How to Explain Neural Networks: an Approximation Perspective

Hangcheng Dong[1], Bingguo Liu[1], Fupeng Wei[2], Fengdong Chen[1], Dong Ye[1], and Guodong Liu[1*]

[1]Harbin Institute of Technology, Harbin, China

[2]North China University of Water Resources and Electric Power, Zhengzhou, China

{hunsen_d, lgd}@hit.edu.cn

Abstract

The lack of interpretability has hindered the large-scale adoption of AI technologies. However, the fundamental idea of interpretability, as well as how to put it into practice, remains unclear. We provide notions of interpretability based on approximation theory in this study. We first implement this approximation interpretation on a specific model (fully connected neural network) and then propose to use MLP as a universal interpreter to explain arbitrary black-box models. Extensive experiments demonstrate the effectiveness of our approach.

Index Terms: Interpretability, Deep Learning, Decision Boundary, Knowledge Distillation

1. Introduction

The interpretability of deep learning plays a key role in many mission-critical applications [1]. In the past years, interpretability has been an active research field of deep learning. Currently, studies on interpretability [2–7] can be divided into two categories: post-hoc explanation and ad-hoc explainable model prototyping [8–11]. Post-hoc explanation methods attempt to offer an explanation for a model when the model is trained, while the ad-hoc explanation methods prototype a more interpretable model from the beginning. Despite that these studies have accomplished great successes, we argue that the fundamental question on interpretability remain unanswered, i.e., what kind of interpretability do we want? what kind of interpretability models can succeed? and how to realize them?

To answer the above questions, we must figure out the desirable characters we want an explanation to have. We think that an explanation shall fulfill **completeness** [12, 13], i.e., the explanation should be faithful to the origin model not locally but in the entire domain. Unfortunately, the majority of existing approaches do not respect the completeness when generating an explanation well. For example, gradient-based saliency maps are usually incomplete since they are essentially based on changes of model out-
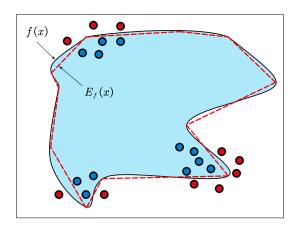

Figure 1. Explain black box models by the approximation mechanism. $f(x)$ is a black-box model, and $E_f(x)$ is a white-box model.

puts without taking the model bias into account. Other than this, we think that the major difficulty of gaining completeness is because too many parameters are interacted complicatedly in a black-box model, rendering it hard to analyze the influence of each parameter on the output of the model.

Our motive is to develop an interpretability method to accomplish completeness. Recently, some researches [13–19] have made huge strides in interpretability by reducing a fully-connected ReLU network into a piecewise linear function over polytopes. Given such a ReLU network, for any input, we can identify the function over its pertaining polytope, thereby favorably providing a complete explanation. Since a fully-connected ReLU network has a complete explanation in the light of piecewise linear functions, our curiosity is can we use a fully-connected ReLU network to explain other black-box neural networks? Specifically, can we distill other black-box neural networks into a fully-connected ReLU network to gain interpretability?

Such an idea is desirable due to the fact that, to the best of our knowledge, there is still a vacuum in global completeness of interpretation methods. We consider interpretability to be an approximation problem. Simultaneously, the approximation procedure is necessary to ensure complete-

ness (refer Fig.1). Therefore, we present the concept of CLIME (complete local interpretable model-agnostic explanations), which is the global formulation of LIME [20]. In this research, we first focused at how to understand any fully-connected neural network (FCNN), and then we introduced an idea of distillation-based CLIME to explain arbitrary black-box models. Our study made the following significant contributions:

(1) We presented the definition and properties of an explanation, which is called CLIME. Furthermore, we show that in the classification problem, CLIME is equivalent to the decision boundary of black-box models.

(2) We explained FCNN by finding the piecewise linear approximation, which is an extended version of the technique in [18]. It is worth to point out that a similar method named LANN (linear approximation neural network) has been proposed in [21]. The main difference is the approximation method of activation functions. Moreover, our goal is to compute approximation decision boundaries, whereas theirs is to measure the model complexity.

(3) We proposed DecisionNet (DNet), a wide network for binary classification, as a white-box representation of PLNN. This transformation allows the parameters of PLNN to be decoupled, which meets our definition of explainable models.

