Explaining Neural Network Model for Regression

Mgane Millan

Sorbonne Universite, CNRS UMR 7222, ISIR
F-75005, Paris, France
Email: millan@isir.upmc.fr

Catherine Achard

Sorbonne Universite, CNRS UMR 7222, ISIR
F-75005, Paris, France
Email: catherine.achard@sorbonne-universite.fr

Abstract—Several methods have been proposed to explain Deep Neural Network (DNN). However, to our knowledge, only classification networks have been studied to try to determine which input dimensions motivated classification decision. Furthermore, as there is no ground truth to this problem, results are only assessed qualitatively in regards to what would be meaningful for a human.

In this work, we design an experimental database where ground truth is reachable: we generate ideal signals and disrupted signals with errors and train a neural network that determines the quality of said signals. This quality is simply a score based on the distance between the disrupted signals and the corresponding ideal signal. We then try to find out how the network estimated this score and hope to find time-steps and dimensions of the signal where errors occurred. This experimental setting enables us to compare several methods for network explanation and to propose a new method, named Accurate GRAdient (AGRA), based on several trainings, that decreases noise present in most state-of-the-art results. Comparative results show that the proposed method outperforms state-of-the-art methods for locating time-steps where errors occur in the signal.

I. INTRODUCTION

Machine learning is increasingly present in today's life since the arrival of the first Convolutionnal Neural Networks (CNN) [1]. Performances achieved by such networks are impressive and have led to their development in many applications, such as smart vehicles. Despite these performances, errors still exist and can have dramatic consequences, especially for applications where lives are at stake. Furthermore, in medical fields, for example, it is desirable not only to have a final classification result but also to know the causes of the decision. For all these reasons, more and more research is being conducted on DNN explanation, as mentioned in recent literature reviews [2], [3], [4].

To our knowledge, all these methods try to explain DNN trained for classification task: the goal is to find out which elements of the input led to the decision of the network. Unfortunately, no ground truth exists. Therefore, network-explanation results are only evaluated by looking at the produced maps and comparing them to what a human operator believes to be correct. Without an objective tool that quantifies results, it is difficult to compare the results of different methods.

In this article, we propose to build an experimental setup, associated with a ground truth, to quantify explanation results of networks. This setup aims at estimating signals quality: we created a database of ideal signals to which errors were added at random positions. A note is associated to each signal, depending on the distance of an example to its ideal version. A CNN is trained in regression to find this note. Then, the network explanation aims at determining which part of the input (temporal position and dimension) occasioned the score provided by the network. Such a setup with a ground truth enables us to compare quantitatively different DNN-interpretability algorithms.

In order to determine time-steps and dimensions of the input signal where errors occurred, we do a gradient descent that transforms the input to a signal that has the best possible note. This gradient descent enables us to have a gradient according to the input signal. Such a strategy is not new and it is known that these gradients are very noisy [5], [6]. During our experiments, we have found that these gradients vary a lot depending on training and weight initialisation. Actually, training the same model several times on the same database leads, for a given input example, to gradients that change a lot from one model to another. Some model gradients find some errors but not others, and some are very noisy while others are not, etc. We chose to take advantage of these variations to estimate an "accurate gradient" from all the models. The proposed method, named Accurate GRAdient (AGRA), consists in averaging the gradients generated by the different trainings and weight initialization for the same input signal.

Thanks to our experimental setup, we quantitatively compare AGRA with several gradients-based methods and show its efficiency. Moreover, AGRA can be combined to other gradient-based methods to improve their performance.

Thus, two main contributions are proposed in this article. First, we develop an experimental database that allows to qualitatively and quantitatively compare DNN explanation methods. Second, we introduce a new DNN explanation technique AGRA, based on gradients, that outperforms state-of-the-art methods.

II. RELATED WORK

A. Explaining Deep Neural Networks methods

Several methods exist in the literature to explain DNN. Their goal is to find the contribution of each input feature to the output and thus, to produce attribution maps. Methods can be grouped into three main categories: class activation based approaches, perturbation based approaches and gradient based approaches.

1) Class activation based approaches: Methods such as Class Activation Map (CAM) [7], Gradient-weighted Class Activation Mapping (Grad-CAM) [8], or Uncertainty based Class Activation Maps (U-CAM) [9] propose to generate Class Activation Maps that highlight pixels of the image the model used to make the classification decision. The goal is thus to produce maps similar to human attention regions. These maps are estimated in a multi-class classification context and are class-discriminative.

