

Center for Brains, Minds & Machines

CBMM Memo No. 44

19 January of 2016

Foveation-based Mechanisms Alleviate Adversarial Examples

by

Yan Luo Xavier Boix Gemma Roig Tomaso Poggio Qi Zhao

Abstract

We show that adversarial examples, *i.e.* the visually imperceptible perturbations that result in Convolutional Neural Networks (CNNs) fail, can be alleviated with a mechanism based on foveations—applying the CNN in different image regions. To see this, first, we report results in ImageNet that lead to a revision of the hypothesis that adversarial perturbations are a consequence of CNNs acting as a linear classifier: CNNs act locally linearly to changes in the image regions with objects recognized by the CNN, and in other regions the CNN may act non-linearly. Then, we corroborate that when the neural responses are linear, applying the foveation mechanism to the adversarial example tends to significantly reduce the effect of the perturbation. This is because, hypothetically, the CNNs for ImageNet are robust to changes of scale and translation of the object produced by the foveation, but this property does not generalize to transformations of the perturbation. As a result, the accuracy after a foveation is almost the same as the accuracy of the CNN without the adversarial perturbation, even if the adversarial perturbation is calculated taking into account a foveation.



This work was supported by the Center for Brains, Minds and Machines (CBMM), funded by NSF STC award CCF - 1231216.

FOVEATION-BASED MECHANISMS ALLEVIATE ADVERSARIAL EXAMPLES

Yan Luo¹ Xavier Boix^{1,2} Gemma Roig² Tomaso Poggio² Qi Zhao¹

¹ Department of Electrical and Computer Engineering, National University of Singapore, Singapore

² CBMM, Massachusetts Institute of Technology & Istituto Italiano di Tecnologia, Cambridge, MA

ABSTRACT

We show that adversarial examples, *i.e.*, the visually imperceptible perturbations that result in Convolutional Neural Networks (CNNs) fail, can be alleviated with a mechanism based on foveations—applying the CNN in different image regions. To see this, first, we report results in ImageNet that lead to a revision of the hypothesis that adversarial perturbations are a consequence of CNNs acting as a linear classifier: CNNs act locally linearly to changes in the image regions with objects recognized by the CNN, and in other regions the CNN may act non-linearly. Then, we corroborate that when the neural responses are linear, applying the foveation mechanism to the adversarial example tends to significantly reduce the effect of the perturbation. This is because, hypothetically, the CNNs for ImageNet are robust to changes of scale and translation of the object produced by the foveation, but this property does not generalize to transformations of the perturbation. As a result, the accuracy after a foveation is almost the same as the accuracy of the CNN without the adversarial perturbation, even if the adversarial perturbation is calculated taking into account a foveation.

1 INTRODUCTION

The generalization properties of CNNs have been recently questioned. Szegedy et al. (2014) showed that there are perturbations that when added to an image make CNNs fail to predict the object category in the image. Surprisingly, these perturbations are usually so small that they are imperceptible to humans. The perturbed images are the so-called *adversarial examples*.

Adversarial examples exist because, hypothetically, CNNs act as a linear classifier of the image (Goodfellow et al., 2015). The perturbation may be aligned with the linear classifier to counteract the correct prediction of the CNN, and since the dimensionality of the classifier is high, as it is equal to the number of pixels in the image, the perturbation can be spread among the many dimensions and make the perturbation of each pixel small, *c.f.* (Goodfellow et al., 2015; Fawzi et al., 2015). This hypothesis of CNNs acting too linearly is supported by quantitative experiments in datasets where the target object is always centered and fixed to the same scale and orientation, namely, MNIST (LeCun et al., 1998) and CIFAR (Krizhevsky, 2009).

In this paper, we challenge the current hypothesis by analyzing adversarial examples using several CNN architectures for ImageNet (Krizhevsky et al., 2012; Simonyan & Zisserman, 2015; Szegedy et al., 2015), which do not assume a positioning and scale of the object as in previous works. Our experiments show that the CNN does not act as a linear classifier, but the CNN acts locally linearly to changes in the image regions where there is an object recognized by the CNN, and in other regions the CNN may act non-linearly (Hypothesis 1). Also, the experiments show that the adversarial perturbations that are generated with a CNN different from the CNN used for evaluation, produce a much smaller decrease of the accuracy compared to the CNN architectures used for the datasets in previous works (Szegedy et al., 2014; Goodfellow et al., 2015).

This leads to the main hypothesis of this paper, which can be used to show that foveation mechanisms can substantially alleviate adversarial examples. A foveation mechanism selects a region of the image to apply the CNN, discarding information from the other regions. From the aforementioned local linearity hypothesis, it follows that the neural responses of a foveated object in the adversarial example can be decomposed into two independent terms, one that comes from the foveated clean

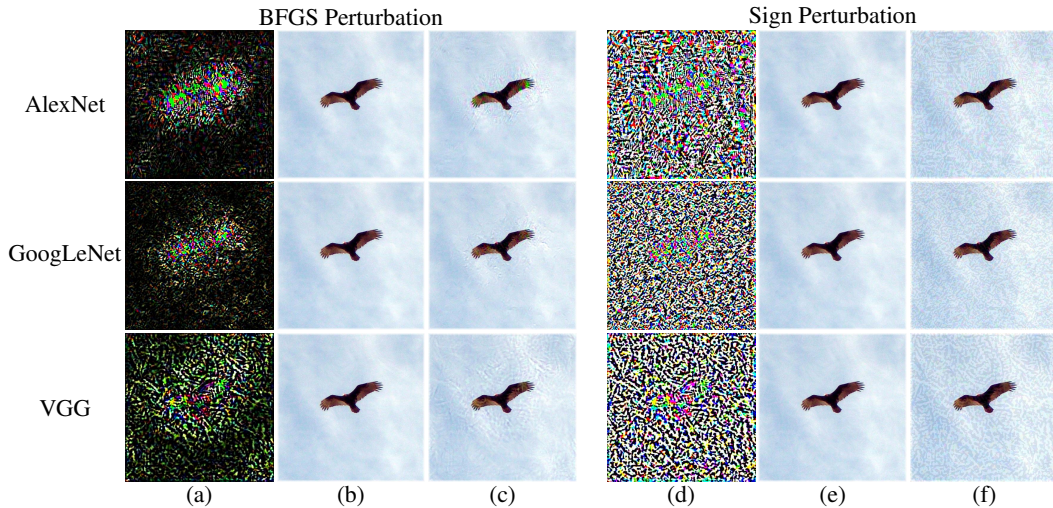


Figure 1: *Example of different CNNs’ minimum perturbations.* Each row corresponds to a different CNN. (a) *BFGS*, (b) its corresponding adversarial example, and (c) the adversarial example with the perturbation multiplied by 10; (d) *Sign*, (e) and (f) the same as (b) and (c), respectively, for *Sign*.

object without perturbation, and another from the foveated perturbation. The hypothesis—supported by a series of experiments—says that the term of the classification score of the clean image without perturbation is not negatively affected by a foveation that contains the target object in the image, as CNNs for ImageNet are trained to be robust to scale and position transformations of the object. Yet, this property does not generalize to transformations of the perturbation, and the effect of perturbation on the classification score after the foveation is lower than before (Hypothesis 2).

In a series of experiments under different conditions of clutter and using different kinds of foveations, we show that the foveation mechanisms on adversarial examples almost recover the accuracy performance of the CNN without the adversarial perturbation. This is also in the case of adversarial perturbations calculated taking into account a foveation, as we can always use a different foveation to exploit the Hypothesis 2.

2 EXPERIMENTAL SET-UP

In this section we introduce the experimental set-up that we use in the rest of the paper.

Dataset We use ImageNet (ILSVRC) (Deng et al., 2009), which is a large-scale dataset for object classification and detection. We report our results on the validation set of ILSVRC 2012 dataset, which contains 50’000 images. Each image is labeled with one of the 1’000 possible object categories, and the ground truth bounding box around the object is also provided. We evaluate the accuracy performance with the standard protocol (Russakovsky et al., 2015), taking the top 5 prediction error.

CNN Architectures We report results using 3 CNN which are representative of the state-of-the-art, namely AlexNet (Krizhevsky et al., 2012), GoogLeNet (Szegedy et al., 2015) and VGG (Simonyan & Zisserman, 2015). We use the publicly available pre-trained models provided by the Caffe library (Jia et al., 2014). We use the whole image as input to the CNN, resized to 227×227 pixels in AlexNet, and 224×224 in VGG and GoogLeNet. In order to improve the accuracy at test time, a commonly used strategy in the literature is to use multiple crops of the input image and then combine their classification scores. We do not use this strategy unless we explicitly mention it.

Generation of Adversarial Perturbations The adversarial perturbation consist in finding the perturbation with minimum norm that produces misclassification, under the top-5 criterion. This is an NP-hard problem, and the algorithms to compute the minimum perturbation are approximate. To generate the adversarial examples we use the two algorithms available in the literature. Namely, the algorithm by Szegedy et al. (2014), that is based on an optimization with L-BFGS, and we also use the algorithm by Goodfellow et al. (2015), that uses the sign of the gradient of the loss function. We

call them *BFGS* and *Sign*, respectively. For more details about the generation of the perturbation we refer to Sec. B, and to (Szegedy et al., 2014; Goodfellow et al., 2015).

Several qualitative examples of the perturbations are shown in Fig. 1. Observe that for *BFGS*, the perturbation is concentrated on the position of the object, while for *Sign*, the perturbation is spread through the image. Also, note that for each CNN architecture the perturbation is different.

When are Adversarial Perturbations Perceptible? The perceptibility of adversarial examples is subjective. We show examples of the same adversarial perturbation varying the L_1 norm per pixel, Fig. 15, 17, 19 in Sec. C, and the L_∞ norm, Fig. 16, 18, 20 in Sec. C. We can see that after the norm of the perturbation reaches a certain value, the perturbation becomes perceptible, and it starts occluding parts of the target object. We give an estimate of the minimum value of the norm that makes the adversarial perturbations perceptible by visually analyzing few hundreds of adversarial examples. For *BFGS*, we can see that when the L_1 norm per pixel is higher than 15 the perturbation has become slightly visible in almost all cases, and for L_∞ this value is 100. This difference between the two values is because the density of *BFGS* is not the same through all image, as commented before. For *Sign*, the threshold for both norms to make the perturbation visible is about 15 (here the value coincides for both because this perturbation is spread evenly through all the image). Note that this values are a rough estimation from qualitative observations and highly depend on the image and CNN used.

Positioning of the Target Object We use the positioning of the target object in the image as a tool to take into account the perturbation’s position in the image. We use the ground truth bounding box around the object provided in the ILSVRC 2012 dataset, which is manually annotated by a human.

3 REVIEW OF THE LINEARITY OF CNNs FOR IMAGENET

We now introduce the analysis of adversarial examples in CNNs for ImageNet. In this dataset, objects can be at different scales and positions, and as we will see, this difference with previous works leads to a revision of the hypotheses introduced in these works.

