Tree-like Decision Distillation

Jie Song[1,*], Haofei Zhang[1,*], Xinchao Wang[2,4], Mengqi Xue[1], Ying Chen[1], Li Sun[1,†], Dacheng Tao[3], and Mingli Song[1]
[1]Zhejiang University, [2]National University of Singapore,
[3]The University of Sydney, [4]Stevens Institute of Technology
{sjie,haofeizhang,mqxue,lynesychen,lsun,brooksong}@zju.edu.cn,
xinchao@nus.edu.sg, dacheng.tao@sydney.edu.au

Abstract

Knowledge distillation pursues a diminutive yet well-behaved student network by harnessing the knowledge learned by a cumbersome teacher model. Prior methods achieve this by making the student imitate shallow behaviors, such as soft targets, features, or attention, of the teacher. In this paper, we argue that what really matters for distillation is the intrinsic problem-solving process captured by the teacher. By dissecting the decision process in a layer-wise manner, we found that the decision-making procedure in the teacher model is conducted in a coarse-to-fine manner, where coarse-grained discrimination (e.g., animal vs vehicle) is attained in early layers, and fine-grained discrimination (e.g., dog vs cat, car vs truck) in latter layers. Motivated by this observation, we propose a new distillation method, dubbed as Tree-like Decision Distillation (TDD), to endow the student with the same problem-solving mechanism as that of the teacher. Extensive experiments demonstrated that TDD yields competitive performance compared to state of the arts. More importantly, it enjoys better interpretability due to its interpretable decision distillation instead of dark knowledge distillation.

1. Introduction

Knowledge Distillation (KD), whose ultimate goal is to craft a lightweight student model with the aid of a capable yet cumbersome teacher model, has become one of the most flourishing research topic in deep learning since the pioneering work of [8]. Its success is largely attributed to the *dark knowledge* learned by the over-parameterized teacher model, which is exploited to regularize the learning of a low-capacity student model without sacrificing too much performance.

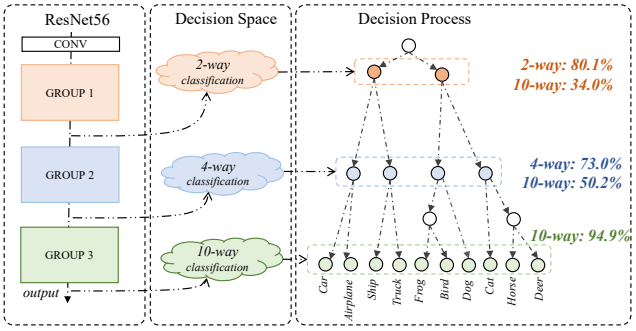

Figure 1. An illustrative diagram of the coarse-to-fine decision process on CIFAR10. After the first group of ResNet56, vehicles and animals can be distinguished with accuracy of 80%, while the 10-way classification here reaches only 34%.

Although remarkable progress has been made in the last several years, most existing KD methods are still stuck in the stage of mimicking shallow behaviors, such as soft targets [8], features [21], or attentions [41], of the teacher. Few of them attempt to figure out the problem-solving process underlying the pre-trained teacher model, which leaves the student produced by KD entirely a *black box*. Moreover, as some behaviors are partially dependent on the network architecture, directly copying these behaviors yields inferior performance especially when the architecture gap between the teacher and the student is significant [16, 31, 19].

In this paper, we argue that what really matters for distillation is the intrinsic problem-solving process captured by the teacher. By dissecting the decision process in the teacher model in a layer-wise manner, we found that although most multi-layered neural networks are designed to make the classification predictions in the last classification layer, the decision-making procedure is in fact learned to be conducted in a coarse-to-fine way and distributed over many layers, as shown in Figure 1. The early layers tend to capture the salient visual cues, and thus be capable of distinguishing between categories that are visually diverse enough, *e.g.*, the human-made categories (vehicles) *versus*

*Equal contribution
†Corresponding author

the natural categories (animals) in Figure 1. The latter layers, on the other hand, are amenable to make the final classification and thus able to conduct more fine-grained recognition. The overall decision process in the deep network is executed progressively in such an increasingly coarse-to-fine manner, rather than being concentrated on the very last classification layer.

We propose a new method, dubbed as *Tree-like Decision Distillation* (TDD), to endow the student with the same problem-solving process as that of the teacher. TDD first empirically analyzes the decisions made in different layers of the teacher model, and then imposes the same decision constraints on the student model, which impels the student to master the same problem-solving solution. As the student in TDD does not need to explore the solution flow again during the training phase, it converges much faster and achieves higher final accuracy. Furthermore, as TDD explores the underlying decision process rather than simply imitating some dark knowledge, it possesses better interpretability than prior methods.

Our main contributions are therefore summarized as follows: (1) we empirically demonstrated that the decision process underlying deep networks is executed in a coarse-to-fine manner which is somewhat similar to that of a decision tree; (2) we propose TDD to distill the decision process from teacher into the student, which relieves the student of the burden of searching in the solution space; (3) extensive experiments are conducted to demonstrate that TDD yields competitive accuracy to the state of the arts. Meanwhile, it enjoys better interpretability.

2. Related Work

We briefly review two research topics which are most related to this work, including knowledge distillation and decision process in neural networks.

2.1. Knowledge Distillation

Knowledge distillation has attracted increasing attention thanks to its important role in deploying deep networks to low-capacity edge devices. The main idea is leveraging the *dark knowledge* encoded in a bulky teacher to craft a lightweight student model with performance on par with the teacher. Over the last several years, most works devote themselves to the exploration of different forms of the dark knowledge, including soft targets [8, 24, 23, 38], features [21, 25, 15, 37], attention [41], factors [11], activation boundary [4], instance relationship [14, 17, 32, 36] and so on. By imitating the teacher to behave in a similar way, the student achieves comparable performance even with much fewer parameters.

