Interpreting Adversarial Attacks in CNNs

Vedasri Nakka
Institut für Informatik, Université de Neuchâtel
Neuchâtel, Switzerland
Email: vedasri.nakka@unine.ch

Prof. Rolf Ingold, Prof. Elena Mugellini and
Prof. Omar Abou Khaled
HumanTech, Human-IST, Hes.so, Université de Fribourg,
Switzerland
Email: rolf.ingold@unifr.ch

***Abstract* — Convolutional Neural Networks (CNNs) is a de-facto standard for vision-related applications. However, the criticism of being a black-box model has been alleviated by few interpretable neural networks. On the other hand, CNN's were shown to be vulnerable to adversarial attacks that add a quasi-imperceptible noise to an input image. In this paper, we study the effect of adversarial attacks in conjunction with interpretable CNNs. Specifically, we understand the reasons for the success of adversarial attacks and visualize the change in discriminative regions. To this end, we make use of three networks namely, Standard VGG16, Attention Pooling framework, and Prototypical Networks. Furthermore, we use off-the-shelf techniques such as CAM and its variants to aid our analysis. We conduct a comprehensive study of interpreting the decisions of CNNs on the CUB dataset with multiple gradient-based adversarial attacks.**

Keywords - Interpretability; Adversarial Attacks; Prototypical Networks; CAM; Grad-CAM; Attention Maps; Fine-grained recognition

I. INTRODUCTION

Recent developments in Convolutional Neural Networks (CNNs) has led to significant developments in various vision related applications such as image recognition [17], semantic segmentation [14] and object detection [16] and to many other related fields [15] Despite the improvements in performance metrics, it is often criticized for being non-interpretable due to abstract high dimensional representation. Further, it is difficult to understand the inference of CNNs; for example, the reason for DNN arrive comes to a given decision for a given input because of non-linear activation functions and complex architectural connections. This major drawback has limited its usability in applications for which the interpretability of decisions is a critical requirement, where a simple black-box prediction is in-sufficient to take a final decision.

Another disadvantage of DNNs is their intrinsic weakness to sources of deliberate noise to trigger DNNs to fool. CNNs have been showed to be vulnerable to adversarial perturbations [7,8,13] which when carefully crafted, with an quasi-imperceptible noise will modify the model predictions. Furthermore, the adversarial examples are shown to be transferable from one network to another network when trained with a different data distribution [18]. Hence, the vulnerability of adversarial attacks poses serious concerns on the security of deployed models in safety-critical applications.

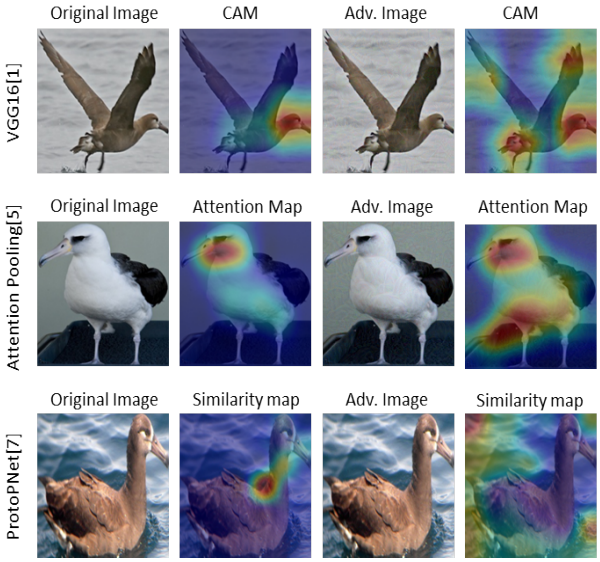


Figure 1. ***Overview of our work***. In the first row, we show the outputs of CAM [1] for normal image and adversarial image. In the second row, we visualize the attention maps [5] for both clean and adversarial attacked image. In the third row, we visualize the top activated prototype from ProtoPNet [6]. All the results consistently show the shift in discriminative regions after the attack to either background regions or to aa large region confusing the image classifier. Note that, all attacks are run with 10 step PGD [10] with perturbation strength ϵ equal to 8.

In this work, we study adversarial attacks through the lens of interpretability. To this end, we study the adversarial attacks on fine-grained dataset using three interpretable CNNs. For example, in Fig. 1, we study the result of adversarial attack on three interpretable CNNs and observe the shift in interpretability through CAM [1], Grad-CAM [2], attention map [5] and prototype similarity maps [6]. Our motivation for this work is the necessity to have deeper understanding on the reasons for the success of adversarial attacks. We have carefully chosen three frameworks that complements each other in the analysis. Our experiments consistently confirm the shift in discriminative regions to a non-sensical regions after the performance of the attack. We believe this can be a guidance to detect thee adversarial attacks. Nevertheless, our comprehensive experiments on CUB12 [19] dataset will

motivate the researchers to study adversarial attacks in conjunction with interpretability.

The report is organized as follows; we first review the related works in Section II and introduce each sub-module at detail in Section III; we then study simple yet powerful gradient based adversarial attacks; finally, in Section IV we perform experiments by conducting adversarial attacks on the interpretable CNN and we conclude with qualitative analysis and future directions in Section V.