Let \mathbf{x} be an image that contains an object whose object category is of class label ℓ , and we denote as $f(\mathbf{x})$ the mapping of the CNN from the input image to the classification scores. We use ϵ to denote a perturbation that produces a misclassification when added to the input image, *i.e.*, $\ell \notin C(f(\mathbf{x} + \epsilon))$ in which $C(\cdot)$ maps the classification scores to class labels. The set of all perturbations that produce a misclassification is denoted as $\mathcal{E}_x = \{\epsilon \mid \ell \notin C(f(\mathbf{x} + \epsilon))\}$. Also, we define ϵ^* as the perturbation in \mathcal{E}_x with minimal norm, *i.e.*, $\epsilon^* = \arg \min_{\epsilon \in \mathcal{E}_x} \|\epsilon\|$, which may be imperceptible.

The causes of adversarial examples have been studied by Goodfellow et al. (2015); Fawzi et al. (2015), that showed that small perturbations can affect linear classifiers, and CNNs may act as a high-dimensional linear classifier (of the same dimensionality as the image size). The learned parameters of the CNN make that the CNN acts as a linear classifier, bypassing the non-linearities in the architecture. In this way, $f(\mathbf{x} + \epsilon^*)$ can be rewritten as $\mathbf{w}'\mathbf{x} + \mathbf{w}'\epsilon^*$, where \mathbf{w}' is the transpose of a high-dimensional linear classifier, and it yields a classifier approximately equivalent to $f(\cdot)$. Observe that the high-dimensionality of the classifier allows that even when $\mathbf{w}'\epsilon^*$ is a high value, the average per pixel value of ϵ^* may be low, since the perturbation can be spread among all the pixels, and make it imperceptible.

A direct consequence of the linearity of the CNN is that multiplying the adversarial perturbation by a constant value larger than 1 also produces misclassification, since $c\mathbf{w}'\epsilon^*$ is a factor c times larger than before. Thus, the set of adversarial examples, \mathcal{E}_x , is not only ϵ^* , but rather a dense set of perturbations around ϵ^* . Another phenomenon that can be explained through the linearity of the CNN is that adversarial examples generated for a specific CNN model produce misclassification in other CNNs (Szegedy et al., 2014). This may be because the different CNNs act as a linear classifier, that are very similar among them.

In the following, we review the linearity of CNNs in ImageNet. Then, we analyze the influence of the position of the perturbation in the image, which leads to a new hypothesis.

Review of the Properties of Adversarial Examples We now revisit the aforementioned properties of the set of adversarial examples, \mathcal{E}_x , that can be derived from CNNs acting too linearly. To analyze that multiplying adversarial perturbations by a factor also leads to misclassification, we measure the

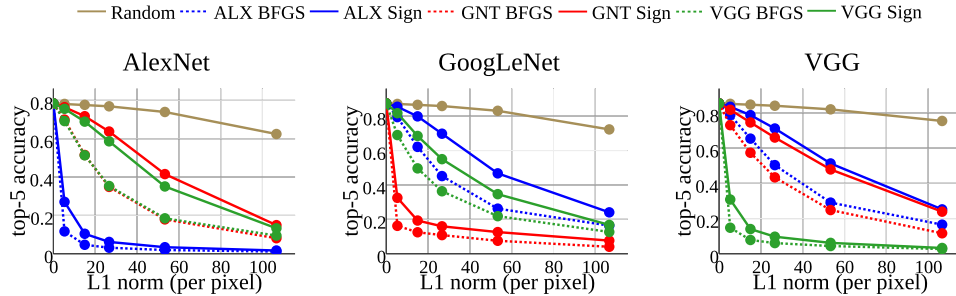


Figure 2: Accuracy when Varying the L_1 Norm per Pixel of the Perturbation. Accuracy for the three CNNs we evaluate. We denote the perturbation as $X Y$, where X is the network that generated the perturbation — ALX (AlexNet), GNT (GoogLeNet), VGG — and Y indicates the *BFGS* perturbation or *Sign*. See Fig. 6 in A.1 for the same results using L_∞ .

classification accuracy when we vary the norm of the perturbation. We use the L_1 norm per pixel and also the L_∞ norm, as *BFGS* and *Sign* perturbations are optimized for each of these norms respectively. At the same time, we also analyze the effect of adversarial perturbations generated on a CNN architecture different from the CNN we evaluate.

Results can be seen in Fig. 2 for L_1 norm per pixel. We additionally provide the accuracy for a perturbation generated by adding to each pixel a random value uniformly distributed in the range $[-1, 1]$. We observe a clear tendency that when we increase the norm of the perturbation, it results in more misclassification on average, which is in accordance to the hypothesis that CNNs are linear (Goodfellow et al., 2015). In Fig. 6 in Sec. A.1, the same conclusions can be extracted for L_∞ . Also, we observe that for the L_1 norm per pixel, *BFGS* produces more misclassification on average than *Sign*, and the opposite is true if we evaluate with L_∞ . This is not surprising as the *BFGS* optimizes the L_1 norm of the perturbation, and the *Sign* optimizes the maximum. Thus, in the rest of the experiments of the paper we only evaluate *BFGS* with the L_1 norm per pixel, and *Sign* with the L_∞ norm.

Interestingly, we observe that there is not an imperceptible adversarial perturbation for every image. Recall that an L_1 norm per pixel roughly higher than 15, and L_∞ also roughly higher than 15 produce a perturbation slightly perceptible (Sec. 2). Thus, the results in Fig. 2 show that the accuracy is between 10% and 20%, depending on the CNN, when the adversarial perturbations are constrained to be no more than slightly perceptible.

Finally, we see that for a particular CNN architecture, the perturbation that results in more misclassification with lower norm is in all cases the one that is generated using the same CNN architecture. Observe that the perturbation generated for a CNN architecture, does not affect so severely the other CNN architectures as reported in MNIST, *c.f.* (Szegedy et al., 2014). When the adversarial perturbations are constrained to be no more than a bit perceptible, the accuracy is between 40% and 60% in the worst case, depending on the CNN. This may be because the differences among the learned CNNs in ImageNet are bigger than the differences among the CNNs learned in MNIST.

Role of the Target Object Position in the Perturbation We now analyze the effect of the perturbation at different regions of the image. We report the accuracy performance that yields the perturbation computed for the whole image when it is masked to be positioned only on the target object, or when it masked to be only on the background. Specifically, we create an image mask using the ground truth bounding box (denoted as *Object Masked* in the figures), where the mask’s pixels are 1 inside the bounding box, and 0 otherwise. Before the perturbation computed for the whole image is added to the image, we multiply (pixel-wise) the perturbation by the mask, such that the resulting perturbation is only positioned inside the bounding box. To analyze the perturbation that is in the background, we also create the inverted mask, denoted as *Background Masked*, in which the pixels outside the bounding box are 1, and 0 for the pixels inside.

In Fig. 3a we report the accuracy performance for *BFGS* for *Object Masked* and *Background Masked*, and also for the perturbation applied in the full image generated for the AlexNet CNN (*minimum perturbation (MP)*). See Fig. 7 in A.1 for the results for each CNN architecture and *Sign* perturbation. We observe that the accuracy performance for the perturbation positioned only on the

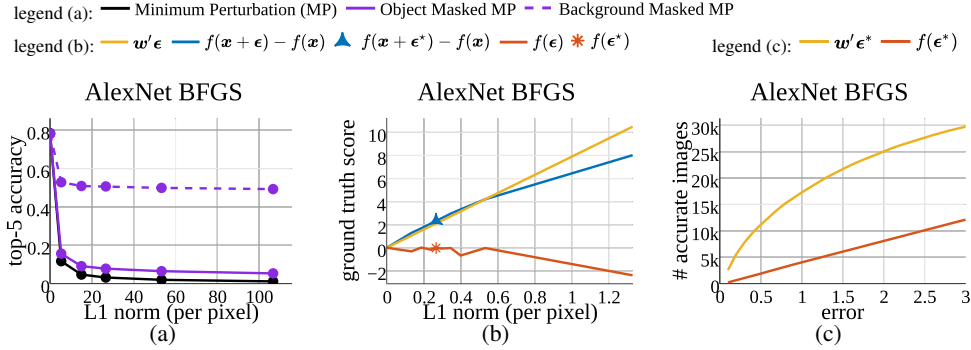


Figure 3: *Experiments for the Local Linearity Hypothesis.* (a) Accuracy of the masked perturbations for the AlexNet CNN, when varying the norm of *BFGS*. See Fig. 7 in A.1 for the results with *Sign* and L_∞ norm for all CNNs we evaluate; (b) Classification score of the ground truth object category for the image in Fig. 1, when varying the L_1 norm per pixel of *BFGS*. See Fig. 8, 9, 10 in A.1 for more examples, and also *Sign* with L_∞ norm; (c) Cumulative histogram of the number of images with an L_1 error smaller than the value in the horizontal axis for *BFGS* computed for AlexNet. Only the images that are correctly classified by the CNN are included. See Fig.11 in A.1 for the results of all CNNs.

object decreases in a similar way as *MP*. When adding the perturbation only on the background, the accuracy decreases significantly less than for *Object Masked* (there is a difference between them of about 40% of accuracy). Note that this result is also clear for small values of the perturbation norm, *i.e.*, the values that are imperceptible.

From the hypothesis that the CNN acts as a linear classifier, the result of this experiment suggests that the perturbation, ϵ^* , is aligned with the classifier, w , on the same position of the target object, otherwise there is not an alignment between them. However, if the CNN is linear, to produce misclassification, ϵ^* could be aligned with w in any position, because $w'\epsilon^*$ only needs to surpass the value of $w'x$, independently of the image, x . This begs the question why there is such a clear relationship between ϵ^* and x . An answer could be that for the algorithms that generate the adversarial perturbation (*BFGS* and *Sign*), it is easier to find perturbations on the position of the object than in other positions, and hypothetically, we could also find adversarial perturbations that are aligned with the classifier in positions different from the object. Yet, this needs to be verified. The next experiment clarifies this point.

CNNs act Locally Linearly on the Positions of a Recognized Object We test if CNNs act as a linear classifier by evaluating whether $f(x + \epsilon^*) = f(x) + f(\epsilon^*)$ holds in practice. We evaluate $f(\epsilon^*)$ and $f(x + \epsilon^*) - f(x)$ for each image, varying the norm of ϵ^* , for the ground truth object category. We remove from the CNNs the last soft-max layer, because this layer may distort the results of evaluating the linearity, and the accuracy of the CNN at test time is the same (the soft-max preserves the ranking of the classification scores). In Fig. 3b, we show an example for *BFGS* and AlexNet CNN for the image of Fig. 1. We indicate the point at which the perturbation produces the misclassification, ϵ^* . We show more examples for *BFGS* and *Sign* in Fig. 8, 9, 10 in A.1. We can see that $f(\epsilon^*)$ is completely unrelated to $f(x + \epsilon^*) - f(x)$, but there is some local linearity in the vicinity of the image without perturbation, x , in the direction of ϵ^* . Thus, the hypothesis that the CNNs behave as a linear classifier needs to be reviewed in order to reconcile the hypothesis with these results. We introduce the next hypothesis that is able to explain all the previous results.

Hypothesis 1 *CNNs are locally linear to changes on the regions of the image that contain objects recognized by the CNN, otherwise the CNN may act non-linearly.*

We use the expression “locally linear” to denote that the linearity of the CNN only holds for perturbations of x that are small, *i.e.*, images that are in the vicinity of x . Also, note that the local linearity is only for perturbations on the image regions of x that cause the CNN to recognize an object. The Hypothesis 1 leaves open whether the CNN behaves locally linearly or not in image regions that do not cause the CNN to detect an object, as we have not empirically analyzed what type of objects that are not detected by the CNN may produce a locally linear behavior of the CNN.

