolutional layer, with kernel 9×9, a convolutional capsule layer, with kernel 5×5, and a fully connected capsule layer. We implement it in TensorFlow, to perform classification on the GTSRB dataset [15] with an accuracy of 97.6%. This dataset has images of size 32×32 and it is divided into 34,799 training examples and 12,630 testing examples. Each pixel intensity assumes a value from 0 to 1. The number of classes is 43. For evaluation purposes, we compare the CapsNet with a 5-layer CNN (LeNet) [35], trained for 30 epochs, and a 9-layer CNN (VGGNet) [36], trained for 120 epochs. They are both implemented in TensorFlow and their accuracy with clean test images are 91.3% and 97.7%, respectively.

For the analysis on the CIFAR10 dataset, we consider the DeepCaps architecture of Fig. 2, which is composed of 18 layers. We implement it in TensorFlow, and its classification accuracy on the CIFAR10 dataset [17] is 91.52%. The CIFAR10 dataset contains 50,000 training images and 10,000 testing images of size 32×32, divided into 10 classes. We compare it with a 20-layer ResNet [37], which has an accuracy with clean test images of 91.48%.

Table 1
Robustness analysis after employing affine transformations to an image of the GTSRB dataset.

	CapsNet confidence		VGGNet confidence		LeNet confidence	
Original	Stop	0.057	Stop	1.000	Stop	0.999
Zoom 1.5x	Road work	0.026	General caution	0.735	Yield	0.574
Zoom 0.8x	Stop	0.054	Stop	0.999	Stop	0.989
Shift (2,2)	Stop	0.050	Stop	0.999	Stop	0.984
Shift (4,4)	Yield	0.037	Yield	0.900	Yield	0.999
Rotate 10°	Stop	0.056	Stop	0.999	Stop	0.999
Rotate 30°	Stop	0.032	Stop	0.747	Go straight or left	0.668

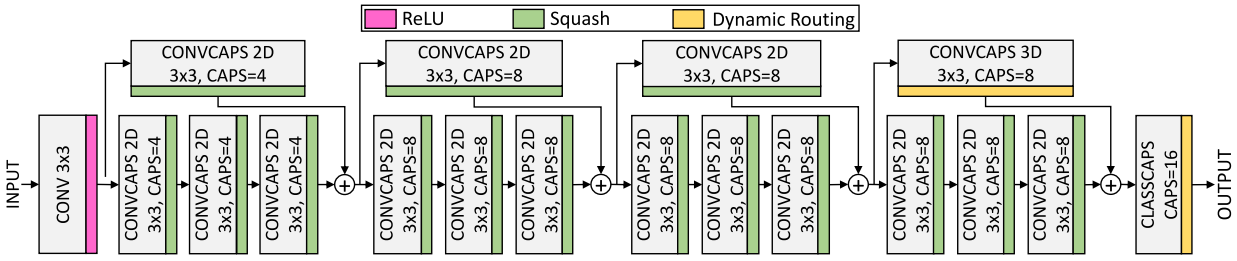

Fig. 2. Architecture of the DeepCaps for the CIFAR10 dataset.

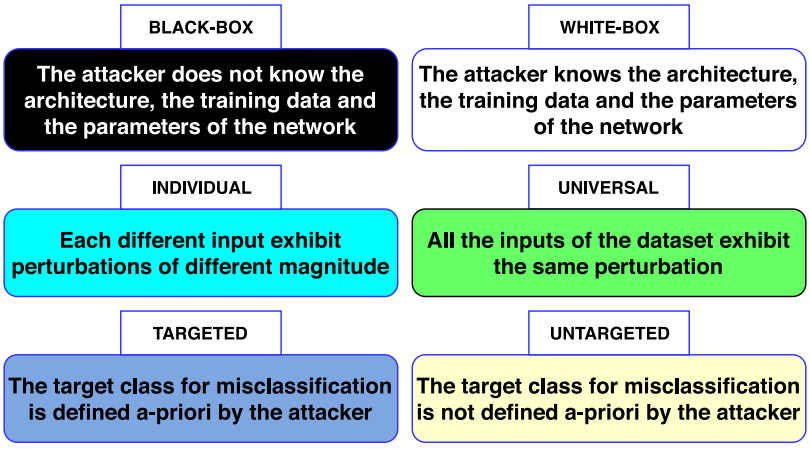

Fig. 3. Taxonomy of adversarial examples.

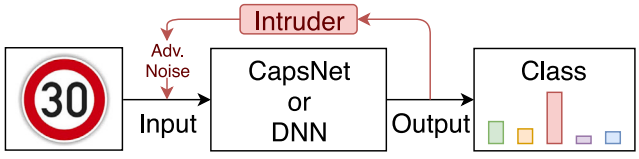

Fig. 4. A black-box adversarial attack, showing the classification for an image perturbed by an adversarial intruder.

3.2. Robustness under affine transformations on the GTSRB dataset

Table 1 reports some results for affine transformations applied to the input. The analysis shows that both the CapsNet and the VGGNet can be fooled by some affine transformations, like zoom or shift. While the confidence of the CapsNet is lower.[1] Moreover, the LeNet, since it has lower number of layers, compared to the VGGNet, is more vulnerable to this kind of transformations. Since the CapsNet has similar depth and number of parameters as the LeNet (i.e., much less compared to the VGGNet), an expected outcome at a first sight would have been a reduced robustness compared to the VGGNet. However, the CapsNet can overcome such deficit with capsules and advanced algorithms, such as the dynamic routing. Indeed, as we can notice in the example of the image rotated by 30°, the confidence is lower, but both the CapsNet and the VGGNet are able to classify correctly, while the LeNet is fooled.

3.3. Robustness under affine transformations on the CIFAR10 dataset

Table 2 shows the results of an image of the CIFAR10 dataset that is labeled as "truck". The DeepCaps and the ResNet20 correctly classify the image in its original form, while a certain set of affine transformations fools both networks. Note that, the ResNet20 is fooled by the Zoom 1.5× transformation, while the DeepCaps correctly predicts, despite the lower confidence than for the original image. Hence, the DeepCaps exhibits higher robustness than the ResNet20 to this kind of transformations.

[1] We noticed that the prediction confidence of the CapsNets are very low, with a small difference between classes, compared to the CNNs. As reported in the work of Guo et al. [38], in a deep learning model, based on its calibration, the probability associated with the predicted class label may not reflect its ground truth correctness likelihood. Therefore, such a behavior can be attributed to the fact that CNNs and CapsNets are calibrated in a different way.

Table 2
Robustness analysis after employing affine transformations to an image of the CIFAR10 dataset.

