Features-Based Fast Gradient Sign Method

Hamzeh Alzwairy
Michigan State University
East Lansing, MI
alzwerih@msu.edu

Abstract

Fast Gradient Sign Method [2] is used to find perturbations of the input image that causes the victim model to misclassify it, the perturbation is found using the gradient direction of the loss function with respect to the input pixels. An adversarial example is misclassified when the features move from the region of the correct class to a region of a different class in the feature space, it essentially means we force the example to cross a classifier's boundary. In this project, we will demonstrate a new version of this attack that will also cause the image to be misclassified by changing the region of the input features in the feature space. However, in the Feature-based FGSM we will not find the perturbation using the training loss and output neurons, but rather we will be using a new loss applied to the intermediate features, neurons, of the model.

1 Introduction and problem formulation

White box attacks are the type of adversarial attacks where the adversary has access to the parameters of the victim model, which allows the usage of gradient methods to produce unnoticeable perturbations to inputs in order to misclassify them. Fast Gradient Sign Method is an effective and computationally efficient attack, it simply perturbs the input pixels of the image pixels by a step size, ϵ, in the direction of the gradient of the loss in order to increase the loss and hence misclassify the image. The higher the value of ϵ, the more detectable are the perturbations and the lower the accuracy of the model, if we want to achieve an overall low accuracy for the model, we need to use high values of ϵ only cause a small portion of the data.

This means that in order to reduce the overall accuracy of the model we need to sacrifice confidentiality of the attack. The proposed method, Feature-Based FGSM, tries to find a gradient direction which allows the usage of a small perturbation step ϵ and still reduces the overall accuracy of the victim model. The idea is to make the features of the input image in the last layers of the network change such that they look more similar to other classes more than the original class, this means that the classifier will misclassify this image and the accuracy will decrease, more details in the next section.

2 Methodology

When we are using FGSM attack, we find a perturbation that will increase the training loss using the gradient of the input with respect to the loss function. This can be seen as moving the input towards the classifier's boundary in the feature space as in Figure 1. The assumption is that direction of this gradient is guaranteed to increase the loss, however it does not guarantee that it is the optimal direction towards a boundary because the loss would still increase even if we only change a subset of the features. For example, if a point has 2 features x and y and we change only one feature we

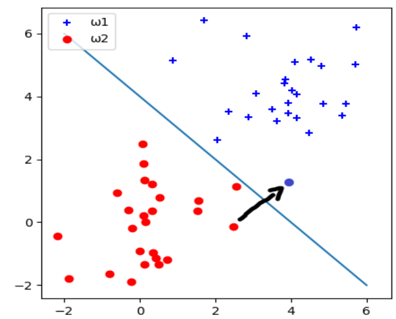

Figure 1: Increasing loss intuition

can only move in a limited direction, whether the x axis or y axis, in the feature space, we can move the point away from its original class but we will need bigger steps as we are not using an optimal direction. If we are able to change all the features then we can find a direction that uses fewer steps to reach the target (cross the boundary) and hence this means a smaller ϵ. In order to find the optimal direction in the feature space, we need to change the pixels such that target is to move the features of the image away from original class directly and not just increase a classification loss which may cause us to use sub-optimal directions. We find this gradient direction by extracting the features of the source image, feeding it to a Euclidean distance loss function between these features and a mean feature vector which will pull the features away from the original class, and finding the gradient of the input pixels with respect to the loss. There are two variations of this attack:

1. **Targeted:** Where our target is to minimize the euclidean distance between the features of the source image and a mean feature vector of the target class so that input image features look like the target class features. The mean feature vector is obtained by passing a batch of target class images through the network, extracting and flattening their features from the last convolutional layer, and finding the mean of these features. This loss can be represented as:

   $$J = \left\| f(x) - f(B)^{'} \right\|_2^2$$

   Where x is the input image, f is the network feature extractor, B a batch of images that belong to the target class, and $f(B)^{'}$ is the mean vector of the target class batch which is mean to server as a typical "style" vector that the input image. Changing more features towards $f(B)^{'}$ is what will decrease this loss, while in FGSM even changing fewer number of features will decrease loss but we wouldn't guarantee changing most of the features when needed.

