Interpretability in machine learning models by Local Interpretable Model-Agnostic Explanations (LIME)

Bingquan Cai, Chuwei Chen and Xiaowei Ge *
Department of Electrical and Computer Engineering, College of Engineering, Boston University, Boston, MA

Abstract

Machine learning models are widely adopted in many fields, and specialists make important decisions based on the machine learning model's prediction. These decisions could be vital in fields like medical, autonomous driving, etc. Thus, understanding the reasons behind a prediction is necessary to make the prediction trustworthy.

In this project, we applied LIME, an explanation technique that can explain any classifier's predictions with human interpretable features. We applied it to three machine learning models: SVM, random forest, and CNN that take either 1D textual data or 2D image data. Our work follows by evaluations of LIME's results on the three machine learning models. The LIME results we obtained explained how these machine learning models get their prediction, and why sometimes a model gets the wrong prediction. We also compared the LIME result between different models to understand the difference in underlying logic between them.

1 Introduction

After the enormous growth in the machine learning field in the recent years, the "black box" helps solve many difficult problems in human life. When we are benefit from all these convenience, there are confusions and curiosity on how these "black boxes" work. With the growth of the machine learning model complexity, understanding these models becomes more and more difficult. This problem also impedes the usage of machine learning in some fields that hugely rely on human experience and related to human health, Thus, the need of machine learning explanation tool arises.

One related topic of the model interpretation is important feature selection. However, the interpretability is more instance labeled related. For simple evaluation on the working scheme of the machine learning model, the most intuitive method is adding perturbation to the input data and then evaluate the change induced by the perturbation on each input. This method is not time efficient. The two major approaches in model interpretation are output-gradient based (1)(2) and local model approximation (3)(4). With the consideration of model-agnostic and training independence, we used LIME (4) in this project as a generalized machine learning model explanation tools.

Table 1: Summary of the properties of different model explanation tools(5)

Methods	**Training**	Efficiency	Additive	Model-agnostic
Parzen(1)	Yes	High	Yes	Yes
Salient map (2)	No	High	Yes	No
LRP (3)	No	High	Yes	No
LIME (4)	No	Low	Yes	Yes

*Chuwei Chen contributed to SVM-LIME part; Bingquan Cai contributed to the Random Forest-LIME part; Xiaowei Ge contributed to the CNN-LIME part. All participated in the concept and model establishment.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

2 Methods

2.1 What is LIME

LIME is the acronym for Local Interpretable Model-agnostic Explanations. It is a technique that approximates any black-box machine learning model with a local and interpretable model to explain each individual prediction. Each letter of LIME represents what we seek in explanations. "Local" refers to local fidelity, which means that we want the explanation to accurately reflect the classifier's behavior around the instance being predicted. "Interpretable" means we want human interpretable explanations, such that even people who don't know machine learning can understand. "Model-agnostic" means we treat the machine learning model as a black box and LIME is able to explain it without needing to dig into it. That also being said, LIME is universally applicable to any machine learning classifiers.

2.2 Idea of LIME

Suppose we have a data set which contains red points and blue points. We train a black box algorithm on this data set. After the training, the decision boundary of the complex model could be non-linear (Figure 1). For a new data point, if it falls into the red area, we predict it with the red class, otherwise we predict it with blue class. So, the prediction we make is highly non-linear or in other terms there is no easy to explain relationship in how we perform a prediction. The model just learns some complex patterns as a combination of those two features. Now if we make a prediction for instance, it is belongs to red class, how could we explain that why our model outputs the red class. We cannot easily summarize the whole decision boundary into one explanation.

So, the basic idea of LIME is that we just zoom into the local area of the individual prediction, there we can easily create a simple explanation the makes sense in that local region. This way we do not have to worry about the rest of the model and still get a valid explanation why the prediction was made. For instance, in this example (Figure 1), we could say that the prediction was made simply because its value is big enough to fall on the upper side of the boundary line. To get such kind of explanations, LIME simply fits a linear interpretable model in that local area which is often also called surrogate. It is basically a local approximation of our complex model in this local area. So that is the basic idea and now let us have a look at the mathematical optimization problem used in LIME.

