Evaluating correlation between saliency maps and the solution of maze puzzles

Eleftherios Ioannidis, Antonis Michael
Department of EECS
Massachussets Institute of Technology
Cambridge, MA 02139
elefthei@mit.edu, antonism@mit.edu

Abstract

Despite remarkable progress in the area of Convolutional Neural Networks, and despite the ongoing need to create interpretable machine learning models [3, 4], this task remains one of the most challenging ones in the field. Prior work has shown that saliency maps, that is, regions that are particularly important during the process of image classification, could provide meaningful insights to the inner workings of image classification models. That is because they visualize features of the image that also seem important to humans, such as the eyes of an animal figure 1. However, this approach as it is, cannot provide insightful information for images that represent abstract objects such as a maze, for which it is not entirely clear what would count as a meaningful feature. In this paper we aim to show that perhaps we could utilize certain properties of the maze, namely the fact that it has a solution, to show that when classified correctly, the saliency regions used, correlate well with the solution path. This could provide an additional way in evaluating and understanding the classification and misclassification process that doesn't involve features that are directly and immediately visible to humans but are still interpretable when mapped appropriately.

1 Introduction

Although research on data-driven approaches in machine learning is developing rapidly at the moment, complex machine learning models are still often seen as "opaque, uninterpretable black boxes" [5]. Providing good qualitative interpretations of high level features represented by such models remains a challenge [6]. At the same time, there is a growing need for understanding how such models behave, not just for the sake of it but also for overcoming the limitations of current systems by extracting interpretable information [3, 4].

Prior work specifically in the area of deep convolutional neural networks (CNNs) has attempted to address this issue by providing saliency maps that allow humans to visualize, to some extend, the inner workings of image classification models. By highlighting specific pixels in grayscale versions of the input images, the saliency map algorithms aim to show which features where the most important in the CNN classification process [1, 6, 8, 7]. The idea is that these salient regions will be meaningful to humans in order to allow us to evaluate when and why a CNN would misclassify an image. For instance, figure 1 shows the saliency map for the classification process of a cheetah; it is especially clear in the bottom right saliency map that the salient points are the nose, eyes and periphery of the animal, features that are highly interpretable by humans.

The question becomes though, in images where it is not even clear what would *count* as an interpretable feature, how can we meaningfully evaluate the classification process? That is, if the image represents something abstract, such as a maze, then what feature could humans extract such

32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.

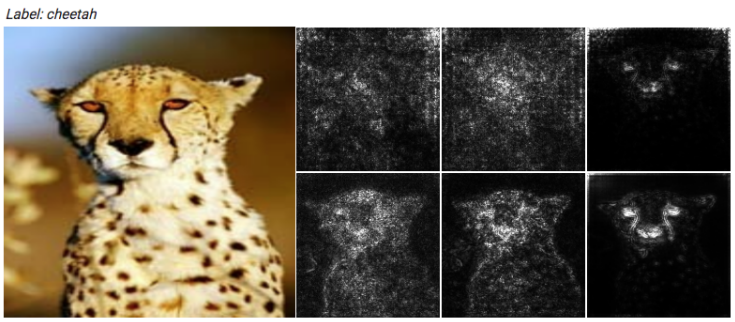

Figure 1: An image of a cheetah on the left, and various saliency maps of the image on the right. Image provided by SmoothGrad [6].

that we can get insights into the classification and misclassification of the image? We believe that positive correlation of the salient regions with essential components that constitute the identity and functionality of the object displayed, could provide good qualitative interpretations. In the case of the maze this translates to evaluating how well salient regions correlate with the solution of the maze.

The success of this could show that when attempting to interpret features used by CNNs, besides what is directly visible to the eye one could attempt to match the features to something else that counts as essential to the identity of the image class.[1] This mapping would provide an additional way in evaluating the mechanisms of CNNs in an indirect but yet interpretable way. To evaluate our hypothesis we need to generate random mazes, solve them, generate saliency maps, both when the maze is classified correctly or not, and then calculate the fraction of salient regions that pass through the solution in each case. Given that the Inception v3 [9] used here was highly successful in the classification process we instead compared the fraction of salient points extracted when the classification was correct against the fraction of random points generated that passed through the solution. Positive correlation between the salient regions and the solution would be an initial good indicator in the evaluation of our hypothesis but of course further work is necessary for more thorough conclusions; we outline such future work at the end of this paper.