Albeit great successes achieved by these methods, existing methods still suffer from two shortcomings: (1) the dark knowledge, as its name implies, is generally hard to explain; (2) some dark knowledge is affected by not only the task, but also the architecture itself. To alleviate these issues, we propose to distill the underlying decision process to address the KD problem. The work which shares the most similar idea to our method is [39], where the authors utilize *Flow of Solution Procedure* (FSP) to train the student. However, FSP is simply defined as the Gramian matrix that is computed by the inner products between features from two consecutive layers. It is still "dark" in interpretability and has high requirements for architecture similarity.

2.2. Decision in Neural Networks

Albeit the widespread successes of deep models in various fields in recent years, the *black-box* pecularity still remains an open problem to be resolved. Uncovering the mystery of deep models has been the desiderata in the community, and various approaches have been proposed. For example, attribution methods [42, 27, 3, 26, 2, 35, 34] attempt to understand how deep models work by identifying the important dimensions in the input space. Here we only review those which explicitly explore the decision process in deep neural networks. Frosst and Hinton [5] proposes a soft decision tree that is more transparent to mimic the output of a neural network. However, as every decision is made in the original input space, the better interpretability is achieved by sacrificing the performance. Tanno *et al*. [30] proposes *adaptive neural tree* to unify neural networks and decision trees, which also enjoys some human-interpretable properties. Recently, Wan *et al*. [33] proposes *Neural-Backed Decision Trees* (NBDT) to resolve the tension between accuracy and interpretability. However, the decision tree is constructed from only the weight space of the classification layer, which actually leaves a large portion of the network unexplainable. In this paper, we dissect the decision in a layer-wise manner, and our goal is to craft a lightweight student model resorting to the decision process, which is vastly different from previous works.

3. Method

In this section, we first introduce the preliminaries of vanilla knowledge distillation, then delineate the proposed method in more details.

3.1. Preliminaries

The goal of knowledge distillation is to optimize a student model under the supervision from a pre-trained teacher. [8] propose to distill the "dark knowledge" from the teacher via aligning the soft targets

$$\mathcal{O}_{KD} = \ell(f_s(\boldsymbol{x}_i), y_i) + \alpha D_{KL}\left[p_\tau(f_t(\boldsymbol{x}_i)), p_\tau(f_s(\boldsymbol{x}_i))\right], \tag{1}$$

where $\boldsymbol{x}_i$ is the input and y_i is its associated category label. f_t and f_s denote the functions underlying the teacher

and the student models, respectively. ℓ is the conventional cross entropy loss for classification problems, and D_{KL} is the Kullback-Leibler divergence between the predicted categorial distributions from the teacher and the student models. p_τ transforms the logits into softened probability:

$$p_\tau(f(\boldsymbol{x}_i)) = softmax(f(\boldsymbol{x}_i)/\tau), \tag{2}$$

where τ is the non-negative temperature which is used to smooth the distributions.

3.2. Tree-like Decision Distillation

In this paper, the term "decision" refers to making some form of classification. A multi-layered network for classification can be viewed as making different decisions in different layers. The early layers, which tend to capture salient visual cues, make some vague and coarse-grained classification. The last layers, which produce linear discriminant features, make the precise and fine-grained decision. The goal of TDD is to distill the decisions in different layers into the student. Specifically, TDD consists of three main steps: *layer-wise discriminant analysis*, *intermediate decision making* and *decision distillation*. Layer-wise discriminant analysis dissects the discrimination ability of the features from the teacher in a layer-wise manner. Intermediate decision making determines the intermediate decision based on the discriminant analysis. Decision distillation utilizes the intermediate decision objective from step 2 to train the student model. Now we describe the details of TDD step by step.

3.2.1 Layer-wise discriminant analysis

Given the pre-trained teacher model, we first analyze the discrimination of features from every layer. Let there be N different categories, and each of which has K associated training images. For the l-th layer in the teacher model, the features extracted from the k-th image from the n-th category are denoted by $\boldsymbol{z}_k^n \in \mathbb{R}^{w_l h_l c_l}$, where w_l, h_l and c_l denote the width, the height and the channels of the feature map. As the dimensionality is high and different for different layers, we adopt *global average pooling* to squeeze out the spatial dimensions, *i.e.*, $\boldsymbol{z}_k$ becomes a c_l-dimensional vector. After that we adopt *Linear Discriminant Analysis* (LDA) to reduce the dimensions $(c_1, c_2, ...)$ of features from different layers to a fixed dimension c ($c \leq c_1, c_2, ...$). Specifically, for each layer we maximize the following Fisher criterion

$$\mathcal{J}(\boldsymbol{W}) = \frac{\det\left(\boldsymbol{W}^T \boldsymbol{S}_B \boldsymbol{W}\right)}{\det\left(\boldsymbol{W}^T \boldsymbol{S}_W \boldsymbol{W}\right)}, \tag{3}$$

where $\det$ represents matrix determinant. $\boldsymbol{S}_B$ and $\boldsymbol{S}_W$ are the between-class and the within-class scatter matrices, respectively:

$$\boldsymbol{S}_B = \sum\nolimits_{i=1}^{N} K \cdot (\overline{\boldsymbol{z}^i} - \bar{\boldsymbol{z}})(\overline{\boldsymbol{z}^i} - \bar{\boldsymbol{z}})^T, \tag{4}$$

$$\boldsymbol{S}_W = \sum\nolimits_{i=1}^{N} \sum\nolimits_{k=1}^{K} (\boldsymbol{z}_k^i - \overline{\boldsymbol{z}^i})(\boldsymbol{z}_k^i - \overline{\boldsymbol{z}^i})^T. \tag{5}$$

$\overline{\boldsymbol{z}^i}$ and $\bar{\boldsymbol{z}}$ denote the average of feature vectors from the i-th category and the all the categories, respectively. LDA reduces the dimensions and meanwhile preserves as much of the class discriminatory information as possible. With the optimal $\boldsymbol{W}^* = \arg\max_{\boldsymbol{W}} \mathcal{J}(\boldsymbol{W})$, all original features are projected to the lower-dimensional subspace.

