A New Method to Visualize Deep Neural Networks

Luisa M. Zintgraf LMZINTGRAF@GMAIL.COM
Informatics Institute, University of Amsterdam

Taco Cohen T.S.COHEN@UVA.NL
Informatics Institute, University of Amsterdam

Max Welling M.WELLING@UVA.NL
Informatics Institute, University of Amsterdam
Canadian Institute for Advanced Research

Abstract

This short paper presents a method for visualizing the response of a deep neural network to a specific input image. Our method highlights areas in the image that provide evidence for or against a certain class. The method provides insight into the decision-making process of deep neural networks, which is important both to improve models and to accelerate the adoption of black-box classifiers in application areas such as medicine. In experiments, we show how the method can be used to gain insight into the classification made by deep convolutional neural networks trained on ImageNet data.

1. Introduction

Deep neural networks (DNNs) have become increasingly powerful in recent years and deliver state-of-the-art performance on difficult classification tasks. To achieve such high performances, these networks have also become much larger and deeper. Consequently, this comes at a price: it is very hard to comprehend how they operate, even if the data is well understood (e.g., images). To date, training a DNN involves a lot of trial-and-error, until a satisfying solution is found. The resulting networks resemble complex non-linear mathematical functions with millions of parameters. This makes it difficult to improve the networks and develop new training algorithms. Moreover, the adoption of black-box methods such as DNNs in industry, government and healthcare is complicated by the fact that their responses are very difficult to understand.

Presented at the Workshop on Visualization for Deep Learning, International Conference on Machine Learning, NY, USA, 2016.

Thus, methods for visualizing the decision-making process and inner workings of DNNs can be of great value for their qualitative assessment. Understanding them better will enable us to find new ways to guide training and improve existing successful networks by detecting weaknesses, as well as accelerating their adoption in society.

This paper builds on a collection of new and intriguing methods for analyzing DNNs through visualization, which has emerged in the literature recently. We present a novel visualization method and exemplify it for deep convolutional networks (DCNNs), which are tailored specifically to image recognition. The method finds and highlights the regions in image space that activate the nodes (hidden and output) in the neural network.

2. Related Work

We base our method on the work of Robnik-Šikonja and Kononenko (2008), who propose a method for explaining predictions of probabilistic classifiers, given a *specific* input instance. The basic idea of their method is that the relevance of a feature x_i can be determined by comparing the prediction $p(c|\mathbf{x})$ to $p(c|\mathbf{x}_{\setminus i})$, where $\mathbf{x}_{\setminus i}$ denotes the set of all input features except x_i. A large prediction difference means that the feature contributed much to the prediction of class c, and vice versa.

To estimate the class probability $p(c|\mathbf{x}_{\setminus i})$ where feature x_i is unknown, it is approximately marginalized out:

$$p(c|\mathbf{x}_{\setminus i}) \approx \sum_{x_i} p(x_i)p(c|\mathbf{x}_{\setminus i}, x_i) \tag{1}$$

(with the sum running over all possible values of x_i). The underlying assumption is $p(x_i|\mathbf{x}_{\setminus i}) \approx p(x_i)$, i.e., that x_i is independent of the other features, $\mathbf{x}_{\setminus i}$. In practice, the prior probability $p(x_i)$ is usually approximated by the empirical distribution for that feature.

The authors evaluate the difference between $p(c|\mathbf{x}_{\setminus i})$ and $p(c|\mathbf{x})$ as the *weight of evidence*, given by

$$\mathrm{WE}_i(c|\mathbf{x}) = \log_2\left(\mathrm{odds}(c|\mathbf{x})\right) - \log_2\left(\mathrm{odds}(c|\mathbf{x}_{\setminus i})\right), \quad (2)$$

where

$$\mathrm{odds}(c|\mathbf{x}) = p(c|\mathbf{x})/(1-p(c|\mathbf{x})) . \quad (3)$$

The resulting relevance vector has positive and negative entries. A positive value means that the corresponding feature has contributed *for* the class of interest, and a negative value indicates that the feature contributed evidence *against* the class.

