

Information-Theoretic Greedy Layer-wise Training for Traffic Sign Recognition

Shuyan Lyu^{1†}, Zhanzimo Wu^{2†}, Junliang Du^{3*}

¹School of Software, Taiyuan University of Technology, Taiyuan, 100190, Shanxi, China.

²Department of Mathematics and Statistical Science, University College London, London, WC1E 6BT, United Kingdom.

³*AI Institute, Shanghai Jiao Tong University, Shanghai, 200240, China.

*Corresponding author(s). E-mail(s): jldu@acm.org;

Contributing authors: 2024005980@link.tyut.edu.cn; zcahwuf@ucl.ac.uk;

†These authors contributed equally to this work.

Abstract

Modern deep neural networks (DNNs) are typically trained with a global cross-entropy loss in a supervised end-to-end manner: neurons need to store their outgoing weights; training alternates between a forward pass (computation) and a top-down backward pass (learning) which is biologically implausible. Alternatively, greedy layer-wise training eliminates the need for cross-entropy loss and backpropagation. By avoiding the computation of intermediate gradients and the storage of intermediate outputs, it reduces memory usage and helps mitigate issues such as vanishing or exploding gradients. However, most existing layer-wise training approaches have been evaluated only on relatively small datasets with simple deep architectures. In this paper, we first systematically analyze the training dynamics of popular convolutional neural networks (CNNs) trained by stochastic gradient descent (SGD) through an information-theoretic lens. Our findings reveal that networks converge layer-by-layer from bottom to top and that the flow of information adheres to a Markov information bottleneck principle. Building on these observations, we propose a novel layer-wise training approach based on the recently developed deterministic information bottleneck (DIB) and the matrix-based Rényi's α -order entropy functional. Specifically, each layer is trained jointly with an auxiliary classifier that connects directly to the output layer, enabling the learning of minimal sufficient task-relevant representations. We empirically validate the effectiveness of our training procedure on CIFAR-10

and CIFAR-100 using modern deep CNNs and further demonstrate its applicability to a practical task involving traffic sign recognition. Our approach not only outperforms existing layer-wise training baselines but also achieves performance comparable to SGD.

Keywords: Layer-wise Training, Information Bottleneck, Traffic Sign Recognition

1 Introduction

End-to-end back-propagation (E2EBP) (see Fig. 1(a)) by stochastic gradient descent (SGD) or its variants such as ADAM [1] is a dominating strategy to train modern deep neural networks (DNNs). Typically, the gradients of the cross-entropy loss with respect to the network weights are computed at the final layer and propagated backward through the network, layer by layer, to update the weights. However, E2EBP suffers from several limitations. First, it lacks biological plausibility, as the human brain does not appear to have a centralized mechanism for computing and distributing a global error signal across all neurons [2]. Second, it is prone to issues such as vanishing or exploding gradients [3, 4]. Additionally, backpropagation requires all hidden layer weights to be retained in memory until the forward and backward passes are complete, leading to significant memory usage. This backward locking phenomenon prevents memory reuse, which poses a critical limitation for resource-constrained or on-device settings [5].

To mitigate the above limitations, there have been several attempts to design layer-wise learning approaches without a global cross-entropy loss or backpropagation. Specifically, a DNN is first divided into several gradient-isolated layers or modules and is then trained successively based on local supervision signals [5]. Well-known examples of local supervision signals include target label information [6], synthetic gradients [7], and information-theoretic metrics [8]. The layer-wise training approach is more computationally efficient than E2EBP and mitigates the risk of gradient vanishing or exploding, as forward and backward propagation are performed only within a single layer or module. Furthermore, layer-wise training is considered more biologically plausible, reflecting the modular nature of the brain, which predominantly learns from local signals [9].

Previous studies on layer-wise training can be broadly categorized into the following approaches: Target Propagation [10–12], Synthetic Gradients [7, 13, 14], and Proxy Objective [6, 15–17]. Neither target propagation-based nor synthetic gradients-based approaches are scalable to challenging benchmark datasets or modern deep models, such as VGG [18] and ResNet [19]. In contrast, the proxy objective achieves the best empirical performance among all alternatives to E2EBP training. A significant class of proxy objective-based methods directly integrates target labels into the design of a local objective for latent representations, encouraging the latent outputs to closely align with the target labels, as illustrated in Fig. 1(b). However, these local proxy objectives often lack robust theoretical or empirical validation. Furthermore, their performance is typically assessed only on small-scale benchmark datasets (e.g.,

MNIST) using simple deep architectures (e.g., fully connected neural networks). No prior approach can yet match the performance of E2EBP, making them less practical in comparison [20]. Consequently, their practical effectiveness on real-world applications involving more complex deep architectures remains unclear.

In this work, we first conduct a systematic analysis of the learning dynamics of DNNs trained with E2EBP through an information-theoretic lens. Our analysis demonstrates that DNNs converge progressively from shallower layers to deeper layers and that the information flow within feed-forward deep architectures follows the Markov information bottleneck principle [21], as discussed in detail in Section 3. These observations imply that even when a DNN is trained using E2EBP, its internal training dynamics inherently follow a layer-wise progression. In other words, E2EBP implicitly performs layer-wise training. This insight helps bridge the conceptual gap between E2EBP and explicit layer-wise training approaches. Building on our observations, we propose a novel greedy layer-wise training approach inspired by the recently developed deterministic Information Bottleneck (DIB) [22, 23]. We evaluate its performance using ResNet on CIFAR-100 and further demonstrate its applicability by extending the approach to a real-world task of traffic sign detection.

To summarize, our main contributions include:

- We conduct a systematic analysis to bridge the conceptual gap between E2EBP and explicit layer-wise training. Our empirical observations also reveal a more promising local proxy objective that differs from previous proposals by explicitly introducing an information compression term.
- Based on our observations, we propose a novel greedy layer-wise training approach called Greedy Deterministic Information Bottleneck (Greedy-DIB). Our Greedy-DIB outperforms existing state-of-the-art (SOTA) layer-wise training approaches and achieves performance on par with SGD. Note that, previous studies typically evaluated these approaches on benchmark classification tasks (e.g., MNIST) using MLPs, while our Greedy-DIB demonstrates strong generalization on CIFAR-100 with ResNet-18.
- To further demonstrate the effectiveness of our approach, we also test its performance on a real-world traffic sign recognition task, which involves both object classification and bounding box regression (i.e., a two-head neural architecture). To the best of our knowledge, we are among the first to apply a local loss-based CNN for traffic sign recognition.

2 Related Work

2.1 Layer-wise Training with Proxy objective

Recently, proxy objective methods have become increasingly popular. Greedy LS [6] and Greedy LE [15] add an auxiliary model to each single layer during the training. Specifically, Greedy LS utilizes only the cross-entropy loss between the output of the auxiliary model and labels. In contrast, Greedy LE combines cross-entropy loss with a cosine similarity regularization between the output of the auxiliary model and

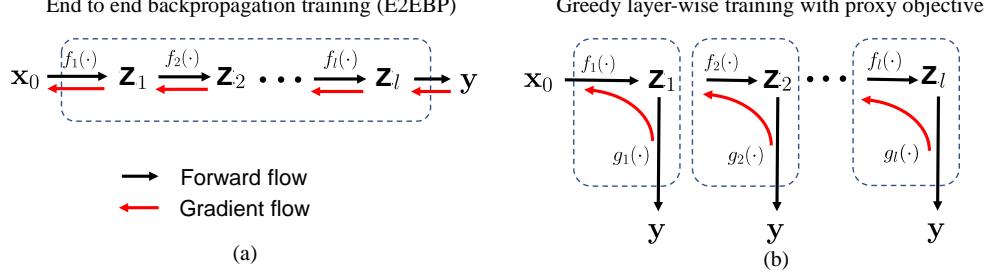


Fig. 1 Diagram of (a) E2EBP and (b) greedy layer-wise training with a local proxy objective. $f_l(\cdot)$ represents the l -th layer of the whole model, $g_l(\cdot)$ denotes an auxiliary classifier. $g_l(\cdot)$ was trained together with $f_l(\cdot)$ to minimize a local proxy objective and was discarded at test time.

the label. The objective of Neural Indicator Kernel (NIK) [16] is essentially the normalized Hilbert-Schmidt Independence Criterion (nHSIC) [24], which measures the dependency between the latent representation and the label without relying on any auxiliary models. However, these objectives are also prone to learning redundant information in each layer, as the designed learning objective includes only a prediction term that evaluates the latent representation’s predictive power with respect to the label. As a result, the learned representation often captures noisy or redundant information from the input X that is irrelevant to downstream tasks. This partially explains why no prior approach has been able to match the performance of E2EBP, rendering them less practical.

