ECAI 2023
K. Gal et al. (Eds.)

doi:10.3233/FAIA230525

SPAT: Semantic-Preserving Adversarial Transformation for Perceptually Similar Adversarial Examples

Subrat Kumar Swain[a], **Vireshwar Kumar**[b], **Dan Dongseong Kim**[c] **and Guangdong Bai**[c]

[a]University of Queensland-IIT Delhi Academy of Research (UQIDAR)
[b]Indian Institute of Technology Delhi
[c]The University of Queensland

Abstract. Although machine learning models achieve high classification accuracy against benign examples, they are vulnerable to adversarial machine learning (AML) attacks which generate adversarial examples by adding well-crafted perturbations to the benign examples. The perturbations can be increased to enhance the attack success rate, however, if the perturbations are added without considering the semantic or perceptual similarity between the benign and adversarial examples, the attack can be easily perceived/detected. As such, there exists a trade-off between the attack success rate and the perceptual similarity. In this paper, we propose a novel Semantic-Preserving Adversarial Transformation (SPAT) framework which facilitates an advantageous trade-off between the two metrics. SPAT modifies the optimisation objective of an AML attack to include the goal of increasing the attack success rate as well as the goal of maintaining the perceptual similarity between benign and adversarial examples. Our experiments on a variety of datasets including CIFAR-10, GTSRB, and MNIST demonstrate that SPAT-transformed AML attacks achieve better perceptual similarity while maintaining the attack success rates as the conventional AML attacks.

1 Introduction

In recent years, machine learning (ML) models have surpassed traditional algorithms and have been successfully used in various applications such as image classification [11, 47], object detection [5], and natural language processing [2, 41]. However, despite their remarkable performance, such ML models are highly vulnerable to small and imperceptible noise that are called *adversarial perturbation*. Inputs generated by these perturbations called *adversarial examples*, can significantly degrade the performance of an ML model. As ML models are used in security-sensitive systems, these adversarial examples pose a significant security risk and raise concerns about the integrity of these systems.

An adversary generates these adversarial examples by solving an optimisation problem to maximise the classifier's loss while keeping the adversarial perturbation amount minimum. Existing adversarial attack algorithms for classification generally consider simple L_p threat models. In an adversarial threat model, a predefined set of perturbations is established to manipulate the input and generate an adversarial example. Many existing attack and defence strategies use bounded threat models that restrict the adversarial examples by L_2 or L_{inf} distance [15,27]. Norm-constrained adversarial perturbations in the input space uncovered some intriguing properties of neural networks and also gave us insights into the generalisation power of the neural networks.

However, metrics such as the L_2 norm distance are insufficient to evaluate structured data, such as images, as they assume pixel-wise independence. Moreover, adding adversarial perturbation directly to the input data sometimes renders the input semantically unacceptable. Perturbing a face image could change the structural properties of the face, which might not look like an actual face. Hence, the challenge in an adversarial noise-based attack is twofold: determining *how much* adversarial perturbation to add and *where* to add it.

Adding adversarial perturbation in an appropriate place will not only help preserve the semantics of the input but also remove the bound on the perturbation amount. For instance, changing the colour of a person's hair from black to red will be semantically meaningful, and the noise amount will be much higher than the L_p bound. Furthermore, the adversary may require larger perturbations to achieve higher adversarial strength, leading to more noticeable changes in the original image. This trade-off between adversarial effectiveness and semantic correctness will always exist if p-norm bounds are applied. Also, recent work shows that defences trained on L_p bounded perturbation are not robust against new types of unseen attacks [21]. Therefore, exploring diverse adversarial examples, especially those with "unrestricted" magnitudes of perturbation, has attracted much attention in academia and industry [3].

To address this issue, some studies employ threat models which allow spatial perturbations [12,44,45] and recolouring [1,25] of an image. Hosseini *et al.* [18] manipulated the hue and saturation of an image to create adversarial perturbations. Even though they are able to generate natural-looking adversarial images, changes such as colour, saturation, and textures are easily perceptible to humans. A successful adversarial attack without constraints should have the ability to cause significant harm while remaining imperceptible and semantically correct. However, the task of measuring semantic correctness remains a difficult challenge, as visual patterns are complex and subjective measures of similarity often rely on human perception.

One way to model semantic correctness is to mathematically model human perception. An adversarial example is said to be *semantically correct* if it is perceptually similar to the original input, typically determined by how less the perceptual is distance between two inputs. To formalise, we can define true perceptual distance $s^*(x_1, x_2)$ between two inputs x_1 and x_2 to be how humans perceive two different inputs. For a given threshold, λ^*, if $s^*(x_1, x_2) \leq \lambda^*$, it is safe to say that inputs x_1 and x_2 are similar given that the value of

λ^* may vary from domain to domain and input to input. True perceptual distance is a mathematical formalisation of human perception. We assume that, when presented with two images, we implicitly try to make a judgement by deciding on a threshold. Since it is implicit, it is not trivial to approximate the true perceptual distance $s^*(.,.)$.

However, some works in the literature have tried to approximate it using the L_2 distance and the Peak Signal-to-Noise Ratio (PSNR) [20]. PSNR is a metric used to quantify the quality of a reconstructed or compressed image by measuring the ratio of the maximum possible power of the original signal to the power of the noise introduced during compression or reconstruction. However, these measures do not always align with human vision when it comes to changes in images like blurring and spatial transformations. This motivated other metrics, such as SSIM [43], GMSD [46], FSIM [48], and VIF [35] to model it. Recently, it has been established that the internal activation layers of a convolutional neural network approximate the true perceptual distance, Learned Perceptual Image Patch Similarity (LPIPS) [49], that closely models human perception.

