Can Perceptual Guidance Lead to Semantically Explainable Adversarial Perturbations?

Charantej Reddy Pochimireddy, Aditya T. Siripuram, Sumohana S. Channappayya *Senior Member, IEEE*

***Abstract*—It is well known that carefully crafted imperceptible perturbations can cause state-of-the-art deep learning classification models to misclassify. Understanding and analyzing these adversarial perturbations play a crucial role in the design of robust convolutional neural networks. However, their mechanics are not well understood. In this work, we attempt to understand the mechanics by systematically answering the following question: do imperceptible adversarial perturbations focus on changing the regions of the image that are important for classification? In other words, are imperceptible adversarial perturbations semantically explainable? Most current methods use l_p distance to generate and characterize the imperceptibility of the adversarial perturbations. However, since l_p distances only measure the pixel to pixel distances and do not consider the structure in the image, these methods do not provide a satisfactory answer to the above question. To address this issue, we propose a novel framework for generating adversarial perturbations by explicitly incorporating a "perceptual quality ball" constraint in our formulation. Specifically, we pose the adversarial example generation problem as a tractable convex optimization problem, with constraints taken from a mathematically amenable variant of the popular SSIM index. We use the MobileNetV2 network trained on the ImageNet dataset for our experiments. By comparing the SSIM maps generated by our method with class activation maps, we show that the perceptually guided perturbations introduce changes specifically in the regions that contribute to classification decisions i.e., these perturbations are semantically explainable.**

***Index Terms*—SSIM, Adversarial perturbations, Explainability.**

I. Introduction

THE fields of Artificial Intelligence (AI) and Machine Learning (ML) have seen tremendous growth and development in the last decade that is expected to have a major impact on humankind in the foreseeable future. This growth can largely be attributed to advances in deep learning, which in turn can be attributed to the availability of large data sets and efficient hardware. These advances allow us to accurately train highly complex deep learning models which are used in a myriad of application areas, including computer vision, speech and audio processing, natural language processing, medical imaging, and finance, to name a few.

In theory, neural networks can represent/approximate any real-valued function (or at least a wide variety of functions) when appropriate weights and architecture are chosen according to the *Universal Approximation Theorem* [1]. With sufficient data and the right choice of architecture, one can achieve state-of-the-art results on a variety of machine learning problems using deep neural networks.

Analyzing the robustness of deep learning models to adversarial inputs has received significant attention over the past few years. A majority of these works can be classified into four broad areas: 1) generating adversarial examples/crafting adversarial perturbations, 2) defending against such 'attacks' [2], [3], 3) evaluation and certification [4], [5], and 4) interpretability of adversarial examples/perturbations [6]. However, we still lack a clear understanding of the key underlying factors for the existence of this phenomenon. Because of this, it is challenging to design defence mechanisms that work for all kinds of adversarial attacks/perturbations. The adversarial example generation problem is highly non-convex, and heuristic or other approximation techniques are required to solve it.

In their seminal work, Szegedy et al. introduced adversarial examples in [7] that are crafted by solving an optimization. Later, faster adversarial perturbation methods with closed-form solution (direct expression to generate perturbations) were proposed by using the norm bounded constraints; examples include FGSM (l_∞ bounded perturbations) [8], FGM (l_2 bounded perturbations) [9], and PGD (l_∞ and l_2 bounded perturbations). Several attack and defense algorithms have since been proposed, and many of these methods are summarized in [2], [3]. Most existing defences are not completely robust: adversarial attacks/perturbations can be specifically engineered to target the said defense technique. This has led to a self-sustaining cycle - with attacks leading to defenses, and defenses paving the way for new attacks. Despite the many results in this area, a complete understanding of the adversarial perturbation landscape is still lacking. Some proposed hypotheses for explaining adversarial perturbations include high dimensional spaces [8], data in-completion, model capacity [2], and the presence of highly predictive but non-robust features (spurious correlations) [6]. The authors in [10] proposed that the distribution shift between training and test data set, combined with the high dimensional continuous data space, as the key reasons for adversarial examples.

Along these lines, our interest is not to create another attack but rather to understand the adversarial landscape, primarily with respect to perceptual similarity. Our investigation aims to systematically explore the semantic significance of the regions affected by adversarial perturbations. In particular, we try to answer the following question: *do imperceptible adversarial perturbations focus on changing the regions of the image that are important for classification? In other words, are imperceptible adversarial perturbations semantically explainable?*

In most of the adversarial example attacks, l_p distance metrics are used for crafting the adversarial perturbations. However, it is well-known that l_p distance metrics are not good at measuring perceptual similarity between two images

[11], and that these metrics are neither necessary nor sufficient for perceptual similarity [12]. For example, a mean shift in the image results in a high l_p distance, but perceptually, the mean shifted image remains close to the original image. For the perturbations to be imperceptible, the original and modified image must be similar in some metric that respects structural (perceptual) similarity. Thus to answer the question above, we use the Structural Similarity (SSIM) index [11] to generate the adversarial perturbations instead of l_p distance metrics. The SSIM index is a popular measure to evaluate the perceptual similarity/quality of images. It is a full-reference image quality metric that measures the quality of a distorted image with respect to the ground-truth pristine image. Compared to traditional l_p distance metrics, the SSIM index is known to correlate better with the human perception of image quality/distortion. The following are the main contributions of this work:

1) In order to address the question posed earlier, we propose a perceptually guided adversarial example generation technique by leveraging the useful mathematical properties of the SSIM index. The SSIM metric is analyzed in depth in order to construct a novel convex formulation (which we call perceptually guided adversarial perturbation (PGAP)) to generate the adversarial perturbations. We also provide a closed-form approximation (called Faster PGAP (FPGAP)) to the proposed convex problem.
2) We then try to systematically investigate (both qualitatively and quantitatively) adversarial examples and their relationship to semantically significant regions of the image. We do this by comparing the perturbations generated using the SSIM index with class activation maps generated using GradCAM++ [13]. We quantify this comparison using IOU-based metrics and precision; and observe that compared to other norm-bounded approaches, our method gives about a factor of 2-3 higher precision scores (Fig 5).
3) We thus conclusively answer the question posed in the introduction. Adversarial perturbations generated by incorporating an SSIM ball constraint (instead of the l_p ball constraint as in other works) seem to be changing only the regions of the image significant for classification, at least to a degree much higher than other norm-bounded attack methods.
4) In order to validate our adversarial example generation method, we also establish that our method gives a much higher fooling rate at a given average SSIM compared to other similar methods.

We find it very intriguing that a *perceptually-aware* formulation makes the adversarial example generation *semantically-aware*.

II. Related work

The SSIM index has been considered previously in the adversarial perturbation setting, and we briefly summarize two relevant methods next. In [14], the authors proposed stronger attacks by combining different types of attacks (adding adversarial noise, rotating, translating, or performing spatial transformations on images) and used the SSIM index to quantify the strength of the adversarial attack. In [15], the authors introduced a new measure called Perceptual Adversarial Similarity Score (PASS) using the SSIM index to quantify the adversarial examples and use the PASS in the process of generating adversarial examples.

Our contribution differs significantly from both of these methods, which we outline below. By imposing constraints from a mathematically amenable variant of the SSIM index [16] (and exploiting useful properties like quasi convexity), and by taking a suitable approximation of the loss function, we pose the adversarial example generation problem as a quadratically constrained quadratic program (QCQP). We also provide a closed-form solution to the approximation of the QCQP that allows for faster implementation. In fact, none of the existing adversarial example techniques that use image quality metrics provide a closed-form solution to (1). We not only give a convex formulation (9), but also provide a closed-form approximation (10). Thus our method is in line with other norm bounded techniques (PGD, FGSM, etc.), which employ a closed-form solution to generate adversarial perturbations. The perturbations we generate are not necessarily additive, nor are they obtained by a parameterized rotation or spatial transformation; we propose a model-free technique to generate adversarial examples that are structurally and thereby perceptually similar to the original image.

While this manuscript was in development, we also became aware of another line of work [17] that uses SSIM to generate adversarial examples. However, our work differs in the following key aspects: First, our goal, as opposed to the work in [17], is to investigate the question posed in the introduction, and not simply to generate another attack. As such, we have a detailed and systematic analysis of the regions of perturbations as compared to the regions identified by GradCAM++, which are not present in [17]. Secondly, we provide a tractable convex formulation of the adversarial example generation problem. We use the analysis from [16] primarily to achieve this goal. Thirdly, in addition to the convex formulation, we also provide a fast approximation to the convex formulation (FPGAP), which brings our technique (computationally and structurally) in line with existing norm-bounded methods. This is in contrast to the gradient-descent-based Lagrangian optimization used in [17] (as a consequence, our method also does not need any additional hyper parameters). This fast approximation also allows us to perform experiments on datasets like ImageNet, which would not be possible with the initial formulation. Indeed such experiments are not done in [17]. Finally, we also provide a systematic analysis of the comparative impact of the hyper-parameters involved in our algorithm (ϵ_1 and ϵ_2 in (8)).

A. Organization

The paper is organized as follows: in section III we briefly discuss the standard norm bounded adversarial example generation methods and formulate an optimization problem (2). In section III-A, we explain the SSIM index along with its mathematically amenable variant and discuss some of its

properties which will be used to generate adversarial examples using SSIM constraints. In section IV, we use ideas from III-A to modify (2) to formulate an optimization problem (9) to generate adversarial examples using SSIM index. We also propose an approximate solution (10) to this optimization problem which enables the faster generation of adversarial examples. In section V, we analyze the norm bounded methods along with the proposed method qualitatively and quantitatively and present some additional results on the perceptual quality of adversarial examples generated by different methods. We conclude the paper with some closing remarks in section VI.

III. Problem Setup

Suppose $\mathcal{D}$ represents the data set with entries (x_i, y_i), where x_i represents the i^{th} data point and y_i represents the corresponding label. Let f denote the machine learning model and its prediction $\hat{y}$ (i.e., $\hat{y}_i = f(x_i)$). In the supervised learning framework we try to minimize the loss function $\mathcal{L}$ with respect to given data set $\mathcal{D}$ and find the best suited model parameters w.

$$\min_{w} \quad \sum_{i} \mathcal{L}(w, x_i, y_i).$$

An adversarial example x_{adv} is similar (in some metric) to a data point x in the data set $\mathcal{D}$, such that the machine learning models misclassifies the input (i.e., $f(x_{adv}) \neq f(x)$). Finding an adversarial example at a given data point x with label y can be formulated as an optimization problem based on loss function: In this formulation, we find x_{adv} close to a given data point x, which maximizes the loss function

$$\underset{x_{adv}}{\operatorname{argmax}} \ \mathcal{L}(w, x_{adv}, y) \quad \text{s.t.} \quad d(x, x_{adv}) \leq \epsilon. \tag{1}$$

Here ϵ is the allowed level of perturbation. We refer to x_{adv} as adversarial example and $x - x_{adv}$ as adversarial perturbation.

Note that the feasibility region of the optimization problem above (1) varies based on the value of ϵ. The formulation in (1) is similar to the inner maximization problem from [18]. The distance metric $d(x, x_{adv})$ is typically an l_p distance

$$d(x, x_{adv}) = ||x - x_{adv}||_p \text{ where } ||a||_p = \left(\sum_i |a_i|^p \right)^{\frac{1}{p}}.$$

Typically, a linear approximation of the loss function around the data point x is used:

$$\mathcal{L}(w, x_{adv}, y) \approx \mathcal{L}(w, x, y) + (x_{adv} - x)^T \nabla_x \mathcal{L}(w, x, y).$$

So to find x_{adv} we maximize the second term above. Note that since

$$\underset{x_{adv}}{\operatorname{argmax}} \ \mathcal{L}(w, x_{adv}, y) \approx \underset{x_{adv}}{\operatorname{argmax}} (x_{adv})^T \nabla_x \mathcal{L}(w, x, y),$$

we can just maximize the inner product between x_{adv} and $\nabla_x \mathcal{L}(w, x, y)$. Also note that the gradients $\nabla_x \mathcal{L}(w, x, y)$ can be readily extracted from the model.

Important distance metrics used in the literature are l_∞, which measures the maximum absolute change in the pixel values [8], [19], [20]; l_2 which measures the Euclidean distance of change in the pixel values [20], [9]; l_1 measures the total absolute change in the pixel values [21], and l_0 measures the number of pixels that differ [20], [22], [23].

As discussed earlier, l_p distance metrics are not good at measuring perceptual similarity between two images. If we consider

$$d(x, x_{adv}) = \sqrt{1 - \text{SSIM}(x, x_{adv})},$$

then the optimization problem for generating adversarial example becomes:

$$\begin{aligned} \underset{x_{adv}}{\operatorname{argmax}} \quad & (x_{adv})^T \nabla_x \mathcal{L}(w, x, y) \\ \text{s.t.} \quad & \text{SSIM}(x, x_{adv}) \geq 1 - \epsilon^2 \end{aligned} \tag{2}$$

However, the metric $d(x, x_{adv}) = \sqrt{1 - \text{SSIM}}$ defined above is non-convex, and so the above problem can become intractable. So we use a variant of the SSIM index [16] to generate imperceptible adversarial perturbations. We review these ideas next. Readers familiar with the SSIM index can skip to Section IV.

A. Structural Similarity (SSIM) Index

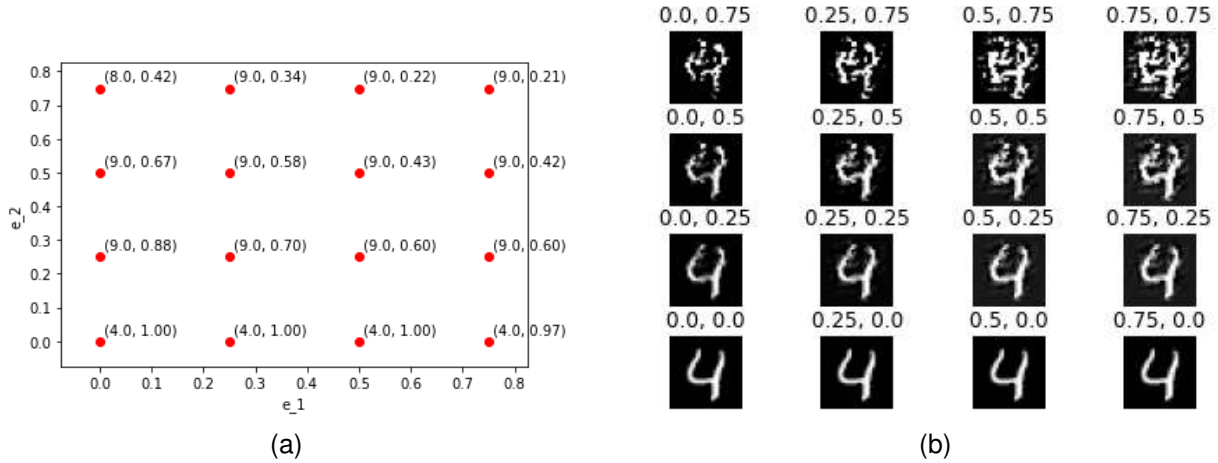
The SSIM index between the two image patches X and Y is computed using a combination of three distortion measurement components: luminance (l), contrast (c), and structure/correlation (s), that are defined as follows.

$$l = \frac{2\mu_X \mu_Y + c_1}{\mu_x^2 + \mu_Y^2 + c_1}, \quad c = \frac{s_X s_Y + c_2}{s_X^2 + s_Y^2 + c_2} \quad s = \frac{s_{X,Y} + c_3}{s_X s_Y + c_3},$$

where μ_X, μ_Y represent the mean of X and Y respectively, s_X^2, s_Y^2 represent the variances of X and Y respectively, and $s_{X,Y}$ represents the co-variance between the X and Y. Here, c_1, c_2 and c_3 are small numerical constants that ensure numerical stability when the denominators are close to zero. We can also say that these constants aim to characterize the saturation effects of the visual system at low luminance and contrast regions. The first two terms l and c measure nonstructural distortion, while the last term s measures structural distortion (or absence of correlation) between the two images. The structural similarity or SSIM between the images X and Y is defined as the product of the luminance, contrast and structure terms defined above, i.e., $\text{SSIM}(X, Y) = l.c.s.$

An SSIM quality map is constructed by computing the SSIM index between pairs of corresponding local patches in the two images, and the overall SSIM index is computed by averaging the patch level values in the SSIM map.

While the SSIM index is indeed a better method compared to MSE for measuring perceptual similarity between two images, it does not satisfy the triangle inequality and thus is not a distance metric, limiting its use in convex problem formulations. However, the SSIM index can be converted to a normalized root mean square error (NRMSE) measure, which is a valid distance metric [16]. The square of such a metric is not convex but is locally convex, and quasi-convex [16], thereby making the SSIM index a feasible target for optimization. We use these insights for our problem formulation. Next, we briefly review these ideas, develop the notation.