(4) We proposed to explain other black-box models using PLNNs (FCNNs), i.e., a preliminary implementation of CLIME based on knowledge distillation.

2. Related Works

Feature Visualization is a straightforward method for interpretability [22], whose idea is to generate saliency maps depicting the concerned parameters. Observing activation maps [23] is one of the most intuitive notions. However, this strategy has shown limited results. Exploration of the link between activation maps and the input space, also known as activation maximization [24] and deconvolution [25], is another consideration. Both of them have demonstrated that as the number of layers in a well-trained network grows, the semantic information on which neurons are focused becomes more richer. A number of gradient-based approaches [12, 26–32] have been presented to quantify the relevance of input characteristics, and perturbation-based [33–35] methods have also attracted a lot of attention. These technologies have shown that neural networks can learn semantic information, but they are susceptible and sensitive to small changes in the input space [36]. CAM (Class Activation Mapping) [37] and its derivatives [38–40] are other well-known visualization approaches. Heatmaps are created using linear combinations of activation maps from convolution layers in CAM-like techniques. However, as previously discussed, feature visualization approaches are not complete.

Proxy Method tries to train simple models based on the black-box model. Hinton et al. demonstrated how knowledge distillation might increase generalization performance by leveraging the output of a big neural network, known as soft targets, to train another model (such as a tiny neural network or decision trees). Due to a shortage of efficient explainable student models, model distillation is frequently employed to improve the performance of them [41–44]. LIME [20] based on local approximation are presented to solve the problems of global proxy. LIME retrains a linear classifier to act as a local proxy for the origin model by sampling in the neighborhood of the sample to be explained. On the other hand, LIME can only understand models locally and is difficult to execute on data with high dimensions. We propose to use PLNN as an explainable model that enables the implementation of CLIME, which can combine the benefits of global and local proxy methods.

ReLU Network is a piecewise linear function, which divide the input space into several linear regions [15, 16, 18]. The upper bound on the number of the linear regions has been analyzed in [15, 16, 19]. The influence of BN and dropout strategies on the linear interval of ReLU networks was also observed in the literature [17]. Hu [21] el at. proposed linear approximation neural network by using piecewise linear functions instead of curve activation functions. Inspired by these studies, we believe that the piecewise linear function satisfies both completeness and explicitness, making it a quiet perfect explainable model.

3. Definitions of Interpretability

In this section, we discuss the following questions: 1) What is the explainable model? 2) What is interpretability? 3) How to explain black-box models?

We contend that there are two levels of interpretability: the first is the explanation of external appearances, and the second is the comprehension of inner mechanisms. We believe that because a ReLU network is a piecewise linear function, it is hard to describe the intrinsic mechanism of a neural network, as the intrinsic mechanism most likely does not exist. As a result, the explainable model is defined as follows:

Definition 1 (Explainable Model). *If a model $E_f(\mathbf{x}; \boldsymbol{\theta})$ is explainable, implies that we are able to know the **global** impact of each parameter $\boldsymbol{\theta}_i$ on the output, which termed explicitness.*

We would like to to underline that the derivative itself is insufficient to indicate the global influence of a parameter. Kernel SVM [45], ensemble Learning [46], and deep learning [47] are black-box, while (piecewise) linear models are white-box, according to Definition 1. Based on the explainable model, we further define interpretability according to approximation theory.

Definition 2 (Model Interpretation). *For a given error*

$\delta > 0$ and model $f(\cdot)$, if there exist an explainable model $E_f(\cdot)$, we say $E_f(\cdot)$ is an interpretation of $f(\cdot)$, implying that it satisfies the following conditions:

$$\forall \mathbf{x} \in \chi \subseteq \mathbb{R}^d, |f(\mathbf{x}) - E_f(\mathbf{x})| < \delta, \quad (1)$$

where χ is the input space, $\mathbf{x}$ and d are an instance and the dimension of input features respectively.

We define interpretation as an approximation process employing a specific white-box model, which implies that the precision that an interpretation may attain is heavily determined by the white-box model utilized. The white-box model that fits Definition 2 is referred to as the complete local interpretable model-agnostic explanation (**CLIME**). However, because achieving a model-agnostic explanation is challenging, we will first discuss implementing in a specific model, namely the FCNN. Following that, we will explore deployment in any model. In this work, they are all referred to as CLIME without distinction.

