A VULNERABILITY OF ATTRIBUTION METHODS USING PRE-SOFTMAX SCORES

MIGUEL LERMA[1] AND MIRTHA LUCAS[2]

Abstract. We discuss a vulnerability involving a category of attribution methods used to provide explanations for the outputs of convolutional neural networks working as classifiers. It is known that this type of networks are vulnerable to adversarial attacks, in which imperceptible perturbations of the input may alter the outputs of the model [2]. In contrast, here we focus on effects that small modifications in the model may cause on the attribution method without altering the model outputs.

1. Introduction

The black box nature of current artificial intelligence (AI) models is considered problematic in areas with low tolerance to errors, such as Computer Aided Diagnosis (CAD) and autonomous vehicles. To palliate the effect of mistakes and increase confidence in the model, explanation methods have been developed to justify the model outputs [1].

A class of explanation methods widely used on convolutional neural networks (CNN) take the form of attribution methods that determine how much different parts of the input of a model contribute to produce its final output. In general, the networks on which these methods are used consist of several convolutional layers that produce a vector of outputs $\mathbf{z} = (z_1, z_2, \dots, z_n)$, which is then transformed with a softmax function into a vector of probabilities $\mathbf{y} = (y_1, y_2, \dots, y_n)$, where n is the number of classes. (Figure 1). Each post-softmax output can be interpreted as the amount of confidence about the input sample belonging to each of the several classes $1, 2, \dots, n$. In classification tasks, the output with maximum value corresponds to the class to which the input sample is considered to belong.

Gradient-based attribution methods for convolutional networks work by computing the gradient $\nabla_{\mathbf{x}} S = (\partial S/\partial x_1, \dots, \partial S/\partial x_N)$ of an output or "score" S of the network respect to a set of inputs or unit activations $\mathbf{x} = (x_1, \dots, x_N)$, where N is the number of inputs or internal units, and S may represent either one of the pre-softmax outputs z_i, or one of the post-softmax outputs y_i. The assumption is that each derivative $\partial S/\partial x_i$ provides a measure of the impact of x_i on the score S. A few examples of attribution methods using this approach are Grad-CAM [8], Integrated Gradients (IG) [11], and RSI Grad-CAM [6].

[1]Northwestern University, Evanston, USA
[2]DePaul University, Chicago, USA
E-mail addresses: [1]`mlerma@math.northwestern.edu`, [2]`mlucas3@depaul.edu`.
Date: July 6, 2023.

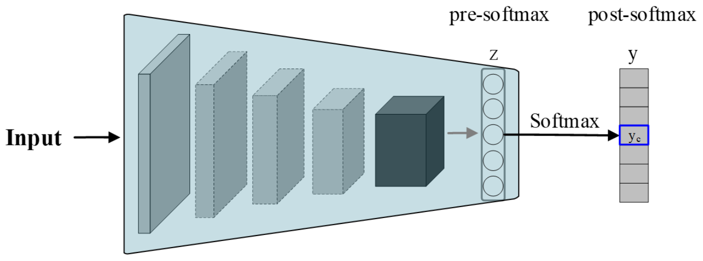

FIGURE 1. Structure of a typical classifier network. After a number of convolutional blocks this kind of network ends with a fully connected network producing a (pre-softmax) output $\mathbf{z}$, followed by a softmax activation function with (post-softmax) output $\mathbf{y}$.

In [5] there is a detailed analysis of the differences between using gradients of pre-softmax versus post-softmax outputs. In that paper it is argued that the post-softmax version of gradient-based methods is more robust and not affected by a vulnerability suffered by the pre-softmax version. Here we will provide a brief overview of the main argument leading to that conclusion, and a way in which the vulnerability could be exploited.

2. A vulnerability of attribution methods using pre-softmax scores.

In this section we examine a vulnerability that affects attribution methods for CNNs that work with pre-softmax scores, with a special emphasis on gradient-based methods, although many of the considerations can be easily extended to methods that work with finite differences rather than gradients, such as Layer-wise Relevance Propagation (LRP) [7] and DeepLIFT [9].