2) Perturbation based approaches: The idea of these approaches is to disturb some portions of the input image and look at their influence on the output. Work in [10] consists in systematically occulting different portions of the input image with a grey square, and monitoring the output of the classifier. As the probability of the correct class drops significantly when the object is occluded, this technique localizes objects in the scene. Another approach, based on perturbation, proposed by Ribeiro *et al.* [11], is the Local Interpretable Model-Agnostic Explanation (LIME). A model is explained by perturbing the input and constructing a local linear model that can be interpreted. Thus, LIME makes local approximations of the complex decision surface.

3) Gradient based approaches: Simonyan et al. [12] proposed to compute sensitivity maps as the gradient of the output according to input pixels in a classification task. If $S_c(x)$ is the score function of the classification network for the class c and input image x, then sensitivity maps are defined as:

$$M_c(x) = \frac{\partial S_c(x)}{\partial x}. \tag{1}$$

By intuition, important gradient values correspond to locations in the image that have a strong influence on the output.

In practice, these sensitivity maps are very noisy. A first solution to improve them is to change the back-propagation algorithm. Thus, deconvolution networks [10] and Guided Backpropagation [10] propose to discard negative gradient values during the back-propagation step. The idea is to keep only entries that will have a positive influence on the score.

Another problem with gradient-based techniques is that the score function S_c may saturate for important input characteristics [13]. Thus, the function may be flat (but important) around these inputs and thus, has a small gradient. Some methods address this problem by computing the global importance of each pixel. Thus, DeepLIFT (Deep Learning Important FeaTures) [14] decomposes the output prediction by back-propagating contributions of all neurons in the network to every feature of the input.

Layer-wise relevance propagation (LRP) [15] uses a pixel-wise decomposition to understand the contribution of each single pixel of the input image x to the score function $S_c(x)$. A propagation rule, applied from the output back to the input, distributes class relevance found at a given layer onto the previous layer. It leads to a heatmap that highlights pixels responsible for the predicted class.

Three other methods, based on the classical back-propagation algorithm, exist to explain DNN: Gradient $\times$ Input [14], Integrated gradient [16] and SmoothGrad [5].

Gradient $\times$ Input [14], [15] was proposed to improve attribution maps. They are simply computed as the product between the gradients of the output with respect to the input and the input itself:

$$GradInput(x) = M_c(x) \times x \tag{2}$$

Instead of computing the gradients of the output according to the input pixels x, Sundararajan et al. [16] integrate the gradients along a path from a baseline x' to the input x. The integrated gradient, for the ith dimension of the input x is defined as:

$$IntGrad_i(x) = (x_i - x_i') \times \int_1^{\alpha=0} \frac{\partial S_c(x' + \alpha(x - x'))}{\partial x_i} d\alpha \tag{3}$$

where $\frac{\partial S_c(x)}{\partial x_i}$ is the gradient of S_c according to x along the ith dimension.

During computation, the integral is approximated via a summation: gradients at the N points lying on the straight line from the baseline x' to the input x, are added. Integrated gradients add up to the difference between the outputs S_c at x and the baseline x'. Thus, if the baseline has a near-zero score, integrated gradients form an attribution map of the prediction output $S_c(x)$.

Given the rapid fluctuations of the gradient for an input image x, it is less meaningful than a local average of gradient values. Thus, SmoothGrad [5] proposes to create an improved sensitivity maps based on a smoothing of $\partial S_c(x)$ with a Gaussian kernel. As the direct computation of such a local average in a high-dimensional input space is intractable, Smilkov et al. compute a stochastic approximation by taking random samples in a neighborhood of the input x and averaging the resulting sensitivity maps:

$$SmoothGrad(x) = \frac{1}{N} \sum_{i=0}^{N} M_c(x + \mathcal{N}(0, \sigma^2)) \tag{4}$$

where N is the number of noised inputs, and $\mathcal{N}(0, \sigma^2)$ is a Gaussian noise with a 0 mean and a σ standard deviation.

In this article, we also propose to use a gradient-based approach and to denoise the so-obtained gradient. The proposed approach, based on several trainings, can be combined to other gradient-based methods to improve their performances.

III. AGRA METHOD TO OBTAIN ACCURATE GRADIENT

In this work, we first propose to design an experimental setup to explain DNN. Then, we introduce a new method allowing to denoise the gradient of the output according to an input using several trainings of the same DNN.