4. Interpretability of Classification Models

In this section, we introduce the relationship between CLIME and the dicision boundary corresponding to the classification model. The decision boundary of a classifier is defined as follows.

Definition 3 (Decision Boundary). *Given a classifier $F(\mathbf{x}) = argmax_i(f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_i(\mathbf{x}), ..., f_d(\mathbf{x}))$ where $f : \mathbb{R}^d \to \mathbb{R}^c$ and $f_i(\mathbf{x})$ is the i-th value of $f(\mathbf{x})$. Denote by $C = \{C_i | i = 1, 2, ..., n\}$ the labels. The decision boundary between class C_i and $C_j (i \neq j)$ is:*

$$DB_{ij} = \{\mathbf{x} | \forall \delta > 0, \exists \mathbf{x}_i, \mathbf{x}_j \in U(\mathbf{x}, \delta), f(\mathbf{x}_i) = f(\mathbf{x}_j)\}, \quad (2)$$

where $U(\mathbf{x}, \delta)$ means a open region satisfying $U(\mathbf{x}, \delta) = \{z | dist(\mathbf{x}, z) < \delta\}$, $dist(\cdot, \cdot)$ is the distance metric.

Next, we extend Definition 2 to the classification problem.

Definition 4 (Classifier Interpretation). *For a given classier $F(\cdot)$, if there exist an explainable model $E_F(\cdot)$, we say $E_F(\cdot)$ is a classifier interpretation of $F(\cdot)$, implying that it satisfies the following conditions:*

$$\forall x \in \chi \subseteq \mathbb{R}^d, F(x) = E_F(x). \quad (3)$$

Proposition 1. *When the error δ defined in Definition 2 is small enough, we have that Definition 2 is a sufficient condition for Definition 4.*

Proof. Let the classifier be $F(\mathbf{x})$, and the i-th output of logit $f_i(\mathbf{x})$ be the maximum, namely $f_i - f_j > 0, (\forall j \in k | 1, 2, ..., n, k \neq i)$. Denote by $E_F(\mathbf{x})$ the explanation. Notice the completeness by Definition 1 of $E_f(\cdot)$, we have:

$$\lim_{\delta \to 0}(E_{f_i} - E_{f_j}) =$$
$$\lim_{\delta \to 0}[(E_{f_i} - f_i) + (f_i - f_j) + (f_j - E_{f_j})] = f_i - f_j > 0. \quad (4)$$

Thus, $\exists \delta > 0, \mathbf{x} \in U(\mathbf{x}, \delta)$, we have $E_{f_i}(\mathbf{x}) - E_{f_j}(\mathbf{x}) > 0, (\forall j \in k | 1, 2, ..., n, k \neq i)$, namely, $E_F(x) = F(x)$, which concludes the proof. $\square$

In consequence we still call the classifier interpretation as CLIME. At last, we would like to show the relationship between CLIME and the corresponding decision boundary in the scenario of a classification problem.

Theorem 1. *For a classifier, the classifier interpretation (i.e. CLIME) is equivalent to the corresponding decision boundary.*

Proof. In a classification problem, the necessity is that, for a trained classifier $F : \mathbb{R}^d \to \mathbb{R}$, given the decision boundary, the input space $\chi \subseteq \mathbb{R}^d$ can be divided into several maximum decision regions $R_i = \{\mathbf{x} | F(\mathbf{x}) = i(i \in 1, 2, ..., n)\}$, then CLIME $E_F(\cdot)$ can be described as $\{R_i\}_1^n$.

The sufficiency is that, $\forall \mathbf{x} \in \chi$, by the Proposition 1, CLIME $E_F(\cdot)$ is a classifier interpretation, that is $F(\mathbf{x}) = E_F(\mathbf{x})$, then the decision boundary set is $\{DB_{ij} | i, j = 1, 2, ..., n, i \neq j\}$, where $DB_{ij} = \bigcup_{\mathbf{x} \in R_i \cap R_j} (\mathbf{x})$, which concludes the proof.