2.1. **The softmax function.**

The output of the softmax function applied to a vector $\mathbf{z} = (z_1, z_2, \dots, z_n)$ is the vector $\mathbf{y} = (y_1, y_2, \dots, y_n)$ whose components are:

$$y_c = \frac{e^{z_c}}{\sum_{i=1}^{n} e^{z_i}} \,. \tag{1}$$

The outputs of the softmax verify $0 < y_c < 1$ for all classes $c = 1, \dots, n$, and $\sum_{c=1}^{n} y_c = 1$, so the y_c are usually interpreted as probabilities.

Note that adding an amount t independent of the class i to all the arguments of the softmax, $z'_i = z_i + t$, has no effect on its outputs:

$$y'_c = \frac{e^{z'_c}}{\sum_{i=1}^{n} e^{z'_i}} = \frac{e^{z_c+t}}{\sum_{i=1}^{n} e^{z_i+t}} = \frac{e^t\, e^{z_c}}{\sum_{i=1}^{n} e^t e^{z_i}} = \frac{e^t\, e^{z_c}}{e^t \sum_{i=1}^{n} e^{z_i}} = \frac{e^{z_c}}{\sum_{i=1}^{n} e^{z_i}} = y_c \,. \tag{2}$$

So, the change $z_i \mapsto z_i + t$ for every i does not change the network post-softmax outputs y_c. Note that t does not need to be a constant, all that is required is that t is independent of i.

Since adding t has no effect in the output of the softmax, the derivatives of the outputs of the softmax won't change after adding t to its arguments:

$$\frac{\partial y_i'}{\partial x} = \frac{\partial y_i}{\partial x} \, ,$$

however the derivatives of the pre-softmax z_i may change:

$$\frac{\partial z_i'}{\partial x} = \frac{\partial (z_i' + t)}{\partial x} = \frac{\partial z_i}{\partial x} + \frac{\partial t}{\partial x} \, ,$$

so that $\frac{\partial z_i'}{\partial x} \neq \frac{\partial z_i}{\partial x}$ if $\frac{\partial t}{\partial x} \neq 0$.

This theoretical result and its potential impact in gradient-based attribution methods are carefully examined in [5]. In the following section we will provide a proof of concept showing how this results can be used to radically modify a heatmap produced by an attribution method such as Grad-CAM.

2.2. **A vulnerability of attribution methods using pre-softmax scores.** Equation (2) shows that the softmax function has no unique inverse because we can add to its arguments $z_1, \ldots, z_n$ any scalar t independent of i without changing the output of the softmax.

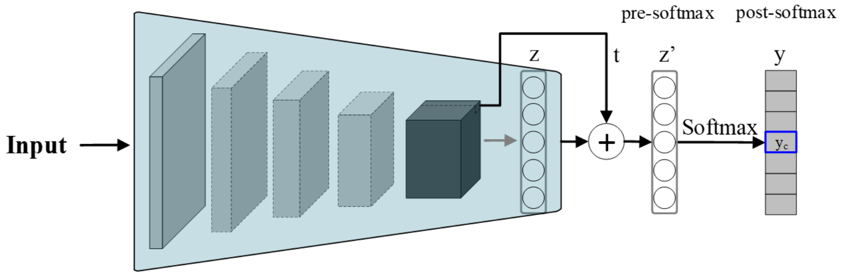

FIGURE 2. Example of alteration of a classifier network that changes attributions based on pre-softmax scores without changing post-softmax scores.

In the example shown here (Figure 2) the network is a VGG19 pretrained on ImageNet [10]. Then, t is the result of adding the activations of the units placed in position $(0, 0)$ of the final pool layer (block5_pool) across all its channels multiplied by a constant K. More specifically, if A_{ijk} presents the activation of unit in position (i, j) of channel k of the last pooling layer, then:

$$t = K \sum_k A_{00k} \, ,$$

where K is a constant—in our experiment we used $K = 10$.