In the subspace, the average feature vectors are computed to represent the corresponding categories. Then the agglomerative hierarchical clustering algorithm is utilized to generate the category similarity tree, *i.e.*, the dendrogram as shown in Figure 2. This tree describes the category relationships between categories in the l-th feature space of the teacher model, which we believe is beneficial for the student to master the same problem solution.

3.2.2 Intermediate Decision Making

As hierarchical clustering exhaustively separates each category apart, it is hard for us to understand what decision is making in each layer. To resolve this problem, at this step we need determine the intermediate decision that is made in every layer. We first convert the dendrogram obtained from hierarchical clustering to a decision tree. Here we adopt a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ to describe the decision tree, where $\mathcal{V} = \{v_1, v_2, ...\}$ is the set of nodes, and $\mathcal{E} = \{e_1, e_2, ...\}$ the set of edges between the nodes. The root node subsumes all the categories, while each leaf node contains only one. We use $\mathcal{Y}_v$ to denote the categories in node v. From the root node to the leaf nodes, the decision tree is actually conducting a coarse-to-fine decision process: from the root node to its two children, the network makes the coarsest decision; when it arrives to the leaves, the most precise decision should be provided. Layers in different depth of the network is actually making different levels of decision.

The hierarchical clustering exhaustively expands the decision tree such that every category go to an individual leaf. However, the intermediate layers only conduct some coarse-grained classification. To determine the level of decision the layer makes, we adopt a bottom-up breadth-first strategy to merge leave nodes into their parents until the terminal criterion is triggered. Specifically, each node v in $\mathcal{G}$ is associated with two matrices, including a within-class scatter matrix $\boldsymbol{S}_W^v$ and a between-class scatter matrix $\boldsymbol{S}_B^v$. The two matrices depict the compactness and the separation between projected features of the data from categories under the node. The linear discrimination of features from differ-

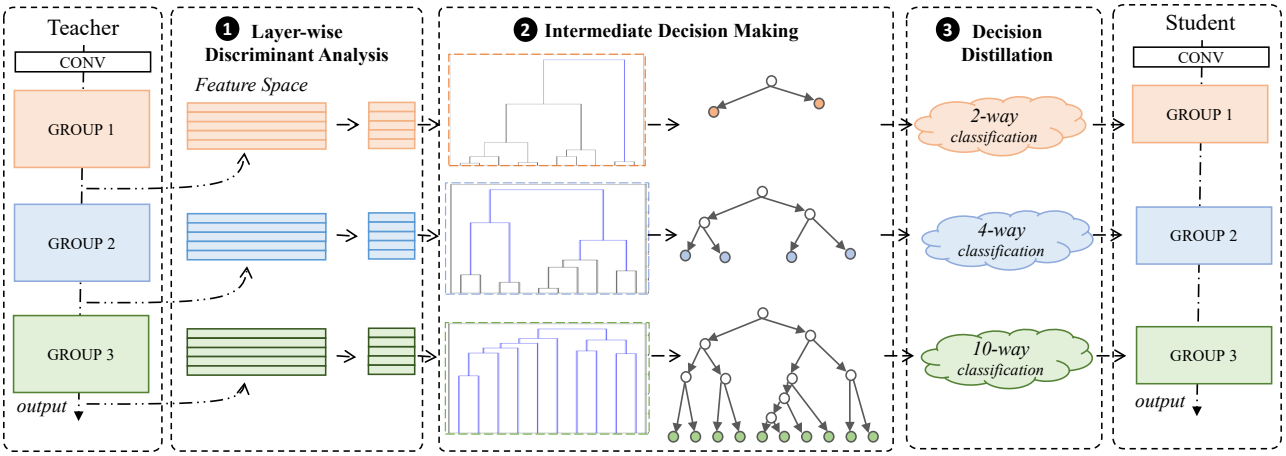

Figure 2. The pipeline of the proposed *tree-like decision distillation*. It mainly consists of three steps: 1) layerwise discriminant analysis, 2) intermediate decision making, and 3) decision distillation.

ent categories under node v is measure by

$$\mathcal{J}(v) = \frac{\sqrt{tr(\boldsymbol{S}_B^v)}}{\sum_{y \in \mathcal{Y}_v} tr(\boldsymbol{S}_{W,y}^v)}, \tag{6}$$

where tr denotes the matrix trace, and $\mathcal{Y}_v$ denotes the categories under node v. If $\mathcal{J}(v)$ is less than a pre-defined threshold t, node v and its sibling node are merged into their parent and the parent node becomes a new leaf, with its associated $\boldsymbol{S}_B^v$ and $\boldsymbol{S}_W^v$ updated accordingly[1]. The process repeats recursively until all leaves can not be merged any more or the root node is reached. Finally, the original cumbersome tree $\mathcal{G}$ collapses to a simplified version $\mathcal{G}^*$ where each leaf node may subsume more than one category, *i.e.*, $|\mathcal{Y}_{v^*}| \geq 1$.