$\square$

5. Interpret FCNNs

In this section, we show how to obtain CLIME of an FCNN with common activation functions. First, the FCNN with piecewise linear activation function (PLNN) and its properties are discussed. Second, we demonstrate how to calculate CLIME for an FCNN with a nonlinear activation function. The computation of the approximation decision boundary corresponding to an FCNN is then shown. Finally, we look into how to transform an FCNN into a white-box model.

5.1. Interpretability of PLNNs

A PLNN can divide the input space into several linear regions, since it is a composite of continuous piecewise linear functions. Samples with the same activation state will be in the same linear interval. Activation state means that on each neuron, the sample is activated by the same linear region of the activation function, as illustrated in Fig.2.

According to the linear regions of PLNNs, we can naturally obtain the CLIME. On each linear interval Ω_i, CLIME can be expressed as the following formula [18]:

$$\forall \mathbf{x} \in \Omega_i, E_f(\mathbf{x}) = (\partial f / \partial \mathbf{x})^T \mathbf{x} + (\partial f / \partial b)^T b. \quad (5)$$

However, the time complexity of scanning all of the activation state is $O(k^n)$, where k is the number of segments of the activation function, and n is the number of all the hidden neurons. Thus, we can use the OpenBox method [18], which reduces the complexity by querying samples.
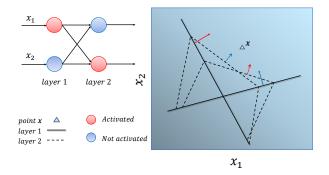
Figure 2. A hyperplane in data space correlates to the neuron's activation state. The first layer of neurons will split the whole space, the second layer will divide all of the first layer's subspaces except the preceding layer's dead zone (all neurons are not active), and so on.

In summary, PLNN can be equivalently approximated by a piecewise linear model. However, since it relies on sample queries, we term it the **lazy explainable model**.

5.2. Interpretability of FCNNs

We extend the interpretability of PLNNs to FCNNs, motivated by their interpretability. We propose piecewise linear activations $p_n(x)$ as an alternative to the commonly utilized activation functions $\sigma(x)$, which are generally Lipschitz continuous and have infinite asymptotes. We require $p_n(x)$ to meet the following condition to satisfy Definition 2:

$$\lim_{n \to \infty} (\sigma(x) - p_n(x)) = 0, \tag{6}$$

where n is the number of segments.

Algorithm 1 summarizes the construction process of $p_n(x)$. The linear approximation of sigmoid function is described in Fig.3. By the linear approximation, the CLIME of FCNNs can be obtained as the same as PLNNs. A remaining question is whether the scheme meets Definition 2.
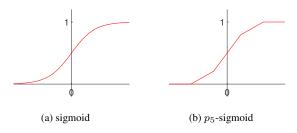

Figure 3. The piecewise linear approximation with 5 segments of the sigmoid function calculated by Algorithm 1.

For simplicity, consider an FCNN with two layers, $f : \mathbb{R}^d \to \mathbb{R}^c$. The layers are noted as $f_1(\cdot)$ and $f_2(\cdot)$ respec-

---

Algorithm 1 Linear approximate activation function
***
Require: $\sigma(x) :=$ the target activation function, $n :=$ the number of pieces, $(n \geq n_0)$, $n_0 :=$the number of asymptotic lines of $\sigma(x)$.
Ensure: $p_n(x) :=$a piece-wise linear function with n pieces
 1: Initialization: caulate the hard-$\sigma(x)$ which is consist of the asymptotic lines of $\sigma(x)$, $p_0(x) =$hard-$\sigma(x)$
 2: **for** i in range(1,$n - n_0 + 1$) **do**
 3:   compute the point by $x \leftarrow argmax_x(\sigma(x) - p_n(x))$
 4:   **if** $\#x > 1$ **then**
 5:     $x \leftarrow min(x)$
 6:   **end if**
 7:   compute tangent line l_i at point x
 8:   $p_n \leftarrow$ add new line l_i to p_n
 9: **end for**
10: **return** p_n.