   And here, the perturbation δ is:

   $$\delta = \epsilon sign(\Delta_x J(\theta, x, B))$$

   Where ϵ is the step size, θ is the parameters of the model, x is the input image and B is the target class batch.

   We change x to x^* through:

   $$x^* = x - \delta$$

2. **Untargeted:** Similar to the targeted attack in terms of the idea and the loss used, but different in terms of the value of $f(B)^{'}$ and optimization goal. Here $f(B)^{'}$ denotes the mean features vector or style vector of the source image, which is the input image, and the optimization goal is to maximize the Euclidean distance which yields:

   $$x^* = x + \delta$$

   This has the effect of moving the image away from its original "style" or other source images in the features space.

3 Experiments

3.1 Implementation

This attack was tested on a small network, LeNet, and a large network, GoogLeNet [3] which was used in the original FGSM paper. LeNet was trained on the MNIST dataset while GoogLeNet was trained on ImageNet [1]. For both models, we loop over test images obtained from either the validation or test data, extract features of the image, extract the features of the target batch (the source class in the untargeted case, and the target class in the targeted case), feed those two feature vectors to the loss function and then find derivative of input pixels with respect to the loss and update them accordingly. Below you can find the implementation algorithm:

Algorithm 1 Features-Based FGSM

$LS \leftarrow$ Loss type
$B \leftarrow$ None
$T \leftarrow$ Target class
$M \leftarrow$ Target model
$n \leftarrow$ **function** $\text{LENGTH}(data)$
$correct \leftarrow 0$
for $x, label$ in $data$ **do**
  if $LS ==$ "Targeted" **then**
    if $label == T$ **then**
      $n = n - 1$
      Continue
    else
      $B \leftarrow$ **function** $\text{GETBATCH}(T)$
    end if
  else
    $B \leftarrow$ **function** $\text{GETBATCH}(label)$
  end if

  $f(B)^{'} \leftarrow$ **function** $\text{AVG}(f(B))$

  $loss \leftarrow J(f(x), f(B)^{'})$

  $\delta \leftarrow \epsilon \, sign(\Delta_x loss)$

  if $LS ==$ "Targeted" **then**
    $x^* = x - delta$
  else
    $x^* = x + delta$
  end if

  $l^* = M(x^*)$
  if $l^* == label$ **then**
    $correct \leftarrow correct + 1$
  end if
end for
$Accuracy \leftarrow \frac{correct}{n}$

Where $label$ is the true label of the source image, f is the feature extractor of the network, J is the loss function, x^* is the adversarial example, l^* is the model's output label for the adversarial example, and n is the total number of attempted adversarial images. We removed all images belonging to the target class from the test images in the targeted case. We also removed images that were initially misclassified by the model without any perturbation so that we only try to perturb images that the model classifies correctly.

3.2 Results

We report the results for each model separately, we compare the results of our attack to the performance of the FGSM attack on these models below.

3.2.1 GoogLeNet results

We tried 7 different epsilons for each variation of the attack, the test data used consists of 5000 images of 1000 classes, 5 examples were sampled from every class. The attack reduce accuracy of the model from 100% to approximately 49.5% using $\epsilon = 0.05$ for the untargeted attack. The targeted attack reduce accuracy from 100% to approximately 48.6% using the same value of ϵ. Figures 2 and 3 show results for different epsilon values, while Figures 4 and 5 show samples of perturbed images for each epsilon value.

