SymDNN: Simple & Effective Adversarial Robustness for Embedded Systems

Swarnava Dey  Pallab Dasgupta  Partha P Chakrabarti
Indian Institute of Technology Kharagpur, Kharagpur, India - 721302
swarnava.dey@kgpian.iitkgp.ac.in, {pallab, ppchak}@cse.iitkgp.ac.in

Abstract

We propose SymDNN, a Deep Neural Network (DNN) inference scheme, to segment an input image into small patches, replace those patches with representative symbols, and use the reconstructed image for CNN inference. This approach of deconstruction of images, and the reconstruction from cluster centroids trained on clean images, enhances robustness against adversarial attacks. The input transform used in SymDNN is learned from very large datasets, making it difficult to approximate for adaptive adversarial attacks. For example, SymDNN achieves 23% and 42% robust accuracy at $L_\infty$ attack strengths of $8/255$ and $4/255$ respectively, against BPDA under a complete white box setting, where most input processing based defenses break completely. SymDNN is not a future-proof adversarial defense that can defend any attack, but it is one of the few readily usable defenses in resource-limited embedded systems that defends against a wide range of attacks. Our code is available at: https://github.com/swadeykgp/SymDNN.

1. Introduction

Convolutional Neural Networks (CNNs) can automatically learn effective features from images, making those suitable in many computer vision tasks as classifiers and backbone feature extractors. Since the past few years, Edge Computing is showing explosive growth [51], with mobile and embedded computer vision being one of the killer apps [2], and CNNs dominating that landscape [13].

Albeit being one of the most popular architectures for image tasks, a CNN inference can be forced to generate unexpected output on images that contain visually imperceptible, well crafted modifications [8], referred to as adversarial perturbations [57]. Brittleness of CNNs against adversarial attacks make those unsuitable for deployment in safety critical systems. With adversarial attacks very much possible in a real world setting [32], the mobile and embedded vision tasks on resource-limited systems are also affected by this.

Despite significant progress in the research on adversarial robustness, there are hardly any studies targeting robustness under adversarial attacks on embedded systems.

Brief Background of Adversarial Robustness. Adversarial attacks aim to perturb a benign input with small changes to create a malicious input, such that the CNN output differs by a significant extent. For instance, in a successful targeted attack on the input to a CNN based classifier, the classifier is forced to assign a class that is desired by the adversary. Whereas in a successful untargeted attack, a CNN classifier fails to assign the same class to the benign example and the visually similar adversarial example. The difference between a clean and the corresponding adversarial example is often quantified using $l_p$ norms ($p \in \{1, 2, \infty\}$).

Adversarial examples can be generated in a complete white box setting, where the model parameters (e.g., architecture, loss function etc.), and the defense parameters (e.g., transform, randomness) are known. Effective adversarial inputs [43] can also be generated in a complete black box setting, where the adversary has no access to the model and defense. Defenses against the adversarial attacks often claim robustness in the above two and several different intermediate threat models.

In summary, some of the very strong attacks include enhancements of PGD [37] attack (e.g., APGD [19], EOTPGD [5]), translation invariant versions of FGSM [26] attack (e.g., TI-FGSM [23], MI-FGSM [23]), ensemble attacks (e.g., AutoAttacks [19]), and finally customized attacks adapted [58] to each defense.

We refer the reader to a recent survey [48] and a series of works [4,8,9,11,15,19,23,26,37,43,44,57,58] that have shaped the field of adversarial robustness during the past eight years.

The adversarial defenses render a CNN inference robust to the adversarial attacks. It is believed that the process of training a DNN, that tries to generalize on an unconstrained, real-valued input space, based on a finite training set example, leads to imperfect generalization [7,26,44].

Adversarial Defenses for Embedded Systems? To address the problem of adversarial attacks, the strongest defenses, that is, the ones that are provably robust [15,16,40,
In this context we propose SymDNN, a model agnostic pre-processing step that can boost the accuracy of any arbitrary CNN for adversarial images with a limited change per pixel. We observe this gain in robust accuracy under complete black box and partial white box threat models, where the model parameters and dataset are known. We have extensively evaluated SymDNN under these threat models and show that SymDNN boosts the robust accuracy of a ResNet model in the face of several recent strong attacks, namely enhanced PGD variants [5,19,37], translation invariant attacks [22,23] and ensemble attacks [19], by 30-50% at attack strength of 4/255 and by at least 10% at attack strength of 8/255.

