REVIEW, IMPLEMENTATION, AND ANALYSIS OF APPROACHES TO INTERPRET DEEP LEARNING MODELS

Bingqing Song (song0409), Xiang Zhang (zhan6668)

1. INTRODUCTION

In recent years, Artificial Neural Networks (ANNs) have achieved extraordinary performance on many classification and recognition problems. Despite high accuracy in many areas, Artificial Neural Networks suffer from the problem of lacking interpretation and identifying important input features. After thorough investigation, we have noticed that there exist various interpretation methods that have pros and cons in different tasks, which resulted in this review of all the major interpretation approaches for machine learning (especially deep learning) models on which we conducted implementation, comparison, and analysis.

2. METHODS

To understand the black-box Neural Networks, we have done sufficient literature survey and grouped the methods into three classes. Besides visualizing the activation[1][2][3][4], we will illustrate how these methods work, the weakness of each method, and what kind of data it is suitable for.

2.1. Approximation Method

While the black-box model is not easy to understand, it is possible to approximate these Neural Networks structure by some easier models like linear model[5] or decision trees[6]. Here we introduce the methods in the two papers above.

LIME

LIME (Local Interpretable Model-agnostic Explanations) supports explaining individual predictions for text classifiers or classifiers that act on tables or images. It treats any classifier as a black box, conducts a local approximation with an interpretable (linear) model, and derives the contribution of each feature based on certain metric to evaluate the permutation effect.

DeepRED

DeepRED is a decompositional rule extraction method, the idea of which is simple: approximate the NN structure by decision trees. At first step, the famous $C4.5$ algorithm is used to transform each output unit of an NN into a decision tree. The leaves represent the class an example belongs to. Then, intermediate rules are extracted from these derived trees. In this way, for each class it processes every hidden layer in a descending order until it reaches the input layer. In the end, all the decision trees are merged together so that an integrated rule is extracted.

2.2. Backpropogation-based methods

Backpropogation-based method is an important branch of explainable image classification problem. The idea of this line of work is to track information from the output layer back to the input layer. Sometimes only a few layers, say, the last few intermediate layers are powerful enough to explain the classification problem. Since they are slightly modified from backpropagation algorithm, these methods are popular because of high effieciency. Starting from saliency map[7], approaches to visualize and analyze the intermediate gradient have been investigated widely, such as gradient method[7] and SmoothGrad[8]. We introduce several well-known methods below.

Plain gradient-based saliency map[7]

We can utilize a score function which takes input pixels as variables. If we compute the derivative of the function over the input pixels, we will know the 'important' pixels by recognizing the larger gradient. More formally, we denote $S_c(I)$ as the score function of class c, where I is the input picture. Consider the linear score model of class c:

$$S_c(I) = w_c^T I + b_c,$$

where w_c is the weight vector and w_b is the bias, and I represents the vectorized pixel input. Since the score function $S_c(I)$ could be highly nonlinear, it's reasonable to linearize the score function around a certain pixel input I_0. So around I_0, the score function can be approximated as:

$$S_c(I) \approx w^T I + b,$$

where w can be viewed as the the derivative of $S_c(I)$ at I_0. When one component of w is large, it means that the corresponding pixel is important. So visualizing the gradient will lead us to a heat map which indicates the importance of each pixel.

Deconvolution method[9]

The idea of deconvolution method is to derive a deconvnet and visualize the region in the original picture corresponding to the important features. There are two main differences between the traditional CNN and deconvnet. First, since max-pooling is not invertible, we can get an approximate inverse of the max-pooling by looking for the location of the maximum pixel within a certain region in each pooling process. Second, the *ReLU* function is modified. We want to pass the pixels that have positive gradients.

Guided-backpropogation[10]

Since the deconvnet has obvious weakness in capturing features in deeper neural networks, we need to find a better way to replace the max-pooling procedure. In[10], all the max-pooling layers are replaced by convolution layers. So the proposed network only contains convolutional layers and it outperforms the deconvnet in many cases. The only difference between deconvnet and guided-backpropogation in passing gradient backwards is that the latter method restricts each pixel with negative gradient from being passed. The restriction is important because the pixels with negative gradients affect the visualization.