A. Designing an experimental setup

A problem often encountered with DNN explanation algorithms is the lack of ground truth. It is therefore difficult to quantitatively estimate the performance of such algorithms. To

address this issue, we design a setup where this ground truth is available. This setup is composed of 2D temporal signals. Both dimensions are generated using sinusoids with different lengths, to which a small Gaussian noise has been added. These signals represent ideal signals in the database. Then we artificially create perturbations in both dimensions by adding high-frequency Gaussians. The number of perturbations varies uniformly between 0 and 8 and their position and the dimension where they appear are also drawn according to a uniform law.

A score, re-scaled between 0 and 10, is then given to each of these signals. This score is based on the Mean Square Error (MSE) between the signal without perturbation and the disrupted signal. 0 is attributed to ideal signals while score gets close to 10, when many perturbations are present. 1000 signals are thus generated, 750 are used for training and 250 for testing, drawn according to a uniform law.

The goal of the network will then be to regress the score of each input signal while the goal of the DNN explanation will be to find time-steps and dimensions of the errors. Three examples of signals extracted from the database are presented in Figure 1.
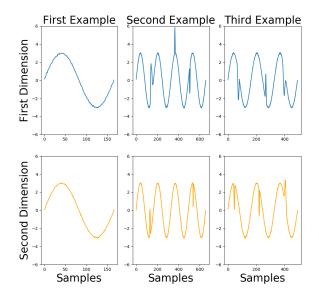

Fig. 1. Examples of one ideal signal (left) and two perturbed signals (middle with 5 pertubations and right with 6 pertubations)

Even if we are working on synthetic examples with a ground truth regarding the DNN explanation, this setup corresponds to a real application that aims to determine the quality of gestures in sports [17] or surgical context [18], for instance. In addition to assigning a score, DNN explanation will make it possible to determine where gestures are poorly carried out.

B. The AGRA method

First a CNN is trained to regress the scores with a MSE loss between the predicted scores $\hat{s}(x)$ and the scores of the

ground truth $s(x)$: $l_1(x) = (\hat{s}(x) - s(x))^2$. Then, for DNN explanation, a gradient of the output according to the input example x, as that proposed in [12], is computed, without changing the weights of the networks. It is used to change the input x so that its note increases. As the goal is to find differences difference between ideal signals and perturbed ones, the loss used for gradient back-propagation is the MSE between the predicted score and the optimal note (0 in our case): $l_2(x) = (\hat{s}(x) - 0)^2$. Several iterations are done until the ideal note is reached as explained in Algorithm 1 where λ is the learning rate and ϵ is the tolerance: loop stops when the loss is below ϵ.

Algorithm 1 Compute Gradient

Input: x, λ, ϵ
Output: $GRAD(x)$
  $x' = x$
  while $l_2(x') > \epsilon$ **do**
    $grad = \frac{\partial l_2(x')}{\partial x'}$
    $x' \leftarrow x' - \lambda \times grad$
  end while
  GRAD(x) = x - x'

Unfortunately, and as stated before, this gradient is very noisy [5], [6]. Moreover, during our experiments, we observed that it depends significantly on weights initialisation and training of the network. Thus, even if two different trainings lead to similar regression scores, gradients are highly variable. Two examples of gradient can be found on Figure 2.

We decided to take advantage of these variations and average gradients of different models with different trainings, to obtain a noise-reduced and more accurate gradient. So, we trained N times the same network to obtain N models. Let $GRAD_i$, the gradient of the output according to the input, obtained with model i, as described in the algorithm 1. AGRA is then obtained as described in Algorithm 2. AGRA method needs several trainings of the same model, which is computationally expensive. However, as shown in Figure 2, the so-obtained gradients are more accurate. Moreover, they no longer depend on training and initialisation, which was the case before when either good or bad gradients were obtained.

Algorithm 2 Compute Accurate Gradient: AGRA

Input: x
Output: $AGRA(x)$
  $grad = 0$
  for $it = 0$ to N **do**
    $grad = grad + GRAD_{it}(x)$
  end for
  $AGRA(x) = grad/N$

IV. EXPERIMENTAL RESULTS

For all methods involved in this section, we use the loss function $l_2(x)$ previously defined to compute gradients.
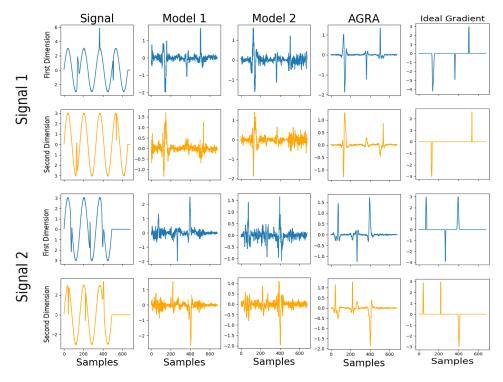
Fig. 2. Gradients obtained from two models with different initialization and with the AGRA method.