In a complete white box setting, where the transformation due to the defense is also known to the adversary, SymDNN exhibits the gradient shattering property [4]. We believe that the transformations used in earlier input processing based defenses [10, 27, 67] were based on some analytical formulation, which made it possible for the adaptive attacks to easily use a surrogate approximation to perform gradient descent and find the adversarial examples even if the gradients were obfuscated. For SymDNN, the computational overhead of building this transformation function from large image datasets, is prohibitive even using a fast indexing and clustering library. For the same reason, approximating such a function is difficult, even if gradients are obfuscated [4]. To support this belief, SymDNN accuracy drops to 23%, far better than other defenses that drop to 0% [4, 58] under Backward Pass Differentiable Approximation (BPDA) attack.

We do not claim that SymDNN can defend any adversarial attack, which is what the adversarial research community aims at. The major contributions of SymDNN are

- We associate symbols with the clusters and store these associations in a codebook.
- The image is coded using the codebook by replacing each patch with a symbol. The symbolic coded image is orders of magnitude smaller than the original image, which may be useful for communication and / or storage.
- For CNN inference, an approximation of the image is reconstructed by replacing each symbol with the centroid of the cluster it represents. The reconstructed image is then used as input by CNN.
- When the image is adversarially perturbed, this reconstruction process removes some of the adversarial changes, as the centroids that are used to replace the image patches are learned from the clean images.

In practice, we implement SymDNN training and index search with a fast similarity search [29], to handle large patch datasets. The theoretical and algorithmic basis for this approach has been detailed in this paper.

SymDNN is a model agnostic pre-processing step that can boost the accuracy of any arbitrary CNN for adversarial images with a limited change per pixel. We observe this gain in robust accuracy under complete black box and partial white box threat models, where the model parameters and dataset are known. We have extensively evaluated SymDNN under these threat models and show that SymDNN boosts the robust accuracy of a ResNet model in the face of several recent strong attacks, namely enhanced PGD variants [5,19,37], translation invariant attacks [22,23] and ensemble attacks [19], by 30-50% at attack strength of 4/255 and by at least 10% at attack strength of 8/255.

In a complete white box setting, where the transformation due to the defense is also known to the adversary, SymDNN exhibits the gradient shattering property [4]. We believe that the transformations used in earlier input processing based defenses [10, 27, 67] were based on some analytical formulation, which made it possible for the adaptive attacks to easily use a surrogate approximation to perform gradient descent and find the adversarial examples even if the gradients were obfuscated. For SymDNN, the computational overhead of building this transformation function from large image datasets, is prohibitive even using a fast indexing and clustering library. For the same reason, approximating such a function is difficult, even if gradients are obfuscated [4]. To support this belief, SymDNN accuracy drops to 23%, far better than other defenses that drop to 0% [4, 58] under Backward Pass Differentiable Approximation (BPDA) attack.

We do not claim that SymDNN can defend any adversarial attack, which is what the adversarial research community aims at. The major contributions of SymDNN are

- In the offline training phase we take all the images from the training set, divide them into small patches, accumulate these patches into a single patch dataset, and apply unsupervised clustering on this dataset.

Proposed Approach. In this context we propose SymDNN, yet another input processing based defense that enhances adversarial robustness of a CNN for small attack strengths, and under various threat models. The broad working principle of SymDNN is shown in Fig. 1.

- We associate symbols with the clusters and store these associations in a codebook.
- The image is coded using the codebook by replacing each patch with a symbol. The symbolic coded image is orders of magnitude smaller than the original image, which may be useful for communication and / or storage.
- For CNN inference, an approximation of the image is reconstructed by replacing each symbol with the centroid of the cluster it represents. The reconstructed image is then used as input by CNN.
- When the image is adversarially perturbed, this reconstruction process removes some of the adversarial changes, as the centroids that are used to replace the image patches are learned from the clean images.

In practice, we implement SymDNN training and index search with a fast similarity search [29], to handle large patch datasets. The theoretical and algorithmic basis for this approach has been detailed in this paper.