The most relevant work to ours is the HSIC bottleneck [8], which also employs the Information Bottleneck (IB) principle [25] to train deep networks without relying on error backpropagation. Unlike the HSIC bottleneck, our objective is based on the deterministic IB framework [22], which, as we will demonstrate in later sections, offers more theoretical justifications. Additionally, instead of utilizing nHSIC, we adopt the recently proposed matrix-based Rényi’s α -order entropy functional [26, 27], which simplifies the estimation of the mutual information term. Moreover, our approach significantly outperforms the HSIC Bottleneck, as will be demonstrated in Section 5. We also observed that [17] utilizes kernel alignment [28] as a proxy objective. However, its effectiveness has only been evaluated on kernel networks, a specialized type of neural network where neurons are replaced with kernel machines. The objective function for the above approaches and our proposed method are shown in Table 1.

2.2 Understanding Training Dynamics of Deep Neural Networks

Although deep neural networks have achieved great success, interpreting and understanding their training dynamics remain a challenge. Recently, information-theoretic tools have attracted much attention in understanding DNNs. One particular concept is the so-called information plane [29], which depicts the trajectory in \mathbb{R}^2 of the mutual information pair $\{I(X; Z), I(Y; Z)\}$ across training epochs, as a lens to

Table 1 Existing layer-wise training approaches and their corresponding objective in the l -th layer. x refers to the input and z_l refers to the l -th hidden layer representation, y is the class label. $g(\cdot)$ represents the auxiliary classifier and $S(\cdot)$ represents cosine similarity.

Method	Objective function
Greedy LS [6]	$\text{CE}(g(z_l); y)$
Greedy LE [15]	$(1 - \beta)\text{CE}(g(z_l); y) + \beta(S(g(z_l)) - S(y) _F^2)$
HSIC bottleneck [8]	$n\text{HSIC}(z_l; x) - \beta n\text{HSIC}(z_l; y)$
NIK [16]	$-n\text{HSIC}(z_l; y)$
Ours	$\beta H(z_l) + \text{CE}(g(z_l); y)$

analyze dynamics of learning of DNNs [30]. According to [29], there are two training phases in the common stochastic gradient descent (SGD) optimization: an early “fitting” phase, in which both $I(X; Z)$ and $I(Y; Z)$ increase rapidly, and a later “compression” phase, in which there is a reversal such that $I(X; Z)$ continually decrease. This work attracted significant attention, culminating in many follow-up works that tested the proclaimed narrative and its accompanying empirical observations. So far, the “fitting-and-compression” phenomena of the layered representation Z have been observed in other types of DNNs, including the multilayer perceptrons (e.g., [29, 31]), the deterministic autoencoders (e.g., [32]), and the CNNs (e.g., [33, 34]).

However, previous literature defines the x -axis of the information plane as $I(X; Z)$. In contrast, we define the x -axis of the information plane as the mutual information between the hidden layer representation and its preceding layer, i.e., $I(Z_i; Z_{i-1})$. This modification aligns with the essence of layer-wise training, as it reflects the progressive nature of network training, where information is processed and refined layer by layer in a sequential manner.

Another research direction focuses on studying the loss landscapes [35] during training. A common concept in this area is the so-called sharpness of local minima, which can be quantified using first-order (e.g., Jacobian, Lipschitz constant) or second-order (e.g., Hessian spectrum) sensitivity measures. [36] observed that DNNs generalize well when they converge to flat minima. [37] further improves the generalizability by encouraging the neural networks to find flat minima through the stochastic weight averaging densely. However, the existing sharpness-based metrics fail to provide consistent observations and can be influenced by various training techniques (e.g., adversarial training [38], ℓ_2 regularization [39]).

2.3 Traffic Sign Recognition

Traffic sign recognition holds significant industrial potential for advanced driver assistance systems and intelligent autonomous vehicles. Generally, traffic signs are categorized based on their functions, such as warning, prohibitory, or mandatory signs. Within each category, signs may be further divided into subclasses that share similar shapes and appearances but differ in specific details (e.g., speed limits of 40, 60, and 80 within the “speed limit” category). This structure suggests that traffic sign recognition should be approached as a two-phase task: detection followed by classification [40]. The detection phase identifies bounding boxes in an image that are likely to contain traffic signs, while the classification phase determines the exact type of

sign present (if any). However, several challenges hinder accurate recognition by computer algorithms, including variations in illumination, color degradation, motion blur, cluttered backgrounds, and partial occlusion.

In the deep learning era, convolutional neural networks (CNNs) have been extensively applied to traffic sign detection and classification. [40] proposed two fully convolutional networks, one for traffic sign detection and the other for simultaneous detection and classification, with both networks sharing the same architecture except for the branches in the final layer. [41] utilized the Single Shot Multibox Detector (SSD) framework with VGG-16 as the backbone for traffic sign detection. [42] explored a lightweight MobileNet to balance memory efficiency and accuracy. In recent years, the YOLO family of models has gained popularity in this domain. YOLO [43], as an initiator, introduced a fully connected layer to directly predict both the object class and bounding box. [44] proposed an end-to-end framework based on YOLOv3, while [45] adopted YOLOv5 for this task.

All these approaches focus on modifying network architectures but overlook the potential of alternative training algorithms. In this paper, we aim to demonstrate that a greedy layer-wise training approach offers a promising alternative to traditional back-propagation, making it particularly suitable for vehicle systems with limited memory resources.

3 Analyzing End-to-End Backpropagation through an Information-Theoretic Lens

In this section, we analyze the learning dynamics of end-to-end training from an information-theoretic perspective. In Section 3.1, We design and visualize the changes in two mutual information matrices over the entire training epochs to demonstrate that, even when a network is trained using E2EBP, its learning dynamics implicitly follow a layer-wise training pattern. Specifically, the convergence of layer representations begins with the shallower layers and gradually progresses to the deeper layers. In Section 3.2, we introduce a modification to the original information plane proposed in [29] by replacing the mutual information $I(x; z)$ with the entropy of the latent representation $H(z)$ on the x -axis. Our goal is to demonstrate that the latent representation undergoes both a fitting phase and an information compression phase. Previous literature on layer-wise training only models the fitting phase (see Table 1), while neglecting the modeling of information compression.

3.1 Understanding Learning Dynamics by Mutual Information Matrices

3.1.1 Cross-Mutual Information Matrix

We first analyze the learning dynamics by designing a cross-mutual information matrix between a network under training (denoted by \mathcal{S}) and a reference network (denoted by \mathcal{T}). Here, the training network refers to the network during the training whose parameters are updated in each epoch, while the reference network refers to the final learned network at the end of training in which the parameters are fixed. We consider

the entire network F as the composition of L layers, where $F = f_1 \circ \dots \circ f_L$. Given an input $X \in \mathbb{R}^{n \times d}$ with n samples and d -dimensional features, the intermediate representation in the i -th layer can be denoted as $\mathbf{z}_i \in \mathbb{R}^{n \times d_i}$ with d_i dimension, where $\mathbf{z}_i = f_i(\mathbf{z}_{i-1})$.

We compute the mutual information value between the i -th layer representation in \mathcal{S} at training epoch t (denoted by \mathbf{z}_i^t) and the j -th layer in \mathcal{T} (denoted by \mathbf{z}_j^T). This procedure results in a cross-mutual information matrix G^t , the size of which is $L \times L$ and the (i, j) -th entry is $G_{i,j}^t = I(\mathbf{z}_i^t; \mathbf{z}_j^T)$. Intuitively, G^t quantifies the shared information between all layer representations at epoch t and those at the end of training. In other words, G^t measures how similar or dependent the network is after t -epochs of training compared to the fully trained network. A high value in G^t implies that the network after t -epoch of training is highly similar or relevant to the final trained network; whereas a low value indicates dissimilarity. For clarity, we illustrate this procedure in Figure 2, where we assume that both \mathcal{S} (top left figure) and \mathcal{T} (top right figure) consist of 4 layers. The cross-MI matrices are evaluated at training epochs t_1, t_2 , and at the end of the training.