	DeepCaps confidence		ResNet20 confidence	
Original	Truck	0.619	Truck	0.844
Zoom 1.5x	Truck	0.528	Automobile	0.515
Zoom 0.8x	Truck	0.626	Truck	0.739
Shift (2,2)	Truck	0.441	Truck	0.620
Shift (4,4)	Ship	0.385	Ship	0.464
Rotate 10°	Truck	0.716	Truck	0.672
Rotate 30°	Ship	0.397	Ship	0.530

4. Generation of targeted imperceptible and robust adversarial examples

An efficient adversarial attack can generate *imperceptible* and *robust* examples to fool the network. Before describing the details of our algorithm, we discuss the importance of these two aspects.

4.1. Imperceptibility and robustness

An adversarial example is typically considered *imperceptible* if the modifications of the original sample are so small that humans cannot notice them, or they are hard to be noticed. To create an imperceptible adversarial example, we need to add the perturbations in the pixels of the image with the highest standard deviation. In fact, the perturbations added in high variance zones are less evident and more difficult to detect with respect to the ones applied in low variance pixels. Considering an area of $M \cdot N$ pixels x, the standard deviation (SD, Eq. (1)) of the pixel $x_{i,j}$ can be computed as the square root of the variance, where μ is the average of the $M \cdot N$ pixels:

$$SD(x_{i,j}) = \sqrt{\frac{\sum_{k=1}^{M}\sum_{l=1}^{N}(x_{k,l}-\mu)^2 - (x_{i,j}-\mu)^2}{M \cdot N}} \tag{1}$$

Hence, when the pixel is in a high variance region, its standard deviation is high and the probability to detect a modification of the pixel is low. To measure the imperceptibility, it is possible to define the distance (D, Eq. (2)) between the original sample X and the adversarial sample X*, where $\delta_{i,j}$ is the perturbation added to the pixel $x_{i,j}$:

$$D(X^*, X) = \sum_{i=1}^{M}\sum_{j=1}^{N}\frac{\delta_{i,j}}{SD(x_{i,j})} \tag{2}$$

This value indicates the total perturbation added to all the pixels under consideration. We define D_{MAX} as the maximum total perturbation tolerated by the human eye. The value of D_{MAX} can vary among different datasets or images, because it depends on the resolution and the contrast between neighboring pixels.

An adversarial example is typically called *robust* if the gap function, i.e., the difference between the probability of the target class and the maximum class probability is maximized:

$$GAP = P(target\,class) - max\{P(other\,classes)\} \tag{3}$$

If the gap function increases, the adversarial example becomes more robust, because the changes of the probabilities caused by some image transformations (e.g., compression or resizing) tend to be less effective. Indeed, if the gap function is high, a variation of the probabilities could not be sufficient to misclassify.

4.2. Generation of the adversarial examples

Obtaining at the same time imperceptibility and robustness is complicated. Typically, a robust attack would require perceptible input changes, while an imperceptible attack does not change the classification much. We propose an iterative methodology deploying a heuristic algorithm to automatically generate targeted imperceptible and robust adversarial examples in a black-box scenario, i.e., we assume that the attacker has access to the input image and to the output probabilities vector, but *not* to the network model. Fig. 5 shows our attack generation methodology and Algorithm 1 provides details. The goal of our methodology is to modify the input image to maximize the gap function (i.e., *imperceptibility*) until the distance between the original and the adversarial example is under D_{MAX} (i.e., *robustness*).

The algorithm considers that every pixel is composed of three different values, since the images are based on three channels (red, green and blue: RGB). Compared to the algorithm proposed by Luo et al. [24], we apply our attack to a set of pixels with the highest standard deviation at every iteration to create imperceptible perturbations. Moreover, *our algorithm automatically decides if it is more effective to add or subtract the noise, to maximize the gap function*, according to the values of two parameters, $GAP(+)$ and $GAP(-)$. These modifications increase the imperceptibility and the robustness of the attack. For clarity, we have expressed the formula used to compute the standard deviation in a more comprehensive form.

Algorithm 1 : Generating Adversarial Attacks

Given: original sample X, maximum human perceptual distance D_{MAX}, noise magnitude δ, $M \cdot N$ pixels, target class, P, V
while $D(X^*, X) < D_{MAX}$ **do**
-Compute *Standard Deviation SD* for every pixel
-Select a subset P of pixels included in the area of $M \cdot N$ pixels with the highest *SD* for every channel
-Compute GAP, $GAP(-)$, $GAP(+)$
if $GAP(-) > GAP(+)$ **then**
$VariationPriority(x_{i,j}) = [GAP(-) - GAP] \cdot SD(x_{i,j})$
else
$VariationPriority(x_{i,j}) = [GAP(+) - GAP] \cdot SD(x_{i,j})$
end if
-Sort in descending order *VariationPriority* for every channel
-Select V pixels with highest *VariationPriority* between the three channels
if $GAP(-) > GAP(+)$ **then**
Subtract noise with magnitude δ from the pixel in the respective channel
else
Add noise with magnitude δ to the pixel in the respective channel
end if
-Compute $D(X^*, X)$ as the sum of the $D(X^*, X)$ of every channel
-Update the original example with the adversarial one
end while

Our algorithm operates in the following steps:

1. Select a subset P of pixels, included in the area $M \cdot N$, with the highest SD for every RGB channel, so that their possible modification is difficult to detect.
2. Compute the gap function as the difference between the probability of the target class and the maximum output probability.
3. For each pixel of P, compute $GAP(+)$ and $GAP(-)$: these quantities correspond to the values of the gap function, estimated by adding and by subtracting, respectively, a perturbation unit to each pixel. These gaps are useful to decide if it is more effective to add or subtract the noise. For each pixel of P, we consider the greatest value between $GAP(+)$ and $GAP(-)$ to maximize the distance between the two probabilities.
4. For each pixel of P, calculate the Variation Priority by multiplying the gap difference to the SD of the pixel. This quantity indicates the efficacy of the pixel perturbation.
5. For every channel, we order the P values of Variation Priority and perturb the highest V values.

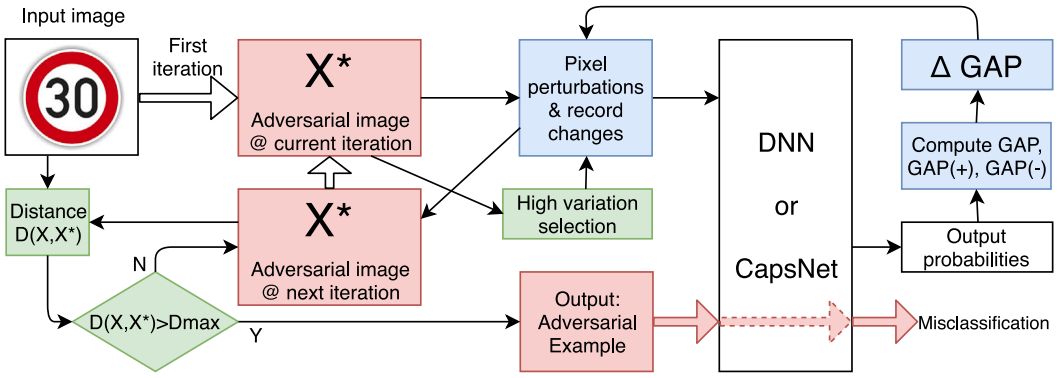

Fig. 5. Our methodology to generate adversarial examples. The blue-colored boxes are aimed to fool the network, while the green-colored boxes control the imperceptibility of the adversarial attack, whose images at the various steps are showed in the red boxes. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

