Classifier Robustness Enhancement Via Test-Time Transformation

Tsachi Blau [1] Roy Ganz [1] Chaim Baskin [2] Michael Elad [2] Alex Bronstein [2]

Abstract

It has been recently discovered that adversarially trained classifiers exhibit an intriguing property, referred to as perceptually aligned gradients (PAG). PAG implies that the gradients of such classifiers possess a meaningful structure, aligned with human perception. Adversarial training is currently the best-known way to achieve classification robustness under adversarial attacks. The PAG property, however, has yet to be leveraged for further improving classifier robustness. In this work, we introduce Classifier Robustness Enhancement Via Test-Time Transformation (TETRA) – a novel defense method that utilizes PAG, enhancing the performance of trained robust classifiers. Our method operates in two phases. First, it modifies the input image via a designated targeted adversarial attack into each of the dataset's classes. Then, it classifies the input image based on the distance to each of the modified instances, with the assumption that the shortest distance relates to the true class. We show that the proposed method achieves state-of-the-art results and validate our claim through extensive experiments on a variety of defense methods, classifier architectures, and datasets. We also empirically demonstrate that TETRA can boost the accuracy of any differentiable adversarial training classifier across a variety of attacks, including ones unseen at training. Specifically, applying TETRA leads to substantial improvement of up to $+23\%$, $+20\%$, and $+26\%$ on CIFAR10, CIFAR100, and ImageNet, respectively.

[1]Department of Electrical and ComputerEngineering, Israel Institute of Technology [2]Department of Computer Science, Israel Institute of Technology. Correspondence to: Tsachi Blau <tsachiblau@campus.technion.ac.il>, Roy Ganz <ganz@campus.technion.ac.il>, Chaim Baskin <chaimbaskin@cs.technion.ac.il>, Michael Elad <elad@cs.technion.ac.il>, Alex Bronstein <bron@cs.technion.ac.il>.

2^{nd} AdvML Frontiers workshop at 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

1. Introduction

Deep neural networks (DNNs) have revolutionized the visual recognition domain by achieving unprecedentedly high classification accuracy. DNN classifiers, however, have been shown to be highly sensitive to minor input perturbations (Hosseini et al., 2017; Dodge & Karam, 2017; Geirhos et al., 2017; Temel et al., 2018; Temel & AlRegib, 2018), thereby giving rise to the field of adversarial attacks (Szegedy et al., 2013; Goodfellow et al., 2014; Kurakin et al., 2016; Athalye et al., 2018; Biggio et al., 2013; Carlini & Wagner, 2017; Kurakin et al., 2018; Nguyen et al., 2015). Adversarial attacks are maliciously designed imperceptible perturbations added to an image intended to fool a classifier. These examples are worrisome because of the vast use of DNNs in critical tasks such as autonomous driving. Consequently, this field has gained considerable research attention, leading to the development of both new defenses and attack methods. One common way to create an adversarial example is achieved by an iterative optimization process that searches for a norm-bounded perturbation $||\delta||_p \leq \epsilon$ that is added to the input image. The properties of the perturbation are determined by a threat model, characterized by the choice of the radius ϵ and the norm ℓ_p (typical choices for the latter are $p \in \{1, 2, \infty\}$).

One leading defense method is adversarial training (AT) (Madry et al., 2017; Zhang et al., 2019; Rebuffi et al., 2021; Gowal et al., 2020; Salman et al., 2020; Goodfellow et al., 2014; Carlini & Wagner, 2017; Croce & Hein, 2020; Tramer et al., 2020), in which the classifier is trained in the presence of adversarial perturbation and learns to classify the maliciously modified training samples correctly. A major drawback of AT is its reliance on a specific attack type, which leads to poor generalization to unseen ones. This issue has been mentioned in (Hendrycks et al., 2021) as one of the unsolved problems in this field, and a few methods (Bai et al., 2021; Blau et al., 2022) have recently attempted to address it.

Another line of research (Zhang et al., 2021; Schwinn et al., 2022) focuses on test-time methods that aim to improve trained AT classifiers by changing their inference methodology. These methods change the standard prediction process where an instance is fed into a DNN model that outputs a class prediction. Instead, they follow an inference algo-

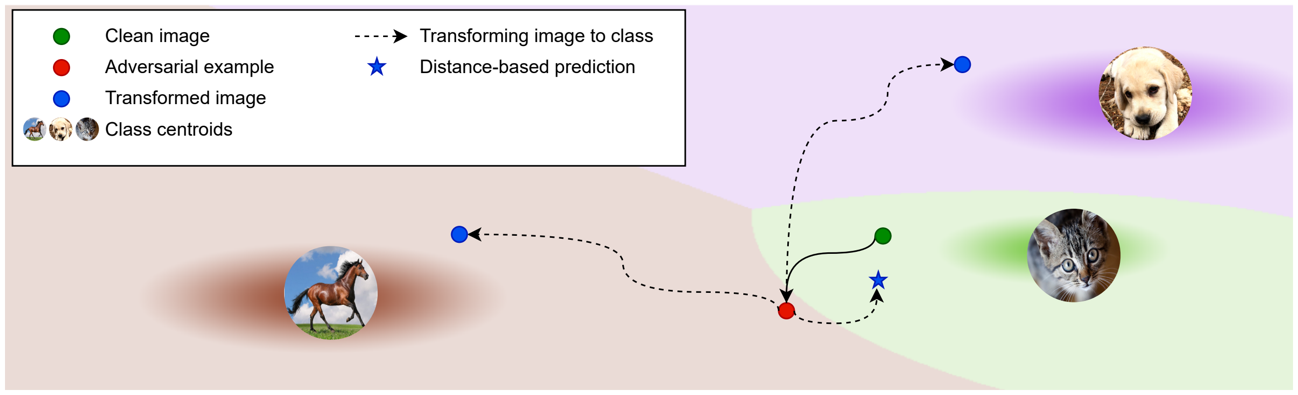

Figure 1. TETRA algorithm. A cartoon depiction of the classifier decision rule describes regions of the input space classified as distinct classes in different colors. Initially, a clean input image (green dot) belonging to the *cat* class is attacked, creating an adversarial example (red dot) that has a wrong predicted class (*horse* instead of *cat*). When TETRA is applied, in its first phase, it transforms the adversarial image into each one of the dataset classes. The transformation is represented by the dotted black arrows leading to a set of new images, one per class (blue dots). In its second phase, TETRA calculates the distances between the adversarial example to all of the generated images. Finally, TETRA classifies the input based on the calculated shortest distance, which leads to a correct class label prediction (*cat*).

rithm that strengthens the prediction. These methods are appealing since they can be applied to any trained classifier, without any further training or access to the training dataset. These advantages significantly enhance DNNs' robustness to adversarial attacks.

Recently, researchers discovered that AT classifiers exhibit a fascinating property that was called perceptually aligned gradients (PAG) (Engstrom et al., 2019; Etmann et al., 2019; Ross & Doshi-Velez, 2018; Tsipras et al., 2018). Simply put, the gradients of a classifier, with respect to the input image, manifest an intelligible spatial structure perceptually similar to that of some class of the dataset. As a result, maximizing the conditional probability of such classifiers towards some target class, via a gradient-based method, results in class-related semantic visual features. The discovery of PAG has drawn considerable research interest in the past few years, leading to a sequence of works. Tsipiras *et al.* (Tsipras et al., 2018) showed that PAG does not arise in every classifier; they emerge in AT classifiers and not in "vanilla-trained" ones. The work of (Ganz et al., 2022) demonstrated the surprising bidirectional connection between PAG and robustness. From the applicative point of view, to date, PAG has mainly been studied in the context of generative tasks (Ganz & Elad, 2021; Kawar et al., 2022).