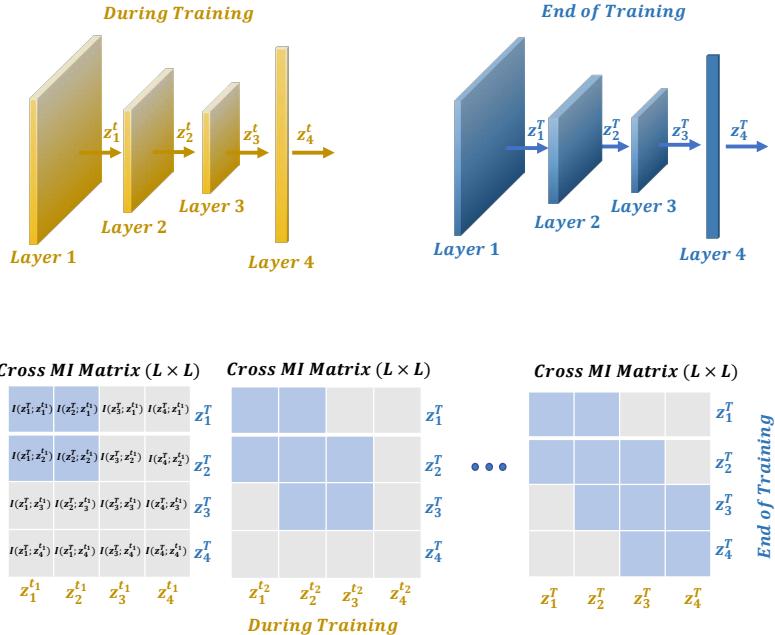
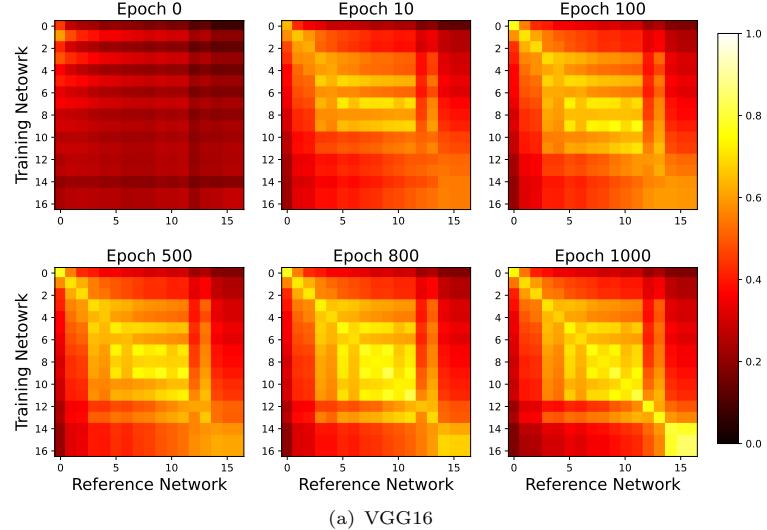
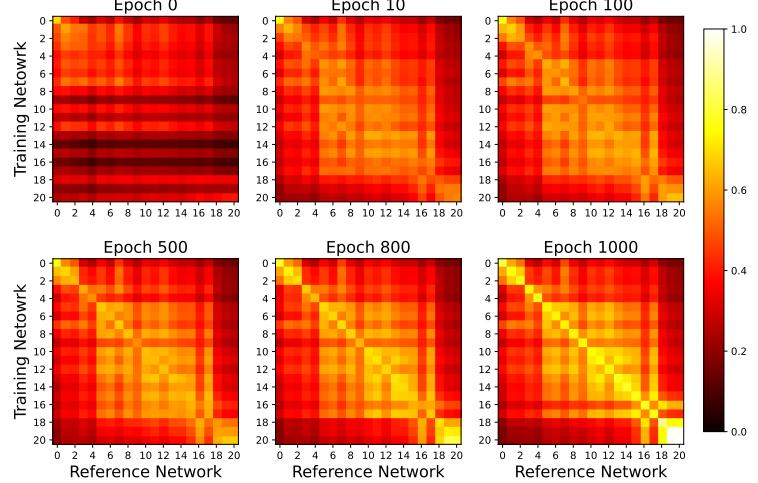


Fig. 2 Illustrating E2EBP learning dynamic based on the cross MI matrix. We assume both \mathcal{S} and \mathcal{T} have 4 layers for simplicity.

Throughout this paper, we measure mutual information values with the matrix-based Matrix-based Rényi's α -order entropy functional [26, 27] (see Appendix A for more details), which is computationally efficient and scalable to high-dimensional variables.



(a) VGG16



(b) ResNet18

Fig. 3 Cross-mutual information matrix of VGG16 and ResNet18 trained on CIFAR-10. Each pane is a $L \times L$ matrix, the x-axis denotes all layers' output for reference network and y-axis represents all layers' output for training network, and each entry of the matrix represents the mutual information between two layers. The larger value means high dependency.

The cross-mutual information matrices of VGG-16 and ResNet18 trained on CIFAR-10 for a total of 1,000 epochs with SGD are shown in Figure 3. As can be seen, the values of the diagonal elements in shallow layers rapidly converge to large values within a short period of time, while the diagonal elements in deeper layers increase more gradually over a longer training period. These figures suggest that the end-to-end training of deep neural networks progresses sequentially, starting from the shallow

layers and moving toward the deep layers. Our observations are consistent with [46], which relies on Canonical Correlation Analysis (CCA), a method that quantifies only linear dependence.

3.1.2 Auto-Mutual Information Matrix

We propose an alternative method to analyze the learning dynamics of E2EBP by visualizing the trajectory of different layers' outputs in a 2-D plane. Unlike the cross-mutual information matrix, we directly construct a pairwise distance matrix D^t , with size $L \times L$, using an auto-mutual information matrix. The (i, j) -th entry of this matrix is given by $I(z_i^t, z_j^t)$ rather than $I(z_i^t, z_j^T)$. Intuitively, the auto-mutual information matrix captures all pairwise dependencies between the layer representations at training epoch t .

The distance between two layers' output at the t -th epoch $d(z_i^t; z_j^t)$ is defined as:

$$D_{i,j}^t = 1 - \frac{I(z_i^t; z_j^t)}{\max\{H(z_i^t), H(z_j^t)\}}, \quad (1)$$

where H denotes the entropy, and the second term is the normalized mutual information, which ranges from 0 to 1. $D_{i,j}^t$ is a distance metric, as demonstrated in [47, 48]. We then project the pairwise distance matrix D^t into a 2-D plane using multidimensional scaling (MDS). The coordinates of different layers reflect their pairwise distances. This allows us to continuously observe the trajectories of different layers throughout the entire training process. An illustrative figure of this procedure is demonstrated in Figure 4.

We plot trajectories of six layers from shallow (2-nd and 4-th), middle (10-th and 11-th), to deep (15-th and 16-th) layer of VGG-16 architecture during the end-to-end training in Figure 5. We observe that the coordinates of the shallow and middle layers do not change significantly during training, whereas the coordinates of the deeper layers show substantial movement. This observation indicates that the shallow and middle layers are trained much more quickly than the deeper layers, which further supports our claim that, even when a network is trained with E2EBP, its internal learning dynamics still follow a layer-wise convergence pattern.

3.2 Modified Information plane

In this section, we use the information plane to visualize $I(z_{l-1}; z_l)$ (the mutual information between the output of the l -th layer, z_l , and the output of the previous layer, z_{l-1}) and $I(z_l; y)$ (the mutual information between the current layer's output, z_l , and the target y) for VGG16 and ResNet10 trained on CIFAR-10 across 1,000 training epochs.