Hence, the final goal of the adversary is to generate an adversarial example such that it should (1) evade the classifier, and (2) be perceptually similar to the original input, i.e. LPIPS distance between the original input and the generated adversarial example should be minimum. In this work, we address both of the adversary's goals by generating adversarial examples that having high attack success rates along with low LPIPS scores depicting high perceptual similarity and semantic correctness. To achieve this, we model the data manifold using autoencoders, and then instead of perturbing the input space of the data, we add perturbation to the manifold space of the input data. Since the underlying manifold of a dataset is a set of all the meaningful data, adding a small perturbation yields a semantically meaningful adversarial example. Our contributions include:

- We propose a unified algorithmic framework, SPAT, to transform any conventional AML attack to its semantic-preserving version while maintaining a comparative attack success rate. SPAT modifies the objective function of the conventional AML attack to maintain perceptual similarity between the adversarial and benign examples. We have published our code for reproducibility and further study [38].
- We propose a constrained loss function for autoencoder that uses class information to learn a latent representation that avoids the problem of class invariance. While perturbing the latent space of the autoencoder, data points often leave the corresponding class manifold to other class manifolds rendering invalid adversarial examples. The proposed loss function concentrates inputs of a particular class on a few latent units of the autoencoder. Perturbing those units helps restrict the adversarial examples to stay in the same class.
- We test the generated adversarial examples against state-of-the-art robust models from the existing literature. We found that adversarial examples generated using SPAT method lower the state-of-the-art robust accuracy of the classifiers making them stealthier than prior adversarial attacks.

2 Related Work

The study of AML saw mushroom growth after Szegedy et al. [39] found adversarial examples for image classification models. After observation of this phenomenon on deep neural networks, an arms race started between researchers to formulate new ways for crafting adversarial examples [6, 15, 28, 30]. Most of these attacks have focused on digital inputs, such as images and text. With the growing deployment of DNNs in the physical world, some researchers have also demonstrated the feasibility of adversarial attacks in the physical world [24].

To make the attack more imperceptible and stealthy, some researchers used elements such as patches [4] and stickers [13]. However, even though patches and stickers were not a matter of concern initially, recent studies have motivated researchers to look for spurious patches and stickers in the physical world making them less effective. Some works also tried to modify the colour and texture of the input sample to craft adversarial examples [18, 34, 45]. Qiu et al. [31] tried to leverage attribute-conditioned image editing to craft indistinguishable adversarial examples. Even though they are able to generate natural-looking adversarial images, changes such as colour, saturation, and textures are easily perceptible to humans.

Some studies explored on-manifold adversarial examples for crafting natural-looking adversarial examples. As the real-world data lie on a near-low dimensional unknown manifold, it was assumed that adversarial examples usually leave the underlying data manifold [14, 40]. This assumption was broken when Gilmer et al. [14] found on-manifold adversarial examples on a toy dataset. This was later validated for real datasets by [3,36,50]. Other works [10,37,50] used autoencoder-based models to create on-manifold adversarial examples. The study in [10] used a Stein-variational autoencoder to learn adversarial examples while the study in [50] used a Generative Adversarial Network (GAN) to learn the manifold and perturbed the inputs on the manifold. Recently, the study in [37] used an ensemble of autoencoders to learn the semantic space of the inputs. Like the above methods, we utilise a vanilla autoencoder to approximate the underlying data manifold. However, unlike the above works, we give a generalised algorithmic framework that utilises existing adversarial attack methods to search for adversarial examples in the approximated manifold.

One of the primary issues in crafting on-manifold adversarial examples is the problem of class invariance which was not discussed in [50] and [10]. The authors in [37] acknowledged this problem and proposed a solution by using different autoencoders to learn different class manifolds. This method does not scale well with a higher number of classes. In contrast, we propose *class-constrained autoencoders* that add an additional penalty to the vanilla-autoencoder loss function to make the distribution of the data in the latent space far away from each other. Even though on-manifold adversarial examples have been found on real-world datasets, the scope and limitations of this method still remain unexplored.

3 Proposed Approach

Notations. Let $\mathcal{D} = \{x_i, y_i\}_{i=1}^m$ is a set of m samples, where any $x \in \mathcal{D}$ is sampled from $\mathcal{X} \subset R^n$ with its associated label $y \in \mathcal{D}$ is sampled from $\mathcal{Y}$ and $\mathcal{Y}$ is a discrete set of labels of size $k, y \in \{1, 2, \ldots, k\}$. Let $f_c(.)$ be the classifier with parameter θ that maps $f_c : \mathcal{X} \mapsto \mathcal{Y}$. It is safe to assume that the classifier is well-trained to correctly classify an input sample x as its corresponding label y with state-of-the-art accuracy. Let $f_e(.)$ denote the encoder, with parameter ϕ, that maps the input space to the latent space, i.e. $f_e : \mathcal{X} \mapsto \mathcal{Z}$, and for $x \in \mathcal{X}$, $f_e(x) = z$ where $z \in \mathcal{Z}$ is of dimension d and $d \ll N$. Further, $f_d(.)$ denote the decoder, with parameter ψ that maps the latent space to the output space by reconstructing the input, i.e., $f_d : \mathcal{Z} \mapsto \mathcal{X}$, and $x' = f_d(z)$, where x' is the reconstructed output. The encoder $f_e(.)$ and the decoder $f_d(.)$ together constitute the autoencoder $f_a(.)$ with parameter ω, $f_a(.) = f_e(.) \odot f_d(.)$ and $\omega = \{\phi, \psi\}$. The output from the en-

coder $f_e(.)$ is passed along as an input to the decoder $f_d(.)$. For notation simplicity, parameters $\theta, \phi, \psi, \omega$ of the corresponding models f_c, f_e, f_d and f_a will be omitted further in this paper.