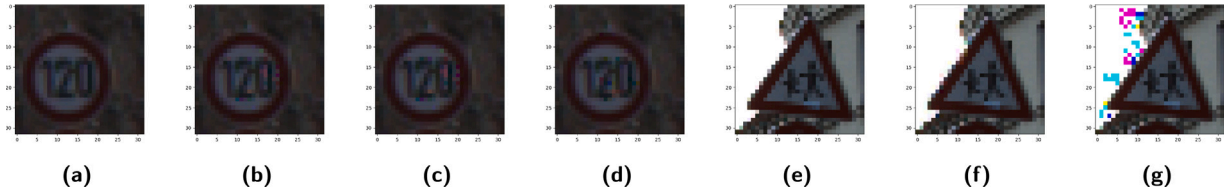

Fig. 6. Images for the attack applied to the CapsNet: (a) Original input image of Example 1. (b) Image misclassified by the CapsNet at iteration 13 for Case I. (c) Image misclassified by the CapsNet at iteration 16 for Case I. (d) Image at iteration 12 for Case II. (e) Original input image for Example 2. (f) Image at iteration 5, applied to the CapsNet. (g) Image misclassified by the CapsNet at iteration 21.

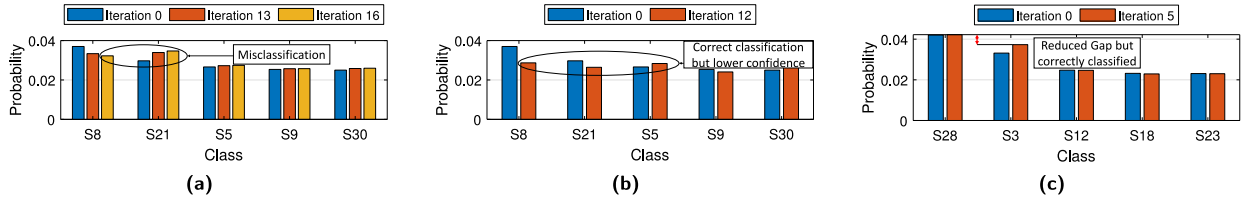

Fig. 7. CapsNet results for the GTSRB dataset: (a) Output probabilities of the Example 1 - Case I: blue bars represent the starting probabilities, orange bars the probabilities at the point of misclassification and yellow bars at the D_{MAX}. (b) Output probabilities of the Example 1 - Case II: blue bars represent the starting probabilities and orange bars the probabilities at the D_{MAX}. (c) Output probabilities of the Example 2: blue bars represent the starting probabilities, and orange bars the probabilities at the D_{MAX}. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

6. Only V values of the total $3 \cdot P$ are perturbed. We add or subtract the noise according to the highest value of the previously computed $GAP(+)$ and $GAP(-)$.
7. Once the original input image is replaced by the adversarial one, the next iteration starts. The iterations continue until the distance D overcomes D_{MAX}.

4.3. Attack methodology discussion

Note, if a valid solution cannot be found, i.e., a successful attack cannot be generated with sufficient imperceptibility, then our methodology will output the best possible solution that is misclassified and can potentially have slightly perceptible noise in certain regions. In practice, we can discard any such solution where the noise is perceptible, and re-run the algorithm for the next image at the operational time. It is important to note that there is a tradeoff between imperceptibility and robustness. Having both a very high robustness and a complete imperceptibility may be a really hard target to achieve, as typically achieving high robustness requires a strong noise which is more likely to be visible.

5. Evaluating our attack methodology

5.1. Experimental setup

We apply our algorithm, shown in Section 4.2, to the previously described CapsNet, DeepCaps, LeNet, VGGNet and ResNet. To verify how our algorithm works, we test it on two different examples. We consider M = N = 32, because the GTRSB and CIFAR10 datasets are composed of 32 × 32 images, P = 100 and V = 20. The value of δ is equal to the 10% of the maximum value between all the pixels. The parameter D_{MAX} depends on the SD of the pixels of the input image: its value changes according to the examples because $D(X^*, X)$ does not increase in the same way for each example.

5.2. Our methodology applied to the CapsNet

We test the CapsNet on two different examples of the GTSRB dataset, shown in Fig. 6(a) (Example 1) and Fig. 6(e) (Example 2). To test whether our methodology works independently from the choice of the target class, we distinguish two cases:

Case I: the target class is the class relative to the second highest initial output probability.

Case II: the target class is the class relative to the fifth highest probability between all the initial output probabilities.

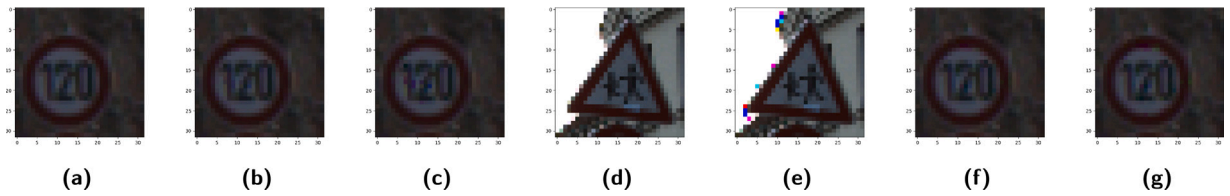

Fig. 8. Images for the attack applied to the CNNs: (a) Original input image for Example 1 (b) Image at the iteration 3, applied to the VGGNet. (c) Image at the iteration 9, misclassified by the VGGNet. (d) Original input image for Example 2. (e) Image at the iteration 2, applied to the VGGNet. (f) Image at the iteration 6, misclassified by the LeNet. (g) Image at the iteration 13, misclassified by the LeNet.

