Saliency detection from attacking and defensing deep learning model

1st Ting-Wei Wu
dept. Electrical and Computer Engineering
Georgia Institute of Technology
waynewu@gatech.edu

2nd Tzu-Han Wang
dept. Electrical and Computer Engineering
Georgia Institute of Technology
twang475@gatech.edu

***Abstract*—In this project, the objective focused on investigating the image saliency for a target-specific convolutional neural network on the classification task by evaluating its robustness with the presence of visually-subtle adversarial perturbations. The main goal is to reach a saliency mask generated from local perturbed noise attack to demonstrate how neural network classifies its given object. The experiments carry out different attack methods on a ResNet50 CNN model pretrained with a retinal image dataset which is security-sensitive and should be explored more on saliency detection for abnormality. Several mechanisms include random attack and mask attack with gradient saliency mask, which intend to disclose model vulnerability which may be imperative parts for classification that should not be perturbed. To justify our findings, a baseline ResNet50 CNN model was also pretrained on ImageNet dataset and repeated all mechanisms on typical normal images. Initially, random Gaussian noise will be added to verify if the final of output from the processing model is degraded. Subsequently, same attack mechanisms would be applied to different or learned segmented regions from original images to explore the most vulnerable segments, which might allude their saliency for the model's classifying criteria. The results give us explanatory visualization about certain object localization in the image throughout targeted perturbed noise attack and justifies through human evaluation.**

***Index Terms*—Saliency, , Noise generator, Smooth gradient**

I. Introduction

With an increase of interests in applying deep learning techniques in security-sensitive image datasets, it is important to prevent the model from intentional or unintentional perturbations. The current pattern recognition technology (Pattern Recognition), including speech and item recognition to modern network security tasks, has made great progress with the development of the deep learning technology. However, these techniques are easily confused by adversarial examples. The so-called adversarial examples refer to some samples deliberately confusing and misleading for these recognition tasks. For example, for a picture recognition task, the adversarial sample can perturb the picture at the pixel level, so that the human eye can't see the problem, but the machine will fail to recognize.

Because this field involves security, it is naturally an arms race between offense and defense just like the traditional cyber security field. Moreover, for the defensive application towards security-sensitive image or medical images, recent works such as [1]- [3] using defensive methods shows a great potential in saliency detection. As for saliency detection, work like [4]- [6] proposed a novel image saliency techniques that learns where an algorithm should focus by discovering the vulnerable parts of an image when it is perturbed. In this paper, the main goal is to show the effectiveness of saliency detection using various attack methods, and thus ,through these ways we actually demonstrated validation of salience detection with adversarial attack [7]- [9].

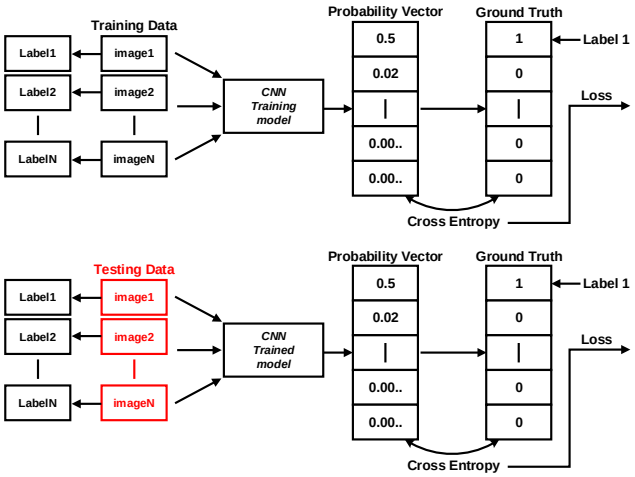


Fig. 1: The basic building blocks of the project image testing environments.

II. Methodology

In this project, several methods adversarial attack topology has been presented for later comparison. The propose is to demonstrate and compare the effectiveness of the saliency detection through different methods. The building blocks of the trained CNN model is first established for the project experiments validation. The fundamental classification testbench along with CNN model is shown Fig. 1. Then, adversarial attack for saliency detection with different methodology is introduced.