3.1 Problem Formulation

The method of generating adversarial examples can be broadly classified into two major categories: *Maximum Allowable Attack* and *Minimum Norm Attack*. The maximum allowable attack solves an optimisation problem that aims to maximise the loss of the classifier against the correct class such that the perturbed input gets misclassified and the adversarial perturbation lies within an ϵ bound, i.e.,

$$\begin{aligned} \underset{\delta}{\text{argmax}} \quad & \mathcal{L}(f_c(x+\delta), y) \\ \text{s.t.} \quad & \|\delta\|_p \leq \epsilon. \end{aligned} \tag{1}$$

The minimum norm attack solves an equivalent optimisation problem that aims to find the minimum adversarial perturbation that gets the input misclassified, i.e.,

$$\begin{aligned} \underset{\delta}{\text{argmin}} \quad & \|(x+\delta) - x\|_p \\ \text{s.t.} \quad & f_c(x+\delta) \neq y. \end{aligned} \tag{2}$$

Unlike Equation (1), Equation (2) seeks to optimise adversarial perturbation instead of constraining it to a predefined limit. In this work, we will transform both formulations to their semantic variant which will be responsible for generating perceptually similar adversarial examples.

3.2 SPAT Attack

For a sample $x \in \mathcal{X}$ with corresponding class label $y \in \mathcal{Y}$, the goal is to craft an adversarial example x^{adv} such that it evades the classifier f_c and it is perceptually similar to x. Hence, the adversary's goal here is two-fold; *how much* adversarial perturbation to add and *where* to add it. We propose a unified framework SPAT that modifies a conventional AML algorithm to craft corresponding perceptually similar adversarial examples.

As we operate under a white-box setting, the adversary has the knowledge of training data distribution $\mathcal{D}$. Hence, we train the autoencoder f_a using the training data. Given the sample x, the trained autoencoder f_a can efficiently reconstruct the input x with minimum errors. The input x is first passed to the encoder f_e to get the latent code z, $z = f_e(x)$. Since the latent layer of an autoencoder models the semantic space of the inputs, to preserve the semantics of the crafted adversarial example x^{adv} similar to x, the adversarial perturbations are added in the latent space $\mathcal{Z}$ learned by the autoencoder. These adversarial perturbations yield adversarial latent codes z^{adv}, i.e., $z^{adv} = z + \delta_z$, where δ_z is the adversarial perturbation in the latent space. Then the perturbed z^{adv} is passed to the decoder f_d to generate reconstructed output $x' = h(z^{adv})$. This reconstructed output is the adversarial input to the classifier model f_c, i.e., $x^{adv} = x'$. The proposed attack framework for SPAT is illustrated in Figure 1.

Since the latent space of an autoencoder approximates the underlying data manifold, we add the noise to the underlying data manifold instead of adding directly to the input data. As a n-dimensional manifold is a topological space with each point possessing a neighborhood that is similar in structure to an open subset of n-dimensional Euclidean space, *i.e.* locally linear, the resultant latent vector z_{adv} after adding a small adversarial perturbation δ_z will also lie on the manifold. However, it is important to note that the success of this approach depends on the quality of the autoencoder and the adversarial example generation technique used.

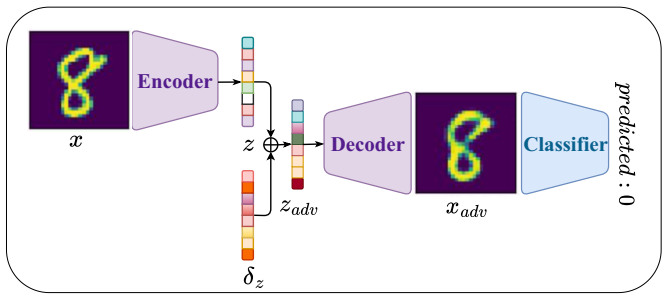

Figure 1: Framework for SPAT attacks.

This completes one of the two above-mentioned adversary's goals, *where* to add the adversarial perturbations. To accomplish the goal, *how much* adversarial perturbations to add, prior techniques optimise the function mentioned in Equations (1) and (2) to obtain the desired adversarial perturbations. In this framework, pre-formulated optimisation functions are extended to obtain adversarial perturbations for the latent space. The modified objective for getting optimal δ_z, where δ_z is the adversarial perturbations for the latent space, is to: *"maximise the loss of the classifier against the correct class such that the reconstructed output from the perturbed latent code gets misclassified and the perturbation lies within an ϵ bound."*

Hence, the modified formulation of the objective function will be:

$$\begin{aligned} \underset{\delta_z}{\text{argmax}} \quad & \mathcal{L}(f_c(f_d(z+\delta_z)), y) \\ \text{s.t.} \quad & \|\delta_z\|_\infty \leq \epsilon. \end{aligned} \tag{3}$$

where z is the encoding obtained by passing x to the encoder, $z = f_e(x)$.

To solve the above-mentioned optimisation problem and obtain δ_z, all the prior methods are extended to obtain the optimal latent adversarial perturbations δ_z. Instead of calculating the adversarial perturbations δ with respect to the input x, we back-propagate it to the latent layer of the autoencoder, and we calculate the perturbations δ_z with respect to the latent layer using the prior techniques. Formally, we pass the input sample x to the encoder f_e to get the latent code, $z = f_e(x)$. Then we treat the decoder f_d and the classifier f_c as a single unified classifier $f_{dc} = f_d \odot f_c$ that maps $f_{dc} : \mathcal{Z} \mapsto \mathcal{Y}$. We run the adversarial attack algorithms by considering the target classifier as f_{dc} and inputs to perturb as z. On successful completion of the attack, it gives corresponding δ_z and z_{adv}. Then the input is reconstructed from the perturbed adversarial latent code z^{adv} to obtain a semantically preserved adversarial example. The detailed algorithm is mentioned in Algorithm 1.

