Machine Learning (2022) 111:1159–1179
https://doi.org/10.1007/s10994-021-06074-8

Towards interpreting deep neural networks via layer behavior understanding

Jiezhang Cao[1,2,3] · Jincheng Li[1,2,3] · Xiping Hu[4] · Xiangmiao Wu[1] · Mingkui Tan[1,3]

Received: 19 May 2021 / Revised: 25 August 2021 / Accepted: 22 September 2021 /
Published online: 28 January 2022

Abstract
Deep neural networks (DNNs) have achieved success in many machine learning tasks. However, how to interpret DNNs is still an open problem. In particular, how do hidden layers behave is not clearly understood. In this paper, relying on a teacher-student paradigm, we seek to understand the layer behaviors of DNNs by "monitoring" the distribution evolution for both across-layer and single-layer along the depth and training epochs, respectively. Relying on the optimal transport theory, we employ the Wasserstein distance (W-distance) to measure the divergence between the layer distribution and the target distribution. Theoretically, we prove that (i) the W-distance between the distribution of any layer and the target distribution tends to decrease along the depth; (ii) for a specific layer, the W-distance between the distribution in an iteration and the target distribution tends to decrease along training epochs; (iii) a deeper layer, however, is not always better than a shallower layer. Relying on these properties, we are able to propose an early-exit inference method to improve the performance of the multi-label classification. Moreover, our results help to analyze the stability of layer distributions and explain why auxiliary losses are helpful in training DNNs. Extensive experiments justify our theoretical findings.

Keywords Layer behavior · Wasserstein distance · Teacher-student paradigm

Editors: Yu-Feng Li, Mehmet Gönen, Kee-Eung Kim.

J. Cao and J. Li have contributed equally.

✉ Xiangmiao Wu
xmwu@scut.edu.cn

✉ Mingkui Tan
mingkuitan@scut.edu.cn

[1] South China University of Technology, Guangzhou, China

[2] Pazhou Laboratory, Guangzhou, China

[3] Key Laboratory of Big Data and Intelligent Robot, Ministry of Education, Guangzhou, China

[4] Lanzhou University, Lanzhou, China

1 Introduction

Deep neural networks (DNNs) have been successfully applied in computer vision, such as image classification (Zou et al. 2020; Wang et al. 2020; Ye et al. 2020; Fang et al. 2020; Guo et al. 2020), image generation (Brock et al. 2019; Sun et al. 2019; Hussain et al. 2020; Li et al. 2020) and speech recognition (Yeh et al. 2019; Chen et al. 2019a). Despite their success, the internal mechanism of deep neural networks is still a black box. In particular, understanding what hidden layers do remains persistently elusive. To answer this question, we seek to understand the across-layer and single-layer behaviors.

Most existing methods study the across-layer behaviors by investigating the classification performance of the intermediate layers and the last layer, and thus are limited to analyze the change of distributions. Recently, some methods only focus on final predictions of a DNN in different tasks (He et al. 2016). Due to end-to-end training, the interpretation of the behaviors of each intermediate layer in a DNN is still not clear. In addition, some works aim to produce a single prediction and observe the classification performance of each layer (Papernot and McDaniel 2018; Szegedy et al. 2014; Kaya et al. 2019). For example, Montavon et al. (2011) analyze the layer-wise evolution of DNNs based on kernel methods and empirically observe that the prediction error decreases layer after layer in a DNN. Alain and Bengio (2016) show that the classification error decreases monotonically along the depth of a DNN. Unfortunately, such monotonic property has no theoretical justifications to support these experimental findings. Moreover, these methods are hard to explore the across-layer behaviors by monitoring how distributions change across different layers. To interpret DNNs, it is important and necessary to investigate the across-layer behaviors of a DNN.

For the single-layer behaviors, most existing methods visualize the distribution information experimentally, which, however, lack of theoretical justifications to analyze the distribution stability. Recently, the distribution stability of DNNs attracts extensive attention in many machine learning tasks (Santurkar et al. 2018; Sonoda and Murata 2019). For example, some studies aim to show the training behaviors of one layer by visualizing the mean and variance of features (Santurkar et al. 2018). Unfortunately, these visualizations are subjective and lack the necessary theoretical justification. To address this, the transport analysis (Sonoda and Murata 2019) uses a denoising Autoencoder to transport mass to decrease the Shannon entropy of the data distribution. However, this method only considers a specific network, and thus is limited and inflexible to analyze a general case of DNNs. Moreover, these methods are hard to analyze the single-layer behavior and the distribution stability of layers. To address this, it is important to develop a new analytical method to interpret the distribution stability of layers.

In this paper, we apply the optimal transport theory to analyze the across-layer and single-layer behaviors. Specifically, we exploit the Wasserstein distance (W-distance) to measure the difference between the distribution of any layer and the target distribution. By monitoring the change of the W-distance, we are able to study both across-layer and single-layer behaviors.

Our contributions are summarized as follows.

- We analyze the across-layer behaviors and prove that the W-distance between the distribution of any layer and the target distribution decreases along the depth of a DNN. This means that the layers of the network can express the target distribution progressively.

- We analyze the single-layer behaviors and prove that for a specific layer, the W-distance between the distribution in an iteration and the target distribution decreases across training iterations when introducing a loss in the layer.
- We provide experimental and theoretical justifications for these findings. Moreover, these findings help to develop an early-exit inference method to improve the performance of the multi-label classification. The proposed analytical framework provides a different view of understanding and interpreting a DNN.

2 Related work

Although deep neural networks (DNNs) have achieved good performance in many machine learning tasks, the internal mechanism of DNNs is still unknown. Existing methods seek to understand DNNs from two perspectives: across-layer analysis and single-layer analysis.

2.1 Across-layer analysis

Previous studies on across-layer analysis try to visualize features in every layer of a DNN to help humans to understand. For example, Zeiler and Fergus (2014) present a important technique to visualize the intermediate features in different layers of a pre-trained classifier. This technique helps to understand why a DNN model performs so well or how to improve the DNN model. Moreover, Yosinski et al. (2015) understand deep neural networks through two deep visualization tools. Specifically, the first tool visualizes the activations on each layer when processing an image, and helps to build valuable intuitions about how the neural networks work. The second tool visualizes features at each layer via the regularized optimization in the image space.

In addition, Bau et al. (2017) evaluate how hidden units align a set of semantic concepts to quantify the interpretability of latent representations of a DNN. Dosovitskiy and Brox (2016) invert image representations with up-convolutional networks for studying image representations. Based on optimizing an objective function with gradient descent, Mahendran and Vedaldi (2015) propose a framework to invert shallow and deep representations (such as HOG (Dalal and Triggs 2005) and SIFT (Lowe 1999)) more accurately than recent methods, and is applicable to DNNs. Moreover, in several layers of a DNN, this method shows photographically accurate information about an image. These information have different degrees of geometric and photometric invariance. Zhang et al. (2018) reveal the knowledge hierarchy hidden inside a pre-trained DNN by learning an explanatory graph. Moreover, Zhang et al. (2019) learn a decision tree to clarify the specific reason for each semantic prediction of a DNN. However, these methods understand a DNN from the feature level, and thus are hard to monitor the distribution propagation across different layers.

On the other hand, some studies try to interpret the across-layer behaviors of deep neural networks by exploiting the prediction error of the intermediate layers in DNNs. For example, Montavon et al. (2011) use kernel-based tools to analyze the cross-layer behaviors and find that the prediction error decreases along the depth of the DNN. Raghu et al. (2017) propose a SVCCA tool and find that a DNN converges to final representations from the bottom up. However, these methods use additional tools to understand DNNs. In this paper, we aim to theoretically and empirically analyze the cross-layer behaviors of DNNs by label distribution mapping without any tools.