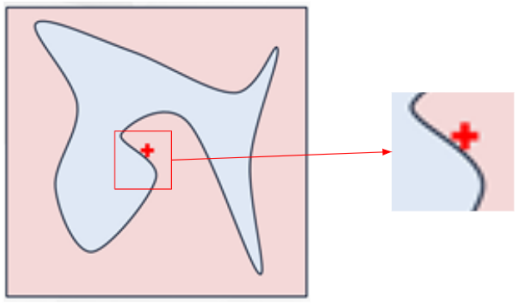

Figure 1: Idea of LIME

The idea behind the mathematical optimization problem is quite intuitive (Equation 1). As already mentioned, we want to create a local approximation of our complex model for a specific input denoted with x. And here the complex model is denoted with f and the simple local model is denoted with g. This simple model small g comes from a set of interpretable models which is denoted with a capital G. Here capital G is a family of linear models such as linear regression. Now the first loss term in our optimization function simply means that we look for an approximation of the complex model f by the simple model g in the neighborhood of our data point x. It means we want to get a good approximation in that local neighborhood. The third argument pi here defines the local neighborhoods of that data and is some sort of proximity measure. The second loss term is used regularize the complexity our simple local model. So, this loss function says that we look for a simple model g that minimize those two loss terms so it should approximate the complex model in that local area and additionally stay as simple as possible.

$$\xi(x) = \underset{g \in G}{\operatorname{argmin}} \ \mathcal{L}(f, g, \pi_x) + \Omega(g) \tag{1}$$

$$\mathcal{L}(f, g, \pi_x) = \sum_{z, z' \in \mathcal{Z}} \pi_x(z)(f(z) - g(z'))^2 \tag{2}$$

2.3 How LIME works

In this section, we will explain how LIME works step by step (Figure 2).

Back to the example, given a new data point input, in the first step we simply generate some new data points in the neighborhood of our input data point. These data points are generated by perturbations, and this can be achieved by sampling from a normal distribution.

Then we get the prediction for these data points using our complex model f. What we have now is a new data set we can use to fit a classifier. So we minimize the first loss term by simply getting the highest accuracy on that new data set using a simple linear model. In the LIME paper they use this loss function (Equation 2) for optimizing a linear model. It's basically the sum of the squared distances between the label comes from the complex model and the prediction of the simple model g which looks like the square loss that we covered in class. But additionally, the proximity pi is added to weight the loss according to how close a data point is. The points that are close to our input data points are weighted the most. That is how we ensure that the model is locally faithful.

And for the omega term, we said we use it to make sure that our model stays simple. In LIME paper a sparse linear model is used which aims to produce as many zero weights as possible. In practice, this can be achieved by using a regularization technique such as lasso linear regression.

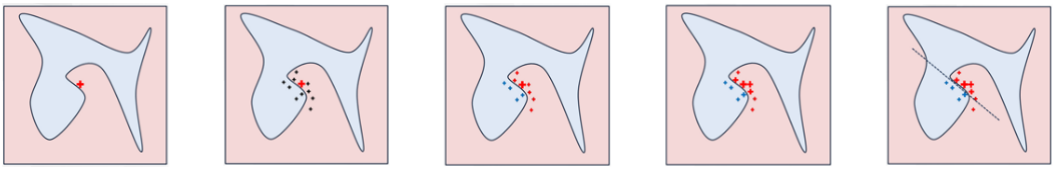

Figure 2: LIME working process

3 Experiments and Results

3.1 Explainable SVM by LIME

"Support Vector Machine" (SVM) is a supervised machine learning algorithm that is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is number of features) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes.

In this section, we implement LIME on a SVM classifier with linear kernel to understand why SVM is making its decisions. The SVM classifier is fed with 1D textual data input. We trained the SVM to classify human comments from the internet as either normal or offensive. We set the distribution of the two classes' data to be entirely balanced to avoid any biased prediction. The SVM classifier that we trained has an accuracy of 88.32%. We then evaluated the LIME result on this relatively accurate model to discover the logic behind its correct or wrong prediction. The LIME results are shown in the "Result with LIME" section below.