As in previous works, the hypothesis suggests that adversarial examples are a consequence of CNNs acting as a high-dimensional linear classifier. In our hypothesis this is because the CNN acts as a linear classifier in the vicinity of the images with recognized objects. Also, our hypothesis predicts the result in Fig. 3b about why $f(\epsilon^*)$ is not equal to $f(\mathbf{x} + \epsilon^*) - f(\mathbf{x})$, because $f(\epsilon^*)$ is in the non-linear region of the CNN since ϵ^* is not an image of an object that is recognized by the CNN. Finally, our hypothesis explains why the perturbation is always aligned with the classifier on the position of the recognized object, as in a different position the CNN may not behave linearly.

We provide quantitative results in Fig. 3c to show that the local linearity hypothesis reasonably holds in the vicinity of \mathbf{x} . We approximate the alignment of the perturbation with an hypothetical linear classifier, *i.e.*, $\mathbf{w}'\epsilon$, and we evaluate how much the classification score given by this hypothetical linear classifier deviates from the real classification score, $f(\mathbf{x} + \epsilon) - f(\mathbf{x})$. To calculate $\mathbf{w}'\epsilon$ we trace a line between $f(\mathbf{x})$ and $f(\mathbf{x} + 2\epsilon^*)$, which gives $\mathbf{w}'\epsilon$ in the direction of ϵ^* (see the yellow line in Fig. 3c). If the CNN would approximately act as a linear classifier between \mathbf{x} and $\mathbf{x} + 2\epsilon^*$, then the error between $\mathbf{w}'\epsilon^*$ and $f(\mathbf{x} + \epsilon^*) - f(\mathbf{x})$ would be 0. In Fig. 11c we report the cumulative histogram of the number of images with an error derived from the Hypothesis 1 smaller than a threshold, and we compare it with the error that yields calculating $f(\mathbf{x} + \epsilon^*) - f(\mathbf{x})$ as $f(\epsilon^*)$, which comes from the assumption that the CNN is always a linear classifier. We can see that the error derived from the Hypothesis 1 is orders of magnitude smaller than the error from the hypothesis that the CNN is always a linear classifier.

Note that the Hypothesis 1 can be motivated from the non-linearities induced by the linear rectifier units (ReLU) (Krizhevsky et al., 2012), which are used in all CNNs we tested. ReLUs are applied to the neural responses after the convolutional layers of the CNNs. They have a linear behavior only for the active neural responses, otherwise they output a constant value equal to 0. Thus, if the perturbation is added in a region where the neural responses are active, the CNN tends to behave more linearly than when the neural responses are not active. When we increase the norm of the perturbation, the effect of the perturbation to the final classification score ($f(\mathbf{x} + \epsilon^*) - f(\mathbf{x})$) stops increasing at the same linear pace because the number of ReLUs that return a 0 value is higher than before increasing the norm of the perturbation. This can be seen in Fig. 3b, and Fig. 8, 9, 10 in A.1, when we increase the norm of the perturbation.

4 FOVEATION-BASED MECHANISMS ALLEVIATE ADVERSARIAL EXAMPLES

We define a foveation as a transformation of the image that selects a region in which the CNN is applied, discarding the information from the other regions. The foveation provides an input to the CNN that always has the same size, even if the size of the selected region varies. Thus, the foveation may produce a change of the position and scale of the objects in the input of the CNN. It is well-known that the representations learned by the CNNs are robust to changes of scale and position of the objects, and we assume that the transformation produced by the foveation does not negatively affect the performance of the CNN. Also, we assume that the foveation mechanism does not introduce non-linearities, *e.g.*, it does not modify pixel values from the original image, and the interpolations for re-sizing the image are linear. Without loss of generality, we use as the foveation mechanism a crop of a region that includes most of the object or the whole object. Below, the details of several foveation mechanism we use are introduced.

Let $T(\mathbf{x})$ be the image after the foveation. Recall the local linearity hypothesis introduced in Hypothesis 1, in the previous section, and recall the linearity of $T(\cdot)$. If the perturbation ϵ is applied in the same position of an object recognized by the CNN, we have that hypothetically, $f(T(\mathbf{x} + \epsilon^*))$ is equal to $\mathbf{w}'T(\mathbf{x}) + \mathbf{w}'T(\epsilon^*)$ for small perturbations. Since the representations learned by the CNNs are robust to changes of scale and position of the objects produced by the foveation, the term without the perturbation after the foveation, $\mathbf{w}'T(\mathbf{x})$, is not expected to have a lower accuracy than before, $\mathbf{w}'\mathbf{x}$. In addition, since $T(\cdot)$ may remove clutter from the image, applying the foveation could even improve the accuracy when there is not the perturbation, if $T(\mathbf{x})$ does not remove any part of the target object. Yet, the main reason why foveations may improve the performance of CNNs for adversarial examples is due to the hypothesis we now introduce:

Hypothesis 2 *The aforementioned robustness of the CNNs to changes of scale and position of the objects does not generalize to the perturbations.*

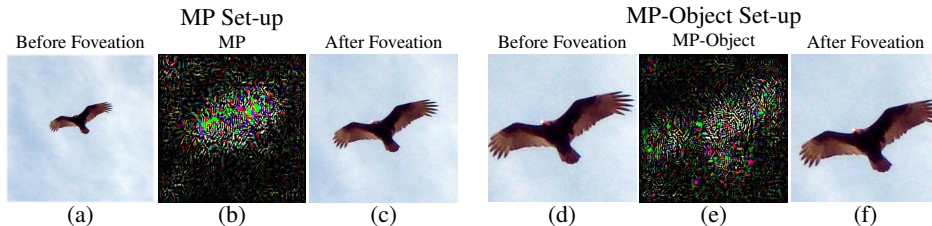


Figure 4: *Example of the two set-ups with AlexNet with BFGS perturbation.* (a) Adversarial example *MP*, (b) adversarial perturbation *MP*, (c) *Saliency Crop MP*; (d) adversarial example *MP-Object*, (e) adversarial perturbation *MP-Object*, and (f) 1 *Shift MP-Object*.

Thus, the foveation mechanism reduces the alignment between the classifier and the perturbation, $\mathbf{w}'T(\epsilon^*)$, but does not negatively affect the alignment between the classifier and the image without perturbation, $\mathbf{w}'T(\mathbf{x})$. Note that CNNs are trained with objects at many different positions and scales, that produce representations robust to the transformations of the object, such as the transformation produced by $T(\cdot)$. However, objects are visually very different from the adversarial perturbations, and the robustness of CNNs to transformations of objects does not generalize to the perturbations.

In the following, we show experimental evidence that supports the Hypothesis 2, and show that foveations can be used to greatly alleviate the effect of the adversarial examples.

4.1 FOVEATION MECHANISMS USED IN EXPERIMENTS

A foveation may increase the accuracy of the CNNs by removing clutter rather than alleviating the effect of the adversarial perturbation. To analyze the effect of clutter, we use two different experimental set-ups.

MP set-up (with clutter) In the first set-up, we use the adversarial example used in the previous section (see Fig. 4a and b). We use the following foveations:

- *Object Crop MP*: We use as foveation mechanism the crop of the target object using the ground truth bounding box. If there are multiple bounding boxes in the image, because there are multiple target objects, we crop each of them, and average the classification scores. Note that this foveation mechanism does not remove any part of the object, and it removes most of the clutter. See the purple bounding box in Fig. 12 in A.2. The rest of the foveations we use do not guarantee that all the clutter is removed, and also, they may remove part of the target object.

- *Saliency Crop MP*: This foveation is based on using a state-of-the-art saliency model to select 3 regions to crop from the most salient locations of the image, depicted in Fig. 4c and the cyan bounding box in Fig. 12 in A.2. The crops are generated selecting three centroids of the saliency map. We use the SALICON saliency map, which extracts the saliency map using the same CNNs we test (Huang et al., 2015). We observed that these saliency maps are robust to the adversarial perturbations, since the adversarial examples are generated to produce misclassification for object recognition, but not to affect the saliency prediction. The classification accuracy is computed by averaging the confidences from the multiple crops.

- *10 Crop MP*: Another foveation strategy we evaluate is the 10 crops that is implemented in the Caffe library to boost the accuracy of the CNNs (Jia et al., 2014). The crops are done on large regions of the image, which in most cases the target object is contained in the crops (each crop discards about 21% of the area of the image). The crops are always of the same size, and there are 5 of them (4 clamped at each corner of the image, and one in the center of the image). The images resulting from these 5 crops are flipped, which makes a total of 10 crops.

- *3 Crop MP*: This is the same as *10 Crop MP* but only with 3 random crops selected among the 10 crops. It can be used for a fair comparison between *Saliency Crop MP* with *10 Crop MP*, and to evaluate how much improvement comes from averaging multiple foveations.

MP-Object set-up (without clutter) In the second set-up, we generate the adversarial example for the image produced after cropping the target object with the ground truth bounding box (denoted as

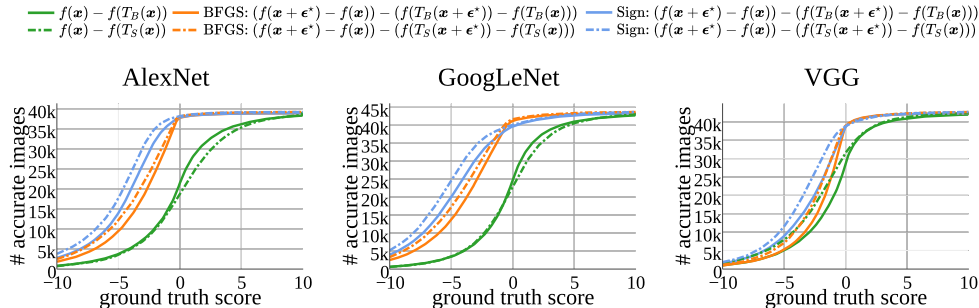


Figure 5: *Effect of the Foveation to the Perturbations.* Cumulative histogram of the number of images with a change of classification score smaller than the indicated in the horizontal axis. $T_B(\cdot)$ is the foveation with *Object Crop MP*, and $T_S(\cdot)$ is the foveation with 1 *Shift MP-Object*. Only the images that are correctly classified by the CNN are included.

MP-Object, see Fig. 4d and e, and Fig. 13). Note that *MP-Object* is the adversarial example for an image with almost no clutter. The foveations for this set-up are:

- *Embedded MP-Object*: It consists on embedding the image of the cropped object with the perturbation (*MP-Object*) to the full image. Note that this foveation adds the clutter back to the image.

- *10 Shift MP-Object*: It is based on the 10 crops of Caffe we introduced before for 10 *Crop MP*. In this case, each of the 10 crops is a shifted version of the target object, that removes part of the object and uses part of the background to do the crop (the foveation shifts a bit to the background, and it yields a cropped region with 12% of background, as shown in Fig. 4f and Fig. 12 in Sec. A.2). We set the size of the crops such that they do not modify the original scale of the object.

- *1 Shift MP-Object*: It consists on selecting 1 random crop from the 10 shifts of 10 *Shift MP-Object*.