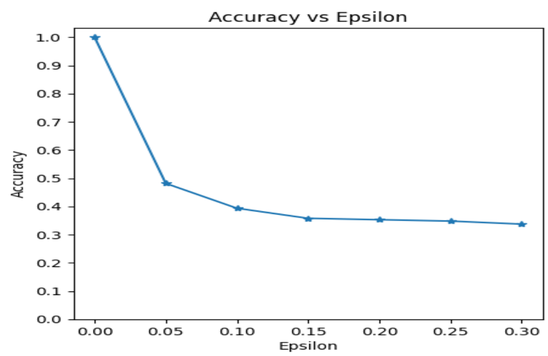

Figure 2: Targeted attack on GoogLeNet

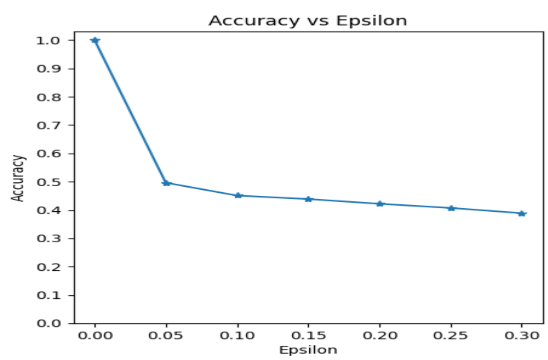

Figure 3: Untargeted attack on GoogLeNet

3.2.2 FGSM on GoogLeNet results

The FGSM attack was applied to the same test data used for Features-Based FGSM using the same values of ϵ, it reduces the accuracy of the model from 100% to 0.733% using $\epsilon = 0.05$, results are reported in Figure 6.

3.2.3 LeNet results

Features-Based FGSM attack was tested on a LeNet model trained on the MNIST dataset, attack was tested on 10000 images for each variation of the attack. The untargeted attack reduce accuracy only slightly from 100% to 98.9% for $\epsilon = 0.05$ but the accuracy kept decreasing until it reached 39%

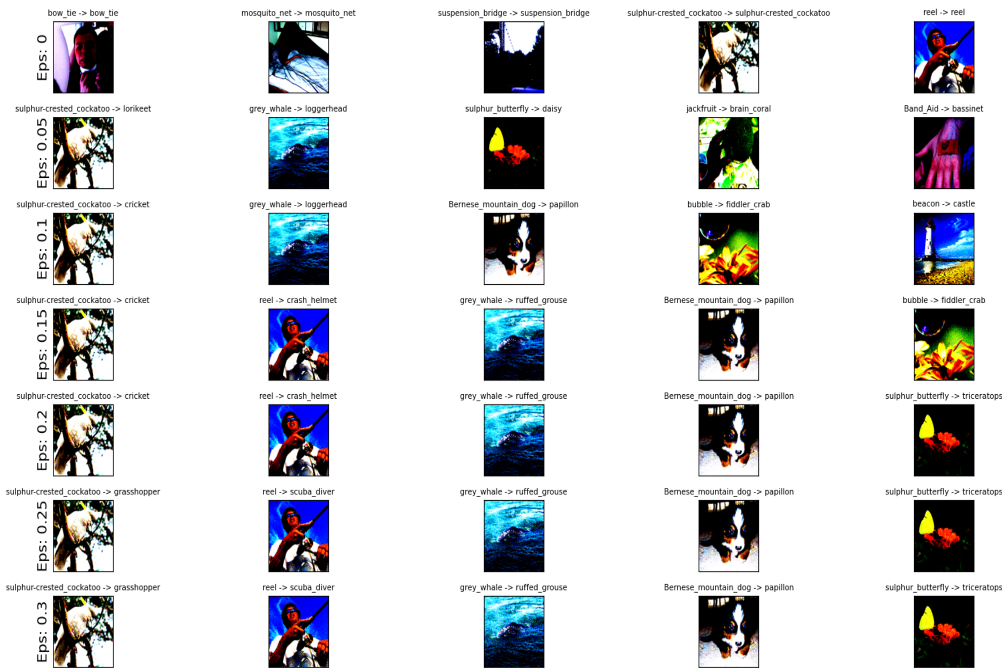

Figure 4: ImageNet adversarial samples from targeted attack

as we increased ϵ, check Figure 7 and 8. The targeted version reduce the accuracy from 100% to 88.46% for $\epsilon = 0.05$. Results are in Figure 9 and 10. However, FGSM outperforms both the attacks as we can see in Figure 11.