---

tively, and the approximation layers (activations are generated from Algorithm 1) are noted as $g_1(\cdot)$ and $g_2(\cdot)$ respectively. For an instance $\mathbf{x} \in \mathbb{R}^d$, we have:

$$\|f_1(f_2(\mathbf{x})) - g_1(g_2(\mathbf{x}))\| =$$
$$\|f_1(f_2(\mathbf{x})) - g_1(f_2(\mathbf{x})) + g_1(f_2(\mathbf{x})) - g_1(g_2(\mathbf{x}))\|$$
$$\leq \|f_1(f_2(\mathbf{x})) - g_1(f_2(\mathbf{x}))\| + \|g_1(f_2(\mathbf{x})) - g_1(g_2(\mathbf{x}))\| \tag{7}$$

Considering the Lipschitz continuity of f_1, f_2, g_1, g_2, we have:

$$\|f_1(f_2(\mathbf{x})) - g_1(g_2(\mathbf{x}))\| \leq C \cdot \delta(n), \tag{8}$$

where C is a constant and $\lim_{n \to \infty} \delta(n) = 0$. It concludes that the approximation scheme meets Definition 2.

In summary, we can transform an FCNN into a PLNN by linearizing the activations, which means that we can obtain the interpretation of an FCNN.

5.3. Decision boundaries of FCNNs

In addition to the interpretability of FCNNs, we further desire to understand its CLIME in classification problems, i.e., the decision boundaries. However, it remains an open problem to calculate the decision boundary of black-box models. Existing methods often rely on adversarial samples to estimate the decision boundary [48–52]. In [18], a method for computing the decision boundary of PLNNs has been given. Moreover, we can implement the technology in [18] to obtain the decision boundary of FCNNs based on the interpretability of FCNNs. We would like to emphasize that FCNN is still a lazy explainable model, and its calculation of decision bounds is sample-dependent.

5.4. DecisionNet : Make Black Box White

We propose the DecisionNet (DNet), which is a totally white-box representation of corresponding PLNN that fits

Definition 1. The architecture of the DNet, which is a network with only one hidden layer, is shown in Fig.4. The parameters of DNet are copied from all of the decision boundary hyperplanes. The output of DNet is a vector, with the sign of each component reflecting the input's relative location to the related hyperplane. If the output of DNet is exactly the same in the input space, then the label of such samples must be the same. According to this characteristic, we first use the training set to mark all states of outputs. By comparing the state of an output, we can obtain the label. When a new state emerges that is not covered by the training set, we can rely on the original network for recalibration.
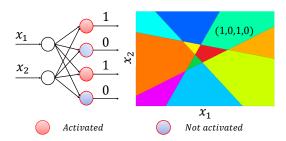

Figure 4. DNet is constructed with parameters of a black-box model's decision boundaries, which means that each parameter has a clear geometric meaning. For convenience, we display the DNet with ReLU activation, where 0 and 1 means the input is located on the negative and positive axes respectively.

In addition, DNet is more suitable as a white-box representation of a binary classification model. DNet decouples the parameters by turning the deep network into a width network. In some simple cases, DNet can save the number of neurons, while the saving of parameters depends more on the dimensionality of the input data.

6. Distillation-based CLIME

Currently, there is a huge performance gap between white-box and complicated models, resulting in proxy method failure. We propose to use PLNNs (FCNNs) to explain other black-box models, considering remarkable abilities of neural networks. For details, we propose to train PLNN (FCNN) students directly using the knowledge distillation technology in [42]. Then, the CLIME of any black-box model can be calculated based on the interpretability of PLNNs (FCNNs).

We would like to point out that distillation-based CLIME is only a preliminary scheme and does not exactly meet the demand. One reason is that the performance of FCNN still needs to be improved on large-scale data, and the other is that it is hard to obtain a precise CLIME by distillation methods.

7. Experiment

In this section, we verify the effectiveness of the proposed strategy through systematic experiments. We would like to address the following questions: (1) Is the piecewise linear approximation explanation of FCNN consistent? (2) How is the performance of DNet? (3) How is the effectiveness of distillation-based CLIME? We used 2 toy datasets [53] and 2 commonly-used datasets [54, 55]. The experiments are performed on a PC with an Intel Xeon Gold 6240 CPU and 4 Nvidia Geforce RTX 2080 Ti GPU.

7.1. Interpretability of FCNNs

Fig.5 illustrates the results of decision boundary of an FCNN on a toy dataset [53]. We trained an FCNN (2-8-5-2) with sigmoid and tanh activation functions respectively, and used Algorithm 1 to compute their linear approximation by taking the number of pieces $n = 3$ (noted as p_3-sigmoid/tanh). The results in the figure show that our method is highly efficient.