3.3 Mitigating Class In-Variance

Even though adversarial examples generated using SPAT attacks are perceptually similar to the original image, they often suffer from the class in-variance problem. This problem makes an adversarial example invalid. By definition, an adversarial example is said to be invalid if after adding adversarial perturbations, the image is misclassified to some other class, but the image actually looks to be from that class to a human.

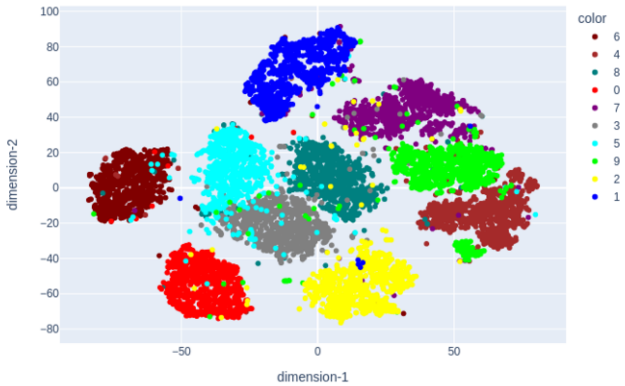

Figure 2: Latent space distribution of MNIST in vanilla autoencoder obtained using t-SNE.

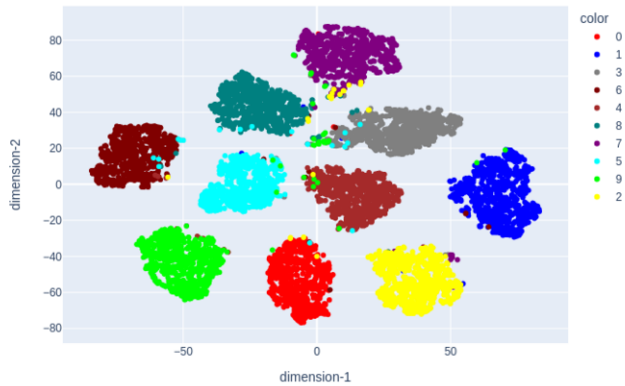

Figure 3: Latent space distribution of MNIST in class-constrained autoencoder obtained using t-SNE.

Algorithm 1 SPAT

Require: Data samples $\mathcal{D} = \{x_i, y_i\}_{i=1}^{m}$
Require: Classifier f_c
Require: Encoder and decoder f_e and f_d
1: Sample $\{x, y\}$ from the data samples $\mathcal{D}$
2: $z \leftarrow f_e(x)$
3: $f_{dc} = fd \odot f_c$
4: $\delta_z \leftarrow X(f_{dc}, z, y)$
5: $z^{adv} \leftarrow z + \delta_z$
6: $x' \leftarrow f_d(z^{adv})$
7: $x^{adv} \leftarrow x'$

7: **Return** x^{adv}

Three out of four prior works that explored adversarial examples in the semantic space did not report or address this issue. [37] reported this problem and addressed the same by using different autoencoders for learning different class-manifold corresponding to that class. By this method, each autoencoder will have the representation knowledge of only one class. As they do not model the manifold of any other class, perturbing the latent representation will not generate any invalid adversarial examples. However, the space complexity of this method for training class-specific autoencoder will increase linearly with the number of classes, $\mathcal{O}(N)$. This method will not be effective against classification task with high number of classes such as Imagenet [19].

To mitigate this problem, SPAT uses just one autoencoder and adds an additional constraint on the latent dimension to decrease the probability of crafting invalid adversarial examples. In a vanilla autoencoder, the encoder learns those features that can best describe the data and the original data can be reconstructed from them; i.e., it learns representative features. In the proposed autoencoder, called *class-constrained autoencoder*, the dimension of the latent space is made equal to the number of classes; $d = k$, where k is the number of classes. Along with the reconstruction loss on the generated output, a classification loss is also imposed on the latent layer of the autoencoder. Hence, for a data sample $\{x, y\}$, classifier f_c, encoder f_e, and decoder f_d,

$$z = f_e(x), \quad x' = f_d(z),$$

$$\mathcal{L} = \mathcal{L}_{cls}(z, y) + \mathcal{L}_{recon}(x, x').$$

Here, $\mathcal{L}_{recon}$ represents the traditional reconstruction loss computed between the original and the reconstructed image, and $\mathcal{L}_{cls}$ represents the classification loss computed on the latent code and the actual class. The intuition behind this constraint is that this will force the encoder to learn representative and predictive features to better classify the data. This will push the latent codes far from each other, prioritising clear decision boundaries between them. Hence, this will increase the gap between the data samples and thereby decrease the probability of moving to another class in the semantic space while applying perturbation. The difference in the distribution structure of the latent code obtained using t-SNE of vanilla and class-constrained autoencoder is shown in Fig. 2 and 3.

4 Evaluation

Here, we conduct an extensive evaluation of SPAT.

Table 1: Classifier specifications for MNIST (architecture and hyperparameters) and CIFAR-10 (hyperparameters).

(a) Architecture for the MNIST classifier

Layer Type	MNIST Model
Linear	784×512
ReLU	-
Linear	512×100
ReLU	-
Linear	100×10
Softmax	10

(b) Hyperparameters for MNIST, CIFAR-10, and GTSRB classifiers

Hyperparameters	MNIST	CIFAR-10	GTSRB
Optimizer	Adam	SGD	Adam
Learning Rate	0.0001	0.05	0.001
Momentum	-	0.9	-
Weight Decay	-	5e-4	-
Batch Size	64	64	64
Epochs	10	50	25

Figure 4: Qualitative comparison of conventional PGD and SPAT-PGD on CIFAR-10. The first row is original image, second row have adversarial images generated using PGD, third row is corresponding PGD noise, fourth row have adversarial images generated using SPAT-PGD, and the end row have SPAT-PGD noise.