II. Related Work

Interpretable CNNs. Earlier works for interpreting the decisions of include performing posthoc analysis for a pretrained CNN. In the posthoc analysis, we understand the filters in trained CNN by maximizing the neuron activation [7] through gradient ascent of input image. Further, proposed deconvolutional network to interpret the filters. While the above methods looks at the filter to understand the internal functioning of CNN, it is shown by the technique of Class Activation Maps (CAM) to understand to regions responsible for predicting to a particular class using the final backbone spatial feature map and final classifier weights. It was then further extended in Grad-CAM which can work for any network without limiting global average pooled networks. This has revolutionized the interpretability to be extended to other topics in vision such as image visualization, VQA, Image Captioning, etc. However, the interpretation in Grad-CAM when the image as multiple occurrences of discriminate regions. It was showed that Grad-CAM do not localize well and does not capture entire object completely. In [3] eliminated this limitation by using pixel-wise weighting of the gradients of the output w.r.t. a particular spatial position in the final convolutional feature map of the CNN.

Another direction in interpreting CNNs is using attention layers to focus on important discriminative regions at multiple layers [4]. Besides, the authors in [5] propose aa simple modification to final layer to get spatial attention map as a score of for object-ness. Both CAM and Attention-based models can only tell us which parts of the input they are looking at – they do not point us to prototypical cases to which the parts they focus on are similar. Further, [7] builds upon the earlier works to propose ProtoPNet where it provides with visual explanations of the network's decisions using trained class-specific prototypes.

Adversarial Attacks. Convolutional Neural Networks were first exposed to vulnerabilities to adversarial perturbations in [7] where quasi-imperceptible noise is added to fool the system. Similar attacks were then extended to many vision related applications such as for object detection, semantic segmentation and other tasks. Initially, a fast attack named FGSM [8] is studied using gradients of loss with respect to input which was further extended in BIM [9] method using multi-step approach. Later, several variants to this iterative method is proposed using either momentum [10], saliency [12]. Further, [13] proposed arguably the strongest adversarial attacked called PGD attack which uses the maximum loss increment within given bound at each iteration. Despite adversarial training is proven to be effective to alleviate the effects of attacks, it is shown that defenses are vulnerable to white box attacks.

III. Methodology

In this section, we first review the tools for visualization and then explain deeply the working of three interpretable CNNs that we chose to conduct experiments. Finally, we introduce the adversarial attack methods that are based on gradients.

A. *Class Activation Mapping (CAM)*

In CAM [1], the authors proposes to avoid using fully connected layers to limit the large number of trainable parameters while maintaining superior performance. To accomplish this, they utilize global average pooling (GAP) which acts as a primary regularizer, thereby reducing the over-fitting during training.

As shown in Fig. 2, a class activation map is computed by the weighted sum of feature maps. These weights are taken from the classification layer associated with each label. Formally, given an image x with associated label y, the extracted feature map from the backbone is global average pooled. Here, $\boldsymbol{f_k(x, y)}$ is the activation of unit k at particular location. Then the output of feature map is multiplied with weights corresponding to that of class c to get class activation map. We denote M_c as the class activation map of class C as below

$$M_c(x, y) = \sum_k w_k^c \, f_k(x, y) \tag{1}$$

where $M_c(x, y)$ denotes class activation map for class c at location (x, y); w_k^c weight of class c feature map; $f_k(x, y)$ denotes the activation of unit k at particular location; f_k dimensions are given as below,

$$f_k \in R^{H \times W}$$

$$w_k^c \in R^{K \times 1}$$

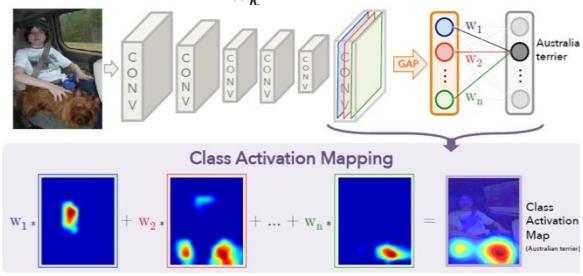


Figure 2: ***CAM framework***. Framework for computing Class Activation Maps using the weights of final classification layer weights.

We visualize the discriminative regions for two classes on the below Fig. 3 where the left image shows the salient regions responsible for predicting as brushing teeth. Similarly, the second image contains the regions responsible for the label cutting trees.

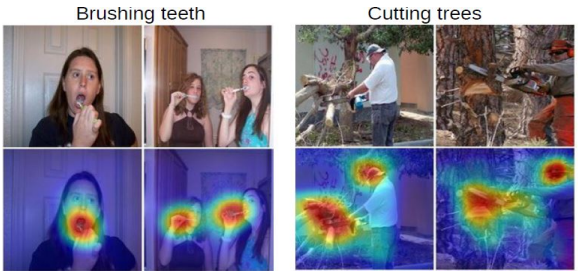


Figure 3: ***CAM visualization***. Discriminative regions highlighted in red color using CAMs.

*B. **Grad-CAM**:*

Grad CAM [2] utilizes the gradient data streaming into the last convolutional layer of the CNN to allot significance values to every neuron for a specific choice of interest. We present the overview of Grad-CAM in Fig. 4.