(a) (b)

Fig. 1. Image on the left: x-axis and y- axis correspond to ϵ_1 and ϵ_2 respectively, at each highlighted point on the grid has label with first co-ordinate as model prediction and second co-ordinate as ssim value. Image on the right: corresponding images with ϵ_1 and ϵ_2 mentioned on top of them.

In the standard form of the SSIM index, we set the numerical constants $c_3 = c_2/2$, resulting in the SSIM having only two terms

$$\text{SSIM}(X,Y) = S_1(X,Y) \;\; S_2(X,Y), \tag{3}$$

where $S_1 = l$ and

$$S_2 = c.s = (2s_{X,Y} + c_2) \,/\, \left(s_X^2 + s_Y^2 + c_2\right).$$

It can be seen that $\sqrt{1-\text{SSIM}}$ is not a metric but $\sqrt{1-S_1}$ and $\sqrt{1-S_2}$ are normalized metrics [16]. Now we set $d_1 = \sqrt{1-S_1}$ and $d_2 = \sqrt{1-S_2}$, and the vector $d = [d_1, d_2]$. It can be seen that d is a vector of normalized metrics obtained from the root mean square error [16].

The SSIM index can be approximated with the vector of metrics $d(X,Y)$ as

$$||d(X,Y)||_2 = \sqrt{(d_1)^2 + (d_2)^2} = \sqrt{2 - S_1 - S_2}. \tag{4}$$

We note that

$$\sqrt{1-\text{SSIM}} = \sqrt{1 - S_1 S_2} = \sqrt{d_1^2 + d_2^2 - d_1^2 d_2^2}. \tag{5}$$

We observe that $||d(X,Y)||_2$ serves as a lower order approximation of $\sqrt{1-\text{SSIM}}$. We can also write

$$\begin{aligned} S_1 &= 1 - \text{NMSE}(\mu_X, \mu_Y, c_1) \\ S_2 &= 1 - \text{NMSE}(X - \mu_X, Y - \mu_Y, c_2), \end{aligned} \tag{6}$$

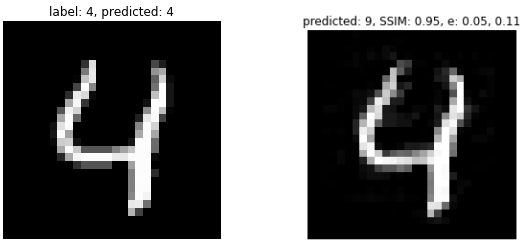
where NMSE is the normalized mean squared error given by

$$\text{NMSE}(X,Y,c) = \frac{||X-Y||^2}{||X||^2 + ||Y||^2 + c}. \tag{7}$$

We use these ideas to modify problem (2).

IV. Adversarial example generation using SSIM

The structural similarly index can be written as a product of two terms i.e., SSIM $= S_1.S_2$ where S_1 captures the luminance similarity, and S_2 captures the structure and contrast similarity. From [16] (summarized in Section III-A above) we know that S_1 and S_2 have appealing convexity properties.



(a) (b)

Fig. 2. Image on the left: original test image with label $y = 4$ and model prediction $\hat{y} = 4$. Image on the right: adversarial example generated by solving (8) ($\epsilon_1 = 0.05$ and $\epsilon_2 = 0.11$) with perceptual quality 0.95 (SSIM).

Going back to the problem formulation of (2), we see that the condition on SSIM(x, x_{adv}) in equation (2) can be replaced by conditions on $S_1(x, x_{adv})$ and $S_2(x, x_{adv})$, as discussed in (3). By choosing ϵ_1 and ϵ_2 suitably, we may rewrite the optimization problem from (2) as

$$\begin{aligned} \underset{x_{adv}}{\text{argmax}} \quad & (x_{adv})^T \nabla_x \mathcal{L}(w,x,y) \\ \text{s.t.} \quad & S_1(x, x_{adv}) \geq 1 - \epsilon_1^2 \\ & S_2(x, x_{adv}) \geq 1 - \epsilon_2^2 \end{aligned} \tag{8}$$

We analyse the constraints of (8) in more detail in Appendix A. The first constraint $S_1(x, x_{adv}) \geq 1 - \epsilon_1^2$ is a linear constraint (this forces x_{adv} to lie in an intersection of two half-spaces). The second constraint in (8) $S_2(x, x_{adv}) \geq 1 - \epsilon_2^2$ is a quadratic constraint (this forces x_{adv} to be in a high dimensional sphere). Based on the constraints (12), (13), and the objective, the optimization problem in (8) is convex; in particular it is a Quadratically Constrained Quadratic Program (QCQP).

Conceptually, Constraint 1 corresponds to non-structural perceptual features (luminance), and Constraint 2 corresponds to structural/perceptual features in the image. The parameters ϵ_1 and ϵ_2 fix the allowed tolerances in these features and decide the feasibility region of the optimization problem.

To understand the relative impact of ϵ_1 and ϵ_2, we take an image with label $y = 4$ from the MNIST digits test data set and generate adversarial examples for different values of ϵ_1 and ϵ_2. For this exercise, we use a CNN model, which has around 99% accuracy on the test data. In Figure 1, the image on the left shows the model prediction of optimization problem output and its SSIM index with respect to original image, as a function of ϵ_1 and ϵ_2. On the right, the corresponding output images are shown. From Figure 1 one important observation that can be made is that the impact of ϵ_2 (Constraint 2) is more on the solution (model prediction and perceptual quality of optimization output) compared to the ϵ_1 (Constraint 1). This observation can also be validated by calculating the corresponding optimal dual variables. For the above example in Figure 2, the dual variable corresponding to Constraint 2 is around 145 times larger compared to the dual variable of Constraint 1. Since ϵ_2 has significantly higher impact compared to ϵ_1, the algorithm can be simplified by taking $\epsilon_1 = 0$, effectively removing the Constraint 1 above. This modification results in solving the following optimization problem:

$$\begin{aligned} \underset{x_{adv}}{\text{argmax}} \quad & (x_{adv})^T \nabla_x \mathcal{L}(w, x, y) \\ \text{s.t.} \quad & \mu_{x_{adv}} = \mu_x \\ & S_2(x, x_{adv}) \geq 1 - \epsilon_2^2 \end{aligned} \tag{9}$$

A. Proposed Method – Perceptually Guided Adversarial Perturbation (PGAP)

In practice, solving (9) may not lead to an adversarial example. This is because the formulation of (8) assumed a linear approximation to the loss function, which may be accurate. Along the lines of [18], we propose an iterative technique that repeatedly solves (9). We first fix an ϵ_2 and solve (9). Note that at this point the obtained solution x_{adv} may not be adversarial (i.e., we may not have $f(x_{adv}) \neq f(x)$). We recalculate the gradients at the obtained x_{adv} and solve (9) using the updated gradient. This process is repeated until an adversarial example is found (see Algorithm 1 for a summary). The iterative approach helps in overcoming the limitations imposed by the linear approximation to the loss function.

One drawback of the proposed method (Algorithm 1) is that the QCQP is slow to solve on large datasets (for e.g., ImageNet). Hence next, we present next a faster algorithm that uses an efficient approximation of the solution to (9).

B. Approximate Solution – Faster PGAP

We formulate an equivalent optimization problem from (9) by relaxing Constraint 1 and substituting it in Constraint 2, and converting it into a minimization problem. The solution to this problem is given by:

$$x_{adv} = 1(\mu_x) + k_{22} + (\sqrt{k_{21}}) \left(\frac{\nabla_x \mathcal{L}(w, x, y)}{||\nabla_x \mathcal{L}(w, x, y)||} \right), \tag{10}$$

where k_{21}, k_{22} are defined in (13) and 1 in $1(\mu_x)$ is all ones of size x_{adv}; thus providing a closed form solution to (14). We refer the reader to Appendix B for the intermediate steps.

Algorithm 1 Adversarial example generation

PGAP(x, model, label, ϵ_2, iterNum)
- $i = 0$
- **While** $i \leq$ iterNum **do**
- $x_{adv} \leftarrow \underset{x_{adv}}{\text{argmax}} \ (x_{adv})^T \nabla_x \mathcal{L}(w, x, y)$
  - s.t. $\mu_{x_{adv}} = \mu_x, S_2(x_{adv}, x) \geq 1 - \epsilon_2^2$
- $y_{adv} \leftarrow$ model.predict(x_{adv})
- **If** label $\neq y_{adv}$
  - **return** x_{adv}
- **else**
  - $x = x_{adv}$
- **return** x_{adv}

Algorithm 2 presents the steps of FPGAP (a fast approximate variant of PGAP) by incorporating the closed-form solution above in the iterations.

Algorithm 2 Faster adversarial example generation

FPGAP(x, model, label, ϵ_2, iterNum)
- $i = 0$
- **While** $i \leq$ iterNum **do**
- $x_{adv} \leftarrow 1(\mu_x) + k_{22} + (\sqrt{k_{21}}) \left(\frac{\nabla_x \mathcal{L}(w, x, y)}{||\nabla_x \mathcal{L}(w, x, y)||} \right)$
- $y_{adv} \leftarrow$ model.predict(x_{adv})
- **If** label $\neq y_{adv}$
  - **return** x_{adv}
- **else**
  - $x = x_{adv}$
- **return** x_{adv}

C. Some remarks

We would like to point out that the structure of the presented approximate (10) solution is very similar to adversarial examples generated by FGM [9] attack, which generates l_2 norm bounded additive perturbations. The expression for adversarial examples given by FGM is

$$x_{adv} = x + \epsilon \left(\frac{\nabla_x \mathcal{L}(w, x, y)}{||\nabla_x \mathcal{L}(w, x, y)||} \right). \tag{11}$$

Both the formulations (10) and (11) use normalized gradient $\nabla_x \mathcal{L}(w, x, y)/||\nabla_x \mathcal{L}(w, x, y)||$. However the significant difference in the performance of the proposed attack is due to the presence of the terms k_{21} and k_{22} which are derived from the structure term in the SSIM index. It is very intriguing that such a simple change results in a huge improvement in the precision scores (Section V-B) and fooling rate (Section V-C).

An alternate approach to find adversarial examples with high SSIM could be to subtract a differentiable version of SSIM from the loss function, similar to other such approaches in literature [24]. Note that the existing norm bounded approaches impose a norm ball constraint on the perturbation. This allows for the intermediate formulation (as in equation 1) to be interpreted as a dual norm, hence enabling a closed-form solution. This would not be possible if the norm is

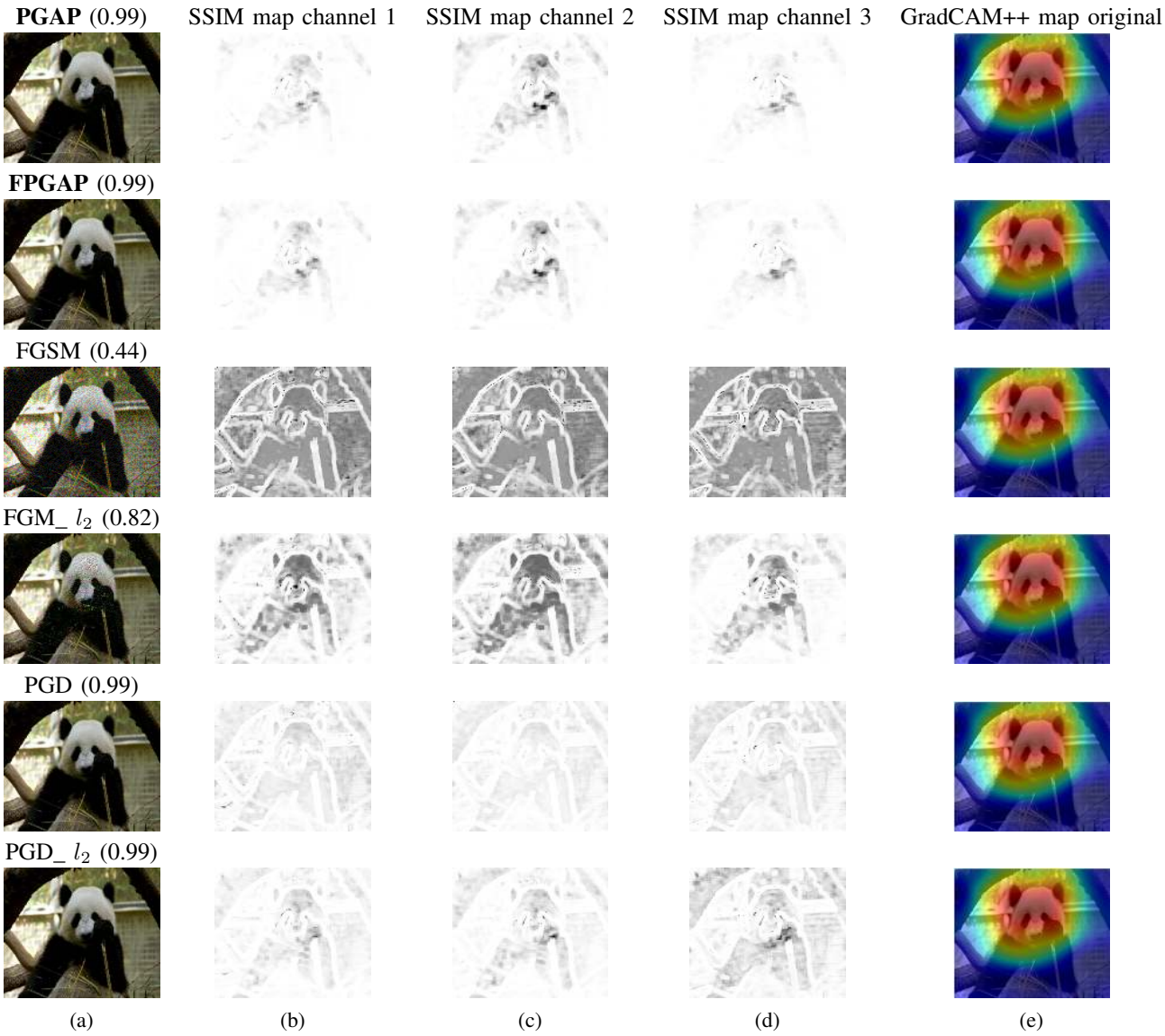


Fig. 3. SSIM maps comparison of adversarial examples generated. (a): Adversarial perturbations with different methods and SSIM index value (rounded off to two decimal places), (b),(c) and (d): SSIM maps of RGB channels respectively, (e): GradCAM++ output of original image.

added directly to the loss function instead. Our formulation follows a similar philosophy, where we impose a perceptual similarity constraint instead of subtracting the SSIM from the loss function. Specifically, the mathematically amenable variant of SSIM allows us to derive a closed-form solution in this perceptually guided setting as well. As in the l_p norm case, adding the SSIM index directly to the loss term does not admit a closed-form solution. In addition, the resulting non-convex formulation may also lead to increased complexity and convergence-related issues.

V. Results and Discussion

As discussed earlier, our key goal is to analyze the image regions which the adversarial perturbations affect the most. In particular, our interest is more on understanding the impact of adversarial perturbations on the structurally (perceptually) important regions of the image. We first start with a qualitative analysis with some illustrative examples and then quantify our observations using some well-known metrics.