Here we adopt $\mathcal{G}_l = \{\mathcal{V}_l, \mathcal{E}_l\}$ to denote the decision tree produced by the l-th layer. The simplified version $\mathcal{G}_l^* = \{\mathcal{V}_l^*, \mathcal{E}_l^*\}$ is harnessed to determine the intermediate decision which is made by this layer. If the tree is collapsed to a single node, *i.e.*, $|\mathcal{V}_l^*| = 1$, the node subsumes all the categories, which implies that the network cannot make any decision in this layer. On the other hand, if $\mathcal{G}_l^*$ contains more than one node, *i.e.*, $|\mathcal{V}_l^*| > 1$, then the l-th layer is deemed to be making the differentiation between some super-classes denoted by the leaves in $\mathcal{G}_l^*$. In other words, the l-th layer is making the $\left|\mathcal{V}_{l,leaf}^*\right|$-way classification where $\mathcal{V}_{l,leaf}^*$ refers to the leaves of $\mathcal{G}_l^*$, $\mathcal{V}_{l,leaf}^* \in \mathcal{V}_l^*$. Additionally, as the teacher model is not explicitly supervised by corresponding classification objective in the middle layers, these layers are actually making soft decisions.

3.2.3 Decision distillation

This step utilizes the soft decision made by different layers from the teacher to optimize the student model. We omit those layers where $|\mathcal{V}_l^*| = 1$ and utilize the layers with $|\mathcal{V}_l^*| > 1$ to guide the training of the student. Specifically, let $\mathcal{L} = \{l_1, l_2, ...\}$ be the set of layers where $|\mathcal{V}_l^*| > 1$ in the teacher model. For each $l \in \mathcal{L}$, a micro classification module is attached to the corresponding layer in the student model to help the student form the same decision process as the teacher. The micro classification module is composed of a global average pooling layer and a fully connected layer which outputs the classification logits for the intermediate classification.

Determining the corresponding layer in the student model for the layer in the teacher model is not a trivial problem, as the student and the teacher models may be of different number of layers or in heterogeneous architectures. We solve this problem based on the following three observations: (1) experiments show that features from the first several layers are usually not discriminative, *i.e.*, they can not make any decisions yet ($|\mathcal{V}_l^*| = 1$); (2) features from adjacent layers are approximately equally discriminative, *i.e.*, they are making the similar decisions; (3) existing widely used deep networks, albeit in different architectures or layers, have roughly the same number of pooling layers, *e.g.*, ResNet [6] and VGG [28] for ImageNet.

Based on these observations, we ignore the first half part of network and only distill the decisions from the layers from the second half part. TDD are conducted only at the feature space following the pooling layers, which are sparsely distributed over the deep neural network. In thus way, the layers for TDD in the teacher and the student models can correspond one by one. Finally, the TDD optimization objective for training the student model is

$$\mathcal{O}_{TDD} = \mathcal{O} + \beta \sum\nolimits_{l \in \mathcal{L}} D_{KL}\left[p_\tau(g^l(y_i)), p_\tau(f_s^l(\boldsymbol{x}_i))\right], \tag{7}$$

where $f_s^l(\boldsymbol{x}_i)$ denotes the logits produced from the layer in the student model which corresponds to the l-th layer in the teacher. $\mathcal{O}$ is the original classification loss. g^l is the

[1] Updating rules are provided in the supplementary materials.

function converting the original category label to the coarse-grained category which is differentiated by the l-th layer in the teacher:

$$g^l(\boldsymbol{x}_i) = \sum\nolimits_{v \in \mathcal{V}^*_{l,leaf}} \mathbb{I}(y_i \in \mathcal{Y}_v)\boldsymbol{p}_l(y_i), \tag{8}$$

where $\mathbb{I}$ denotes the indicator function, and $\boldsymbol{p}_l(y_i)$ is a $\left|\mathcal{V}^*_{l,leaf}\right|$-dimensional vector where the i-th element denotes the probability of data from category y_i is classified into the i-th superclass in the subspace after LDA by nearest neighbor classification. After the training of the student model, those micro classification modules are removed when the student is deployed to real-world scenarios.

Formally speaking, the proposed TDD objective is akin to some previous works which also impose some constraints on the intermediate layers, such as GoogleNet [29]. Here we underline two vital differences: (1) GoogleNet is thus designed for alleviating vanishing gradient [9], while TDD is designed for regularizing the student to search within the same solution space as that captured by the teacher; (2) GoogleNet adopts the same classification objectives in different layers, while TDD adopts coarse-to-fine classification objectives which avoids the overcorrection induced by the final classification objective.

4. Experiments

4.1. Implementation Details

4.1.1 Datasets

Three widely-used datasets as benchmarks for distillation are adopted to validate the proposed method, including CIFAR-10 [12], CIFAR100 [12] and tiny-ImageNet [13]. The CIFAR-10 dataset consists of $60,000$ 32x32 colour images in 10 classes, with $6,000$ images per class. There are $50,000$ training images and $10,000$ test images in the original split. CIFAR-100 is just like the CIFAR-10, except it has 100 classes containing 600 images each. There are 500 training images and 100 testing images per class. The 100 classes in the CIFAR-100 are grouped into 20 superclasses. Tiny-ImageNet is a subset of ImageNet with 200 classes, where each image is down-sized to 64x64 pixels. Each class has 500 training images, 50 validation images, and 50 test images.

4.1.2 Models

Various architectures, including ResNet [6], Wide Residual Network [40], MobileNet [22], ShuffleNet [43] and VGG [28], are used to evaluate TDD. TDD is tested under two distillation schemes: homogeneous distillation and heterogeneous distillation. Under the homogeneous distillation, the student and the teacher are in different capacity, but of the same type of architecture (*e.g.*, Resnet56 $\rightarrow$ ResNet20, WRN-40-2 $\rightarrow$ WRN-16-2). Under the heterogeneous distillation, the student mode is in a different architecture from the teacher model (*e.g.*, ResNet50 $\rightarrow$ MobileNetV2, WRN-40-2 $\rightarrow$ ShuffleNetV1).