In this work, we propose a novel test-time defense method that leverages the PAG property for further improving the robustness of trained robust classifiers. Our method improves a classifier's accuracy even when attacking with the same attack that it was trained on and significantly improves the robustness to unseen attacks. It can be applied to any differentiable classifier – hence the name Classifier Robustness Enhancement Via Test-Time Transformation (TETRA). Our method operates in two phases, as depicted in Figure 1. First, it modifies the input image via a designated targeted adversarial attack into each dataset class. This modification is meaningful due to the PAG property, which guarantees worthy gradients. Then, it classifies based on the modification distances, with the assumption that the shortest attack relates to the true class.

We validate our method through extensive experiments on a variety of defense methods, classifier architectures, and datasets. We empirically demonstrate that TETRA can boost the accuracy of any differentiable AT classifier across a variety of attacks, generalizing well to unseen ones. Specifically, applying TETRA leads to substantial improvement of up to $+23\%$, $+20\%$, and $+26\%$ on CIFAR10, CIFAR100 and ImageNet, leading to state-of-the-art performance on test-time defense tasks. To summarize, the key contributions of our work are:

- We present a novel test-time robust classification method that utilizes the PAG property and requires neither additional training nor access to the original training set.
- We show improved classifiers' robustness, evaluated both on trained attacks and on unseen ones.
- We evaluate our method's performance, achieving state-of-the-art results compared to other test-time methods.

2. Our method

In this section, we present the proposed TETRA method – a novel test-time defense against adversarial attacks. We introduce TETRA intuition and supporting evidence. We then propose a speed-up method referred to as FETRA, and a novel attack dubbed ranking projected gradient descent (RPGD), designed to find the worst-case examples for FETRA.

2.1. Test-time transformation

An adversarial example is a small norm perturbation that humans will hardly notice, yet it changes the classifier's prediction. It is widely accepted that the attack deviates an image off the image's manifold (Shamir et al., 2021; Tanay & Griffin, 2016; Khoury & Hadfield-Menell, 2018; Stutz et al., 2019). Hence, it seems reasonable to preprocess perturbed images (Xie et al., 2017; Buckman et al., 2018; Xu et al., 2017; Samangouei et al., 2018; Yang et al., 2019; Song et al., 2017; Du & Mordatch, 2019; Grathwohl et al., 2019; Hill et al., 2020; Yoon et al., 2021; Nie et al., 2022; Blau et al., 2022), resulting in a projection back to the image's manifold. One unexplored way to perform a projection is through a transformation into a target class, which we refer to as targeted transformation. When performing targeted transformation toward the true class, we are projecting the image back to the image's manifold, resulting in an image with close resemblance to the input image. Unfortunately, we do not know the true label; therefore, we perform a targeted transformation of the image into each and every one of the dataset classes. We assume that a transformation into a different class changes the image considerably. Hence, we can classify the image based on the distances between the input image to the transformed images, where the shortest distance relates to the true class.

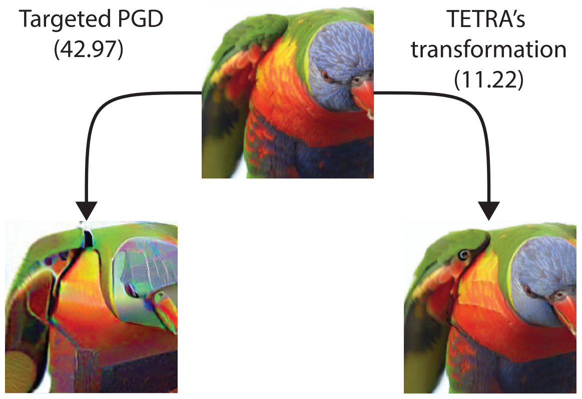

Figure 2. TETRA vs. PGD targeted transformation. In this figure, we present a comparison between targeted PGD and TETRA's transformation, where both methods transform an image into the desired class. PGD changes the image considerably in order to change the classification into the *toucan* class, while TETRA remains pixel-wise close to the input image and changes the wing of the *lorikeet* only slightly so it resembles a *toucan*. Pixel-level distance is presented next to the method as the ℓ_2 distance.

As presented in Appendix B, AT classifiers possess the PAG property, which was studied in the context of generative tasks (Ganz & Elad, 2021; Kawar et al., 2022). We propose to leverage this generative power for the targeted transformation task that lies at the core of TETRA, avoiding the use of a specialized generative model. PAG guarantees that the classifier gradients possess a meaningful structure. When these are put into an iterative optimization task, the input image is gradually transformed into a target class. When transforming an image without regularization, however, it might undergo significant changes, greatly departing from the initial image. Figure 2 visualizes targeted image transformations obtained from PGD and TETRA. While PGD changes the image considerably as it moves toward the target class, TETRA's effect is more gentle, pertaining only to certain regions of the image and adding semantic features that are related to the target class. An additional demonstration is provided in Figure 3, where we visualize a targeted transformation of images into different classes.

To better explain our method, we move now to the cartoon depiction presented in Figure 1. In this example, we attack an image of a *cat* (green dot), creating an attacked image (red dot). The attacked image looks like a *cat*, but the classifier mistakenly confuses it with a *horse*. Next, we transform the attacked image into every one of the dataset's classes, resulting in transformed images (blue dots). Without regularization, the transformed images would end up in the classes' centroids. However, a regularized transformation produces more gentle transformations (blue points). Finally, we calculate the distances between the attacked image and the transformed images, correctly classifying based on the shortest distance (blue star).

Algorithm 1 Test-Time Transformation

```
    Input classifier $f(\cdot)$, input $x$, target label $y$, norm
          radius $\epsilon$, step size $\alpha$, number of steps $N$,
          hyperparameter $\gamma$, classification classes $K$
 1: procedure TETRA
 2:     for $i$ in $1:K$ do
 3:         $\delta_i \leftarrow 0$
 4:         for $j$ in $1:N$ do
 5:             $\delta_i \leftarrow \Pi\left(\delta_i - \alpha \frac{\nabla_{\delta_i} L_{CE}(f(x+\delta_i),y)+\gamma\,\delta_i}{\|\nabla_{\delta_i} L_{CE}(f(x+\delta_i),y)+\gamma\,\delta_i\|_2}\right)$
 6:         end for
 7:         $d_i \leftarrow \|\delta_i\|_2$
 8:     end for
 9:     $\hat{y} = \arg\min_{i=1,\ldots,K} d_i$
10:     return $\hat{y}$
11: end procedure
```

The operator $\Pi(z)$ projects z onto the image domain $\mathbb{R}^{M\times N\times 3} \in [0,1]^{M\times N\times 3}$.

In more detail, let us be given an input image x, which might be attacked. We then perform an iterative optimization task, which is presented in Algorithm 1. The optimization objective,

$$L_{\mathrm{CE}}\left(f\left(x+\delta\right),y\right)+\frac{\gamma}{2}\cdot\|\delta\|_2^2 \qquad (1)$$

comprises two terms: a cross-entropy (CE) loss, and an ℓ_2 loss. The first term drives the image to change in a manner that makes the classifier predict the target class y. Because the classifier possesses PAG, the CE gradients exhibit a target class related semantic. In contrast, the second term regularizes the transformation so that the transformed image remains close to the input image x. More specifically, this expression is optimized iteratively for N steps via gradient descent, where every step is of size α. In addition, to balance these two terms, the hyperparameter γ needs to be fine-tuned.

Our method performs an exhaustive search for the correct class, as we transform the image into each dataset class. When scaling to datasets with many classes, this method becomes impracticable. Hence, we developed fast TETRA (FETRA) – a method that prunes some classes, saves computations, and allows scaling to large datasets. FETRA operates by pruning the classifier's low-rank class predictions and keeping only the top k classes. These classes, in turn, are classified using TETRA. FETRA can speed up runtime drastically. For example, we can boost performance by $\times 100$ over a dataset with 1000 classes, when performing FETRA with $k = 10$. FETRA, however, relies heavily on the hypothesis that the true class is highly ranked among the classifier's predictions. This assumption can be used maliciously by the attacker, who can exploit this knowledge to modify the attack. Standard attacks focus on changing the image so that the classifier does not predict the true class as the most likely one. Nevertheless, there might be an attack that aims at lowering the rank of an image, not just to the second most likely class, but even further down.