A similar method (in the sense that the explanation is visualized directly in image space) is proposed by Simonyan et al. (2013). They perform *image-specific class saliency visualization*, where the influence of the input features x_i (typically pixels) on the assigned class c is computed by the partial derivative of the class score S_c with respect to the features, $s_i = \partial S_c/\partial x_i$. The scores S_c are given by the nodes in the fully connected layer before the output layer.

Zhou et al. (2014) generate, for a specific input image, a simplified version of that image that still gets classified correctly, to visualize the regions most important for the classification. They do this by iteratively removing segments of the image and thus keeping as little visual information as possible.

3. Approach

There are two main drawbacks to the method of Robnik-Šikonja and Kononenko (2008) which we address in this paper. We also introduce a way to visualize the role of hidden layers of a DNN using the principles of their method.

3.1. Conditional Sampling

The approximation $p(x_i|\mathbf{x}_{\setminus i}) \approx p(x_i)$ used for equation (1) is not very accurate, especially for image data. To adapt the method for the use with DCNNs, we utilize the following observations about natural images: a pixel's value depends most strongly on pixels in some neighborhood around it (and not so much on pixels far away), and it does not depend on its location in the image (in terms of coordinates). Therefore, we can condition a feature's value x_i on its surrounding pixels, by finding a patch $\hat{\mathbf{x}}_i$ of size $l \times l$ that contains x_i and condition on the remaining pixels in that patch,

$$p(x_i|\mathbf{x}_{\setminus i}) \approx p(x_i|\hat{\mathbf{x}}_{\setminus i}) . \quad (4)$$

We can use the same probabilistic model for all pixels, independent of their location in the image. The advantage of this approximation is that we do not have to model a distribution over the whole feature space, but only on small image patches, which makes it much more feasible.

Algorithm 1 Evaluating the Prediction Difference

input a classifier function, input image $\mathbf{x}$ of size $n \times n$, inner patch size k, outer patch size $l > k$, class of interest c, probabilistic model over patches of size $l \times l$, number of samples S
  initialization: WE = zeros(n*n), counts = zeros(n*n)
  for every patch $\mathbf{x}_w$ of size $k \times k$ **in** $\mathbf{x}$ **do**
    $\mathbf{x}' = \mathrm{copy}(\mathbf{x})$
    $\mathrm{sum}_w = 0$
    define patch $\hat{\mathbf{x}}_w$ of size $l \times l$ that contains $\mathbf{x}_w$
    for $s = 1$ **to** S **do**
      $\mathbf{x}'_w \leftarrow \mathbf{x}_w$ sampled from $p(\mathbf{x}_w|\hat{\mathbf{x}}_w \setminus \mathbf{x}_w)$
      evaluate the classifier to get $p(c|\mathbf{x}')$
      sum_w += $p(c|\mathbf{x}')$
    end for
    $p(c|\mathbf{x}\setminus\mathbf{x}_w) := \mathrm{sum}_w/S$
    WE[coordinates of $\mathbf{x}_w$]
      += $\log_2(\mathrm{odds}(c|\mathbf{x})) - \log_2(\mathrm{odds}(c|\mathbf{x}\setminus\mathbf{x}_w))$
    counts[coordinates of $\mathbf{x}_w$] += 1
  end for
output WE / counts (point-wise)

For a feature to become relevant when using conditional sampling, it has to satisfy two conditions: being relevant to predict the class of interest, and be hard to predict from the neighboring pixels. Relative to the marginal method, we therefore downweight the pixels that can easily be predicted and are thus redundant in this sense.

3.2. Multivariate Analysis

Robnik-Šikonja and Kononenko (2008) take a *univariate* approach: one feature at a time is removed. We would expect that a neural network is robust to just one feature of a high-dimensional input being unknown, like a pixel in an image. Therefore, we will remove several features at once by again make use of our knowledge about images by strategically choosing these feature sets: patches of connected pixels. So instead of going through all individual pixels, we go through all patches of size $k \times k$ in the image, implemented in a sliding windows fashion. The patches are overlapping, so that ultimately an individual pixel's relevance is obtained by taking the average relevance obtained from the different patches it was in.