4.1.3 Cross Validation

There are several hyper-parameters involved in the proposed method, including α and τ in Eqn. 1, the pre-defined threshold t for leaves merging in intermediate decision making, and β in Eqn. 7. Exhaustively exploring the optimal combinations of all these hyper-parameters is an unaffordable burden for us. For α and τ , we directly adopt the values adopted in prior work [31]. For t and β, we opt for cross-validation to set their values. As CIFAR-10 and CIFAR-100 have no validation set in their original split, we randomly reserve $5,000$ images from their training set for validation.

4.1.4 Training Details

As we are devoted to improve the knowledge distillation performance, we adopt the same experimental settings for training both the teacher and the student models, regardless of the differences in model architectures and datasets. It simplifies our experiments and help us make fair comparisons with state-of-the-arts. Specifically, we adopt *Stochastic Gradient Descent* to optimize the network, and the initial learning rate is set 0.05, decaying by a factor of 0.1 at $\{150, 180, 210\}$ epochs, respectively. The training phase ceases at 240 epochs. The batch size is 64 and the weight decay factor is 0.0005.

4.2. Benchmark Comparisons

As the architecture is found to be an important factor for distillation, the proposed TDD is evaluated under two settings: homogeneous distillation where the teacher and the student models are in the same architecture, and heterogeneous distillation where the student model is in a different architecture from that of the teacher model.

4.2.1 Homogeneous Distillation

Firstly we validate the proposed method under the homogeneous distillation settings. We compare the proposed TDD with a bundle of existing distillers, including the vanilla Knowledge Distillation (KD) [8], Fitnets [21], Attention Transfer (AT) [41], Flow of Solution Procedure (FSP) [39], Neural Selectivity Transfer (NST) [10], Factor Transfer (FT) [11], Probabilistic Knowledge Transfer (PKT)[18], Similarity-Preserving Knowledge Distillation (SPKD)[32], Variational Knowledge Distillation (VID)[1], Correlation Congruence Knowledge Distillation (CCKD) [20], Relational Knowledge Distillation (RKD) [17] and Contrastive

Table 1. Top-1 accuracy of homogeneous distillation on CIFAR-10, CIFAR-100 and tiny-ImageNet (in %). Experiments are repeated for three times and the average results are provided. Results on CIFAR-100 are copied from CRD [31]. The best results are shown in **bold** font and the second best in blue font. "TDD$_{+\text{CRD}}$" denotes that the proposed TDD is combined with CRD.

	CIFAR-10			**CIFAR-100**			**tiny-ImageNet**		
Teacher	ResNet56	WRN_40_2	WRN_40_2	ResNet56	WRN_40_2	WRN_40_2	ResNet56	WRN_40_2	WRN_40_2
Student	ResNet20	WRN_16_2	WRN_40_1	ResNet20	WRN_16_2	WRN_40_1	ResNet20	WRN_16_2	WRN_40_1
Teacher	93.90	94.77	94.77	72.34	75.61	75.61	58.34	61.26	61.26
Student	92.49	93.65	93.46	69.06	73.26	73.26	52.66	57.17	56.25
KD [8]	92.78	94.54	93.95	70.66	74.92	73.54	53.04	59.16	57.75
Fitnets [21]	92.55	93.73	93.73	69.21	73.58	72.24	51.73	57.75	N/A
AT [41]	93.03	94.17	94.34	70.55	74.08	72.77	54.01	58.71	57.41
FSP [39]	91.93	93.43	N/A	69.95	72.91	N/A	53.55	57.33	N/A
NST [10]	92.81	94.15	93.90	69.60	73.68	72.24	51.89	–	–
FT [11]	93.14	94.26	94.40	69.84	73.25	71.59	54.20	58.31	56.30
PKT [18]	93.19	94.61	94.12	70.34	74.54	73.45	54.31	59.06	57.27
SPKD [32]	93.05	94.16	94.01	69.67	73.83	72.43	54.03	55.69	53.74
VID [1]	92.80	94.17	93.60	70.38	74.11	73.30	53.20	58.51	57.45
CCKD [20]	92.39	93.67	93.29	69.63	73.56	72.21	52.38	58.32	55.72
RKD [17]	92.71	94.37	93.85	69.61	73.35	72.22	53.13	57.38	55.90
CRD [31]	-	-	-	71.16	75.48	74.14	-	-	-
TDD	93.25±0.11	94.60±0.08	94.25±0.15	71.53±0.21	75.01±0.18	74.04±0.08	54.45±0.07	59.22±0.15	58.42±0.16
TDD$_{+\text{CRD}}$	**93.42**±0.12	**94.68**±0.13	**94.51**±0.10	**71.88**±0.24	**75.71**±0.19	**74.35**±0.14	**54.85**±0.13	**59.53**±0.20	**59.20**±0.12

Table 2. Top-1 accuracy of heterogeneous distillation on CIFAR-10, CIFAR-100 and tiny-ImageNet (in %). Experiments are repeated for three times and the average results are provided. Results on CIFAR-100 are copied from CRD [31]. The best results are shown in **bold** font and the second best in blue font.