Moreover, some works (Tian 2017; Goldt et al. 2020) exploit the Teacher-Student (TS) paradigm to analyze the across-layer behaviors. Specifically, Tian (2017) uses the TS scheme to show that the population gradient on the weight for a two-layered network has an analytical formula and provides some theoretical analysis of critical points and convergence behaviors. Goldt et al. (2020) study the relations between the final generalization error of the student and the network size in a two-layered network. Unlike these methods, we exploit the Teacher-Student paradigm to analyze the behaviors in both the across-layer and the single-layer for **multi-layer** networks.

2.2 Single-layer analysis

Many single-layer analysis methods analyze the classification accuracy of the output layers to understand a DNN. For example, DSN (Lee et al. 2015) minimizes the classification error and makes the learning process of hidden layers direct and transparent. This method focuses on the transparency of the intermediate layers to the overall classification and considers the robustness of learned features in the hidden layers.

Similarly, Kaya et al. (2019) introduce internal classifiers into off-the-shelf DNNs to understand the overthinking phenomenon of neural networks by studying how the prediction changes along the depth of a DNN. Gupta and Schütze (2018) explain recurrent neural networks by understanding the layer-wise semantic accumulation behaviors. Santurkar et al. (2018) visualize the mean and variance of features to show the behaviors of one layer to understand DNNs. In addition, Sonoda and Murata (2019) investigate the feature map inside a DNN by tracking the transport map, and prove that a deep Gaussian DAE transports mass to decrease the Shannon entropy of the data distribution. However, these methods try to explain this phenomenon with experimental evidence but lack of theoretical guarantees.

To address this, existing studies apply the information theory in a DNN to improve interpretability. For example, the information bottleneck method (Tishby and Zaslavsky 2015) quantifies the performance of DNNs using the mutual information, and shows that the representations at any layer are related to the structural phase transitions along the information curve. Saxe et al. (2018) argue three claims of the information bottleneck theory of DNNs, and exploit some phenomena to support their argument by analytical results and simulation. In addition, the variational information bottleneck (Bang et al. 2019) is a system-agnostic interpretable method to consider both briefness and comprehensiveness to explain DNNs more efficiently. However, these methods analyze the information entropy of every hidden layer. How to analyze the change of distribution in a specific layer through different iterations remains an open question. In this paper, we aim to theoretically and experimentally interpret deep neural networks by analyzing the single-layer behaviors.

3 Notation and preliminaries

Throughout the paper, we use the following notations. Specifically, we use bold lower-case letters (*e.g.*, $\mathbf{x}$) to denote vectors, and bold upper-case letters (*e.g.*, $\mathbf{X}$) to denote matrices. We denote the transpose of a vector (*e.g.*, $\mathbf{x}^\mathsf{T}$) or matrix (*e.g.*, $\mathbf{X}^\mathsf{T}$) by the superscript $^\mathsf{T}$. For two matrices $\mathbf{A}$ and $\mathbf{B}$ of the same size, the inner-product can be defined as $\langle \mathbf{A}, \mathbf{B} \rangle = \mathrm{tr}(\mathbf{A}^\mathsf{T}\mathbf{B})$, where $\mathrm{tr}(\cdot)$ is the trace of a matrix. Let $\mathbf{I}$ be an identity matrix, $\mathbf{1} = [1, \ldots, 1]^\mathsf{T}$ be a vector

where every element is equal to one, and let $[n]=\{0, 1, \ldots, n\}$. Let supp (μ) be the support of a distribution μ (Lee et al. 2017).

3.1 Optimal transport

The optimal transport problem originally seeks to measure the difference between two probability distributions on a given metric space. In this paper, we will exploit the optimal transport theory (Villani 2008) to study the layer behaviors in a DNN. Specifically, since the Wasserstein distance is an effective metric to establish a geometric tool for effectively comparing probability distributions, it helps to narrow the gap between the interpretability of DNNs and human understanding. Formally, the Wasserstein distance can be defined as follows.

Definition 1 (*Wasserstein distance* (Villani 2008)) Given a target distribution μ and a predicted distribution $\widehat{\mu}$, and the cost matrix $\mathbf{C}$ defined as $C_{\kappa,\kappa'}= d^p_{\mathcal{K}}(\kappa, \kappa')$ with the metric $d_{\mathcal{K}}$, where κ, κ' are label tags, then the Wasserstein distance seeks to find a transportation matrix $\mathbf{T}$ by transporting a mass $\widehat{\mu}$ to μ,

$$\mathcal{W}(\widehat{\mu}, \mu) = \inf_{\mathbf{T}\in\Pi(\widehat{\mu},\mu)} \langle \mathbf{T}, \mathbf{C}\rangle, \tag{1}$$

where $\Pi(\widehat{\mu}, \mu)$ is the set of couplings and defined as

$$\Pi(\widehat{\mu}, \mu)=\{\mathbf{T}\in\mathbb{R}_+^{K\times K}\colon \mathbf{T}\mathbf{1}=\widehat{\mu}, \mathbf{T}^{\mathsf{T}}\mathbf{1}=\mu\}. \tag{2}$$

In practice, the cost matrix can be constructed by word2vec (Mikolov et al. 2013). Note that Problem (1) is non-convex, leading to an intractable optimization problem (Genevay et al. 2018). To address this, we apply the entropic Wasserstein distance (Cuturi 2013) as follows,

$$\mathcal{W}'_{\alpha}(\widehat{\mu}, \mu) := \inf_{\mathbf{T}\in\Pi(\widehat{\mu},\mu)} \langle \mathbf{T}, \mathbf{C}\rangle - \frac{1}{\alpha}H(\mathbf{T}), \tag{3}$$

where $H(\mathbf{T})=-\sum_{\kappa,\kappa'} T_{\kappa,\kappa'}(\log(T_{\kappa,\kappa'})-1)$.

Note that Problem (3) is a convex optimization problem (Peyré and Cuturi 2019), and thus it can be solved efficiently by some gradient methods, *e.g.*, the Sinkhorn algorithm (Knight 2008).

3.2 Label distribution

To explore the across-layer and single-layer behaviors, we consider learning a label distribution in every layer. Such a label distribution can be obtained by optimizing the multi-label classification problem (Geng 2016). In this problem, a predicted distribution or a target distribution can be modeled as a label distribution (*i.e.*, probability distribution), indicating the relative importance of each label involved in the description of an instance, as shown in Fig. 2. Note that the sum of the target distribution or the predicted distribution is equal to one.

Specifically, given training samples $\mathcal{S}=\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$, we seek to learn a model f from data space $\mathcal{X}$ into the label space $\mathcal{Y}$. In Fig. 1 (a), for L-layered neural networks, we denote $\tilde{f}_{0:l}=\tilde{f}_l\circ\cdots\circ\tilde{f}_0, l\leq L$ as the output of the l-th layers, then the label

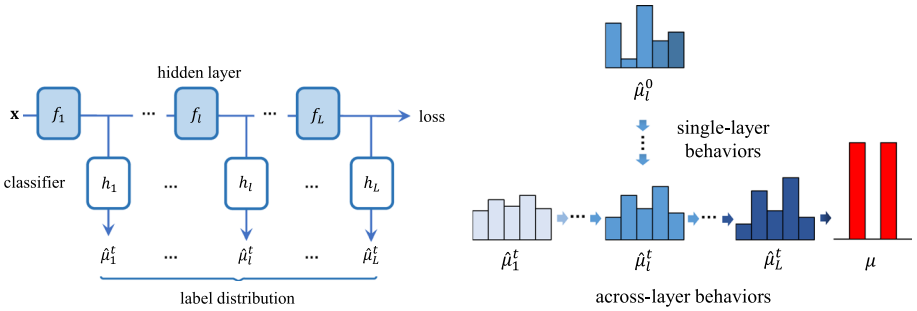

Fig. 1 Demonstration of the forward propagation of a deep neural network. **a** Network forward propagation: Given an input $\mathbf{x}$, we fix the l-th hidden layer and only optimize the corresponding classifier to output label distributions $\widehat{\mu}_l^t$ at time t. Their network architecture can be referred to Supplementary materials. **b** Label distribution behaviors: they contain the across-layer behaviors and the single-layer behaviors. For the across-layer behaviors, the label distribution $\widehat{\mu}_0^t$ of the first layer propagates to $\widehat{\mu}_L^t$ of the L-th layer such that it can be close to the target distribution μ. For the single-layer behaviors, the label distribution $\widehat{\mu}_l^0$ propagates to $\widehat{\mu}_l^t$ during the epochs in one layer