Algorithm 1 illustrates how this method can be implemented, incorporating the two proposed improvements.

3.3. Visualizing Hidden Layers

When trying to understand neural networks, it is not only interesting to analyze the input-output relation, but also to look at what is going on at the hidden layers. We can adapt the method to see how the units of any layer of the network

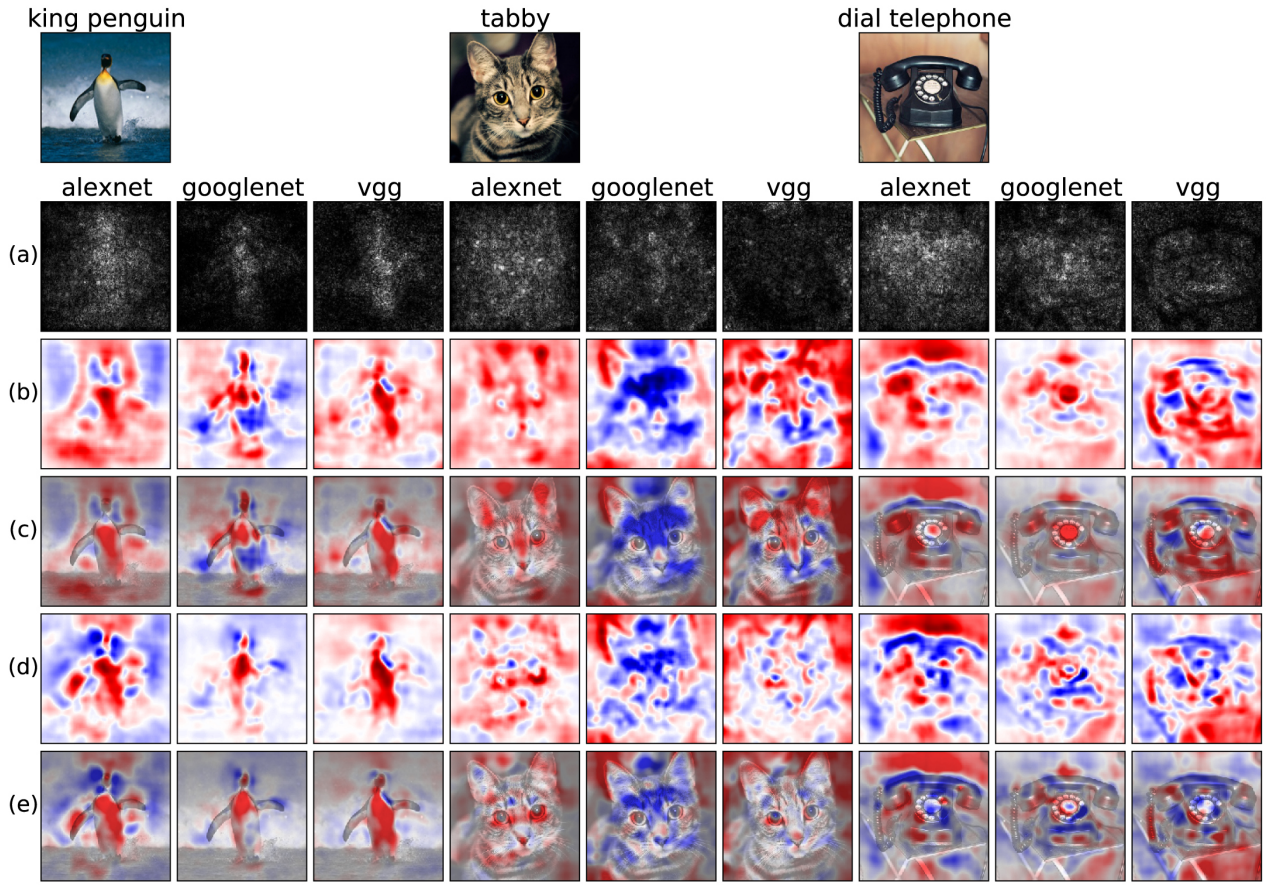

Figure 1. **Visualization of the relevance of input features for the predicted class.** All networks make correct predictions, except AlexNet predicts tiger cat instead of tabby cat. We show the relevance of the input pixels on the highest predicted class (output layer). (a) shows the sensitivity map (Simonyan et al., 2013), (b) the prediction difference with marginal sampling, and (c) the same result overlaid with the input image. (d)+(e) show the results with conditional sampling. We used patch sizes $k = 10$ and $l = 14$ (see alg. 1). For each image, we show the results for the three networks AlexNet, GoogLeNet and VGG net (columns). The **colors** in the visualizations have the following meaning: *red* stands for evidence for the predicted class; *blue* regions are evidence against the class. White/transparent pixels do not have an influence on the decision.

influence a unit from a deeper layer by rewriting equation (1) as an expectation and taking the direct difference in activations instead of equation (2). For feature maps in convolutional layers, an average over the relevance values of all nodes in that map can be shown.

4. Experiments

In this section, we illustrate how the proposed visualization method can be applied, using images from the ILSVRC challenge (Russakovsky et al., 2015) (a large dataset of natural images from 1000 categories), and three DCNNs: the AlexNet (Krizhevsky et al., 2012), the GoogLeNet (Szegedy et al., 2015) and the (16-layer) VGG network (Simonyan & Zisserman, 2014). We used the publicly available pre-trained models that were implemented using the deep learning framework caffe (Jia et al., 2014). Analyzing one image took us 0.5, 1 and 5 hours for the respective classifiers AlexNet, GoogLeNet and VGG (with 4GB GPU memory and using mini-batches).

We compare our method to the sensitivity analysis by Simonyan et al. (2013) (see section 2).

For marginal sampling we use the empirical distribution, i.e., replace a patch with samples taken from other images in the dataset at the same location. For conditional sampling we use a multivariate normal distribution. For both sampling methods we use 20 samples to estimate $p(c|\mathbf{x}_{\setminus i})$.

4.1. Explaining the Classification Outcome

Figure 1 shows visualizations of the spatial support for the highest scoring class, for the three different classifiers, using marginal and conditional sampling. We also compare our method to the sensitivity analysis by Simonyan et al.