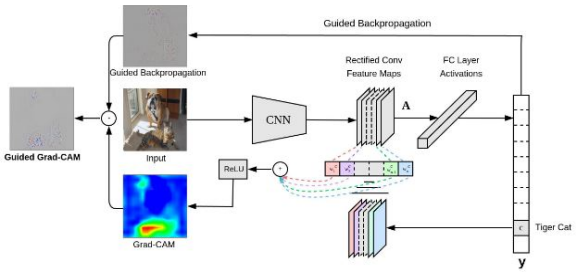


Figure 4: ***Grad-CAM framework***. G-CAM Overview with tiger cat example.

We initially estimate the gradient of the score for class c for the logit y_c, w.r.t. feature map activation A_k of a particular convolutional layer. Then, we employ the global average pooling over width and height. Where α_k^c indicates the weight of a feature map k for the target class c; Z denotes the number of feature maps; y^c denotes output for class c; A_{ij}^k denotes feature map activation k at i and j location; $\frac{\partial y^c}{\partial A_{ij}^k}$ denotes calculate gradients of class c with respect to feature map k;

$$\alpha_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k} \quad (2)$$

We obtain the weights α_k^c while backpropagating gradients w.r.t the activation map till the final convolution layer. We assume these gradients as the significance feature map k for an output class c. Then, finally, we perform a ReLU activation map to get only positive values of the weighted sum of the feature map.

$$L_{Grad-CAM}^c = ReLU\left(\sum_k \alpha_k^c A_k^c\right) \quad (3)$$

where $L_{Grad-CAM}^c$ location of $Grad-CAM$ for class c; $ReLU$ denotes rectified linear unit; $\sum_k \alpha_k^c A_k^c$ denotes weighted sum of feature map of k with its weights for class c;

In the same paper, the author further extended with the approach of proposing guided Grad-CAM. The key takeaway from this approach, we can easily localize the discriminative region by additional user-level inputs. For example, in the Fig. 5, the second column the contributing features for Dog class for the given input image, the third column shows the Grad-CAM results followed by the final column which locates the important regions for Dog class at a finer level.

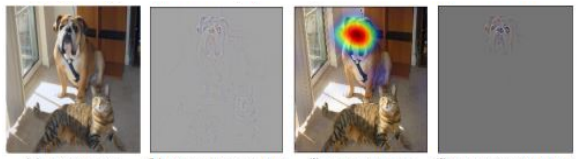


Figure 5. ***Guided Grad-CAM Visualization***. From left to right; **1.** Original Image **2.** Guided back propagation **3.** G-CAM **4.** Guided Grad-CAM.

*C. **Grad CAM++**:*

Both the CAM and Grad-CAM suffers from an important drawback. For instance, if there were multiple events of an object occur i.e., more than one discriminate location present in the image, the visualization is flawed or limited to few small regions. This issue can be fixed by taking a weighted average of the pixel-wise gradients. Specifically, authors in [3] reformulate the weight of G-CAM equation by expressly coding the construction of the weights w_c We can see the difference between G-CAM and G-CAM++ in Fig. 6 where G-CAM is able to locate limited discriminate region, whereas in G-CAM++, it is able to locate all discriminate regions even the input image as more than one discriminative location. The difference can show in Figure 6.

A_{ij}^k is 1 if visual patter of the input image is detecting, otherwise it is 0.$\frac{\partial Y^c}{\partial A_{ij}^k}$. We calculate gradients of output class c with respect to each feature map. $\alpha_{ij}^{kc} \in R^K \times 1$

$$w_k^c = \sum_i \sum_j \alpha_{ij}^{kc} . relu\left(\frac{\partial Y^c}{\partial A_{ij}^k}\right) \quad (4)$$

where Y^c the output channel is; A_{ij}^k is Feature map over width and height dimensions; α_{ij}^{kc} is weighted coefficients for pixel wise gradients for class c and convolutional feature map A^k;

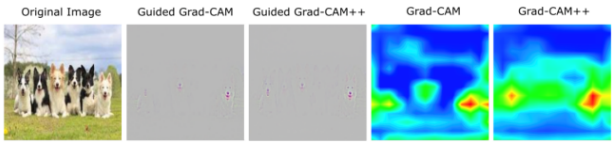


Figure 6: ***Discriminative regions of G-CAM, G-CAM++***. The original image has more than one discriminate region's (here for e.g. dogs). Grad-CAM captures only limited discriminate

regions, but Grad CAM++ can captures whole discriminative regions present in the image.

D. *Residual Attention Network*:

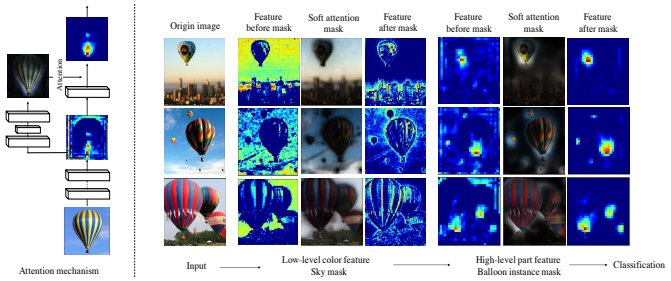


Figure 7: ***Residual Attention Network***. We visualize the outputs at few intermediate layers before and after passing through the attention branch.