SymDNN is a model agnostic pre-processing step that can boost the accuracy of any arbitrary CNN for adversarial images with a limited change per pixel. We observe this gain in robust accuracy under complete black box and partial white box threat models, where the model parameters and dataset are known. We have extensively evaluated SymDNN under these threat models and show that SymDNN boosts the robust accuracy of a ResNet model in the face of several recent strong attacks, namely enhanced PGD variants [5,19,37], translation invariant attacks [22,23] and ensemble attacks [19], by 30-50% at attack strength of 4/255 and by at least 10% at attack strength of 8/255.

In a complete white box setting, where the transformation due to the defense is also known to the adversary, SymDNN exhibits the gradient shattering property [4]. We believe that the transformations used in earlier input processing based defenses [10, 27, 67] were based on some analytical formulation, which made it possible for the adaptive attacks to easily use a surrogate approximation to perform gradient descent and find the adversarial examples even if the gradients were obfuscated. For SymDNN, the computational overhead of building this transformation function from large image datasets, is prohibitive even using a fast indexing and clustering library. For the same reason, approximating such a function is difficult, even if gradients are obfuscated [4]. To support this belief, SymDNN accuracy drops to 23%, far better than other defenses that drop to 0% [4, 58] under Backward Pass Differentiable Approximation (BPDA) attack.
as follows: (A) Forming a first line of readily usable defense against a wide range of adversarial attacks with very little computational overhead that embedded vision systems currently lack, and (B) Significant reduction in the size of the images in their symbolic forms without any significant negative impact on their classification post reconstruction.

2. Related Work

The method proposed in this paper is in principle similar to all input processing based defenses, which cleans or purifies an image before DNN inference.

Previous works [6, 14, 27, 35, 36, 39, 67] have used specialized processing on the input or features to nullify the effects of adversarial perturbations. In [27] total variance minimization is used to reconstruct pixels of an image based on its neighboring pixels, with an aim of smoothing out small and localized perturbations. This work also uses image-quilting, where an image is reconstructed using patches from an image patch dataset. This is very similar to SymDNN. However, neither this paper [27], nor the original paper on patch based texture synthesis [25], elaborate the patch dataset, resolution of the patches and the computation overhead for texture synthesis. For SymDNN, this patch K-nearest neighbor graph is built from the full training dataset, rendering the image reconstruction process hard to approximate.

Another approach is using discretization or quantization of attacked image pixels [10, 62, 69]. In [62, 69] K-means clustering is used to quantize each pixel of an attacked image into 2-5 levels to reduce perturbation and achieve robustness against simple untargeted attacks. However, there are no details on how such a clustering model is trained for quantizing image pixels. In [10] a specialized thermometer encoding is proposed that preserves image properties, inference accuracy and enhances adversarial robustness for a limited set of adversarial attacks. However, most of these input processing based defenses are now considered ineffective, as these methods obfuscate gradients and therefore are often only effective against gradient based attacks [4, 58]. These methods are ineffective in the face of strong black-box attacks that are not gradient dependent and against adaptive attacks that can easily approximate the simple transformation function to perform gradient descent.

In [45] it is shown that a series of transforms can prevent adaptive attacks. Another recent input processing defense [42] using an autoencoder based purifier, shows gains in adversarial robustness under various threat models except complete white box. However, a series of transforms or autoencoders are computationally demanding.

SymDNN addresses these problems of adversarial robustness under different threat models, with a single, computationally efficient transformation function that is hard to approximate.

3. The Goal of Symbolic Abstraction

Our goal is to create a symbolic abstraction of an image which is compact, inference preserving, and robust to adversarial attacks. In formal terms, given an input $x \in \mathbb{R}^{I_h \times I_w \times C}$ from an input space $\mathcal{X}$ and a set of labels $y_1, y_2, \ldots, y_k$ from an output space $\mathcal{Y}$, a DNN learns a non-linear mapping $\mathcal{N}_\theta : \mathcal{X} \rightarrow \mathcal{Y}$, where $\theta$ denotes the network parameters, and $I_h, I_w$, and $C$ are respectively the height (in pixels), width (in pixels), and number of channels of the image, $x$. We aim to learn a mapping, $\tau : \mathcal{X} \rightarrow \Sigma^\tau$, where $\Sigma^\tau$ is a symbolic abstraction. We also define the mapping: $\tau_R : \Sigma^\tau \rightarrow \mathcal{X}$, which reconstructs an image from the symbolic abstraction. We address the following requirements:

1. **Compaction.** We show that $\tau(x)$ is orders of magnitude smaller than $x$. This is useful when the image has to be communicated, say, from an edge device to an edge server.