After t is added to the original z_i pre-softmax scores of the network we get new pre-softmax scores $z_i' = z_i + t$. This makes the new pre-softmax scores strongly dependent on the units in position $(0, 0)$ of the final pool layer without altering the post-softmax scores of the network. Consequently, we expect that heatmaps produced by Grad-CAM to strongly highlight the upper left area of the image regardless of whether that part of the image is related to the network final output.

Figures 3–5 show that, for the altered model, the heatmaps produced using pre-softmax scores are strongly distorted, while the heatmaps produced using post-softmax scores remain unchanged.

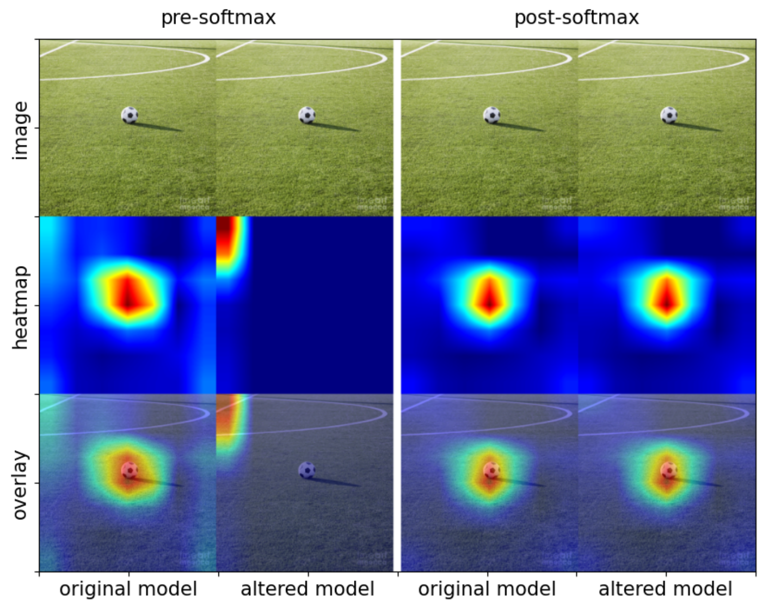

FIGURE 3. Heatmaps produced by Grad-CAM using pre-softmax and post-softmax outputs respectively, intended to locate the position of the soccer ball. The original model is a VGG19 network pretrained on ImageNet. The altered model is the same VGG19 network slightly modified, but still functionally equivalent (same final outputs) to the original network. The heatmaps are computed at the last convolutional layer of each model. Note that Grad-CAM working on pre-softmax outputs has been tricked to produce wrong heatmaps. The heatmaps obtained using post-softmax outputs remain unchanged.

On the other hand, since the final (post-softmax) output of the network remains unchanged, the loss function used for training would sit on the same local minimum for both models (original and modified). Further training of the models won't make a difference since the added connection cannot backpropagate error. More specifically, if E is the loss function used for training, then for the modified model we have (using multivariate chain rule):

$$\frac{\partial E}{\partial t} = \sum_{i=1}^{n} \frac{\partial E}{\partial y'_i} \frac{\partial y'_i}{\partial t} = 0$$

because $y'_i = y_i$, which does not depend on t, hence $\frac{\partial y'_i}{\partial t} = \frac{\partial y_i}{\partial t} = 0$ for all i. Consequently, the trainable parameters of both models would change in the same way, and if the error function E is at or near a minimum for the original model, the same would hold for the modified model. Also, if we trained the modified VGG19 network from scratch and with the same parameter initialization, the final trainable parameters would be the same as those of the original VGG19.