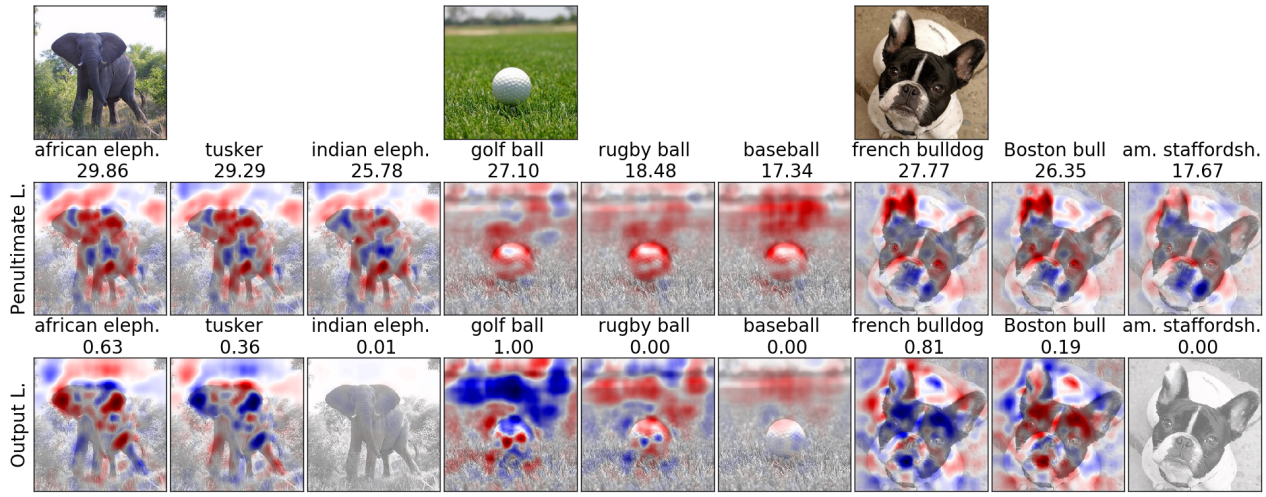

Figure 2. **Visualization of the support for the top-three scoring classes in the penultimate layer and output layer**. The second row shows the results with respect to the penultimate layer; the third row with respect to the output layer. For each image, we additionally report the values of the units (unnormalized activation value and probability, respectively). We used the AlexNet with conditional sampling and patch sizes $k = 10$ and $l = 14$ (see alg. 1). *Red* pixels are evidence for a class, *blue* against it.

(2013). Red areas indicate evidence *for* the class, while blue indicates evidence *against* the class. For example, large parts of the cat's face are blue for the GoogLeNet, while the ear is red with high intensity. This indicates that the classifier does not recognize the face as being indicative of the tabby cat class (but e.g. looks more like another cat class), while the ear appears very distinctive.

One obvious difference to the sensitivity map is that with our method, we have *signed* information about the feature's relevance. (The partial derivatives are of course signed, but this encodes a different kind of information.) We can see that often, the sensitivity analysis highlights the class object in the image. Our method does not necessarily highlight the object itself, but the things that the classifier uses to detect what is in the image, which can also be contextual information.

When comparing marginal and conditional sampling, we see that in general, conditional sampling gives sharper results. For the rest of our experiments, we will use conditional sampling only.

Comparing the visualizations of the three classifiers, we see that the explanations for their decisions differ. For example, we can see that in (d) for the penguin, the VGG network considers the penguin's head as evidence for the class, whereat the AlexNet considers it evidence against the class.

4.2. Pre-Softmax versus Output Layer

If we visualize the influence of the input features on the penultimate (pre-softmax) layer, we show only the evidence for/against this particular class, without taking other classes into consideration. After the softmax operation however, the values of the nodes are all interdependent: a drop in the probability for one class could be due to less evidence for it, or because a different class becomes more likely. Figure 2 compares visualizations for the last two layers. By looking at the top three scoring classes, we can see that the visualizations in the penultimate layer look very similar if the classes are similar (like different dog breeds). When looking at the output layer however, they look rather different. Consider the case of the elephants: the top three classes are different elephant subspecies, and the visualizations of the penultimate layer look similar since every subspecies can be identified by similar characteristics. But in the output layer, we can see how the classifier decides for one of the three types of elephants and against the others: the ears in this case are the crucial difference. Consider further the golf ball example. In the penultimate layer, the evidence for the top three classes concentrates around the ball, and additionally the classifier seems to be looking at the background. In the output layer, we can now observe that the golf ball gets predicted with very high certainty, although large parts of the image are blue, i.e., represents evidence against the class. That the classifier still makes the correct decision could be due to the two intensive red dots on the golf ball: maybe the surface structure of the ball is the deciding factor and outweighs the evidence in favor of the rugby ball.

4.3. Deep Visualization of Hidden Network Layers

Section 3.3 illustrated how our method can be used to understand the role of hidden layers of a DCNN. To get a