4.2 RESULTS

We now show that a foveation can produce significant improvements of the CNN accuracy with adversarial examples, and also that the impact of removing clutter or averaging the predictions of multiple foveations is lower than the effect of our Hypothesis 2.

Evaluation of the Hypothesis 2 We now test the hypothesis by analyzing the effect of the foveation to the classification scores of the adversarial perturbation and of the image without the perturbation. Namely, we evaluate $f(\mathbf{x}) - f(T(\mathbf{x}))$ (how the classification score varies after the foveation for the image without perturbation), and $(f(\mathbf{x} + \epsilon^*) - f(\mathbf{x})) - (f(T(\mathbf{x} + \epsilon^*)) - f(T(\mathbf{x})))$ (how the classification score varies after the foveation for the adversarial perturbation). We do so for the classification score for the ground truth object category (without the last soft-max layer, as in Sec. 3), and for both set-ups. In Fig. 5, we show the cumulative histogram of the number of images with a change of the classification score smaller than the indicated in the horizontal axis in the figure. We can see that the alignment between the perturbation and the classifier decreases after the foveation, and that the term of the image without perturbation is not affected as much as the term of the perturbation (*BFGS* and *Sign* are compared independently), which strongly supports Hypothesis 2.

Accuracy of the adversarial examples after the foveation Table 1 shows the accuracy before and after the foveation for the adversarial examples with the perturbation with minimum norm. In all cases, we can see that the accuracy after the foveation is almost the same as the accuracy without the adversarial perturbation, suggesting that foveations are a powerful mechanism to alleviate the adversarial examples (it improves the accuracy from 0% to more than 70% in all cases). *Object Crop MP* produces the biggest improvement over all foveation mechanisms, probably because it is the only foveation mechanism that guarantees that parts of the target object are not removed, and at the same time it removes the clutter. Note that the accuracy for the adversarial examples is not exactly 0% because in some images the minimum norm of the perturbation is out of the range of the line search of the algorithm to calculate *BFGS* and *Sign* perturbations.

How much does removing clutter or average the result of multiple foveations improve the accuracy? In Table 3 in Sec. A.2 we show the accuracy after the foveation when there is

Table 1: *Evaluation of the Foveation Mechanisms.* Quantitative results of the top-5 accuracy with minimum perturbation. See Table 2 in A.2 for the results with *Sign*.

	BFGS Minimum Perturbation							
	Before Foveation		After Foveation		Before Foveation		After Foveation	
	w/o MP	MP	Object Crop MP	Saliency Crop MP	w/o MP-Object	MP-Object	1 Shift MP-Object	Embedded MP-Object
ALX	0.7841	0.0048	0.7804	0.7076	0.8192	0.0175	0.7683	0.7514
GNT	0.8736	0.0104	0.8313	0.7875	0.8939	0.0284	0.8329	0.8418
VGG	0.8536	0.0055	0.8258	0.8162	0.9122	0.0308	0.8151	0.7806

no adversarial perturbation. We can see that the foveation slightly improves the accuracy in the absence of adversarial perturbation (about 5%), when we crop the bounding box or average over multiple crops. This suggests that removing clutter and averaging multiple crops always improve the accuracy, independently of the Hypothesis 2. However, note that the improvement from removing clutter can not be the main reason of the improvement of the accuracy by the foveation when there is an adversarial perturbation, since in the set-up of *MP-Object*, in which there is no clutter, the accuracy significantly improves after the foveation, and this improvement is almost the same as for the foveations that remove clutter in the *MP* set-up. Also, we can see that the improvement of the accuracy from averaging multiple foveations is relatively smaller by comparing the accuracy of 1 and 10 *Shift MP-Object*, and the accuracy of 1 and 10 *Crop MP*.

How much does the accuracy decrease when the norm of the adversarial perturbation is increased? In Fig. 14 in A.2 we report the accuracy using different values of the norm of the perturbation before the foveation. When the perturbations are imperceptible or almost imperceptible, all the foveation mechanisms we introduce improve the accuracy between 30% to 40% in both set-ups, which further confirms that foveations considerably alleviate adversarial examples. Note that this result can not be directly compared with Table 1, because here the norm of the perturbation is fixed for all images, while in Table 1 uses the minimum norm for each image. Fig. 14 also shows that when we increase the value of the norm of the perturbation, the accuracy decreases. This is because large perturbations bring the CNN to the non-linear region (Sec. 3), and our hypothesis of the foveations assumes that the CNN is working in the linear region. Also, when we increase the norm of the perturbation, CNNs fail because the perturbation is occluding the object rather than acting as an adversarial perturbation. We report the increase of the norm of the adversarial perturbation after the foveation in Table 4 in A.2. Note that after the foveation the norm of the perturbation has to substantially increase to produce misclassification (about 5 times for *BFGS*, and between 5 to 8 times for *Sign*).

Is there an adversarial perturbation for a CNN that already uses foveations? Table 1 shows that this is the case, as the accuracy of the adversarial examples *MP* is very similar to the adversarial example for the cropped object, *MP-Object*. However, our results also show that we can apply a foveation mechanism different from the foveation mechanism used to generate the adversarial perturbation, and improve again the accuracy. For example, 1 *Shift MP-Object* and *Embedded MP-Object* substantially improve the accuracy of the adversarial perturbation calculated to affect the crop of the target object (*MP-Object*).

Previous works to alleviate adversarial perturbations are based on training a CNN that learns to classify adversarial examples generated at the training phase (Gu & Rigazio, 2015; Goodfellow et al., 2015). Analogously to the foveations, new adversarial examples can be generated for the CNN that has been re-trained with adversarial examples. Yet, with the foveations, at testing phase a different foveation mechanism from the one used to generate the adversarial example could be exploited, as we have previously shown, but we can not re-train the CNN at testing phase. Thus, foveations are a promising tool to design object recognition systems robust to adversarial examples.

5 CONCLUSIONS

Adversarial examples are a consequence of CNNs acting as a high-dimensional linear classifier in the vicinity of the images with objects recognized by the CNN. Also, the transformation of the image produced by a foveation decreases the effect of the adversarial perturbation to the classification

scores, because the robustness of the CNNs to transformations of objects does not generalize to the perturbations.

This suggests that a system similar to Mnih et al. (2014), which integrates information from multiple fixations for object recognition, may be more robust to the phenomenon of the adversarial examples. Note that this system, which is based on integrating information from several image regions, is more similar to human vision than current CNN architectures. Yet, a remaining puzzle is the robustness of human perception to the perturbations. Some hints towards an answer could be that humans fixate their eyes on salient regions, and the eccentricity dependent resolution of the retina help eliminate the background outside the eye fixation.

ACKNOWLEDGMENTS

This work was supported by the Singapore Defense Innovative Research Program 9014100596, Ministry of Education Academic Research Fund Tier 2 MOE2014-T2-1-144, and also, by the Center for Brains, Minds and Machines (CBMM), funded by NSF STC award CCF-1231216.

REFERENCES

- Deng, Jia, Dong, Wei, Socher, Richard, Li, Li-Jia, Li, Kai, and Fei-Fei, Li. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.
- Fawzi, Alhussein, Fawzi, Omar, and Frossard, Pascal. Fundamental limits on adversarial robustness. In *ICML*, 2015.
- Goodfellow, Ian J., Shlens, Jonathon, and Szegedy, Christian. Explaining and harnessing adversarial examples. In *ICLR*, 2015.
- Gu, Shixiang and Rigazio, Luca. Towards deep neural network architectures robust to adversarial examples. In *ICLR*, 2015.
- Huang, Xun, Shen, Chengyao, Boix, Xavier, and Zhao, Qi. SALICON: Reducing the semantic gap in saliency prediction by adapting neural networks. In *ICCV*, 2015.
- Jia, Yangqing, Shelhamer, Evan, Donahue, Jeff, Karayev, Sergey, Long, Jonathan, Girshick, Ross, Guadarrama, Sergio, and Darrell, Trevor. Caffe: Convolutional architecture for fast feature embedding. In *MM*, 2014.
- Krizhevsky, Alex. Learning multiple layers of features from tiny images. *Tech. Rep.*, 2009.
- Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.
- LeCun, Yann, Bottou, Léon, Bengio, Yoshua, and Haffner, Patrick. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- Mnih, Volodymyr, Heess, Nicolas, Graves, Alex, and Kavukcuoglu, Koray. Recurrent models of visual attention. In *NIPS*, 2014.
- Russakovsky, Olga, Deng, Jia, Su, Hao, Krause, Jonathan, Satheesh, Sanjeev, Ma, Sean, Huang, Zhiheng, Karpathy, Andrej, Khosla, Aditya, Bernstein, Michael S., Berg, Alexander C., and Fei-Fei, Li. Imagenet large scale visual recognition challenge. *IJCV*, 2015.
- Simonyan, Karen and Zisserman, Andrew. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.
- Szegedy, Christian, Zaremba, Wojciech, Sutskever, Ilya, Bruna, Joan, Erhan, Dumitru, Goodfellow, Ian J., and Fergus, Rob. Intriguing properties of neural networks. In *ICLR*, 2014.
- Szegedy, Christian, Liu, Wei, Jia, Yangqing, Sermanet, Pierre, Reed, Scott, Anguelov, Dragomir, Erhan, Dumitru, Vanhoucke, Vincent, and Rabinovich, Andrew. Going deeper with convolutions. In *CVPR*, 2015.

SUPPLEMENTARY MATERIAL

A RESULTS

A.1 THE LINEARITY OF CNNs FOR IMAGENET

Review of the Properties of Adversarial Examples in ImageNet In Fig. 6, we extend the results of Fig. 2 in the paper with the results for *Sign perturbation* using L_∞ norm. We observe the same tendency than in Fig. 2.

Role of the Target Object Position in the Perturbation In Fig. 7 we report the accuracy performance of the different CNN architectures for *BFGS*, and *Sign* using L_∞ norm, when changing the norm of the perturbation that is inside the mask, or the inverted mask, and the result for the perturbation applied in the full image (denoted as *minimum perturbation (MP)*). We make the same observations as Fig. 3a in the paper.

CNNs act Locally Linear on the Positions of a Recognized Object In Fig. 8, 9, 10 we show more examples as the one in Fig. 3b in the paper, and for *BFGS* and *Sign* perturbations, for the images in Fig. 15, 17 and 19, in Sec. C, respectively. We can see that $f(\epsilon^*)$ is completely unrelated to $f(\mathbf{x} + \epsilon^*) - f(\mathbf{x})$, and we need to review the hypothesis that the CNNs behave too linearly in order to reconcile the hypothesis with these results.

In Fig. 11, we approximate the alignment of the perturbation with the hypothetical linear classifier, *i.e.*, $\mathbf{w}'\epsilon$, and we evaluate how much this hypothetical linear classifier deviates from the real classification score, $f(\mathbf{x} + \epsilon) - f(\mathbf{x})$, for all evaluated CNNs and perturbations. As observed in Fig. 3c in the paper for AlexNet using *BFGS*, we can see that the error derived from our hypothesis is much smaller than from the hypothesis that the CNN is always a linear classifier.

A.2 FOVEATION-BASED MECHANISMS ALLEVIATE ADVERSARIAL EXAMPLES

Accuracy of the adversarial examples after the foveation Table 2 shows the accuracy before and after the foveation for the adversarial examples with the perturbation with minimum norm using *BFGS* and *Sign* perturbations. We can extract the same conclusions as in Table 1 in the paper.