A. Un-targeting and Targeting Attack

The basic two adversarial attacks used in this project can be categorized into targeting and un-targeting attack. The un-targeting attack model is shown in Fig. 2. To begin with, a Gaussian noise vector $n \in \mathbb{R}^{100}$ is randomly generated.

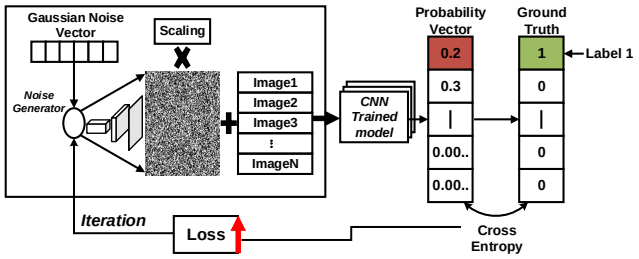


Fig. 2: The building blocks of un-targeting attack along with noise generator.

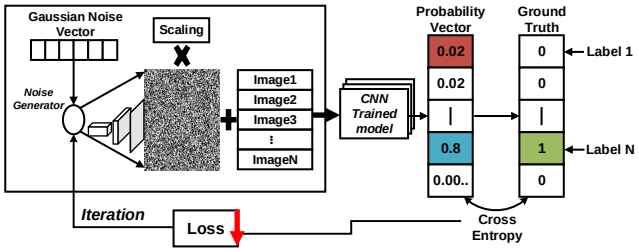


Fig. 3: The building blocks of targeting attack along with noise generator.

Through a random initialized generator $\phi(x)$ from generative adversarial networks (GAN), the magnitude of the output noise $\phi(n) \in \mathbb{R}^{224\times224}$ is scaled with an moderate varying factor α and eventually applied to the image datasets before entering the CNN trained model $S_c(x)$. After the image I passes the CNN model, the vector represented the probability of labels is then formed. The original label could be determined as $class(I) = argmax_{c\in C} S_c(I)$ where C is the possible label set. Finally, with the use of cross entropy loss between the probability vector and ground can be calculated for the further iteration at noise generator.

$$max_{\phi(n)} L(S_c(I), S_c(I + \phi(n) * \alpha)) \tag{1}$$

$$min_{\phi(n)} L(label_{targeted}, S_c(I + \phi(n) * \alpha)) \tag{2}$$

The goal of un-targeting attack is to increase the loss specified in Eq. 1 between the original probability vector from the original image I and that for perturbed image $I+\phi(n)*\alpha$. As it shows in Fig. 2, for instance, the confidence score of $label_1$ is reduced, thus the loss increases and corrupt the image classification.

Unlike un-targeting attack, the effectiveness of the targeting attack relies on the decrease of loss specified in Eq. 2 . For example, as shown the Fig. 3, to corrupt the $label_1$'s confidence score, one can switch $label_1$ ground truth to $label_N$ ground truth. Furthermore, the targeting attack forces a specific confidence score of label (e.g. $label_N$) to move approach the corresponding ground truth. Therefore the original $label_1$ can be corrupted.

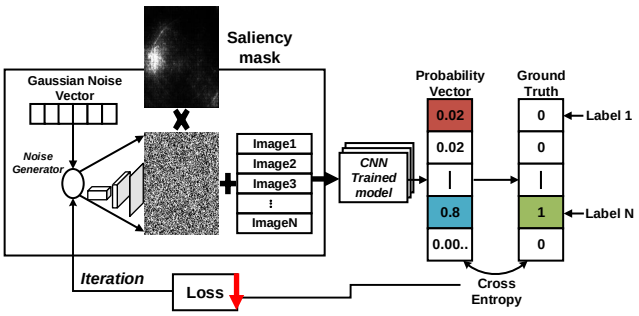


Fig. 4: The configuration of saliency mask attack.