Grad-CAM[11]

Gradient-weighted Class Activation Mapping investigates the gradient fed into the final convolutional layer to produce a localized heat map to show the important regions in the image for predicting a certain concept. First, we need to compute the derivative of score y^c for class c:

$$\alpha_k^c = \overbrace{\frac{1}{Z}\sum_i\sum_j}^{\text{global average pooling}} \underbrace{\frac{\partial y^c}{\partial A_{ij}^k}}_{\text{gradients via backprop}},$$

where A_{ij}^k is the output of the last convolutional layer which represents the last feature map. k is the channel index and i, j is the pixel index in the feature map. So the derived α_k^c represents the average sensitivity of class c towards the k-th channel of the last feature map. Then we compute the weighted average of each channel to derive the importance map of the last layer. Notice that the *ReLU* activation is used to filter the negative pixels.

$$L_{\text{Grad-CAM}}^c = \text{ReLU}\underbrace{\left(\sum_k \alpha_k^c A^k\right)}_{\text{linear combination}}$$

This provides a localization map for a modified image classification architecture, which allows us get a visual explanation of the CNN model.

Excitation backpropogation[12]

The CNN model is made up of neurons of type:

$$x_i = \varphi\left(\sum_j w_{ji}x_j + b_j\right),$$

where w_{ji} is the weight, x_j is the input and φ is the activation function. We call these neurons as $Activation\ neuron$. There are two assumptions about these neurons:

- Neurons are non-negative
- Activation neurons are used to detect visual features. We assume that a larger value of activation neuron implies a higher confidence of detection.

Since activation neurons are connected by weights, we define the connection to be $excitatory$ if the weight is non-negative. We only want to pass the active neurons connected by excitatory weights, so the backpropogation should be modified as:

$$p_j' = \sum_{i=1}^N \frac{w_{ij}^+ x_j}{\sum_{k=1}^N w_{ik}^+ x_k} p_i \quad \text{where} \quad w_{ij}^+ = \max\{0, w_{ij}\},$$

where p_j is the output pseudo-gradient, p_i is the input pseudo-gradient, and N is the number of input gradients. We can rearrange it as:

$$p_j' = x_j \sum_{i=1}^N w_{ij}^+ \hat{p}_i, \quad \text{where} \quad \hat{p}_i = \frac{p_i}{\sum_{k=1}^N w_{ik}^+ x_k},$$

where $\hat{p}_i$ is the normalized pseudo-gradient. We can rewrite this in vector form:

$$p' = x \odot f^*\left(x, w^+, p \oslash f\left(x, w^+\right)\right),$$

where $f^*(x, w, p)$ is the backward function and $\odot$ and $\oslash$ are element-wise multiplication and division respectively.

2.3. Perturbation method

Another line of work focuses on such problem: if the input is perturbed, what is the influence on the output? If changes in some part of the input influence the output greatly, this could be viewed as proof of 'important' features[13][14]. Other works are based on real-time saliency[15] to build a second NN or perturb on the intermediate layer[16]. We introduce two methods below. Unlike all the methods mentioned before, perturbation methods do not rely on other information about the model such as gradients or intermediate layers, so these methods are black-box methods.

Extremal perturbation[17]

Let $\mathbf{x}$ denote the input image and same as before, $S_c(\mathbf{x})$ is the score function with variable $\mathbf{x}$ predicting class c. We want to investigate which part of $\mathbf{x}$ strongly excites the score function, which leads to a larger value of the function.To be specific, we want to find a mask m assigning each pixel u as $m(u) = \{0, 1\}$, where $m(u) = 1$ means the pixel contributes greatly to the output result while $m(u) = 0$ does not. We use the defined mask to build a perturbed image $\hat{\mathbf{x}} = \mathbf{m} \otimes \mathbf{x}$. The aim is to find a sparse m in which only a small part of pixels is preserved, but still induces a large value of score function. The mask we want to find is:

$$\boldsymbol{m}_{\lambda,\beta} = \underset{\boldsymbol{m}}{\text{argmax}} S_c(\boldsymbol{m} \otimes \boldsymbol{x}) - \lambda\|\boldsymbol{m}\|_1 - \beta\mathcal{S}(\boldsymbol{m})$$