(a) sigmoid   (b) p_3-sigmoid
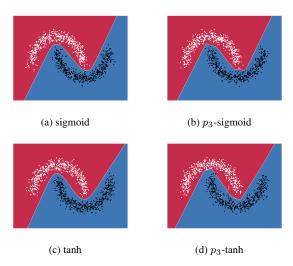
(c) tanh   (d) p_3-tanh

Figure 5. (a) and (c) are the decision boundary of the FCNN with sigmoid and tanh functions, respectively, (b) and (d) show the decision boundary of the FCNN with p_3 linear approximate function

We trained FCNNs (784-256-64-10) on FashionMNIST dataset using sigmoid, tanh and elu as activation functions respectively, and calculated their approximation PLNNs according to the subsection 5.2. As shown in Table 1, the accuracy of the approximation PLNNs tend to increase as the number of segments n increases. Sometimes we can also find that the accuracy of the approximation PLNN exceeds that of the original model.

The saliency maps of the original model and the approximation ones are shown in Fig.6. We can see that the approximation network has a semantic meaning that is extremely similar to the origin, supporting the efficacy of our method.

Table 1. Accuracy (%) of FCNNs and approximation models on FashionMNIST dataset.

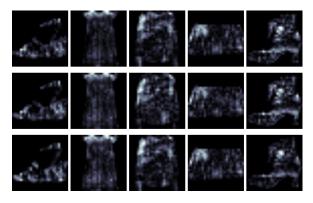| activations	baseline	$n = 2$	$n = 3$	$n = 5$
sigmoid	88.44	86.73	88.13	88.51
tanh	81.48	80.44	81.48	81.59
elu ($\alpha = 1$)	91.38	90.93	90.95	90.91

Figure 6. The comparison of the saliency maps [26] (input gradient× input) between FCNN and the approximation models. **First Row**: sigmoid, **Second Row**: $n = 3$, **Third Row**: $n = 5$ (n is the number of segments).

7.2. DNet

For intuition, we first explored the notion of DNet on a toy dataset [53] with 10000 samples. DNet has the same accuracy (100%) as the origin MLP (2-10-10-10-2) with ReLU activation, as demonstrated in Fig.7. As a comparison, there are only 15 neurons required by DNet.

Furthermore, we created a sub-dataset of FashionMNIST with only two classes (T-shirt and Trouser). After that, we trained an MLP (784-10-10-4-2) with ReLU activation using a learning rate of 0.003 and a batch size of 128. The network converges after 10 epochs of training, at which point we compute its equivalent DNet. Finally, DNet (784-5) just requires 5 neurons to match the accuracy (99.80%) of the original.

The effectiveness of DNet as demonstrated by the above experiments. We must emphasize drawbacks of DNet, which include the fact that it is not a traditional neural network and is hence less scalable.

7.3. Distillation-based CLIME

Convolution network is widely used in computer vision tasks, whose interpretability is more concerned. Therefore, we evaluate the performance of the distillation-based CLIME by interpreting the LeNet (The output layer is replaced with the softmax classifier). We use an MLP (784-256-64-10) with ReLU activation as student model, a well-trained LeNet as teacher model. We determine the best hy-
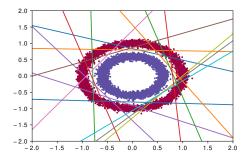

Figure 7. DNet on a toy dataset.

permeters by a grid search, where learning rate and batch size are form candidate sets [1e-5, 3e-5, 5e-5, 8e-5, 1e-4, 3e-4, 5e-4, 8e-4, 1e-3, 3e-3, 5e-3, 8e-3, 1e-2] and [32 ,64, 128, 256] respectively. Similarly, the hyperparameters of Hinton distillation are derived from two candidate sets: α from [0.5, 0.9, 0.95] and temperature t from [1, 3, 5, 10, 20]. The results on two datasets are shown in Table 2.

Table 2. Test accuracy on MNIST and FashionMNIST datasets.