Here, the authors in [4] propose a Residual Attention Network as shown in Fig 7. which employs multiple Attention Modules at multiple layers. The benefits of such a network are in two folds: it can capture mixed attention and is an extensible convolutional neural network. The first benefit lies in that different Attention Modules capture different types of attention to guide feature learning. The second benefit comes from encoding top-down attention mechanism into bottom-up top-down feedforward convolutional structure in each Attention Module. Thus, the basic Attention Modules can be combined to form a larger network structure.

E. *Attention Pooling*:

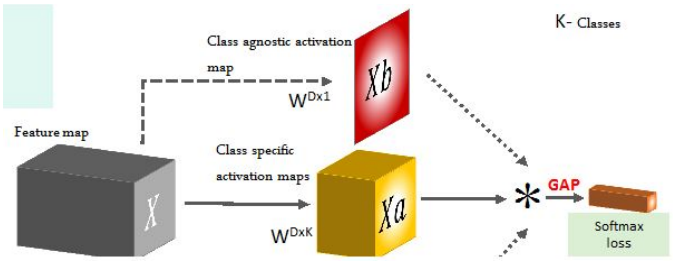


Figure 8: ***Attention Pooling Framework***.

Further, in [5], the authors propose a simple yet surprisingly powerful network modification that learns attention maps which focus computation on specific parts of the input relevant to the task at hand. This requires no additional supervision and automatically leads to significant improvements over the baseline architecture. The formulation as shown in Fig 8. can be viewed as a novel factorization of attentional processing into bottom-up saliency joined with top-down attention.

F. *ProtoPNet: Interpretable Fine-Grained Network*:

Unlike earlier approaches which visualize the decisions of CNN is post-hoc training stage, ProtoPNet [6] explains the decisions of CNN through class-specific prototypes in a feed-forward manner. For the given input image for e.g. the bird image, the model can distinguish a few parts of the picture where it feels that this part of the picture resembles that prototypical part of some class, and makes its forecast dependent on a weighted mix of the similarity scores between parts of the picture and the learned models. Thus, the decisions of ProtoPNet [6] are interpretable using the similarity score associated with activated prototypes.

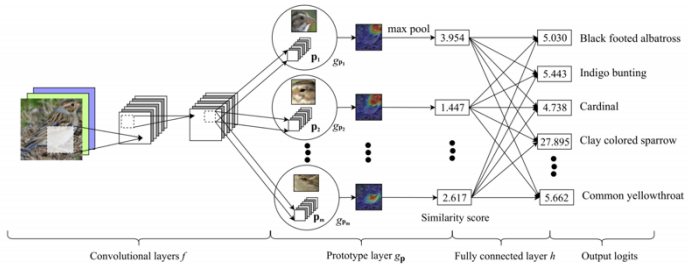


Figure 9: ***ProtoPNet***. Architecture of ProtoPNet which uses class-specific prototypes to predict the label

We briefly summarize the working of ProtoPNet. As shown in Fig. 9, the input image is first passed through the backbone network to get the convolutional spatial feature map. The resulting backbone feature is passed through the prototype layer which contains class-specific prototypes to obtain the similarity maps. Note that, the prototype layer calculates the L2 distance between patches of the feature map and prototypes of different classes. These similarity maps are spatially max-pooled to get the similarity score of the input image with respect to each prototype. Finally, the similarity scores are connected to output logits either by a positive connection if the prototype belongs to same class as that of output logit or negative weight otherwise. Formally, the ProtoPNet solves the below objective function:

$$\min_{P,w_{conv}} \frac{1}{n}\sum_{i=1}^{n} CrsEnt\big(h \circ g_p \circ f(x_i), y_i\big) + \lambda_1 Clst + \lambda_2 Sep \tag{5}$$

where $CrsEnt$ denote the cross-entropy loss between the predicted and true label; $Clst$ represent the cluster loss and Sep represents the Separation loss; h denotes the classification layer; g_p as the prototype layer; $f(x_i)$ as the backbone layer and y_i is the true label. Furthermore, the regularization losses are given as below;

$$Clst = \frac{1}{n}\sum_{i=1}^{n} \min_{j:P_j \in P_{y_i}} \min_{z \in patches(f(x_i))} \|z - p_j\|_2^2 \tag{6}$$

$$Sep = -\frac{1}{n}\sum_{i=1}^{n} \min_{i:p_j \notin P_{y_i}} \min_{z \in patches(f(x_i))} \|z - p_j\|_2^2 \tag{7}$$

Where P_{y_i} denotes set of prototypes that belongs to classy_i; $f(x_i)$ is the feature map after backbone; $p_j$denotes the learned prototype; $z$denotes patches of convolution output;

Essentially, the clustering loss pulls the discriminative regions in a sample close to the nearest prototype of its own class. On the other hand, the separation loss pushes discriminative regions away from the nearest prototype of any other class. We maximize the separation loss by a negative sign and train the ProtoPNet.