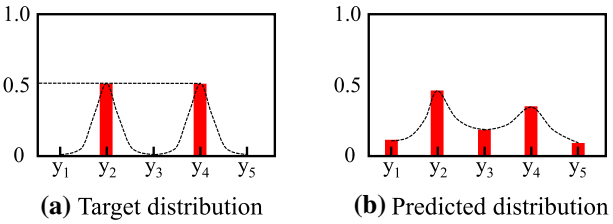

Fig. 2 An example of the label distribution. Taking five labels as an example, a target distribution or a predicted distribution can be modeled as a label distribution. Here, we build the target distribution using ground-truth labels and consider the output of DNNs as the predicted distribution. Note that the sum of the label distribution is equal to 1

distribution mapping can be defined as $f_l := h_l \circ \tilde{f}_{0:l}$, where h_l is a probability function (*e.g.*, FC+softmax (Frogner et al. 2015)). Here, $\tilde{f}_l, l \in [L]$ have different input and output domains. In contrast, all $f_l, l \in [L]$ have the same input domains and the same output domains, because they feed the same input and then output a label distribution to be close to the target distribution. Our goal is to learn a model f_l to let the predicted distribution $\widehat{\mu}_l$ to be close to the target distribution μ. Then, we optimize the empirical risk as follows:

$$\widehat{\mu}_l = \arg\min \mathbb{E}_{\mathcal{S}}[d(\mu_l, \mu)], \tag{4}$$

where $\mu_l = f_{l\#}(\mu_0)$, $l \leq L$, and $f_{l\#}$ is a pushforward operator of the original distribution μ_0 (Villani 2008), and $d(\cdot, \cdot)$ is some distribution divergence, such as a cross-entropy (CE) loss and the Wasserstein loss (Frogner et al. 2015). In practice, the label distribution is obtained by optimizing Problem 4. In general, the changes of label distributions across different layers or iterations can reflect the layer behaviors.

4 Teacher-student analysis for layer behaviors

It is well known that the neural network has a strong ability to fit an unknown function or distribution. However, the fitting behaviors of the neural network have not been well studied. To resolve this, we start from the perspective of whether the internal layers have sufficient fitting ability as that of the whole network. To this end, we use the final outputs to represent the fitting ability of the whole network (namely the **teacher**). In this sense, it is natural for us to obtain the distribution of intermediate layers by training a **student** network under the teacher's guidance.

Moreover, the Teacher-Student (TS) framework has a good theoretical basis to help us analyze and understand neural networks (Tian 2017; Goldt et al. 2020). Note that if we directly use the ground truth labels to represent the target distribution, the analysis would be very difficult. Fortunately, under TS, we can represent the target distribution via a certain function (*i.e.*, the teacher), and thus exploit some mathematical properties (*e.g.*, Lipschitz continuous) for further analyses. Specifically, we can transform the problem of learning an internal classifier to fit the ground-truth into a problem of learning a student to fit the teacher. Relying on this framework, we are able to analyze the across-layer behaviors and the single-layer behaviors. Specifically, we learn a neural network to approximate the ground-truth (i.e., the target distribution). Here, the ground-truth can be represented by the output of a teacher network (T-net) [52]. Then, the output of a student network (S-net) can be close to the output of the T-net (See Fig. 3).

To measure the difference between the output of the S-net and the output of the T-net, we may apply some distribution distance, such as Jensen-Shannon (JS) divergence, Kullback-Leibler (KL) divergence and Chebyshev distance. However, these metrics neglect to consider the geometric structure of the probability distributions. For example, when two distributions overlap slightly, JS-divergence becomes a constant and KL-divergence becomes meaningless. Moreover, the Chebyshev distance is ℓ_∞ distance, which is unsuitable to measure the distance between two distributions. On the contrary, the Wasserstein distance is an alternative metric to establish a geometric tool for effectively comparing probability distributions. Furthermore, the Wasserstein distance is more suitable to conduct our theoretical analyses which are based on the optimal transport. In this paper, we propose to apply the Wasserstein distance to measure the divergence to analyze how the layer distributions change across different layers or iterations in a specific layer, as shown in Fig. 1b.

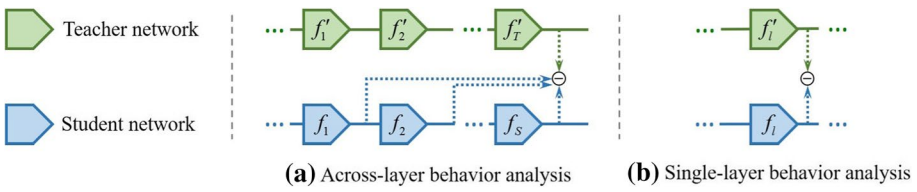

Fig. 3 The teacher-student paradigm for analyzing layer behaviors of DNNs. This paradigm contains a teacher network (T-net) and a student network (S-net). **a** For the across-layer behavior, we study the ability of the student network to express distributions of the teacher network. **b** For the single-layer behavior, we investigate the change of the Wasserstein distance between the distributions of the student network and the teacher network

4.1 Across-layer behavior analysis

Relying on the teacher-student analysis framework, we propose to analyze the across-layer behaviors. To this end, we measure the across-layer Wasserstein distance between the distribution $\widehat{\mu}_l$ in the l-th layer and the target distribution μ, *i.e.*, $\mathcal{W}(\widehat{\mu}_l, \mu)$. The detailed algorithms of the across-layer W-distance can be referred to Algorithm 1.

By using the Wasserstein distance, we are able to analyze the ability of each layer of the S-net to express the distribution of the T-net, as shown Fig. 3a. Based on the definition of the Barron function (See Supplementary materials), we derive an approximation bound for finitely deep neural networks. Specifically, given a T-net composited by L Barron functions, we derive an approximation bound *w.r.t.* the across-layer Wasserstein distance as follows.

Theorem 1 (Across-layer Wasserstein distance approximation) *Given an input distribution* μ_0 *and a function* φ_i, *and let* $L_l = \log(l)+1$, *if* $\text{supp}\,(\mu_0) \subset \mathcal{K}_0$ *and* $\varphi_i(\mathcal{K}_{i-1}) \subseteq \mathcal{K}_i$, $1 \le i \le l$, φ_i *is an* $\left(L_{l-1}-1/L_l\right)$*-Lipschitz and is a Barron function, in other words,* $\varphi_1 \in \Omega_{\mathcal{K}_0}(C_0)$, $\varphi_i \in \Omega_{\mathcal{K}_{i-1}+s\mathcal{B}_{m_{i-1}}}(C_i)$, *then there exists a network f with l hidden layers with* $\left\lceil 4C_l^2 m_l L_{l-1}^2 L_l^2/\epsilon^2 \right\rceil$ *neurons in the i-th layer,*

$$\mathcal{W}(\widehat{\mu}_l, \mu) \le \frac{\epsilon^2}{L_l^2}\left(\left(2C_l\sqrt{m_l} + D_l\right)^2 \frac{\delta}{s^2} + 1\right), \tag{5}$$

where $l \le L$, $\epsilon, \delta, s > 0$ *and* D_l *is the diameter of the set* $\mathcal{K}_l$.