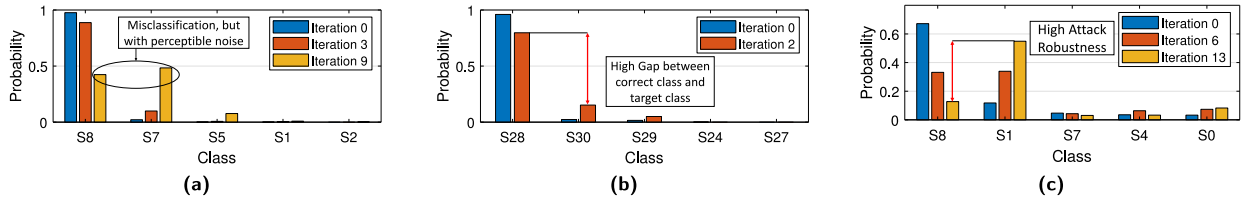

Fig. 9. CNNs results for the GTSRB dataset: (a) Output probabilities of the Example 1 on the VGGNet: blue bars represent the starting probabilities, orange bars the probabilities at the point of misclassification and yellow bars at the D_{MAX}. (b) Output probabilities of the Example 2 on the VGGNet: blue bars represent the starting probabilities and orange bars the probabilities at the D_{MAX}. (c) Output probabilities of the Example 1 on the LeNet: blue bars represent the starting probabilities, orange bars the probabilities at the point of misclassification and yellow bars at the D_{MAX}. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

By analyzing Case I and Case II, we can make the following key observations:

1. The CapsNet classifies the input image shown in Fig. 6(a) as "120 km/h speed limit" (S8) with a probability equal to 0.0370. For the Case I, the target class is "Double curve" (S21) with a probability equal to 0.0297. After 13 iterations of our algorithm, the image (Fig. 6(b)) is classified as "Double curve" with a probability equal to 0.0339. Hence, *the probability of the target class has overcome the initial one*, as shown in Fig. 7(a). At this step, the distance $D(X^*, X)$ is equal to 434.20. By increasing the number of iterations, the robustness of the attack increases as well, because the gap between the two probabilities increases, but also the perceptibility of the noise becomes more evident. After the iteration 16, the distance grows above $D_{MAX} = 520$: the sample is represented in Fig. 6(c). This analysis shows that there is a tradeoff between robustness and imperceptibility. For the Case II, the probability relative to the target class "Beware of ice/snow" (S30) is equal to 0.0249, as shown in Fig. 7(b). The gap between the maximum probability and the probability of the target class is higher than the gap in Case I. After 12 iterations, the network has not misclassified the image yet (Fig. 6(d)). In Fig. 7(b) we can observe that the gap between the two classes has decreased, but not enough for a misclassification. However, at this iteration, the value of the distance overcomes $D_{MAX} = 520$. In this case, we show that our algorithm would need more iterations to misclassify, at the cost of slightly perceivable perturbations.
2. The CapsNet classifies the input image shown in Fig. 6(e) as "Children crossing" (S28) with a probability equal to 0.042. The target class is "60 km/h speed limit" (S3) with a probability equal to 0.0331. After 5 iterations, the distance overcomes $D_{MAX} = 250$, while the network has not misclassified the image yet (Fig. 6(f)), because the probability of the target class does not overcome the initial maximum probability, as shown in Fig. 7(c). The misclassification appears at the iteration 21 (Fig. 6(g)). However, the perturbation is highly perceivable. Therefore, if this noise perception is not acceptable, such solution would be discarded by our methodology, and a new solution would be searched, for which an adaptation of the constraints may be required, or a different input image is captured in a real-world system (e.g., a new image at a different distance from the camera).

5.3. Our methodology applied to the VGGNet and the LeNet

To compare the robustness of the CapsNet and the 9-layer VGGNet, we choose to evaluate the previous two examples, which have been applied to the CapsNet. For the Example 1, we consider only the Case I as the benchmark, because the Case II shows a similar behavior. The VGGNet classifies the input images with different output probabilities, compared to the ones obtained by the CapsNet. Therefore, our metric to evaluate the resistance of the VGGNet against our attack is based on the value of the gap at the same distance.

To compare the robustness of the CapsNet and the 5-layer LeNet, we consider only the Example 1 (Fig. 8(a)), because Example 2 (Fig. 8(d)) is already classified incorrectly by the LeNet.[2] Applying our algorithm to the LeNet, we observe that it is more vulnerable than the CapsNet and the VGGNet.

From our experiments, we make these key observations:

1. The VGGNet classifies the input image (Fig. 8(a)) as "120 km/h speed limit" (S8) with a probability equal to 0.976. The target class is "100 km/h speed limit" (S7) with a probability equal to 0.021. After 3 iterations, the distance overcomes $D_{MAX} = 520$, while the VGGNet has not misclassified the image yet (Fig. 8(b)). Hence, our algorithm would need to perform more iterations before fooling the VGGNet, since the two initial probabilities were very distant, as shown in Fig. 9(a). Such scenario appears after 9 iterations (Fig. 8(c)), where the probability of the target class is 0.483.
2. The VGGNet classifies the input image (Fig. 8(d)) as "Children crossing" (S28) with a probability equal to 0.96. The target class is "Beware of ice/snow" (S30) with a probability equal to 0.023. After 2 iterations, the distance overcomes $D_{MAX} = 250$, while the VGGNet has not misclassified the image yet (Fig. 8(e)). As in the previous case, this scenario is due to the high gap between the initial probabilities, as shown in Fig. 9(b). We can also notice that the VGGNet reaches D_{MAX} in a lower number of iterations, as compared to the CapsNet.

[2] Note: the image of Example 2 belongs to the subset of images that are correctly classified by the CapsNet and the VGGNet, but incorrectly by the LeNet.

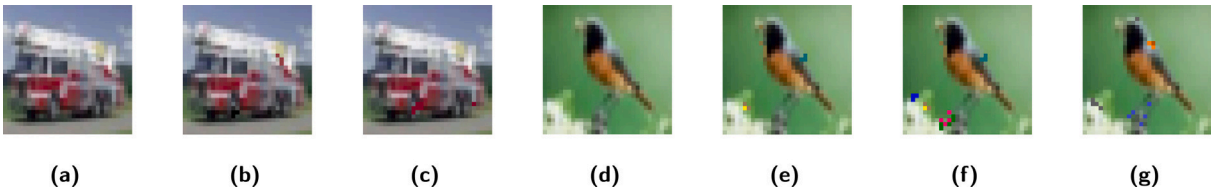

Fig. 10. Images for the attack applied to the DeepCaps and ResNet20: (a) Original input image for Example 3. (b) Image at the iteration 11, misclassified by the DeepCaps. (c) Image at the iteration 8, misclassified by the ResNet20. (d) Original input image for Example 4. (e) Image at the iteration 9, applied to the DeepCaps. (f) Image at the iteration 14, misclassified by the DeepCaps. (g) Image at the iteration 9, misclassified by the ResNet20.