1) *Model used:* We use the MobileNetV2 [25] architecture with pre-trained weights, which has 72% top-1 accuracy on the ImageNet validation set [26] for quantitative and qualitative analysis. All the algorithms (PGAP, FPGAP, and the competing methods) are applied to this model to generate adversarial perturbations, one for each image in the ImageNet validation set.
2) *Methods being compared:* We compare our methods (PGAP, FPGAP) with other popular norm-bounded techniques-FGSM (l_∞ bounded perturbations) [8], FGM (l_2 bounded perturbations) [9], and PGD (l_∞ and l_2 bounded perturbations) [18].
3) *Techniques used:* For each of the methods (FPGAP, FGSM, etc.), we generate the *SSIM maps* between the adversarial examples and original images. As discussed in section III-A, SSIM maps [11] provides a visualiza-

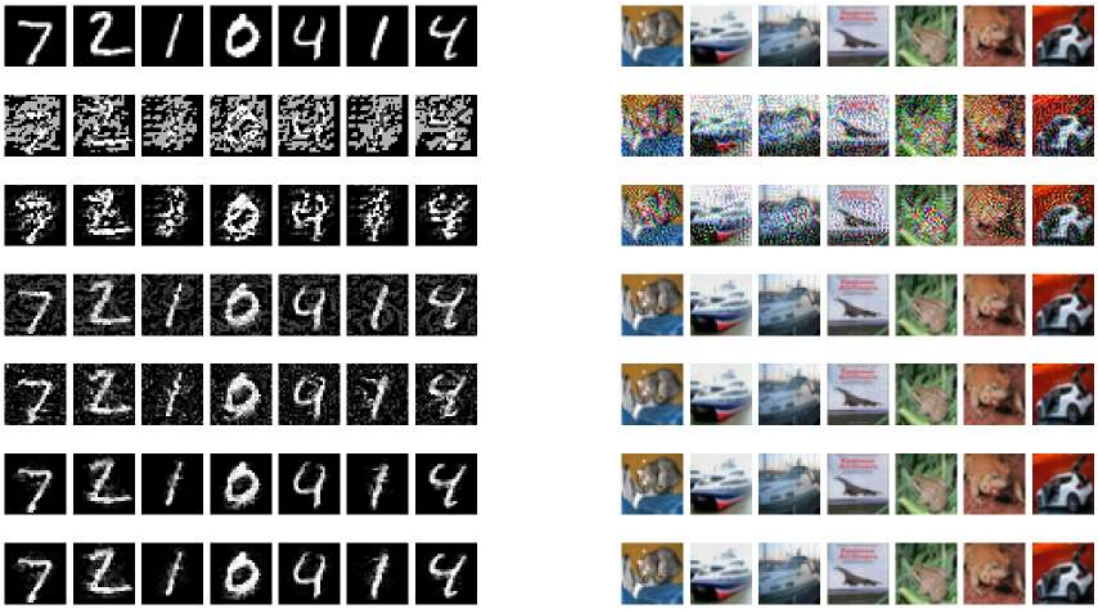


Fig. 4. Adversarial examples generated using multiple methods with similar fooling rate on images from the MNIST (left panel) and CIFAR-10 (right panel) datasets. From top to bottom each row corresponds to test images, FGSM, FGM_$l2$, PGD, PGD_$l2$, PGAP and FPGAP respectively.

tion of the distortions in the test image with respect to the reference image. Lighter regions in these maps correspond to lower distortion, while darker regions correspond to higher distortion levels.
In order to obtain the semantically important regions of an image, we use Class Activation Maps. In particular, we use GradCAM++ [13]. The heat maps generated by GradCAM++ provide spatial regions that are most important for classification.

4) *Hyper-parameter tuning:* Considering that each of the involved algorithms has different hyperparameters, we propose the following setup to compare them on an even ground. Consider the following Fooling Rate (FR) score:

$$\frac{1}{N}\sum_{i=1}^{N}\mathbb{1}\left(\text{pre-attack label}(i) \neq \text{post-attack label}(i)\right),$$

where N is the number of images in the dataset. For a fair comparison, we select the parameters for different methods such that the Fooling Rate is ≈ 1.

A. Qualitative Analysis

We first provide some illustrative examples that compare the SSIM maps generated by all the methods with GradCAM++ heat maps. For instance, consider Figure 3, we see that locations of the distortion in the proposed methods (PGAP, FPGAP) agree very well with the GradCAM++ maps. In particular, the number of distortions generated *outside* the GradCAM++ regions of interest is very few. This, in turn provides evidence that our method does indeed perturb the regions in the image important for classification decisions. We can also observe (from the much the lighter maps corresponding to the proposed methods) that the proposed methods introduce far lesser perceptual distortions (or are more localized) compared to other methods.

Similar observation can be made from Figure 4 that illustrates performance on the MNIST and CIFAR-10 datasets. On MNIST data, the proposed method introduces perturbations in and around the digit present in the image, whereas the perturbations added by the other methods are distributed across the image. On the natural images in the CIFAR-10 dataset, our method is able to carefully identify regions important for classification decisions and introduces perturbations only in such regions.

B. Quantitative Analysis

In this section, we systematically quantify the observations made in the previous section. For this, we consider the following maps

- *Noise map corresponding the adversarial examples:* For each image i in the dataset, we generate a binary image N_i given by

$$N_i = \{(1 - \text{SSIM map}(i)) > \texttt{noise threshold}\},$$

we evaluate each of the algorithms being compared at various values of `noise threshold`.
- *Noise map corresponding to GradCAM++:* For each image i in the dataset, we generate the binary image G_i given by

$$G_i = \{\text{GradCAM++ output}(i)) > 0.75\}.$$

The binary image N_i gives the regions affected by the adversarial perturbations, and the binary image G_i gives the regions that are important for classification. We would like to note here

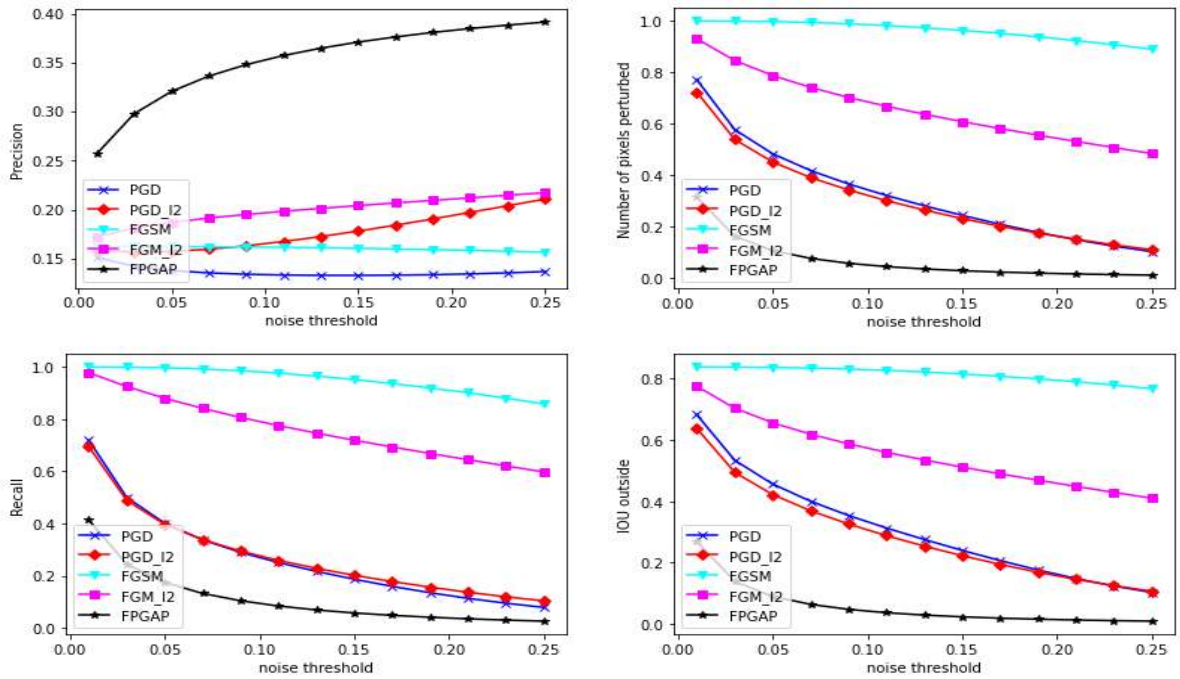


Fig. 5. Quantitative evaluation of adversarial perturbations using IOU based metrics (Avg precision, Avg number of pixels, Avg recall, Avg IOU outside) on ImageNet dataset.

that the GradCAM++ scale is normalized to lie between 0 and 1, with regions contributing to the classification decision taking on higher values. We empirically fixed the threshold on GradCAM++ to 0.75 to extract these important regions for classification (for example, the regions that are highlighted red in GradCAM++ in Figure 3).

By considering the entries in G_i to be the ground truth, we can use traditional metrics such as precision and IOU (intersection over union). We first compute the True Positives ($\texttt{TP}_i$), this is the number of positive pixels in N_i that are also positive in G_i, and the number of false positives ($\texttt{FP}_i$), which is the number of pixels positive in N_i but not in G_i. Similarly, we compute the true negatives ($\texttt{TN}_i$) and the false negatives ($\texttt{FN}_i$).

$$\texttt{TP}_i = \text{sum}(N_i \wedge G_i) \qquad \texttt{FP}_i = \text{sum}(N_i \wedge \neg G_i)$$
$$\texttt{FN}_i = \text{sum}(\neg N_i \wedge G_i) \qquad \texttt{TN}_i = \text{sum}(\neg N_i \wedge \neg G_i).$$

Based on these, we evaluate the algorithms on the following metrics

1) *Precision:* the ratio of number of pixels perturbed in the important regions to the total number of perturbed pixels.

$$\text{Precision} = \frac{1}{N}\sum_i \texttt{TP}_i/(\texttt{TP}_i + \texttt{FP}_i),$$

where N is the total number of images in the dataset. High values of this metric implies that the perturbations are precise in targeting the regions that are sensitive for classification.

2) *Number of pixels perturbed:* This is the fraction of the number of positives in N_i, averaged over the dataset.

3) *Recall* ($= \sum_i \texttt{TP}_i/N(\texttt{TP}_i + \texttt{FN}_i)$ gives the ratio of number of pixels perturbed in the important regions to the total number of pixels in the important regions. Note that this metric may not be relevant enough for our setup: as an example, consider a trivial case where the perturbation perturbs all the pixels; in this case, the recall will be high, even though the perturbation is completely agnostic to the structure of the image. Nevertheless, we include this metric for completion.

4) *Intersection over union (IOU)* Similar to other such metrics for image segmentation we can use

$$\text{IOU} = \frac{1}{N}\sum_i \frac{\texttt{TP}_i}{\texttt{FP}_i + \texttt{TP}_i + \texttt{FN}_i} = \frac{1}{N}\sum_i \text{IOU}(N_i, G_i).$$

This is also the ratio of the intersection of the regions of N_i and G_i to their union. This metric has similar issues to recall, discussed above. To over come this, we define *Intersection over union outside (IOU outside)*. Consider the following:

$$\text{IOU outside} = \frac{1}{N}\sum_i \texttt{FP}_i/(\texttt{TP}_i + \texttt{FP}_i + \texttt{TN}_i)$$
$$= \frac{1}{N}\sum_i \text{IOU}(N_i, \neg G_i).$$

This gives us the amount of perturbation the adversarial example generation method introduces in the regions that are not important for classification (i.e. outside G_i) averaged on entire data set. Ideally, if a perturbation only targets the important regions (G_i), the value of IOU outside should be very small.

For each of the algorithms being compared, these scores are calculated for many different values of `noise threshold`

TABLE I
QUANTITATIVE EVALUATION OF ADVERSARIAL PERTURBATIONS USING IOU BASED METRICS ON IMAGENET DATASET AT NOISE THRESHOLD = 0.11.

Attack (ϵ)	Precision	Number of pixels perturbed	Recall	IOU outside
FGSM (0.5)	0.1613	0.9817	0.9759	0.8263
FGM_*l*2 (100)	0.1982	0.6682	0.7750	0.5594
PGD (0.03)	0.1329	0.3217	0.2503	0.3128
PGD_*l*2 (5.0)	0.1672	0.3017	0.2585	0.2880
FPGAP (0.005)	0.3577	0.0454	0.0837	0.0369

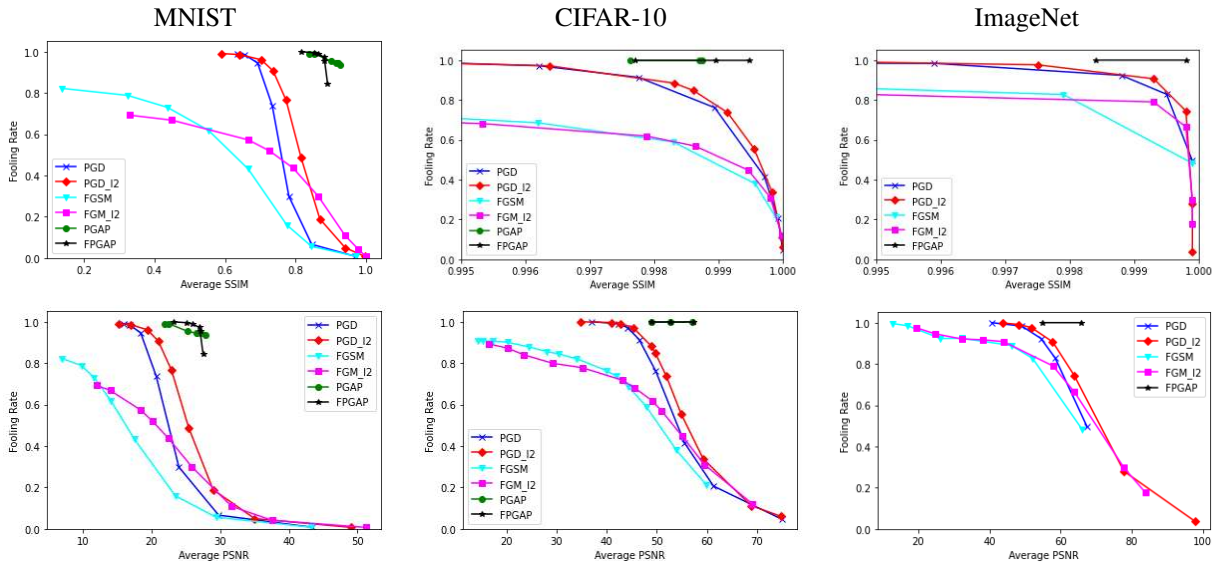


Fig. 6. Perceptual quality Average SSIM (top row) and Average PSNR (bottom row) versus Fooling Rate (FR) comparison of multiple adversarial perturbations on MNIST, CIFAR-10 and ImageNet datasets.

TABLE II
FOOLING RATE COMPARISON OF DIFFERENT ATTACKS ON DIFFERENT DATASETS AT A FIXED QUALITY A) MNIST DATASET (AVERAGE SSIM $\approx$ 0.85), B) CIFAR-10 DATASET (AVERAGE SSIM $\approx$ 0.998), AND C) IMAGENET DATASET (AVERAGE SSIM $\approx$ 0.998). PROPOSED METHODS ARE IN ITALICS.

Attack	FGSM	FGM_*l*2	PGD	PGD_*l*2	*FPGAP*
Fooling Rate – MNIST	0.0556	0.3004	0.066	0.1891	0.9953
Fooling Rate – CIFAR-10	0.5892	0.5681	0.7612	0.8472	1.0
Fooling Rate – ImageNet	0.82696	0.79008	0.92332	0.97614	1.0

and the resulting plots are shown in Fig 5. For reference, Table I lists the scores at a particular `noise threshold`.

These experiments clearly validate the qualitative observations made earlier in section V-A. Our method (FPGAP) beats other existing norm bounded methods by a factor of ≈ 2 on the precision scores. The number of pixels perturbed by our method is also lower compared with the other methods. For FPGAP, the recall scores are low; this observation is in line with our earlier observations that FPGAP introduces a smaller number of perturbations (as compared to the FGSM, which introduces perturbations all over the image).

C. Additional Results

In addition to the quantitative and qualitative analysis presented earlier, in this section, we analyze the perceptual quality of the adversarial examples generated by different methods. For each of the algorithms considered, we compute the structural similarity (SSIM) and PSNR between the adversarial examples and original images, averaged over the dataset. We plot the obtained Average SSIM and Average PSNR values vs FR (as defined in Section V-A) in Figure 6.

From Figure 6, it is clear that the approximation method FPGAP (Algorithm 2) does a fairly good job in approximating PGAP (Algorithm 1) on MNIST and CIFAR-10 datasets. Note that on the ImageNet dataset, we only report the results of FPGAP and ignore PGAP since it is computationally expensive to solve the QCQP iteratively.

From Figure 6, we can see that the adversarial perturbations generated by the proposed method lead to a much higher FR (than norm bounded approaches) in the high-SSIM regime; thus, our technique is able to generate adversarial examples with minimal impact on the image quality across all the three datasets. For illustration, we consider a parameter configuration that leads to roughly the same Average SSIM for all the techniques (see Table II). It is evident that the proposed method clearly achieves high FR within the given perceptual

TABLE III
FOOLING RATE COMPARISON OF DIFFERENT ATTACKS ON DIFFERENT DATASETS AT A FIXED QUALITY
A) MNIST DATASET (AVERAGE PSNR $\approx$ 25), B) CIFAR-10 DATASET (AVERAGE PSNR $\approx$ 57), AND C) IMAGENET DATASET (AVERAGE PSNR $\approx$ 65).
PROPOSED METHODS ARE IN ITALICS.

Attack	FGSM	FGM_*l*2	PGD	PGD_*l*2	*FPGAP*
Fooling Rate – MNIST	0.1592	0.3004	0.2982	0.4867	0.9953
Fooling Rate – CIFAR-10	0.2102	0.3099	0.4157	0.3376	1.0
Fooling Rate – ImageNet	0.4822	0.66434	0.4942	0.74458	0.99998

quality range. We also corroborate this using FR comparison at the same Average PSNR (see Table III).

D. Remarks

For some data points, we observed that the QCQP was not solved accurately (the cvxpy logs showed a much higher duality gap), resulting in anomalous adversarial examples with a much lower SSIM. This, in turn, underestimates the true average SSIM of PGAP, leading to higher fooling rates at lower (estimated) SSIM. This underestimation can be observed in Figure 6 (first column), where we see that the FR achieved by FPGAP is higher than the FR achieved by PGAP for the same average SSIM. This shortcoming is overcome by our Faster PGAP (FPGAP) solution.