4 Thoughts and conclusion

The proposed attack works and reduces the overall accuracy of the model. However, it does not overperform the FGSM attack. Smaller values of ϵ in the FGSM attack yielded lower accuracy than the Features-Based FGSM attack. Results were different for targeted and untargeted versions of the attack; the untargeted attack greatly outperformed the targeted attack on the MNIST dataset while unperformed slightly on the ImageNet data, the reason is, to the best of my knowledge, the difference in the number of classes that the model is predicting. The MNIST dataset only has few classes and hence the distributions of each class could be far away from other classes in the feature space, so when we choose a specific class to move the adversarial example in its direction, we will need a high epsilon as they might be very separated in the feature space. For the untargeted attack on the MNIST, a larger epsilon value was able to reduce the accuracy down to 39%, because we do not specify when class the adversarial example needs to follow, the gradient will point towards the closest boundary or set of classes and so a smaller ϵ is needed. So, the more classes the model works with, the more vulnerable it is because you get a higher chance of getting 2 classes close to each other in the feature space so you will need a smaller perturbation value to cause a misclassification.

For the ImageNet dataset, there is a large number of classes in the feature space and hence the boundary is more complex and easier to reach through perturbations, both targeted and untargeted attacks reduced accuracy of the model to less than 50% with a small value of 0.05 for ϵ.

To conclude, the proposed Features-Based FGSM attack works and finds a proper perturbation. However, it does not use a smaller perturbation amount ϵ as assumed before the experiments.