2. **Inference Preserving.** Our experiments show that the abstraction is almost always inference preserving, that is, $\mathcal{N}_\theta(x) = \mathcal{N}_\theta(\tau_R \tau(x))$.

3. **Enhancing Robustness to Adversarial Attacks.** If an adversarial attack modifies $x$ to $x'$, we show that $\mathcal{N}_\theta(x) \neq \mathcal{N}_\theta(\tau_R \tau(x'))$ in most cases, even when the attack succeeds on the original image, that is, $\mathcal{N}_\theta(x) \neq \mathcal{N}_\theta(x')$. We propose a model agnostic scheme to defend a DNN inference against adversarial attacks.

In cases where enhancing robustness is the primary goal, we do not need to learn $\tau$ and $\tau_R$ separately. Instead, it suffices to learn the combined function, $\tau_R \tau : \mathcal{X} \rightarrow \mathcal{X}$, and pass on $\tau_R \tau(x')$ instead of $x'$ to the DNN classifier.

4. The SymDNN Methodology

The SymDNN methodology has two broad stages. The first stage learns an alphabet $\Sigma$ from the training dataset, which forms the basis for defining the mapping $\tau : \mathcal{X} \rightarrow \Sigma^\tau$. In the second stage, the symbolic abstraction is used for new images. The workflow of SymDNN is illustrated in Figure 1.

4.1. Symbolic Abstraction Design

The proposed method for symbolic abstraction design works on a given training set, $\mathcal{D}$, of images. The steps are as follows:

1. We choose a patch dimension, $P$ (a parameter of our algorithm). For the chosen, $P$, we extract patches of dimension $P \times P$ pixels from all the images in $\mathcal{D}$ and populate them into a patch dataset, $\mathcal{D}_{patch}$.

2. We use a similarity based clustering algorithm to partition $\mathcal{D}_{patch}$ into disjoint clusters $\{C_1, C_2, \ldots, C_M\}$,
such that $D_{\text{patch}} = \bigcup_{i=1}^{M} C_i$. We associate a symbol, $\alpha_i$, with each $C_i$. We define the alphabet $\Sigma = \{\alpha_1, \ldots, \alpha_M\}$.

3. We identify the cluster centroid, $\mu_i$, for each partition, $C_i$. Note that $\mu_i$ is a $P \times P$ image patch. Let $\mu = \{\mu_1, \ldots, \mu_M\}$ be the set of cluster centroids.

4. We prepare a Codebook containing two mappings, namely:

   - **Patch to symbol.** $\eta : D_{\text{patch}} \rightarrow \Sigma$, such that for each patch $p \in D_{\text{patch}}$, we have $\eta(p) = \alpha_i$ iff $p \in C_i$.

   - **Symbol to patch.** $\eta_R : \Sigma \rightarrow \mu$, such that for each symbol, $\alpha_i \in \Sigma$, we have $\eta_R(\alpha_i) = \mu_i$.

   It may be noted that $\eta_R \eta(p)$ maps the patch $p$ to the centroid of the cluster containing $p$.

4.1.1 Clustering the Patches

Lloyd’s algorithm, also known as, $k$-means clustering can be used to achieve the above partitioning. The basic steps of the above algorithm are as follows:

1. For each cluster, $C_j$ the initial cluster centroid, $\mu_j$ is selected. Selected centroids generate the initial partitioning $C^0$. It may be noted that $\mu_j$ is a $P \times P$ image patch.

2. The Euclidean distance of each patch is calculated from all the centroids, that is, $||p - \mu_j||^2$.

3. Based on the distance computed, the cluster assignment for patch $p_i$ is obtained by finding the nearest centroid, that is, $\arg \min_{\mu_1, \mu_2, \ldots, \mu_M} ||\mu_j - p_i||^2$.

4. For each partition, a new centroid is obtained by computing the average of all the patches assigned to that partition, that is, $\mu_j = \frac{1}{|\mu_j|} \sum_{p_i \in \mu_j} p_i$. This generates the updated set of partitions $C^1$.