As mentioned earlier, we find it very intriguing that incorporating an SSIM constraint (which enforces perceptual similarity) also makes the perturbations semantically meaningful. One potential hypothesis is that the constraints derived from the SSIM index restrict the space of perturbations, and together with the guidance provided by the gradients calculated from the network, enable us to find the semantically important regions while inducing minimal perturbations in those regions.

VI. CONCLUSION

In this work, we investigated adversarial examples and their relationship to semantically significant regions of the image. Also, as a byproduct of trying to answer this question, we proposed a perceptually guided technique to generate adversarial examples that are structurally similar to the original image. By leveraging useful mathematical properties of the SSIM index, we presented a convex formulation to find adversarial examples. To the best of our knowledge, this is the first convex formulation that explicitly incorporates the SSIM index into the adversarial framework. In addition, we also provide a (fast) closed form approximation that enables solving the proposed convex formulation on large datasets. In fact, none of the existing adversarial example techniques that use image quality metrics have a closed-form solution. This is in stark contrast to other norm bounded techniques (PGD, FGSM, etc.), which employ a closed-form solution to generate adversarial perturbations.Our formulation also does not assume any model on the adversarial perturbations. We analyzed the adversarial perturbations generated by the proposed technique on images from the ImageNet validation set using SSIM maps and GradCAM++. By comparing the precision and IOU scores, we observed that, unlike standard techniques, the proposed technique is semantically-aware, i.e., it specifically targets the regions of the image that are important for classification. The proposed method also generates high-quality adversarial examples while achieving a Fooling Rate similar to comparable techniques.

APPENDIX

A. Constraints Analysis

We analyse the constraints of (8) in more detail in this section.

1) *Constraint 1:* The first constraint $S_1(x, x_{adv}) \geq 1 - \epsilon_1^2$ is a linear constraint (this forces x_{adv} to lie in an intersection of two half-spaces). We can see this using (6) and (7):

$$||\mu_{x_{adv}} - \eta_1 \mu_x||^2 \leq ||\mu_x||_2^2 (\eta_1)^2 + c_1 \left(\epsilon_1^2 \eta_1\right),$$

or equivalently

$$N(-\sqrt{k_{11}} + k_{12}) \leq 1^T x_{adv} \leq (\sqrt{k_{11}} + k_{12})N, \quad (12)$$

where

$$k_{11} = (\eta_1)^2 ||\mu_x||_2^2 + c_1 \left(\epsilon_1^2 \eta_1\right),$$
$$k_{12} = \eta_1 \mu_x, \text{ and } \eta_1 = 1/(1 - \epsilon_1^2).$$

2) *Constraint 2:* The second constraint in (8) $S_2(x, x_{adv}) \geq 1 - \epsilon_2^2$ is a quadratic constraint (this forces x_{adv} to be in a high dimensional sphere). We can see this using (6):

$$||(x_{adv} - 1\mu_{x_{adv}}) - \eta_2 (x - 1\mu_x)||^2 \leq ||x - 1\mu_x||_2^2 (\eta_2)^2 + c_2(\epsilon_2^2 \eta_2),$$

or equivalently

$$||(x_{adv} - 1\mu_{x_{adv}}) - k_{22}|| \leq \sqrt{k_{21}}; \quad (13)$$

where

$$k_{21} = ||x - 1\mu_x||_2^2 (\eta_2)^2 + c_2(\eta_2 \epsilon_2^2),$$
$$k_{22} = \eta_2 (x - 1\mu_x) \text{ and } \eta_2 = 1/(1 - \epsilon_2^2).$$

Based on the constraints (12), (13), and the objective, the optimization problem in (8) is convex; in particular it is a Quadratically Constrained Quadratic Program (QCQP).

B. Problem Approximation

We formulate an equivalent optimization problem (14) from (9), by relaxing the first constraint and substituting it in the second. While this solution is an approximation to PGAP on account of relaxing one of the constraints, it has the advantage of having a closed form solution.

Consider the following approximation to (9).

$$\begin{aligned} \underset{x_{adv}}{\operatorname{argmin}} \quad & -(x_{adv})^T \nabla_x \mathcal{L}(w,x,y) \\ \text{s.t.} \quad & ||(x_{adv} - 1(\mu_x)) - k_{22}||^2 \leq k_{21} \end{aligned} \tag{14}$$

We first define the Lagrangian [27] of the above problem:

$$\begin{aligned} L = & -(x_{adv})^T \nabla_x \mathcal{L}(w,x,y) \\ & + \lambda(||(x_{adv} - 1(\mu_x)) - k_{22}||^2 - k_{21}) \end{aligned} \tag{15}$$

We first take gradient of Lagrangian L and equate it to zero: $\nabla_{x_{adv}} L = 0$. This leads us to

$$\nabla_x \mathcal{L}(w,x,y) = 2\lambda((x_{adv} - 1(\mu_x)) - k_{22}). \tag{16}$$

Since the objective in (14) is linear, from geometrical intuition the maximum/minimum occurs only on the boundary of the sphere (specified by the constraint):

$$||(x_{adv} - 1(\mu_x)) - k_{22}||^2 = k_{21},$$

which can be rewritten using (16) as

$$\nabla_x \mathcal{L}(w,x,y)^T \nabla_x \mathcal{L}(w,x,y) = 4\lambda^2 k_{21} \tag{17}$$

Using the value of λ from (17) in (16) we get:

$$x_{adv} = 1(\mu_x) + k_{22} + (\sqrt{k_{21}}) \left(\frac{\nabla_x \mathcal{L}(w,x,y)}{||\nabla_x \mathcal{L}(w,x,y)||} \right),$$

where k_{21}, k_{22} are defined in (13) and 1 in $1(\mu_x)$ is all ones of size x_{adv}; thus providing a closed form solution to (14).

C. Supplementary material

In this work, we try to understand the landscape of adversarial perturbations through the perceptual quality lens. To achieve this goal, we rely on the perceptual quality metric SSIM index. We generate adversarial perturbations by maximizing the linear approximation of the loss function subject to the constraints derived from the mathematically amenable variant of the SSIM index. We use Carlini-Wagner (CW) loss function (with confidence parameter k =50) [20] for generating adversarial examples. We analyze the perturbations generated by proposed method qualitatively and quantitatively and show the efficacy in terms of localization to semantically important regions compared to the norm-bounded adversarial perturbations.

Tools used: Tensorflow [28], Cvxpy [29], Foolbox [30], tf-keras-vis [31].

D. Qualitative Analysis

To understand and analyze the impact of adversarial perturbations on the spatial regions of images, we use SSIM maps. Compared to the standard l_p norm bounded perturbations, the proposed approach generates perturbations that are perceptually-aware (structure-aware) and able to find the regions that are important for classification. We provide corroborative evidence of the same using GradCAM++ output.

1) SSIM Map: SSIM maps contain local SSIM index values at pixel level calculated using pixels in the local neighbourhood of 8×8 block or Gaussian window of size 11×11 centered at the pixel. The global SSIM index value is calculated by using/pooling these local SSIM index values. Typically these maps are defined for grayscale images or one colour plane of colour images, here we present the SSIM maps of three channels (R, G, and B). These maps help us to visualize the distortion locally at a pixel level. In this work, we use these maps to locate the image regions that are affected by adversarial perturbations. For example, in Figure 10 the dark pixels in the second column of images are the locations where adversarial perturbations are introduced in the channel one (Red), and columns three and four corresponds to channels two and three (Green and Blue).

2) Supporting Results: To further illustrate efficacy of our method we present several examples (Figures: 10, 11, 12, 13, 14, 15 and 16) of the proposed method along with other norm bounded perturbations in the following:

We observed that the iterative methods (PGD, PGD_$l2$, PGAP, FPGAP) are doing well compared to the basic (non-iterative) methods (FGSM and FGM). In iterative approaches, by updating the image at each iteration, non-linearity is introduced in crafting the adversarial perturbations, which could be the reason for better performance. To understand these iterative methods further, we analyse the impact of adversarial perturbations on spatial regions of images using absolute difference maps.

3) Understanding Iterative Methods: We use random test images from MNIST and CIFAR-10 test datasets for this analysis. To understand and analyze the impact of adversarial perturbations on the spatial regions of images, we use absolute difference maps. These maps are generated by taking the absolute difference of adversarial images with respect to original image.These maps will help us in locating the perturbations introduced in different parts of the image. Following is the setup we use for this analysis:

1) Fix number of iterations for all the methods (for example: 10, 30 etc).
2) Find the smallest ϵ to generate the adversarial example (for random test images at fixed number of iterations specified in step 1).
3) Compute the absolute difference map, PSNR and SSIM with respect to original image.

Figures 19, 20, 21 and 22 are generated using random test images from the MNIST test images (at different iterations: 10, 30, 50 and 70 respectively). We make the following observations from these figures:

1) From the absolute difference maps we can say that the proposed method introduces perturbations only at important regions of the image (around the digit present in the image) in all iteration settings.
2) Since the proposed method is able to identify the structural regions and adding perturbations only in those regions the required amount of perturbation is small compared to other methods which adds perturbations all over the image. We believe that this could be the reason

for high PSNR and SSIM index values of adversarial images generated by proposed method compared to the other methods (PGD and PGD_$l2$).

3) As the number of iterations increase the performance of PGD and PGD_$l2$ is improving. In particular, as we increase the number of iterations the perturbations introduced by PGD_$l2$ are more localized around the digit area.
4) The proposed method seems to achieve higher PSNR and SSIM index values in fewer iterations compared to other methods.

Similar observations can be made on the CIFAR-10 test images (Figures 23, 24, 25 and 26).

E. Additional Results

We compare our method with FGSM (l_∞ bounded perturbations) [8], FGM (l_2 bounded perturbations) [9], and PGD (l_∞ and l_2 bounded perturbations) [18]. Note that for iterative methods the number of iterations used are 50. Each of the techniques under consideration has a parameter ϵ that controls the fooling rate: for norm bounded perturbations this is the largest allowed norm of the perturbation. For the proposed method, the parameter is ϵ_2 ((9) in the main draft) which bounds the structural distortion in the generated adversarial example. We generate adversarial examples at different parameter values on MNIST, CIFAR-10 and ImageNet datasets for all these techniques. We can see the relation between the parameter ϵ and the Fooling Rate for these methods in Figures 7, 8 and 9.

However, the parameter ϵ has different role/significance (and different range of values) for each method. This makes it difficult to compare the success of different methods with respect to ϵ. Hence we take a different route, and compare the imperceptibility and the Fooling Rate of the attack. To measure the imperceptibility we use image quality metric SSIM and PSNR. We compute Average SSIM and Average PSNR over the dataset at different parameter values (presented in Figures 7, 8 and 9) for each method. Then we compare these with the Fooling Rate (Figure 6 in the main draft).

REFERENCES

[1] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of control, signals and systems*, vol. 2, no. 4, pp. 303–314, 1989.

[2] X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial examples: Attacks and defenses for deep learning," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 9, pp. 2805–2824, 2019.

[3] S. Qiu, Q. Liu, S. Zhou, and C. Wu, "Review of artificial intelligence adversarial attack and defense technologies," *Applied Sciences*, vol. 9, no. 5, p. 909, 2019.

[4] E. Wong and J. Z. Kolter, "Provable defenses against adversarial examples via the convex outer adversarial polytope," *arXiv preprint arXiv:1711.00851*, 2017.

[5] A. Raghunathan, J. Steinhardt, and P. S. Liang, "Semidefinite relaxations for certifying robustness to adversarial examples," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 877–10 887.

[6] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, "Adversarial examples are not bugs, they are features," *arXiv preprint arXiv:1905.02175*, 2019.

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

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

[9] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, "Boosting adversarial attacks with momentum," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9185–9193.

[10] D. Hendrycks and T. G. Dietterich, "Benchmarking neural network robustness to common corruptions and surface variations," *arXiv preprint arXiv:1807.01697*, 2018.

[11] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[12] M. Sharif, L. Bauer, and M. K. Reiter, "On the suitability of lp-norms for creating and preventing adversarial examples," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 1605–1613.

[13] A. Chattopadhay, A. Sarkar, P. Howlader, and V. N. Balasubramanian, "Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2018, pp. 839–847.

[14] M. Jordan, N. Manoj, S. Goel, and A. G. Dimakis, "Quantifying perceptual distortion of adversarial examples," *arXiv preprint arXiv:1902.08265*, 2019.

[15] A. Rozsa, E. M. Rudd, and T. E. Boult, "Adversarial diversity and hard positive generation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016, pp. 25–32.

[16] D. Brunet, E. R. Vrscay, and Z. Wang, "On the mathematical properties of the structural similarity index," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 1488–1499, 2011.

[17] M. Z. Hameed and A. Gyorgy, "Perceptually constrained adversarial attacks," *arXiv preprint arXiv:2102.07140*, 2021.

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

[19] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," *arXiv preprint arXiv:1607.02533*, 2016.

[20] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 IEEE Symposium on Security and Privacy (sp)*. IEEE, 2017, pp. 39–57.

[21] P.-Y. Chen, Y. Sharma, H. Zhang, J. Yi, and C.-J. Hsieh, "Ead: elastic-net attacks to deep neural networks via adversarial examples," in *Thirty-second AAAI conference on artificial intelligence*, 2018.

[22] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 5, pp. 828–841, 2019.

[23] K. Xu, S. Liu, P. Zhao, P.-Y. Chen, H. Zhang, Q. Fan, D. Erdogmus, Y. Wang, and X. Lin, "Structured adversarial attack: Towards general implementation and better interpretability," *arXiv preprint arXiv:1808.01664*, 2018.

[24] K. Parimala and S. Channappayya, "Quality aware generative adversarial networks," *Advances in neural information processing systems*, vol. 32, pp. 2948–2958, 2019.

[25] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.

[26] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[27] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[28] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: A system for large-scale machine learning," in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, 2016, pp. 265–283. [Online]. Available: https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf

[29] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.

[30] J. Rauber, W. Brendel, and M. Bethge, "Foolbox: A python toolbox to benchmark the robustness of machine learning models," in *Reliable Machine Learning in the Wild Workshop, 34th International*

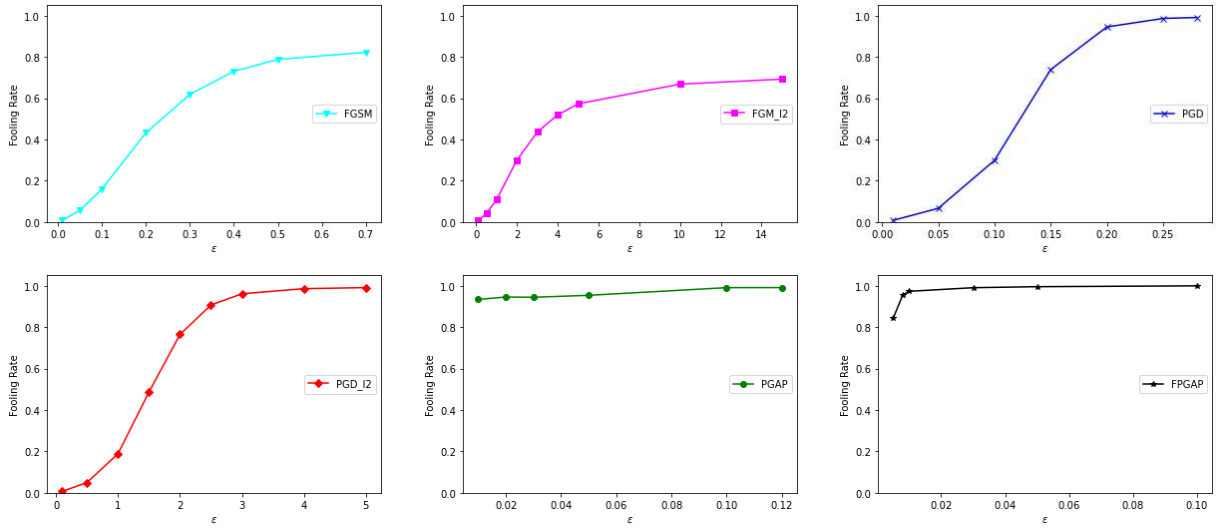


Fig. 7. ϵ versus Fooling Rate (FR) comparison of multiple adversarial perturbations on MNIST.