3.1.1 Result with LIME

LIME interpretation of a correctly classified prediction Figure 4 below shows an example of the LIME result when our SVM classifier correctly classifies an offensive comment. LIME provides us a visualization of the reasons that the SVM believes it is an offensive comment. As the local

explanations shown, which is located in the middle of figure 3, the word "stupid" has the greatest prediction probability of the class "offensive". Other words such as "unfair", "the", have prediction probability to the "normal" class instead. By summing the prediction probability of each class's features, the "offensive" class has a higher overall prediction probability of 0.76 which makes SVM concludes that the comment is offensive. We also have an intuitive view by highlighting each word's prediction probability in the text, as shown on the right-hand side of figure 3.

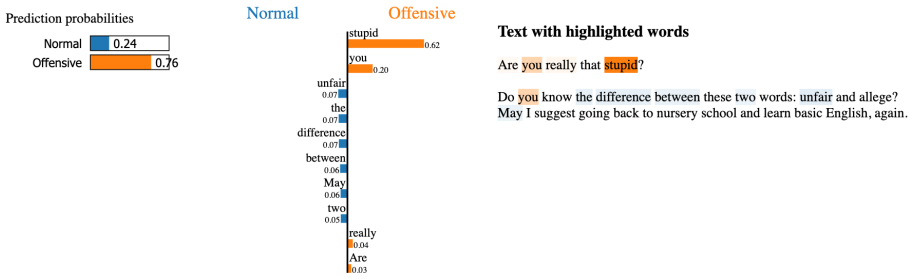

Figure 3: Interpretation of a correct prediction made by SVM

LIME interpretation of a wrongly classified prediction As we know, machine learning models make mistakes, even if the model has superb accuracy. Here, we used LIME to explore the reasons that a highly accurate SVM makes wrong prediction. In figure 4, we present the LIME result when the SVM wrongly predicts a normal comment as an offensive one. Surprisingly, we found that the word "You", a word with neutral meaning, has the greatest prediction probability to "offensive" class, and that is the leading factor that the SVM predicts the comment as "offensive".

The reason that the word "You" has high offensive attributes in the SVM is probably because it has a high occurrences of being used in combination with offensive words which makes it usually appears in the offensive comments. This example explains why we should not easily trust a machine learning model's prediction, even if it is highly accurate. We must understand the explanations that the model makes its prediction using model explainer such as LIME before we decide whether we should trust it.

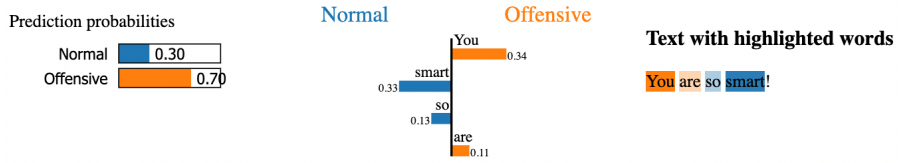

Figure 4: Interpretation of a correct prediction made by SVM

3.2 Explainable Random Forest by LIME

Random forest is a powerful machine learning method which is widely used. It is a supervised learning algorithm, which is an integrated learning algorithm with decision tree as the base learner. It is inherently more accurate than most individual algorithms due to the use of integrated algorithms. However, it might be difficult to be interpret for it is a black box algorithm.

In this section, we will apply LIME on the random forest model to show us how LIME interpret the random forest model. The random forest model is training on the given MNIST data set.

3.2.1 Random forest

Introduction Random Forests grows many decision trees (Figure 5). It first selects some random samples from a given data set. Then it will construct a decision tree for each sample and get prediction from each decision tree. Later it votes among each predicted result and make the final prediction based on the voting result.

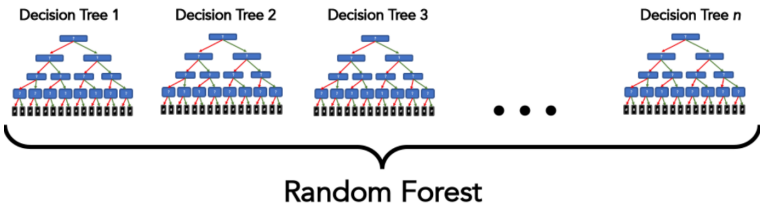

Figure 5: Random forest structure

Feature importance evaluation In reality, there are often hundreds or thousands of features in a data set, and we are concerned about how to select the features that have the greatest impact than the results, so as to reduce the number of features when building the model. GINI index is the one often used in random forest to measure the feature importance. Those digital data might be helpful for the random forest model to make a good prediction, but it might be difficult for us human being to fully understand how it works.