For a deterministic neural network such as VGG16, we can replace $I(z_{l-1}; z_l)$ with $H(z_l)$ ¹ and plot $H(z_l)$ with respect to $I(z_l; y)$ in Figure 6. We observe that $I(z_l; y)$ continues to increase during the whole training epochs while $I(z_{l-1}; z_l)$ increases at

¹ $I(z_{l-1}; z_l) = H(z_l) - H(z_l|z_{l-1})$ and $H(z_l|z_{l-1}) = 0$ because there is no uncertainty in the mapping from $(l-1)$ -th layer to the l -th layer [22].

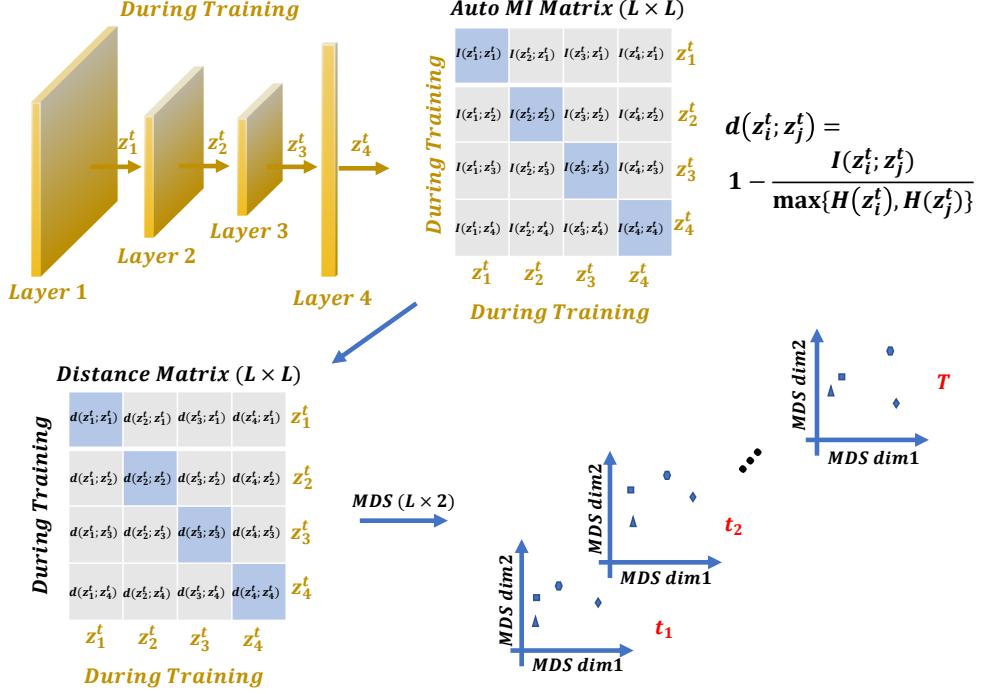


Fig. 4 Illustrating layer's trajectory of E2EBP during training with MDS based on the auto-MI matrix.

the beginning and followed by an obvious decrease until convergence. The increase of $I(z_l; y)$ is expected from the cross-entropy loss minimization. The trajectory of $H(z_l)$ indicates the fitting (information increasing) and compression phase (information decreasing) during the E2EBP. Hence, when designing a local learning objective for each layer, the objective should include an information compression term to explicitly model this phenomenon.

Summarization of empirical observations on E2EBP: The experiments above reveal that DNNs converge sequentially from the shallower layers to the deeper layers, even when trained using E2EBP. Moreover, the training of each layer within DNNs under the traditional E2EBP adheres to the IB principle, where the hidden layer representations first increase and then decrease, effectively removing redundant or noisy information from the input X that does not contribute to classifying the label Y . Based on these observations, we consider whether it is possible to explicitly train a DNN from shallow layers to deeper layers in a manner that follows exactly the IB principle. We shall give an answer in the next section.

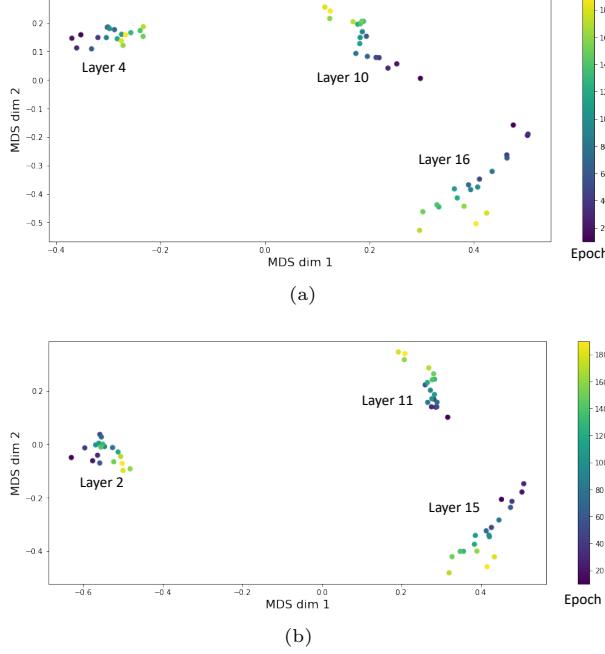


Fig. 5 VGG16 learning dynamic with MDS measurement for shallow, middle and deep layer. (a) Shallow 4 layer, middle 10 layer and deep 16 layer. (b) Shallow 2 layer, middle 11 layer and deep 15 layer.

4 Layer-wise Training with Deterministic Information Bottleneck

We utilize the standard convolutional and fully connected model structures, but each layer is trained by a local learning signal motivated by deterministic information bottleneck, instead of globally back-propagating errors. According to Figure 6, we observe that there exists a compression phase during which $H(z_l)$ decreases, while the $I(z_l; y)$ continue increases until the end of training. This observation motivates us to design a new layer-wise training method such that the mutual information $I(z_l; y)$ is maximally preserved whereas the redundant information in z_l is dropped out by minimizing $H(z_l)$.

4.1 Deterministic Information Bottleneck (DIB)

Let X be an input signal, Z represents the latent representation of X , and Y refers to the corresponding label. Given the joint distribution $p(x, y)$, the original IB objective [25] seeks to find an encoding distribution $q(z|x)$ through the following optimization problem:

$$\min_{q(z|x)} \mathcal{L}_{\text{IB}}[Z] = -I(Z; Y) + \beta I(Z; X), \quad (2)$$

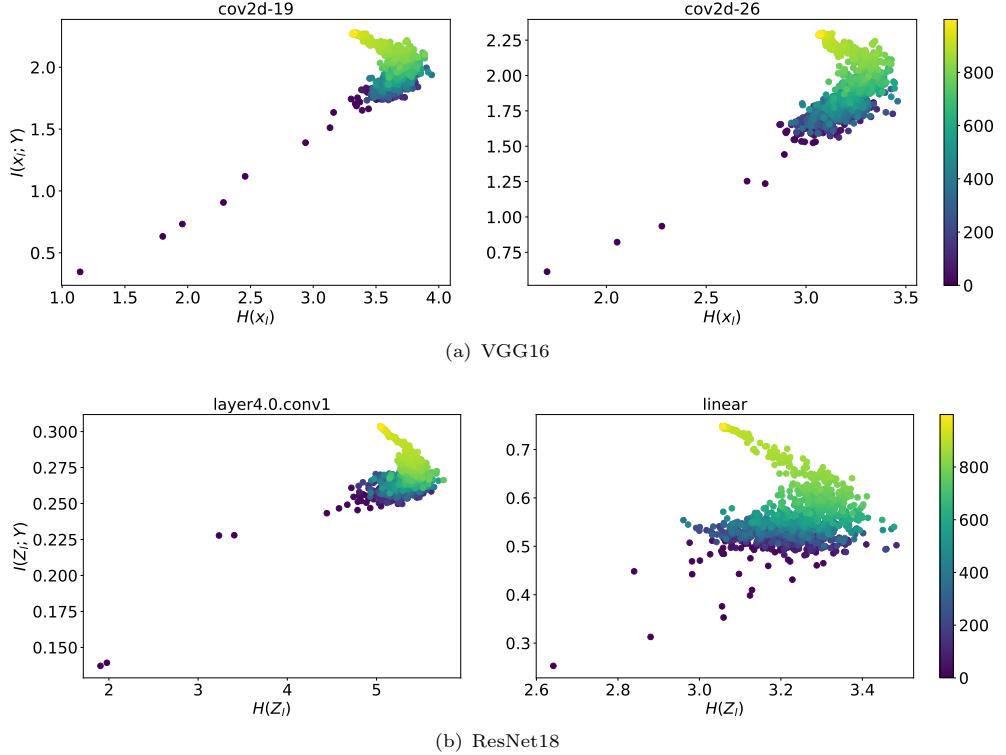


Fig. 6 Information plane of VGG16 and ResNet18 on CIFAR-10 with different convolution and linear layers (*i.e.*, shallow layer: “cov2d-19”, “layer4.0.conv1”, deep layer: “cov2d-26”, “linear”). Different colors correspond to the number of training epochs.