5. Step 2 to 4 are repeated until a fixed point is reached, that is, $C^m = C^{m+1}$.

$$\min_{(C_1 \cup C_2 \cup \ldots \cup C_M = C)} \sum_{j=1}^{M} \sum_{p_i \in \mu_j} \left( p_i - \frac{1}{|\mu_j|} \sum_{p_i \in \mu_j} p_i \right)^2$$ (1)

The $k$-means objective, stated in Eq. (1) is NP-hard. The iterative approach (Lloyd’s algorithm), with suitable initial-
ization runs in polynomial time to converge to a local minima. However, the size of the input space which is in the range of billions, makes the algorithm computationally expensive. Specifically, step 2 of the above iterative algorithm has a time complexity of $O(MNP^2)$.

The FAISS [29] $k$-means implementation accelerates this step by a large factor, which helps us to experiment with different numbers of clusters, patch sizes and handle around 38.5 million patches for the CIFAR-10 dataset and 5.6 billion for ImageNet.

4.2. Inferencing using Symbolic Abstraction

We consider two use cases for inferencing using the proposed symbolic abstraction. In the first case, we study the gain in adversarial robustness by virtue of symbolic abstraction, and reconstruction. In the second case, we study the compaction of the image by virtue of symbolic abstraction, which may be useful for communication and/or storage, and study the loss of inferencing accuracy post reconstruction. This section outlines these flows.

4.2.1 Resistance Against Adversarial Attacks

SymDNN follows the same defense model as other input processing based defenses: given a pre-trained classifier $N_\theta(\cdot)$, the preprocessor $\tau_R(\cdot)$ is almost always inference preserving, that is, $N_\theta(x) \approx N_\theta(\tau_R(x))$, and the symbolic reconstruction removes the adversarial perturbation.

The seminal work on gradient obfuscation [4, 58] and gradient masking [43] have highlighted that in a complete white-box setting, these types of non-differentiable defenses cannot be backpropagated through to generate adversarial examples. In [4], BPDA attack creates a differentiable approximation of these transformation functions and uses that to backpropagate and generate effective adversarial examples.

SymDNN, being a non-differentiable defense, also exhibits the same behavior. However, the success of BPDA or other adaptive attacks depend on how easily and precisely the transformation function can be approximated. Compared to the transformations used in earlier input processing based defenses [10, 27, 67], which are based on some analytical formulation, SymDNN’s approximation function $\tau_R(\cdot)$ is very difficult to approximate. It is learned from a large image dataset, and the computational overhead of building this transformation function is prohibitive even using a fast indexing and clustering library.

In [4], defenses that employ randomized transformations to the input are attacked using Expectation over Transformation (EOT) [5]. Samples of transforms used by a defense is used by EOT to iteratively approximate an expected transform for generating adversarial examples. SymDNN also uses a test time randomness, where instead of replacing patch $p_{ij}$ in image $x$ with $\eta(p_{ij})$, we replace it with a randomly chosen symbol from among the $k$ nearest centroids. We name this mechanism MSR: Multi-Sym Randomized inference, and evaluate the Top-1 and Top-5 accuracy of our models based on it. In this paper we use 25 most similar symbols ($k = 25$) for a given patch from the $k$-nearest neighbor graph of centroid patches to achieve the best clean vs. robust accuracy trade-off. To optimize over the $k$-nearest centroid patches for a given centroid patch, and generate an expected transform is as computationally demanding as learning the $k$-NN graph itself. This makes the test time randomness of SymDNN difficult to break by an adversary using EOT attack variants, within practically feasible computation capability and time.

Thus, the key to withstand the adaptive attacks in [4,58], is to design the transformation as hard as possible, when proposing an input processing based defense. For deploying in embedded systems, such transformation needs to be computationally efficient as well. Similar results, where BPDA is less effective can be observed in cases where the transform is difficult to approximate [45, 50]. In [45] a series of transforms are used, and in [50] a projection into a Generative Adversarial Network manifold is performed to achieve the robustness. In contrast, SymDNN executes a single, computationally efficient transform to achieve similar robustness.