2 Methods

The project has five components: (1) A generative model that generates randomized mazes, (2) a maze solver that always finds the shortest path from entrance to exit, (3) saliency map algorithms that identify the salient regions in the maze, (4) a saliency extractor that extracts the most salient points in the image, and finally (5) a checker that calculates the number of salient points passing through the solution. See figure 2 for a block diagram representation of the process.[2]

The maze generator constitutes the *Generative model*, while the solver and the checker as seen in figure 2 contribute to the evaluation of our hypothesis. The most important components are the saliency map algorithms and saliency points extractor, since these are our primary source of data.

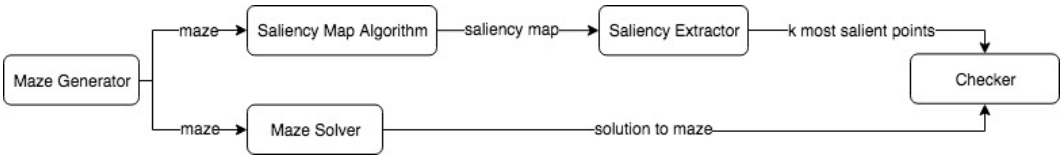

Figure 2: Block diagram representation of the process.

[1] Here we are implicitly assuming that having a continuous and interpretable solution path is an essential feature of what makes a maze, a maze and not just a collection of random black and white spots.

[2] All the code used can be found in the Jupyter Notebook at `https://github.com/Maze-solving-learners/maze-saliency`.

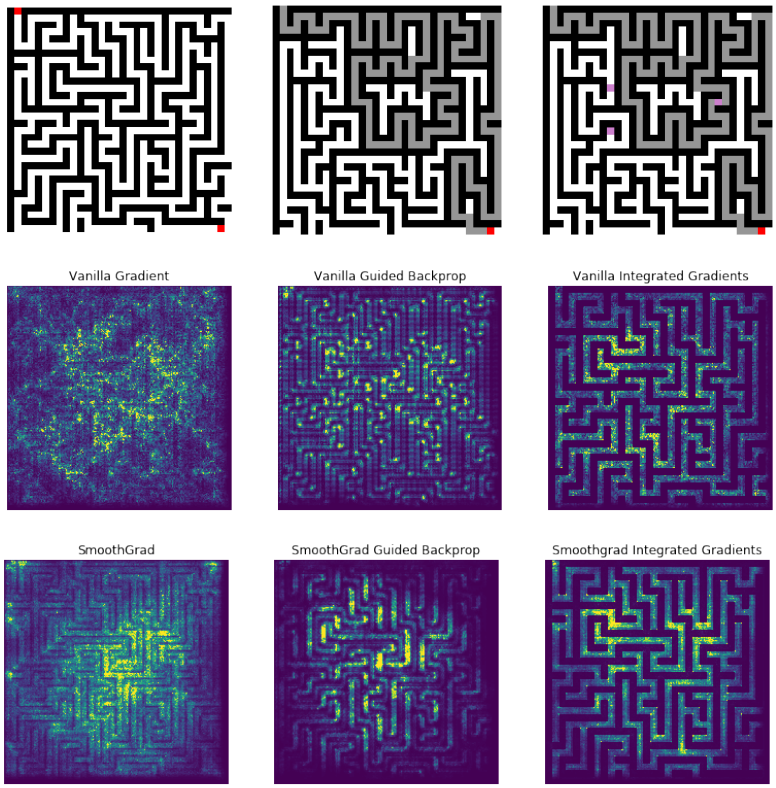

Figure 3: *Top-left*: Auto-generated maze, *Top-middle*: Solved maze by Dijkstra's shortest path, *Top-right*: Salient points from SmoothGrad integrated gradient, *Middle-left*: Vanilla gradient saliency map applied to the original image, *Middle-middle*: Vanilla with guided back-propagation saliency map, *Middle-right*: Vanilla with integrated gradients, *Bottom-left*: SmoothGrad gradient, *Bottom-middle*: SmoothGrad with guided back-propagation, *Bottom-right*: SmoothGrad with Integrated gradients