4.1 Experimental Setup

To evaluate our attacks, we chose three popular datasets for conducting experiments: CIFAR10 [22], GTSRB [19], and MNIST [26]. Before performing the attacks, we trained three state-of-the-art classifiers on these datasets. For MNIST, we use a custom architecture and training paradigm. For CIFAR10 and GTSRB, we use a pre-defined (but not pre-trained) architecture. Since it is essential to evaluate proposed attacks on regular models as well as robust models (models trained with adversarial images), we evaluated SPAT attacks against five state-of-the-art robust models from the literature. For the evaluation, we used the open-source robustbench [9] library, and we did the evaluation of SPAT attacks only for CIFAR-10. For CIFAR-10, we evaluated SPAT attacks against Wang *et al.* [42], two models from Rebuffi *et al.* [32], Gowal *et al.* [16], and Sehwag *et al.* [33].

4.1.1 CIFAR10

CIFAR10 is a dataset of 32×32 coloured real-world images of 10 classes, with $50,000$ training images and $10,000$ test images. Since it is a dataset of real-world images, it is relatively more complex than the other two datasets. For training CIFAR10, we used the architecture of Resnet-18 [17]. We used momentum-based SGD along with the hyper-parameters mentioned in Table 1b. With this, we achieved a test accuracy of 88.26%.

4.1.2 GTSRB

GTSRB (German Traffic Sign Recognition Benchmark) is a dataset of more than $50,000$ images of traffic signs, with 43 classes, used for benchmarking traffic sign recognition algorithms. We used Resnet-34 [17] architecture without any pre-trained weights for training the GTSRB from scratch. We used Adam as an optimiser along with the hyper-parameters mentioned in the Table 1b. We achieved an accuracy of 96.35% on the test data.

4.1.3 MNIST

MNIST is a dataset of 28×28 grayscale images of handwritten digits, with $60,000$ training images and $10,000$ test images. We give the model architecture for MNIST in Table 1a and hyper-parameters used for training in Table 1b. After training, we achieved a state-of-the-art accuracy of 99.5%.

4.1.4 Autoencoders

Along with classifiers, we also trained corresponding autoencoders for all three datasets. For CIFAR-10, we used a Resnet-18 autoencoder with a latent dimension of 256. For training, we used Adam optimiser with a learning rate of 0.001. Since the CIFAR-10 dataset is a collection of real-world images, its complexity makes it harder for the autoencoder to reconstruct the input images. Hence, for CIFAR-10, we trained the autoencoder differently than the way vanilla autoencoders are usually trained. To boost the exact reconstruction capability of the CIFAR-10 autoencoder, we add an additional loss along with the reconstruction loss of the vanilla autoencoder. Given $\mathcal{L}_{recon}(x, x')$ is the reconstruction loss between the input x and the reconstructed output x' and $\mathcal{L}_{cls}(x', y)$ is the classification loss on the reconstructed output x', the resultant loss is mentioned as follows.

$$\mathcal{L}_{ae} = \mathcal{L}_{recon}(x, x') + \mathcal{L}_{cls}(x', y). \tag{4}$$

For GTSRB, we used a custom CNN autoencoder with no additional loss. We used the same hyper-parameters as we used for training the CIFAR-10 autoencoder. The model architecture is defined in the Table 2b.

Finally, we used a class-constrained ANN autoencoder with a 100-dimensional latent layer for MNIST using the architecture in Table 2a. We trained it for 20 epochs using Adam as the optimiser with a learning rate of 0.001.

4.1.5 Attacks

We took five different yet fundamental AML attacks for transforming them into SPAT attacks. Those conventional AML attacks are the following: Fast Gradient Sign Method (FGSM) [15], Projected Gradient Descent (PGD) [27], Carlini & Wagner (C&W) [6], DeepFool (DF) [28], and Elastic Net Adversarial Attack on DNNs (EAD) [7] attacks. We applied SPAT and transformed these conventional AML attacks into the following SPAT attacks: SPAT-FGSM, SPAT-PGD, SPAT-C&W, SPAT-DeepFool, and SPAT-EAD. We used the IBM art [29] library to carry out all our attacks. For evaluation, we used 1000 images from each dataset to carry all the conventional AML attacks and corresponding SPAT attacks.

Table 2: Autencoder architecture for MNIST and GTSRB.

(a) Architecture for the MNIST autoencoder

Encoder		Decoder	
Layer Type	**Architecture**	**Layer Type**	**Architecture**
Linear	784×512	Linear	100×264
ReLU	-	ReLU	-
Linear	512×264	Linear	264×512
ReLU	-	ReLU	-
Linear	264×100	Linear	512×784
-	-	Sigmoid	-

(b) Architecture for the GTSRB autoencoder

Encoder		Decoder	
Layer Type	**Architecture**	**Layer Type**	**Architecture**
Conv2d	$3 \times 16 \times 3 \times 3$	ConvTranspose2d	$32 \times 32 \times 3 \times 3$
ReLU	-	ReLU	-
Conv2d	$16 \times 32 \times 3 \times 3$	ConvTranspose2d	$32 \times 16 \times 3 \times 3$
ReLU	-	ReLU	-
Conv2d	$32 \times 32 \times 3 \times 3$	ConvTranspose2d	$16 \times 3 \times 3 \times 3$
ReLU	-	Tanh	-

4.1.6 Metrics

To evaluate the proposed SPAT attacks, we used two different metrics. To examine the evasiveness of SPAT attacks, we used **Attack Success Rate** which measures the percentage of adversarial examples that the model misclassifies when subjected to an adversarial attack. A higher attack success rate indicates a more successful adversarial attack. To measure the perceptual similarity of generated adversarial images from SPAT attacks, we used **LPIPS Score** [49]. We used this to measure the perceptual similarity between the generated adversarial images of SPAT attacks and the original images. The LPIPS score value ranges from 0 to 1; with 0 being both the images are identical and 1 being both images are completely distinct. As an underlying model architecture for LPIPS score, we used AlexNet [23]. An ideal attack would have a higher attack success rate and a lower LPIPS score.