3.2.2 LIME interpretation

Random forest is an integrated algorithm consisting of decision trees which is difficult for us to understand the internal logic. However, LIME will help us to interpret the complex random forest model.

Interpretation of a correct prediction made by random forest model Take the number 4 in the MNIST data set as an example (Figure 6). The actual number for the image is 4, and the prediction made by random forest is also 4. Here, random forest model make a correct prediction on the image of number 4. Then, we apply LIME on that particular input image to see why the random forest model makes such a prediction. We can see that LIME shows the positive region, which is the red area on the image, for each number. Among all the numbers, LIME captures the feature of number 4 the most while there is only a partial positive region or even no positive region on other numbers. We can think of LIME as a way of visualization for the voting and selecting process in the random forest model. It helps us to understand why the random forest model predicts the image as number 4.

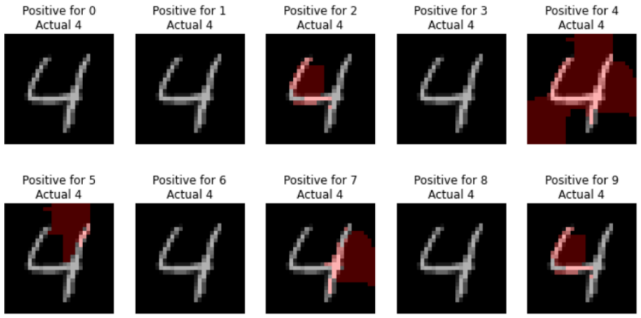

Figure 6: Interpretation of a correct prediction made by random forest

Interpretation of a wrong prediction made by random forest model On the other hand, LIME also helps us to understand why the random forest model makes a wrong prediction. It is really meaningful for us to know a wrong process inside the random forest model. Take the number 7 in the MNIST data set as an example (Figure 7). We apply LIME on the input image as what we did above. The actual number of the image is 7 and the random forest model gives a wrong prediction which is number 9. By looking into the positive region of each number provided by LIME, we can see that there is a little curve which at the top left of the number 7 which makes the number 7 look like the number nine, and the positive region of number 9 does cover that little curve. That might be the reason why the random forest model makes a wrong prediction.

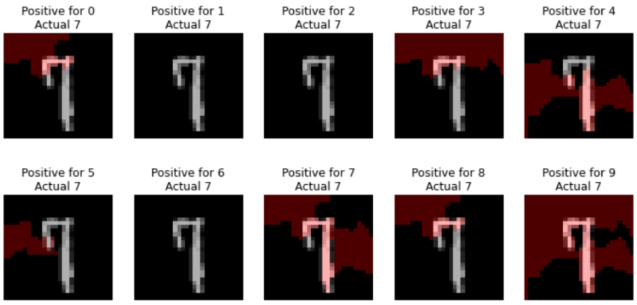

Figure 7: Interpretation of a wrong prediction made by random forest

In summary, random forest model is a great machine learning method while the logical meaning inside is hard for us human being to understand. Lime helps us to understand the reason why random forest model makes such prediction.

3.3 Explainable CNN by LIME

Convolutional neural network (CNN) is one of the most powerful machine learning tool. However, its ability to learn the difficult tasks is based on its complexity and thus, it is hard to explain its result, which involves hundreds to billions of parameters.

In this section, we will apply LIME on two CNN models with different levels of complexity to help us understand how CNN is making decisions. The first CNN model is build for gray scale MNIST data set, which also provides comparison to the random forest model in last section. The second CNN model is a classical and more complex model for ImageNet data set.