Dataset	MNIST	FMNIST
LeNet	98.93	89.16
MLP-baseline	98.05	89.02
MLP-KD	**98.71**	**89.80**

The performance of MLP can be significantly boosted by distillation. Even on the FashionMNIST dataset, MLP dramatically beats LeNet. Of course, it is critical that the performance of LeNet does not significantly outperform MLP. Moreover, we plot the confusion matrix between LeNet and the MLP interpreter on the test set of FashionMNIST, as shown in Fig.8. The MLP interpreter has 92.82% of the same sample predictions as between LeNet, which confirms the validity of our distillation based CLIME.

Finally, we compare our distillation-based CLIME with LIME. Since LIME is difficult to handle high-dimensional data, we use LIME based on super-pixel. We randomly select 2000 samples in the test set of FashionMNIST, interpret and re-predict them using LIME, and then calculate the consistency with the output of LeNet. Consistency is measured by the proportion of samples with the same label to the total samples. As shown in Table.3, our distillation-based CLIME substantially outperforms LIME, which demonstrates the advantages of using a more efficient interpreter.
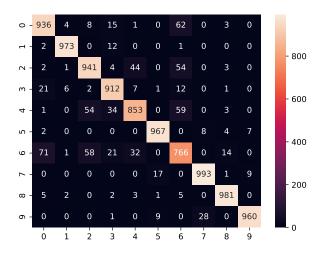
Figure 8. The confusion matrix between LeNet and MLP on FashionMNIST dataset. The horizontal axis is the LeNet classes, and the vertical one is the MLP classes.

Table 3. Comparison of the consistency (%) on FashionMNIST dataset.

method	LIME	CLIME
consistency	9.75	**93.45**

8. Conclusions

We present a method to interpretation based on approximation theory in this study, which elucidates the fundamental issues in interpretability. We find piecewise linear approximations for fully-connected neural networks to construct an explanation. We propose the MLP interpreter to generalize this notion to arbitrary black-box models. The distillation-based interpreter, on the other hand, falls short of our need for complete approximation. Our trials also reveal that the correctness of the interpretation is more dependent on the performance of white-box model, indicating a direction for further study.

References

[1] Zachary C. Lipton. The mythos of model interpretability. *Communications of the ACM*, 61(10):36–43, 2018.

[2] Feng-Lei Fan, Jinjun Xiong, Mengzhou Li, and Ge Wang. On Interpretability of Artificial Neural Networks: A Survey. *IEEE Trans. Radiat. Plasma Med. Sci.*, 7311(c):1–1, 2021.

[3] Yu Zhang, Peter Tiňo, Aleš Leonardis, and Ke Tang. A survey on neural network interpretability. *arXiv preprint:2012.14261*, 2021.

[4] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador Garcia, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Inf. Fusion*, 58:82–115, 2020.

[5] Amina Adadi and Mohammed Berrada. Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI). *IEEE Access*, 6:52138–52160, 2018.

[6] Wojciech Samek, Grégoire Montavon, Sebastian Lapuschkin, Christopher J. Anders, and Klaus Robert Müller. Explaining Deep Neural Networks and Beyond: A Review of Methods and Applications. *Proc. IEEE*, 109(3):247–278, 2021.

[7] Giulia Vilone and Luca Longo. Explainable Artificial Intelligence: a Systematic Review. *arXiv preprint:2006.00093*, 2020.

[8] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

[9] Quanshi Zhang, Ying Nian Wu, and Song-Chun Zhu. Interpretable convolutional neural networks. In *CVPR*, pages 8827–8836, 2018.

[10] David Alvarez-Melis and Tommi S. Jaakkola. Towards robust interpretability with self-explaining neural networks. In *NeurIPS*, page 7786–7795, 2018.

[11] Oscar Li, Hao Liu, Chaofan Chen, and Cynthia Rudin. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *AAAI*, pages 3530–3537, 2018.

[12] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *ICML*, pages 3319–3328, 2017.

[13] Suraj Srinivas and Francois Fleuret. Full-gradient representation for neural network visualization. In *NeurIPS*, pages 4124–4133, 2019.

[14] Stéphane Avner. Extraction of comprehensive symbolic rules from a multi-layer perceptron. *Engineering Applications of Artificial Intelligence*, 9:137–143, 1996.

[15] Razvan Pascanu, Guido Montúfar, and Yoshua Bengio. On the number of inference regions of deep feed forward networks with piece-wise linear activations. *arXiv preprint:1312.6098*, 2014.

[16] Boris Hanin and David Rolnick. Deep relu networks have surprisingly few activation patterns. In *NeurIPS*, pages 361––370, 2019.