2.1 Generative model

The maze generative model is written in Python. It is initialized with a random seed and generates random mazes of a given height and width. It uses a modified version of Prim's minimum spanning tree (MST) algorithm [2]. Prim's algorithm ensures one and only one solution from the entrance to the exit exists, with the additional bonus property that no cycles exist in the maze, which prevents any chance of infinite loops. The symbolic graph representation of the maze is stored in a numpy bitmap array of size 32x32 and a scaled-up, 320x320 image version is passed to the Convolutional Neural Network (CNN) for classification. Walls are painted black and traversable paths are marked white, two red pixels are used to mark the entrance to the maze at the top left corner and the exit to the maze always in the bottom right as seen in the top-left image in figure 3.

2.2 Maze solver

The maze solver accepts a symbolic representation of the maze as its input and outputs the shortest path to the exit as a series of cartesian coordinates. The initial maze bitmap representation is preprocessed into an adjacency matrix representation, which becomes the input to Dijkstra's algorithm. The preprocessing that happens translates the 2D array into an undirected, acyclic graph, since the acyclic property is given by construction with Prim's algorithm. Dijkstra's shortest path algorithm computes a dictionary of predecessors for every node, so backtracking from the exit all the way to the entrance by following the predecessors effectively gives the shortest path across the maze, as seen in the top-middle image in figure 3.

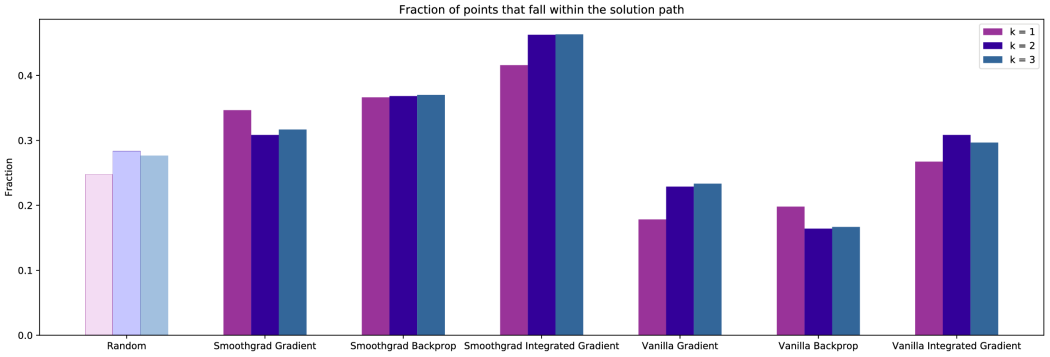

Figure 4: Benchmark results for three values of k, where we picked the k most salient points produced from saliency maps. SmoothGrad overtakes the vanilla salient map algorithms in all cases, with the SmoothGrad with Integrated Gradients surpassing the rest.

2.3 Saliency maps

The saliency map algorithms identify the important (salient) features used by a CNN during the classification process. Here we use the Inception v3 [9] to categorize the input image — the generated maze — and then use two main approaches along with their variants to get its salient map. The first one is a simple algorithm (Vanilla Gradient and its variations) that calculates the gradient of a class prediction neuron with respect to the input pixels [1]. This allows us to see how the class prediction of an image changes when individual pixels are perturbed. Although the modifications of this approach, namely the additions of guided back-propagation [7] and integrated gradients [8] improve the clarity of the output, the results are considerably noisy (see figure 3, top row). The second algorithm we use is the SmoothGrad algorithm [6] that attempts to denoise the saliency maps by adding pixel-wise Gaussian noise to the input image. To do that it takes the image, adds noise to it in order to create similar images and then averages the saliency maps of these images. This produce a crisper output where the brightness of each pixel corresponds to how important that particular pixel is.