in which $I(Z; Y)$ characterizes the *sufficiency* of Z to address a given task, whereas $I(Z; X)$ characterizes the *minimality* of Z with respect to X . β is a hyperparameter that balances the trade-off between $I(Z; Y)$ and $I(Z; X)$. In this sense, IB can be interpreted to learn a minimum sufficient representation.

For a deterministic neural network, we could replace the mutual information $I(Z; X)$ with entropy $H(Z)$. This is because $I(Z; X) = H(Z) - H(Z|X)$ and $H(Z|X) = 0$ [49, 50]. Therefore, the IB objective naturally reduces to the deterministic IB objective [22]:

$$\min_{q(z|x)} \mathcal{L}_{\text{DIB}}[Z] = -I(Z; Y) + \beta H(Z). \quad (3)$$

Additionally, we have $-I(Z; Y) = -H(Y) + H(Y|Z)$. Therefore, minimizing $-I(Z; Y)$ is equivalent to minimizing $H(Y|Z)$, as the entropy $H(Y)$ of the ground truth labels Y is a constant and does not depend on the network parameters.

Given the training set $\{x_i, y_i\}_{i=1}^N$, let $p_\theta(z|x)$ and $p_\theta(y|z)$ denote the unknown distributions we aim to estimate, parameterized by θ . The term $H(Y|Z)$ can be further

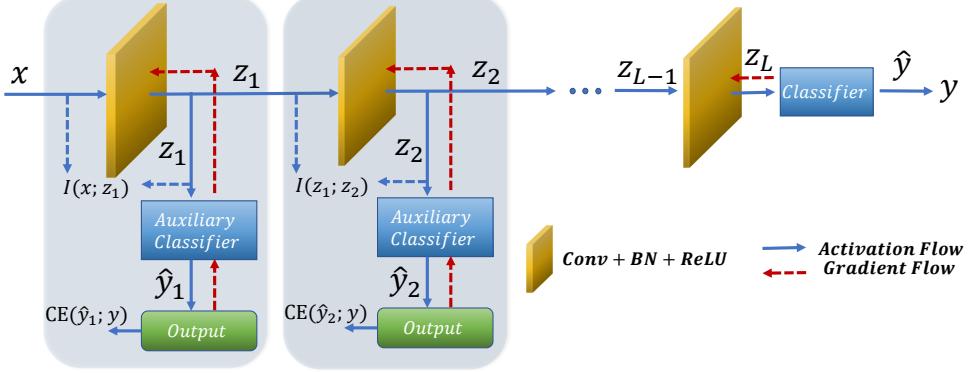


Fig. 7 Diagram of our proposed method. Each layer is jointly trained with the auxiliary classifier with cross-entropy loss related to the label, and regularized with the entropy of the layer’s output.

approximated as the empirical cross-entropy (CE) as shown in Eq. (4):

$$H(Y|Z) \simeq \mathbb{E}_{x,y \sim p(x,y)} [\mathbb{E}_{z \sim p_\theta(z|x)} [-\log p_\theta(y|z)]] = \frac{1}{N} \sum_{i=1}^N \mathbb{E}_{z \sim p(z|x_i)} [-\log p(y_i|z)], \quad (4)$$

Therefore, the DIB objective reduces to a basic cross-entropy loss, regularized by a weighted differentiable entropy term $H(Z)$, as defined in Eq. (5):

$$\mathcal{L}_{\text{DIB}} = \text{CE}(\hat{Y}; Y) + \beta H(Z). \quad (5)$$

4.2 Greedy layer-wise training with DIB

4.2.1 Architecture Formulation

Our architecture has L blocks (see Figure 7), which are trained in succession. For an input signal x , it is propagated from the first hidden layer to the l -th hidden layer, generating z_l . Each z_l feeds into an auxiliary classifier to obtain \hat{y}_l , the intermediate classification output solely based on z_l . At the l -th block, the network component is denoted as f_{θ_l} , parameterized by θ_l , and consists of standard convolutional layers followed by a batch normalization layer and a ReLU activation function. The corresponding auxiliary classifier for the l -th block is represented as g_{ϕ_l} , parameterized by ϕ_l , which can be a simple fully connected layer or a convolutional layer. Formally, given an input signal x , we iterate as follows:

$$\begin{cases} z_l = f_{\theta_l}(z_{l-1}), & l = 1, 2, \dots, L \\ \hat{y}_l = g_{\phi_l}(z_l) \in \mathbb{R}^c, \end{cases} \quad (6)$$

where c is the number of classes.

4.2.2 Training with Deterministic Information Bottleneck

Our training procedure is layer-wise. Specifically, to train the l -th layer, we only update layer parameters θ_l and corresponding auxiliary classifier parameters ϕ_l by a DIB objective, whereas all other parameters are fixed. In particular, the learning objective for the l -th hidden layer is represented as:

$$\begin{aligned}\hat{\mathcal{R}}_l(z_l; \theta_l, \phi_l) &= \beta I(z_{l-1}; z_l) - I(z_l; y) = \beta H(z_l) - I(z_l; y) \\ &= \underbrace{\beta \frac{1}{1-\alpha} \log_2 \left(\sum_{i=1}^N \lambda_i(A)^\alpha \right)}_{\textcircled{1}} + \underbrace{\frac{1}{N} \sum_{i=1}^N \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z}|\mathbf{x}_l)} [-\log p(y_i|z_l^i)]}_{\textcircled{2}}.\end{aligned}\quad (7)$$

The first term in Eq. (7) is estimated with the Rényi's α -order entropy functional (see part (1)). Specifically, given a minibatch of samples $\{x_i, y_i\}_{i=1}^B$ of size B , we obtain the hidden layer representations $\{(z_l)_i\}_{i=1}^B$ at the l -th layer and construct a kernel Gram matrix $K \in \mathbb{R}^{B \times B}$ with Gaussian kernel of kernel width σ as $K_{i,j} = \kappa((z_l)_i, (z_l)_j)$ and $\kappa = \exp(-\frac{\|(z_l)_i - (z_l)_j\|_2^2}{2\sigma^2})$. $A = K / \text{tr } K$ is the trace normalized kernel Gram matrix and $\lambda_i(A)$ denotes the i -th eigenvalue of A . The second term $I(z_l; y)$ in Eq. (7) is approximated with the cross-entropy as discussed earlier (see parts (2)). The full algorithm of our layer-wise training is presented in Algorithm 1.

Algorithm 1 Layer Wise CNN

Require: Training samples $\{x_i, y_i\}_{i=1}^N$
Ensure: $\{\theta_l\}_{l=1}^L$ and ϕ_L .

```

while stopping criterion not met do
    Sample a minibatch of  $B$  samples from the training set.
    for  $j \in 1, \dots, L$  do
         $(\theta_l^*, \phi_l^*) = \arg \min_{\theta_l, \phi_l} \hat{\mathcal{R}}_l(z_l; \theta_l, \phi_l)$ 
    end for
end while
```

The most relevant work is the HSIC bottleneck, which also employs the IB principle to train deep classification networks without error backpropagation. In addition to differences in the mutual information estimator and the IB objective, we would like to further clarify that the training method and network architecture are also distinct. Specifically, our method utilizes greedy layer-wise training, while the HSIC bottleneck adopts multi-round layer-wise training, which is more computationally complex and less memory efficient. Regarding the network architecture, HSIC bottleneck does not attach an auxiliary model to each layer's output, whereas our method leverages an auxiliary model to map the higher-dimensional latent representations to a lower dimension, making it more convenient to design and compute the proxy objective function.