4.2 Results

We evaluated SPAT attacks against three classifiers trained for CIFAR-10, GTSRB, and MNIST. To summarise our results, we created attack success rate vs. LPIPS score trade-off plots for all three datasets. We put the LPIPS score on the X-axis and the attack success rate on the Y-axis. Since the goal is to generate adversarial examples

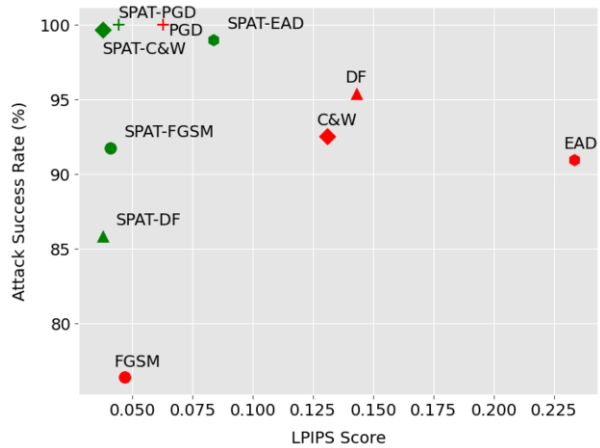

Figure 5: Trade-off between Attack Success Rate and LPIPS Score of conventional AML attacks and SPAT attacks for CIFAR-10.

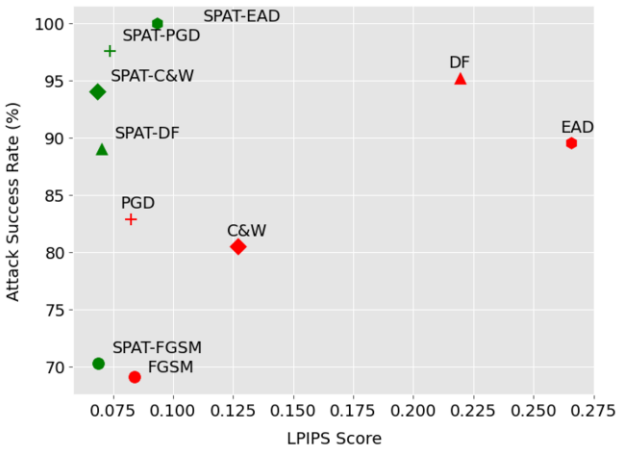

Figure 6: Trade-off between Attack Success Rate and LPIPS Score of conventional AML attacks and SPAT attacks for GTSRB.

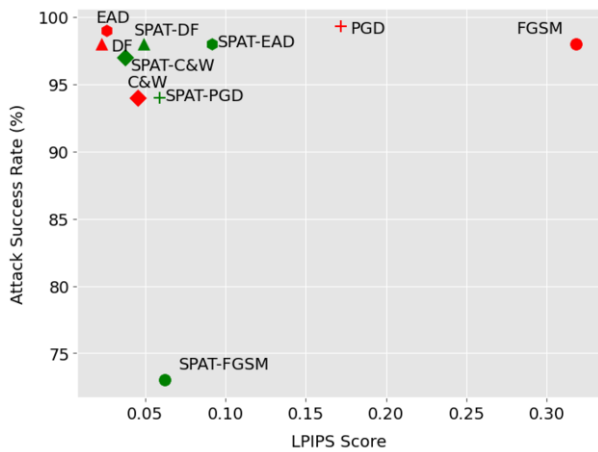

Figure 7: Trade-off between Attack Success Rate and LPIPS Score of conventional AML attacks and SPAT attacks for MNIST.

Table 3: Evaluation of adversarial samples generated using SPAT attacks (SPAT-DF & SPAT-C&W) against state-of-the-art robust models.

Robust Models	Clean Accuracy	Robust Accuracy	SPAT-DF	SPAT-C&W
Wang *et al.* [42]	93.25%	70.69%	66.80%	67.50
Rebuffi *et al.* [32]	92.23%	66.56%	60.93%	61.71
Gowal *et al.* [16]	88.74%	66.10%	53.90%	54.68
Rebuffi *et al.* [32]	88.50%	64.58%	57.03%	59.26
Sehwag *et al.* [33]	87.30%	62.79%	08.59%	09.26

with lower LPIPS scores and higher attack success rates, attacks at the top-left corners of the plot indicate better attacks than the rest.

4.2.1 CIFAR-10

We evaluated the robustness of the Resnet-18 classifier trained on the CIFAR-10 dataset using adversarial examples generated using SPAT attacks and summarised the results in Figure 5. We found that SPAT attacks achieve a higher attack success rate than their corresponding conventional AML attacks. Even though some SPAT attacks such as SPAT-DF and SPAT-FGSM have relatively lower attack success rates than other conventional AML attacks such as DF and PGD, they achieved better LPIPS score than other conventional AML attacks including DF and PGD. Out of all the SPAT attacks, SPAT-PGD and SPAT-C&W have achieved the most advantageous trade-off between attack success rate and LPIPS score.

4.2.2 GTSRB

We summarised the attack success rate and LPIPS score of adversarial examples generated using SPAT attacks in Figure 6. We found that, for GTSRB, almost all SPAT attacks have a higher or comparable attack success rate than all conventional AML attacks except SPAT-FGSM. In terms of LPIPS score, SPAT attacks overcome conventional AML attacks by a large margin. Overall, SPAT-PGD, SPAT-C&W, and SPAT-EAD have better trade-offs between attack success rate and LPIPS score than all other attacks.