How much does removing clutter or average the result of multiple foveations improve the accuracy? In Table 3 we show the accuracy after the foveation when there is no adversarial perturbation, and when the adversarial perturbation has norm equal to 5.3. As we have explained in the paper, we can see that the impact of removing clutter or averaging multiple foveations is much smaller than the impact of Hypothesis 2.

How much does the accuracy decrease when the norm of the adversarial perturbation is increased? In Fig. 14 we report the accuracy using different values of the norm of the perturbation before the foveation. As explained in the paper, when we increase the value of the norm of the perturbation, the accuracy decreases because the CNN may enter to the non-linear region due to the occlusions caused by the perturbation.

Finally, in Table 4, we report the increase of the norm of the adversarial perturbation after the foveation, which is about 5 times for *BFGS*, and between 5 to 8 times for *Sign*.

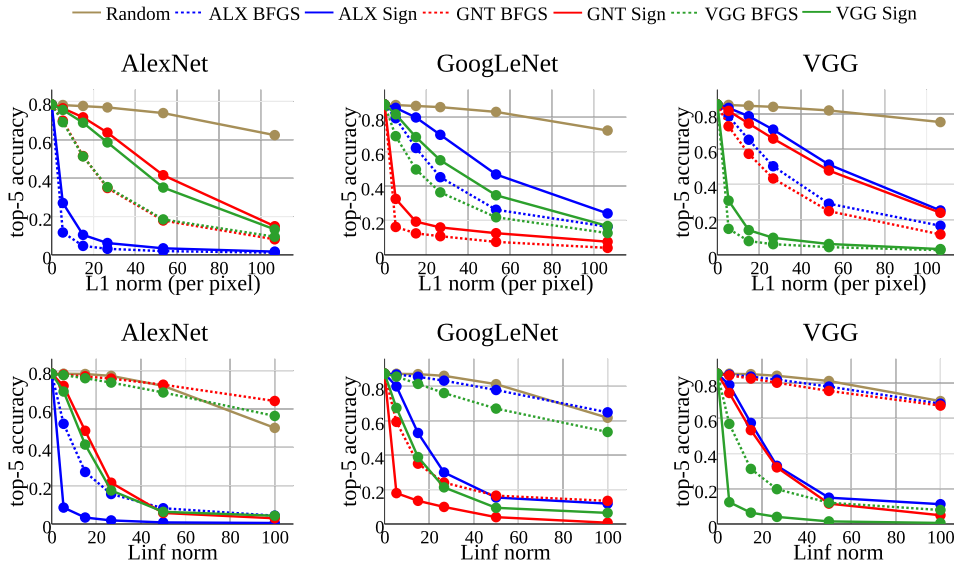


Figure 6: Accuracy when Varying the L_∞ Norm of the Perturbation. Accuracy for the three CNNs we evaluate. We denote the perturbation as XY , where X is the network that generated the perturbation — ALX (AlexNet), GNT (GoogLeNet), VGG — and Y indicates the *BFGS* or *Sign*.

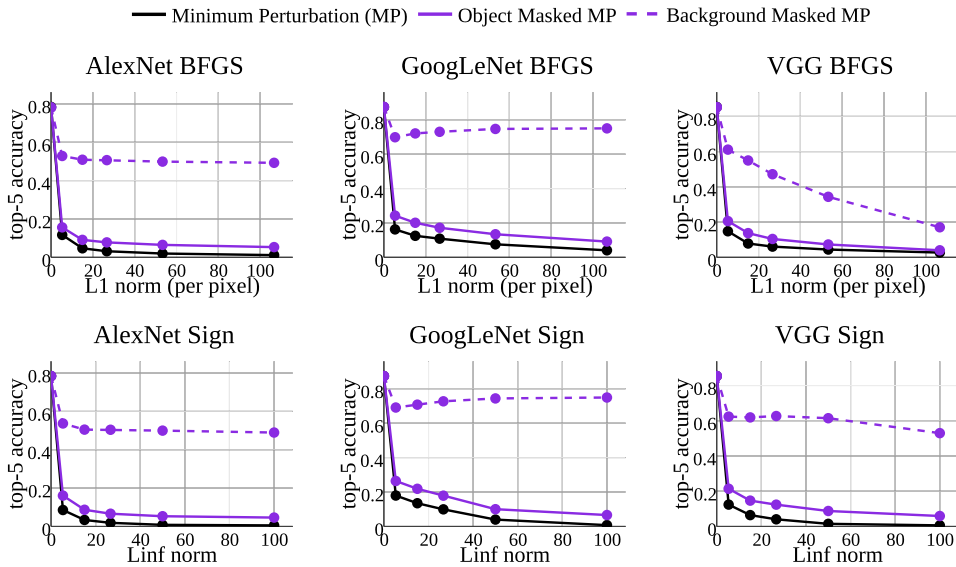


Figure 7: Accuracy of the Masked Perturbations. Accuracy for the three CNNs we evaluate, when varying the norm of *BFGS* (first row) and *Sign* (second row).

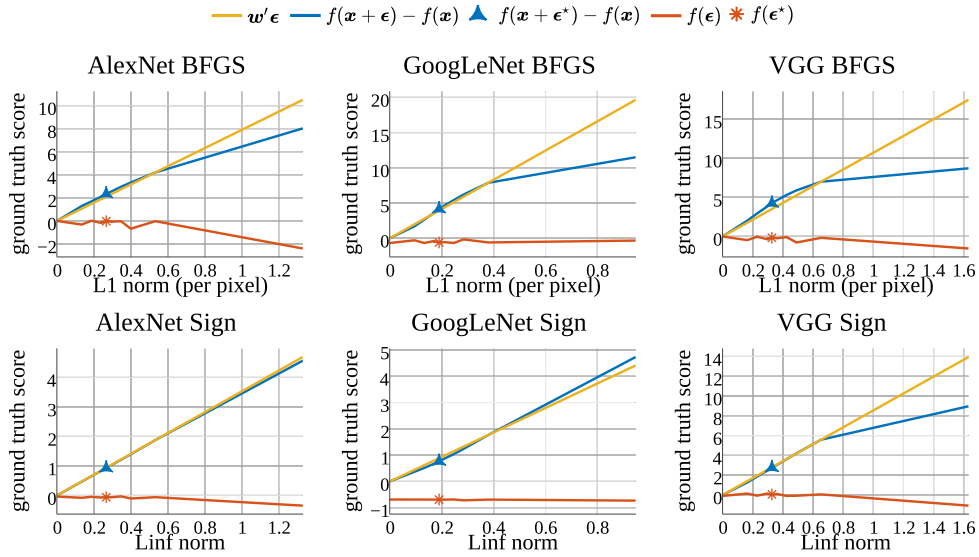


Figure 8: *Local Linearity of the CNNs*. Classification score of the ground truth object category for the image in Fig. 15, when varying the L_1 norm per pixel of *BFGS* and L_∞ for *Sign*.

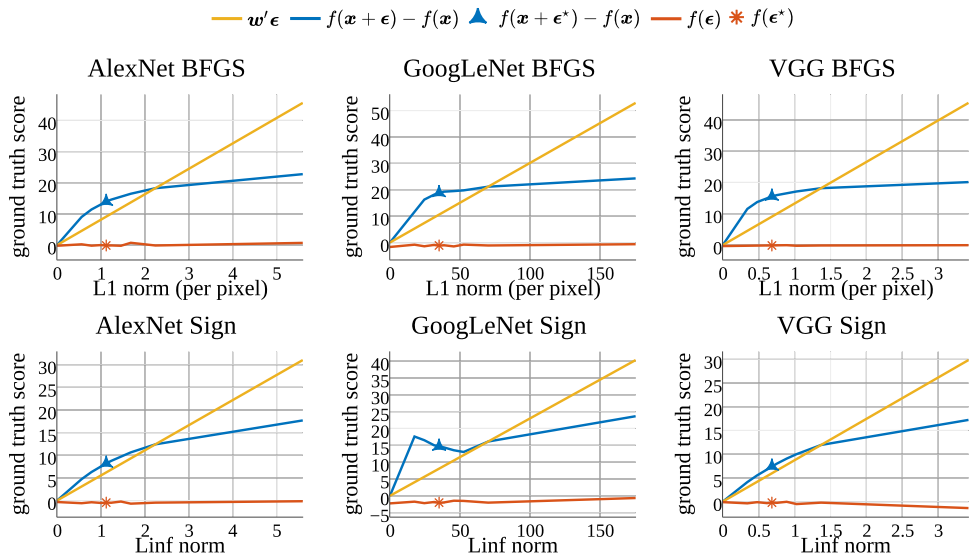


Figure 9: *Local Linearity of the CNNs*. Classification score of the ground truth object category for the image in Fig. 17, when varying the L_1 norm per pixel of *BFGS* and L_∞ for *Sign*.

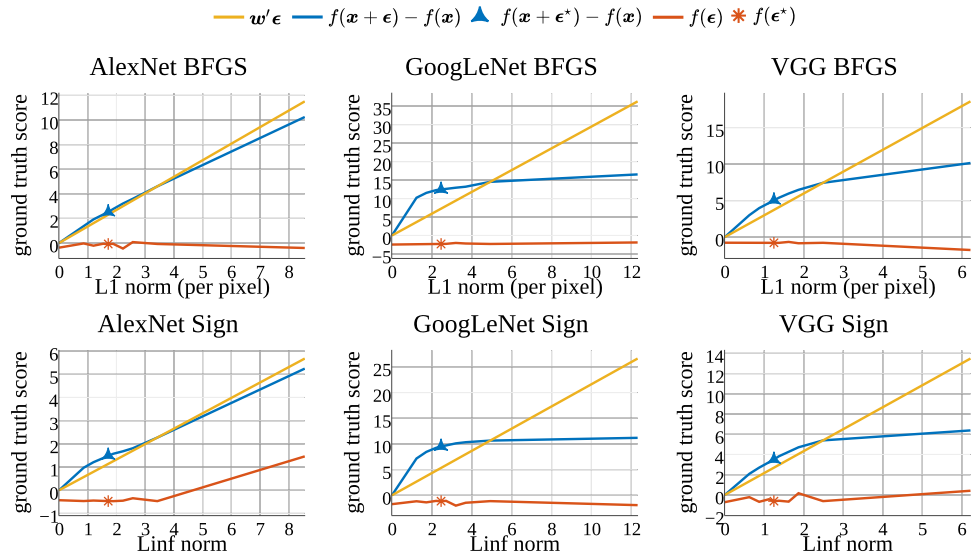


Figure 10: *Local Linearity of the CNNs*. Classification score of the ground truth object category for the image in Fig. 19, when varying the L_1 norm per pixel of *BFGS* and L_∞ for *Sign*.