SymDNN’s robustness is not only limited to complete white-box setting, against gradient based attacks. The symbolic reconstruction process replaces a bunch of pixels of patch resolution, with a clean centroid patch from the clustering model. This operation reinstates the values of most of the pixels as shown in Fig. 2a. Although it throws off some of the values randomly, our experiments show that the inference accuracy is preserved in most cases. As shown in Fig. 2b, this reduction of change per pixel happens at different noise levels, although it is most effective at lower attack strengths.

4.2.2 Compaction and Reconstruction

We define the following mappings for compaction and reconstruction of an image:

- **Symbolic Abstraction.** The given image $x \in \mathbb{R}^{I_h \times I_w \times C}$ is partitioned into patches of size $P \times P$ pixels (with suitable zero padding if the dimensions are not multiples of $P$), and arranged into a sequence $\pi$ of patches, $p_{i1}, \ldots, p_{in}$. We define $\tau(x) = s$, where $s \in \Sigma^*$ is the string $\alpha_{i1}, \ldots, \alpha_{in}$, such that $\eta(p_{ij}) = \alpha_{ij}$. We shall refer to $s$ as the code for $x$.

- **Symbolic Reconstruction.** Given the code, $s$, for the image $x$, we define $\tau_R(s) = x'$, where the image $x'$ is obtained by replacing each $\alpha_i \in s$ by $\eta_R(\alpha_i)$ and rearranging the patches in the same order as in the original image, $x$. In other words, in the reconstructed image, each patch
In $x$, $p_{ij}$ is replaced by $\eta[p_{ij}]$, namely the centroid of the cluster that contains $p_{ij}$.

The reduction due to abstraction, $\Delta^a$, can be expressed as follows:

$$\Delta^a = \left(1 - \frac{\log M}{P^2 \times B}\right) \times 100\%$$

(2)

where $B$ denotes the bits per pixel representation of an image and $M$ is the number of clusters. For storage/communication entropy encoding e.g., Huffman encoding can be used for reducing the amortized number of bits per pixel.

5. Experimental Evaluation

5.1. Experimental Setup


Adversarial Robustness. We choose Torchattacks [30] for attack implementations. For demonstrating efficacy of SymDNN on known robust models, we use certified CIFAR-10 models from RobustBench [17]. We use FoolBox [46] to perform adversarial attacks on ImageNet pre-trained models. We specify the threat models, attack iterations and attack strengths corresponding to each result in the result section.

5.2. Adversarial Robustness

Tab. 1 shows the robust accuracy gains of SymDNN under all possible attack models described in Sec. 1 and Sec. 4.2.1, compared to the standard inference. SymDNN performs best under the white-box model, black-box defense attack model (columns 3-4 of Tab. 1), where the adversary has no knowledge about the defense being used.

The accuracy gain in the complete white-box setting (last column of Tab. 1) is possibly due to obfuscated gradients [4, 58], except for the BPDA attack (top 3 rows of Tab. 1). However SymDNN’s transform is difficult to approximate, and hence it does not break completely against BPDA. SymDNN is only effective to complete black-box attacks under lower attack strengths. This is currently a limitation, although this is the best we have in an embedded resource limited setting. We also hypothesize that a more strongly separable clustering may enable SymDNN to resist attacks with larger values of $\epsilon$. We discuss this aspect further in Sec. 5.4.

Tab. 2 presents a comparison of SymDNN with NRP [42] and DefenseGAN [50], two state-of-the-art input defenses. SymDNN provides a better defense than NRP in 12 out of the 15 attacks. Specifically, NRP’s defense fails completely against a recent attack (Jitter [52]). DefenseGAN [50] performance is highly dependent on the generator training. We trained a generator with $10^k$ iterations, which did not perform well against the wide range of attacks we used.

NRP uses a DNN of 16.5 million parameters for the purification step. DefenseGAN uses a resource intensive Generative Adversarial Network generator to generate cleaned images. In the case of the adversarially trained model [47] (last column of Tab. 2), the model size is 705Mb. Compared to these methods, SymDNN is suitable for resource-limited embedded systems. It has very low computation overhead. SymDNN, with $2 \times 2$ patches, takes 0.5 milliseconds to lookup clusters & encode/decode images, with a peak memory load of 44 Mb.

For MNIST, we observe a minimum of 93% and 68% robust SymDNN accuracy at ($\epsilon = 16/255$) and ($\epsilon = 32/255$) respectively, for the 7 attack we tried.