G. *Adversarial Attacks:*

Adversarial attacks were first studied in [7] for image classification and further extended to different applications. [21,22]. The aim of adversarial attack is to add quasi-imperceptible noise to the input so that resulting predictions are changed. For example, if we give input image with ground truth label as panda, by performing adversarial attacks to the machine leaning it predicts output as a different label, gibbon. We study different types of gradient based methods in our experiments. They are Fast Gradient Sign Method (FGSM)[7], Basic Iterative Method (BIM)[8], Momentum Iterative Method (MIM)[9], and Projected Gradient Descent (PGD)[10].

a. ***FGSM:***

FGSM is a single step gradient-based adversarial attack which takes input image and computes the gradient of loss with respect to input image. In addition, we pass the gradients into the signum function and multiply with epsilon ϵ perturbation budget as given below

$$x_{adv} = x + \epsilon \cdot \text{sign}(\nabla_x L(x, y)) \quad (8)$$

Where x_{adv} denotes adversarial sample of input image x; ϵ denotes perturbation strength; ∇_x denotes gradient of loss w.r.t to the input image; $L(x, y)$ is cross entropy loss between predicted output and true output; and y denotes true label.

b. ***BIM:***

It is similar to the FGSM but, in this method we find noise in multiple steps. Specifically, we move the adversarial sample in smaller steps as computed below;

$$x_m = clip_\epsilon(x_{m-1} + \frac{\epsilon}{i} \cdot sign(\nabla_{x_{m-1}} L(x_{m-1}, y))) \quad (9)$$

Where x_m denotes the adversarial sample at iteration m; $clip$ is the function to clip the input image within l_∞ norm; $\nabla_{x_{m-1}}$ denotes gradient of the loss w.r.t. input x; L denotes cross entropy loss; y denotes true label; x_{m-1} denotes the current perturbed image at step $m-1$.

c. ***MIM:***

It extends the BIM approach by an additional momentum term to stabilize the direction of gradient. The adversarial example is as given below;

$$g_m = \mu \cdot g_{m-1} + \frac{\nabla_{x_{m-1}} L(x_{m-1}, y)}{\|\nabla_{x_{m-1}}(L(x_{m-1}, y))\|_1} \quad (10)$$

$$x_m = clip_\epsilon\left(x_{m-1} + \frac{\epsilon}{i} \cdot sign(g_m)\right) \quad (11)$$

where μ denotes decay factor and rest of the terms as denoted in the earlier BIM approach.

d. ***PGD:***

PGD is arguably the strongest attack till date with subtle difference to BIM. It starts from a random position in the clean image neighborhood and then clip the noise in an iterative manner as below;

$$x_m = x_{m-1} + \gamma \cdot sign(\nabla_{x_{m-1}} L(x_{m-1}, y)) \quad (12)$$

$$x_m = clip(x_m, x_m - \epsilon, x_m + \epsilon) \quad (13)$$

Furthermore, for the sake of completeness, it is worth noting about the optimization based adversarial attacks such as CW [11], JSMA [12] which are computationally expensive to perform.

IV. EXPERIMENTS

Experimental Setup. We conduct experiments on VGG16 backbone on mini dataset containing first 12 classes from CUB [19] dataset. We train the networks with Adam optimizer [20] with learning rate of 2e-4 with default momentum. We train on 300 images and test on 299 images. We train 3 networks namely, simple VGG16, Attention pooling framework and ProtoPNet. For ProtoPNet, we set 10 prototypes per class and choose same hyper-parameters. For the adversarial attacks, we set perturbation strength to either {2,8} and run for up to maximum of 10 iterations. In addition, we experiment with following visualization schemes on the three networks,

a. *We visualize the simple VGG16 network with CAM, Grad-CAM and Grad-CAM++*

b. *We visualize the attention-pooling framework with the attention maps coming from the last layer*

c. *For ProtoPNet, we visualize the top-5 activated prototypes for each input image*.

Furthermore, we plot the visualizations for normal and its adversarial counterpart and analyze the shift in the discriminative regions.

Discussion. We report the main results of our work in Table 1. For normal samples, the accuracy of VGG16, AP and ProtoPNet is 88.9%, 84.6%, and 76.9% for VGG16, AP and ProtoPNet, respectively.

We draw following conclusions from our experiments. As expected, we find the adversarial attacks with larger ϵ has decreased the accuracy by a large margin. Further, we observe the AP is generally robust to adversarial attacks which indicates that the simple formulation to last layer in AP [5] has resulted in larger robustness to attacks. On the other hand, ProtoPNet while being interpretable has shown to be vulnerable among the three networks. Nevertheless, ProtoPNet has additional usefulness in interpreting the decisions with activated class-specific prototypes.

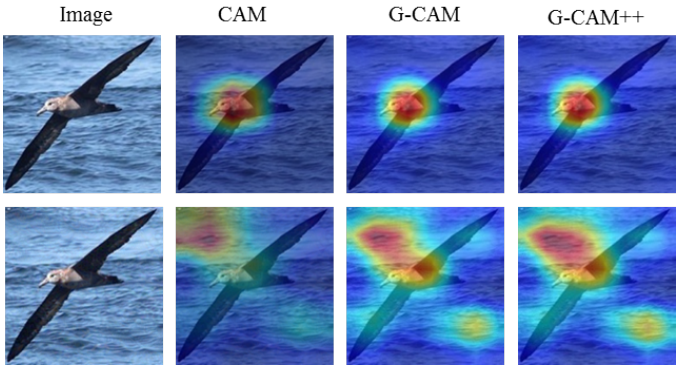
Network	Clean	FGSM (2,1)	FGSM (8,1)	BIM (2,1)	BIM (8,1)	PGD (2,1)	PGD (8,1)	MIM (2,1)	MIM (8,1)
VGG16 [1]	**88.9**	39.4	21.1	10.4	3.01	5.35	2.1	13.4	3.0
AP [4]	84.6	42.1	33.1	**20.7**	**16.7**	**17.4**	**14.7**	**22.7**	**17.4**
ProtoPNet[6]	76.9	**44.1**	**37.7**	4.3	3.34	3.68	0.3	2.6	1.6

Table 1: ***Experiment results***. We report accuracy (in %) on CUB12 test-set samples containing 299 images with four different attacks. Note that, (2,1) notation denotes perturbation strength ϵ of 2 units and attack is run for total of 10 iterations. Our experiments indicate the AP network is robust to adversarial attacks relatively than the other two.