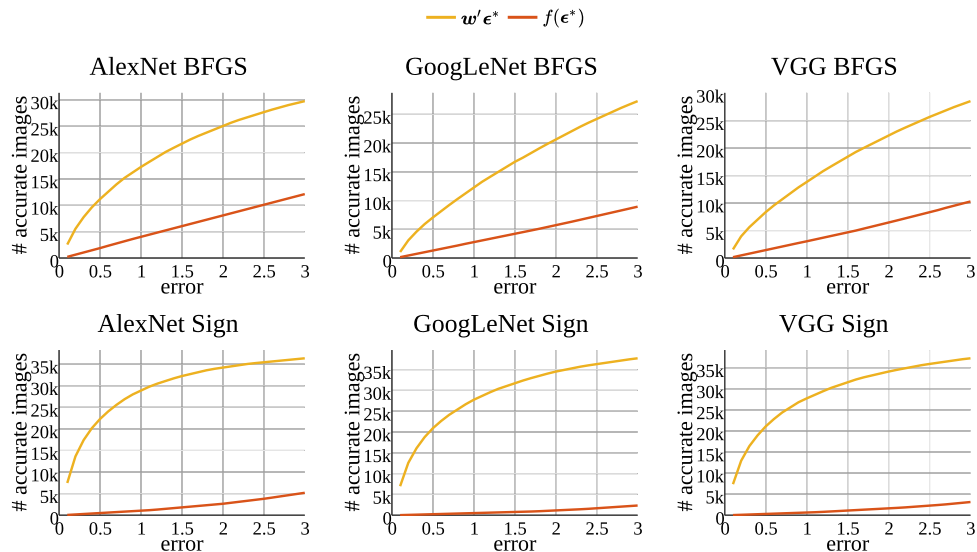


Figure 11: *Error of the Linearity Hypothesis*. Cumulative histogram of the number of images with an L_1 error smaller than the value in the horizontal axis. Only the images that are correctly classified by the CNN are included.

— Object Crop MP — 1 Shift MP-Object — Saliency Crop MP

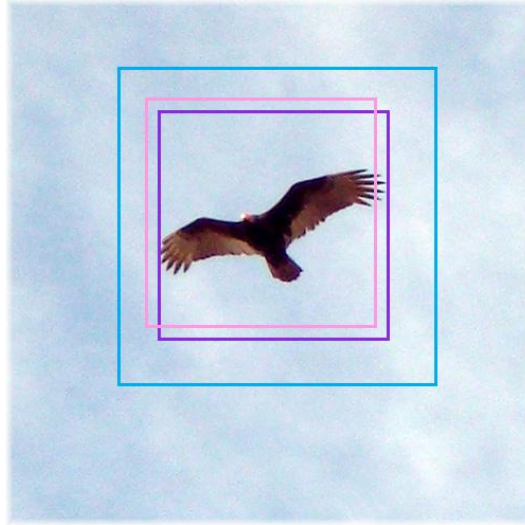


Figure 12: *Example of Different Foveations. Object Crop MP is the purple bounding box and 1 Shift MP-Object is the pink bounding box. The blue bounding box corresponds to a crop from Saliency Crop MP. Note that the center crop of 10 Shift MP-Object is exactly the same as Object Crop MP.*

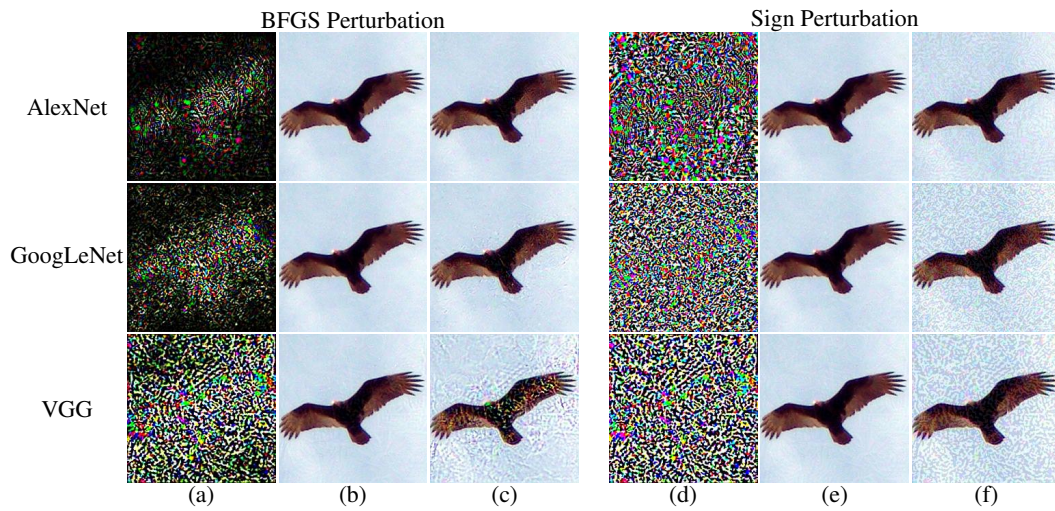


Figure 13: *Example of different CNNs' MP-Object, i.e., MP generated on the crop object subimage. Each row corresponds to a different CNN. (a) BFGS, (b) its corresponding adversarial example, and (c) the adversarial example with the perturbation multiplied by 10; (d) Sign, (e) and (f) the same as (b) and (c), respectively, for Sign.*

Table 2: *Evaluation of the Foveation Mechanisms.* Quantitative results of the top-5 accuracy with minimum perturbation.

BFGS Minimum Perturbation								
	Before Foveation		After Foveation		Before Foveation		After Foveation	
	w/o MP	MP	Object Crop MP	Saliency Crop MP	w/o MP-Object	MP-Object	1 Shift MP-Object	Embedded MP-Object
	ALX	0.7841	0.0048	0.7804	0.7076	0.8192	0.0175	0.7683
GNT	0.8736	0.0104	0.8313	0.7875	0.8939	0.0284	0.8329	0.8418
VGG	0.8536	0.0055	0.8258	0.8162	0.9122	0.0308	0.8151	0.7806

Sign Minimum Perturbation								
	Before Foveation		After Foveation		Before Foveation		After Foveation	
	w/o MP	MP	Object Crop MP	Saliency Crop MP	w/o MP-Object	MP-Object	1 Shift MP-Object	Embedded MP-Object
	ALX	0.7841	0	0.7210	0.6656	0.8192	0.0022	0.7066
GNT	0.8736	0.0006	0.7285	0.6995	0.8939	0.0051	0.7330	0.7764
VGG	0.8536	0.0007	0.6856	0.7304	0.9122	0.0065	0.6397	0.6766

Table 3: *Evaluation of the Foveation Mechanisms.* Quantitative results of the top-5 accuracy in Fig. 14.

No Perturbation ($L_1 = 0$)										
	Before Foveation		After Foveation			Before Foveation		After Foveation		
	MP	MP	Object Crop MP	10 Crop MP	3 Crop MP	Saliency Crop MP	MP-Object	10 Shift MP-Object	1 Shift MP-Object	Embedded MP-Object
ALX	0.7841		0.8192	0.8111	0.8026	0.8073	0.8192	0.8341	0.8123	0.7841
GNT	0.8736		0.8939	0.8951	0.8903	0.8922	0.8939	0.9030	0.8910	0.8736
VGG	0.8536		0.9122	0.8912	0.8872	0.8957	0.9122	0.9212	0.9132	0.8536

BFGS Perturbation ($L_1 = 5.3$)										
	Before Foveation		After Foveation			Before Foveation		After Foveation		
	MP	MP	Object Crop MP	10 Crop MP	3 Crop MP	Saliency Crop MP	MP-Object	10 Shift MP-Object	1 Shift MP-Object	Embedded MP-Object
ALX	0.1166		0.5401	0.4782	0.4016	0.4985	0.1043	0.3632	0.3304	0.5100
GNT	0.1622		0.5972	0.5623	0.4913	0.5866	0.1934	0.4662	0.4394	0.6204
VGG	0.1477		0.5180	0.3966	0.3391	0.5234	0.1679	0.3355	0.3190	0.4838

Sign Perturbation ($L_\infty = 5.3$)										
	Before Foveation		After Foveation			Before Foveation		After Foveation		
	MP	MP	Object Crop MP	10 Crop MP	3 Crop MP	Saliency Crop MP	MP-Object	10 Shift MP-Object	1 Shift MP-Object	Embedded MP-Object
ALX	0.0855		0.5225	0.4515	0.3619	0.4986	0.1501	0.5506	0.4989	0.5970
GNT	0.1793		0.6133	0.5666	0.4856	0.6161	0.2292	0.6235	0.5763	0.6938
VGG	0.1238		0.4867	0.3319	0.2812	0.4869	0.1995	0.4259	0.3995	0.5520

Table 4: *Evaluation of the Foveation Mechanisms.* Increase factor of the norm of the perturbation to produce misclassification after the foveation mechanisms. Only the images that are correctly classified before and after the foveation are included.

ratio	AlexNet		GoogLeNet		VGG	
	Sign	BFGS	Sign	BFGS	Sign	BFGS
Object Crop MP / MP	6.2484	14.4476	6.3432	25.8569	5.0216	17.1506
10 Shift MP-Object / MP-Object	4.9085	9.1912	5.6342	19.4249	2.9630	8.5588
1 Shift MP-Object / MP-Object	4.5336	8.4306	5.1256	18.1737	2.9676	8.8038

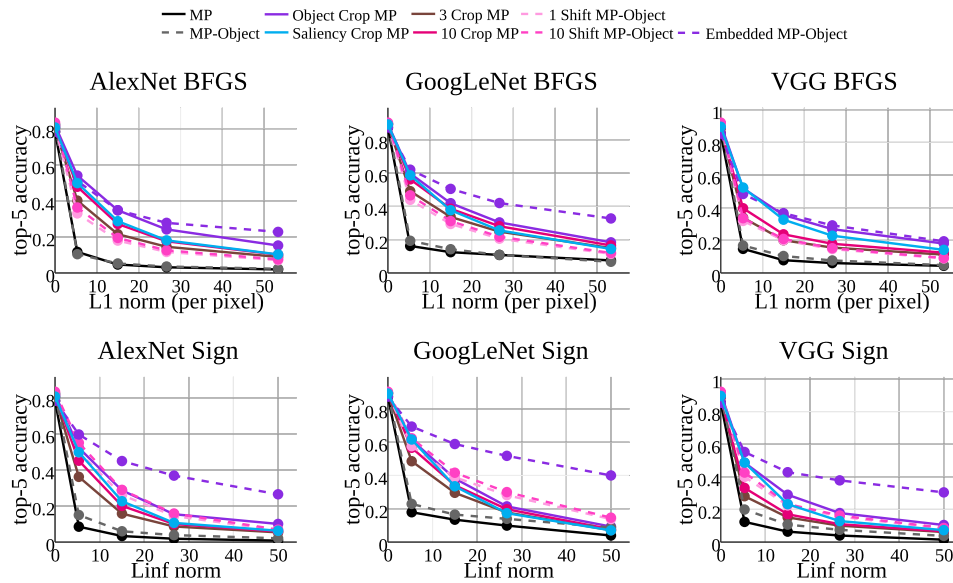


Figure 14: *Accuracy of the Foveations*. Accuracy for the three CNNs we evaluate, when varying the L_1 norm of *BFGS* and varying the L_∞ norm of *Sign*.