Proof See supplementary materials for the proof. □

Algorithm 1 Across-layer Wasserstein distance.

Input: Training data $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N$, pre-trained hidden layers f_l, pre-trained classifiers $h_l, l = 1, \ldots, L$.
Output: $\bar{\mathcal{W}}(\widehat{\mu}_1, \mu), \ldots, \bar{\mathcal{W}}(\widehat{\mu}_L, \mu)$
1: **for** $l = 1, \ldots, L$ **do**
2: Obtain the label distribution $\widehat{\mu}_l(\mathbf{x}_i)$
3: Calculate the Wasserstein distance

$$\bar{\mathcal{W}}(\widehat{\mu}_l, \mu) = \frac{1}{n}\sum\nolimits_{i=1}^{N} \mathcal{W}'(\widehat{\mu}_l(\mathbf{x}_i), \mu(\mathbf{x}_i))$$

4: **end for**

In Theorem 1, we provide an error bound when using a neural network with l hidden layers to approximate L Barron functions. The error bound decreases with the increasing of layers under certain conditions. In this sense, deeper layers have a smaller approximation errors than shallow layers. In other words, the W-distance between the distribution of any layer and the target distribution has a decreasing tendency along with the depth.

In addition, deep layers are not always better than shallow layers for some specific samples. Such samples are often classified correctly in the shallow layer rather than the deep layer when the diameter D_l is very large for this set of samples. Hence, shallow layers can behave better than deep layers for these kinds of samples.

4.2 Single-layer behavior analysis

Based on the teacher-student framework, we can also analyze the single-layer behaviors. To this end, we measure the single-layer Wasserstein distance between the distribution $\widehat{\mu}_l^t$ in a training iteration and the target distribution μ, *i.e.*, $\mathcal{W}(\widehat{\mu}_l^t, \mu)$. The detailed algorithms of the single-layer W-distance can be referred to Algorithm 2.

By using the Wasserstein distance, we measure the difference between the output of the intermediate layer of the S-net and the output of the T-net, as shown in Fig. 3b. Specifically, given a T-net f' and training dataset $\mathcal{S}$, we first learn an S-net f that minimizes the following training loss as follows,

$$\min_f \quad \mathbb{E}_{\mathbf{x}\sim\mathcal{S}}\Big[\big\|f(\mathbf{x}) - f'(\mathbf{x})\big\|^2\Big]. \tag{6}$$

By learning the S-net f and the corresponding distribution μ^t in a specific layer, we explore how the distribution propagates in one layer as follows.

Theorem 2 *At the initialization t*=0, *the pushforward* $f_{\#}\mu_t$ *with Gaussian distribution satisfies the backward heat equation* (*Sonoda and Murata* 2019), *then the gradient of the Wasserstein distance satisfies*

$$\partial_t \mu^{t=0}(\mathbf{x}) = -\mathrm{grad}\big(\mathcal{W}(\mu^t, \mu)\big), \tag{7}$$

where grad(·) *is a gradient operator* (*Sonoda and Murata* 2019).

Proof See supplementary materials for the proof. □

Algorithm 2 Single-layer Wasserstein distance.

Input: Training data $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N$, training epoch T
Output: $\bar{\mathcal{W}}(\widehat{\mu}_l^0, \mu), \ldots, \bar{\mathcal{W}}(\widehat{\mu}_l^T, \mu)$

1: **for** $t = 0, \ldots, T$ **do**
2: Train the l-th classifier h_l at epoch t
3: Obtain the label distribution $\widehat{\mu}_l^t(\mathbf{x}_i)$
4: Calculate the Wasserstein distance

$$\bar{\mathcal{W}}(\widehat{\mu}_l^t, \mu) = \frac{1}{n}\sum\nolimits_{i=1}^N \mathcal{W}'(\widehat{\mu}_l^t(\mathbf{x}_i), \mu(\mathbf{x}_i))$$

5: **end for**

Note that Theorems 1 and 2 do not constrain that the predicted distributions $\hat{\mu}_l$ must be derived from a linear classifier. In other words, it can be any differentiable probability functions. From Theorem 2, by introducing an target distribution μ to supervise the corresponding distribution μ^t in a specific layer during training, the single-layer W-distance can be decreased along the negative gradient. It means that the label distribution can be close to the target distribution across training iterations. However, the stability of distribution in a specific shallow layer is difficult to be guaranteed even with Batch Normalization. We provide experimental results in Sect. 7.1 and the proof in Supplementary materials.

In practice, we employ an auxiliary loss (*i.e.*, Eq. (6)) in a specific layer as the supervision. In contrast, the distribution stability in each layer is hard to be guaranteed when

we do not take an auxiliary loss as the supervision information. These results help to analyze the stability of layer distributions and explain why auxiliary losses are helpful in training DNNs. We provide experimental justifications in Sect. 6.

5 Early-exit inference for multi-label classification

From Theorem 1, we theoretically prove that the W-distance decreases along with the depth of DNNs. In other words, we always get the smallest W-distance in the last layer, suggesting that it can be more likely to make correct predictions. In practice, however, some intermediate layers are sufficiently representative to make correct predictions and more layers may even make wrong predictions. In this sense, it is possible and reasonable to improve the performance via an early-exit strategy, which attempts to early exit the correct predictions in the intermediate layers instead of the last layer.

Recently, existing methods (Kaya et al. 2019; Scardapane et al. 2020) apply confidence-based early-exit method in the multi-class classification problem. However, these methods are not suitable for the multi-label classification because they consider single probability only. To address this, we propose a new early-exit method for the multi-label classification problem. An intuitive understanding of the early-exit method is shown in Fig. 4.

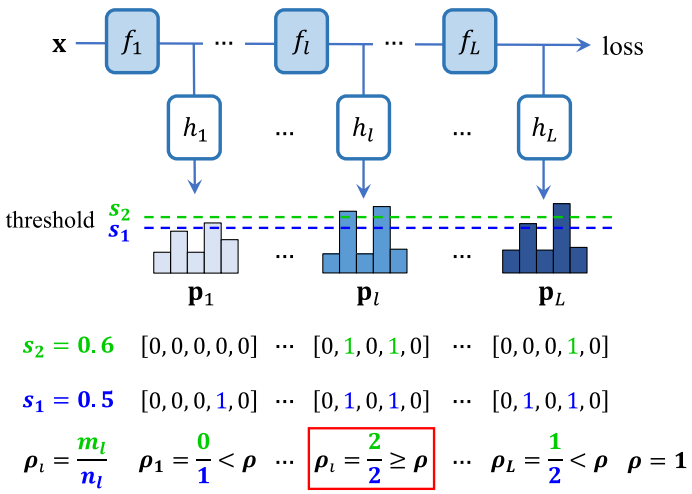

Fig. 4 An intuitive understanding of the early-exit method. Given a deep neural network and classifiers of intermediate layers, we obtain label distributions in every layer. When the early-exit condition is satisfied (See the red box), *i.e.*, the ratio ρ_l exceeds the ratio ρ, then the label distribution can be exited as the final output prediction (Color figure online)

Algorithm 3 Early-exit inference method.