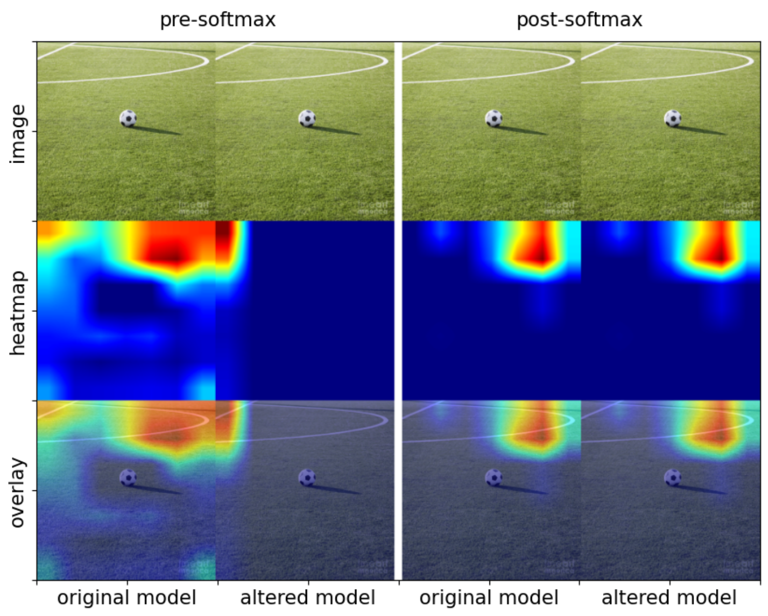

FIGURE 4. The altered model tends to produce the same heatmap regardless of the class assigned to the image. In this case Grad-CAM is used to locate a "maze" rather than a soccer ball in the image. The pre-softmax version of the heatmap on the altered model keeps highlighting the same upper left corner, while the other heatmaps focus on the lines drawn on the grass.

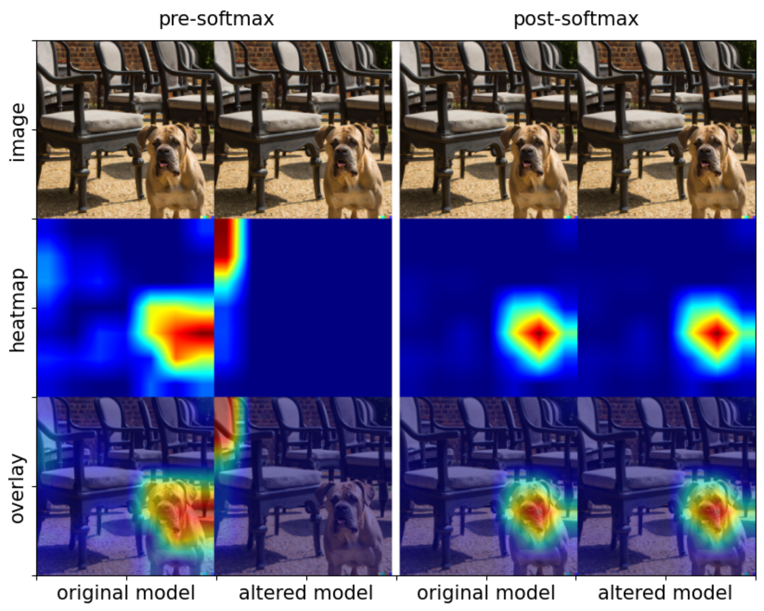

FIGURE 5. Another example showing the heatmap computed with pre-softmax outputs of the altered model concentrated in the upper left corner of the image. Heatmaps computed with post-softmax outputs remain unaltered highlighting the position of the dog.

3. Discussion

We note that the main property behind the vulnerability shown here is the possibility of altering pre-softmax scores of a classifier CNN without altering its post-softmax scores. One question could be whether this vulnerability can be exploited to deploy a malicious attack intended to undermine confidence in the model. This kind of attack would be available for anybody having access to model repositories. Since after modification the new model would be functionally equivalent to the original one (its outputs will not change) it would be hard to notice that it has been modified. Also, it is conceivable that the problem pointed out may manifest itself in an unintended way because, after training, both the original and modified model may end up at the same local minimum of the loss function used for training.