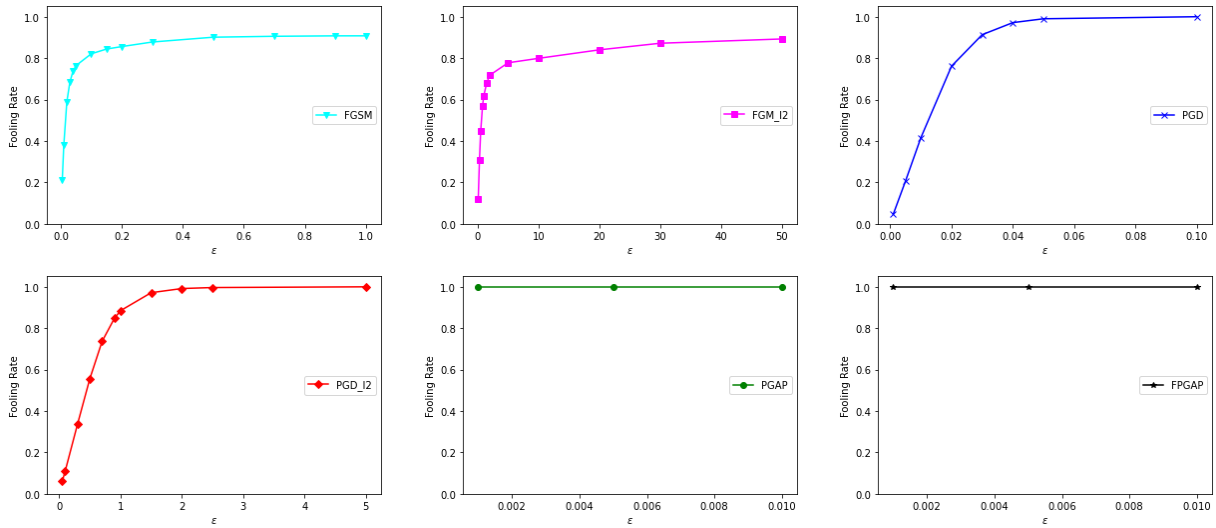


Fig. 8. ϵ versus Fooling Rate (FR) comparison of multiple adversarial perturbations on CIFAR-10.

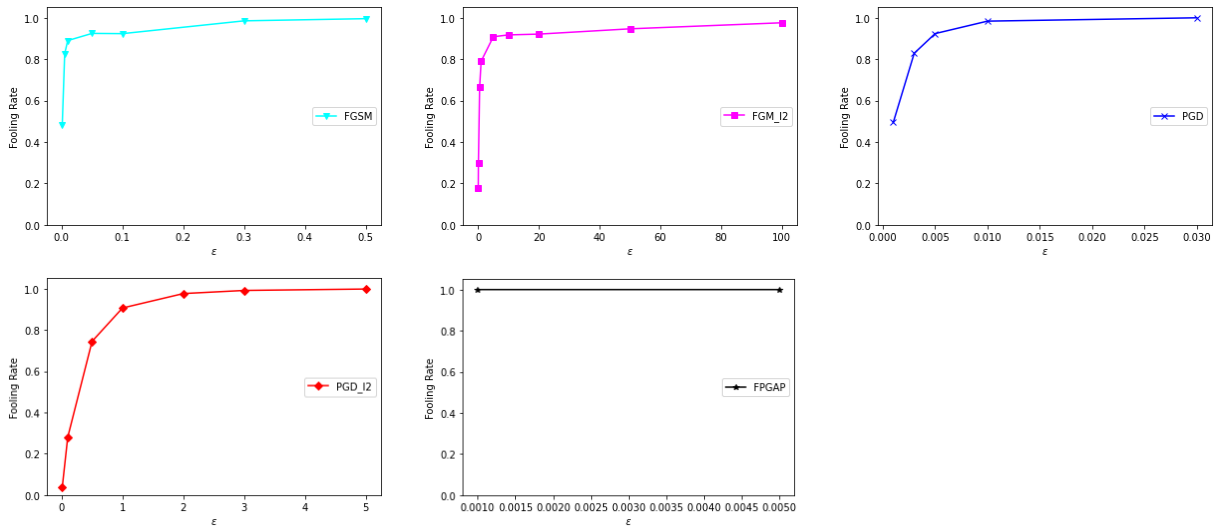


Fig. 9. ϵ versus Fooling Rate (FR) comparison of multiple adversarial perturbations on ImageNet.

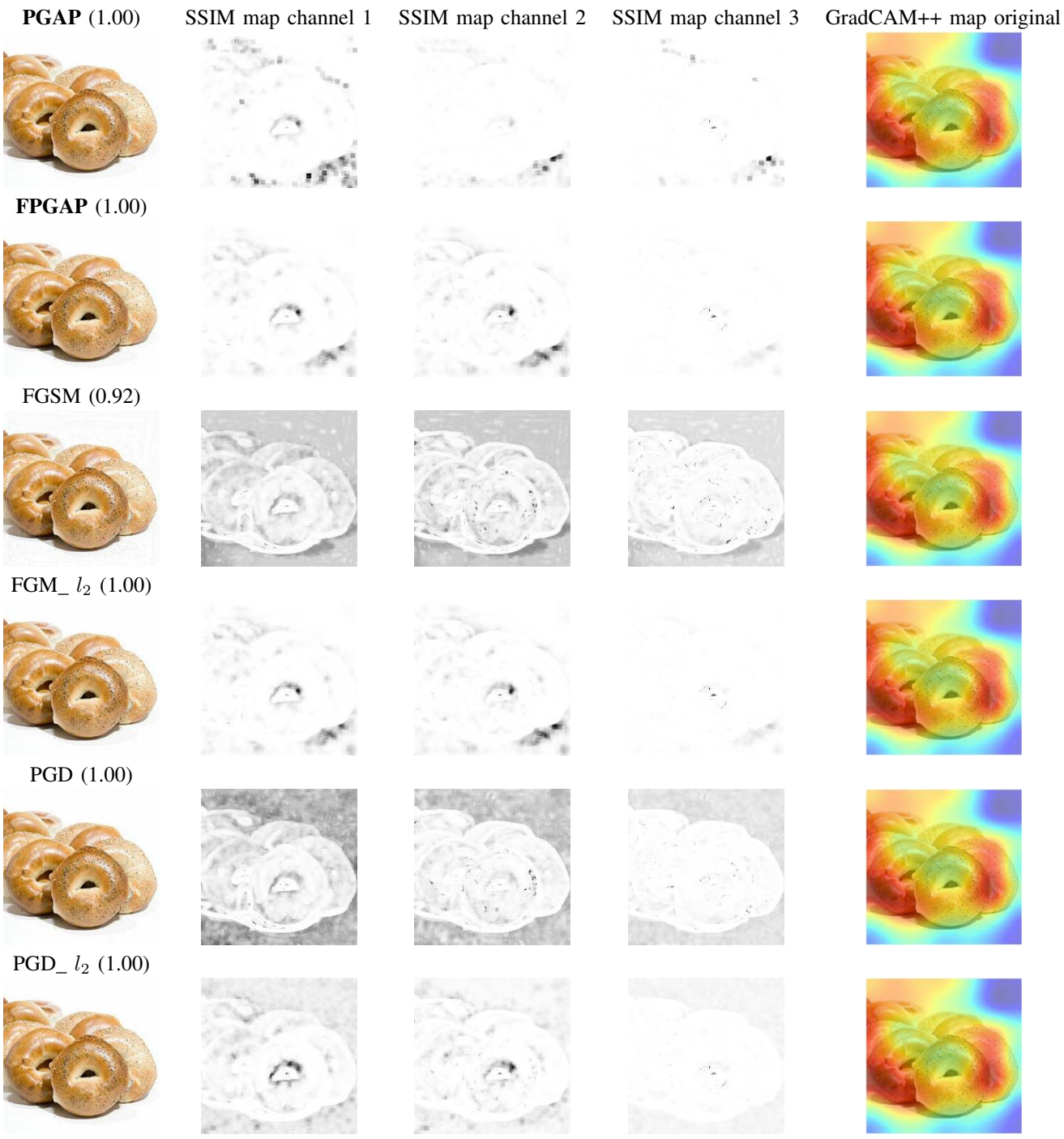


Fig. 10. SSIM maps comparison of adversarial examples generated. (a): Adversarial perturbations with different methods and SSIM index value (rounded off to two decimal places), (b),(c) and (d): SSIM maps of RGB channels respectively, (e): GradCAM++ output of original image.

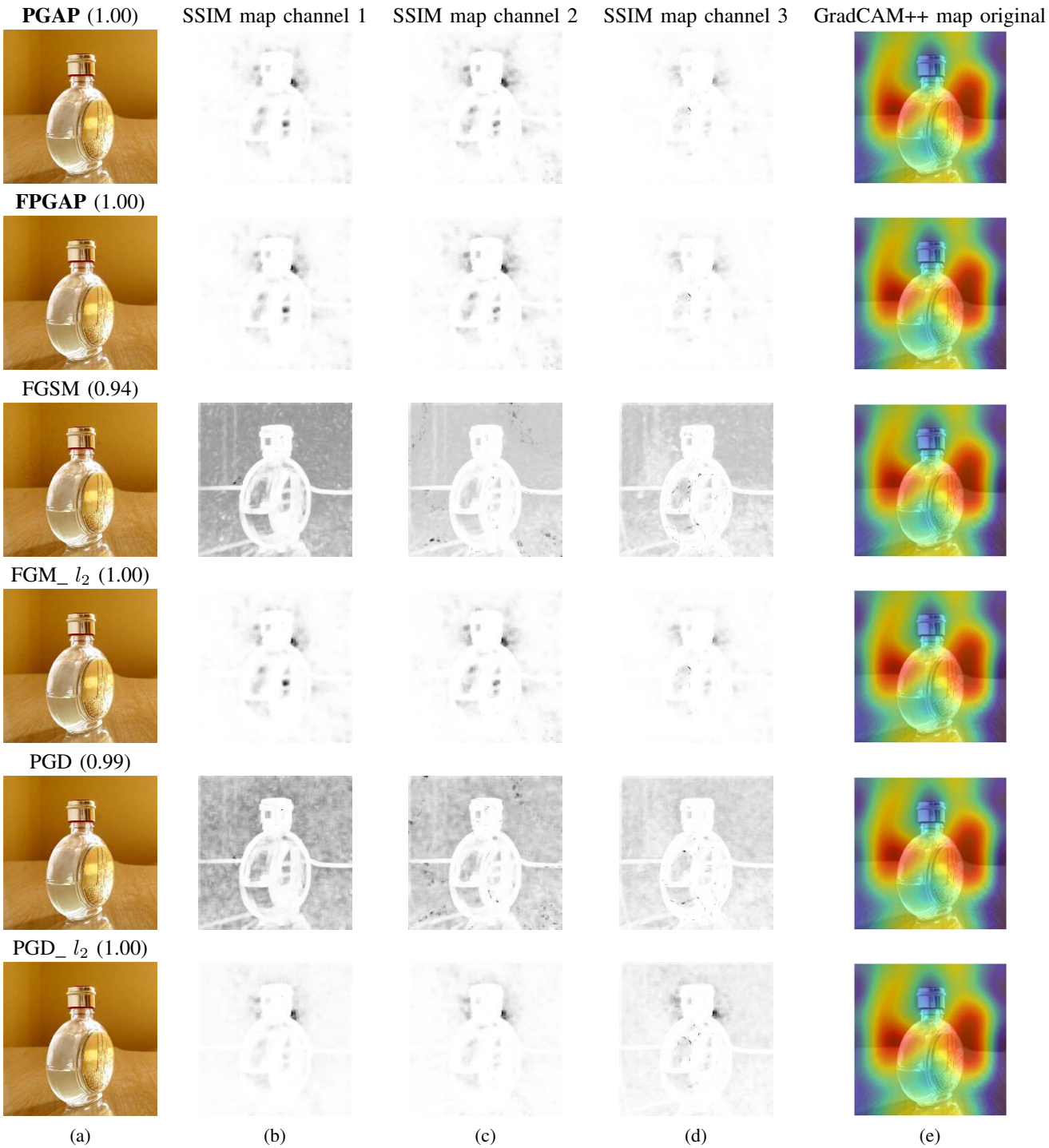


Fig. 11. SSIM maps comparison of adversarial examples generated. (a): Adversarial perturbations with different methods and SSIM index value (rounded off to two decimal places), (b),(c) and (d): SSIM maps of RGB channels respectively, (e): GradCAM++ output of original image.

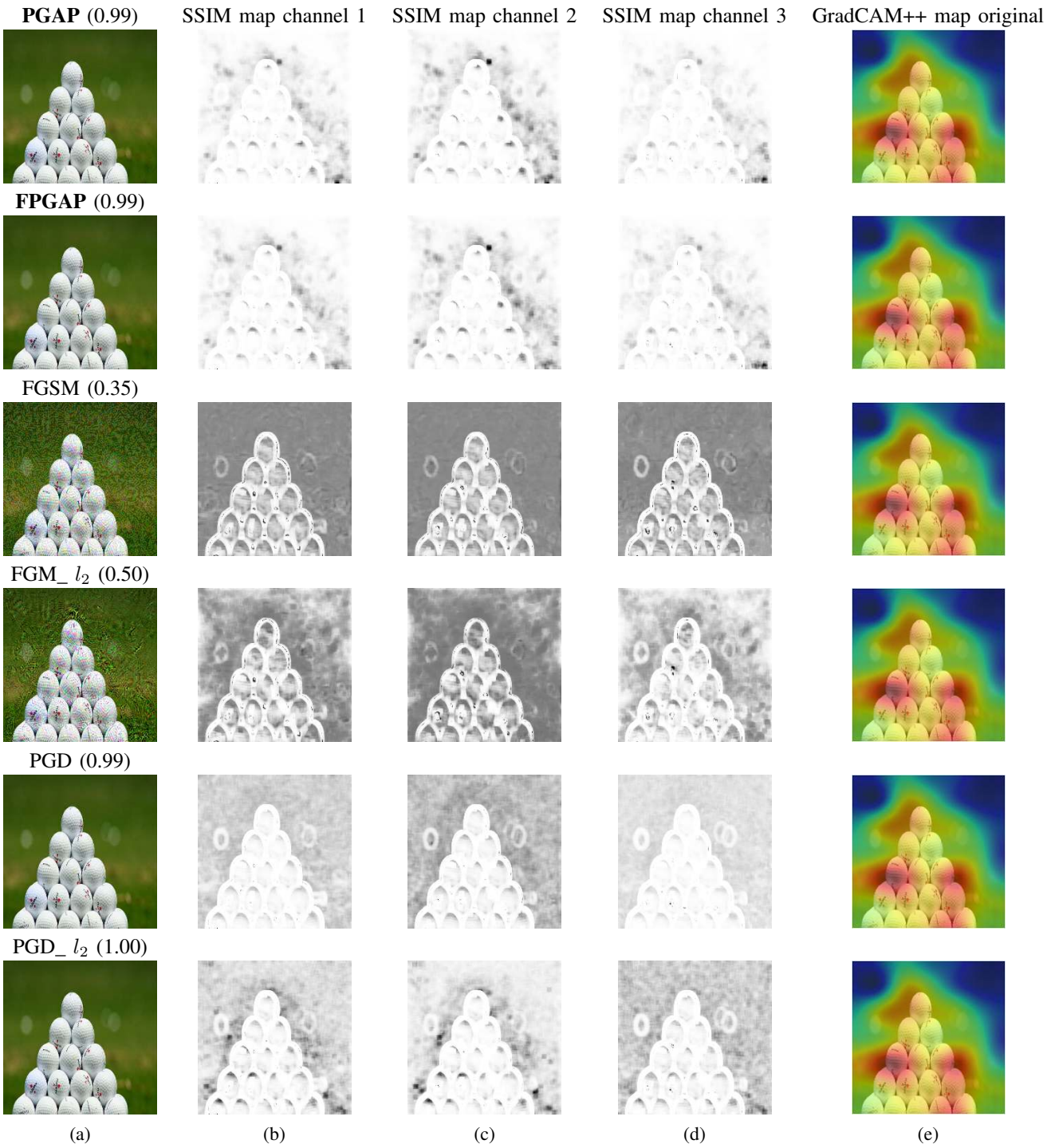


Fig. 12. SSIM maps comparison of adversarial examples generated. (a): Adversarial perturbations with different methods and SSIM index value (rounded off to two decimal places), (b),(c) and (d): SSIM maps of RGB channels respectively, (e): GradCAM++ output of original image.