B GENERATION OF THE ADVERSARIAL PERTURBATION

Before introducing the generation of the perturbation, we introduce a more specific mathematical notation, that later will serve to clarify the details about the perturbation generation. Recall that $\mathbf{x} \in \mathbb{R}^N$ is an image of size N pixels. This image contains an object whose object category is of class label $\ell \in \{1 \dots k\}$, where k is the number of object categories (e.g., $k = 1000$ in ImageNet). $f : \mathbb{R}^N \rightarrow \mathbb{R}^k$ is the mapping from the input image to the classification scores returned by the CNN. Note that $f(\mathbf{x}) \in \mathbb{R}^k$ is a vector that contains the classification scores for all object categories. We use $e(f(\mathbf{x}), \ell)$ to denote a function to evaluate the error of the classification scores, where ℓ is the label of the ground truth object category. In ImageNet, this error function is based on the top-5 scores (Russakovsky et al., 2015). Thus, $e(f(\mathbf{x}), \ell)$ returns 0 when ℓ corresponds to one of the five highest scores in $f(\mathbf{x})$, otherwise $e(f(\mathbf{x}), \ell)$ returns 1.

We use $\epsilon \in \mathbb{R}^N$ to denote the perturbation image that produce a misclassification when added to the input image, i.e., $e(f(\mathbf{x} + \epsilon), \ell) = 1$. The image $\mathbf{x} + \epsilon$ is the *adversarial example*. The set of all perturbation images that produce a misclassification of an image can be grouped together. We use $\mathcal{E}_{\mathbf{x}} \subset \mathbb{R}^N$ to denote such set, $\mathcal{E}_{\mathbf{x}} = \{\epsilon \in \mathbb{R}^N \mid e(f(\mathbf{x} + \epsilon), \ell) = 1\}$. Then, we define ϵ^* as the perturbation with minimal norm to produce misclassification of the image, i.e., $\epsilon^* = \arg \min_{\epsilon \in \mathcal{E}_{\mathbf{x}}} \|\epsilon\|$. Observe that the optimal perturbation depends on the norm we choose to minimize. We will analyze the L_1 and L_∞ norms, since the perturbations we analyze optimize one of these two norms.

BFGS perturbation. As in (Szegedy et al., 2014), we approximate the perturbation ϵ^* by using a box-constrained L-BFGS¹. It consists on minimizing the L_1 norm of the perturbation, $\|\epsilon\|_1$, that produces a misclassification of the image using the top-5 accuracy criteria, i.e., $e(f(\mathbf{x} + \epsilon), \ell) = 1$. To do so, we minimize the L_1 norm plus a loss function, which we denote as $\text{loss}(\epsilon, \{\mathbf{x}, \ell\})$. Let η be a constant that weights the L_1 norm with respect to the loss function.

Since the minimization should produce a misclassification, the loss function is based on the accuracy. Thus, the loss could be equal to $(1 - e(f(\mathbf{x} + \epsilon), \ell))$. Since directly minimizing the top-5 accuracy is difficult, because the derivative of $e(f(\mathbf{x} + \epsilon), \ell)$ is 0 for any ϵ except in the boundary of producing a misclassification, we use a hinge loss. Thus, $\text{loss}(\epsilon, \{\mathbf{x}, \ell\})$ is equal to 0 when the image is misclassified (which corresponds to the final objective), otherwise the loss function takes the value of the classification score of the class that we aim to misclassify, which can be expressed as $I(\ell)^T f(\mathbf{x} + \epsilon)$ where $I(\ell) \in \mathbb{R}^k$ is an indicator vector that is 1 in the entry corresponding to the class ℓ and 0 otherwise.

The minimization of $\eta\|\epsilon\|_1 + \text{loss}(\epsilon, \{\mathbf{x}, \ell\})$ is done with L-BFGS. This uses the gradient of the loss with respect to ϵ , which can be computed using the back-propagation routines used during training of the CNN. In order to further minimize the norm returned by L-BFGS, we do a line search of the norm given the perturbation by L-BFGS, i.e., $\tilde{\epsilon} = \alpha\epsilon/\|\epsilon\|$, where α is a scalar factor.

In the experiments, we set $\eta = 10^{-6}$ because using this constant L-BFGS could find a perturbation that produces a misclassification in the majority of the images, except for approximately the 0.5% of the images. In all the CNNs tested, these images had a classification score higher than 0.9. To obtain a perturbation that produces a misclassification in these images, we apply the line-search method with the perturbation returned after stopping L-BFGS after one iteration.

Sign perturbation. It was introduced in Goodfellow et al. (2015). The perturbation of the *Sign perturbation* is generated using $\text{sign}(\nabla \text{loss}(\epsilon, \{\mathbf{x}, \ell\}))$, which can be computed using back-propagation. Then, we use the line-search method to minimize the norm of the perturbation to misclassify all images.

¹<http://www.cs.ubc.ca/~schmidtm/Software/minFunc.html>

C PERCEPTIBILITY OF THE ADVERSARIAL PERTURBATION

In the following pages we show examples of the perceptibility of the perturbations of the adversarial examples.

In Fig. 15, 17 and 19, we show examples of the same adversarial perturbation varying the L_1 norm per pixel, and the L_∞ norm, in Fig. 16, 18, 20. We can see that from a certain factor, the perturbation becomes clearly perceptible, and it occludes parts of the target object. This helps us approximately determine at what point the adversarial perturbations become perceptible. For *BFGS*, we can say that when the L_1 norm per pixel is higher than 15 the perturbation becomes visible, and for L_∞ the threshold is 100. This difference is because the density of *BFGS* is not the same through all image, it is mainly in the same position of the object, as shown in Fig. 1. For *Sign*, the threshold for both norms to make the perturbation visible is about 15, and it is the same for both norms because this perturbation is spread evenly through all the image. We use these values for the analysis.

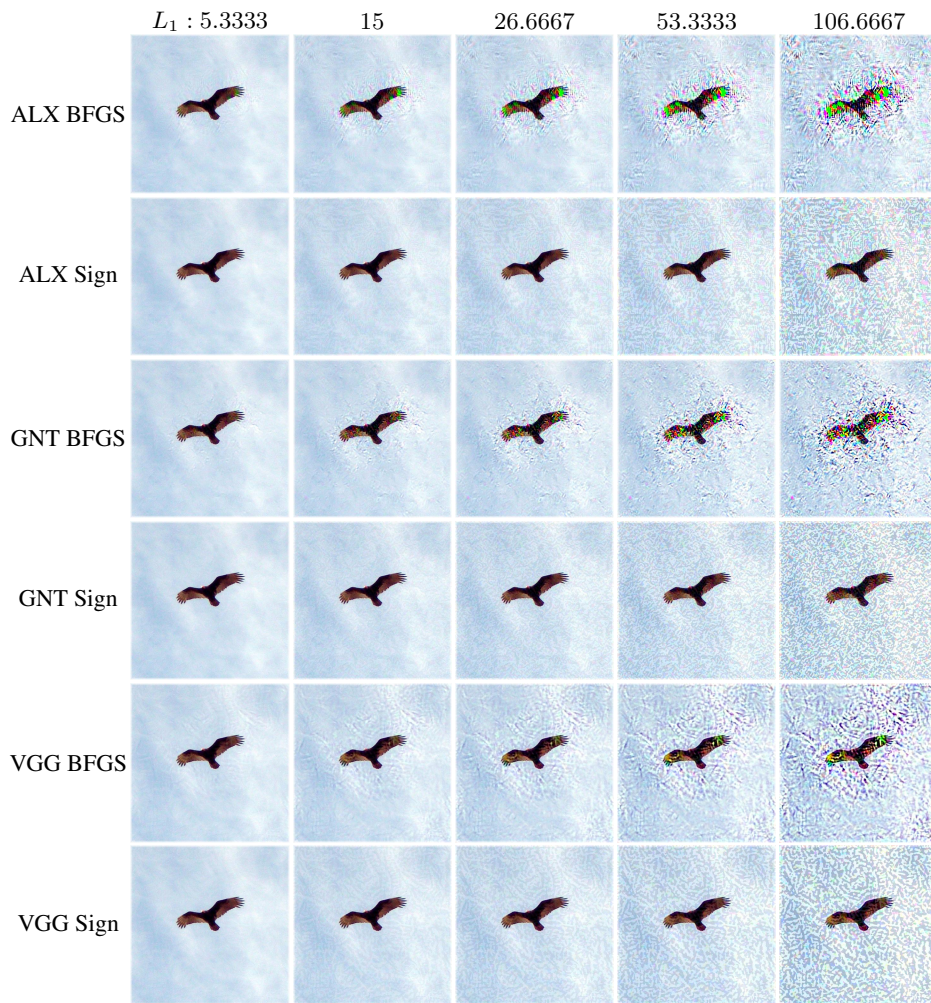


Figure 15: *Qualitative example of the perturbations changing the average per pixel of the L_1 norm. We denote the perturbation as $X Y$, where X is the network that generated the perturbation - ALX (AlexNet), GNT (GoogLeNet), VGG- and Y indicates the *BFGS* or *Sign*.*

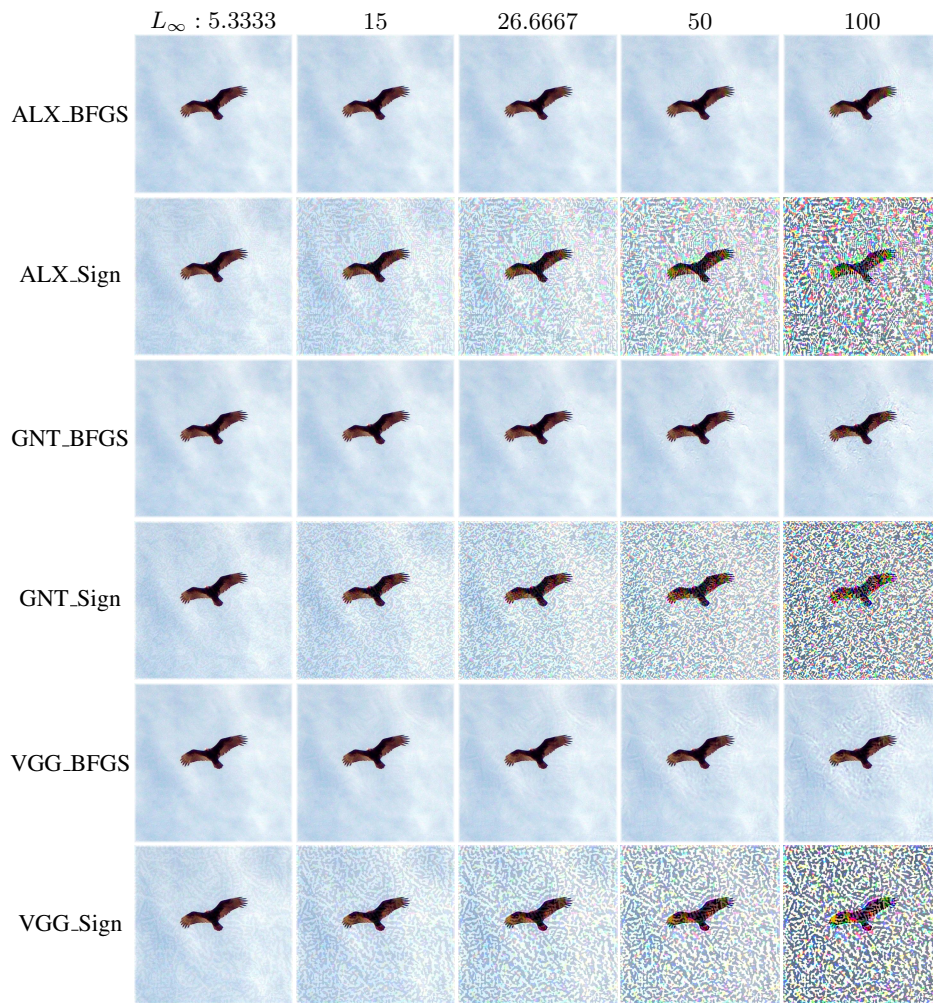


Figure 16: *Qualitative example of the perturbations changing the average per pixel of the L_∞ norm. We denote the perturbation as $X Y$, where X is the network that generated the perturbation - ALX (AlexNet), GNT (GoogLeNet), VGG- and Y indicates the *BFGS* or *Sign*.*