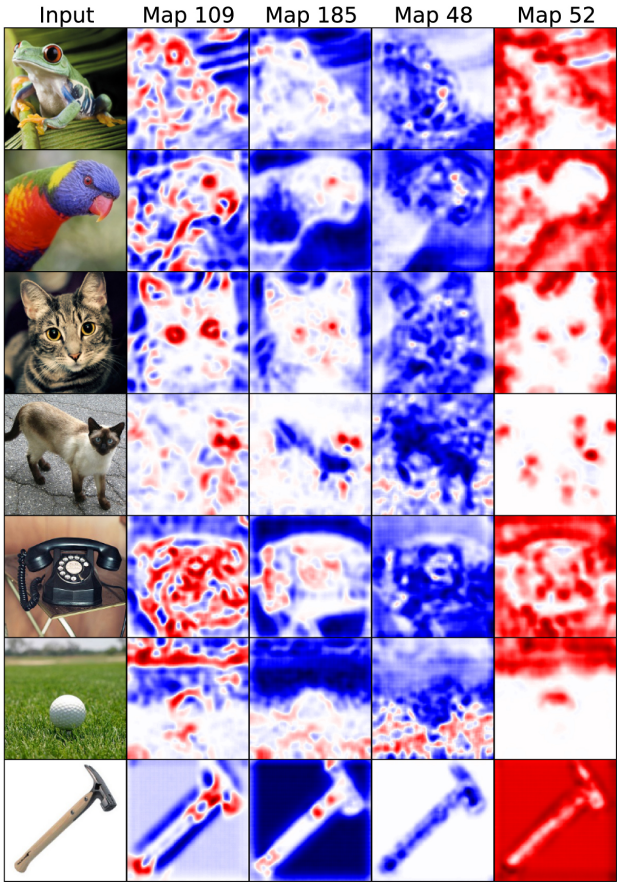

Figure 3. **Visualization of four different feature maps**, taken from the "inception_3a/output" layer of the GoogLeNet (a layer from the middle of the network). Shown is the average relevance of the input features over all activations of the feature map. We used patch sizes $k = 10$ and $l = 14$ (see alg. 1). *Red* pixels activate a unit, *blue* pixels decrease the activation.

sense of what single feature maps in convolutional layers are doing, we can look at their visualization for different input images and search for patterns in their behavior. Figure 3 shows this for four different feature maps from a layer from the middle of the GoogLeNet network. Here, we can directly see which kind of features the model has learned at this stage in the network. For example, one feature map is activated by the eyes of animals (second column), and another is looking mostly at the background (last column).

5. Conclusion

We have presented a new method for visualizing deep neural networks that improves on a method from Robnik-Šikonja and Kononenko (2008), by using a more powerful conditional, multivariate model. The visualization method shows which pixels of a specific input image are evidence for or against a node in the network. Compared to the sensitivity analysis, the signed information offers new insights - for research on the networks, as well as the acceptance and usability in domains like healthcare. However, this information comes at the price of longer computation time. For further reading, we refer to the full paper at https://arxiv.org/abs/1603.02518.

Acknowledgments

This work was supported by AWS in Education Grant award. We thank Facebook and Google.

References

Jia, Yangqing, Shelhamer, Evan, Donahue, Jeff, Karayev, Sergey, Long, Jonathan, Girshick, Ross, Guadarrama, Sergio, and Darrell, Trevor. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Robnik-Šikonja, Marko and Kononenko, Igor. Explaining classifications for individual instances. *Knowledge and Data Engineering, IEEE Transactions on*, 20(5):589–600, 2008.

Russakovsky, Olga, Deng, Jia, Su, Hao, Krause, Jonathan, Satheesh, Sanjeev, Ma, Sean, Huang, Zhiheng, Karpathy, Andrej, Khosla, Aditya, Bernstein, Michael, Berg, Alexander C., and Fei-Fei, Li. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Simonyan, Karen and Zisserman, Andrew. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Simonyan, Karen, Vedaldi, Andrea, and Zisserman, Andrew. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

Szegedy, Christian, Liu, Wei, Jia, Yangqing, Sermanet, Pierre, Reed, Scott, Anguelov, Dragomir, Erhan, Dumitru, Vanhoucke, Vincent, and Rabinovich, Andrew. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9, 2015.

Zhou, Bolei, Khosla, Aditya, Lapedriza, Agata, Oliva, Aude, and Torralba, Antonio. Object detectors emerge in deep scene cnns. *arXiv preprint arXiv:1412.6856*, 2014.