The first term intends to find a large score function, the second term encourages to find a sparse mask, and the last term regularize the smoothness of the mask, while λ and β are the penalty parameters. If we want to fix the proportion of pixels preserved,say, constant $a < 1$, our aim is to find the mask that maximizes the score function with only part of pixels. Then the problem becomes:

$$\boldsymbol{m}_a = \underset{\boldsymbol{m}:\|\boldsymbol{m}\|_1 = a|\Omega|, \boldsymbol{m}\in\mathcal{M}}{\text{argmax}} \Phi(\boldsymbol{m} \otimes \boldsymbol{x})$$

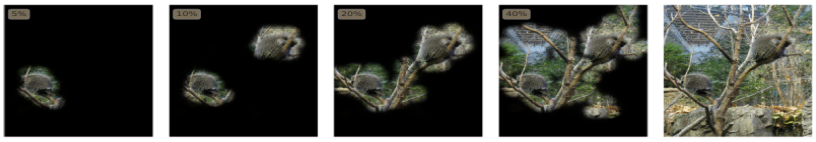


Fig. 1. Extremal perturbation captures features with different size of masks.

RISE[18]

As we indicated before, different from white-box models, in randomized input sampling for explanation of black-box models, we do not have access to any information like gradients or intermediate layers. Also, different from extremal perturbation method, we are sampling over all the pixels in the whole image to evaluate the importance of each pixel. Denote $f : \mathcal{I} \to \mathbb{R}$ as the black-box model, $\mathcal{I}$ is the space of colored image $\mathcal{I} = \left\{I \mid I : \Lambda \to \mathbb{R}^3\right\}$. Denote $M : \Lambda \to \{0, 1\}$ as the random binary mask with distribution $\mathcal{D}$. The score function of the masked image is:

$$f(I \odot M),$$

where $\odot$ is the element-wise multiplication. Then we can define the importance of a pixel as the expected score over all possible masks M conditioned on the event that pixel $\lambda \in \Lambda$ is observed.

$$S_{I,f}(\lambda) = \mathbb{E}_M[f(I \odot M) \mid M(\lambda) = 1]$$

If we sum over all the masks:

$$\begin{aligned} S_{I,f}(\lambda) &= \sum_m f(I \odot m) P[M = m \mid M(\lambda) = 1] \\ &= \frac{1}{P[M(\lambda) = 1]} \sum_m f(I \odot m) P[M = m, M(\lambda) = 1] \end{aligned}$$

There is

$$S_{I,f}(\lambda) = \frac{1}{P[M(\lambda) = 1]} \sum_m f(I \odot m) \cdot m(\lambda) \cdot P[M = m]$$

We can rewrite it in matrix notation:

$$S_{I,f} = \frac{1}{\mathbb{E}[M]} \sum_m f(I \odot m) \cdot m \cdot P[M = m]$$

So we can estimate the score function by Monte-Carlo simulation:

$$S_{I,f}(\lambda) \approx \overset{\text{MC}}{[M]} \cdot N \sum_{i=1}^{N} f\left(I \odot M_i\right) \cdot M_i(\lambda)$$

3. RESULTS AND ANALYSIS

The structure of our project based on the approaches and datasets we investigated is demonstrated by Table 1.

Dataset	Algorithm	Method
student performance	FCN	Linear regression/Decision tree (Gini)/Perturbation
MNIST	FCN/Random Forest	DeepRed/ANNDT/Hypinv/Extremal/LIME
ImageNet/CIFAR10	ANN	Method 2.1/2.2/2.3
drug sensitivity (high-dimension)	FCN	LIME

Table 1. Datasets, algorithms, and interpretation methods that we investigated.