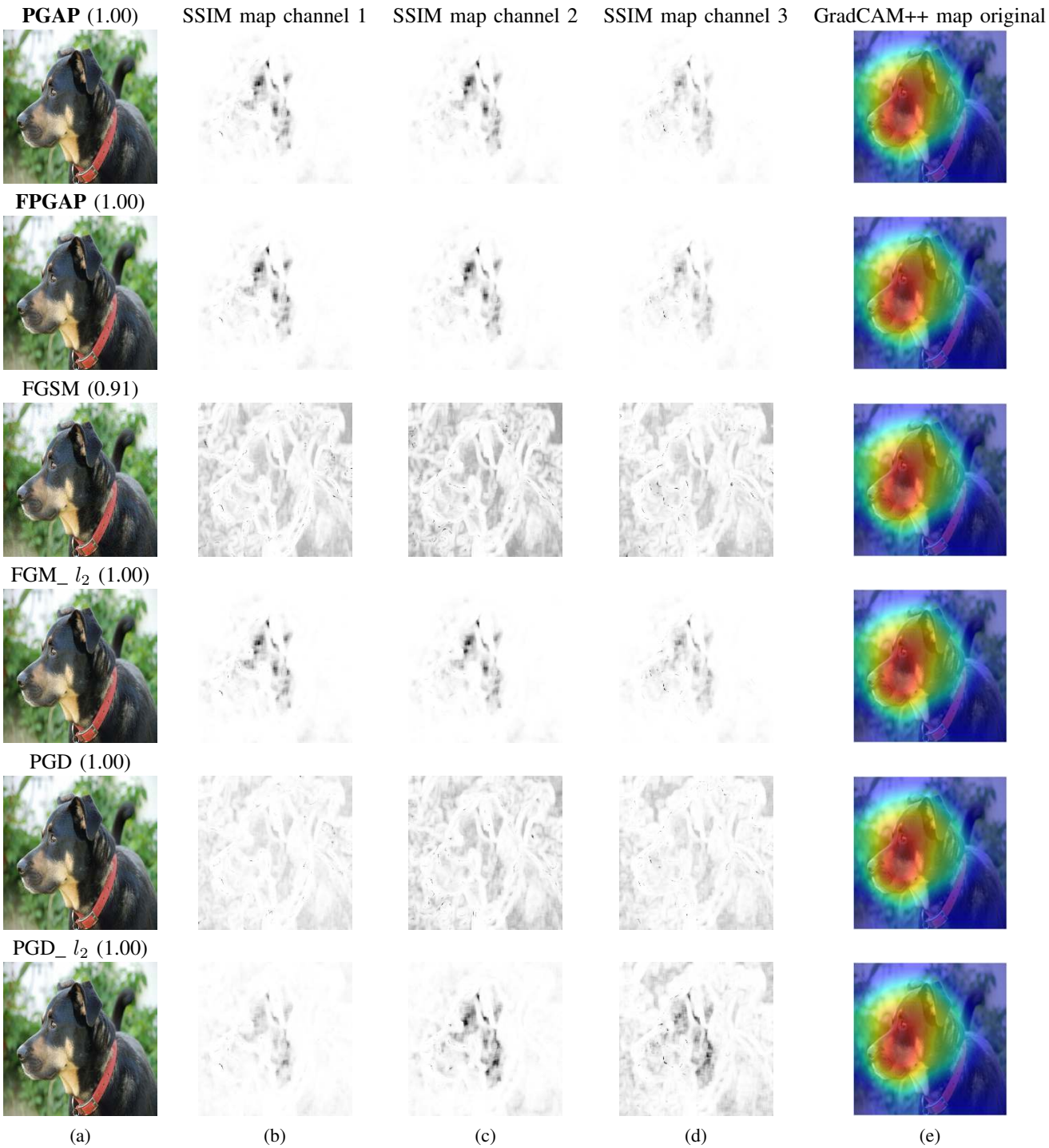


Fig. 13. SSIM maps comparison of adversarial examples generated. (a): Adversarial perturbations with different methods and SSIM index value (rounded off to two decimal places), (b),(c) and (d): SSIM maps of RGB channels respectively, (e): GradCAM++ output of original image.

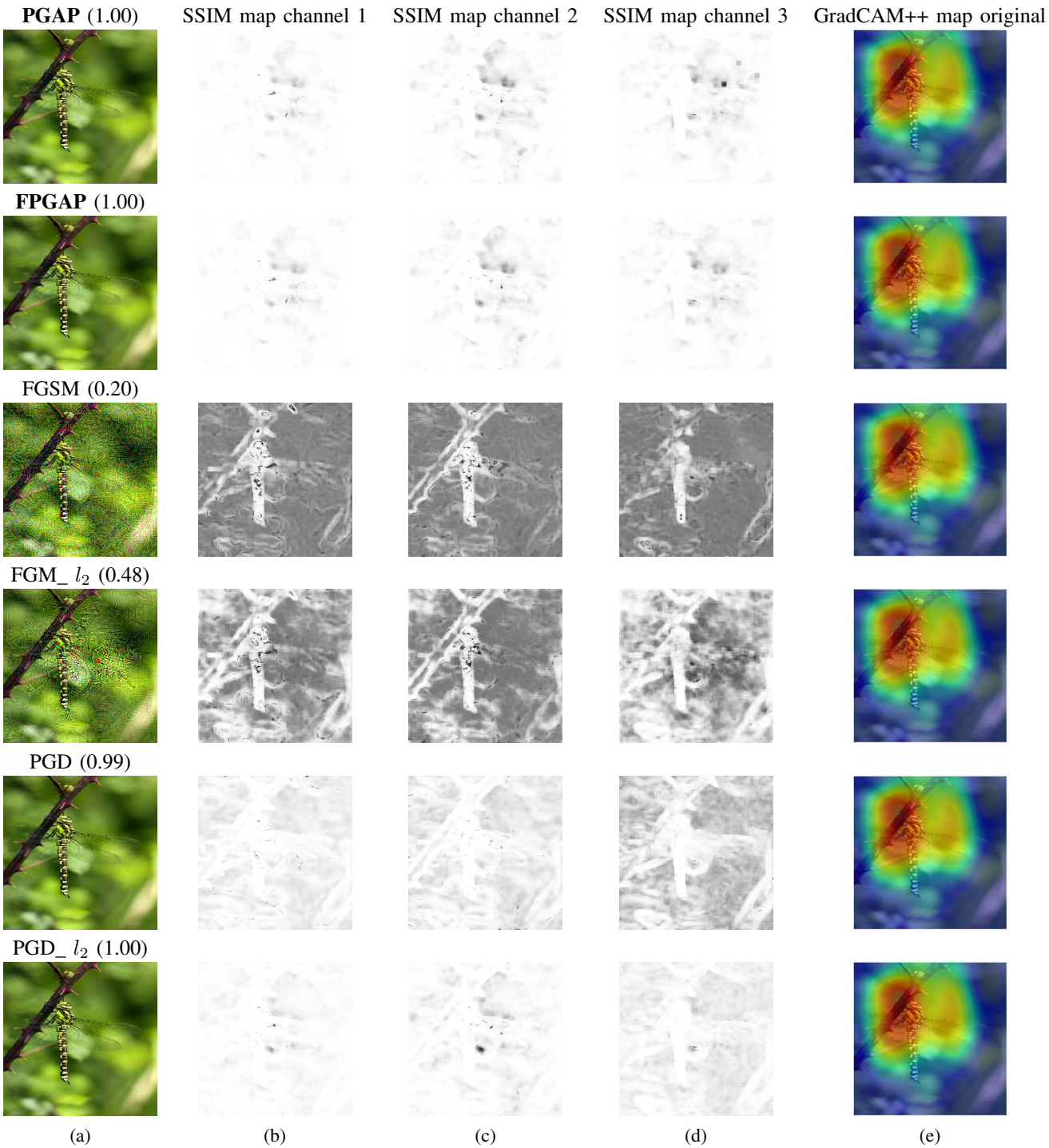


Fig. 14. SSIM maps comparison of adversarial examples generated. (a): Adversarial perturbations with different methods and SSIM index value (rounded off to two decimal places), (b),(c) and (d): SSIM maps of RGB channels respectively, (e): GradCAM++ output of original image.

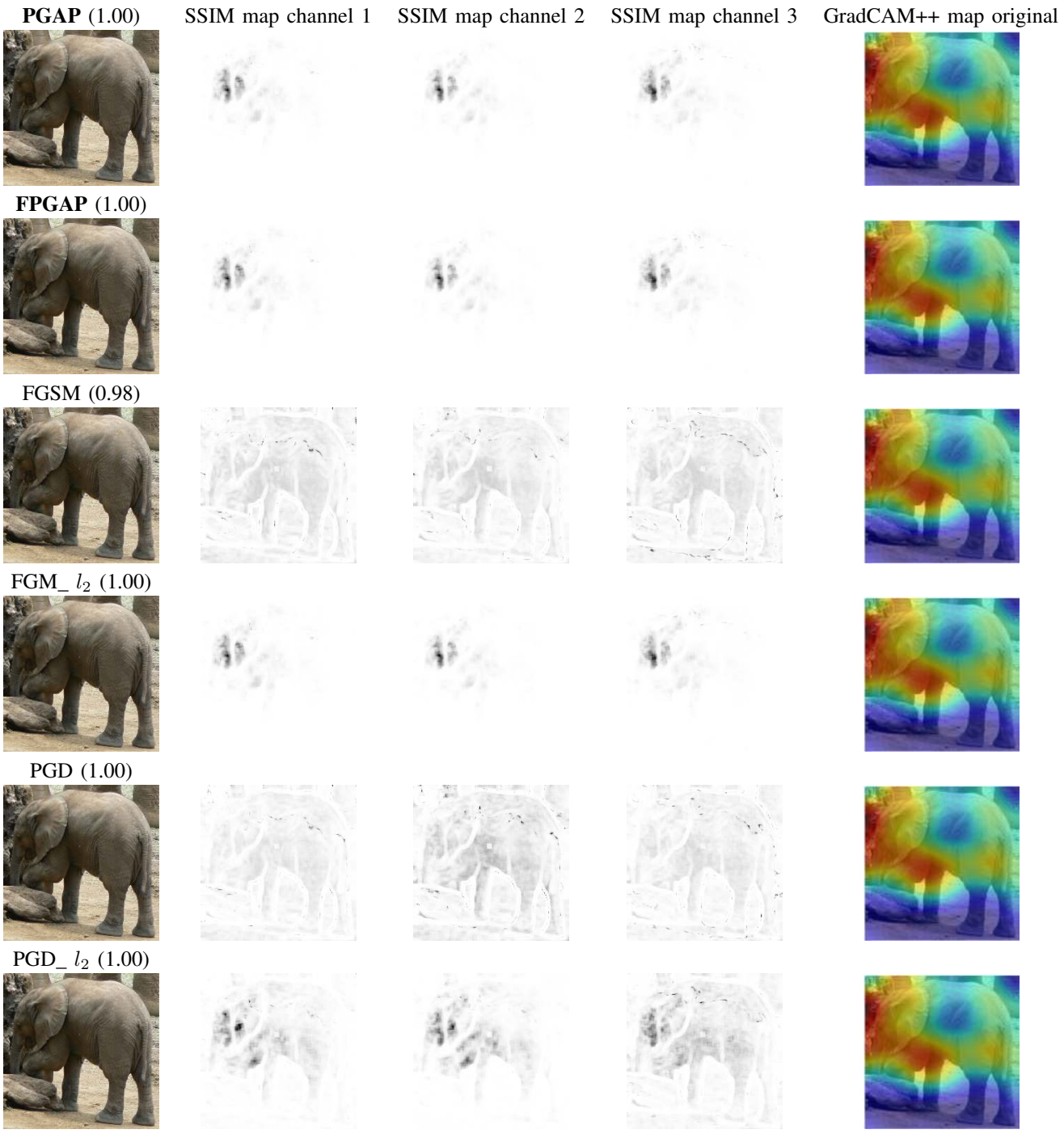


Fig. 15. SSIM maps comparison of adversarial examples generated. (a): Adversarial perturbations with different methods and SSIM index value (rounded off to two decimal places), (b),(c) and (d): SSIM maps of RGB channels respectively, (e): GradCAM++ output of original image.

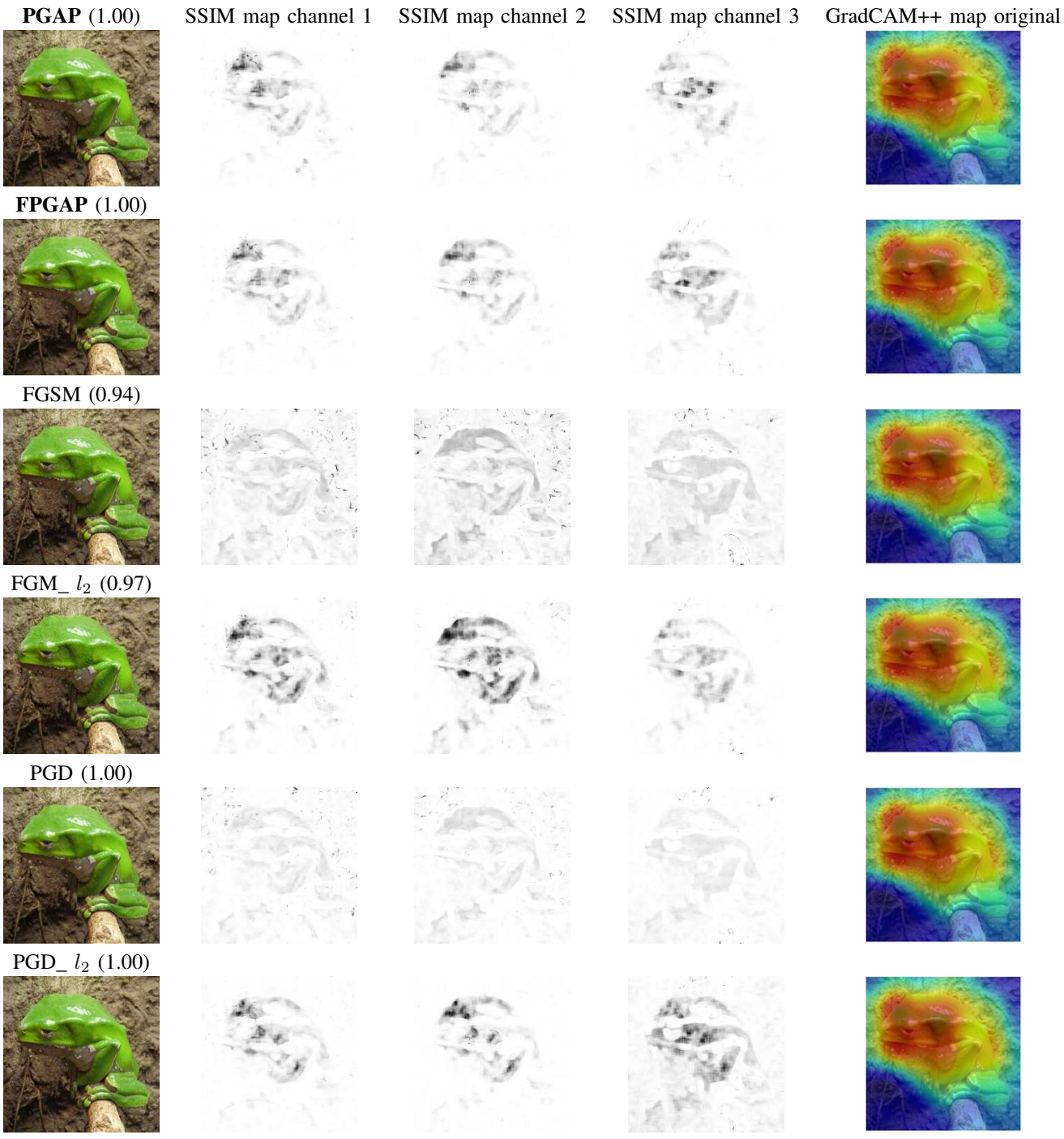


Fig. 16. SSIM maps comparison of adversarial examples generated. (a): Adversarial perturbations with different methods and SSIM index value (rounded off to two decimal places), (b),(c) and (d): SSIM maps of RGB channels respectively, (e): GradCAM++ output of original image.