3.3.1 CNN explained by LIME on MNIST data set

CNN architecture To further validate the model agnostic property of LIME and its performance on models with different complexity, we build a simple CNN model based on LeNet (6) and fine tuned on our MNIST data set. Basically, it contains two convolutional layer and two fully connected layer. We fine tuned the two filter sizes and filter numbers. Besides, we added some dropout layers which has a function of prevent overfitting. However, as MNIST is not a difficult classification problems, we achieved over 98% precision and the modifications does not boost the precision a lot. Thus we choose the architecture shown in Figure 8 with an accuracy of 98.34% on the MNIST test set. The whole network is implemented with pytorch.

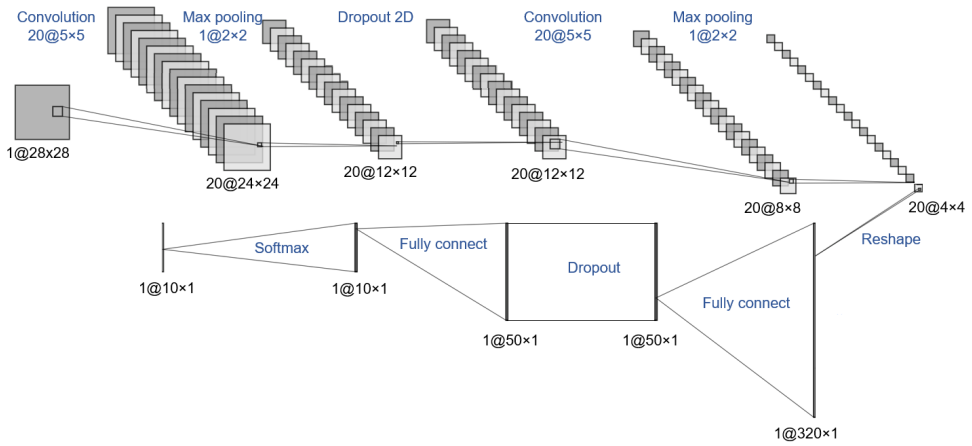

Figure 8: The diagram of CNN structure for MNIST data set

LIME interpretation After CNN trained on MNIST data set as the machine learning prediction model, we applied LIME to explain its decision on MNIST dataset. For human, it is intuitive that we see the digits then we know what number it is. And here LIME will tell us how CNN is looking at in the MNIST dataset. For the ten digits, we can see the clear boundary profiled the digits really well, and the support decision areas covers the digits body (Figure9), thus our CNN makes decision by recognizing the digits, instead of making decision based on information from nowhere. Although some of the blank areas also included, which maybe as the reference to the digits to create the contrast, and also the blank area are related to the decision regions.

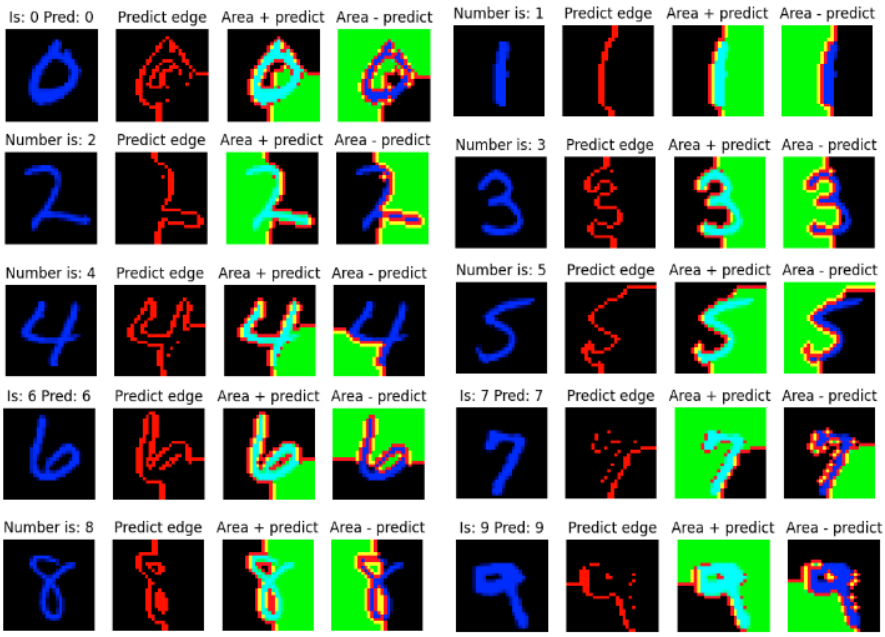

Figure 9: LIME explanation of CNN decision on MNIST data set

LIME results comparison on MNIST with different machine learning models For the better comparison for the two methods we applied on MNIST dataset, we can see how LIME is performance on different model with different complexity. As CNN is a much more complex model compared to random forest, which provides higher precision, we can also see the support area predicted is more close to the exact digits boundary (Figure10).