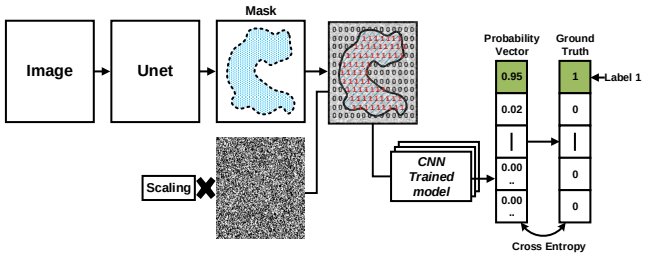


Fig. 5: Block diagram of adverasrial attack with the use of U-Net.

B. Saliency Attack

As previous section mentioned, defensive method is plausible for saliency detection. With the use of saliency map in an image, it discloses the vulnerable part of the image. Conversely, in this project, we applied the saliency masks which were used in SMOOTHGRAD technique [1] to the noise based on the previous structure. The overall methodology building blocks is shown in Fig. 4. With saliency mask, one can utilized it to verify whether the vulnerable part of a image exist at the predicted area.

$$M_c(I) = \frac{\partial S_c(I)}{\partial I} \tag{3}$$

$$\begin{aligned} max_{\phi(n)} L(S_c(I), S_c(I + \phi(n) * \alpha * M_c(I))) \\ min_{\phi(n)} L(label_{targeted}, S_c(I + \phi(n) * \alpha * M_c(I))) \end{aligned} \tag{4}$$

C. Mask Attack

The last attack methods for saliency detection introduced in the project is Mask attack and the diagram of Unet attack is shown in Fig. 11. Different from the saliency mask approach, the main modification in this method is to add a trainable mask from segmentation model or *Unet* $U(I)$. After the mask is generated by Unet, it will be combined with the noise from gain before entering CNN trained model. The overall loss will be further introduced three regularization terms as stated in Eq. 6. The first term $\lambda_1||\mathbf{1}-m||_1$ will encourages most of the mask to be turned off and the second term $\lambda_2 \sum ||\nabla m||_\beta$ calculates the overall gradient of the mask to perform max norm constraint

techniques. The last term makes the trained mask as close as the saliency map but not exactly the same to explore more possible revisited pixels for saliency. For experiments, we set the hyperparameter $\lambda_1 = 10^{-4}, lambda_2 = 10^{-2}, \beta = 3$.

$$m = U(I) \tag{5}$$

$$\begin{aligned} min_{\phi(n),m} L(label_{targeted}, S_c(I + \phi(n) * \alpha * m)) \\ +\lambda_1 ||\mathbf{1} - m||_1 \\ +\lambda_2 \sum ||\nabla m||_\beta \\ +\gamma \times MSE(M_c(I), m) \end{aligned} \tag{6}$$

III. Experiments & Result Analysis

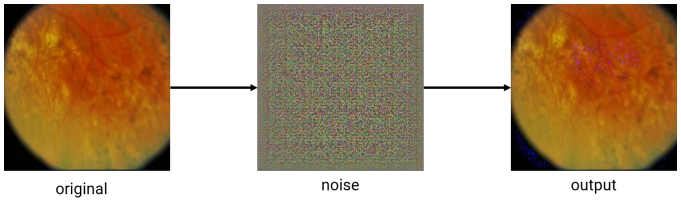
To assess our mask attack for image saliency detection, we performed a series of experiments to attack a neural network trained for image classification. Several attack mechanisms to generate noises are applied to original images and evaluate the number of iterations when the neural network start to fail in correct classification. With varied scaling factor α and bounded iteration times, we wish to discover the local pixels where applied noises will attack successfully in the shortest iteration, corresponding to a sensitivity map with respect to the vulnerability of the model.