2.2. Ranking projected gradient descent

To be able to evaluate FETRA appropriately, we created a novel attack called ranking projected gradient descent (RPGD). This attack is tailored to our defense, trying to attack it at its known weaknesses. The objective of RPGD is to reduce the rank of the true class, predicted by the classifier, out of the top k most probable classes. We modify the loss term of the PGD algorithm (Madry et al., 2017), replacing it with a differentiable ranking loss (Blondel et al., 2020)[1]. Ranking loss is very useful for many tasks such as in ranking statistics. This, however, is a nontrivial expression for a loss term, since it involves ranking and sorting, which are non-differentiable operations. To conclude, we aim at attacking FETRA using its worst-case examples.

[1]We use the implementation supplied in `https://github.com/teddykoker/torchsort`

3. Experiments

In this section, we provide empirical evidence supporting our method. First, we apply TETRA and FETRA to various adversarially trained methods and architectures, evaluating the performance on CIFAR10, CIFAR100 (Krizhevsky, 2009) and ImageNet (Deng et al., 2009) datasets. Next, we evaluate the RPGD attack, providing visual evidence and intuition for our method. Finally, we offer an ablation study in Appendix F that presents the necessity of the PAG property for TERTA and evaluates different distance metrics. Throughout our experiments, we evaluate multiple threat models for every robust classifier, using AutoAttack (Croce & Hein, 2020), as it is a strong and widely accepted attack benchmark. Further experimental details are provided in Appendix C.

3.1. CIFAR10 and CIFAR100

In this part, we supply results over CIFAR10 and CIFAR100 (Krizhevsky, 2009), which are two of the most common datasets used for robustness evaluation. We compare the proposed TETRA to the DQR method from (Schwinn et al., 2022) which is a test-time method that boosts a robust classifier's performance. Hence, for every robust classifier, we show its results and the results of the other methods, creating a few consecutive lines of results per AT classifier. Moreover, we use four common threat models: $(\ell_\infty, \epsilon = 8/255)$, $(\ell_\infty, \epsilon = 16/255)$, $(\ell_2, \epsilon = 0.5)$, $(\ell_2, \epsilon = 1.0)$ and evaluate the performance over the complete test dataset, as the top 1 classification accuracy. The test dataset contains $10,000$ images, in both CIFAR10 and CIFAR100 datasets.

The results on CIFAR10 are summarized in Table 1. We use the top performing methods (Madry et al., 2017; Rebuffi et al., 2021; Gowal et al., 2020) as baselines and demonstrate that TETRA significantly improves their performance. Interestingly, TETRA boosts baseline methods both for seen and unseen attacks. In particular, for seen attacks, we enhance the baseline method performance by up to 9%, outperforming the other methods by up to 1%. For the unseen attacks, TETRA boosts the performance of baseline methods by up to 23%, outperforming the other methods by up to 10%. Additionally, we compare TETRA to the PAT method (Laidlaw et al., 2020) that addresses the issue of robustness to unseen attacks, and report up to 52% higher accuracy.

The results on CIFAR100 are presented in Table 2. We present our defenses, TETRA and FETRA, where FETRA uses the top 10 classes and accelerates the calculation by a factor of 10. We use the defense methods (Rebuffi et al., 2021; Gowal et al., 2020) as baseline methods and demonstrate that TETRA improves performance both for seen and unseen attacks. For seen attacks, we enhance the baseline method performance by up to 10% and the performance of other methods by up to 3%, while for unseen attacks the

Method	Architecture	TTM	Standard	Attack			
				L_∞		L_2	
				$8/255$	$16/255$	0.5	1.0
PAT (Laidlaw et al., 2020)	RN50		71.60%	28.70%	–	33.30%	–
AT (Madry et al., 2017)	RN50	$L_2, \epsilon = 0.5$	90.83%	29.04%	00.93%	69.24%	36.21%
DRQ (Schwinn et al., 2022)			88.79%	45.37%	07.09%	77.56%	51.28%
TETRA			87.40%	**51.66%**	**14.96%**	**78.66%**	**59.82%**
Rebuffi et al. (Rebuffi et al., 2021)	WRN28-10	$L_2, \epsilon = 0.5$	**91.79%**	47.85%	05.00%	78.80%	54.73%
DRQ (Schwinn et al., 2022)			90.99%	58.66%	**13.69%**	84.12%	64.69%
TETRA			88.23%	**59.99%**	11.45%	**85.56%**	**66.15%**
Rebuffi et al. (Rebuffi et al., 2021)	WRN28-10	$L_\infty, \epsilon = 8/255$	**87.33%**	60.77%	25.44%	66.72%	35.01%
DRQ (Schwinn et al., 2022)			87.17%	66.23%	33.62%	72.24%	44.56%
TETRA			85.00%	**66.86%**	**34.88%**	**74.24%**	**53.02%**
Gowal et al. (Gowal et al., 2020)	WRN70-16	$L_\infty, \epsilon = 8/255$	**91.09%**	65.88%	25.95%	66.43%	27.21%
DRQ (Schwinn et al., 2022)			90.77%	71.00%	35.89%	72.87%	39.51%
TETRA			88.18%	**72.02%**	**40.30%**	**75.90%**	**49.21%**

Table 1. CIFAR10 results. The first column displays the method used. For every base method, we report three consecutive lines of results: one for the base method and then two test-time boosting methods, DRQ (Schwinn et al., 2022) and TETRA. The next columns, name the architecture, the trained threat model (TTM), and the four attacks with different threat models.

baseline performance is increased by up to 20%, and the performance of other methods by up to 8%. It is important to emphasize that both methods, TETRA and FETRA, present similar results as the average gap is 0.66%. This demonstrates that FETRA's filtering is not the reason for the performance enhancement, and that it is just a speed-up method.

3.2. ImageNet

The results over ImageNet are reported in Table 3. Evaluating ImageNet is infeasible using TETRA because of memory and runtime issues. Therefore, we only evaluate the faster defense variant FETRA, with top $k = 20$ classes, which accelerates the calculation by a factor of 50. Moreover, we use four common threat models: $(\ell_\infty, \epsilon = 4/255)$, $(\ell_\infty, \epsilon = 8/255)$, $(\ell_2, \epsilon = 3.0)$, $(\ell_2, \epsilon = 6.0)$ and evaluate the performance over the complete test dataset ($50,000$ images), as the top 1 classification accuracy. We use the defense methods (Madry et al., 2017; Salman et al., 2020) as baselines, demonstrating that FETRA improves the performance both for seen and unseen attacks. For seen attacks, we enhance the baseline performance by up to 15% and for unseen attacks we enhance the baseline by up to 26%.

3.3. Ranking projected gradient descent

In this part, we compare the performance of two attacks over ImageNet dataset. We compare RPGD to PGD (RPGD, see Section 2). We show that RPGD better fits FETRA as it is designed for its weakness. Furthermore, we demonstrate that the accuracy drop is neglectable compared to the gain achieved by using FETRA.

We start by comparing the top k accuracy performances of both RPGD and PGD. As stated before in Section 2, AutoAttack does not utilize the known weakness of FETRA, hence does not create its worst-case examples. FETRA keeps only the top k classes, estimated by a robust classifier, and then applies TETRA to classify the remaining ones. AutoAttack does not exploit the knowledge regarding filtering, which leads us to the introduction of RPGD. RPGD is designed to enable a fair robust evaluation, searching for FETRA's worst-case examples. This is achieved by pushing the rank of the true class as low as possible and out of the top k classes. To verify that RPGD indeed achieves its goal, we suggest the analysis presented in Figure 6. This analysis is performed using Madry *et al.*'s defense method, trained on $\ell_2, \epsilon = 3.0$. On the left-hand side of the figure, we analyze the performance of both attacks, RPGD and PGD, by evaluating the top k accuracy for different k values. To better display the gap between the methods, we also provide the right-hand of the graph. This plot presents the difference between the two left-side graphs and demonstrates that each attack is preferred for its objective. For low k values, PGD achieves lower accuracy, but for high k values, RPGD achieves up to 5% lower accuracy.