	CIFAR-10			**CIFAR-100**			**tiny-ImageNet**		
Teacher	ResNet50	ResNet50	WRN_40_2	ResNet50	ResNet50	WRN_40_2	ResNet50	ResNet50	WRN_40_2
Student	MobileNet	VGG8	ShuffleNet	MobileNet	VGG8	ShuffleNet	MobileNet	VGG8	ShuffleNet
Teacher	94.88	94.88	94.77	79.34	79.34	75.61	68.97	68.97	61.26
Student	89.56	91.48	92.62	64.60	70.36	70.50	58.35	56.47	60.52
KD [8]	90.11	93.31	93.31	67.35	73.81	74.83	58.68	60.27	64.80
Fitnets [21]	89.52	90.94	93.34	63.16	70.69	73.73	57.55	57.11	N/A
AT [41]	87.54	92.73	**94.21**	58.58	71.84	73.32	50.91	52.42	63.90
NST [10]	88.81	91.01	93.83	64.96	71.28	74.12	–	–	–
FT [11]	88.98	92.40	94.03	60.99	70.29	72.03	58.65	57.69	62.47
PKT [18]	90.12	92.62	93.61	66.52	73.01	73.89	59.29	58.68	63.10
SPKD [32]	89.73	92.79	93.59	68.08	73.34	74.52	58.11	58.57	64.62
VID [1]	89.27	93.54	91.77	67.57	70.30	73.61	57.50	55.86	63.58
CCKD [20]	89.62	91.45	92.99	65.43	70.25	71.38	57.89	55.37	61.16
RKD [17]	90.02	93.24	93.51	64.43	71.50	72.21	58.33	56.87	60.52
CRD [31]	-	-	-	69.11	74.30	76.05	-	-	-
TDD	90.32±0.17	93.66±0.11	93.57±0.16	68.37±0.08	74.41±0.19	75.60±0.15	59.09±0.15	60.42±0.12	65.27±0.10
TDD$_{+\text{CRD}}$	**90.66**±0.22	**94.25**±0.25	93.71±0.18	**69.22**±0.05	**74.47**±0.15	**76.34**±0.13	**59.72**±0.17	**61.23**±0.19	**65.50**±0.20

Representation Distillation (CRD) [31]. All these works are published within three years and well known in the field of knowledge distillation. They clearly represent the state of the arts.

Experiments are conducted on CIFAR-10, CIFAR-100 and tiny-ImageNet. Experimental results are listed in Table 4.1.4 where top-1 accuracy is provided. From the results, we can see that the proposed TDD produces superior or at least comparable accuracy compared to most existing methods. CRD [31] is a strong competitor which is more likely to produce higher accuracy in our experiments due to the proposed contrastive representation learning in it. When combined with the contrastive learning into our proposed TDD, our method produces superior performance to all ex-

isting methods including CRD. All these results demonstrate the effectiveness of decision distillation for training the lightweight student model.

4.2.2 Heterogeneous Distillation

As the model architecture itself provides strong regularization on solving the problem, the "dark knowledge" explored by prior methods are partially dependent on the architecture. Existing knowledge distillation methods work poorly when the teacher and the student models are vastly different in architecture. Here we validate that the proposed method is able to alleviate the problem thanks to its exploration into the problem-solving process. Here the selected competitors are the same as those used in the homogeneous distillation except FSP which has strong requirements for similar teacher and student architectures and thus can hardly applied to heterogeneous distillation. Experimental results are shown in Table 2. Under different architectures, most prior distillation methods yield inferior performance to the vanilla KD. Some of them even can not match the trivial baseline without any distillation due to the large architecture gap. However, as the proposed method TDD dives into the problem solving process that is much more model-agnostic, it thus exhibits higher accuracy to original KD and other competitors in most settings. Similar to homogeneous distillation, when combined with CRD, the proposed TDD further improves the accuracy by a considerable margin. As the proposed method exhibits better interpretability than all these methods, we believe it is a good complement to the current literature.

4.3. Ablation Study

Here we verify the necessity of ingredients in the proposed method. To this end, we design three variants of the proposed method: *TDD-random*, *TDD-same* and *TDD-soft*.

In TDD-random, the intermediate decisions are randomly generated instead of being distilled from the pre-trained teacher model. For example, on the CIFAR-10 dataset, TDD imposes a 2-way coarse classification objective on the feature space after the fourth pooling layer, with one way subsuming 6 categories and the other way subsuming 4 categories. For random decision, we also introduce a 2-way coarse classification here, however, the categories are randomly determined for each way.

In TDD-same, like GoogleNet where middle layers are imposed on the same optimization objective, we also constraint the middle layers of the student model using the same optimization objective, *i.e.*, the final classification objective. This variant is used to verify the effectiveness of coarse-to-fine decision process for distillation instead of the improvement of gradient issues by intermediate objective functions.

TDD parses the decision process in a hierarchical way.

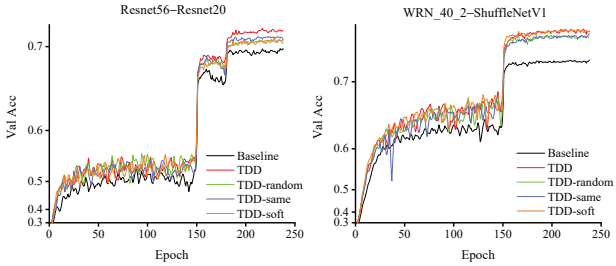

Figure 3. Validation accuracy curves on CIFAR100. **Left**: Resnet56→ResNet20. **Right**: WRN_40_2→ShuffleNet V1.

However, the decision-making process can also be interpreted in another way: the middle layers also make the same target classification, but in a lower precision. TDD-soft exploits the final predictions from the teacher model softened to varying degrees as the targets for middle layers in the student model. We believe this is also an effective method for distillation.

The validation accuracy curves of TDD, TDD-random, TDD-same and TDD-soft are depicted in Figure 3. The baseline is the student model trained without any distillation. It can be seen that TDD consistently yields superior or comparable performance compared to the baseline and the three variants under the two different experimental settings, one for homogeneous distillation and the other for heterogeneous distillation. TDD-soft, which shares the similar idea with TDD, also produces competitive performance in our experiments. However, as it does not interpret the decision process in a coarse-to-fine manner, the merit of higher interpretability is lost in it.

Note that here we do not exhaustively validate every component in the proposed method, as we believe our main idea can be implemented in several ways, not limited to the proposed instantiation. We believe with more sophisticated implementations, the distillation performance will be further improved, which is left for future work.