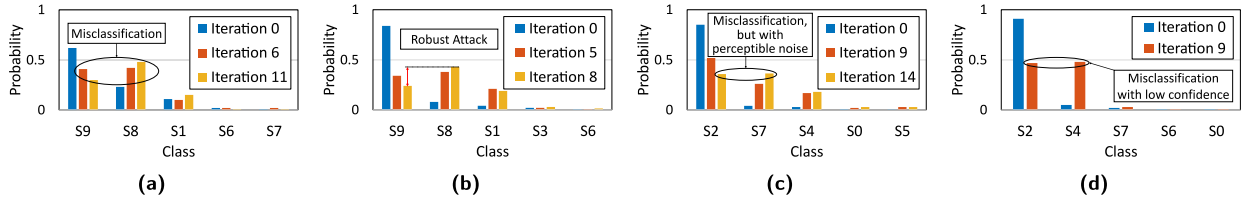

Fig. 11. CIFAR10 results: (a) Output probabilities of the Example 3 on the DeepCaps: blue bars represent the starting probabilities, orange bars represent the probabilities at the point of misclassification and yellow bars denote probabilities at the D_{MAX}. (b) Output probabilities of the Example 3 on the ResNet20: blue bars represent the starting probabilities, orange bars represent the probabilities at the point of misclassification and yellow bars denote probabilities at the D_{MAX}. (c) Output probabilities of the Example 4 on the DeepCaps: blue bars represent the starting probabilities, orange bars denote the probabilities at the D_{MAX}, yellow bars represent the point of misclassification with $D(X^*, X) > D_{MAX}$. (d) Output probabilities of the Example 4 on the ResNet20: blue bars represent the starting probabilities, orange bars denote the probabilities at the point of misclassification, which coincides to the D_{MAX}. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

3. The LeNet classifies the input image (Fig. 8(a)) as "120 km/h speed limit" (S8) with a probability equal to 0.672. The target class is "30 km/h speed limit" (S1) with a probability equal to 0.178. After 6 iterations, the perturbations fool the LeNet, because the image (Fig. 8(f)), is classified as the target class with a probability equal to 0.339. The perturbations become perceptible after 13 iterations (Fig. 8(g)), where the distance overcomes $D_{MAX} = 520$.

5.4. Our methodology applied to the DeepCaps and ResNet20 on the CIFAR10 dataset

We test the DeepCaps and ResNet on two different images of the CIFAR10 dataset, shown in Fig. 10(a) (Example 3) and in Fig. 10(d) (Example 4). Both analyses have been conducted by choosing the target class as the second highest probability between the initial output probabilities.

The following key observations can be derived from our experiments:

1. For the Example 3, both the DeepCaps and the ResNet correctly classify the input image (Fig. 10(a)) as "truck" (S9), and for both networks the target class is "ship" (S8). As shown in Fig. 11(a), *the DeepCaps is fooled by the attack*. After 6 iterations of the attack, the image is classified as "ship" with a probability of 0.421. By increasing the number of iterations, the gap between the probabilities increase, thus making the attack more robust. After 11 iterations, the distance has overcome $D_{MAX} = 520$, and Fig. 10(b) shows the adversarial example generated at this point. Similarly, *the attack also fools the ResNet*. After 5 iterations, it is classified as "ship" with a probability equal to 0.376. At the iteration 8 (see Fig. 10(c)), the probability associated to the class "ship"is 0.433, while the distance has overcome $D_{MAX} = 520$.
2. The Example 4 (Fig. 10(d)) is correctly classified as "bird" by the DeepCaps, with a probability equal to 0.847. The second highest probability, which will be the target class of the attack, is associated to the class "horse" (S7). After 9 iterations, the distance overcomes $D_{MAX} = 450$, while the image, shown in Fig. 10(e) is still correctly classified as "bird" by the DeepCaps with a probability equal to 0.524. The yellow bars in Fig. 11(c) show that the image in Fig. 10(f) is classified as "horse" with a probability equal to 0.365, but the distance is way beyond D_{MAX}. Hence, *only a perceptible noise can mislead the DeepCaps*.
3. The Example 4 (Fig. 10(d)) is correctly classified as "bird" by the ResNet, with a probability equal to 0.910. The target class is "deer" (S4). After 9 iterations, *the probability of the target class has overcome the initial one*. As shown in Fig. 11(d), the ResNet classifies the image in Fig. 10(g) as "deer" with a probability equal to 0.483. At this point, the distance has also reached $D_{MAX} = 450$.

5.5. Attack vulnerability comparison between the CapsNet and the CNNs

From our analyses, we can observe that the vulnerability of the 9-layer VGGNet to our adversarial attack is slightly lower than the vulnerability of the CapsNet, since the former one requires more perceivable perturbations to be fooled. Our observation is corroborated by the results in Fig. 12, where the value of $D(X^*, X)$ increases more sharply for the VGGNet than for the CapsNet. Hence, the noise perception in the image can be measured as the value of $D(X^*, X)$ divided by the number of iterations. Note, the noise in the VGGNet becomes perceivable after few iterations. Moreover, we can observe that the choice of the target class plays a key role for the success of the attack.

We also notice other features that lead to the differences between the VGGNet and the CapsNet. The VGGNet is deeper and contains a larger number of weights, while the CapsNet can achieve a similar accuracy with a smaller footprint. This effect causes a disparity in the prediction confidence between the two networks. It is clear that the CapsNet has a much higher learning capability, compared to the VGGNet, but this phenomena does not reflect in the prediction confidence. Indeed, comparing Figs. 7 and 9, we can notice that the output probabilities predicted by the CapsNet are close to each other, even more than the LeNet. However, the perturbations do not affect the output probabilities of the CapsNet as much as for the cases of the CNNs. The LeNet, even though it has a similar depth and similar number of parameters, is more vulnerable than the CapsNet.

By comparing the DeepCaps with the ResNet20, we can notice that, despite the prediction confidence is slightly higher for the ResNet, the DeepCaps is less vulnerable to these perturbations. For instance, as discussed in Section 5.4, after 9 iterations of the attack running on the Example 4, the ResNet is fooled, while the DeepCaps still correctly classifies.

Table 3
Comparison between the contributions in this work and the related works.

	Analyzed network models	Analyzed datasets	Affine transformation analysis & Techniques used	Adversarial attacks analysis & Techniques used
Gu et al. [34]	CNN, CapsNet, Proposed network	MNIST, Fashion-MNIST, SVHN, CIFAR10	✓ Thickness, width, shift, rotation	×
Michels et al. [5]	CapsNet, ConvNet	MNIST, Fashion-MNIST, SVHN, CIFAR10	×	✓ CW, Boundary, DeepFool, Universal
SeVuC (our work)	CapsNet, LeNet, VGGNet, DeepCaps, ResNet20	GTSRB, CIFAR10	✓ Zoom, shift, rotation	✓ Proposed Attack Methodology

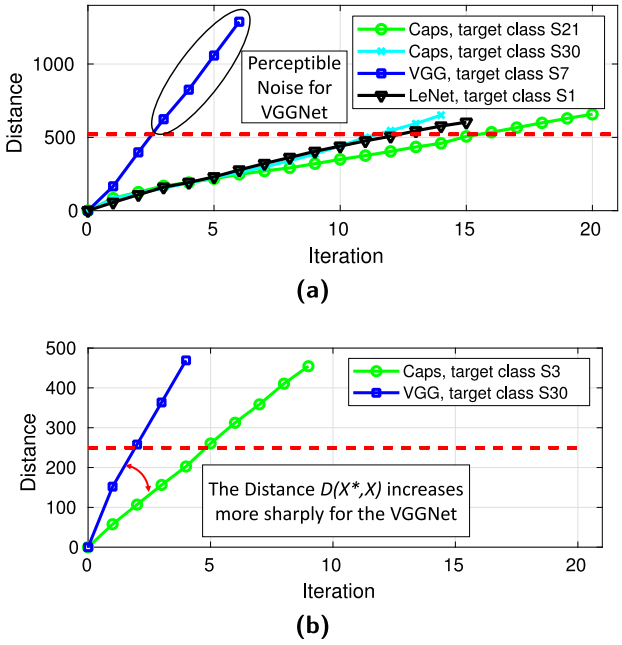

Fig. 12. $D(X^*, X)$ behavior for (a) Example 1, and (b) Example 2.