Next, we verify that FETRA's performance enhancement still holds, even when considering the performance drop caused by attacking with RPGD. To this end, we show Ta-

Method	Architecture	TTM	Standard	Attack			
				L_∞		L_2	
				$8/255$	$16/255$	0.5	1.0
Rebuffi et al. (Rebuffi et al., 2021)	WRN28-10	$L_\infty, \epsilon = 8/255$	62.40%	32.06%	12.47%	38.32%	18.86%
DRQ (Schwinn et al., 2022)	WRN28-10	$L_\infty, \epsilon = 8/255$	61.32%	38.22%	19.41%	44.58%	26.78%
TETRA	WRN28-10	$L_\infty, \epsilon = 8/255$	55.18%	37.61%	19.72%	45.86%	32.79%
FETRA	WRN28-10	$L_\infty, \epsilon = 8/255$	57.95%	**38.73%**	**19.87%**	**47.56%**	**33.01%**
Gowal et al. (Gowal et al., 2020)	WRN70-16	$L_\infty, \epsilon = 8/255$	**69.15%**	36.90%	13.64%	40.86%	17.20%
DRQ (Schwinn et al., 2022)	WRN70-16	$L_\infty, \epsilon = 8/255$	69.12%	43.96%	20.25%	48.95%	25.43%
TETRA	WRN70-16	$L_\infty, \epsilon = 8/255$	59.83%	44.66%	**23.81%**	51.32%	**37.02%**
FETRA	WRN70-16	$L_\infty, \epsilon = 8/255$	62.47%	**46.09%**	23.48%	**53.06%**	36.19%

Table 2. CIFAR100 results. The first column displays the method used. For every base method, we report four consecutive lines of results: one for the base method and then three test-time boosting methods, DRQ (Schwinn et al., 2022), TETRA and FETRA. The next columns name the architecture, the trained threat model (TTM), and the four attacks with different threat models.

Method	Architecture	TTM	Standard	Attack			
				L_∞		L_2	
				$4/255$	$8/255$	3.0	6.0
Madry et al. (Madry et al., 2017)	RN50	$L_2, \epsilon = 3.0$	**57.90%**	24.82%	05.26%	30.85%	10.35%
FETRA	RN50	$L_2, \epsilon = 3.0$	51.51%	**40.64%**	**22.10%**	**45.75%**	**34.0%**
Salman et al. (Salman et al., 2020)	WRN50-2	$L_2, \epsilon = 3.0$	**66.91%**	30.86%	06.19%	38.27%	12.85%
FETRA	WRN50-2	$L_2, \epsilon = 3.0$	63.27%	**47.68%**	**25.23%**	**53.35%**	**38.36%**
Madry et al. (Madry et al., 2017)	RN50	$L_\infty, \epsilon = 4/255$	**62.42%**	28.96%	08.12%	07.06%	00.39%
FETRA	RN50	$L_\infty, \epsilon = 4/255$	55.75%	**40.45%**	**17.97%**	**18.03%**	**04.42%**
Salman et al. (Salman et al., 2020)	WRN50-2	$L_\infty, \epsilon = 4/255$	**68.41%**	37.80%	12.84%	07.03%	00.21%
FETRA	WRN50-2	$L_\infty, \epsilon = 4/255$	60.44%	**48.44%**	**25.44%**	**20.18%**	**04.24%**

Table 3. ImageNet results. The first column displays the method used. For every base method, we report two consecutive lines of results: one is for the base method and the other is FETRA, which is our test-time boosting method. The next columns name the architecture, the trained threat model (TTM), and the four attacks with different threat models.

ble 3, which presents an evaluation of the ImageNet dataset attacked by AutoAttack and defended by FETRA with top $k = 20$. FETRA enhances the base performance by at least 9% and up to 26%. The performance reduction caused by RPGD for the same k value is 2.5%, much lower than the demonstrated enhancement. To conclude, RPGD is a stronger attack for FETRA, resulting in a decreased performance. Nevertheless, FETRA achieves a significant performance enhancement. A similar analysis for CIFAR10 and CIFAR100 is presented in Appendix E.

4. Discussion and conclusion

This work presents a novel test-time adversarially robust classification method called TETRA, which requires neither training nor access to training data. TETRA utilizes the PAG property in order to boost the performance of any differentiable AT classifier. To the best of our knowledge, this is the first time that an AT classifier has exploited the PAG property. Our method was validated through an extensive evaluation, using AutoAttack on different architectures and AT training methods, and three standard datasets: CIFAR10, CIFAR100 and ImageNet. Although our method's improvement comes at the cost of a slight clean accuracy degradation and longer inference time, it significantly improves the performance of adversarially perturbed data compared to previous methods. Future work will mainly focus on improving clean accuracy and on accelerating inference time.

References

Athalye, A., Engstrom, L., Ilyas, A., and Kwok, K. Synthesizing robust adversarial examples. In *International conference on machine learning*, pp. 284–293. PMLR, 2018.

Bai, T., Luo, J., Zhao, J., Wen, B., and Wang, Q. Recent advances in adversarial training for adversarial robustness. *arXiv preprint arXiv:2102.01356*, 2021.

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., Giacinto, G., and Roli, F. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pp. 387–402. Springer, 2013.

Blau, T., Ganz, R., Kawar, B., Bronstein, A., and Elad, M. Threat model-agnostic adversarial defense using diffusion models. *arXiv preprint arXiv:2207.08089*, 2022.

Blondel, M., Teboul, O., Berthet, Q., and Djolonga, J. Fast differentiable sorting and ranking. In *International Conference on Machine Learning*, pp. 950–959. PMLR, 2020.

Buckman, J., Roy, A., Raffel, C., and Goodfellow, I. Thermometer encoding: One hot way to resist adversarial examples. In *International Conference on Learning Representations*, 2018.

Carlini, N. and Wagner, D. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM workshop on artificial intelligence and security*, pp. 3–14, 2017.

Cohen, J., Rosenfeld, E., and Kolter, Z. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pp. 1310–1320. PMLR, 2019.

Croce, F. and Hein, M. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International conference on machine learning*, pp. 2206–2216. PMLR, 2020.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Dodge, S. and Karam, L. A study and comparison of human and deep learning recognition performance under visual distortions. In *2017 26th international conference on computer communication and networks (ICCCN)*, pp. 1–7. IEEE, 2017.

Du, Y. and Mordatch, I. Implicit generation and generalization in energy-based models. *arXiv preprint arXiv:1903.08689*, 2019.

Engstrom, L., Ilyas, A., Santurkar, S., Tsipras, D., Tran, B., and Madry, A. Adversarial robustness as a prior for learned representations. *arXiv preprint arXiv:1906.00945*, 2019.

Etmann, C., Lunz, S., Maass, P., and Schönlieb, C.-B. On the connection between adversarial robustness and saliency map interpretability. *arXiv preprint arXiv:1905.04172*, 2019.

Ganz, R. and Elad, M. Bigroc: Boosting image generation via a robust classifier. 2021.

Ganz, R., Kawar, B., and Elad, M. Do perceptually aligned gradients imply adversarial robustness? *arXiv preprint arXiv:2207.11378*, 2022.

Geirhos, R., Janssen, D. H., Schütt, H. H., Rauber, J., Bethge, M., and Wichmann, F. A. Comparing deep neural networks against humans: object recognition when the signal gets weaker. *arXiv preprint arXiv:1706.06969*, 2017.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Gowal, S., Qin, C., Uesato, J., Mann, T., and Kohli, P. Uncovering the limits of adversarial training against norm-bounded adversarial examples. *arXiv preprint arXiv:2010.03593*, 2020.