4.2.3 MNIST

We summarised the results corresponding to MNIST in the Figure 7. We found that SPAT-C&W and SPAT-PGD learned better trade-offs than their corresponding conventional AML attacks. Surprisingly, for others, conventional AML attacks performed better than SPAT attacks. MNIST is a relatively simple dataset of grayscale images with very low variance across the dataset. Since the goal of SPAT attacks is to find *where* to put the adversarial perturbation, it is harder to find places on the image to hide the adversarial perturbation. We argue that this is a possible case for a relatively weak trade-off of SPAT attacks against conventional AML attacks.

4.3 Evasiveness Against Robust Models

We further evaluate the evasiveness of SPAT attacks against five other robust models. To perform the experiment, we used 1000 adversarial examples generated using SPAT-DF and SPAT-C&W attacks from the CIFAR-10 test dataset. We summarised the results in the Table 3. The first column represents clean accuracy: the accuracy of the model with benign examples. The second column is the robust accuracy: the accuracy of a classifier when injected with adversarial examples. This is the best-known robust accuracy obtained using AutoAttack [8] and other adaptive attacks [9]. Rest two columns represent robust accuracy obtained using SPAT-DF and SPAT-C&W.

Since robust accuracy is a measure of the robustness of a classifier, for an attack to be effective, it should lower the robust accuracy of the target model. We found that SPAT-DF and SPAT-C&W lowered the robust accuracy of all the robust models. Interestingly, we got a lower robust accuracy for a recently proposed robust model [42]. Using the generated adversarial examples, we dropped its robust accuracy by -3.89% using SPAT-DF and -3.19% using SPAT-C&W respectively. Additionally, we saw a significant drop in the robust accuracy of the model proposed in [33]. A difference in the adversarial examples used for adversarial training can explain the drop in robust accuracy in [33]. While other robust models in Table 3 used adversarial examples from the original dataset, authors in [33] utilised generative models (GANs and diffusion models) to generate adversarial examples. This distinction might have resulted in a higher reduction in robust accuracy compared to other models.

In this work, we tried to achieve a higher attack success rate while maintaining the perceptual similarity of the input image. We achieved this by constraining the perturbation location to certain parts of the input data that agree with the notion of semantic correctness for the Human Visual System. Works such as one-pixel attacks and adversarial patches tried perturbing a specific part of the input data instead of perturbing the entire image. We believe these attacks are special cases of constrained perturbations; for one-pixel, it is limited to just one pixel; for the patch, it is limited to the small rectangle area; and for SPAT attacks, it is limited to a region which is explicitly found by the algorithm. We did a qualitative comparison of PGD and SPAT-PGD for CIFAR-10 in Figure 4.

5 Conclusion

In this paper, we have proposed an algorithmic framework, SPAT, that generates semantic-preserving adversarial examples with a high attack success rate and high perceptual similarity with the benign examples. By perturbing the data manifold space using an autoencoder with a constrained loss function, SPAT avoids the problem of class invariance and produces valid adversarial examples. The generated adversarial examples were tested against state-of-the-art robust models and were found to lower the robust accuracy of the classifiers while making them more stealthy than the conventional adversarial attacks. This work provides a promising approach to generating adversarial examples that can evade detection by state-of-the-art attack detection mechanisms, even those that employ unexpected changes in the semantics of the input to detect an adversarial example. Further research in this direction could potentially improve the robustness of ML models against adversarial attacks.

References

[1] Anand Bhattad, Min Jin Chong, Kaizhao Liang, et al., 'Unrestricted adversarial examples via semantic manipulation', *arXiv preprint arXiv:1904.06347*, (2019).

[2] Tom Brown, Benjamin Mann, Nick Ryder, et al., 'Language models are few-shot learners', in *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., (2020).

[3] Tom B Brown, Nicholas Carlini, Chiyuan Zhang, et al., 'Unrestricted adversarial examples', *arXiv preprint arXiv:1809.08352*, (2018).

[4] Tom B Brown, Dandelion Mané, Subhrajit Roy, et al., 'Adversarial patch', *arXiv preprint arXiv:1712.09665*, (2017).

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, et al., 'End-to-end object detection with transformers', in *Computer Vision–ECCV 2020: 16th European Conference*, pp. 213–229. Springer, (2020).

[6] Nicholas Carlini and David Wagner, 'Towards evaluating the robustness of neural networks', in *2017 IEEE Symposium on Security and Privacy, SP*, pp. 39–57. IEEE Computer Society, (2017).

[7] Pin-Yu Chen, Yash Sharma, Huan Zhang, et al., 'EAD: elastic-net attacks to deep neural networks via adversarial examples', in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pp. 10–17. AAAI Press, (2018).

[8] Francesco Croce and Matthias Hein, 'Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks', in *International conference on machine learning*, pp. 2206–2216. PMLR, (2020).

[9] Francesco Croce and Matthias Hein, 'Robustbench: A standardized benchmark for adversarial robustness', *Journal of Machine Learning Research*, **22**(117), 1–29, (2021).

[10] Ousmane Amadou Dia, Elnaz Barshan, and Reza Babanezhad. Semantics preserving adversarial learning, 2019.

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, et al., 'An image is worth 16x16 words: Transformers for image recognition at scale', in *International Conference on Learning Representations*, (2021).

[12] Logan Engstrom, Brandon Tran, Dimitris Tsipras, et al. Exploring the landscape of spatial robustness, 2019.

[13] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, et al., 'Robust physical-world attacks on deep learning visual classification', *CVPR*, (2018).

[14] Justin Gilmer, Luke Metz, Fartash Faghri, et al., 'Adversarial spheres', (2018).

[15] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy, 'Explaining and harnessing adversarial examples', in *3rd International Conference on Learning Representations, ICLR*, (2015).

[16] Sven Gowal, Sylvestre-Alvise Rebuffi, Olivia Wiles, et al., 'Improving robustness using generated data', *Advances in Neural Information Processing Systems*, **34**, 4218–4233, (2021).