A. Training procedure

The regression network consists of four temporal convolutional layers with $(8, 8, 16, 16)$ filters of size $(25, 5, 5, 5)$, with no bias added. Each of them is followed by a pooling layer with size $(3, 3, 3, 3)$. Two fully connected layers with 50 and 1 neurons end the neural network with between them a dropout layer with a 0.5 probability, with no bias. The network is learnt with adam optimizer [19] and a 0.01 learning rate, for 100 epochs. The network regresses a score between 0 and 10 and is trained 50 times to obtain 50 models. The mean MSE across the 50 models, on the test set, is of 0.619 with a standard deviation of 0.089. So, during prediction, these models have a similar behavior.

B. Qualitative results

Firstly, we present qualitative results of the five following methods:

- Classical gradient GRAD [12] computed with Algorithm 1, a learning rate of 0.1 and a tolerance ϵ of 0.015.
- GRAD $\times$ input x as defined in equation 2 and proposed by [14], [15]
- Smooth gradient [5] estimated as the mean of 50 gradients obtained with Algorithm 1 by adding a Gaussian noise with 0 mean and 0.1 as standard deviation on the input signal (equation 4).
- Integrated gradient [16]. As the proposed network has no bias, the baseline x' is fixed to a zero signal with the same length than x. In these conditions, the score of the baseline is $s(x') = 0$ and integrated gradients can been interpreted as an attribution map of the prediction output

$s(x)$. Integrated gradients have already been multiplied by the input as explained in equation 3.

- The AGRA method with 50 trained models.

As shown in Figure 3, classical gradients (GRAD) are noisy and do not lead to clear and easy to interpret results, since peaks at perturbation locations are sometimes too thin and small and can be considered as noise. Furthermore, multiplying these noisy gradients with the input only makes the results worse. Indeed, interesting peaks are enhanced but global results appear noisier than before. Moreover, the sign of the gradient, which gives information on the direction of the error, is lost due to this multiplication. Using smooth gradient instead of classical gradient gives better qualitative results with considerably less noise than before. However, noise is still present and the results are again difficult to interpret. Moreover, the magnitude of the gradient is often under-estimated. Integrated gradients are very noisy and have peaks at undisturbed positions, making them very difficult to interpret. As they are multiplied by the input signal, the sign of the gradient is lost. As shown in Figure 3, less noisy and more accurate results are achieved with AGRA method. Gradients actually highlight the locations corresponding to perturbations and have the correct direction to reconstruct the ideal signal.

C. Quantitative results

To compare methods more thoroughly, giving quantitative results is crucial. Since ground truth is available for each example, it is possible to compute ideal gradients (the difference between perturbed signals and ideal ones) and compare them
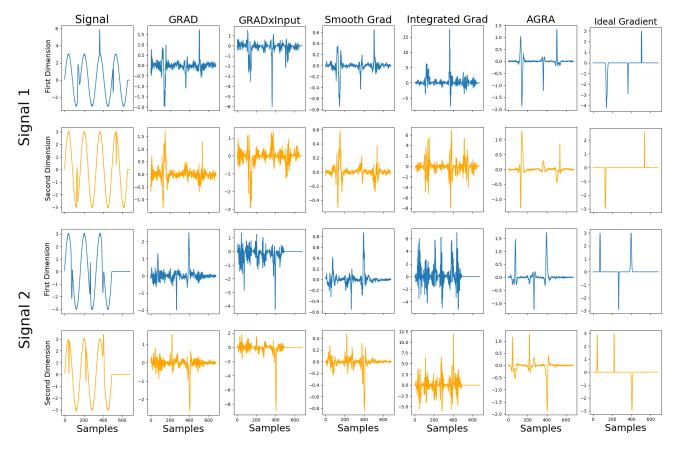
Fig. 3. Results obtained with all the methods on the same two examples.

with results obtained with the different methods. Two metrics are used to make this comparison:

- Mean Squared Error (MSE) between the signal without errors and the reconstructed signal obtained thanks to the gradients. This metric cannot be used for methods such as GRAD×Input or Integrated Gradient, since their goal is only to highlight important time-steps and not to reconstruct a perfect signal.
- Pearson correlation coefficient between the ideal gradient and the gradient obtained with the different methods. To avoid penalising methods, that do not manage the signs (GRAD×Input and Integrated Gradient), this coefficient is computed between the norms of both ideal gradient and gradient from the methods.