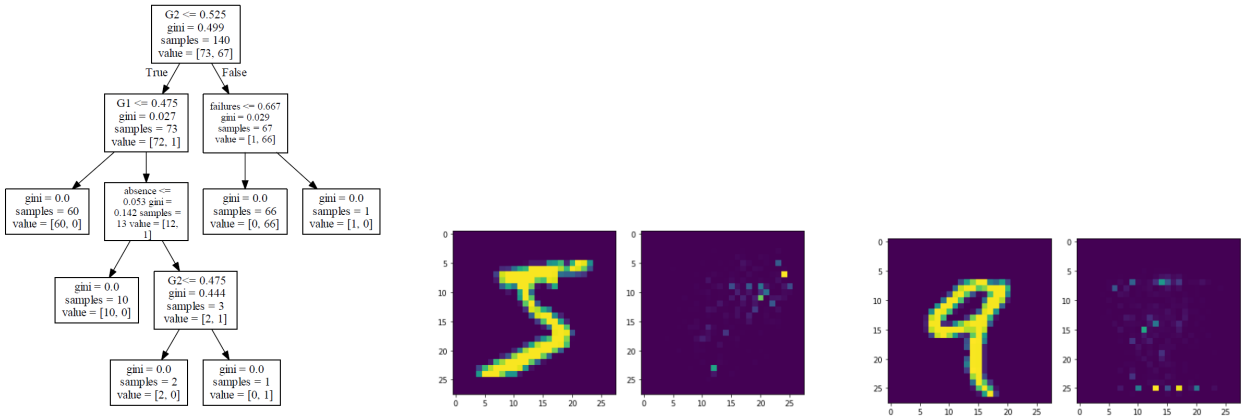
3.1. A toy example

Some simple classification and regresion problems can be solved by both deep learning and some traditional machine learning techniques. We started from a toy example to answer a question: can explainable traditional learning method successfully explain the black-box deep learning? We chose a simple dataset whose samples have 7 features and 1 label each. To illustrate the relationship between the 7 variables and a student's final score, we constructed a simple fully-connected neural network. We tried to explain the model by several interpretable models: (i) Linear regression; (ii) Decision tree; (iii) Perturbation method. All the three methods picked up the same important features and worked well to interpret this toy example: in linear regression, *t-test* significant level of the seventh variable is much higher than the others; in decision tree model, we modified the labels into binary values to indicate whether a student can exceed the average score, and the *gini* index of the seventh variable is much higher than the others; in perturbation method, deleting the seventh variable induced higher mean error.

3.2. Some method can fail for complicated problems

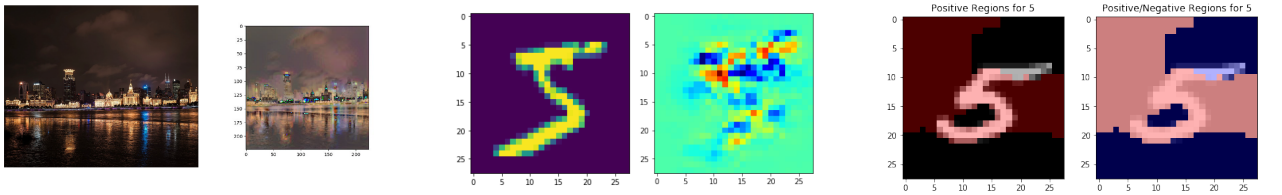
A natural question is: can these methods work for more complicated scenarios, e.g, image classification? Intuitively, linear regression is not suitable for image processing without techniques like localization. We then turned to using *gini* index to know the importance of each pixel by building a decision tree. We visualized the importance of pixels and the captured that important pixels fail to demonstrate the contour of the figure. We concluded that advanced techniques are needed to analyze the image features. The reason decision tree fails in capturing the important pixels is that for different images with the same label, the important pixels are different. Consider the MNIST images with label "5". We can write the number at upper left corner or in the middle of the image, so the important pixel is only identity for one image, but not for all the images with the same label.

Besides these popular methods, some earlier work on explaining image classification has also been investigated. For example, one important branch is visualization of intermediate activations. This line of work tries to answer such a question: given an encoding of an image, to which extent is it possible to reconstruct the image itself? We read two famous paper in this area and reproduced the results, which is much better than the previous *gini* index importance map as shown in Figure 3.



(a) Decision tree by Gini importance index (b) Importance map by Gini index (c) Importance map by Gini index

Fig. 2. Decision trees with Gini index perform poorly on identifying important pixels for classification on MNIST dataset.