Input: Testing data $\mathbf{x}$, a pre-trained neural network f_l and classifiers h_l, $l = 1, \ldots, L$, thresholds $s_1 = 0.5, s_2 \in [0.5, 1]$ and a ratio $\rho \in [0.5, 1]$
Output: Predicted probability vector $\mathbf{p}_l$

```
for l = 1, ..., L do
    Predict a probability vector p_l
    Calculate the number of probability values in p_l greater than s_1 and s_2, respectively, i.e., n_l and m_l
    Calculate the ratio ρ_l = m_l/n_l
    if ρ_l ≥ ρ then
        Early-exit p_l in the l-th layer
        break
    end if
end for
```

Given a sample $\mathbf{x}$ and a pre-trained DNN, the auxiliary classifier outputs a probability vector $\mathbf{p}_l$ in the l-th intermediate layer, as shown in Fig. 4. To early-exit the sample in the multi-label classification setting, we first need to estimate the number of the predicted labels, then we determine whether these labels satisfy some exit conditions. To this end, for each sample, we calculate the numbers (denoted as n_l and m_l) of probability values in $\mathbf{p}_l$ exceeding thresholds s_1 and s_2, respectively. Here, n_l is the number of the predicted labels, and m_l is the number of these labels with the high confidence in $\mathbf{p}_l$. Then, we define a high-confidence ratio for the early-exit strategy, *i.e.*,

$$\rho_l = \frac{m_l}{n_l}, \quad l \leq L. \tag{8}$$

The detailed algorithm of the early-exit reference method is shown in Algorithm 3. In practice, we use a sigmoid function in the last layer of the pre-trained classifiers. Then, we set the threshold s_1 as 0.5 and search the threshold s_2 in [0.5, 1]. Here, we choose 0.5 since it is the boundary of the sigmoid function. Moreover, a sample can be early-exited when $\rho_l \geq \rho$, where ρ is selected in [0.5, 1] since we consider the exited sample is with a high confidence. For every threshold s_2 and ratio ρ, we calculate the classification accuracy with the predicted probability vectors on testing data. Last, we choose the best threshold s_2 and ratio ρ with the highest classification accuracy. The experiments for different parameters s_2 and ρ on the test set are conducted in Table 4 in Sect. 7.1.

6 Experiment

In this section, we conduct several experiments to analyze the across-layer and single-layer behaviors. Specifically, we verify that the across-layer and single-layer Wasserstein distances decrease along the depth and training iterations, respectively. Furthermore, we find that the Wasserstein distance of a deeper layer is not always lower than that of a shallower layer for some samples. Therefore, we propose an early-exit inference method to exit these samples in the intermediate layers of DNNs to improve the classification performance.

6.1 Implementation details

All experiments are implemented on PyTorch. In the training, we use an SGD optimizer with an initial learning rate of 0.01. The learning rate decays by a factor of 10 for every 40 epochs. The momentum and weight decay are set to 0.9 and 10^{-4}, respectively. We

pre-train models (including ResNet-18 (He et al. 2016) and VGG-16 (Simonyan and Zisserman 2015)) on the ImageNet dataset (Deng et al. 2009) for 100 epochs with a batch size of 128. Then, we freeze the model parameters and train the internal classifiers (*i.e.*, the network h_l in Fig. 1a) for 100 epochs. Each internal classifier (*i.e.*, S-net) contains a fully connected layer and a sigmoid function. We train the network using the binary cross entropy loss. We explore the across-layer and single-layer distribution propagation on the training set. In addition, we set α=0.01 in Eq. (3) to achieve balanced results. For simplicity, we use ResNet to represent the ResNet-18 model and VGG to represent the VGG-16 model in the main paper. The detailed partitions of ResNet and VGG can be referred to Supplementary materials.

6.2 Datasets and evaluation metrics

6.2.1 Datasets

We conduct experiments on two benchmark multi-label classification datasets, including VOC2007 (VOC) (Everingham et al. 2010) and MS-COCO (COCO) (Maas et al. 2013). These datasets are widely used in the multi-label classification task (Shi et al. 2018; Wang et al. 2016; Durand et al. 2019).

- VOC2007 (Everingham et al. 2010) has 9,963 images from 20 object categories, which are divided into training, validation and test sets.
- MS-COCO (Maas et al. 2013) contains 82,783 images as the training set and 40,504 images as the validation set. The objects are categorized into 80 classes with approximately 2.9 object labels per image.

6.2.2 Evaluation metrics

For the evaluation metrics, we use the classification accuracy, average per-class F1 (CF1), average overall F1 (OF1) and mean average precision (mAP) as the evaluation metrics. These metrics are widely used to evaluate the performance of the multi-label classification task (Chen et al. 2019c, b).

- Classification accuracy: the number of correctly classified samples divided by the total number of samples.
- Average per-class F1: the average of the F1 score of each class. Here, the F1 score can be interpreted as a weighted average of the precision and recall.
- Average overall F1 (OF1): the average of the F1 score of overall class.
- Mean average precision (mAP): the mean of the average precision (AP) scores for all classes. Here, average precision is defined to find the area under the precision-recall curve above.

6.3 Results on across-layer distribution propagation

In this experiment, we investigate the expression ability of each layer for ResNet and VGG. Such the expression ability can be measured by the Wasserstein distance (W-distance)

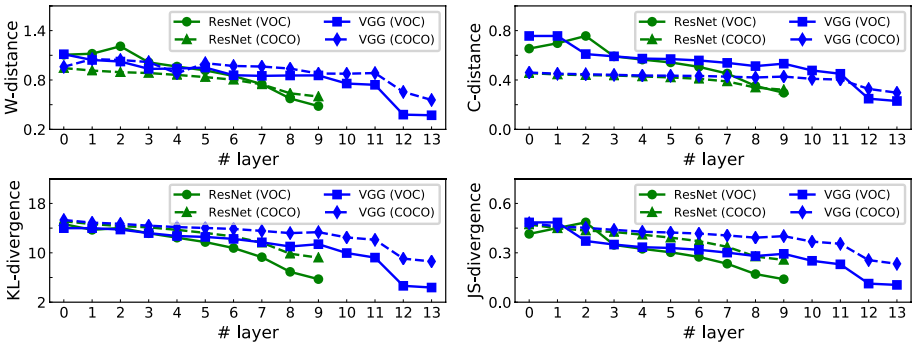

Fig. 5 Distribution distance across different layers for ResNet and VGG networks, including W-distance, C-distance, KL-divergence and JS-divergence. All experiments are conducted with three trials. More details of layers can be referred to supplementary materials

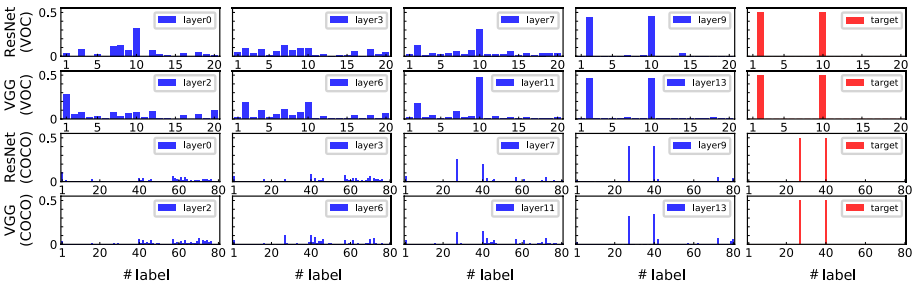

Fig. 6 Distribution propagation across different layers for different networks. The first four columns are learned label distributions in different layers, and the last column is the target label distribution

between the label distribution of a layer and the target distribution. In Fig. 5, deep layers of both ResNet and VGG have a smaller W-distance than shallow layers. It verifies Theorem 1 that deep layers have smaller approximation error than shallow layers. Therefore, deep layers have a better expression ability than shallow layers. In addition, the W-distance tends to decrease across different layers, *i.e.*, the W-distance decreases from shallow layers to deep layers. Note that shallow layers have a similar ability to express the target distribution since they have similar values of the W-distance. When approaching the last layer, the W-distance drops to a very small value. This implies that a sufficient number of layers are able to express the target distribution.