The 250 training examples have been averaged to obtain these metrics. Moreover, for GRAD, GRAD×Input, Smooth Grad and Integrated Gradient, metrics have been computed on the 50 trained models and afterwards averaged.

Table I presents the MSE obtained with different methods. As a reminder, an estimated gradient fitting perfectly to the ground truth one would correspond to a 0 MSE. Both GRAD and Smooth Grad methods are noisy. Moreover, Smooth Grad does not keep gradient magnitude. Thus, AGRA method outperforms both of these methods according to MSE. AGRA is therefore the most suitable method for signal reconstruction.

Methods	Mean Squared Error
GRAD [12]	7.65
Smooth Grad [5]	7.85
AGRA	**5.06**

TABLE I
MEAN SQUARED ERROR BETWEEN GROUND TRUTH GRADIENTS AND
ESTIMATED ONES FOR DIFFERENT METHODS.

Methods	Pearson Correlation
GRAD [12]	0.81
GRADxInput [14], [15]	0.82
Smooth Grad [5]	0.79
Integrated Gradient [16]	0.55
AGRA	**0.94**

TABLE II
PEARSON CORRELATION COEFFICIENT FOR DIFFERENT METHODS
ESTIMATED BETWEEN THE NORM OF GRADIENTS

As shown in Table II, Pearson correlation coefficients vary between 0.55 and 0.94. As Pearson correlation coefficients are standardised (the correlation is divided by the standard deviation of both gradients), they can be estimated in a meaningful way for each method, even when the gradient is multiplied by the input. The best results are obtained with our proposed method, which confirms the previous qualitative study and proves that this method gives better results than

Methods	Pearson Correlation
GRAD [12]	0.68
Smooth Grad [5]	0.66
AGRA	**0.84**

TABLE III
PEARSON CORRELATION COEFFICIENT FOR DIFFERENT METHODS
ESTIMATED ON ALL GRADIENT DIMENSION

Methods	Pearson Correlation	MSE
GRAD [12]	0.81	7.65
AGRA	**0.94**	**5.06**
GRAD×Input [14], [15]	0.82	NA
GRAD×Input with AGRA	0.89	NA
Smooth Grad [5]	0.79	7.85
Smooth Grad with AGRA	0.92	6.91
Integrated Gradient [16]	0.55	NA
Integrated Gradient with AGRA	0.82	NA

TABLE IV
PEARSON CORRELATION COEFFICIENT AND MSE FOR DIFFERENT
METHODS COMBINED WITH AGRA FRAMEWORK.

other state-of-the-art methods.

Table III gives the Pearson coefficients obtained by keeping the sign of the gradients when calculating the correlation: the correlation is estimated for each of the two dimensions and then averaged. Using this metric, only Grad and Smooth Grad methods can be evaluated since for the other two, multiplying by the input will change signs of gradient and results will not be exploitable. AGRA is again the most efficient method, even if Pearson coefficient do not take into account gradient magnitude, which does not penalize Smooth Grad as the MSE did.

To study AGRA behaviour, it is interesting to show the evolution of both MSE and Pearson Correlation, according to the number of averaged models (Figure 4). As stated before, gradients are model-dependant. So, MSE, Pearson coefficient and thus the explanation of the network change a lot according to the model. More particularly, it can been seen in Figure 4 that the two first training lead to bad results while the following ones, before the tenth, have a good explanation. Let's remember that the different model changes just by the initialization of the weights. They all have nearly the same regression scores but their gradients change strongly. It is therefore impossible to define *a priori* the models that lead to a good quality gradient. So, in Figure 4, the MSE is important at the beginning and then decreases before stabilizing. Averaging the gradients obtained by 20 or more models produces good explanation results, independent of learning. The same reasoning can be applied to Pearson correlation coefficient.
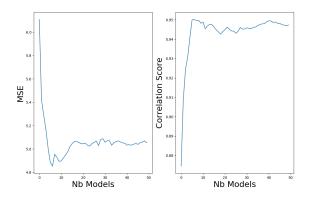
D. AGRA combined with other methods

As stated before, it is possible to combine our approach with different state-of-the-art methods, such as GRAD×Input, Smooth Grad and Integrated gradient, in order to improve both qualitative and quantitative results.