(a) Visualize the approximation of inverse of image (b) Importance map by saliency map (c) Importance map by LIME on Random Forest

Fig. 3. Comparison in visualization of intermediate activations and the approximation-based method (LIME)

3.3. Comparison between classical explainable image classification methods

We have reproduced some of the above widely-used methods to demonstrated their capability. In Figure 1 we present multiple figures to visualize the important pixels found by different methods. The three baseline pictures represent three cases: (i) Multiple objects from the same class; (ii) Single object; (iii) Multiple objects from different classes. The expected results of these three cases are: (i) All objects from the class "tabby" are highlighted; (ii) The important features of "penguin" (Head or claw) are highlighted; (iii) Only the target objective "telephone" is highlighted. The network was pre-trained on ImageNet and VGG16 was applied as network architecture. As a consequence, contrastive excitation backpropagation and extremal perturbation perform well while Deconvolution fails to capture the important features in all three cases. In addition, the other methods seem to work for one or two cases. Motivated by these results, we will further investigate why these methods have different performance on the same tasks.

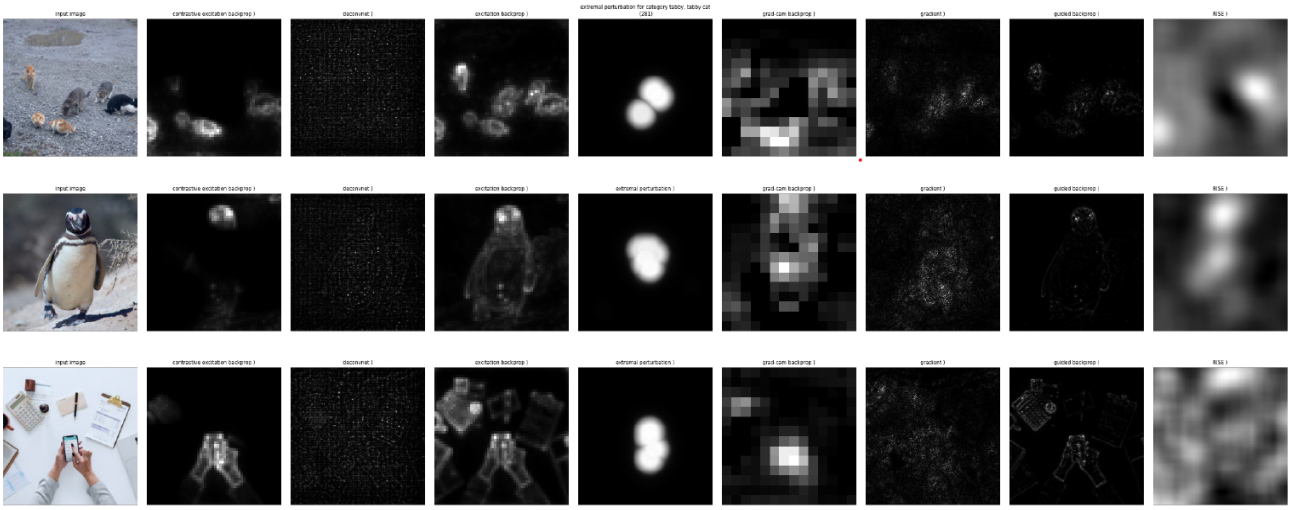


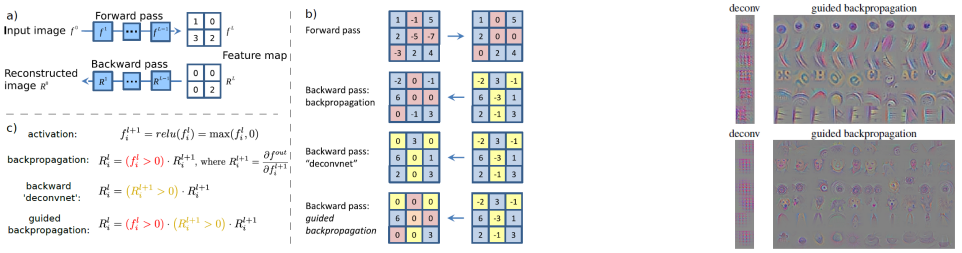
Fig. 4. Implementation and comparison of multiple model interpretation methods on three pictures.