[17] Xiao Zhang and Dongrui Wu. Empirical studies on the properties of linear regions in deep neural networks. In *ICLR*, pages 1–17, 2020.

[18] Lingyang Chu, Xia Hu, Juhua Hu, Lanjun Wang, and Jian Pei. Exact and consistent interpretation for piecewise linear neural networks: A closed form solution. In *ACM SIGKDD*, pages 1244–1253, 2018.

[19] Na Lei, Dongsheng An, Yang Guo, Kehua Su, Shixia Liu, Zhongxuan Luo, Shing-Tung Yau, and Xianfeng Gu. A geometric understanding of deep learning. *Engineering*, 6(3):361–374, 2020.

[20] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why should i trust you?" Explaining the predictions of any classifier. *ACM SIGKDD*, pages 1135–1144, 2016.

[21] X. Hu, W. Liu, J. Bian, and J. Pei. Measuring model complexity of neural networks with curve activation functions. In *ACM SIGKDD*, pages 1521—1531, 2020.

[22] Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. In *ICLR*, pages 1–16, 2018.

[23] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson. Understanding neural networks through deep visualization. *Computer Science*, 2015.

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

[25] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *ECCV*, pages 818–833, 2014.

[26] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *Computer Science*, 2013.

[27] J.T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. In *ICLR*, pages 1–14, 2015.

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

[29] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015.

[30] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and KR Müller. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*, 65:211–222, 2016.

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

[32] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning*, pages 3145–3153, 2017.

[33] Ruth C. Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *ICCV*, pages 3449–3457, 2017.

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

[35] J. Zhang, S. A. Bargal, L. Zhe, J. Brandt, X. Shen, and S. Sclaroff. Top-down neural attention by excitation backprop. *International Journal of Computer Vision*, 126(10):1084–1102, 2018.

[36] Pieter-Jan Kindermans, Sara Hooker, Julius Adebayo, Kristof T. Schütt, Maximilian Alber, Sven Dähne, Dumitru Erhan, and Been Kim. The (un)reliability of saliency methods. In *ICLR*, pages 1–11, 2018.

[37] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *CVPR*, pages 2921–2929, 2016.

[38] 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 *ICCV*, pages 618–626, 2017.

[39] Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *WACV*, pages 839–847, 2018.

[40] Haofan Wang, Zifan Wang, Mengnan Du, Fan Yang, Zijian Zhang, Sirui Ding, Piotr Mardziel, and Xia Hu. Score-cam: Score-weighted visual explanations for convolutional neural networks. *arXiv preprint:1910.01279*, 2019.

[41] Nicholas Frosst and Geoffrey Hinton. Distilling a neural network into a soft decision tree. *arXiv preprint:1711.09784*, 2017.

[42] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint :1503.02531*, 2015.

[43] Xuan Liu, Xiaoguang Wang, and Stan Matwin. Improving the interpretability of deep neural networks with knowledge distillation. In *ICDMW*, pages 905–912, 2018.

[44] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021.

[45] Markus Svensén and Christopher M Bishop. *Pattern recognition and machine learning*. Springer Cham, 2007.

[46] Zhi-Hua Zhou. Ensemble learning. In *Machine Learning*, pages 181–210. Springer, 2021.

[47] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[48] Georgios Vlassopoulos, Tim van Erven, Henry Brighton, and Vlado Menkovski. Explaining predictions by approximating the local decision boundary. *arXiv preprint:2006.07985*, 2020.

[49] Hamid Karimi, Tyler Derr, and Jiliang Tang. Characterizing the decision boundary of deep neural networks. *arXiv preprint:1912.11460*, 2019.

[50] Yu Li, Lizhong Ding, and Xin Gao. On the decision boundary of deep neural networks. *arXiv preprint:1808.05385*, 2018.

[51] Ofer Melnik. Decision region connectivity analysis: A method for analyzing high-dimensional classifiers. *Mach. Learn.*, 48(1–3):321–351, September 2002.

[52] David Mickisch, Felix Assion, Florens Greßner, Wiebke Günther, and Mariele Motta. Understanding the decision boundary of deep neural networks: An empirical study. *arXiv preprint:2002.01810*, 2020.

[53] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[54] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[55] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.