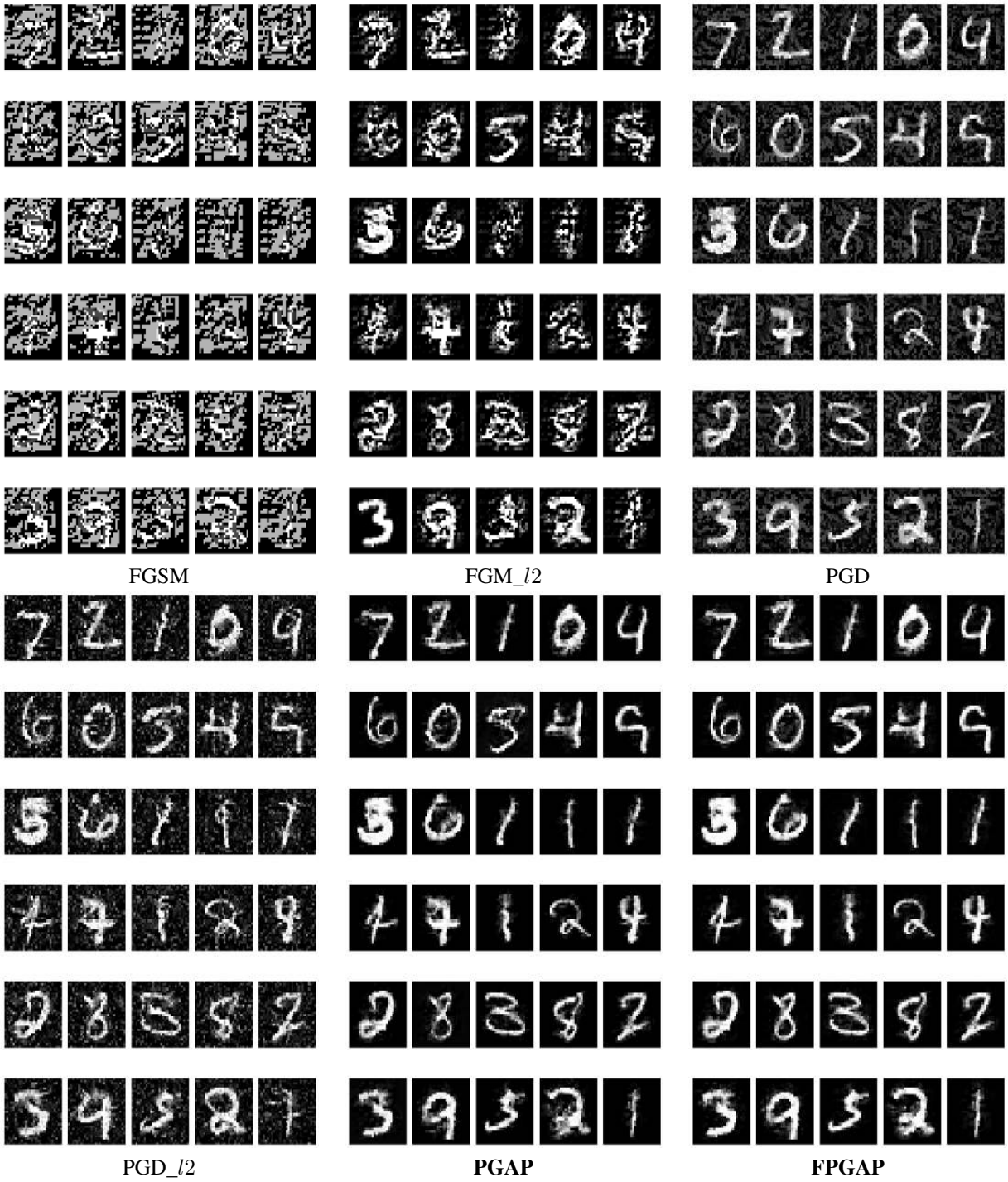


Fig. 17. Adversarial examples generated using multiple methods with similar fooling rate on MNIST dataset.

Conference on Machine Learning, 2017. [Online]. Available: http://arxiv.org/abs/1707.04131

[31] Y. Kubota and contributors, "tf-keras-vis," https://github.com/keisen/tf-keras-vis, 2019.

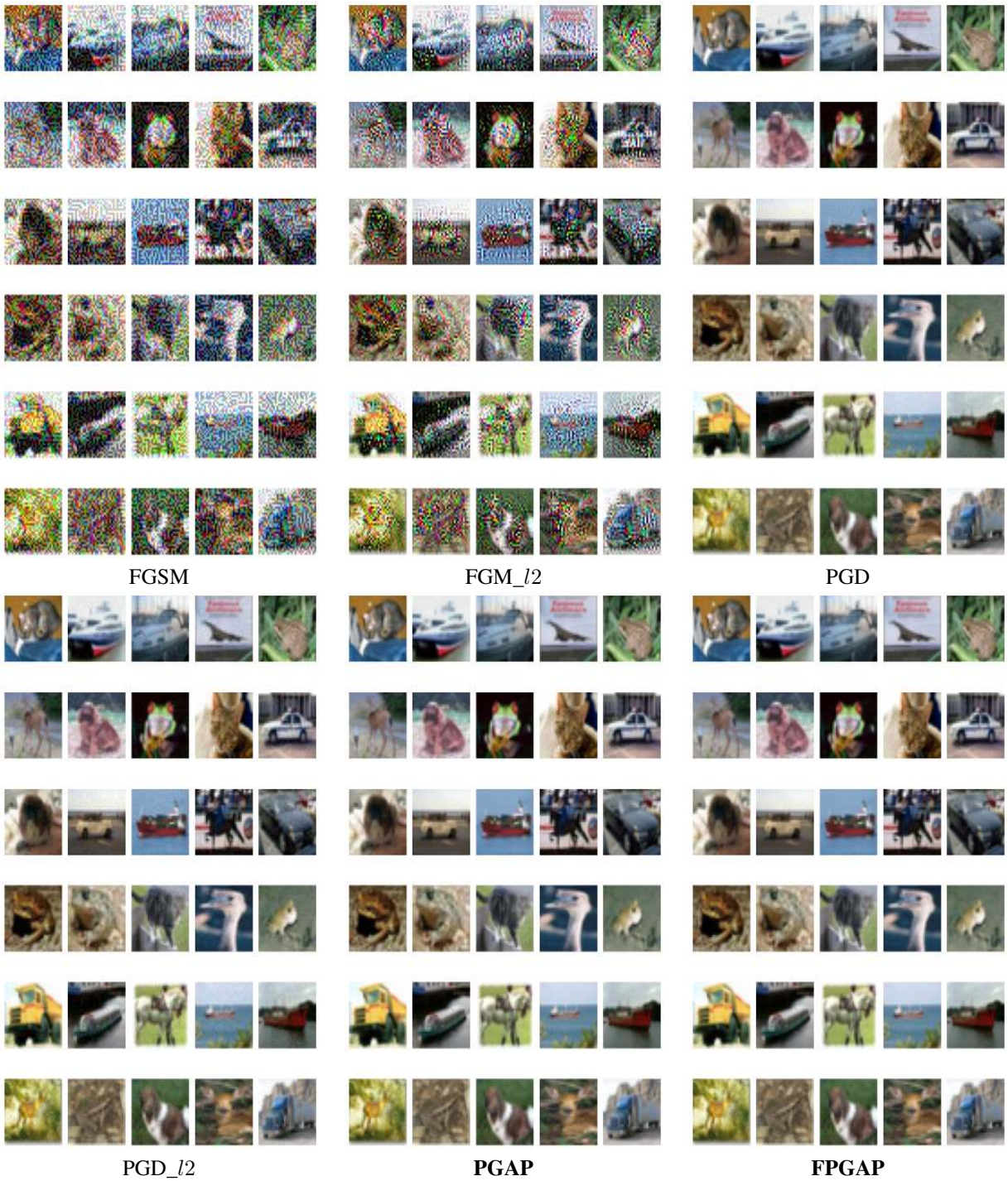


Fig. 18. Adversarial examples generated using multiple methods with similar fooling rate on CIFAR-10 dataset.

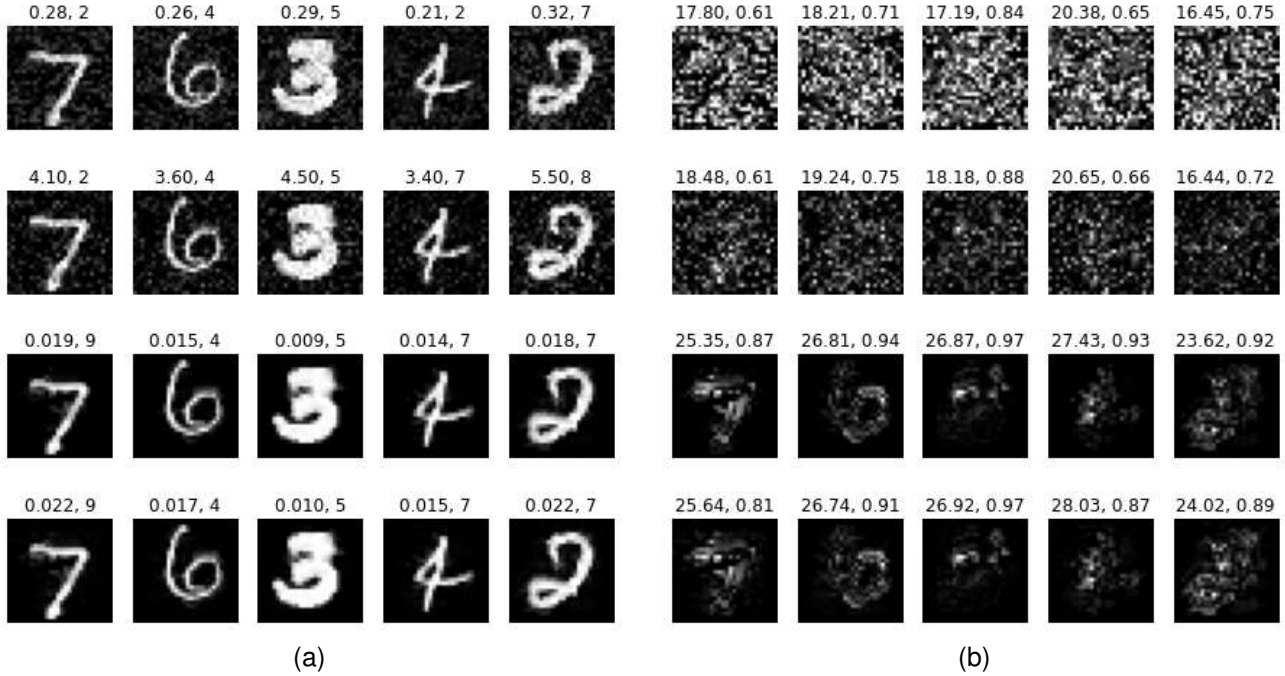


Fig. 19. Image on the left: Adversarial examples generated by multiple methods: from top to bottom PGD, PGD_$l2$, PGAP and FPGAP at fixed number of iterations (10). Numbers on top of each image are ϵ and prediction of adversarial image respectively. Image on the right: Absolute difference maps with respect to original image. Numbers on top of each map are PSNR and SSIM.

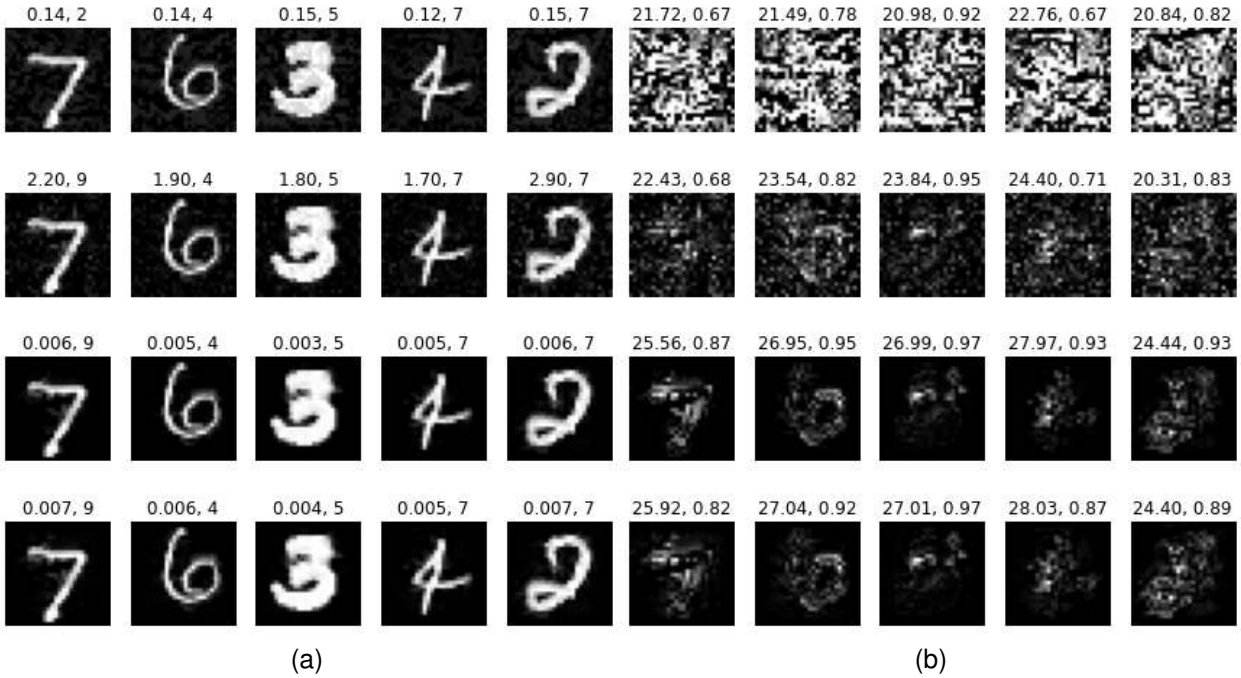


Fig. 20. Image on the left: Adversarial examples generated by multiple methods: from top to bottom PGD, PGD_$l2$, PGAP and FPGAP at fixed number of iterations (30). Numbers on top of each image are ϵ and prediction of adversarial image respectively. Image on the right: Absolute difference maps with respect to original image. Numbers on top of each map are PSNR and SSIM.

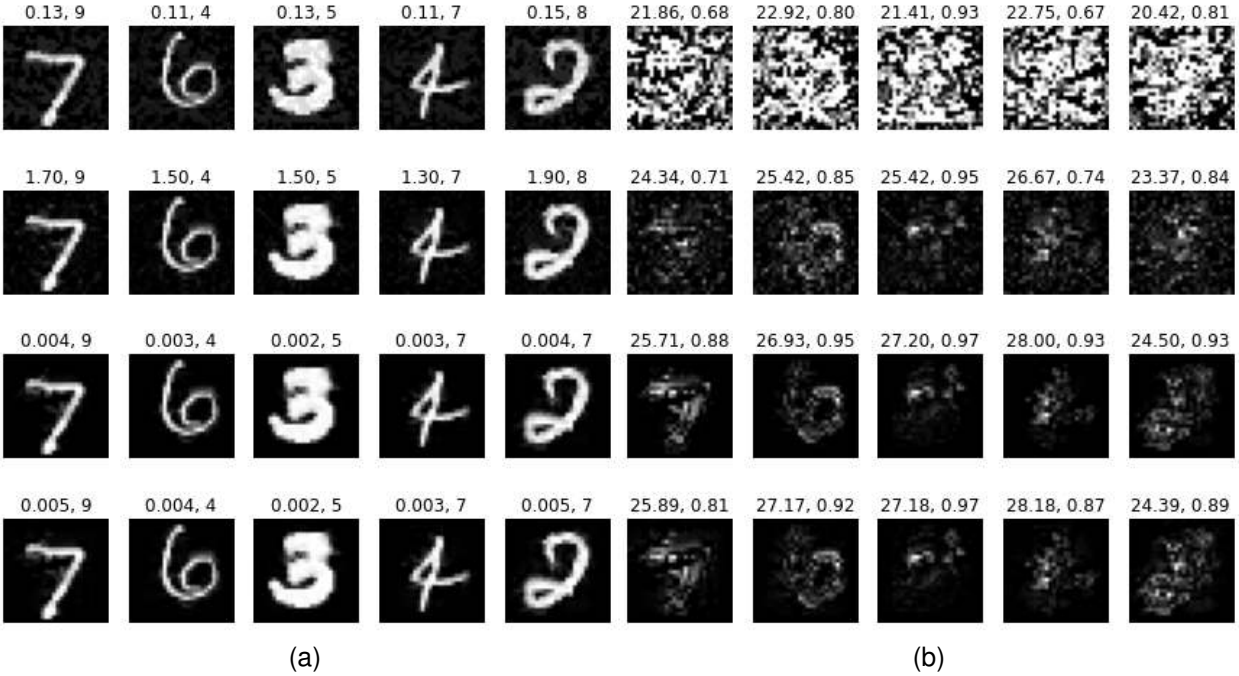


Fig. 21. Image on the left: Adversarial examples generated by multiple methods: from top to bottom PGD, PGD_$l2$, PGAP and FPGAP at fixed number of iterations (50). Numbers on top of each image are ϵ and prediction of adversarial image respectively. Image on the right: Absolute difference maps with respect to original image. Numbers on top of each map are PSNR and SSIM.