3.3.1. Why Deconvolution always fails?

As we indicated before, Deconvolution method inverses the max-pooling by computing 'switches' – positions of maxima within each pooling region. We record the location of the largest value in a region, but the problem is, for different pictures, location of the index is different. In this way, when we perform the deconvolution procedure, we no longer have the unique identity for each image. In lower, the learned features represent a limited region with limited amount of invariance, but in higher layer, the corresponding region is large and learns more invariance. There is no single image that can maximize these neuron with the 'largest' index. In this way, the features learned by devoncolution net in higher layer are not recognizable.

3.3.2. *Why guided-backpropogation performs better than Deconvolution method?*

To overcome the problem of inversion of max-pooling, the guided-backpropogation method replaces the max-pooling with convolution layers, which avoids the problem in the deconvolution method. Furthermore, we will mask out the negative values both in deconvnet and backpropogation. The negative gradients will decrease the saliency of the pixels in the higher layer which we want to visualize. Without these pixels, the visualization of important pixels will be much clearer.



(a) Difference in passing gradient (b) Difference in captured features

Fig. 5. Comparison between Deconvolution method and guided-backpropogation method in passing gradient and capturing features.

3.3.3. *Why Extremal perturbation performs better than RISE?*

Previously we demonstrated both extremal perturbation and RISE as black-box perturbation methods. In practice, we find out that extremal perturbation outperforms RISE in two images and have similar performance in the second image. The reason is that for extremal perturbation there is a smoothness regularization term. So the mask we got in extremal perturbation has a more regular shape. On the contrary, the important pixels in RISE seem to be 'scattered'. This is due to the fact that every pixel is sampled and assessed randomly in the whole image without any restriction on its 'shape'.

3.3.4. *Why contrastive excitation bacpropogation performs better than excitation backpropogation?*

The difference between the contrastive version and plain excitation backpropogation is that for the contrastive variant data is passed twice into the networks. In this case, for each output unit, we construct the dual unit. For example, the label for the image is "tabby", so we create the label "non-tabby" and train on both labels. Then we subtract the feature map of "non-tabby" from the "tabby" feature map. This will filter out the common activation neurons and only keep the discriminative neurons. This can amplify the true feature of "tabby" in visualization.

3.3.5. *What are the suitable methods for different images?*

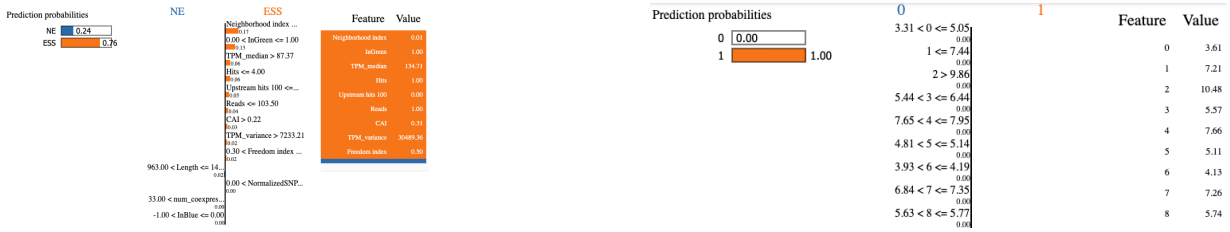
Among three images above, the second one is the easiest for classification and explanation because the target objective is salient and there are no other interference objects. In this case, every method except Deconvolution method works well. For the first image which contains multiple target objectives, we expect that all these objects should be highlighted. Excited backpropogation as well as its contrastive variant, Grad-CAM, plain gradient and guided-backpropogation work well for this case. RISE, extremal perturbation and Deconvolution do not perform well. The reason why extremal perturbation fails is that the limited mask is not large enough to capture all the target objectives, If we increase the face of the mask, it could perform better. But it still shows the weakness of extremal perturbation because we need to decide the mask size manually. For the third image, it is the most difficult case because there are many interference objectives. Only contrastive excitation backpropogation, entremal perturbation and Grad-CAM provide a reasonable explanation for the classification problem. The reason why these three methods perform well is that: contrastive excitation backpropogation subtracts the disturbing background from the original feature map, and thus it successfully filters out the similar objects like laptop and notebook; Grad-CAM works because of the *ReLU* activation in the final layer, which filters out the negative pixels that decrease the saliency of the target object; extremal perturbation works because it restricts the size of the objective, so it only picks up the most salient candidate object. To sum up, the most efficient methods are contrastive backpropogation and Grad-CAM.