5 Experiments

5.1 Implementation details

To validate the scalability of our proposed method, we evaluate it in both simple and complex architectures. For a simple CNNs experiment, we trained a model with five convolutional layers (256 filters with 3×3 kernel size) without any down-sampling layer. We selected the auxiliary classifier as only one fully connected layer, the size of which is 1024. Each convolutional block consists one convolutional layer, one batch-normalization layer, and ReLU activation layer. We train each convolutional block with an auxiliary classifier attached by an averaging pooling layer. We optimize each layer with SGD using a momentum of 0.9, weight decay of $5e - 4$, a batch-size of 256. A learning rate of 0.1 is used initially, followed decreasing by a factor of 0.2 every 15 epochs for a total of 100 epochs in each layer. For more complex architectures (i.e., VGG and ResNet), we set the auxiliary classifier model as two convolutional and one fully connected layer. Each layer is optimized with SGD using a momentum of 0.9, weight decay of $1e - 4$. We set batch-size as 128 and each layer starts with 0.1 learning rate and then decays by 0.1 every 20 epochs, for a total of 50 epochs. We use the standard data augmentation, flipping, and cropping for CIFAR-10 and CIFAR-100 datasets.

5.2 Performance Comparison and Analysis

5.2.1 Simple CNNs on CIFAR-10

In this section, we evaluated our proposed and other baseline methods with simple 5-layers CNN on the CIFAR-10 dataset. Specifically, according to their original setting, we set $\beta = 0.01$ and $\beta = 10$ for Greedy LE and HSIC bottleneck methods. To avoid hyperparameter tuning, we use a fixed value of β for different layers. We find that the performance is stable when β located between 0.001 to 0.01, thus we sweep the β in this range and found that $\beta = 0.006$ provides the best performance shown in Fig. 8(b).

Table 2 Test accuracy (%) on CIFAR-10 with 5-layer CNN. The best performance is in bold and the second best performance is underlined.

Method	Test Acc (%)
Backprop (SGD)	<u>84.63</u>
Greedy LS	83.27
Greedy LE	83.58
HSIC bottleneck	61.56
NIK	60.72
Ours	84.66

We report the results trained with 5-layer CNN in Table 2 and different number of layers in Figure 8(a). Our proposed method outperforms existing proxy objective

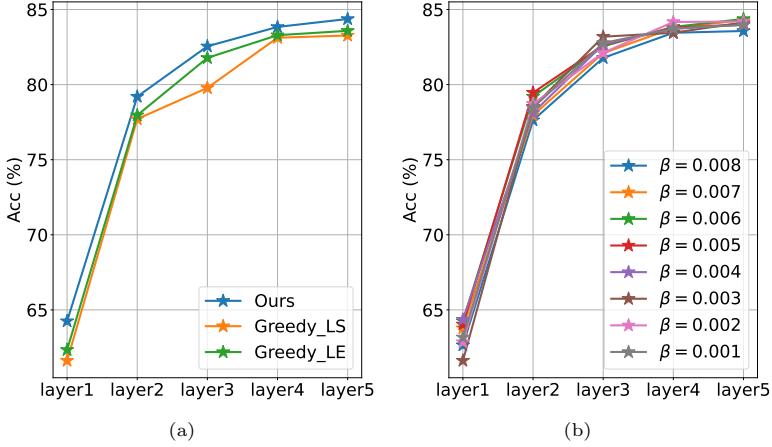


Fig. 8 (a) Accuracy with different layers on CIFAR-10 test set. (b) Proposed method results with different β .

based layer-wise training methods and is the only one that is comparable to the performance of end-to-end training method (with a difference on accuracy less than 1%). Greedy LS and Greedy LE approaches are auxiliary-based methods that achieve better performance than NIK and HSIC bottleneck. This indicates that an explicit auxiliary classifier is more effective than kernel alignment mechanisms to transfer the label information to guide the training of intermediate layers. Our proposed method achieves better results than Greedy LS and Greedy LE because we not only incorporate the label information for training each layer but also leverage the information bottleneck principle to discard the redundancy within the layers. In addition, we also illustrate the latent representation of the CIFAR-10 validation set in Figure 9 to show the effectiveness of the DIB principle. We observe that the latent representations produced by our proposed method are more separable compared to those generated without DIB.

5.2.2 Large-scale Experiments

We further evaluated our method with more complex architectures with the CIFAR-10 and CIFAR-100 datasets. The VGG-11 architecture contains 8 convolutional layers, 1 average-pooling layer, and 2 fully connected layers, while the VGG-16 architecture contains 13 convolutional layers, 1 average-pooling layer, and 2 fully connected layers. Thus, we train total 8 layer blocks for VGG-11 and total 13 layer blocks for VGG-16. We consider each layer block as one convolutional layer, batch-normalization, and ReLU activation layer. For ResNet-18 network, we consider each layer block as two convolutional layers with batch normalization and ReLU activation, and a skip connection and train total 8 layer blocks. Similar to the previous setting, each layer block is trained jointly with the auxiliary classifier. We summarize the results on the CIFAR-10 and CIFAR-100 datasets in Table 3. Our proposed method outperforms other layer-wise training methods and is the only one comparable to end-to-end training. One possible reason for the performance gap between greedy layer-wise training

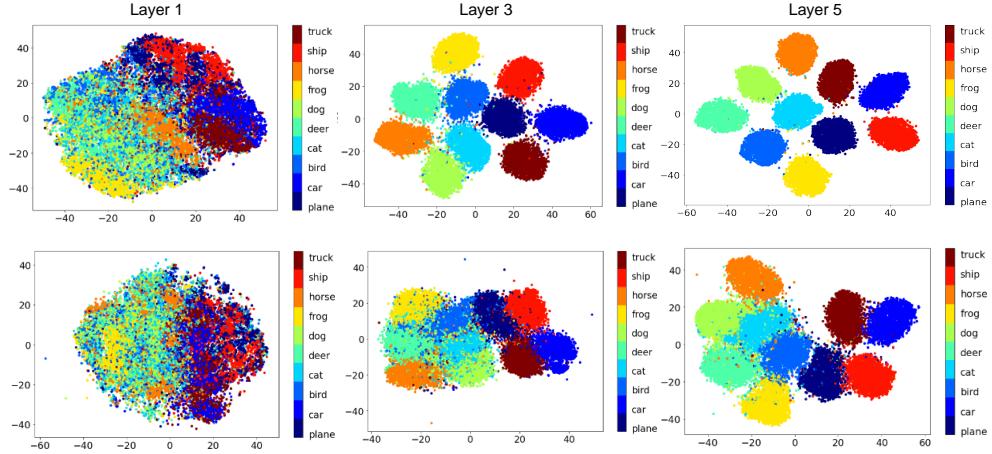


Fig. 9 The latent representation for different layer’s output on CIFAR10 validation set in 5-layer CNN. The first row represents the results for greedy layer-wise training with DIB. The second row represents results for training with only cross-entropy loss (i.e., without DIB).

and E2EBP is that the usable information for classification tends to saturate in the middle layers [20].

Table 3 Test accuracy (%) on CIFAR-10 and CIFAR-100. The best performance is in bold and the second best performance is underlined.