Moreover, recalling from Section 3, we observed that the VGGNet and the CapsNet are more resistant to the affine transformations compared to the LeNet, and that the DeepCaps is more resistant than the ResNet20. This behavior is consistent with the results obtained after applying our adversarial attack as well.

6. Discussion and comparison with the related works

In this work, we have studied the vulnerabilities of CapsNets to affine transformations and adversarial attacks by comparing them to traditional DNNs. Among the related works discussed in Section 2.2, the works of Gu et al. [34] and Michels et al. [5] represent the most closely-related analyses to our work. However, Gu et al. only analyze the robustness against affine transformation, and Michels et al. only study the vulnerability against adversarial attacks. On the other hand, We perform a comprehensive analysis that investigates both types of vulnerabilities. Table 3 summarizes the key differences between our works and the related works. Moreover, we also conduct experiments on the GTSRB dataset, which is not frequently used in prior works despite providing a valuable benchmark for traffic sign recognition. Furthermore, we provide detailed analyses of the output probability variations for different images under various perturbations, which constitutes useful information for understanding the functionality of the process.

7. Conclusion

This paper studied the robustness of CapsNets under adversarial attacks and affine transformations. We proposed an iterative methodology to generate targeted adversarial attacks in a black box scenario. We applied our attack to the GTSRB dataset and analyzed its impact on a CapsNet, a 5-layer LeNet and a 9-layer VGGNet, and to the CIFAR10 dataset analyzing the impact on the DeepCaps and a 20-layer ResNet. Our experiments show that the CapsNet appears more robust than the LeNet to the attack, but slightly less than the VGGNet. However, the modifications of the pixels in the traffic signs are less perceivable when our algorithm is applied to the CapsNet, rather than to the VGGNet. A serious issue for the CapsNet is that the gap between the output probabilities is lower than the one computed on the VGGNet. However, the changes of the probabilities at the output of the CapsNet is lower than their changes in the VGGNet. Hence, further modifications of the CapsNet algorithm, aiming to increase the prediction confidence, would be beneficial to improve its robustness. Towards this, an example is represented by the DeepCaps, which appears more robust than the ResNet, under similar attack settings.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgments

This work has been supported in part by the Doctoral College Resilient Embedded Systems, which is run jointly by the TU Wien's Faculty of Informatics and the UAS Technikum Wien. This work was also jointly supported by the NYUAD Center for Interacting Urban Networks (CITIES), funded by Tamkeen under the NYUAD Research Institute Award CG001 and by the Swiss Re Institute under the Quantum Cities™ initiative, and Center for CyberSecurity (CCS), funded by Tamkeen under the NYUAD Research Institute Award G1104. The authors acknowledge TU Wien Bibliothek for financial support through its Open Access Funding Programme.

References

[1] A. Bhandare, M. Bhide, P. Gokhale, R. Chandavarkar, Applications of convolutional neural networks, Int. J. Comput. Sci. Inf. Technol. (IJCSIT) (2016).

[2] M. Capra, B. Bussolino, A. Marchisio, G. Masera, M. Martina, M. Shafique, Hardware and software optimizations for accelerating deep neural networks: Survey of current trends, challenges, and the road ahead, IEEE Access 8 (2020) 225134–225180, http://dx.doi.org/10.1109/ACCESS.2020.3039858.

[3] S. Sabour, N. Frosst, G.E. Hinton, Dynamic routing between capsules, in: I. Guyon, U. von Luxburg, S. Bengio, H.M. Wallach, R. Fergus, S.V.N. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, 2017, pp. 3856–3866, URL https://proceedings.neurips.cc/paper/2017/hash/2cad8fa47bbef282badbb8de5374b894-Abstract.html.

[4] R. Mukhometzianov, J. Carrillo, CapsNet comparative performance evaluation for image classification, 2018, CoRR abs/1805.11195. arXiv:1805.11195. URL http://arxiv.org/abs/1805.11195.

[5] F. Michels, T. Uelwer, E. Upschulte, S. Harmeling, On the vulnerability of capsule networks to adversarial attacks, 2019, CoRR abs/1906.03612. arXiv:1906.03612. URL http://arxiv.org/abs/1906.03612.

[6] M. Shafique, A. Marchisio, R.V.W. Putra, M.A. Hanif, Towards energy-efficient and secure edge AI: A cross-layer framework ICCAD special session paper, in: IEEE/ACM International Conference on Computer Aided Design, ICCAD 2021, Munich, Germany, November 1-4, 2021, IEEE, 2021, pp. 1–9, http://dx.doi.org/10.1109/ICCAD51958.2021.9643539.

[7] N. Carlini, P. Mishra, T. Vaidya, Y. Zhang, M. Sherr, C. Shields, D.A. Wagner, W. Zhou, Hidden voice commands, in: T. Holz, S. Savage (Eds.), 25th USENIX Security Symposium, USENIX Security 16, Austin, TX, USA, August 10-12, 2016, USENIX Association, 2016, pp. 513–530, URL https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/carlini.

[8] G. Zhang, C. Yan, X. Ji, T. Zhang, T. Zhang, W. Xu, DolphinAttack: Inaudible voice commands, in: B.M. Thuraisingham, D. Evans, T. Malkin, D. Xu (Eds.), Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017, ACM, 2017, pp. 103–117, http://dx.doi.org/10.1145/3133956.3134052.

[9] A. Kurakin, I.J. Goodfellow, S. Bengio, Adversarial examples in the physical world, in: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings, OpenReview.net, 2017, URL https://openreview.net/forum?id=HJGU3Rodl.

[10] X. Yuan, P. He, Q. Zhu, X. Li, Adversarial examples: Attacks and defenses for deep learning, IEEE Trans. Neural Netw. Learn. Syst. 30 (9) (2019) 2805–2824, http://dx.doi.org/10.1109/TNNLS.2018.2886017.

[11] A. Shafahi, W.R. Huang, C. Studer, S. Feizi, T. Goldstein, Are adversarial examples inevitable? in: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019, OpenReview.net, 2019, URL https://openreview.net/forum?id=r1lWUoA9FQ.