Furthermore, we also implemented some approximation methods: LIME and DeepRed. For comparison, we applied multiple ML algorithms with LIME to the MNIST dataset and other selected images. We further verified the usefulness of LIME with another two experiments with tabular data. For a biological dataset with 13 features including both categorical and numerical attributes, powerful machine learning models such as random forest or neural networks are able to make accurate predictions. However, even though we can know the model's feature importance, previously we could not know how exactly each feature contributes to a prediction for a single sample. LIME not only makes it possible, but also provides great visualization as shown in Figure 6(a), which can bridge the gap between a machine learning scientist and an expert from another field. The significance of this can be emphasized again by the other experiment, in which we used neural networks to predict for a high-dimensional drug sensitivity dataset with 46 samples and 18,632 features. The small number of samples and high dimension for features is very likely to cause overfitting for neural networks, which was verified by LIME as shown in Figure 6(b): none of the features was really driving the extremely positive predictions.

4. DISCUSSION AND CONCLUSION

Although we have tried our best to provide theoretical analysis and connect the different approaches with different tasks, many research results from this field lack solid theoretical support. We only have an intuitive understanding about how the objective function of each method works but do not know how the difference in objective functions lead to the tremendous difference in performance, which remains to be a challenge in this field.

In conclusion, we have done a thorough investigation on the major approaches to the interpretation of deep learning models, providing a comprehensive review along with experimental results and analysis. Grouping the methods into three classes, we described how these methods work well or poorly on various tasks and datasets, and explored the suitable jobs for them.



(a) LIME helps interpret each feature's contribution to a single prediction

(b) LIME helps reveal an overfitting case for neural networks

Fig. 6. LIME brings the feature interpretation to the level of a single sample

5. REFERENCES

[1] A. Mahendran and A. Vedaldi, "Understanding deep image representations by inverting them," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 5188–5196.

[2] A. Mordvintsev, N. Pezzotti, L. Schubert, and C. Olah, "Differentiable image parameterizations," *Distill*, vol. 3, no. 7, pp. e12, 2018.

[3] A. Nguyen, A. Dosovitskiy, J. Yosinski, T. Brox, and J. Clune, "Synthesizing the preferred inputs for neurons in neural networks via deep generator networks," *Advances in neural information processing systems*, vol. 29, pp. 3387–3395, 2016.

[4] A. Nguyen, J. Clune, Y. Bengio, A. Dosovitskiy, and J. Yosinski, "Plug & play generative networks: Conditional iterative generation of images in latent space," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4467–4477.

[5] 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.

[6] J. R. Zilke, E. L. Mencía, and F. Janssen, "Deepred–rule extraction from deep neural networks," in *International Conference on Discovery Science*. Springer, 2016, pp. 457–473.

[7] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034*, 2013.

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

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

[10] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," *arXiv preprint arXiv:1412.6806*, 2014.

[11] 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.

[12] J. Zhang, S. A. Bargal, Z. Lin, J. Brandt, X. Shen, and S. Sclaroff, "Top-down neural attention by excitation backprop," *International Journal of Computer Vision*, vol. 126, no. 10, pp. 1084–1102, 2018.

[13] R. C. Fong and A. Vedaldi, "Interpretable explanations of black boxes by meaningful perturbation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3429–3437.

[14] K. K. Singh and Y. J. Lee, "Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization," in *2017 IEEE international conference on computer vision (ICCV)*. IEEE, 2017, pp. 3544–3553.

[15] P. Dabkowski and Y. Gal, "Real time image saliency for black box classifiers," in *Advances in Neural Information Processing Systems*, 2017, pp. 6967–6976.

[16] X. Wang, A. Shrivastava, and A. Gupta, "A-fast-rcnn: Hard positive generation via adversary for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2606–2615.

[17] R. Fong, M. Patrick, and A. Vedaldi, "Understanding deep networks via extremal perturbations and smooth masks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 2950–2958.

[18] V. Petsiuk, A. Das, and K. Saenko, "Rise: Randomized input sampling for explanation of black-box models," *arXiv preprint arXiv:1806.07421*, 2018.