A. Medical Image Classification

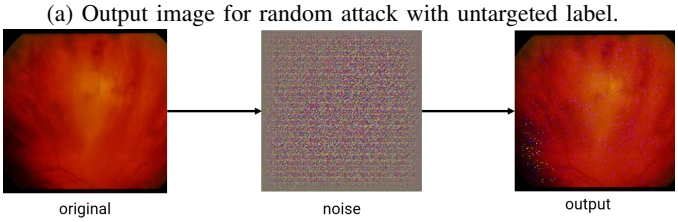
Before we delve into the attack mechanism, first we adopt an ResNet 50 model that was pretrained with ImageNet dataset and fine-tuned on a retinal image dataset from thousands of patients, contains two types of images, grey-scale Fluorescein Angiography (FA) and colorful Color Fundus Photography (CFP). The total amount of images is 15,709, including 1,811 FA and 13,898 CFP. We separate the whole dataset into $60\%/20\%/20\%$, i.e., $9425/3142/3142$, for training/validation/testing, respectively. By setting learning rate as 10^{-4} and mini batch size as 32, we could achieve training accuracy as 98.4% and testing accuracy at 80.2%. For robustness testing and saliency detection purpose, we will perform meaningful image perturbation on mostly training images where we wish to find information to "delete" and explain how the model classify its labels at a more robust scale.

B. Random Attack

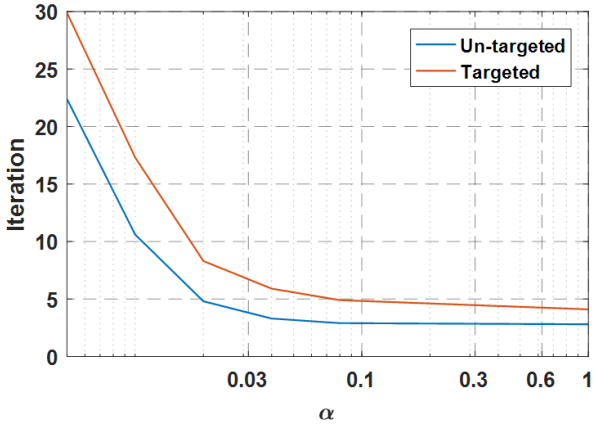
To set up baseline models for saliency mask attack, first we introduce flat gaussian noise from noise generator without any mask into the original image. If we perceive the initial random noise as $n \in R^{100}$, after fed in the noise generator $\phi(x)$ with random initial weights, we shall have the perturbed noise $\phi(n)$ with size same as the initial image. As stated in II-A, the experiments are followed by untargeted or targeted attack. By performing gradient descent on overall loss with respect to $\phi(x)$, we perform 100 rounds of attack and record the average number of iteration when the confirmation score reaches over 0.9 and stops. Untargeted attack will stop iteration when the



(a) Output image for random attack with untargeted label.



(b) Output image for random attack with targeted label.



(c) Comparison for varying α and iteration needed for success attack.

Fig. 6: The resulted images and the comparison result for different scaling factor α

perturbed image has begun to fall into wrong categories and targeted attack will stop iteration when the label is predicted as the given targeted class. The results could be visualized in Fig.6. Here, we set the scaling factor α initially as 0.02. After adding the random noise from noise generator, barely visible change from the original image could lead to huge loss and label change only within 5-6 iterations of gradient descent step, as illustrated by Table.I. Then by varying the scaling factor α which represents the attack magnitude on the original image, we could see slight decrease of α starting from 0.02 will significantly increase the necessary iteration step, which implicitly alludes the model robustness limit of preventing noises. On the other hand, when α increases, overall iteration steps remain under 5 where we shall not consider to add into the original image.

C. Mask Attack for saliency

After introducing flat gaussian noise from noise generator and evaluating model robustness, we would like to discover

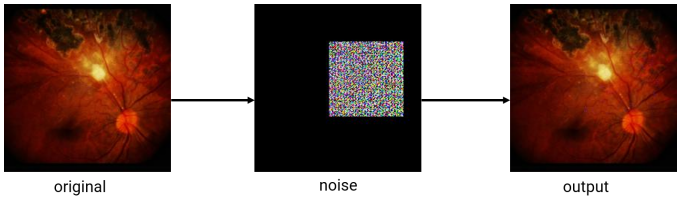


Fig. 7: Output image for local attack with untargeted label.

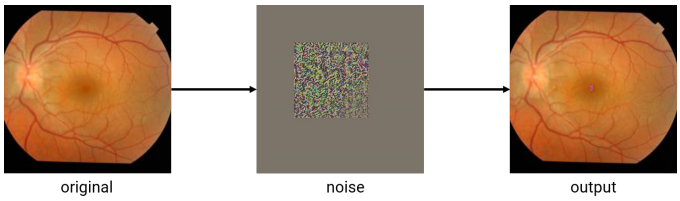


Fig. 8: Output image for local attack with targeted label.

the "local vulnerability" of our trained model, in terms of characterizing a sensitivity map to attack and localize the salient part of the given image. There are mainly three attack mechanism we would like to visit: local attack, saliency attack and mask attack.