Grathwohl, W., Wang, K.-C., Jacobsen, J.-H., Duvenaud, D., Norouzi, M., and Swersky, K. Your classifier is secretly an energy based model and you should treat it like one. *arXiv preprint arXiv:1912.03263*, 2019.

Hendrycks, D., Carlini, N., Schulman, J., and Steinhardt, J. Unsolved problems in ml safety. *arXiv preprint arXiv:2109.13916*, 2021.

Hill, M., Mitchell, J., and Zhu, S.-C. Stochastic security: Adversarial defense using long-run dynamics of energy-based models. *arXiv preprint arXiv:2005.13525*, 2020.

Hosseini, H., Xiao, B., and Poovendran, R. Google's cloud vision api is not robust to noise. In *2017 16th IEEE international conference on machine learning and applications (ICMLA)*, pp. 101–105. IEEE, 2017.

Kawar, B., Ganz, R., and Elad, M. Enhancing diffusion-based image synthesis with robust classifier guidance. *arXiv preprint arXiv:2208.08664*, 2022.

Khoury, M. and Hadfield-Menell, D. On the geometry of adversarial examples. *arXiv preprint arXiv:1811.00525*, 2018.

Krizhevsky, A. Learning multiple layers of features from tiny images. 2009.

Kurakin, A., Goodfellow, I., and Bengio, S. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

Kurakin, A., Goodfellow, I. J., and Bengio, S. Adversarial examples in the physical world. In *Artificial intelligence safety and security*, pp. 99–112. Chapman and Hall/CRC, 2018.

Laidlaw, C., Singla, S., and Feizi, S. Perceptual adversarial robustness: Defense against unseen threat models. *arXiv preprint arXiv:2006.12655*, 2020.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Nguyen, A., Yosinski, J., and Clune, J. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 427–436, 2015.

Nie, W., Guo, B., Huang, Y., Xiao, C., Vahdat, A., and Anandkumar, A. Diffusion models for adversarial purification. *arXiv preprint arXiv:2205.07460*, 2022.

Raff, E., Sylvester, J., Forsyth, S., and McLean, M. Barrage of random transforms for adversarially robust defense. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6528–6537, 2019.

Rebuffi, S.-A., Gowal, S., Calian, D. A., Stimberg, F., Wiles, O., and Mann, T. Fixing data augmentation to improve adversarial robustness. *arXiv preprint arXiv:2103.01946*, 2021.

Ross, A. and Doshi-Velez, F. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Salman, H., Ilyas, A., Engstrom, L., Kapoor, A., and Madry, A. Do adversarially robust imagenet models transfer better? *Advances in Neural Information Processing Systems*, 33:3533–3545, 2020.

Samangouei, P., Kabkab, M., and Chellappa, R. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *arXiv preprint arXiv:1805.06605*, 2018.

Schwinn, L., Bungert, L., Nguyen, A., Raab, R., Pulsmeyer, F., Precup, D., Eskofier, B., and Zanca, D. Improving robustness against real-world and worst-case distribution shifts through decision region quantification. *arXiv preprint arXiv:2205.09619*, 2022.

Shamir, A., Melamed, O., and BenShmuel, O. The dimpled manifold model of adversarial examples in machine learning. *arXiv preprint arXiv:2106.10151*, 2021.

Song, Y., Kim, T., Nowozin, S., Ermon, S., and Kushman, N. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. *arXiv preprint arXiv:1710.10766*, 2017.

Stutz, D., Hein, M., and Schiele, B. Disentangling adversarial robustness and generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6976–6987, 2019.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Tanay, T. and Griffin, L. A boundary tilting persepective on the phenomenon of adversarial examples. *arXiv preprint arXiv:1608.07690*, 2016.

Temel, D. and AlRegib, G. Traffic signs in the wild: Highlights from the ieee video and image processing cup 2017 student competition [sp competitions]. *arXiv preprint arXiv:1810.06169*, 2018.

Temel, D., Lee, J., and AlRegib, G. Cure-or: Challenging unreal and real environments for object recognition. In *2018 17th IEEE international conference on machine learning and applications (ICMLA)*, pp. 137–144. IEEE, 2018.

Tramer, F., Carlini, N., Brendel, W., and Madry, A. On adaptive attacks to adversarial example defenses. *Advances in Neural Information Processing Systems*, 33:1633–1645, 2020.

Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., and Madry, A. Robustness may be at odds with accuracy. *arXiv preprint arXiv:1805.12152*, 2018.

Xie, C., Wang, J., Zhang, Z., Ren, Z., and Yuille, A. Mitigating adversarial effects through randomization. *arXiv preprint arXiv:1711.01991*, 2017.

Xu, W., Evans, D., and Qi, Y. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017.

Yang, Y., Zhang, G., Katabi, D., and Xu, Z. Me-net: Towards effective adversarial robustness with matrix estimation. *arXiv preprint arXiv:1905.11971*, 2019.

Yoon, J., Hwang, S. J., and Lee, J. Adversarial purification with score-based generative models. In *International Conference on Machine Learning*, pp. 12062–12072. PMLR, 2021.

Zhang, H., Yu, Y., Jiao, J., Xing, E., El Ghaoui, L., and Jordan, M. Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning*, pp. 7472–7482. PMLR, 2019.

Zhang, M., Levine, S., and Finn, C. Memo: Test time robustness via adaptation and augmentation. *arXiv preprint arXiv:2110.09506*, 2021.

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. 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.

A. Related work

This section presents related works that use preprocessing and test-time inference methods for enhanced classification robustness. The former preprocesses the image before sending it to the classifier, whereas the latter changes the inference methodology, using an inference algorithm. Finally, we discuss several adversarial defense methods that aim at unseen attacks.

The first work on prepossessing methods, which improve classification robustness, includes random resizing and padding (Xie et al., 2017), thermometer encoding (Buckman et al., 2018), feature squeezing (Xu et al., 2017), defense GAN (Samangouei et al., 2018) and masking and reconstructing (Yang et al., 2019). A newer line of preprocessing methods uses probabilistic models. These methods aim to leverage their generative power to clear perturbations from an attacked image. They perform it by projecting the attacked image back to the data manifold. This line of work includes purification by pixelCNN (Song et al., 2017), EBM for restoring corrupt images (Du & Mordatch, 2019) and a density aware classifier (Grathwohl et al., 2019). The most recent works in this field includes Langevin sampling (Hill et al., 2020) and a gradient ascent score-based model (Yoon et al., 2021). Similarly, two recent studies utilize the generative power of diffusion models (Nie et al., 2022; Blau et al., 2022).

Another group of adversarial defense schemes uses test-time methods. Cohen *et al.* (Cohen et al., 2019) and Raff *et al.* (Raff et al., 2019) suggest multiple realizations of the augmented image to be performed during test-time, followed by averaging the classification predictions. Schwinn *et al.* (Schwinn et al., 2022) suggest to analyze the robustness of a local decision region nearby the attacked image. While (Cohen et al., 2019; Raff et al., 2019) require fine-tuning the model, Schwinn *et al.* require neither fine-tuning nor access to the training data, similar to our method.

The last group of studies aims to provide robustness against unseen attacks. Laidlaw *et al.* (Laidlaw et al., 2020) provides a latent space norm-bounded AT, and Blau *et al.* (Blau et al., 2022) uses a vanilla trained diffusion model as a preprocessing step.

To the best of our knowledge, Schwinn *et al.* (Schwinn et al., 2022) offer the only test time method that enhances adversarially trained robust classifiers without further training, similar to our method.

B. Background

Since the discovery of adversarial attacks (Szegedy et al., 2013; Goodfellow et al., 2014; Kurakin et al., 2016; Athalye et al., 2018; Biggio et al., 2013; Carlini & Wagner, 2017; Kurakin et al., 2018; Nguyen et al., 2015), there has been a continuous development of defense and attack techniques. In this section, we briefly overview key results starting with attacks, moving to defense strategies, and finishing with the PAG property.