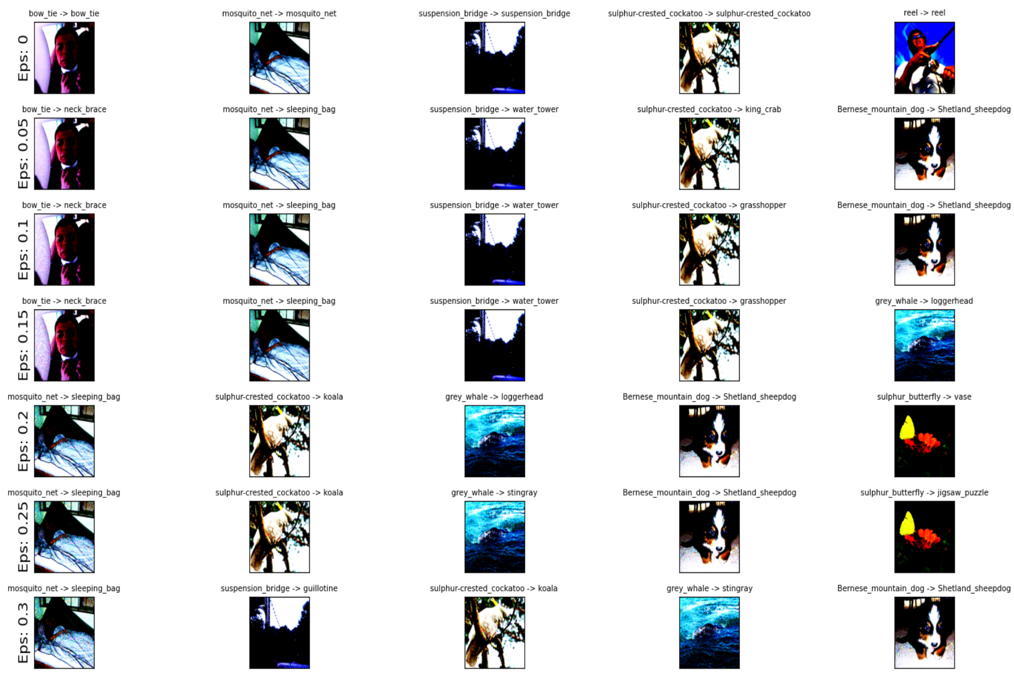

Figure 5: ImageNet adversarial samples from untargeted attack

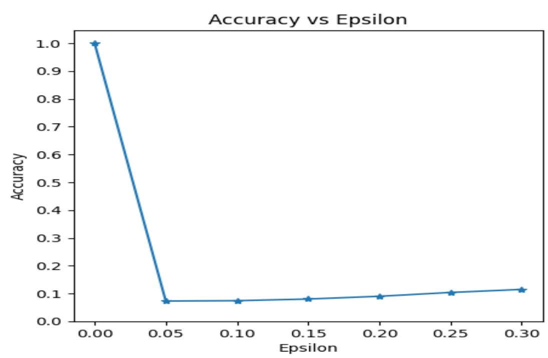

Figure 6: FGSM attack on GoogLeNet

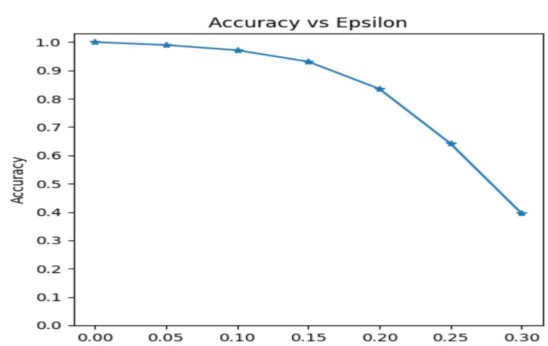

Figure 7: Untargeted attack on MNIST

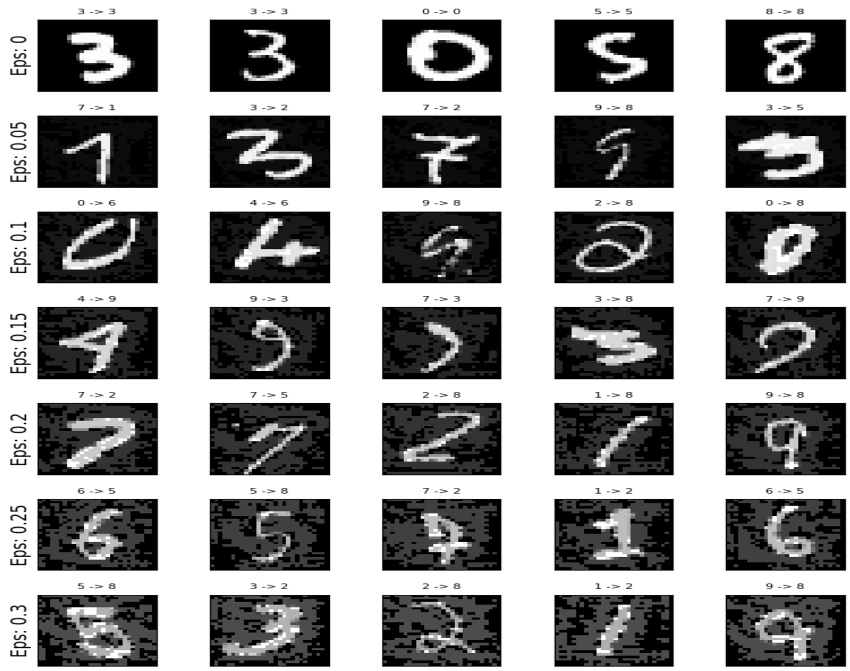

Figure 8: MNIST adversarial samples from untargeted attack

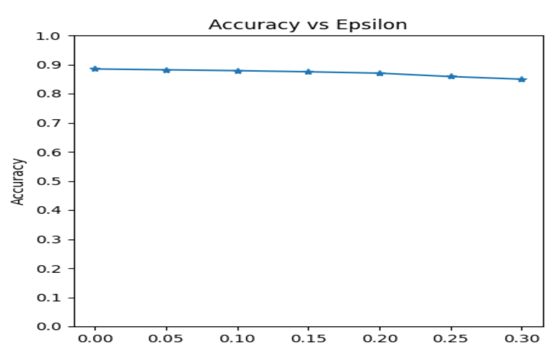

Figure 9: Targeted attack on MNIST

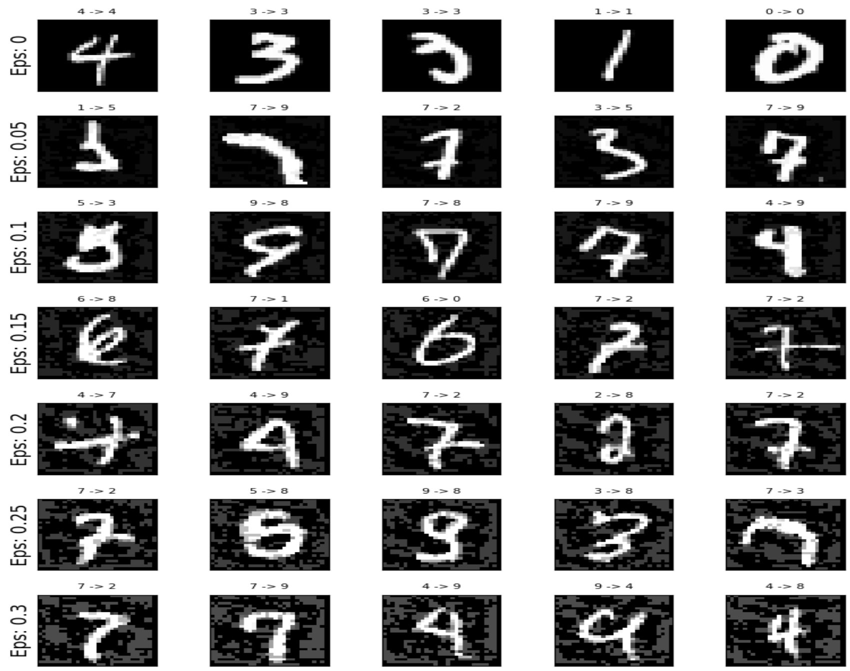

Figure 10: MNIST adversarial samples from targeted attack

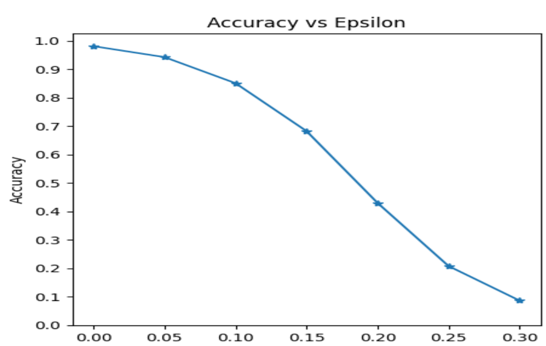

Figure 11: FGSM attack on MNIST

References

[1] Jia Deng et al. “ImageNet: A large-scale hierarchical image database”. In: *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*. IEEE Computer Society, 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848. URL: https://doi.org/10.1109/CVPR.2009.5206848.

[2] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. “Explaining and Harnessing Adversarial Examples”. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2015. URL: http://arxiv.org/abs/1412.6572.

[3] Christian Szegedy et al. “Going Deeper with Convolutions”. In: *CoRR* abs/1409.4842 (2014). arXiv: 1409.4842. URL: http://arxiv.org/abs/1409.4842.