4.4. Interpretability

In this section, we show that TDD enjoys some human-interpretable properties. We first demonstrate the coarse-to-fine decision process in trained deep networks, then the human-interpretable intermediate decision is studied.

4.4.1 Coarse-to-fine Decision Process

To demonstrate the coarse-to-fine decision process underlying pre-trained deep models, here we visualize the feature distributions in different layers of ResNet56 on CIFAR10 in Figure 4. More results of other layers can be found in the supplementary materials. From Figure 4, we can see that from the first layer to the last layer, the extracted features become increasingly distinguishable as expected. The early layers are not capable of differentiating the final cat-

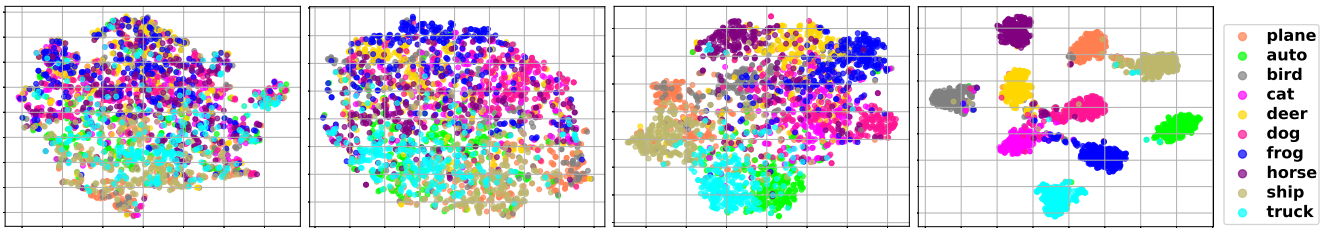

Figure 4. Visualization of feature distributions after linear discriminant analysis from the first, the 19-th, the 37-th, and the last layers (from left to right) using t-SNE [7].

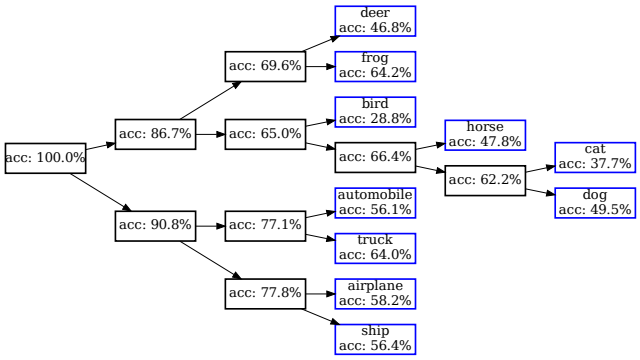

Figure 5. The decision process from a middle layer of ResNet56 on CIFAR10. Please zoom for better view.

egories as well as the last layer. However, these layers including even the first layer exhibit great potential to distinguish coarser-grained classes. For example, the features from the categories under the umbrella of vehicles tend to cluster together, while the features of animal categories tend to cluster in a different region.

To illustrate this point better, we depict the decision tree from the 37-th layer in Figure 5 . In the decision tree, each leaf node denotes one category and each non-leaf node is a decision on its two parents. The "acc" in each node denotes the classification accuracy in the feature space via nearest neighbor search. For leaf nodes, the overall average classification accuracy is 50.95%. Based on this result, the randomly guessing of the decision in the root node should be about 77.0%. However, the nearest neighbor search attains accuracy of 86.70% and 90.83% respectively for the left and the right children, which is significantly higher than randomly guessing. Similar conclusions can also be made from other layers and models. These results again prove that although the layers previous to the final classification layer can not make accurate predictions about the final categories, they have the ability to differentiate between coarser-grained categories, which validates the main assumption underlying the proposed method in this paper.

Table 3. Testing human-interpretable intermediate decision.

Layer	#1	#13	#25	#37	Random
Top-5 Acc	45.0%	60.0%	74.0%	84.0%	19.0%

4.4.2 Human-interpretable Intermediate Decision

The proposed TDD actually decomposed the whole classification task into a sequence of decisions. Although TDD imposes no human intervention on the decision process, some intermediate decisions in middle layers still coincide well with human perceptions. For example, from Figure 5 we can see that the root note is making differentiation between man-made vehicles and natural animals. To make a more comprehensive study on this, we adopt CIFAR100, where the 100 categories are grouped into 20 superclasses, to test the consistency between data-driven decision process and the human perceptions. Formally, every category y is used to query the top-5 nearest categories $\mathcal{Y}'$ in the decision tree from different layers. If one category in $\mathcal{Y}'$ and c are in the same manually-defined superclass, the query is deemed successful. Experimental results on WRN-40-2 is shown in Table 3. It can be seen that almost all layers produce intermediate decisions highly correlated with human perception. As the layers go deeper, the correlation becomes higher.

5. Conclusions and Future Work

We propose Tree-like Decision Distillation (TDD) to address the distillation problem. Unlike previous methods that utilize dark knowledge, such as soft targets, features, or attentions, for distillation, we argue what really matters for distillation is the underlying problem-sovling process. TDD first dissects the problem-solving process in a layer-wise manner, then forces the student to mimic the decision made by the teacher in different layers. Extensive experiments demonstrate that the proposed method enjoys higher accuracy, better interpretability and stronger generalization across heterogeneous architectures. In our future work, we will study more sophisticated implementations of the introduced idea to further improve the distillation performance.

Acknowledgments. This work is funded by the National Key R&D Program of China (Grant No: 2018AAA0101503) and the Science and technology project of SGCC (State Grid Corporation of China): fundamental theory of human-in-the-loop hybrid-augmented intelligence for power grid dispatch and control.