Methods	CIFAR-10			CIAR-100		
	VGG-11	VGG-16	ResNet-18	VGG-11	VGG-16	ResNet-18
Backprop (SGD)	90.37	92.64	93.02	68.64	72.93	75.61
Greedy LS	89.37	89.59	90.40	65.42	67.92	69.72
Greedy LE	89.23	89.56	90.26	66.14	68.22	70.03
HSIC bottleneck	64.32	68.75	68.98	44.52	47.28	53.45
NIK	63.69	65.52	67.32	43.94	46.10	52.75
Ours	<u>90.22</u>	<u>92.21</u>	<u>91.73</u>	<u>67.63</u>	<u>69.75</u>	<u>72.03</u>

5.2.3 Traffic Sign Recognition

We evaluate our approach on two datasets: the Chinese Traffic Sign Recognition Database (CTSRD)² [51] and the German Traffic Sign Recognition Benchmark (GTSRB)³ [52]. The CTSRD dataset contains 6,164 traffic sign images across 58 categories. Each image is resized to 64×64 , with 80% of the data used for training and 20% for testing. The GTSRB dataset includes 43 traffic sign classes, with 39,209 training

²<https://nlpr.ia.ac.cn/pal/trafficdata/recognition.html>

³<https://benchmark.ini.rub.de/>

images and 12,630 test images. This dataset features images captured under varying lighting conditions and with diverse, complex backgrounds. In both datasets, target objects typically occupy a significant portion of the image, with the bounding box of each object of interest averaging about 20% of the image area. The baseline CNNs used for CTSRD and GTSRB are a 5-layer fully convolutional network and VGG-16, respectively.

To tackle this task, we propose a single CNN architecture with two output heads. The first head is a bounding box regression layer, which uses a linear regression output to predict the four coordinates of the traffic sign’s bounding box. The second head is a classification layer equipped with a softmax function, which outputs the probabilities for each class, with the highest probability indicating the predicted class. This two-head design enables our network to simultaneously detect and classify traffic signs in a unified framework.

The learning objective of the l -th layer in traffic sign recognition becomes:

$$\mathcal{R}_l = \beta H(z_l) + \text{CE}(g(z_l); y) + \alpha \text{Bbox}(r(z_l); b), \quad (8)$$

where $g(\cdot)$ represents the auxiliary object classifier, and $r(\cdot)$ denotes the auxiliary bounding box regressor. Bbox represents l_1 loss, y is the ground truth category and b is the target bounding box containing 4 coordinates. Each layer is jointly trained with an auxiliary category classifier and bounding box regressor with one fully connected layer. We set $\alpha = 0.001$ for all baseline methods, and set $\beta = 0.01$ for our proposed method and Greedy LE. We test our method and baseline methods for object detection task in terms of the object detection accuracy and mean intersection over union (mIOU). Tables 4 and 5 suggest that greedy layer-wise training has the potential to exceed the performance of SGD in real-world applications. Again, our method is consistently better than Greedy LS and Greedy LE. The performance of HSIC Bottleneck and NIK is omitted due to their relatively poor results.

Table 4 Results on Chinese Traffic Sign Recognition Database (CTSRD).

Method	Classification Accuracy (%)	mIOU (%)
Backprop (SGD)	98.23	86.31
Greedy LS	98.62	87.45
Greedy LE	98.68	88.56
Ours	98.73	89.23

The authors acknowledge the existence of more challenging datasets in this domain, such as Tsinghua-Tencent 100K [40], which feature a higher number of test images captured in wild environments. In these datasets, traffic signs are typically tiny, often occupying less than 1% of the image. Adapting to such datasets requires more advanced architectures like YOLO and additional strategies. For instance, large images can be divided into smaller patches, which are then fed into a baseline network to produce patch-level object detection results. Considering that the primary focus of

Table 5 Results on German Traffic Sign Recognition Benchmark (GTSRB). The performance in terms of mIOU is omitted due to the absence of ground truth for bounding boxes.

Method	Classification Accuracy (%)
Backprop (SGD)	97.38
Greedy LS	97.32
Greedy LE	97.48
Ours	97.88

this paper is on greedy layer-wise training, and the mainstream literature in this area primarily evaluates simple architectures on benchmark datasets like MNIST or CIFAR-10, we believe our evaluations on CTSRD and GTSRB, along with the extension to a two-head CNN architecture, represent a significant advancement in this field. Extending our approach to YOLOv3 on the Tsinghua-Tencent 100K is planned as future work.

6 Conclusions

In this work, we systematically observed the flow of information inside popular convolutional neural networks by virtue of two mutual information matrices and a modified information plane. We then proposed a new strategy for greedy layer-wise training by incorporating a deterministic information bottleneck (DIB) coupled with the matrix-based Rényi’s α -order entropy as a key ingredient. Our Greedy DIB scales to large-scale data and popular deep architectures and performs better than state-of-the-art layer-wise training methods. Greedy DIB also outperforms SGD in traffic sign detection task which includes both regression and classification heads in a network. To the best of our knowledge, our results provide the first empirical evidence in real-world applications showing that greedy layer-wise training can outperform traditional end-to-end backpropagation.

A Matrix-based Rényi’s α -order entropy functional

Given $\{\mathbf{x}_i\}_{i=1}^n \in \mathcal{X}$, each \mathbf{x}_i can be a real-valued scalar or vector, and the Gram matrix $K \in \mathbb{R}^{n \times n}$ computed as $K_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$, a matrix-based analogue to Rényi’s α -entropy can be given by the following functional:

$$H_\alpha(A) = \frac{1}{1-\alpha} \log_2 (\text{tr}(A^\alpha)) = \frac{1}{1-\alpha} \log_2 \left(\sum_{i=1}^n \lambda_i(A)^\alpha \right), \quad (9)$$

where $\alpha \in (0, 1) \cup (1, \infty)$. A is the normalized version of K , i.e., $A = K / \text{tr}(K)$. $\lambda_i(A)$ denotes the i -th eigenvalue of A .

Similarly, given n pairs of samples $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^n$, the joint entropy between two random variable \mathbf{x} and \mathbf{y} can be defined as:

$$H_\alpha(A, B) = H_\alpha \left(\frac{A \circ B}{\text{tr}(A \circ B)} \right), \quad (10)$$

where $A_{ij} = \kappa_1(\mathbf{x}_i, \mathbf{x}_j)$, $B_{ij} = \kappa_2(\mathbf{y}_i, \mathbf{y}_j)$ and $A \circ B$ denotes the Hadamard product between the matrices A and B . Same to [23], we set κ as radial basis function (RBF) kernel $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2})$ to obtain the Gram matrices and set $\alpha = 1.01^4$. The kernel width σ is set by calculating the k ($k = 10$) nearest distances for each samples in the min-batch with size of n and take the mean. Then we choose kernel width σ of gram matrix K ($n \times n$) as the average of mean values for all n samples. Based on Eqs. (9) and (10), we can estimate mutual information as $I(\mathbf{x}; \mathbf{y}) = H(\mathbf{x}) + H(\mathbf{y}) - H(\mathbf{x}, \mathbf{y})$.

References

- [1] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015)
- [2] Pogodin, R., Latham, P.: Kernelized information bottleneck leads to biologically plausible 3-factor hebbian learning in deep networks. Advances in Neural Information Processing Systems **33**, 7296–7307 (2020)
- [3] Duan, S., Principe, J.C.: Training deep architectures without end-to-end back-propagation: A brief survey. arXiv preprint arXiv:2101.03419 (2021)
- [4] Hinton, G.: The forward-forward algorithm: Some preliminary investigations. arXiv preprint arXiv:2212.13345 (2022)
- [5] Karkar, S., Ayed, I., Bézenac, E., Gallinari, P.: Module-wise training of neural networks via the minimizing movement scheme. Advances in Neural Information Processing Systems **36** (2024)
- [6] Belilovsky, E., Eickenberg, M., Oyallon, E.: Greedy layerwise learning can scale to imagenet. In: International Conference on Machine Learning, pp. 583–593 (2019). PMLR
- [7] Jaderberg, M., Czarnecki, W.M., Osindero, S., Vinyals, O., Graves, A., Silver, D., Kavukcuoglu, K.: Decoupled neural interfaces using synthetic gradients. In: International Conference on Machine Learning, pp. 1627–1635 (2017). PMLR
- [8] Ma, W.-D.K., Lewis, J., Kleijn, W.B.: The hsic bottleneck: Deep learning without back-propagation. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 5085–5092 (2020)

⁴The Rényi's α -order entropy reduces to Shannon entropy when α is approaching to 1. Thus, we chose $\alpha = 1.01$ to approximate Shannon entropy for fair comparisons because most of existing IB approaches are based on Shannon entropy functional.