Next, we investigate how the label distribution propagates across layers to understand the learning process from layer to layer in a DNN. Specifically, we pre-train ResNet and VGG on the VOC and COCO datasets. Without loss of generality, we randomly choose a training sample in the VOC and COCO datasets, respectively. Then, we normalize the prediction of the classifier as the probability distribution (*i.e.*, the sum of the label prediction probabilities of a sample is one). From Fig. 6, the label distribution of one sample propagates from the first layer to the last layer. In the shallow layers, the label distribution is far from the target distribution; but it can be close to the target distribution in deep layers. In contrast to the decreasing tendency in Fig. 5, the label distribution of one sample may not approach the target distribution progressively. For example, in the first row of Fig. 6, the

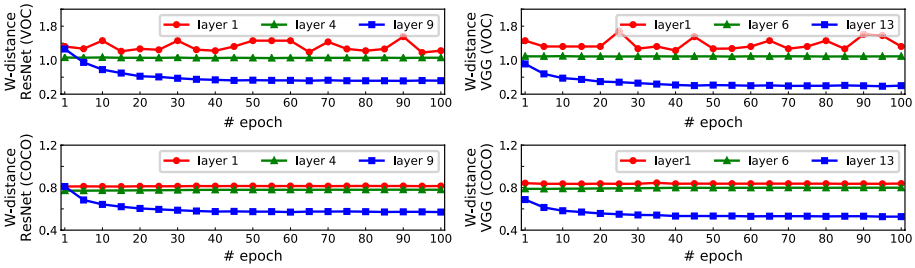

Fig. 7 The Wasserstein distance between the distribution in an epoch and the target distribution across different training epochs for different networks. We choose the 1, 4, 9-th layer for ResNet and the 1, 6, 13-layer for VGG

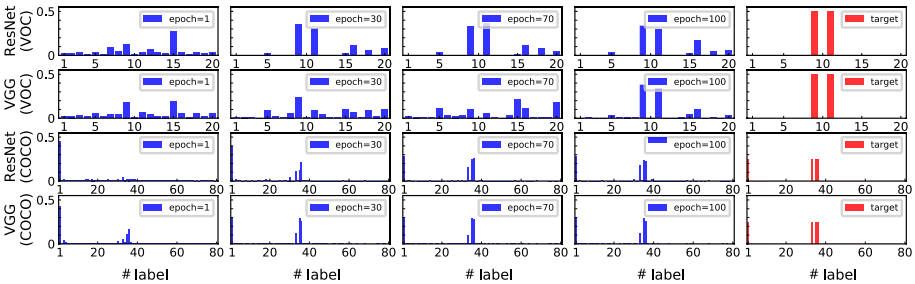

Fig. 8 Distribution propagation across different training epochs of ResNet-18 and VGG-16

probability of the 10-th class is large in the 0-th layer, but decreases by the third layer. It means that existing networks are hard to control the layer distribution of every sample to be close to the target distribution. One possible reason is that there may exist many redundant information in the learning process.

6.4 Results on single-layer distribution propagation

In this experiment, we investigate how distributions propagate when training DNNs. Specifically, we consider using the Wasserstein distance to measure the distance between the distribution in an iteration and the target distribution. The single-layer Wasserstein distance and the distribution propagation are shown in Figs. 7 and 8, respectively.

From Fig. 7, the distributions in the first few layers of ResNet and VGG often fluctuate significantly due to the limited discriminative power of very shallow layers. For the intermediate layers (*e.g.*, the 4-th layer of ResNet), the W-distance is steady at a larger value and the changes of distributions tend to be stable since the layers have certain expression ability and nearby the last layer with the supervision. When approaching the last layer, the supervision is sufficient to decrease the W-distance. This justify Theorem 2,which says the Wasserstein distance can be decreased with sufficient supervision.

From Fig. 8, we show the label distribution of one sample propagates from the first epoch to the last epoch. At the early training stage, the label distribution is far from the target distribution. At the latter learning stage, the label distribution can be close to the target distribution.

6.5 Performance on a single sample

In this experiment, we investigate the contribution of every sample in different layers. In practice, a deep neural network constructs more complex features progressively throughout the layers (Lee et al. 2011). An interesting question arises: does each sample behave the same? To answer this question, we first define the concept of sample difficulty in terms of the W-distance. As shown in Fig. 9, samples can be divided into three categories: easy, hard and confused. The intuition for training or testing samples is that:

- **Easy samples** should have a small W-distance (near zero) in the first few layers, *i.e.*, they should be classified correctly with high confidence in a shallow layer.
- **Hard samples** should have a large W-distance in a deeper layer, *i.e.*, they cannot be resolved at all, or can be resolved only near the last layer.
- **Confused samples** have a small W-distance in shallow layers and a large W-distance in deep layers. It means that although these samples are classified correctly in a shallow layer, they are still misclassified in the last layer.

Compared with the easy and hard samples, the confused samples have a greater effect on the classification performance in practice. Next, we investigate the number of the confused samples in each layer of ResNet. At the 0-th layer, 1.17% confused samples are correctly classified. Ideally, if we can learn a good classifier to early exit these confused samples, the cumulative accuracy of ResNet-18 is 70.44% on the VOC dataset. Considering the difficulty of samples would help to interpret the training of models and design the training loss, which can improve the performance.

6.6 Results of early-exit inference method

Next, we discuss how to exploit the behaviors of different layers to improve the performance of the classifier. In practice, we observe that in deep neural networks, some samples are correctly classified in intermediate layers but misclassified in the last layer. From Tables 1 and 2, the accuracy of ResNet and VGG on VOC are 64.48% and 68.96%, respectively. In other words, 35.52% and 31.04% of the samples are misclassified on the test set of VOC for ResNet and VGG, respectively. For these samples, 5.96% and 7.76% of the samples are correctly classified in all intermediate layers of ResNet and VGG, respectively.

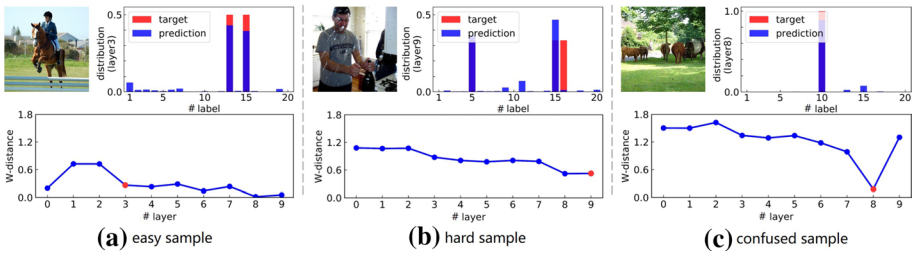

Fig. 9 Demonstration of properties on easy, hard and confused samples. Here, we randomly choose training samples as intuitive examples. Each block has three subgraphs, including an input image, the predicted label distribution and the W-distance across different layers. For each block, we show the predicted label distribution corresponding to the red point on the bottom row (Color figure online)

(The phenomenon on COCO is similar to VOC.) Based on this phenomenon, we can improve performance by early exiting such confused samples. Different from the strategy of SDN (Kaya et al. 2019), performing early-exit for the multi-label classification is very challenging. Therefore, we design a new early-exit inference method for the multi-label prediction. In Table 3, our proposed method consistently outperforms the baseline methods. It means that the confused samples can be early exited in the intermediate layers.

7 Further experiments

In this section, we first provide the ablation study for the parameters in our early-exit inference method. Then we apply the Wasserstein distance to study the stability of the distribution propagation in a specific layer. Last, we further exploit the across-layer behaviors with other distribution distances.