[12] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, C. Igel, Detection of traffic signs in real-world images: The German traffic sign detection benchmark, in: The 2013 International Joint Conference on Neural Networks, IJCNN 2013, Dallas, TX, USA, August 4-9, 2013, IEEE, 2013, pp. 1–8, http://dx.doi.org/10.1109/IJCNN.2013.6706807.

[13] G.E. Hinton, A. Krizhevsky, S.D. Wang, Transforming auto-encoders, in: T. Honkela, W. Duch, M.A. Girolami, S. Kaski (Eds.), Artificial Neural Networks and Machine Learning - ICANN 2011 - 21st International Conference on Artificial Neural Networks, Espoo, Finland, June 14-17, 2011, Proceedings, Part I, in: Lecture Notes in Computer Science, vol. 6791, Springer, 2011, pp. 44–51, http://dx.doi.org/10.1007/978-3-642-21735-7_6.

[14] G.E. Hinton, S. Sabour, N. Frosst, Matrix capsules with EM routing, in: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings, OpenReview.net, 2018, URL https://openreview.net/forum?id=HJWLfGWRb.

[15] A.D. Kumar, Novel deep learning model for traffic sign detection using capsule networks, 2018, CoRR abs/1805.04424. arXiv:1805.04424. URL http://arxiv.org/abs/1805.04424.

[16] J. Rajasegaran, V. Jayasundara, S. Jayasekara, H. Jayasekara, S. Seneviratne, R. Rodrigo, DeepCaps: Going deeper with capsule networks, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019, Computer Vision Foundation / IEEE, 2019, pp. 10725–10733, http://dx.doi.org/10.1109/CVPR.2019.01098, URL http://openaccess.thecvf.com/content_CVPR_2019/html/Rajasegaran_DeepCaps_Going_Deeper_With_Capsule_Networks_CVPR_2019_paper.html.

[17] A. Krizhevsky, Learning Multiple Layers of Features from Tiny Images, University of Toronto, 2012.

[18] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I.J. Goodfellow, R. Fergus, Intriguing properties of neural networks, in: Y. Bengio, Y. LeCun (Eds.), 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings, 2014, URL http://arxiv.org/abs/1312.6199.

[19] I.J. Goodfellow, J. Shlens, C. Szegedy, Explaining and harnessing adversarial examples, in: Y. Bengio, Y. LeCun (Eds.), 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015, URL http://arxiv.org/abs/1412.6572.

[20] R. Feinman, R.R. Curtin, S. Shintre, A.B. Gardner, Detecting adversarial samples from artifacts, 2017, CoRR abs/1703.00410. arXiv:1703.00410. URL http://arxiv.org/abs/1703.00410.

[21] A.N. Bhagoji, D. Cullina, P. Mittal, Dimensionality reduction as a defense against evasion attacks on machine learning classifiers, 2017, CoRR abs/1704.02654. arXiv:1704.02654. URL http://arxiv.org/abs/1704.02654.

[22] A.N. Bhagoji, D. Cullina, C. Sitawarin, P. Mittal, Enhancing robustness of machine learning systems via data transformations, in: 52nd Annual Conference on Information Sciences and Systems, CISS 2018, Princeton, NJ, USA, March 21-23, 2018, IEEE, 2018, pp. 1–5, http://dx.doi.org/10.1109/CISS.2018.8362326.

[23] N. Papernot, P.D. McDaniel, I.J. Goodfellow, S. Jha, Z.B. Celik, A. Swami, Practical black-box attacks against machine learning, in: R. Karri, O. Sinanoglu, A. Sadeghi, X. Yi (Eds.), Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, AsiaCCS 2017, Abu Dhabi, United Arab Emirates, April 2-6, 2017, ACM, 2017, pp. 506–519, http://dx.doi.org/10.1145/3052973.3053009.

[24] B. Luo, Y. Liu, L. Wei, Q. Xu, Towards imperceptible and robust adversarial example attacks against neural networks, in: S.A. McIlraith, K.Q. Weinberger (Eds.), Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th Innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018, AAAI Press, 2018, pp. 1652–1659, URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16217.

[25] N. Carlini, D.A. Wagner, Towards evaluating the robustness of neural networks, in: 2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017, IEEE Computer Society, 2017, pp. 39–57, http://dx.doi.org/10.1109/SP.2017.49.

[26] W. Brendel, J. Rauber, M. Bethge, Decision-based adversarial attacks: Reliable attacks against black-box machine learning models, in: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings, OpenReview.net, 2018, URL https://openreview.net/forum?id=SyZI0GWCZ.

[27] S. Moosavi-Dezfooli, A. Fawzi, P. Frossard, DeepFool: A simple and accurate method to fool deep neural networks, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016, IEEE Computer Society, 2016, pp. 2574–2582, http://dx.doi.org/10.1109/CVPR.2016.282.

[28] S. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, P. Frossard, Universal adversarial perturbations, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017, IEEE Computer Society, 2017, pp. 86–94, http://dx.doi.org/10.1109/CVPR.2017.17.

[29] N. Frosst, S. Sabour, G.E. Hinton, DARCCC: Detecting adversaries by reconstruction from class conditional capsules, 2018, CoRR abs/1811.06969. arXiv:1811.06969. URL http://arxiv.org/abs/1811.06969.

[30] H. Xiao, K. Rasul, R. Vollgraf, Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms, 2017, CoRR abs/1708.07747. arXiv:1708.07747. URL http://arxiv.org/abs/1708.07747.

[31] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, A.Y. Ng, Reading digits in natural images with unsupervised feature learning, in: NIPS Workshop, 2011.

[32] Y. Qin, N. Frosst, S. Sabour, C. Raffel, G.W. Cottrell, G.E. Hinton, Detecting and diagnosing adversarial images with class-conditional capsule reconstructions, in: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020, OpenReview.net, 2020, URL https://openreview.net/forum?id=Skgy464Kvr.

[33] Y. Qin, N. Frosst, C. Raffel, G.W. Cottrell, G.E. Hinton, Deflecting adversarial attacks, 2020, CoRR abs/2002.07405. arXiv:2002.07405. URL https://arxiv.org/abs/2002.07405.

[34] J. Gu, V. Tresp, Improving the robustness of capsule networks to image affine transformations, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020, Computer Vision Foundation / IEEE, 2020, pp. 7283–7291, http://dx.doi.org/10.1109/CVPR42600.2020.00731, URL https://openaccess.thecvf.com/content_CVPR_2020/html/Gu_Improving_the_Robustness_of_Capsule_Networks_to_Image_Affine_Transformations_CVPR_2020_paper.html.

[35] Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE (1998).

[36] L. Wang, S. Guo, W. Huang, Y. Qiao, Places205-vggnet models for scene recognition, 2015, CoRR abs/1508.01667. arXiv:1508.01667. URL http://arxiv.org/abs/1508.01667.