1) Local Attack: The most explicit and fundamental way to detect local saliency is applying local gaussian noise into the original image. That is to say, we could introduce two parameters: width w and stride s to characterize our smaller noise map. For example, for a typical image $I \in \mathbb{R}^{224\times224}$, we will have a noise map $\phi(n) \in \mathbb{R}^{w\times w}$ where $w < 224$ and shift this $\phi(n)$ as a filter across the entire I with a stride s. Our ultimate goal is to find the smallest iteration step with respect to each location that this noise map has been applied to this image. Then, we should conclude the area should correspond to the salient part of the image for classification. Here, we have varied w and s in experiments so as to find optimal value $w = 100, s = 50$ where we should have maximal difference between each region of the image. The results could be illustrated in Fig. 7 and Fig. 8. We could easily see that the noise map has been applied to obvious abnormal area of the retinal images, where gives us clear explanation how the classification model diagnose the image as particular labels.

2) Saliency Attack: In order to define an explanatory rule for the original classification model, local attack is simply too rough for deeper saliency detection. Therefore, a meaningful image perturbation should learn which regions of the image I are used by the black box to produce the output label. As stated in section II-B, a possible sensitivity map could be extracted by differentiating the class activation function S_c with respect to the input image I. We could treat the sensitivity map as an original saliency mask to multiply with original noise $\phi(n)$ to be $\Phi(I, n) = M_c(I) \times \phi(n)$. Then, we could always recalculate the saliency mask at each iteration step to mute some of the pixels from the original noise and

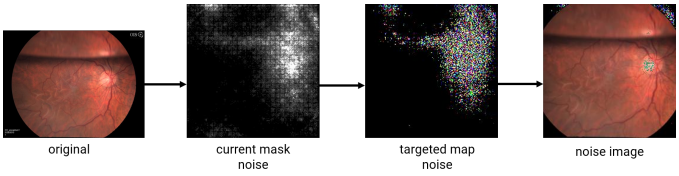


Fig. 9: Output image for saliency mask attack.

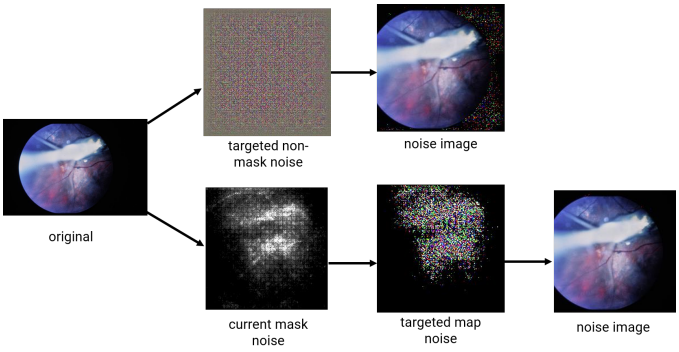


Fig. 10: The comparison with saliency mask attack and targeted un-mask attack.

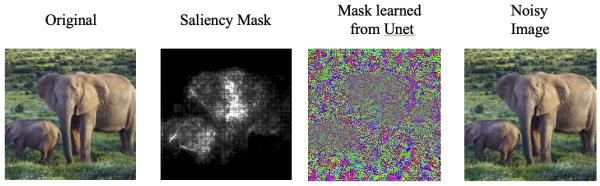


Fig. 11: The adverasrial attack with mask trained with U-Net on a normal image.

introduce in the image. Eventually, our goal is to find out the final $\Phi(I, n)$ that will perform the fastest attack on the original image which gives the better explanatory pixels that depict the classification mechanism. The results are illustrated in Fig. 9 and 10. The current mask noise is the final saliency map we found from the classification model, where the local pixels with high intensity values serve as a mask for the random noise. Eventually we could add the targeted map noise on the original image to generate a noisy image and check the iteration to stop when attack succeeds. We could see mainly the final targeted map noise focuses attack on the red veins across the eyeball which interprets some abnormal instances around the regions. In Fig. 10, we could see if we directly use targeted non-mask noise to attack the model with a smaller scaling factor α, some noise defects could be observed on the right side of the image which may downgrade our attack mechanism. On the contrary, if we apply saliency mask on the generated noise, barely visible changes happen in the noisy image. Also, if we use same image but apply a random mask with same amount of pixels, we should have longer iteration steps for successful attack as stated in table I.