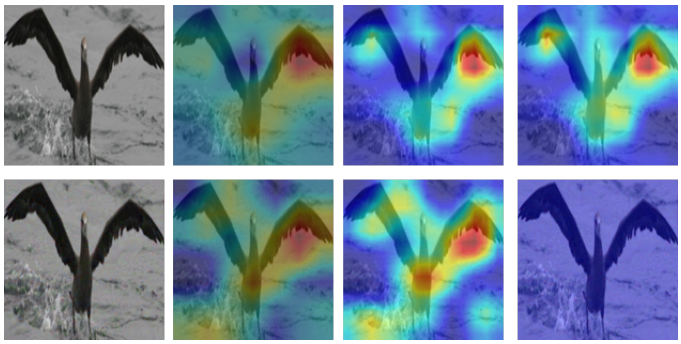
V. QUALITATIVE RESULTS

We visualize the results of shift in attention for each experiment with 10 step PGD attack with ϵ=8 in this section. The results of shift of CAM and its variants are shown in Fig. 10. Further, we report the attention map for AP framework in Fig. 11. Finally, we show the results on ProtoPNet in Fig. 12. Overall, we observe the common trend of shift in the discriminative regions from a finer region in clean image to either background regions or larger object region which confuses the classifier. For example, in Figure 9, we observe the CAM focuses on a small region discriminative to the input image whereas by perturbations the attention is moved to different regions such as to background area. Similar analysis can be understood for the AP and ProtoPNet where the top activated prototypes completely focuses on a different region for adversarial image.

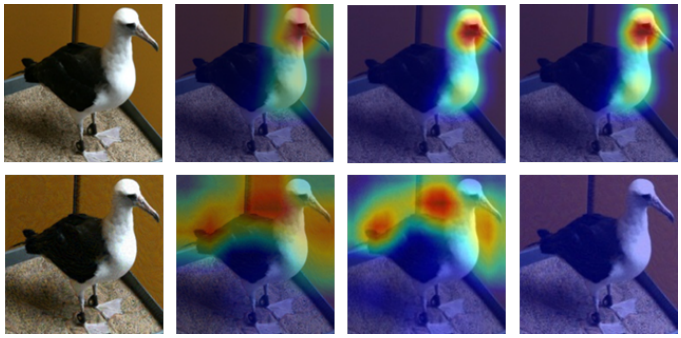
A. CAM Visualization on VGG16:



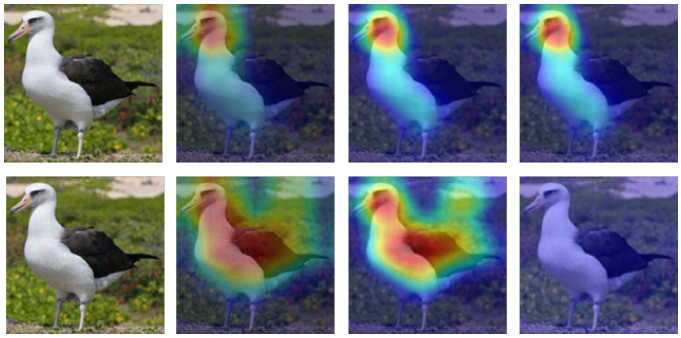
(a)



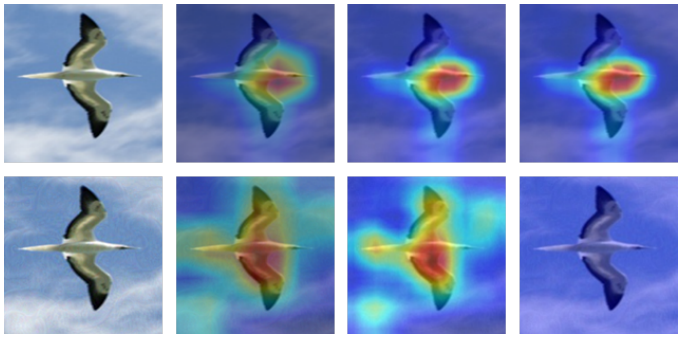
(b)



(c)



(d)



(e)

Figure 10: ***Visualization with CAM, Grad-CAM and Grad-CAM++ approaches on VGG16 baseline.*** In each block containing two rows, the first row the indicates the results with normal image; while the second row shows the results with adversarial images with 10-step PGD attack with ϵ=8. In addition, first column shows the image and the last three

columns visualizes the heatmaps for CAM, Grad-CAM, and Grad-CAM++, respectively.