An adversarial attack is a perturbation δ, added to an image x, intended to push a classifier decision away from the correct prediction. Many important studies researched adversarial attacks, of which it is important to mention Madry *et al.* (Madry et al., 2017) who laid the foundation for many following works, including ours. Madry *et al.* introduced the projected gradient descent (PGD) algorithm, which is an iterative optimization process that searches for the worst-case adversarial example. PGD has access to the classifier's weights, and is, therefore, able to find the worst-case adversary in a small radius around a clean data sample. The allowed perturbation is defined by a threat model, characterized by an ℓ_p norm and a radius ϵ, such that $||\delta||_p \leq \epsilon$. There exist two variants for the loss term. The objective of the first variant is to maximize the classification loss of the perturbed input, given the true label. In other words, the input image is manipulated in order to increase the error, aiming for a wrong classifier decision (*any* class except the correct one). The second variant's objective is to minimize the classification loss of the perturbed image given a *specific* wrong class label y. As a result, the classifier is more likely to predict label y. Our method utilizes the latter *targeted PGD* variant.

One of the leading robustification methods known as adversarial training (AT) (Madry et al., 2017; Zhang et al., 2019; Rebuffi et al., 2021; Gowal et al., 2020; Salman et al., 2020; Goodfellow et al., 2014; Carlini & Wagner, 2017; Croce & Hein, 2020; Tramer et al., 2020). AT is a training method that robustifies a classifier against a specific attack. This is achieved by introducing adversarial examples during the training process, and driving the classifier to learn to infer the true labels of malicious examples. While training, the classifier weights are constantly being changed. As a result, the classifier's worst-case examples keep on changing as well. Hence, in every training batch, one must calculate new adversarial examples that fit the current state of the classifier.

It has been recently discovered that some classifiers exhibit an intriguing property called perceptually aligned gradients

(PAG) (Engstrom et al., 2019; Etmann et al., 2019; Ross & Doshi-Velez, 2018; Tsipras et al., 2018). PAG is manifested through the classification loss gradients, with respect to the input image, appearing visually meaningful to humans, and resembling one of the dataset classes. The structure of the gradients is different when performing untargeted vs. targeted PGD. When performing untargeted PGD, the attack is not leading to a specific class; therefore, the gradients transform the image arbitrarily. When performing targeted PGD, however, the gradients transform the image into the target class, removing current class features. Using this property, our method enables, one to transform an image into a target class, as used by our method.

C. Experimental setup

In this part, we provide some details about our methods. We are performing AutoAttack (Croce & Hein, 2020) on the base classifier, then performing our defense method. Similar to previous works such as (Schwinn et al., 2022). We use $N = 30$ steps of TETRA and perform hyperparameter tuning, finding the best step size α and γ for every classifier. We use these parameters for all the evaluations, clean images and all of the attacks. We state both α and γ for every dataset in Tables 4 to 6.

We evaluated the other test-time defense, DRQ (Schwinn et al., 2022), with the official code using the reported parameters.

Method	Architecture	TTM	α	γ
Madry et al. (Madry et al., 2017) + TETRA	RN50	$L_2, \epsilon = 0.5$	1.5	200
Rebuffi et al. (Rebuffi et al., 2021) + TETRA	WRN28-10	$L_2, \epsilon = 0.5$	0.5	400
Rebuffi et al. (Rebuffi et al., 2021) + TETRA	WRN28-10	$L_\infty, \epsilon = 8/255$	0.1	300
Gowal et al. (Gowal et al., 2020) + TETRA	WRN70-16	$L_\infty, \epsilon = 8/255$	0.3	300
Vanila + TETRA	WRN28-10	-	0.05	300

Table 4. CIFAR10 params. In the first column, we state the method. In the next columns, we state the architecture, the trained threat model (TTM), α which is the step size and γ which is the regularization weight.

Method	Architecture	TTM	α	γ
Rebuffi et al. (Rebuffi et al., 2021) + TETRA	WRN28-10	$L_\infty, \epsilon = 8/255$	0.1	300
Rebuffi et al. (Rebuffi et al., 2021) + FETRA	WRN28-10	$L_\infty, \epsilon = 8/255$	0.1	300
Gowal et al. (Gowal et al., 2020) + TETRA	WRN70-16	$L_\infty, \epsilon = 8/255$	0.1	100
Gowal et al. (Gowal et al., 2020) + FETRA	WRN70-16	$L_\infty, \epsilon = 8/255$	0.1	100

Table 5. CIFAR100 params. In the first column, we state the method. In the next columns, we state the architecture, the trained threat model (TTM), α which is the step size and γ which is the regularization weight.

Method	Architecture	TTM	α	γ
Madry et al. (Madry et al., 2017) + FETRA	RN50	$L_2, \epsilon = 3.0$	6.0	5500
Salman et al. (Salman et al., 2020) + FETRA	WRN50-2	$L_2, \epsilon = 3.0$	6.0	3000
Madry et al. (Madry et al., 2017) + FETRA	RN50	$L_\infty, \epsilon = 4/255$	1.0	6000
Salman et al. (Salman et al., 2020) + FETRA	WRN50-2	$L_\infty, \epsilon = 4/255$	1.0	3000

Table 6. ImageNet params. In the first column, we state the method. In the next columns, we state the architecture, the trained threat model (TTM), α which is the step size and γ which is the regularization weight.

D. Transformed images

We proceed to the visualization of the TETRA transformation, supplying an intuitive explanation of what the transformation looks like and why our method works.

First, we compare our method to PGD. As shown in Figure 2, PGD transforms the image's appearance significantly in order to change its classification. TETRA also modifies the image appearance to the target class, however, it keeps the transformed image pixel-wise close to the input image. The transformation is performed in a rather artistic way, as it is almost imperceptible that the *toucan* appears on the *lorikeet*'s wing.

TETRA relies on the hypothesis that the extent of the image modification relates to the probability of belonging to a certain class. Therefore, it is important to supply visual evidence and intuition. To this end, we present Figure 3, where we demonstrate TETRA transformation towards multiple classes. In the first column, we present the clean image. In the following four columns, we transform the image into target classes. First, to the true class, then to a similar class, and finally to two entirely different class categories. This figure emphasizes that a transformation of an image to the true class, or similar ones, requires minor changes. However, when transforming an image into a different category, the image is modulated considerably by adding some features belonging to the target class.