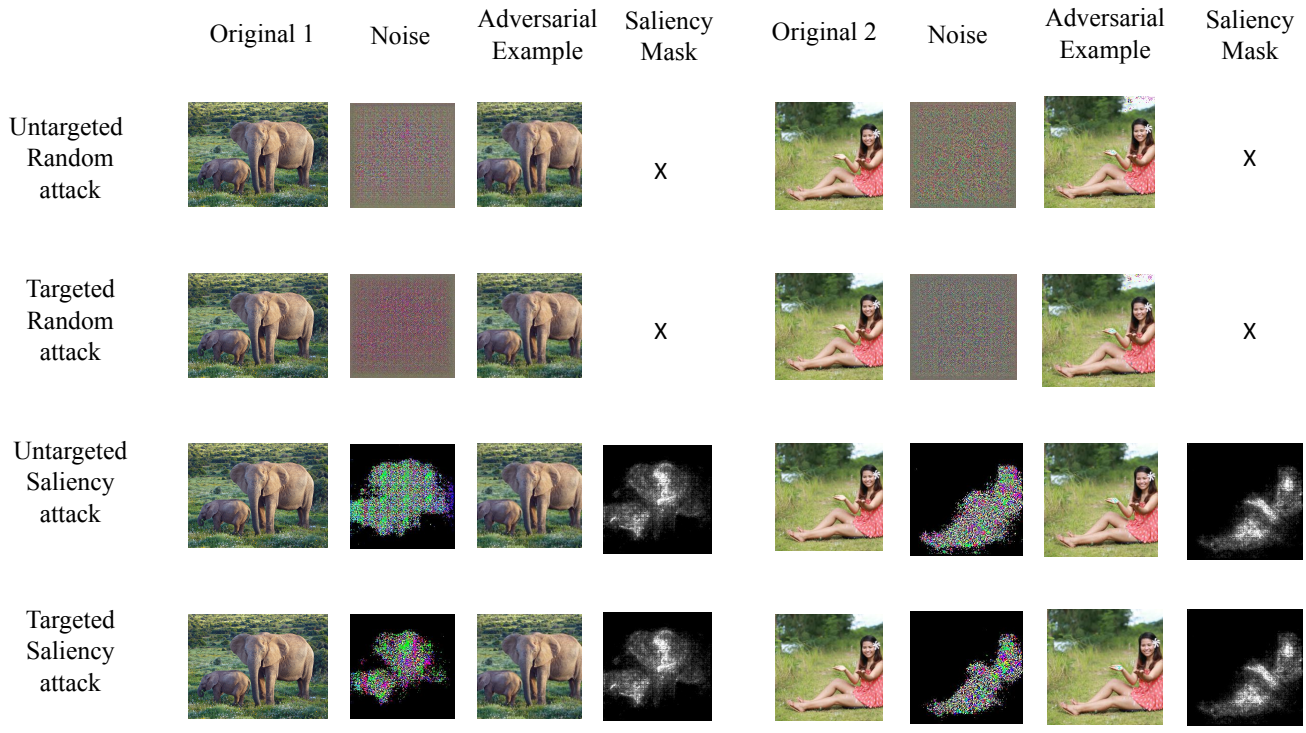


Fig. 12: The overall attack mechanism of the project on a random given image.