2.4 Saliency point extraction

Initially the dimensions of the autogenerated maze are 32x32 but the maze is magnified to 320x320 before passed to the saliency map algorithms. This gives better resolution, while making sure the quality of the maze is not diminished. The bitmap representation chosen for mazes linearly scales up, effectively by duplicating a pixel every time the size of the image doubles. Since we scale the image up 10x, every pixel in the original maze will be a 10x10 block in the saliency map. Scaling the image up allows for more granular control over salient points and allows to bucket the image, then select the block with the highest cumulative saliency. This block maps to the most salient pixel in our original 32x32 representation. An example of going from saliency maps to saliency points can be seen in the top-right image in figure 3.

3 Results

3.1 Checker

To evaluate our model we calculated the ratio of salient points that go through the solution path, and compared it with the fraction of randomly generated points that go through the solution. Specifically, for each set of salient points generated by the saliency map algorithms, we tested how many of the k most prominent ones passed through the solution. We then generated k random points and checked how many of those passed through the solution. For both cases, before checking whether a point passes through the solution we ensured that the point does not hit a wall. If the point was at a black pixel we replaced it with the closest white pixel, that is, with the closest pixel at a distance of one step away that corresponds to a path. The reasoning behind this perturbation is that while some salience algorithms aggressively avoid the black walls (look at the Smoothgrad Integrated Gradients implementation in figure 3), some others do not (such as the SmoothGrad Guided Backprop, figure 3),

and we believe that a salient point that targets a wall next to the solution path still provides meaningful information. In any case, the same process was applied to the randomly generated points in order to ensure that there is no bias and that they can act as an objective control group.

Our results shown in 4 are encouraging. The SmoothGrad salience map method is on average much more successful that the vanilla salience map method which performs poorly compared to the random samples. The SmoothGrad with Integrated Gradients [8] salience map successfully intersects the path to the solution about 46.6% of the time, for $k = 3$, a significant fraction compared to the randomized point sampling which intersects the solution about 27.6% of the time. For every value of $k \in \{1, 2, 3\}$ there was a sample size of $n = 100$ auto-generated mazes that were passed through the salience extractor and evaluated against the randomly generated points. The sample size of $n = 100$ is sufficient to show that there is a positive correlation between the saliency points and the solution to the maze.[3]

3.2 Statistical significance

Define X indicator random variable for saliency, that is, $X(m) = \mathbb{1}[m \in S]\ \forall m \in M$ such that M is the set of all points in a maze, and S the set of the most salient points. Similarly define Y indicator random variable for being in the solution path, that is, $Y(m) = \mathbb{1}[m \in L]\ \forall m \in M$, such that L is the set of points that are in the solution path. Let the empirical mean of Y over n samples be $\hat{\mu}_Y$, and the empirical mean of $Y|X = 1$ over n be $\hat{\mu}_{Y|X}$. Assume these empirical means are actually the true means, that is, $\mathbb{E}[Y] = P(Y = 1), \mathbb{E}[Y|X = 1] = P(Y = 1|X = 1)$. Define $P(X = 1) = p_X, p_X = P(X = 1) = \frac{P(X=1|Y=1)P(Y=1)}{P(Y=1|X=1)} = \frac{\hat{\mu}_Y}{\hat{\mu}_{Y|X}} P(X = 1|Y = 1)$. The covariance between the random variable X, Y is given by $Cov(X, Y) = \mathbb{E}[XY] - E[X]E[Y] = P(X = 1, Y = 1) - P(X = 1)P(Y = 1) = (\hat{\mu}_{Y|X} - \hat{\mu}_Y)p_X$. The standard deviations are $\sigma_X^2 = p_X(1 - p_X)$ and $\sigma_Y^2 = \hat{\mu}_Y(1 - \hat{\mu}_Y)$.