B. *Attention Pooling Visualization:*

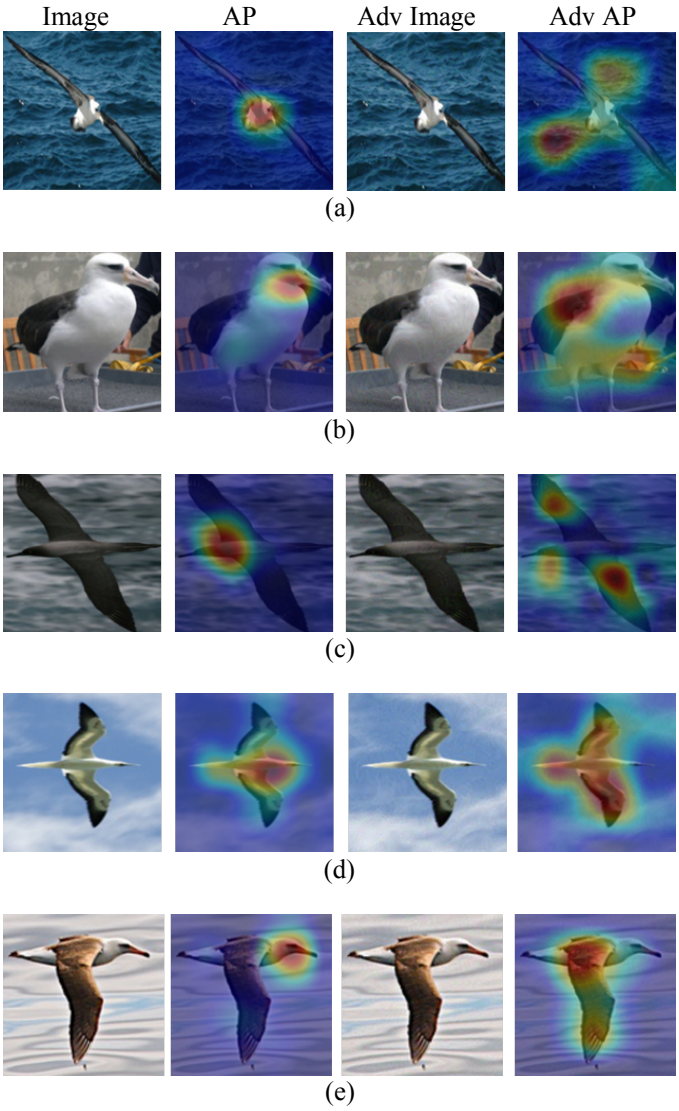
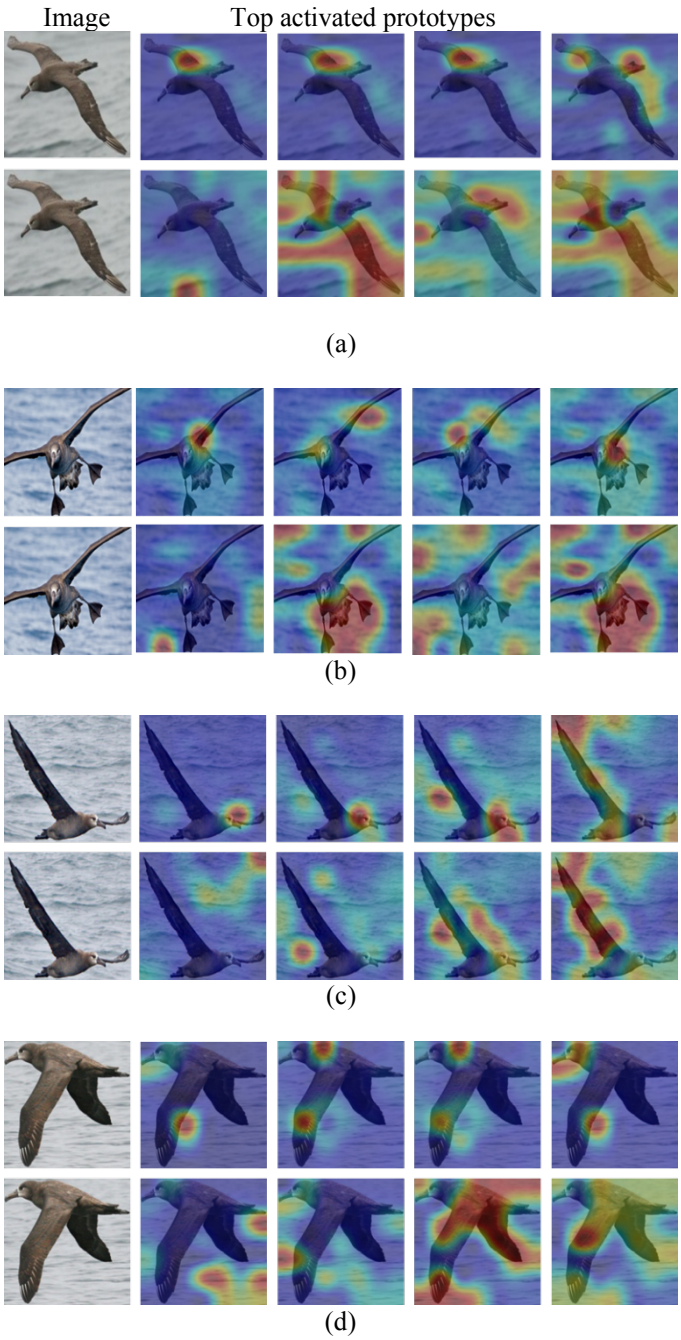
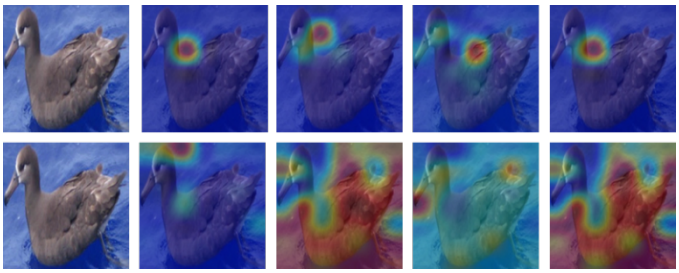