[37] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016, IEEE Computer Society, 2016, pp. 770–778, http://dx.doi.org/10.1109/CVPR.2016.90.

[38] C. Guo, G. Pleiss, Y. Sun, K.Q. Weinberger, On calibration of modern neural networks, in: D. Precup, Y.W. Teh (Eds.), Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017, in: Proceedings of Machine Learning Research, vol. 70, PMLR, 2017, pp. 1321–1330, URL http://proceedings.mlr.press/v70/guo17a.html.

Alberto Marchisio received his B.Sc. and M.Sc. degrees in Electronic Engineering from Politecnico di Torino, Turin, Italy, in October 2015 and April 2018, respectively. Currently, he is Ph.D. Student at Computer Architecture and Robust Energy-Efficient Technologies (CARE-Tech.) lab, Institute of Computer Engineering, Technische Universität Wien (TU Wien), Vienna, Austria, under the supervision of Prof. Dr. Muhammad Shafique. His main research interests include hardware and software optimizations for machine learning, brain-inspired computing, VLSI architecture design, emerging computing technologies, robust design, and approximate computing for energy efficiency. He (co-)authored 20+ papers in prestigious international conferences and journals. He received the honorable mention at the Italian National Finals of Maths Olympic Games in 2012, and the Richard Newton Young Fellow Award in 2019.

Giorgio Nanfa works as Software Developer for Accenture in Turin, Italy, from November 2019. He attended Politecnico di Torino, Turin, Italy, from 2013 to 2019, obtaining his B.Sc. degree in Electronic Engineering in September 2016 and his M.Sc. degree in Electronic Engineering (Electronic Systems) in April 2019. His Master Thesis work in 2019 was a joint research project between Politecnico di Torino and Technische Universität Wien (TU Wien), Vienna, Austria. He spent some months in Vienna working at Computer Architecture and Robust Energy-Efficient Technologies (CARE-Tech.) lab, Institute of Computer Engineering, TU Wien. His work, supervised by Prof. Dr. Maurizio Martina and Prof. Dr. Muhammad Shafique, was based on Machine Learning Security, in particular on Black-Box Adversarial Attacks, and Spiking Neural Networks. After a brief experience as Deep Learning Scientist, he focused on Embedded Systems programming and Software applications development.

Faiq Khalid received his M.S. degree in electrical engineering and his B.E. degree in electronics engineering from the National University of Sciences and Technology (NUST), Pakistan, in 2016 and in 2011, respectively. He is currently pursuing his Ph.D. degree in hardware security and machine learning security at Technische Universität Wien (TU Wien), Vienna, Austria. He is a recipient of the Quaid-e-Azam Gold Medal for his academic achievements and the Richard Newton Young Fellowship Award at DAC 2018. His research interests include formal verification of embedded systems, hardware design security, and security for machine learning systems. He has also served as a TPC member of FIT, WSAV, ARES and ICONS.

Muhammad Abdullah Hanif received the B.Sc. degree in electronic engineering from Ghulam Ishaq Khan Institute of Engineering Sciences and Technology (GIKI), Pakistan, and the M.Sc. degree in electrical engineering with specialization in digital systems and signal processing (DSSP) from National University of Sciences and Technology (NUST), Islamabad, Pakistan. He was a University Assistant with the Department of Informatics, Institute of Computer Engineering, Technische Universität Wien (TU Wien), Austria. He is currently pursuing the Ph.D. degree in computer engineering from TU Wien and working at New York University Abu Dhabi (NYUAD), UAE. His current research interests include brain-inspired computing, machine learning, approximate computing, computer architecture, energy-efficient design, robust computing, system-on-chip design, and emerging technologies.

Maurizio Martina received the M.S. and Ph.D. in electrical engineering from Politecnico di Torino, Italy, in 2000 and 2004, respectively. He is currently Full Professor with the VLSI-Lab group, Politecnico di Torino. His research interests include computer architecture and VLSI design of architectures for digital signal processing, video coding, communications, networking, artificial intelligence, machine learning and event-based processing. He edited one book and published 3 book chapters on VLSI architectures and digital circuits for video coding, wireless communications and error correcting codes. He has more than 100 scientific publications and is co-author of 2 patents. He is now an Associate Editor of IEEE Transactions on Circuits and Systems - I. He had been part of the organizing and technical committee of several international conferences, including BioCAS 2017, ICECS 2019, AICAS 2020. Currently, he is the counselor of the IEEE Student Branch at Politecnico di Torino and a professional member of IEEE HKN.

Muhammad Shafique received the Ph.D. degree in computer science from the Karlsruhe Institute of Technology (KIT), Germany, in 2011. Afterwards, he established and led a highly recognized research group at KIT for several years as well as conducted impactful collaborative R&D activities across the globe. In Oct.2016, he joined the Institute of Computer Engineering at the Faculty of Informatics, Technische Universität Wien (TU Wien), Vienna, Austria as a Full Professor of Computer Architecture and Robust, Energy-Efficient Technologies. Since Sep.2020, Dr. Shafique is with the New York University (NYU), where he is currently a Full Professor and the director of the eBrain Lab at the NYU-Abu Dhabi in UAE, and a Global Network Professor at the Tandon School of Engineering, NYU-New York City in USA. He is also a Co-PI/Investigator in multiple NYUAD Centers, including Center of Artificial Intelligence and Robotics (CAIR), Center of Cyber Security (CCS), Center for InTeractIng urban nEtworkS (CITIES), and Center for Quantum and Topological Systems (CQTS). His research interests are in AI & machine learning hardware and system-level design, brain-inspired computing, autonomous systems, wearable healthcare, energy-efficient systems, robust computing, hardware security, emerging technologies, FPGAs, MPSoCs, and embedded systems. His research has a special focus on cross-layer analysis, modeling, design, and optimization of computing and memory systems. The researched technologies and tools are deployed in application use cases from Internet-of-Things (IoT), smart Cyber–Physical Systems (CPS), and ICT for Development (ICT4D) domains. Dr. Shafique has given several Keynotes, Invited Talks, and Tutorials, as well as organized many special sessions at premier venues. He has served as the PC Chair, General Chair, Track Chair, and PC member for several prestigious IEEE/ACM conferences. Dr. Shafique holds one U.S. patent has (co-)authored 6 Books, 15+ Book Chapters, 350+ papers in premier journals and conferences, and 50+ archive articles. He received the 2015 ACM/SIGDA Outstanding New Faculty Award, the AI 2000 Chip Technology Most Influential Scholar Award in 2020 and 2022, the ASPIRE AARE Research Excellence Award in 2021, six gold medals, and several best paper awards and nominations at prestigious conferences. He is a senior member of the IEEE and IEEE Signal Processing Society (SPS), and a member of the ACM, SIGARCH, SIGDA, SIGBED, and HIPEAC.