Hence, given that for $k = 1, \hat{\mu}_Y = 0.247$ and $\hat{\mu}_{Y|X} = 0.415$, then $Corr(X, Y) = \frac{Cov(X,Y)}{\sigma_X \sigma_Y} = \frac{\hat{\mu}_{Y|X} - \hat{\mu}_Y)p_X}{\sqrt{p_X(1-p_X)}\sqrt{\hat{\mu}_Y(1-\hat{\mu}_Y)}} > 0$. Note that one gets positive covariance whenever $P(Y = 1|X = 1) > P(Y = 1)$ which results in positive correlation between X and Y.[4] Since our data gives us only empirical means we can use concentration bounds (Chernoff and Chebyshev) to obtain regions for $\mathbb{E}[Y], \mathbb{E}[Y|X]$ with high probability. In particular, for an indicator random variable Q with probability $p, \hat{\mu}_Q = \frac{1}{n}\sum_{i=1}^{n} q_i$ and $\mathbb{E}[Q] = p$, then $P(|p - \hat{\mu_Q} \leq \epsilon|) \geq 1 - 2e^{O(\epsilon^2 n^2)}$ which means that we can obtain that $P(Y = 1) = \mathbb{E}[Y]$ can be kept in a region $[\hat{\mu}_Y - \epsilon, \hat{\mu}_Y + \epsilon]$ for small ϵ with high probability. Similarly, $P(Y = 1|X = 1) = \mathbb{E}[Y = 1|X = 1]$ can be kept in a region $[\hat{\mu}_{Y|X} - \epsilon, \hat{\mu}_{Y|X} + \epsilon]$ for small ϵ with high probability.

Using Chebyshev inequality, for $n = 100$ we get $P(|\mu_Y - \hat{\mu_Y} \geq \epsilon|) \leq \frac{Var(Y)}{100\epsilon^2}$. Since we don't have the actual variance, we bound the variance by the maximum possible value, which given that Y is an indicator random variable is $Var(Y) \leq 0.25$. Same logic follows for $Y|X = 1$. Therefore, our mean estimations for both Y and $Y|X = 1$ are accurate to within $\pm 5\%$, at a 95% level of confidence.

4 Conclusions

The results we obtained show a positive statistical correlation between the salience points from SmoothGrad with Integrated gradients [8] and the solution to the maze. Firm conclusions are hard to make though, despite the agreeable benchmarks. A way to explain these results that is in line with our hypothesis, is that classifying an image as a maze entails utilizing in some abstract way the two properties that characterize it: a maze is a complicated network of hedges that has a non-trivial solution. Hence, in classifying a maze as such the CNN may utilize features that capture the underlying structure of a solution regardless of the representation that the CNN has of the solution. Neither the results nor their proposed explanation we provide here are entirely satisfying,

[3] Note that given that the generative model has no upper limit to the amount of randomized maze data it can generate, the system can easily be trained with thousands of samples, just by adjusting a single variable; the only issue with increasing n would be computational time.

[4] The analysis in this part reflects the best performance amongst the saliency map algorithms, i.e. the performance of Smoothgrad Integrated Gradient.

and definitely not conclusive in confirming our hypothesis. Further work needs to be done, namely run much larger sample sizes to evaluate the saliency regions against the solution when the classification for mazes fails. Or instead of using Inception v3 use a lower accuracy classifier and compare the fraction of in-the-path salient points with the fraction found here. Furthermore, more thorough work must be done to provide a more nuanced explanation for the positive correlation observed — or perhaps anchor the current one more elegantly within the literature. Our work, although not conclusive, we believe has exposed some properties of salience maps that were possibly not thought of before and has provided some useful insight into how image classification can expose underlying properties of the image that were previously unknown.

References

[1] Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009.

[2] Shimon Even. *Graph algorithms*. Cambridge University Press, 2011.

[3] Michael C Hughes, Huseyin Melih Elibol, Thomas McCoy, Roy Perlis, and Finale Doshi-Velez. Supervised topic models for clinical interpretability. *arXiv preprint arXiv:1612.01678*, 2016.

[4] Been Kim, Elena Glassman, Brittney Johnson, and Julie Shah. ibcm: Interactive bayesian case model empowering humans via intuitive interaction. 2015.

[5] Neil C Rabinowitz, Frank Perbet, H Francis Song, Chiyuan Zhang, SM Eslami, and Matthew Botvinick. Machine theory of mind. *arXiv preprint arXiv:1802.07740*, 2018.

[6] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

[7] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

[8] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. *arXiv preprint arXiv:1703.01365*, 2017.

[9] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.