For ImageNet, preliminary experiments using FoolBox [46] show approximately 60% boost in robust accuracy for C&W [12], PGD [38], and FGSM [26]. Detailed results and visualizations for MNIST & ImageNet are presented in the supplementary material.

\(^1\)https://pytorch.org/vision/stable/models.html
Table 1. SymDNN accuracy(%) under different attacks: Abbreviations used: “M” - Model, “D” - Defense, “W” - White-box, and “B” - Black-box. The attack models are expressed as combinations of these. SymDNN performs best under the white-box model / black-box defense combination. Different row colors denote different attack strengths (col 2). * indicates SymDNN accuracy with 2048 clusters. The case of gradient obfuscation [4, 58] happens under the complete white-box attack model(last column).

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>BPDA [4]</td>
<td>$\epsilon = 8/255$</td>
<td>8</td>
<td>48</td>
<td>32</td>
</tr>
<tr>
<td>TI-FGSM [23]</td>
<td>$\epsilon = 8/255$</td>
<td>11</td>
<td>28</td>
<td>83</td>
</tr>
<tr>
<td>AutoAttack [20]</td>
<td>$\epsilon = 8/255$</td>
<td>0</td>
<td>10</td>
<td>32</td>
</tr>
<tr>
<td>DI-FGSM [65]</td>
<td>$\epsilon = 8/255$</td>
<td>1</td>
<td>0</td>
<td>32</td>
</tr>
<tr>
<td>MI-FGSM [65]</td>
<td>$\epsilon = 8/255$</td>
<td>12</td>
<td>36</td>
<td>55</td>
</tr>
<tr>
<td>RFGSM [59]</td>
<td>$\epsilon = 8/255$</td>
<td>10</td>
<td>33</td>
<td>55</td>
</tr>
<tr>
<td>EOTPGD [71]</td>
<td>$\epsilon = 8/255$</td>
<td>10</td>
<td>35</td>
<td>57</td>
</tr>
<tr>
<td>APGDN(CE) [20]</td>
<td>$\epsilon = 8/255$</td>
<td>1</td>
<td>15</td>
<td>32</td>
</tr>
<tr>
<td>APGD(DLR) [20]</td>
<td>$\epsilon = 8/255$</td>
<td>34</td>
<td>64</td>
<td>75</td>
</tr>
<tr>
<td>APGD [20]</td>
<td>$\epsilon = 8/255$</td>
<td>33</td>
<td>63</td>
<td>75</td>
</tr>
<tr>
<td>Jitter [59]</td>
<td>$\epsilon = 8/255$</td>
<td>30</td>
<td>59</td>
<td>71</td>
</tr>
<tr>
<td>BPDA [4]</td>
<td>$\epsilon = 4/255$</td>
<td>8</td>
<td>49</td>
<td>61</td>
</tr>
<tr>
<td>TI-FGSM [23]</td>
<td>$\epsilon = 4/255$</td>
<td>50</td>
<td>58</td>
<td>72</td>
</tr>
<tr>
<td>AutoAttack [20]</td>
<td>$\epsilon = 4/255$</td>
<td>0</td>
<td>37</td>
<td>8</td>
</tr>
<tr>
<td>DI-FGSM [65]</td>
<td>$\epsilon = 4/255$</td>
<td>0</td>
<td>39</td>
<td>11</td>
</tr>
<tr>
<td>MI-FGSM [65]</td>
<td>$\epsilon = 4/255$</td>
<td>38</td>
<td>5</td>
<td>18</td>
</tr>
<tr>
<td>RFGSM [59]</td>
<td>$\epsilon = 4/255$</td>
<td>40</td>
<td>4</td>
<td>18</td>
</tr>
<tr>
<td>EOTPGD [71]</td>
<td>$\epsilon = 4/255$</td>
<td>41</td>
<td>5</td>
<td>16</td>
</tr>
<tr>
<td>APGDN(CE) [20]</td>
<td>$\epsilon = 4/255$</td>
<td>1</td>
<td>51</td>
<td>10</td>
</tr>
<tr>
<td>APGD(DLR) [20]</td>
<td>$\epsilon = 4/255$</td>
<td>34</td>
<td>35</td>
<td>52</td>
</tr>
<tr>
<td>APGD [20]</td>
<td>$\epsilon = 4/255$</td>
<td>33</td>
<td>30</td>
<td>51</td>
</tr>
<tr>
<td>Jitter [59]</td>
<td>$\epsilon = 4/255$</td>
<td>30</td>
<td>29</td>
<td>47</td>
</tr>
</tbody>
</table>