As shown in Figure 5, using the average of 50 models for all methods greatly improves their performances and especially denoises results of every methods. Quantitative results are all improved using AGRA as shown in Table IV, for both Pearson correlation and MSE. This shows that even if this method is computationally intensive, obtained results are really improved compared with state-of-the-art.

V. CONCLUSION

In this paper a new approach to explain neural network decisions has been presented, with a specific experimental setup dedicated to neural network explanation. Indeed, the lack of ground truth for network explanation often only allows a qualitative comparison of different approaches. The design of a synthesis device, devoted to this task, enables quantitative comparisons.

In addition to this new database and experimental setup, a novel approach for network decision explanation has been proposed. Indeed, by observing that the explanation strongly depends on the learning of the model, we proposed to carry out several trainings and then to average explanations provided by each of them. It has been shown that this technique improves both qualitative results - indeed explanations are less noisy - and quantitative results, with better scores for both Pearson correlation and MSE of reconstructed signals. However the drawback of this method, is the high computation cost, since many models need to be trained.

In the future, we plan to extend this approach to models learned in classification to see if the same conclusions can be drawn.

Fig. 4. Evolution of MSE and Correlation according to the numbers of averaged models.

REFERENCES

[1] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
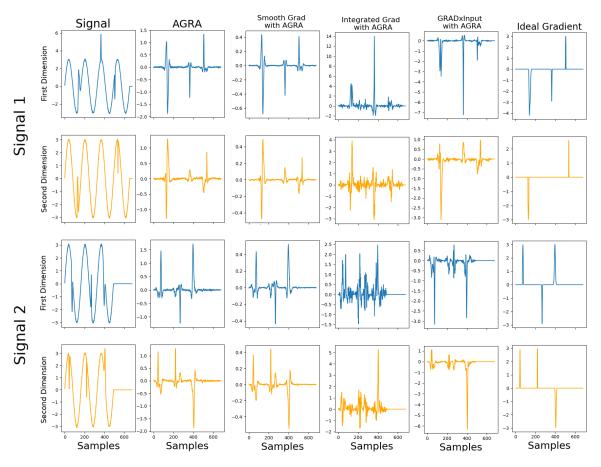
Fig. 5. Results obtained with all the methods averaged over 50 model over the 2 same examples.

[2] F. K. Došilović, M. Brčić, and N. Hlupić, "Explainable artificial intelligence: A survey," in *2018 41st International convention on information and communication technology, electronics and microelectronics (MIPRO)*. IEEE, 2018, pp. 0210–0215.

[3] W. Samek, T. Wiegand, and K.-R. Müller, "Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models," *arXiv preprint arXiv:1708.08296*, 2017.

[4] Q.-s. Zhang and S.-C. Zhu, "Visual interpretability for deep learning: a survey," *Frontiers of Information Technology & Electronic Engineering*, vol. 19, no. 1, pp. 27–39, 2018.

[5] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, "Smoothgrad: removing noise by adding noise," *arXiv preprint arXiv:1706.03825*, 2017.

[6] B. Kim, J. Seo, S. Jeon, J. Koo, J. Choe, and T. Jeon, "Why are saliency maps noisy? cause of and solution to noisy saliency maps," *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pp. 4149–4157, 2019.

[7] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2921–2929.

[8] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 618–626.

[9] B. N. Patro, M. Lunayach, S. Patel, and V. P. Namboodiri, "U-cam: Visual explanation using uncertainty based class activation maps," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 7444–7453.

[10] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*. Springer, 2014, pp. 818–833.

[11] M. T. Ribeiro, S. Singh, and C. Guestrin, """ why should i trust you?" explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144.

[12] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *CoRR*, vol. abs/1312.6034, 2013.

[13] M. Sundararajan, A. Taly, and Q. Yan, "Gradients of counterfactuals," *arXiv preprint arXiv:1611.02639*, 2016.

[14] A. Shrikumar, P. Greenside, and A. Kundaje, "Not just a black box: Learning important features through propagating activation differences," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 3145–3153.

[15] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PloS one*, vol. 10, no. 7, 2015.

[16] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 3319–3328.

[17] M. Millan and C. Achard, "Fine-tuning siamese networks to assess sport gestures quality," in *Proceedings of the 15th International Conference on Computer Vision Theory and Applications*, 2020.

[18] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Evaluating surgical skills from kinematic data using convolutional neural networks," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2018, pp. 214–221.

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2015.