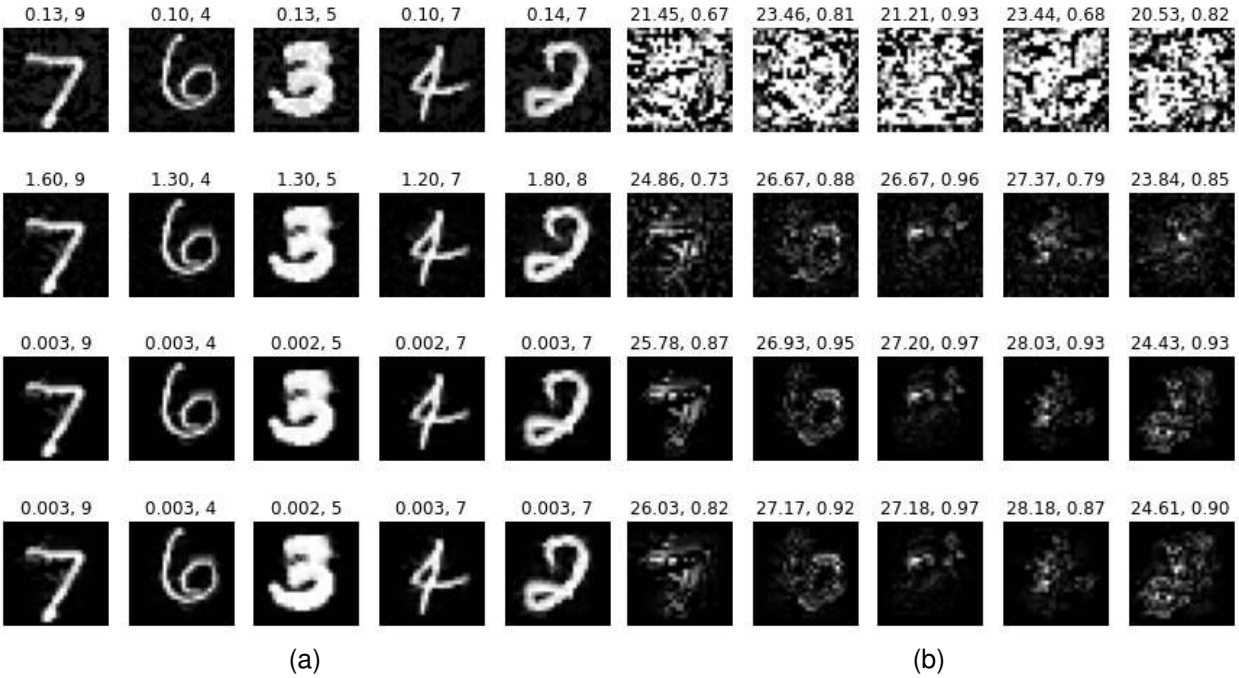
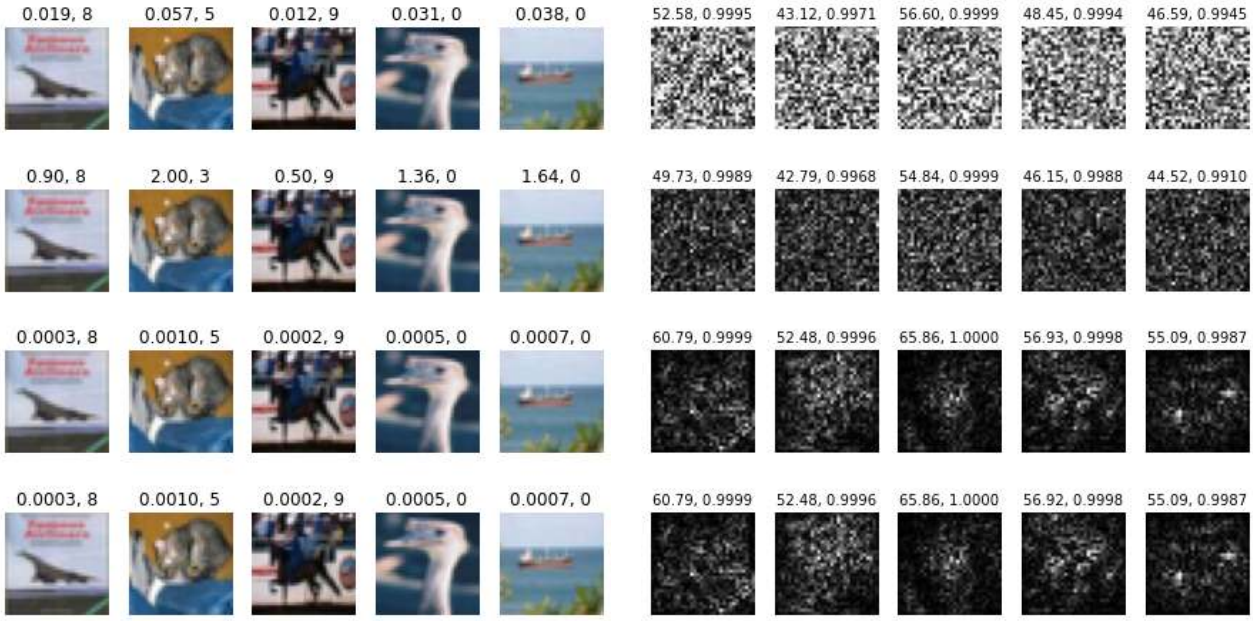


Fig. 22. Image on the left: Adversarial examples generated by multiple methods: from top to bottom PGD, PGD_$l2$, PGAP and FPGAP at fixed number of iterations (70). Numbers on top of each image are ϵ and prediction of adversarial image respectively. Image on the right: Absolute difference maps with respect to original image. Numbers on top of each map are PSNR and SSIM.



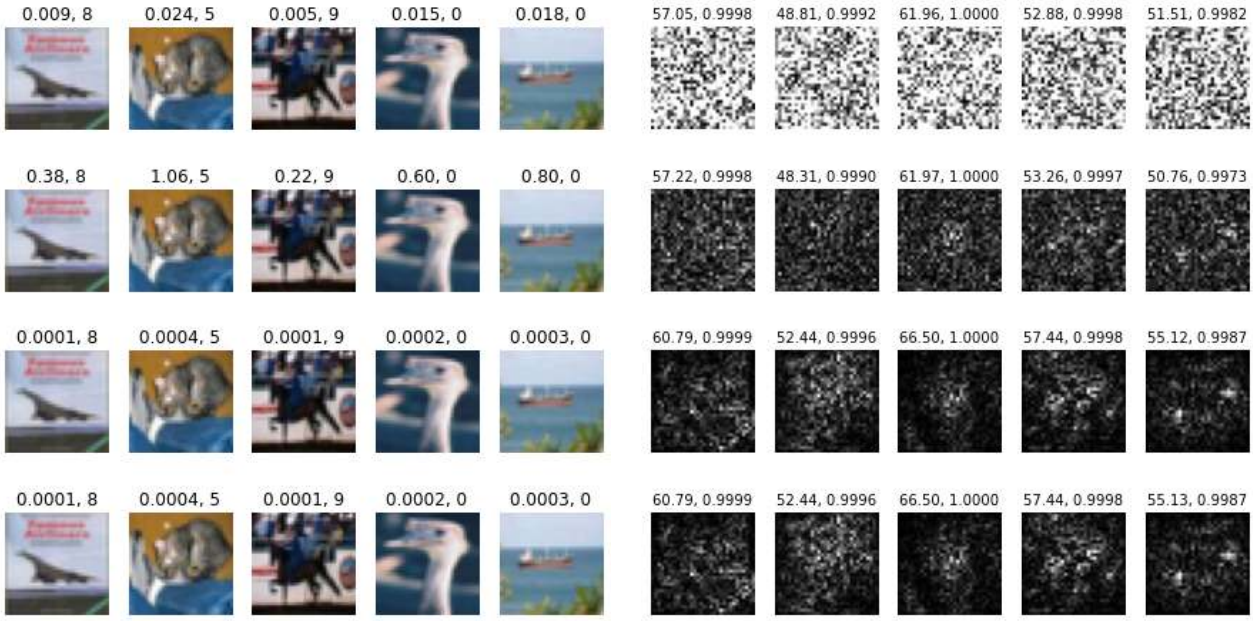
(a) Adversarial examples

(b) Absolute difference map of R channel

(c) Absolute difference map of G channel

(d) Absolute difference map of B channel

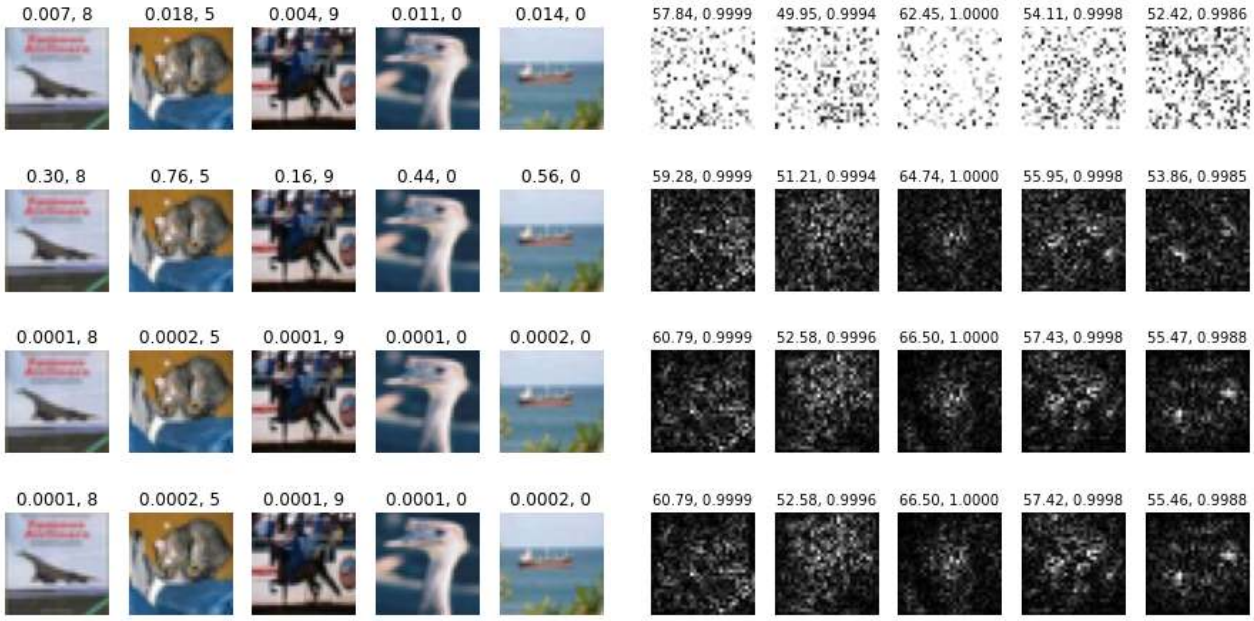
Fig. 23. (a): Adversarial examples generated by multiple methods: from top to bottom PGD, PGD_$l2$, PGAP and FPGAP at fixed number of iterations (10). Numbers on top of each image are ϵ and prediction of adversarial image respectively. (b): Absolute difference maps of channel R with respect to original image, Numbers on top of each map are PSNR and SSIM. (c): Absolute difference maps of channel G with respect to original image. (d): Absolute difference maps of channel B with respect to original image



(a) Adversarial examples (b) Absolute difference map of R channel

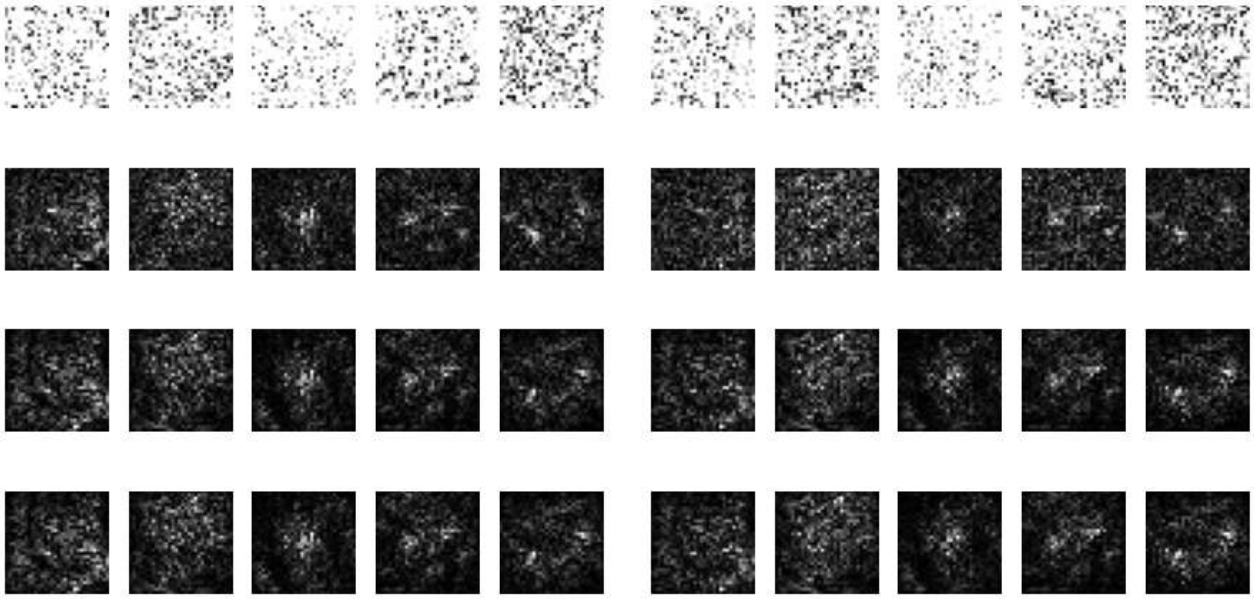
(c) Absolute difference map of G channel (d) Absolute difference map of B channel

Fig. 24. (a): Adversarial examples generated by multiple methods: from top to bottom PGD, PGD_$l2$, PGAP and FPGAP at fixed number of iterations (30). Numbers on top of each image are ϵ and prediction of adversarial image respectively. (b): Absolute difference maps of channel R with respect to original image, Numbers on top of each map are PSNR and SSIM. (c): Absolute difference maps of channel G with respect to original image. (d): Absolute difference maps of channel B with respect to original image



(a) Adversarial examples

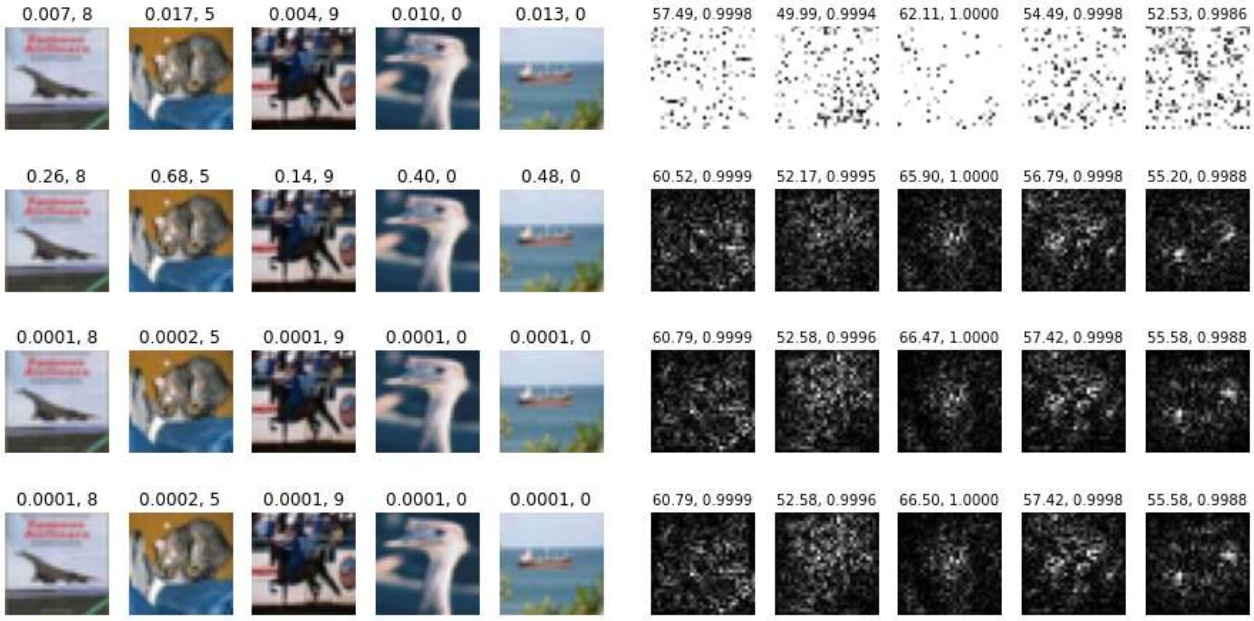
(b) Absolute difference map of R channel



(c) Absolute difference map of G channel

(d) Absolute difference map of B channel

Fig. 25. (a): Adversarial examples generated by multiple methods: from top to bottom PGD, PGD_$l2$, PGAP and FPGAP at fixed number of iterations (50). Numbers on top of each image are ϵ and prediction of adversarial image respectively. (b): Absolute difference maps of channel R with respect to original image, Numbers on top of each map are PSNR and SSIM. (c): Absolute difference maps of channel G with respect to original image. (d): Absolute difference maps of channel B with respect to original image



(a) Adversarial examples (b) Absolute difference map of R channel

(c) Absolute difference map of G channel (d) Absolute difference map of B channel

Fig. 26. (a): Adversarial examples generated by multiple methods: from top to bottom PGD, PGD_$l2$, PGAP and FPGAP at fixed number of iterations (70). Numbers on top of each image are ϵ and prediction of adversarial image respectively. (b): Absolute difference maps of channel R with respect to original image, Numbers on top of each map are PSNR and SSIM. (c): Absolute difference maps of channel G with respect to original image. (d): Absolute difference maps of channel B with respect to original image