TABLE I: This table is to show some example parameters associated with performing untargeted and targeted attack

	α	Origin Loss	Later Loss	Label Change	Iteration times	Confidence Score
Random: Untargeted Attack	0.02	$3.91 * 10^{-5}$	-6.3	From 81 To 44	5	0.9868
Random: Targeted Attack	0.02	$5 * 10^{-4}$	0.07	From 99 to 10	17	0.9688
Mask: Saliency Attack	0.05	$5.72 * 10^{-6}$	0.093	From 187 to 10	48	0.9108
Mask: Random Mask Attack	0.05	$5.72 * 10^{-6}$	0.1018	From 187 to 10	54	0.9032

Moreover, to justify our results, we also perform our attack mechanism on normal images with the resnet50 model pretrained with ImageNet dataset. The results are illustrated in Fig. 12. We could observe that the mechanism still well states in original imagenet dataset where we could observe the local noises focusing on objects instead of background. In addition, intuitively targeted attack should put more efforts than untargeted attack since label and confirmation score change should be more significant in such cases. Therefore, untargeted attack takes less efforts on concentrating the noise regions compared with targeted attack, which results in more ambiguous regions. Targeted attack, instead, narrows down the objects more and gives a more salient noisy mask which helps us to interpret the image much better.

3) Mask Attack: Finally, we would like to incorporate gradient approach to train our own salient mask and discover distinctive region of the image where the model classify. As the loss function we have stated in II-C, we train a mask that could also perform mask attack based on rough gradients. Then we could get the results directly attacking the resnet50 model as illustrated in Fig. 11. We could see that the mask that Unet learns is still noisy compared to the original learned saliency mask at section III-C2. But we could still see some blurry region that corresponds to the saliency mask that makes more reasonable local attack on the original image. In the future work, we could further denoise the learned mask and try to get a clearer image to demonstrate its saliency discovery from the original model.

IV. Conclusions

In this project, we perform different attack mechanisms to explore saliency for a given image, which may help to clarify abnormality for the security-sensitive image dataset. In sum, saliency attack is more effective than random attack or mask attack to demonstrate great saliency visualization in classification tasks. By training a mask with updated gradients from classifier, we could decently localize important area that should be defended from perturbed attack. However the attack method in this work using Unet does not show significant saliency result in our work, which entailed noisy information on the background. Therefore our works using Unet can be further improve with denoisy techniques by comparing with found saliency mask or revisit the targeted loss function

to get a better visualization results. We have also released our source code on https://github.com/waynewu6250/attack-for-saliency for reference and further improvements.

REFERENCES

[1] D. Smilkov, N. Thorat, B. Kim, F. Viégas, M. Wattenberg, "SmoothGrad: Removing noise by adding noise," Proc. Int. Conf. Mach. Learn. Workshop Vis Deep Learn., 2017.

[2] X. Yuan, P. He, Q. Zhu and X. Li, "Adversarial Examples: Attacks and Defenses for Deep Learning," in IEEE Transactions on Neural Networks and Learning Systems, vol. 30, no. 9, pp. 2805-2824, Sept. 2019.

[3] K. Simonyan, A. Vedaldi, and A. Zisserman."Deep in-side convolutional networks: Visualising image classifica-tion models and saliency maps," InProc. ICLR, 2014.

[4] R. C. Fong, A. Vedaldi, "Interpretable explanations of black boxes by meaningful perturbation," arXiv preprint arXiv:1704.03296, 2017.

[5] A. Adadi and M. Berrada, "Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)," in IEEE Access, vol. 6, pp. 52138-52160, 2018.

[6] C. Zhang, Z. Ye, Y. Wang and Z. Yang, "Detecting Adversarial Perturbations with Saliency," 2018 IEEE 3rd International Conference on Signal and Image Processing (ICSIP), Shenzhen, 2018, pp. 271-275.

[7] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, A. Swami, "Practical black-box attacks against machine learning", Proc. ACM Asia Conf. Comput. Commun. Security, pp. 506-519, 2017.

[8] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, P. Frossard, "Universal adversarial perturbations", Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 86-94, 2017.

[9] J. Su, D. V. Vargas and K. Sakurai, "One Pixel Attack for Fooling Deep Neural Networks," in IEEE Transactions on Evolutionary Computation, vol. 23, no. 5, pp. 828-841, Oct. 2019.