7.1 Ablation study for different hyper-parameters s_2 and ρ

In this experiment, we show how the two parameters s_2 and ρ affect the performance of our early-exit inference method on the test set by numerical experiment in Table 4. From Table 4, the accuracy of the model increases from 64.48% to 66.01%. In other words, the confused samples, which are correctly classified in intermediate layers but misclassified in the layer, can be early exited in the intermediate layers.

7.2 Stability of distribution propagation in one layer

Recently, the training stability of a deep neural network attracts extensive attention in the field of machine learning (Bjorck et al. 2018; Wu et al. 2019; Chen et al. 2020). To address this, one can use the Batch Normalization (BN) (Ioffe and Szegedy 2015) to stabilize the distributions of the hidden layers during training. The popular belief is that the success of the Batch Normalization stems from controlling the stability of distribution during training. To justify this, we provide theoretical and empirical justifications for the distribution stability in one layer. Specifically, we apply the Wasserstein distance to measure the stability between the distribution in an iteration and the target distribution across different training iterations in a specific layer. The conventional understanding of BN suggests that the W-distance should decrease. However, the first layer of ResNet or VGG may fluctuate during the training process, as shown in Fig. 7. In contrast, deep layers often have a better ability than shallow layers to express the target distribution. Furthermore, we prove that the stability of distribution is difficult to guarantee even with BN (See the proof in Supplementary materials).

Table 1 The accuracy of different layers of ResNet

#layer	0%	1%	2%	3%	4%	5%	6%	7%	8%	9%	Ideal result %
VOC	1.17	0.44	0.69	0.71	0.55	0.50	0.53	0.85	0.53	64.48	**70.44**
COCO	0.49	0.43	0.47	0.28	0.30	0.31	0.37	0.78	0.50	27.33	**31.26**

Bold values denote the ideal results

Table 2 The accuracy of different layers of VGG

#layer	0%	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%	11%	12%	13%	Ideal result%
VOC	2.34	0.00	0.71	0.67	0.50	0.24	0.48	0.22	0.20	0.14	0.28	0.34	1.62	68.96	**76.72**
COCO	0.35	0.35	0.27	0.38	0.33	0.19	0.21	0.17	0.18	0.13	0.38	0.37	1.63	32.41	**37.35**

Bold values denote the ideal results

Table 3 Improve performance of ResNet and VGG on the VOC and COCO datasets. For comparison, we apply the classification accuracy, CF1, OF1 and mAP as evaluation metrics

Method	VOC				COCO			
	Accuracy (%)	CF1 (%)	OF1 (%)	mAP (%)	Accuracy (%)	CF1 (%)	OF1 (%)	mAP (%)
ResNet	64.48	58.02	**59.10**	85.18	27.33	55.65	60.30	64.36
ResNet+early-exit	**66.01**	**58.67**	58.76	**85.49**	**30.03**	**57.49**	**61.38**	**67.28**
VGG	68.96	58.58	59.71	88.48	32.41	59.37	62.94	70.64
VGG-early-exit	**69.85**	**59.49**	**59.94**	**88.57**	**33.95**	**60.32**	**63.73**	**71.95**

Bold values denote the best results

Table 4 The accuracy (%) of different parameters s_2 and ρ of ResNet-18 on VOC2007

ρ	s_2					
	0.5	0.6	0.7	0.8	0.9	1
0.5	65.99	65.67	65.24	65.02	64.83	64.44
0.6	65.99	65.96	65.46	65.12	64.78	64.46
0.7	65.99	**66.01**	65.64	65.30	64.92	64.46
0.8	65.99	**66.01**	65.64	65.30	64.92	64.46
0.9	65.99	**66.01**	65.64	65.30	64.92	64.46
1	65.99	**66.01**	65.64	65.30	64.92	64.46

Bold values denote the best results

7.3 Across-layer behaviors using other distribution distances

In this experiment, we apply Chebyshev distance (C-distance), Kullback-Leibler divergence (KL-divergence) and Jensen Shannon divergence (JS-divergence) to measure the distance between the distribution of a layer and the target distribution for ResNet and VGG on the VOC and COCO datasets. In this experiment, we verify whether these metrics have the same property as the Wasserstein distance. The results are shown in Fig. 5. These metrics approximately have the decreasing tendency across different layers. In this sense, these metrics demonstrate that deep layers have a better expression ability than shallow layers. More importantly, the decreasing tendency of the Wasserstein distance has experimental and theoretical justifications.

8 Conclusion

In this paper, we have proposed to interpret deep neural networks (DNNs) by understanding layer behaviors. With the help of the optimal transport theory, we propose a teacher-student analysis framework to study the across-layer and single-layer behaviors of a DNN. Theoretically, we prove that the across-layer and single-layer Wasserstein distances decrease along the depth and training iterations, respectively. However, a deeper layer is not always better

than a shallower layer for some samples. Based on these results, we propose an early-exit inference method to improve the multi-label classification performance. Extensive experiments justify these theoretical findings. Moreover, the proposed analytical framework can facilitate a future research to interpret DNNs.

Supplementary information The online version contains supplementary material available at (https://doi.org/10.1007/s10994-021-06074-8).

Author Contributions All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by JC and JL. The first draft of the manuscript was written by JC and MT and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding This work was partially supported by the Key-Area Research and Development Program of Guangdong Province (2018B010107001), Ministry of Science and Technology Foundation Project 2020AAA0106900, National Natural Science Foundation of China (NSFC) 62072190, Program for Guangdong Introducing Innovative and Enterpreneurial Teams 2017ZT07X183.

Data availability We conduct experiments on public datasets which are available on the official website. Our source code is available on https://github.com/SCUTjinchengli/LayerOT.

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose.

References

Alain G, Bengio Y (2016) Understanding intermediate layers using linear classifier probes. arXiv preprint arXiv:1610.01644

Bang S, Xie P, Wu W, Xing E (2019) Explaining a black-box using deep variational information bottleneck approach. arXiv preprint arXiv:1902.06918

Bau D, Zhou B, Khosla A, Oliva A, Torralba A (2017) Network dissection: Quantifying interpretability of deep visual representations. In: IEEE Conference on computer vision and pattern recognition, pp 6541–6549

Bjorck N, Gomes CP, Selman B, Weinberger KQ (2018) Understanding batch normalization. In: Advances in neural information processing systems, pp 7694–7705

Brock A, Donahue J, Simonyan K (2019) Large scale GAN training for high fidelity natural image synthesis. In: International conference on learning representations.

Chen CFR, Fan Q, Mallinar N, Sercu T, Feris R (2019a) Big-little net: An efficient multi-scale feature representation for visual and speech recognition. In: International conference on learning representations.

Chen, Z., Deng, L., Li, G., Sun, J., Hu, X., Liang, L., et al. (2020). Effective and efficient batch normalization using a few uncorrelated data for statistics estimation. *IEEE Transactions on Neural Networks and Learning Systems, 32*(1), 348–362.

Chen ZM, Wei XS, Jin X, Guo Y (2019b) Multi-label image recognition with joint class-aware map disentangling and label correlation embedding. In: IEEE international conference on multimedia and expo, pp 622–627

Chen ZM, Wei XS, Wang P, Guo Y (2019c) Multi-label image recognition with graph convolutional networks. In: IEEE conference on computer vision and pattern recognition, pp 5177–5186

Cuturi, M. (2013). Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in Neural Information Processing Systems, 26,* 2292–2300.

Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: IEEE conference on computer vision and pattern recognition, pp. 886–893

Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L (2009) Imagenet: A large-scale hierarchical image database. In: IEEE conference on computer vision and pattern recognition, pp 248–255

Dosovitskiy A, Brox T (2016) Inverting visual representations with convolutional networks. In: IEEE conference on computer vision and pattern recognition, pp. 4829–4837

Durand T, Mehrasa N, Mori G (2019) Learning a deep convnet for multi-label classification with partial labels. In: IEEE conference on computer vision and pattern recognition, pp. 647–657

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International Journal of Computer Vision, 88*(2), 303–338.