5.3. Compaction and Clean Symbolic Accuracy

As summarized in Tab. 3, our proposed SymDNN has less than 0.4% accuracy drop in Top-1 and Top-5 accuracy metrics, for different pre-trained models on 50,000 ImageNet testset. The Top-5 metric, popularized in ILSVRC [49], can be useful for building ensemble models [64].

The data reduction can be calculated directly from Eq. (2). For instance, with a patch size of $2 \times 2$, considering 8 bits per pixel representation and 2048 symbols, SymDNN achieves around 68% compaction on the ImageNet test set. On the other hand, using 512 symbols brings down the compaction to 74%, with 1% drop in accuracy. Using more than 2048 symbols does not seem to be profitable, as it increases the computation load for cluster index training and patch extraction, with negligible benefits in terms of accuracy.

5.4. Discussion

We observe that the clustering model for ImageNet can be used to obtain 88.73% and 99.07% classification accuracy on CIFAR-10 & MNIST test sets, respectively. These values are comparable to the inference accuracy with their respective clustering models. This shows that a pre-trained clustering model, learned from a large image dataset, can be fairly generalized and useful for symbolic inference on other datasets.

The adversarial robustness of SymDNN depends on the
and the symbolic image after the attack.

The distance between the reconstructed clean symbolic image and the attacked image, and (B) the normalized Levenshtein (edit) distance between clean and attacked symbolic image, for varying $c$. We observe that the mean value of the normalized edit distance remains significantly lower than the $L_2$ norm, till the $c$ value reaches 1. The edit distance remains low as small changes in the image do not result in significant change to the cluster assignments. We believe that the effective adversarial examples that C & W attack generates, are thwarted by this clustering robustness, resulting in a boost in robust symbolic accuracy. However, the performance degrades as the value of $c$ reaches 1 and beyond, i.e., when the perturbations are larger, the symbol map corresponding to the attacked image changes significantly. It may be noted that this is a defense black-box experiment and hence the gradient obfuscation does not happen here.

It is also evident from Fig. 3 that the edit distance curve is more aligned to the step-wise increment of the $c$ value, compared to the $L_2$ distance. When using a symbolic inference scheme, we believe that the edit distances between clean and attacked symbolic images have the potential to reveal further insights about adversarial attacks.

6. Conclusions

In this paper we have presented the algorithmic basis and comprehensive evaluation of SymDNN, a scheme for abstraction & reconstruction of an input image before DNN inference. We highlight that the DNNs deployed on embedded systems lack defense mechanisms against a wide variety of adversarial attacks. Adversarial research community strives for defenses that are robust against any adversarial attacks, in a complete white-box setting. There are only a small number of methods that achieve that aim. However those methods are not suitable for resource-limited embedded systems. Thus embedded vision tasks are unprotected against any adversarial attacks. We show that SymDNN has capability to undo adversarial perturbations for a wide range of recent attacks, under black-box and white-box attack models, when the defense remains a black-box to the adversary. Under a complete white-box one attack model, we show that the key to resist the recent strong adaptive attacks, is designing transformation functions that are as hard as possible. Our proposed SymDNN employs such an input transformation, which is computationally efficient, and results in image compression.

Along with these concrete benefits, we also report several interesting aspects of SymDNN inference, e.g. the fundamental visual symbols learned by the ImageNet codebook, the potential of Levenshtein distance as a measure of dissimilarity between clean and attacked images, etc. We believe that this paper will encourage other researchers to study the larger potential of discretized and abstracted images in computer vision. Our future work will be to strengthen the confluence between symbolic abstraction and DNN inference.
References


[34] Sungyoon Lee and Jaewook Lee. Defensive denoising methods against adversarial attack, 2018.


[54] Yash Sharma and Pin-Yu Chen. Attacking the madry defense model with $l_1$-based adversarial examples, 2018. 2
[65] Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan Yuille. Improving transferability of adversarial examples with input diversity, 2019. 7