- [9] Crick, F.: The recent excitement about neural networks. *Nature* **337**(6203), 129–132 (1989)
- [10] Bengio, Y.: How auto-encoders could provide credit assignment in deep networks via target propagation. arXiv preprint arXiv:1407.7906 (2014)
- [11] Lee, D.-H., Zhang, S., Fischer, A., Bengio, Y.: Difference target propagation. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 498–515 (2015). Springer
- [12] Meulemans, A., Carzaniga, F., Suykens, J., Sacramento, J., Grawe, B.F.: A theoretical framework for target propagation. *Advances in Neural Information Processing Systems* **33**, 20024–20036 (2020)
- [13] Czarnecki, W.M., Świrszcz, G., Jaderberg, M., Osindero, S., Vinyals, O., Kavukcuoglu, K.: Understanding synthetic gradients and decoupled neural interfaces. In: International Conference on Machine Learning, pp. 904–912 (2017). PMLR
- [14] Lansdell, B.J., Prakash, P.R., Körding, K.P.: Learning to solve the credit assignment problem. In: 8th International Conference on Learning Representations, ICLR (2020)
- [15] Nøkland, A., Eidnes, L.H.: Training neural networks with local error signals. In: International Conference on Machine Learning, pp. 4839–4850 (2019). PMLR
- [16] Wu, C.T., Masoomi, A., Gretton, A., Dy, J.: Deep layer-wise networks have closed-form weights. In: International Conference on Artificial Intelligence and Statistics, pp. 188–225 (2022). PMLR
- [17] Duan, S., Yu, S., Chen, Y., Príncipe, J.C.: On kernel method-based connectionist models and supervised deep learning without backpropagation. *Neural Comput.* **32**(1), 97–135 (2020)
- [18] Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: International Conference on Learning Representations (2015)
- [19] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
- [20] Sakamoto, K., Sato, I.: End-to-end training induces information bottleneck through layer-role differentiation: A comparative analysis with layer-wise training. arXiv preprint arXiv:2402.09050 (2024)
- [21] Nguyen, T.T., Choi, J.: Markov information bottleneck to improve information

- flow in stochastic neural networks. *Entropy* **21**(10), 976 (2019)
- [22] Strouse, D., Schwab, D.J.: The deterministic information bottleneck. *Neural Computation* **29**(6), 1611–1630 (2017)
- [23] Yu, X., Yu, S., Principe, J.C.: Deep deterministic information bottleneck with matrix-based entropy functional. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (2021). IEEE
- [24] Gretton, A., Bousquet, O., Smola, A., Schölkopf, B.: Measuring statistical dependence with hilbert-schmidt norms. In: International Conference on Algorithmic Learning Theory, pp. 63–77 (2005). Springer
- [25] Tishby, N., Pereira, F.C.N., Bialek, W.: The information bottleneck method. In: The 37th Annual Allerton Conference on Communication, Control, and Computing (1999)
- [26] Sanchez Giraldo, L.G., Rao, M., Principe, J.C.: Measures of entropy from data using infinitely divisible kernels. *IEEE Transactions on Information Theory* **61**(1), 535–548 (2014)
- [27] Yu, S., Sanchez Giraldo, L.G., Jenssen, R., Principe, J.C.: Multivariate extension of matrix-based renyi’s α -order entropy functional. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019)
- [28] Cristianini, N., Shawe-Taylor, J., Elisseeff, A., Kandola, J.: On kernel-target alignment. *Advances in neural information processing systems* **14** (2001)
- [29] Shwartz-Ziv, R., Tishby, N.: Opening the black box of deep neural networks via information. arXiv preprint arXiv:1703.00810 (2017)
- [30] Yu, S., Giraldo, L.G.S., Príncipe, J.C.: Information-theoretic methods in deep neural networks: Recent advances and emerging opportunities. In: IJCAI, pp. 4669–4678 (2021)
- [31] Chelombiev, I., Houghton, C., O’Donnell, C.: Adaptive estimators show information compression in deep neural networks. arXiv preprint arXiv:1902.09037 (2019)
- [32] Yu, S., Príncipe, J.C.: Understanding autoencoders with information theoretic concepts. *Neural Networks* **117**, 104–123 (2019)
- [33] Noshad, M., Zeng, Y., Hero, A.O.: Scalable mutual information estimation using dependence graphs. In: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2962–2966 (2019). IEEE
- [34] Yu, S., Wickstrøm, K., Jenssen, R., Príncipe, J.C.: Understanding convolutional neural networks with information theory: An initial exploration. IEEE

- [35] Li, H., Xu, Z., Taylor, G., Studer, C., Goldstein, T.: Visualizing the loss landscape of neural nets. *Advances in neural information processing systems* **31** (2018)
- [36] Keskar, N.S., Mudigere, D., Nocedal, J., Smelyanskiy, M., Tang, P.T.P.: On large-batch training for deep learning: Generalization gap and sharp minima. In: 5th International Conference on Learning Representations, ICLR 2017
- [37] Cha, J., Chun, S., Lee, K., Cho, H.-C., Park, S., Lee, Y., Park, S.: Swad: Domain generalization by seeking flat minima. *Advances in Neural Information Processing Systems* **34**, 22405–22418 (2021)
- [38] Yao, Z., Gholami, A., Lei, Q., Keutzer, K., Mahoney, M.W.: Hessian-based analysis of large batch training and robustness to adversaries. *Advances in Neural Information Processing Systems* **31** (2018)
- [39] Granziol, D.: Flatness is a false friend. arXiv preprint arXiv:2006.09091 (2020)
- [40] Zhu, Z., Liang, D., Zhang, S., Huang, X., Li, B., Hu, S.: Traffic-sign detection and classification in the wild. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2110–2118 (2016)
- [41] Meng, Z., Fan, X., Chen, X., Chen, M., Tong, Y.: Detecting small signs from large images. In: 2017 IEEE International Conference on Information Reuse and Integration (IRI), pp. 217–224 (2017). IEEE
- [42] Ayachi, R., Afif, M., Said, Y., Atri, M.: Traffic signs detection for real-world application of an advanced driving assisting system using deep learning. *Neural Processing Letters* **51**(1), 837–851 (2020)
- [43] Redmon, J.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2016)
- [44] Wu, Y., Li, Z., Chen, Y., Nai, K., Yuan, J.: Real-time traffic sign detection and classification towards real traffic scene. *Multimedia Tools and Applications* **79**, 18201–18219 (2020)
- [45] Qu, S., Yang, X., Zhou, H., Xie, Y.: Improved yolov5-based for small traffic sign detection under complex weather. *Scientific reports* **13**(1), 16219 (2023)
- [46] Raghu, M., Gilmer, J., Yosinski, J., Sohl-Dickstein, J.: Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. *Advances in neural information processing systems* **30** (2017)
- [47] Horibe, Y.: Entropy and correlation. *IEEE Transactions on Systems, Man, and Cybernetics* **SMC-15**(5), 641–642 (1985) <https://doi.org/10.1109/TSMC.1985>.

6313441

- [48] Chen, S., Ma, B., Zhang, K.: On the similarity metric and the distance metric. *Theoretical Computer Science* **410**(24-25), 2365–2376 (2009)
- [49] Kirsch, A., Lyle, C., Gal, Y.: Unpacking information bottlenecks: Surrogate objectives for deep learning. *arXiv preprint arXiv:2003.12537* (2020)
- [50] Ahuja, K., Caballero, E., Zhang, D., Gagnon-Audet, J.-C., Bengio, Y., Mitliagkas, I., Rish, I.: Invariance principle meets information bottleneck for out-of-distribution generalization. *Advances in Neural Information Processing Systems* **34**, 3438–3450 (2021)
- [51] Yang, Y., Luo, H., Xu, H., Wu, F.: Towards real-time traffic sign detection and classification. *IEEE Transactions on Intelligent transportation systems* **17**(7), 2022–2031 (2015)
- [52] Stallkamp, J., Schlipsing, M., Salmen, J., Igel, C.: The german traffic sign recognition benchmark: a multi-class classification competition. In: *The 2011 International Joint Conference on Neural Networks*, pp. 1453–1460 (2011). IEEE