Overall, LIME is applicable on various machine learning models, in helping us understand how they make decisions.

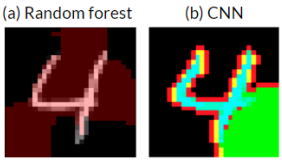

Figure 10: LIME explanation of CNN decision on MNIST data set

3.3.2 CNN explained by LIME on ImageNet data set

Except for the classical benchmark MNIST dataset, we also applied LIME with more complex CNN model on ImageNet dataset. In here, we use a pretrained light-weight model, Inception V3 (7), an upgraded version based on GoogleNet, which utilize the mini-branch of heterogeneous architecture and unsymmetric convolution kernel. We test on two types of images, in which only contains one goal and another contains both goals.

In the water ouzel (Figure11(a)), LIME helps interpret the ouzel body as the decision region. In the image contains both dog and cat (Figure11(b)), as the prediction is Bernese mountain dog, the dog head is taken as the supportive region while the cat head is the recognized as the unsupported region. Thus, LIME also works for this even more advanced models and tasks.

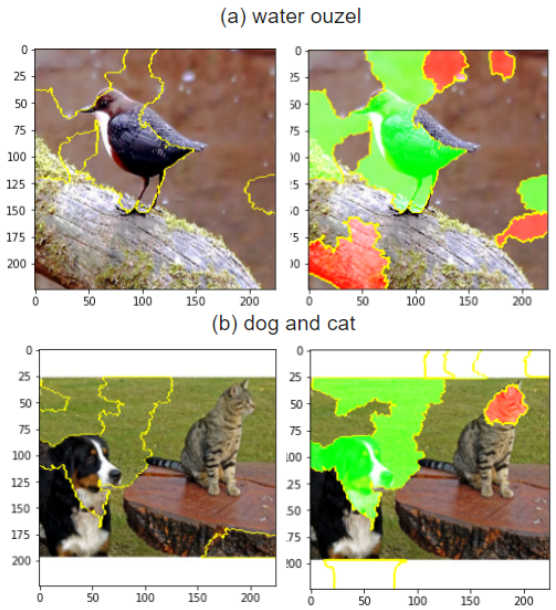

Figure 11: LIME explanation of Inception V3 decision on images in ImageNet

4 Conclusions and Discussions

In this project, we established three machine learning models, SVM, random forest, and CNN on 1D textual data (Comments) or 2D image data (MNIST, ImageNet) with high classification accuracy. Then, we focused on evaluating LIME on the interpretation of the three machine learning models classification results. During the process, we practiced how to build various types of machine learning models we learned in class, while with the inspiration of LIME, we understand the machine learning models characteristics better.

During the implementation of LIME, we also noticed some problems and limitations of this method. As LIME uses simple model to approximate the local prediction property of complex model, which limit itself to the linearity inversely. Meanwhile, LIME's interpretation is hard to evaluate without the intuition from human, especially when the data dimension is high.

In the next step, we would pursue use LIME to guide real world questions, like questions in medical fields. It would be also valuable to explore more methods to have better approximation to the machine learning models. Besides, establish a evaluation scheme for LIME would help the application of this method.

5 Github Link

`https://github.com/chenchuw/CS542-FinalProject.git`

References

[1] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K.-R. Müller, "How to explain individual classification decisions," *The Journal of Machine Learning Research*, vol. 11, pp. 1803–1831, 2010.

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

[3] 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, p. e0130140, 2015.

[4] 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*, pp. 1135–1144, 2016.

[5] J. Chen, L. Song, M. Wainwright, and M. Jordan, "Learning to explain: An information-theoretic perspective on model interpretation," in *International Conference on Machine Learning*, pp. 883–892, PMLR, 2018.

[6] 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.

[7] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.