The phenomenon discussed may seem to have some similarities with *Clever Hans* effects [3,4], which also causes heatmaps to highlight wrong areas of the input. Clever Hans effects are due to the ability of a classifier to exploit spurious or artifactual correlations. For instance, in a dataset in which images of horses contain a watermark, the model may learn to correctly classify the image of a horse by paying attention only to the presence of the watermark rather than the horse. In that case, an appropriate attribution method would consistently highlight the area of the watermark in the images with horses, which is outside the actual area of interest. However, that would not happen because of a problem in the attribution method, which would be correctly revealing a problem with the model (trained with a biased dataset). On the contrary, the vulnerability discussed here tells nothing about the ability of the model to extract the right information from the right parts of its inputs, it only depends on the fact that the gradients of the pre-softmax scores may not provide the right information to determine the impact of the inputs on the final (post-softmax) outputs.

4. Conclusions

We have shown that attribution methods using pre-softmax scores are vulnerable to a class of adversarial attacks that may modify the heatmaps produced without changing the model outputs. Post-softmax outputs are not vulnerable to this kind of attack. We have also noted that the vulnerability discussed here is not a Clever Hans effect. Future work can be used to determine in what extent the problem applies to a wider class of attribution methods.

References

[1] Burkart N., Huber M.F. (2021). A Survey on the Explainability of Supervised Machine Learning. Journal of Artificial Intelligence Research, Volume 70, pp 245–317. https://doi.org/10.1613/jair.1.12228

[2] Goodfellow, I.J., Shlens, J., and Szegedy, C. (2014). Explaining and Harnessing Adversarial Examples. CoRR, abs/1412.6572.

[3] Lapuschkin S., Binder A., Montavon G., Müller K.R. and Samek W. (2016). Analyzing classifiers: fisher vectors and deep neural networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2912–2920 (2016).

[4] Lapuschkin, S., Wäldchen, S., Binder, A. et al. (2019). Unmasking Clever Hans predictors and assessing what machines really learn. Nat Commun 10, 1096 (2019). https://doi.org/10.1038/s41467-019-08987-4

[5] Lerma, M., Lucas M. (2023). Pre or Post-Softmax Scores in Gradient-based Attribution Methods, What is Best? Accepted for presentation in the IEEE 13th International Conference on Pattern Recognition

Systems (ICPRS), July 4th-7th, 2023, Escuela Superior Politécnica del Litoral (ESPOL), Guayaquil - Ecuador

[6] Lucas, M., Lerma M., Furst, J., Raicu, D. (2022). RSI-Grad-CAM: Visual Explanations from Deep Networks via Riemann-Stieltjes Integrated Gradient-Based Localization. In: Bebis, G. et al (Ed.), Advances in Visual Computing. ISVC 2022. Lecture Notes in Computer Science, vol 13598. Springer, Cham. https://doi.org/10.1007/978-3-031-20713-6_20

[7] Montavon, G., Binder, A., Lapuschkin, S., Samek, W., Müller, KR. (2019). Layer-Wise Relevance Propagation: An Overview. In: Samek, W., Montavon, G., Vedaldi, A., Hansen, L., Müller, KR. (eds) Explainable AI: Interpreting, Explaining and Visualizing Deep Learning. Lecture Notes in Computer Science(), vol 11700. Springer, Cham. https://doi.org/10.1007/978-3-030-28954-6_10

[8] Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D. (2019): Grad-CAM: visual explanations from deep networks via gradient-based localization. Int. J. Comput. Vision 128(2), 336–359 (2019). https://doi.org/10.1007/s11263-019-01228-7

[9] Shrikumar A., Greenside P., Kundaje A. (2017). Learning important features through propagating activation differences. In Doina Precup and Yee Whye Teh (eds.), Proceedings of the 34th International Conference on Machine Learning, volume 70 of Proceedings of Machine Learning Research, pp. 3145–3153, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[10] Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2015). https://arxiv.org/abs/1409.1556

[11] Sundararajan, M., Taly, A., Yan, Q.: Axiomatic attribution for deep networks. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 70, pp. 3319–3328. PMLR (2017).