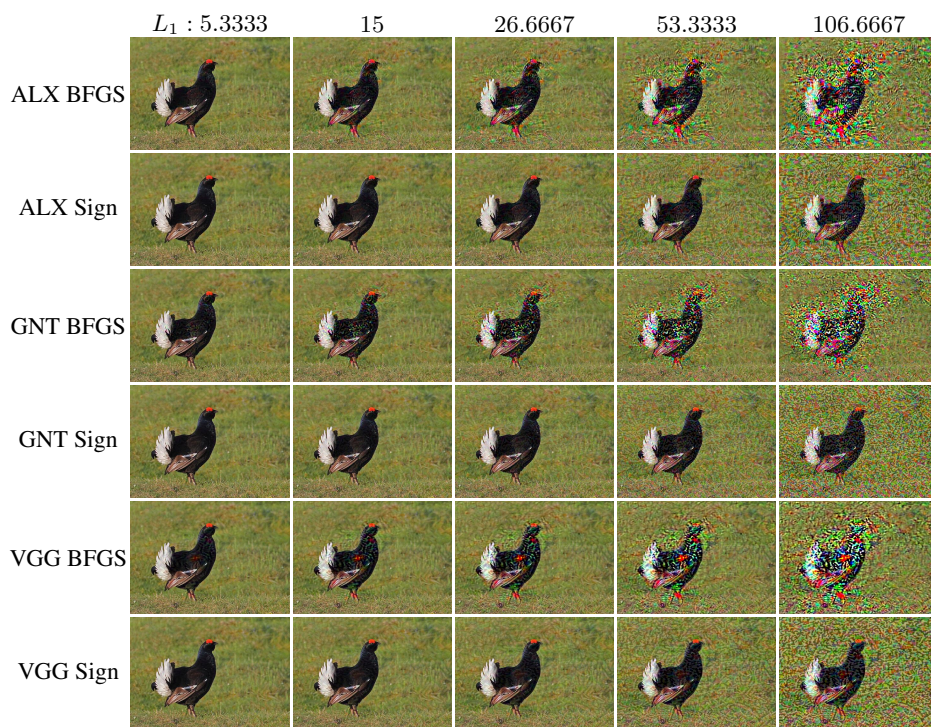


Figure 17: *Qualitative example of the perturbations changing the average per pixel of the L_1 norm. We denote the perturbation as $X Y$, where X is the network that generated the perturbation - ALX (AlexNet), GNT (GoogLeNet), VGG- and Y indicates the *BFGS* or *Sign*.*

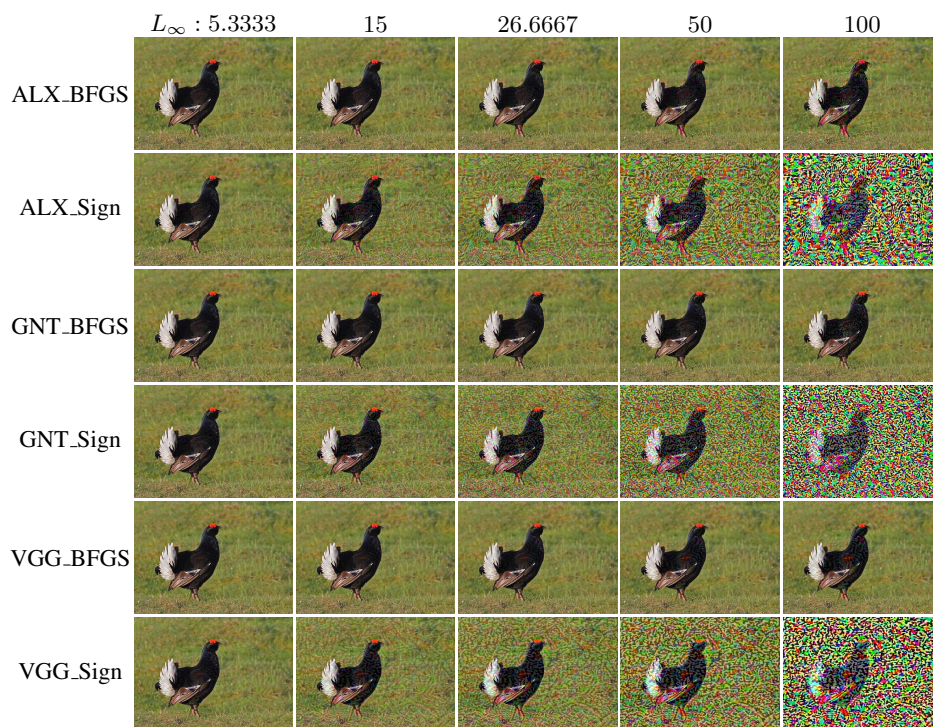


Figure 18: *Qualitative example of the perturbations changing the average per pixel of the L_∞ norm. We denote the perturbation as $X Y$, where X is the network that generated the perturbation - ALX (AlexNet), GNT (GoogLeNet), VGG- and Y indicates the *BFGS* or *Sign*.*

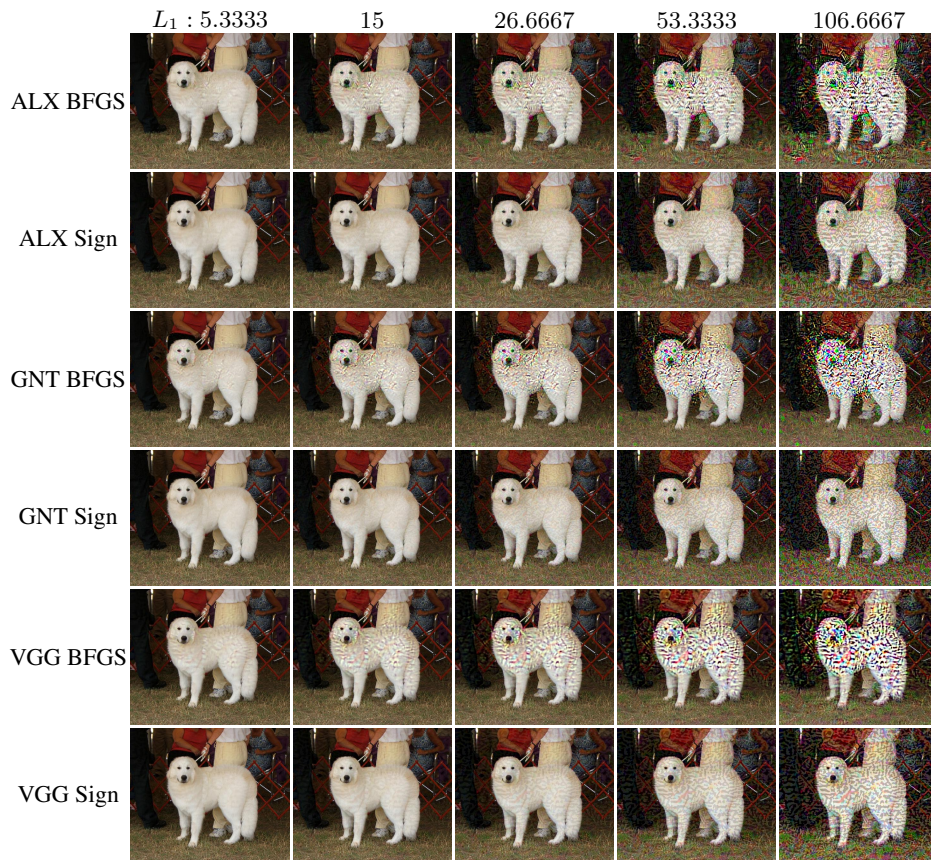


Figure 19: *Qualitative example of the perturbations changing the average per pixel of the L_1 norm. We denote the perturbation as $X Y$, where X is the network that generated the perturbation - ALX (AlexNet), GNT (GoogLeNet), VGG- and Y indicates the *BFGS* or *Sign*.*

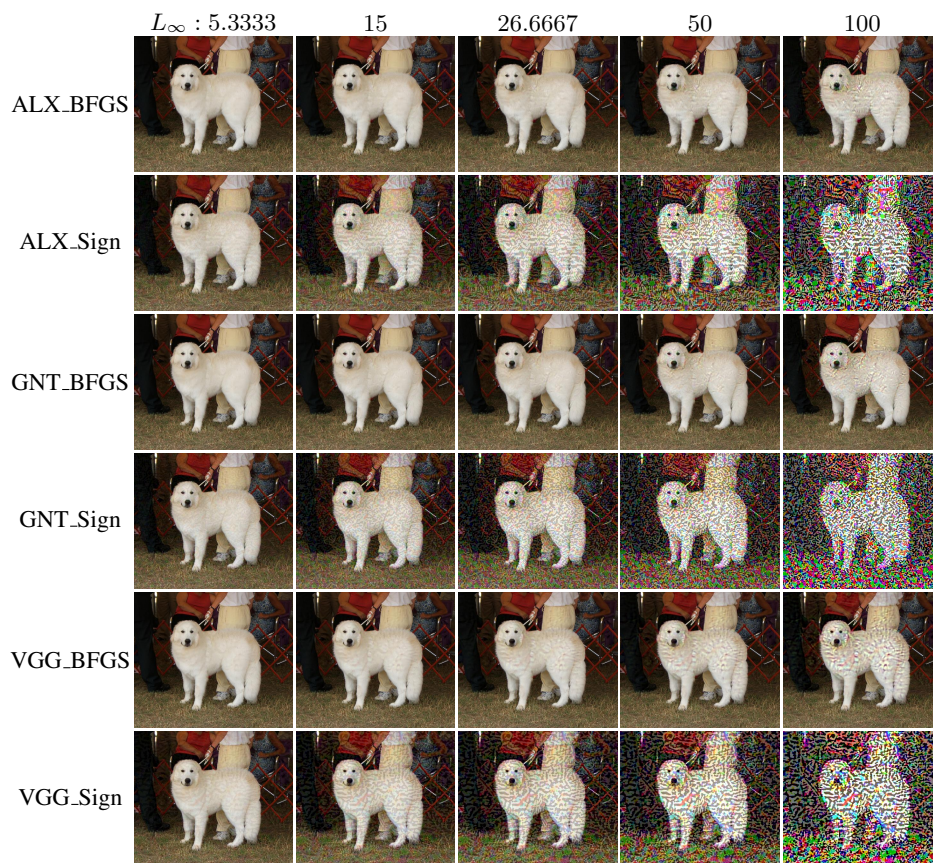


Figure 20: *Qualitative example of the perturbations changing the average per pixel of the L_∞ norm. We denote the perturbation as $X Y$, where X is the network that generated the perturbation - ALX (AlexNet), GNT (GoogLeNet), VGG- and Y indicates the *BFGS* or *Sign*.*