Figure 11: ***Visualizations of Attention Pooling network.*** In each row, the first column indicates original image; second column shows the discriminate regions for the predicted label; third column indicates the adversarial image with 10-step PGD attack with ϵ=8; and the final column shows the discriminative regions after applying adversarial noise to the image.

C. *ProtoPNet Visualizarion:*

Here, in Figure 10, we visualize top activated prototypes for both normal image and adversarial image.





(e)

Figure 12: ***Visualizations of ProtoPNet network.*** In each block, the first column contains the clean image and rest of the columns are top activated prototypes. Similarly, the second row contains the same for adversarial image with 10-step PGD attack with ϵ=8 and its top activated prototypes.

FUTURE WORK

Through series of experiments, we studied the high attention regions in the case of adversarial attack. The common takeaway from each experiment is the shift in discriminative regions to either background regions or large object regions. In the future, we believe the adversarial attacks can be detected by visualizing the discriminative regions responsible for predicting the label. In essence, the shift in discriminative regions to non-sensical regions can be leveraged to detect the attacks, which we believe can alert the system to fallback to other alternatives in the case of adversarial attack.

CONCULSION

In this work, we studied the reasons for success of adversarial attack through the lens of interpretable networks. Specifically, we employed three frameworks and conducted the experiments with four gradient based attacks including well-known PGD based attack. Our results indicate the AP framework is more robust to attacks than the other two frameworks. Moreover, we visualize the shift in discriminative regions after the adversarial attacks. Interestingly, we find the discriminative regions in the case of attack has moved to largely background regions or even covering large regions, thereby confusing the network.

ACHKNOWLEDGEMENT

I would like to thank Prof. Rolf Ingold, Prof. Elena Mugellini, Prof. Omar Abou Khaled, and for their support and suggestions throughout the semester for the seminar Explainable AI.

REFERENCES

[1] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, Antonio Torralba, Learning deep features for discriminative localization, In CVPR 2016.

[2] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam , Devi Parikh, Dhruv Batra, Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization, In ICCV 2017

[3] Aditya Chattopadhyay, Anirban Sarkar, Prantik Howlader, Vineeth N Balasubramanian, Grad-CAM++: Improved Visual Explanations for deep convolutional network, In CVPR 2018.

[4] Fei Wang, Mengqing Jiang, Chen Qian1, Shuo Yang, Cheng Li1, Honggang Zhang , Xiaogang Wang , Xiaoou Tang, Residual Attention Network for Image Classification , In CVPR 2017.

[5] Rohit Girdhar, Deva Ramanan, Attentional Pooling for Action Recognition, In NIIPS 2017.

[6] Chaofan Chen, Oscar Li, Chaofan Tao, Alina Jade Barnett, Jonathan Su, Cynthia Rudin, This Looks Like That: Deep Learning for Interpretable Image Recognition, In NIPS 2019

[7] Lan J Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In ICLR (International Conference on Learning Representations) 2015.

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

[9] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li. Boosting adversarial attacks with momentum. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 9185–9193, 2018.

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

[11] N. Carlini and D.Wagner. Towards evaluating the robustness of neural networks. In 2017 IEEE Symposium on Security and Privacy (SP), pages 39–57. IEEE, 2017.

[12] Rey Wiyatno and Anqi Xu, Maximal Jacobian-based Saliency Map Attack, arXiv 2018.

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

[14] Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431-3440. 2015

[15] Ji, Shuiwang, et al. "3D convolutional neural networks for human action recognition." *IEEE transactions on pattern analysis and machine intelligence* 35.1 (2012): 221-231.

[16] Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun. "Faster r-cnn: Towards real-time object detection with region proposal networks." ." *Advances in neural information processing systems* 25 (2012): 1097-1105.

[17] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." *Advances in neural information processing systems* 25 (2012): 1097-1105.

[18] Mopuri, Konda Reddy, Utsav Garg, and R. Venkatesh Babu. "Fast feature fool: A data independent approach to universal adversarial perturbations." *British Media Vision Conference* (2017).

[19] Wah, C., Branson, S., Welinder, P., Perona, P., & Belongie, S. (2011). The caltech-ucsd birds-200-2011 dataset.

[20] Kingma, D.P. and Ba, J., 2015, January. Adam: A Method for Stochastic Optimization. In *ICLR 2015*.

[21] Arnab, A., Miksik, O., & Torr, P. H. (2018). On the robustness of semantic segmentation models to adversarial attacks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 888-897).

[22] Xie, Cihang, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan Yuille. "Adversarial examples for semantic segmentation and object detection." In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1369-1378. 2017