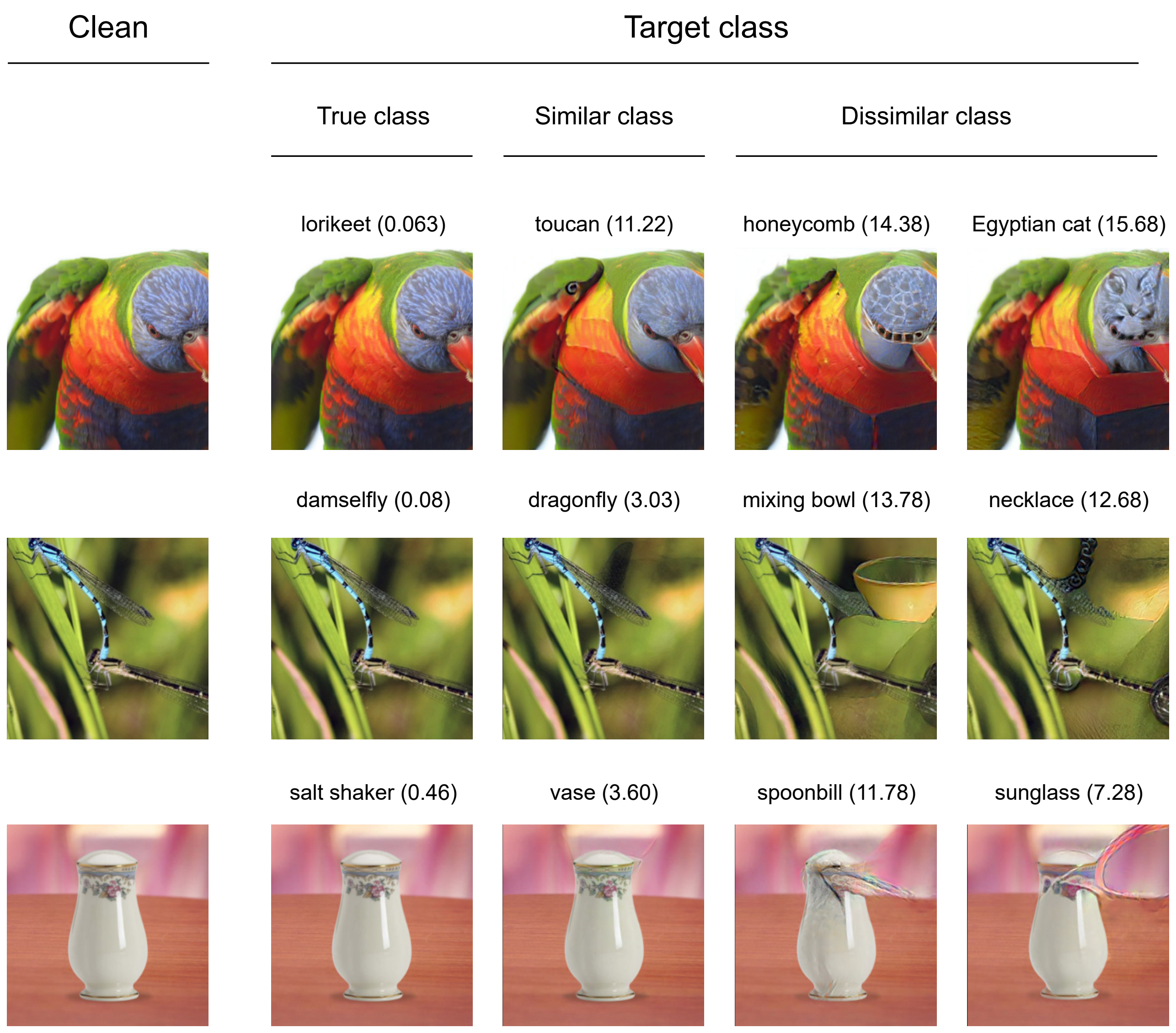

Figure 3. Visualization of TETRA's transformation. Clean images from the ImageNet dataset (Deng et al., 2009) (left column) are transformed by TETRA into (left-to-right): the true class, a class from the same category, and two classes from different categories. Pixel-level distance is more noticeable as the perceptual distance to the target class grows (presented in the images' titles as the ℓ_2 distance).

E. RPGD analysis

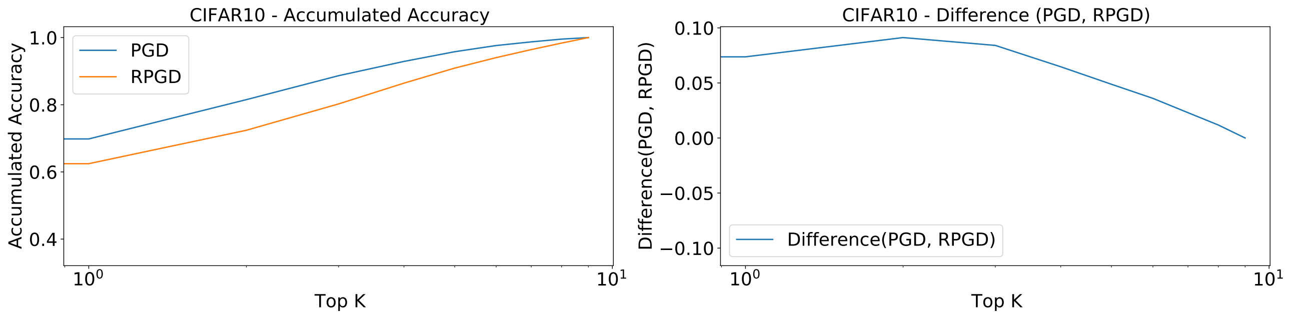

Figure 4. CIFAR10 top k accuracy comparison PGD vs RPGD. the x-axis of the left figure represents the top k group size that we select, using Madry et al. (Madry et al., 2017) $\ell_2, \epsilon = 0.5$. The y-axis represents the top k accuracy, the probability that the true label is contained in the top k group. On the right figure, we present the difference between the two graphs of the left figure, $PGD - RPGD$.

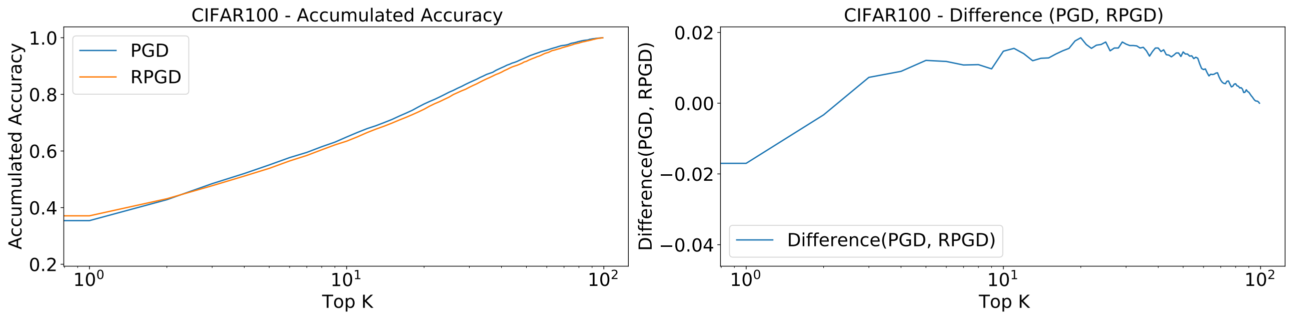

Figure 5. CIFAR100 top k accuracy comparison PGD vs RPGD. the x-axis of the left figure represents the top k group size that we select, using Madry et al. (Rebuffi et al., 2021) $\ell_\infty, \epsilon = 8/255$. The y-axis represents the top k accuracy, the probability that the true label is contained in the top k group. On the right figure, we present the difference between the two graphs of the left figure, $PGD - RPGD$.

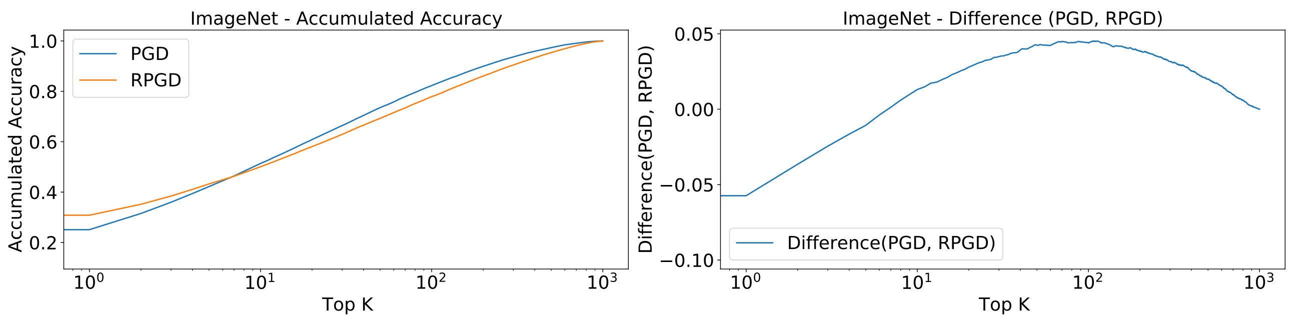

Figure 6. ImageNet top k accuracy comparison PGD vs RPGD. the x-axis of the left-hand side of the figure represents the top k group size that we select, using Madry et al. (Madry et al., 2017) $\ell_2, \epsilon = 3.0$. The y-axis represents the top k accuracy, the probability that the true label is contained in the top k group. For example, if we examine $k = 20$ it means that we seek the probability that the true label is in one of the top 20 predictions of the classifier, which is around 60%. On the right-hand side of the figure, we present the difference between the two graphs of the left-hand side, $PGD - RPGD$.