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, et al., 'Deep residual learning for image recognition', in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, (2016).

[18] Hossein Hosseini and Radha Poovendran, 'Semantic adversarial examples', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1614–1619, (2018).

[19] Sebastian Houben, Johannes Stallkamp, Jan Salmen, et al., 'Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark', in *International Joint Conference on Neural Networks*, number 1288, (2013).

[20] Q Huynh-Thu and M Ghanbari, 'The scope of validity of psnr in image/video quality assessment', *Electronics letters*, **44**(13), 800–801, (2008).

[21] Daniel Kang, Yi Sun, Dan Hendrycks, et al., 'Testing robustness against unforeseen adversaries', *arXiv preprint arXiv:1908.08016*, (2019).

[22] Alex Krizhevsky and Geoffrey Hinton, 'Learning multiple layers of features from tiny images', Technical Report 0, University of Toronto, (2009).

[23] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, 'Imagenet classification with deep convolutional neural networks', in *Advances in Neural Information Processing Systems*, eds., F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, volume 25. Curran Associates, Inc., (2012).

[24] Alexey Kurakin, Ian Goodfellow, and Samy Bengio, 'Adversarial examples in the physical world', *arXiv preprint arXiv:1607.02533*, (2016).

[25] Cassidy Laidlaw and Soheil Feizi, 'Functional adversarial attacks', *Advances in neural information processing systems*, **32**, (2019).

[26] Yann LeCun and Corinna Cortes, 'The mnist database of handwritten digits', (1998).

[27] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, et al., 'Towards deep learning models resistant to adversarial attacks', in *5th International Conference on Learning Representations, ICLR*, (2017).

[28] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard, 'Deepfool: A simple and accurate method to fool deep neural networks', in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 2574–2582. IEEE Computer Society, (2016).

[29] Maria-Irina Nicolae, Mathieu Sinn, Minh Ngoc Tran, et al., 'Adversarial robustness toolbox v1.0.0', in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18. ACM, (2018).

[30] Nicolas Papernot, Patrick McDaniel, Somesh Jha, et al., 'The limitations of deep learning in adversarial settings', in *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 372–387, (2016).

[31] Haonan Qiu, Chaowei Xiao, Lei Yang, et al. Semanticadv: Generating adversarial examples via attribute-conditional image editing, 2019.

[32] Sylvestre-Alvise Rebuffi, Sven Gowal, Dan A Calian, et al., 'Fixing data augmentation to improve adversarial robustness', *arXiv preprint arXiv:2103.01946*, (2021).

[33] Vikash Sehwag, Saeed Mahloujifar, Tinashe Handina, et al., 'Robust learning meets generative models: Can proxy distributions improve adversarial robustness?', *arXiv preprint arXiv:2104.09425*, (2021).

[34] Ali Shahin Shamsabadi, Ricardo Sanchez-Matilla, and Andrea Cavallaro, 'Colorfool: Semantic adversarial colorization', in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1151–1160, (2020).

[35] Hamid R Sheikh and Alan C Bovik, 'Image information and visual quality', *IEEE Transactions on Image Processing*, **15**(2), 430–444, (2006).

[36] Yang Song, Rui Shu, Nate Kushman, et al., 'Generative adversarial examples', *arXiv preprint arXiv:1805.07894*, (2018).

[37] David Stutz, Matthias Hein, and Bernt Schiele, 'Disentangling adversarial robustness and generalization', in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6976–6987, (2019).

[38] Subrat Swain. SPAT: Semantic Preserving Adversarial Transformation for Perceptually Similar Adversarial Examples. https://github.com/swainsubrat/SPAT, 2023.

[39] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, et al., 'Intriguing properties of neural networks', in *2nd International Conference on Learning Representations, ICLR*, (2014).

[40] Thomas Tanay and Lewis Griffin, 'A boundary tilting persepective on the phenomenon of adversarial examples', *arXiv preprint arXiv:1608.07690*, (2016).

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al., 'Attention is all you need', in *Advances in Neural Information Processing Systems, NIPS*.

[42] Zekai Wang, Tianyu Pang, Chao Du, et al. Better diffusion models further improve adversarial training, 2023.

[43] Zhou Wang, Alan C Bovik, Hamid R Sheikh, et al., 'Image quality assessment: from error visibility to structural similarity', *IEEE transactions on image processing*, **13**(4), 600–612, (2004).

[44] Eric Wong, Frank Schmidt, and Zico Kolter, 'Wasserstein adversarial examples via projected sinkhorn iterations', in *International Conference on Machine Learning*, pp. 6808–6817. PMLR, (2019).

[45] Chaowei Xiao, Jun-Yan Zhu, Bo Li, et al., 'Spatially transformed adversarial examples', *arXiv preprint arXiv:1801.02612*, (2018).

[46] Wufeng Xue, Lei Zhang, Xuanqin Mou, et al., 'Gradient magnitude similarity deviation: A highly efficient perceptual image quality index', *IEEE Transactions on Image Processing*, **23**(2), 684–695, (2014).

[47] Zhilin Yang, Zihang Dai, Yiming Yang, et al. Xlnet: Generalized autoregressive pretraining for language understanding, 2019.

[48] Lin Zhang, Lei Zhang, Xuanqin Mou, et al., 'Fsim: a feature similarity index for image quality assessment', *IEEE transactions on image processing*, **20**(8), 2378–2386, (2011).

[49] Richard Zhang, Phillip Isola, Alexei A Efros, et al., 'The unreasonable effectiveness of deep features as a perceptual metric', in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, (2018).

[50] Zhengli Zhao, Dheeru Dua, and Sameer Singh, 'Generating natural adversarial examples', *arXiv preprint arXiv:1710.11342*, (2017).