Fang, X., Bai, H., Guo, Z., Shen, B., & Xu, Z. (2020). Dart: Domain-adversarial residual-transfer networks for unsupervised cross-domain image classification. *Neural Networks, 127,* 182–192.

Frogner C, Zhang C, Mobahi H, Araya M, Poggio TA (2015) Learning with a wasserstein loss. In: Advances in neural information processing systems, pp 2053–2061

Genevay A, Peyré G, Cuturi M (2018) Learning generative models with sinkhorn divergences. In: Artificial intelligence and statistics

Geng, X. (2016). Label distribution learning. *IEEE Transactions on Knowledge and Data Engineering, 28*(7), 1734–1748.

Goldt, S., Advani, M. S., Saxe, A. M., Krzakala, F., & Zdeborová, L. (2020). Dynamics of stochastic gradient descent for two-layer neural networks in the teacher-student setup. *Journal of Statistical Mechanics: Theory and Experiment, 12,* 124010.

Guo, Y., Chen, J., Du, Q., Hengel, A. V. D., Shi, Q., & Tan, M. (2020). Multi-way backpropagation for training compact deep neural networks. *Neural Networks, 126,* 250–261.

Gupta P, Schütze H (2018) Lisa: Explaining recurrent neural network judgments via layer-wise semantic accumulation and example to pattern transformation. In: Empirical methods in natural language processing workshop BlackboxNLP.

He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: IEEE conference on computer vision and pattern recognition, pp 770–778

Hussain, S., Anees, A., Das, A., Nguyen, B. P., Marzuki, M., Lin, S., et al. (2020). High-content image generation for drug discovery using generative adversarial networks. *Neural Networks, 132,* 353–363.

Ioffe S, Szegedy C (2015) Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning

Kaya, Y., Hong, S., & Dumitras, T. (2019). Shallow-deep networks: Understanding and mitigating network overthinking. *International Conference on Machine Learning, 97,* 3301–3310.

Knight, P. A. (2008). The sinkhorn-knopp algorithm: Convergence and applications. *SIAM Journal on Matrix Analysis and Applications, 30*(1), 261–275.

Lee CY, Xie S, Gallagher P, Zhang Z, Tu Z (2015) Deeply-supervised nets. In: Artificial intelligence and statistics, pp 562–570

Lee, H., Grosse, R., Ranganath, R., & Ng, A. Y. (2011). Unsupervised learning of hierarchical representations with convolutional deep belief networks. *Communications of the ACM, 54*(10), 95–103.

Lee, H., Ge, R., Ma, T., Risteski, A., & Arora, S. (2017). On the ability of neural nets to express distributions. *Proceedings of the Conference on Learning Theory, 65,* 1271–1296.

Li, W., Xiong, W., Liao, H., Huo, J., & Luo, J. (2020). Carigan: Caricature generation through weakly paired adversarial learning. *Neural Networks, 132,* 66–74.

Lowe DG (1999) Object recognition from local scale-invariant features. In: International conference on computer vision

Maas AL, Hannun AY, Ng AY (2013) Rectifier nonlinearities improve neural network acoustic models. In: International conference on machine learning

Mahendran A, Vedaldi A (2015) Understanding deep image representations by inverting them. In: IEEE conference on computer vision and pattern recognition, pp 5188–5196

Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J (2013) Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems, pp 3111–3119

Montavon, G., Braun, M. L., & Müller, K. R. (2011). Kernel analysis of deep networks. *Journal of Machine Learning Research, 12*(9), 2563–2581.

Papernot N, McDaniel P (2018) Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. arXiv preprint arXiv:1803.04765

Peyré, G., Cuturi, M., et al. (2019). Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning, 11*(5), 355–607.

Raghu M, Gilmer J, Yosinski J, Sohl-Dickstein J (2017) Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. In: Advances in neural information processing systems, pp 6076–6085

Santurkar S, Tsipras D, Ilyas A, Madry A (2018) How does batch normalization help optimization? In: Advances in neural information processing systems, pp 2483–2493

Saxe AM, Bansal Y, Dapello J, Advani M, Kolchinsky A, Tracey BD, Cox DD (2018) On the information bottleneck theory of deep learning. In: International conference on learning representations.

Scardapane S, Scarpiniti M, Baccarelli E, Uncini A (2020) Why should we add early exits to neural networks? arXiv preprint arXiv:2004.12814

Shi, W., Gong, Y., Tao, X., & Zheng, N. (2018). Training dcnn by combining max-margin, max-correlation objectives, and correntropy loss for multilabel image classification. *IEEE Transactions on Neural Networks and Learning Systems, 29*(7), 2896–2908.

Simonyan K, Zisserman A (2015) Very deep convolutional networks for large-scale image recognition. In: International conference on learning representations.

Sonoda, S., & Murata, N. (2019). Transport analysis of infinitely deep neural network. *The Journal of Machine Learning Research, 20*(1), 31–82.

Sun, J., Zhong, G., Chen, Y., Liu, Y., & Huang, K. (2019). Generative adversarial networks with mixture of t-distributions noise for diverse image generation. *Neural Networks, 122,* 374–381.

Szegedy C, Zaremba W, Sutskever I, Bruna J, Erhan D, Goodfellow I, Fergus R (2014) Intriguing properties of neural networks. In: International conference on learning representations.

Tian Y (2017) An analytical formula of population gradient for two-layered relu network and its applications in convergence and critical point analysis. In: International conference on machine learning.

Tishby N, Zaslavsky N (2015) Deep learning and the information bottleneck principle. In: IEEE Information theory workshop, pp 1–5.

Villani, C. (2008). *Optimal transport: Old and new* (Vol. 338). Berlin: Springer.

Wang J, Yang Y, Mao J, Huang Z, Huang C, Xu W (2016) Cnn-rnn: A unified framework for multi-label image classification. In: IEEE conference on computer vision and pattern recognition, pp 2285–2294.

Wang, L., Zhang, H., Yi, J., Hsieh, C. J., & Jiang, Y. (2020). Spanning attack: Reinforce black-box attacks with unlabeled data. *Machine Learning, 109*(12), 2349–2368.

Wu, S., Li, G., Deng, L., Liu, L., Wu, D., Xie, Y., & Shi, L. (2019). *l*1 -norm batch normalization for efficient training of deep neural networks. *IEEE Transactions on Neural Networks and Learning Systems, 30*(7), 2043–2051.

Ye, H. J., Sheng, X. R., & Zhan, D. C. (2020). Few-shot learning with adaptively initialized task optimizer: A practical meta-learning approach. *Machine Learning, 109*(3), 643–664.

Yeh CK, Chen J, Yu C, Yu D (2019) Unsupervised speech recognition via segmental empirical output distribution matching. In: International conference on learning representations.

Yosinski J, Clune J, Nguyen A, Fuchs T, Lipson H (2015) Understanding neural networks through deep visualization. In: International conference on machine learning workshop.

Zeiler MD, Fergus R (2014) Visualizing and understanding convolutional networks. In: European conference on computer vision, pp 818–833.

Zhang Q, Cao R, Shi F, Wu YN, Zhu SC (2018) Interpreting cnn knowledge via an explanatory graph. In: AAAI conference on artificial intelligence.

Zhang Q, Yang Y, Ma H, Wu YN (2019) Interpreting cnns via decision trees. In: IEEE conference on computer vision and pattern recognition, pp 6261–6270.

Zou, D., Cao, Y., Zhou, D., & Gu, Q. (2020). Gradient descent optimizes over-parameterized deep relu networks. *Machine Learning, 109*(3), 467–492.