F. Ablation study

In this part, we discuss the ablations that we performed in order to better understand the contribution of different parts of our method. In Appendix F.1 we discuss the necessity of the PAG property in the TETRA algorithm, next in Appendix F.2 we discuss different options for the distance metric used for classification.

F.1. Vanila classifier

TETRA can be applied to any differentiable classifier. We claim, however, that it enhances the classifier robustness only over classifiers that possess PAG. In this part, we empirically support this claim.

In Table 7, we present TETRA accuracy on CIFAR10 dataset, where the classifier is vanilla trained. As we can see, TETRA achieves an accuracy of around 1% for all of the attacks. When applying TETRA to PAG classifiers, it achieves much better results, as presented in Table 1. Meaning that TETRA performs well only when applied to classifiers that possess the PAG property. The reason is that our method heavily relies on the generative power of PAG, which does not exist in vanilla-trained classifiers.

Method	Architecture	TTM	Standard	Attack			
				L_∞		L_2	
				$8/255$	$16/255$	0.5	1.0
Vanila	WRN28-10	None	**95.26%**	00.00%	00.00%	00.00%	00.00%
DRQ (Schwinn et al., 2022)			10.95%	**11.31%**	**11.04%**	**11.75%**	**11.38%**
TETRA			93.04%	01.11%	01.12%	01.24%	01.10%

Table 7. CIFAR10 vanilla classifier results. In the first column, we state the method. We report three consecutive lines of results. One for the base method and then two test time boosting methods: DRQ (Schwinn et al., 2022) and TETRA. In the next columns, we state the architecture, the trained threat model (TTM), and four attacks with different threat models.

F.2. Distance metrics

In TETRA's second phase, we calculate the distance between the input image and the transformed images, and we classify based on the shortest one. Hence, the distance metric that we use for the classification is important. Different metrics have different properties, and we aim at a distance metric that is able to measure the semantic distance between images.

We compare ℓ_2, ℓ_1 and LPIPS (Zhang et al., 2018) distances over CIFAR10 dataset, and presente the results in Table 8. We compare the results using the following defense methods (Madry et al., 2017; Rebuffi et al., 2021; Gowal et al., 2020). As demonstrated, ℓ_2 distance metric performs better, therefore is a favorable choice.

Method	Architecture	TTM	Standard	Attack			
				L_∞		L_2	
				$8/255$	$16/255$	0.5	1.0
AT (Madry et al., 2017)	RN50	$L_2, \epsilon = 0.5$	**90.83%**	29.04%	00.93%	69.24%	36.21%
TETRA LPIPS	RN50	$L_2, \epsilon = 0.5$	85.91%	54.49%	17.46%	78.70%	61.68%
TETRA L_1	RN50	$L_2, \epsilon = 0.5$	85.91%	**54.55%**	**17.64%**	78.70%	61.68%
TETRA L_2	RN50	$L_2, \epsilon = 0.5$	85.76%	**54.55%**	**17.64%**	**78.74%**	**61.87%**
Rebuffi et al. (Rebuffi et al., 2021)	WRN28-10	$L_2, \epsilon = 0.5$	**91.79%**	47.85%	05.00%	78.80%	54.73%
TETRA LPIPS	WRN28-10	$L_2, \epsilon = 0.5$	87.31%	58.57%	09.97%	85.30%	**67.03%**
TETRA L_1	WRN28-10	$L_2, \epsilon = 0.5$	87.31%	58.57%	09.97%	85.30%	**67.03%**
TETRA L_2	WRN28-10	$L_2, \epsilon = 0.5$	88.33%	**59.74%**	**11.06%**	**85.57%**	66.01%
Rebuffi et al. (Rebuffi et al., 2021)	WRN28-10	$L_\infty, \epsilon = 8/255$	**87.33%**	60.77%	25.44%	66.72%	35.01%
TETRA LPIPS	WRN28-10	$L_\infty, \epsilon = 8/255$	80.97%	66.49%	33.54%	74.73%	**59.25%**
TETRA L_1	WRN28-10	$L_\infty, \epsilon = 8/255$	80.97%	66.49%	33.54%	74.73%	**59.25%**
TETRA L_2	WRN28-10	$L_\infty, \epsilon = 8/255$	84.86%	**66.96%**	**35.15%**	**74.84%**	53.32%
Gowal et al. (Gowal et al., 2020)	WRN70-16	$L_\infty, \epsilon = 8/255$	**91.10%**	65.88%	25.95%	66.44%	27.22%
TETRA LPIPS	WRN70-16	$L_\infty, \epsilon = 8/255$	83.41%	71.51%	38.49%	76.53%	**58.60%**
TETRA L_1	WRN70-16	$L_\infty, \epsilon = 8/255$	83.41%	71.51%	38.49%	**76.56%**	**58.60%**
TETRA L_2	WRN70-16	$L_\infty, \epsilon = 8/255$	87.58%	**72.00%**	**40.44%**	75.65%	49.22%

Table 8. CIFAR10 results. In the first column, we state the method. For every base method, we report three consecutive lines of results. One for the base method and then two TETRA distance metric variations used for classification: L_2 and LPIPS (Zhang et al., 2018). In the next columns, we state the architecture, the trained threat model (TTM), and four attacks with different threat models.

G. Runtime analysis

In this part, we compare the inference time of the test-time methods that we used, over CIFAR10 and CIFAR100. For CIFAR10 we compare TETRA to DRQ (Schwinn et al., 2022), and for CIFAR100 we compare FETRA to DRQ (Schwinn et al., 2022). As can be seen, for both of the datasets, our method is slower than the baseline. Our method, however, is faster than DRQ (Schwinn et al., 2022). These experiments were performed using one GeForce RTX 3080 with batch size $= 1$.

Method	Architecture	TTM	Inference time
AT (Madry et al., 2017)			$\times 1$
DRQ (Schwinn et al., 2022)	RN50	$L_2, \epsilon = 0.5$	$\times 160$
TETRA			$\times 23$
Rebuffi et al. (Rebuffi et al., 2021)			$\times 1$
DRQ (Schwinn et al., 2022)	WRN28-10	$L_2, \epsilon = 0.5$	$\times 117$
TETRA			$\times 26$
Rebuffi et al. (Rebuffi et al., 2021)			$\times 1$
DRQ (Schwinn et al., 2022)	WRN28-10	$L_\infty, \epsilon = 8/255$	$\times 119$
TETRA			$\times 26$
Gowal et al. (Gowal et al., 2020)			$\times 1$
DRQ (Schwinn et al., 2022)	WRN70-16	$L_\infty, \epsilon = 8/255$	$\times 290$
TETRA			$\times 127$

Table 9. Inference time comparison over CIFAR10. In this table we perform an inference time comparison between 3 defense methods. For every base classifier, we report three consecutive lines of inference time. One for the base method, next we present DRQ, and finally TETRA. In the first column we present the method name. Next we present the architecture, and the trained threat model (TTM), and finally we present the inference time. This value stands for how much time it takes for every method to perform.

Method	Architecture	TTM	Inference time
Rebuffi et al. (Rebuffi et al., 2021)			$\times 1$
DRQ (Schwinn et al., 2022)	WRN28-10	$L_{inf}, \epsilon = 8/255$	$\times 686$
FETRA			$\times 27$
Gowal et al. (Gowal et al., 2020)			$\times 1$
DRQ (Schwinn et al., 2022)	WRN70-16	$L_\infty, \epsilon = 8/255$	$\times 1380$
FETRA			$\times 121$

Table 10. Inference time comparison over CIFAR100. In this table we perform an inference time comparison between 3 defense methods. For every base classifier, we report three consecutive lines of inference time. One for the base method, next we present DRQ, and finally TETRA. In the first column we present the method name. Next we present the architecture, and the trained threat model (TTM), and finally we present the inference time. This value stands for how much time it takes for every method to perform.