References

[1] S. Ahn, Shell Xu Hu, A. Damianou, N. Lawrence, and Z. Dai. Variational information distillation for knowledge transfer. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9155–9163, 2019. 5, 6

[2] Marco B Ancona, Enea Ceolini, Cengiz Oztireli, and Markus H. Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. In *International Conference on Learning Representations 2018*, 2018. 2

[3] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, Wojciech Samek, and Oscar Deniz Suarez. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. In *PloS one*, 2015. 2

[4] Sangdoo Yun Jin Young Choi Byeongho Heo, Minsik Lee. Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2019. 2

[5] Nicholas Frosst and Geoffrey E. Hinton. Distilling a neural network into a soft decision tree. *ArXiv*, abs/1711.09784, 2017. 2

[6] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 4, 5

[7] Geoffrey E. Hinton. Visualizing high-dimensional data using t-sne. 2008. 8

[8] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *ArXiv*, abs/1503.02531, 2015. 1, 2, 5, 6

[9] Sepp Hochreiter and Yoshua Bengio. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. 2001. 5

[10] Zehao Huang and Naiyan Wang. Like what you like: Knowledge distill via neuron selectivity transfer. *arXiv preprint arXiv:1707.01219*, 2017. 5, 6

[11] Jangho Kim, Seonguk Park, and Nojun Kwak. Paraphrasing complex network: Network compression via factor transfer. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 2760–2769. Curran Associates, Inc., 2018. 2, 5, 6

[12] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. 5

[13] Y. Le and X. Yang. Tiny imagenet visual recognition challenge. 2015. 5

[14] Yufan Liu, Jiajiong Cao, Bing Li, Chunfeng Yuan, Weiming Hu, Yangxi Li, and Yunqiang Duan. Knowledge distillation via instance relationship graph. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7089–7097, 2019. 2

[15] Sihui Luo, Wenwen Pan, Xinchao Wang, Dazhou Wang, Haihong Tang, and Mingli Song. Collaboration by competition: Self-coordinated knowledge amalgamation for multi-talent student learning. In *European Conference on Computer Vision*, pages 631–646, 2020. 2

[16] Seyed Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant. In *AAAI*, 2020. 1

[17] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3962–3971, 2019. 2, 5, 6

[18] N. Passalis and A. Tefas. Learning deep representations with probabilistic knowledge transfer. In *ECCV*, 2018. 5, 6

[19] Nikolaos Passalis, Maria Tzelepi, and Anastasios Tefas. Heterogeneous knowledge distillation using information flow modeling. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1

[20] Baoyun Peng, Xiao Jin, Jiaheng Liu, Shunfeng Zhou, Y. Wu, Y. Liu, Dong sheng Li, and Z. Zhang. Correlation congruence for knowledge distillation. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5006–5015, 2019. 5, 6

[21] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *CoRR*, abs/1412.6550, 2015. 1, 2, 5, 6

[22] Mark Sandler, A. Howard, Menglong Zhu, A. Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018. 5

[23] Chengchao Shen, Xinchao Wang, Jie Song, Li Sun, and Mingli Song. Amalgamating knowledge towards comprehensive classification. In *AAAI Conference on Artificial Intelligence*, pages 3068–3075, 2019. 2

[24] Chengchao Shen, Xinchao Wang, Youtan Yin, Jie Song, Sihui Luo, and Mingli Song. Progressive network grafting for few-shot knowledge distillation. In *AAAI Conference on Artificial Intelligence*, 2021. 2

[25] Chengchao Shen, Mengqi Xue, Xinchao Wang, Jie Song, Li Sun, and Mingli Song. Customizing student networks from heterogeneous teachers via adaptive knowledge amalgamation. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 2

[26] Avanti Shrikumar, Peyton Greenside, Anna Shcherbina, and Anshul Kundaje. Not just a black box: Learning important features through propagating activation differences. *CoRR*, abs/1605.01713, 2016. 2

[27] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, abs/1312.6034, 2013. 2

[28] K. Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2015. 4, 5

[29] Christian Szegedy, W. Liu, Y. Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, D. Erhan, V. Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015. 5

[30] Ryutaro Tanno, Kai Arulkumaran, D. Alexander, A. Criminisi, and A. Nori. Adaptive neural trees. In *ICML*, 2019. 2

[31] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *International Conference on Learning Representations*, 2020. 1, 5, 6

[32] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1365–1374, 2019. 2, 5, 6

[33] Alvin Wan, Lisa Dunlap, Daniel W. C. Ho, Jihan Yin, Sungwook Lee, Henry Jin, Suzanne Petryk, Sarah Adel Bargal, and Joseph E. Gonzalez. Nbdt: Neural-backed decision trees. *ArXiv*, abs/2004.00221, 2020. 2

[34] Yiding Yang, Zunlei Feng, Mingli Song, and Xinchao Wang. Factorizable graph convolutional networks. In *Advances in Neural Information Processing Systems*, 2020. 2

[35] Yiding Yang, Zunlei Feng, Mingli Song, and Xinchao Wang. Learning propagation rules for attribution map generation. In *European Conference on Computer Vision*, 2020. 2

[36] Yiding Yang, Jiayan Qiu, Mingli Song, Dacheng Tao, and Xinchao Wang. Distilling knowledge from graph convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 2

[37] Jingwen Ye, Yixin Ji, Xinchao Wang, Xin Gao, and Mingli Song. Data-free knowledge amalgamation via group-stack dual-gan. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 2

[38] Jingwen Ye, Yixin Ji, Xinchao Wang, Kairi Ou, Dapeng Tao, and Mingli Song. Student becoming the master: Knowledge amalgamation for joint scene parsing, depth estimation, and more. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 2

[39] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7130–7138, 2017. 2, 5, 6

[40] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *ArXiv*, abs/1605.07146, 2016. 5

[41] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR*, 2017. 1, 2, 5, 6

[42] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision 2014*, 2014. 2

[43] X. Zhang, X. Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6848–6856, 2018. 5