Proto2Proto: Can you recognize the car, the way I do?

Monish Keswani Sriranjani Ramakrishnan Nishant Reddy Vineeth N Balasubramanian
Indian Institute of Technology, Hyderabad
{monish.keswani01, sriranjani.ramakrish, s.nishantreddy024}@gmail.com, vineethnb@iith.ac.in

Abstract

Prototypical methods have recently gained a lot of attention due to their intrinsic interpretable nature, which is obtained through the prototypes. With growing use cases of model reuse and distillation, there is a need to also study transfer of interpretability from one model to another. We present Proto2Proto, a novel method to transfer interpretability of one prototypical part network to another via knowledge distillation. Our approach aims to add interpretability to the "dark" knowledge transferred from the teacher to the shallower student model. We propose two novel losses: "Global Explanation" loss and "Patch-Prototype Correspondence" loss to facilitate such a transfer. Global Explanation loss forces the student prototypes to be close to teacher prototypes, and Patch-Prototype Correspondence loss enforces the local representations of the student to be similar to that of the teacher. Further, we propose three novel metrics to evaluate the student's proximity to the teacher as measures of interpretability transfer in our settings. We qualitatively and quantitatively demonstrate the effectiveness of our method on CUB-200-2011 and Stanford Cars datasets. Our experiments show that the proposed method indeed achieves interpretability transfer from teacher to student while simultaneously exhibiting competitive performance. The code will be made available at https://github.com/archmaester/proto2proto

1. Introduction

Interpretability in machine learning helps humans understand the reasoning behind a model in arriving at a particular decision. From an end-user's perspective, interpretability increases the trust in the models, eventually leading to better machine learning systems, especially the ones involved in making high-stake decisions. Lipton [28] points out various desiderata for interpretability like trust, causality, transferability, etc. In the past decade, the primary focus of the vision community was on designing models to improve performance on tasks like classification, object detection, segmentation, etc. Deep learning models such as CNNs [12, 40, 44] have played a considerable role in this success. However, they are often criticized as non-interpretable due to their black-box nature.

To add interpretations to the CNN models [26], numerous efforts falling into categories of model approximation, exemplar-based, gradient-based and concept-based interpretations have been studied in recent years. In this work, we specifically focus on methods that perform model approximation using prototypes, which allow both post hoc interpretability (explaining in terms of concepts after a model is trained) and ante hoc interpretability (learning to predict and explain through prototypes jointly during training itself). ProtoPNet [6] and ProtoTree [33] are two recent methods that add interpretability to prototypical models. ProtoPNet learns class-specific prototypes, while ProtoTree learns class-agnostic prototypes to provide global and local interpretability. However, the efforts in this space are nascent, with no effort yet to translate to shallower networks or transfer interpretability through the learned prototypes to other models, which could have many applications in model compression, few-shot learning, continual learning, etc.

Knowledge Distillation (KD) is a well-known technique to transfer the "dark" knowledge from the *Teacher* model to a *Student* network. Many works [1, 7, 9, 13–16, 23, 34, 35, 37, 46, 47, 51, 53] have focused on improving the performance and faithfulness of the student model to the teacher model in terms of accuracy, without much focus on the interpretability aspect. Liu *et al*. [29] distilled a black-box deep learning model to a decision tree to make it more interpretable. Song *et al*. [41] constructed an intermediate decision tree to capture the intrinsic problem-solving process of the teacher and transfer it to a student. The goal of these few works was to add interpretability to a black-box teacher model through knowledge distillation. We, on the other hand, consider an already implcitly interpretable teacher model and show how our distillation method can retain faithfulness of interpretability while transferring to a student model.

To this end, we present ***Proto2Proto***, a novel method to transfer the interpretability of one prototypical part network

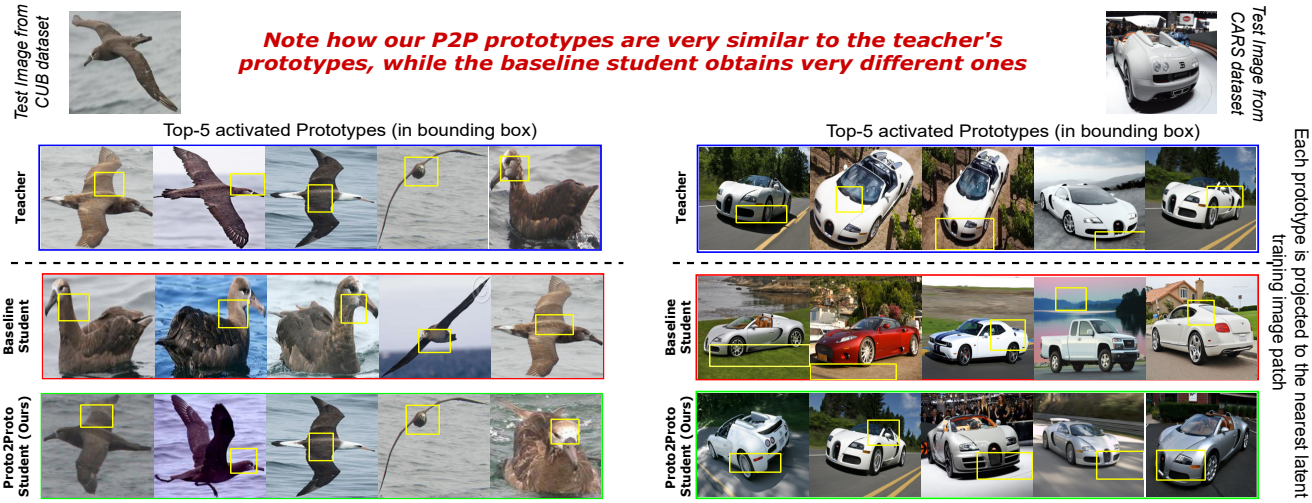

Figure 1. Comparison of sample prototypes of test image between Teacher, Baseline Student and Proto2Proto (P2P) Student.

to another. We consider a *Teacher* network, whose interpretability we would like to transfer to a shallower *Student* network. Prototype in this paper refers to a latent representation of a training image with a smaller spatial dimension called a *patch*. It represents the prototypical part of images allowing finer comparisons similar to the prototypes defined in ProtoPNet [6]. Fig. 1 illustrates the motivation of our approach. For a given test image, we visualize top-k prototypes, which play the most significant role in the decision making. We compare these prototypes of teacher, baseline student and our student. As evident, our student is more faithful to the teacher in retaining similar prototypes to make decisions compared to baseline student.

In particular, we propose two novel losses: ***Global Explanation loss*** and ***Patch-Prototype Correspondence loss*** to achieve our objective of transferring the interpretability of the teacher to the student. In prototypical networks [6, 33, 39], knowledge is stored in the prototypes that these models learn. These prototypes can act as global explanations for the model, i.e., irrespective of the input, the model can tell which parts/regions it may focus on to make decisions. Global Explanation loss helps to transfer these global explanations or prototypes to the student.

Similarly, for a given input, local representations obtained from the model are compared with the prototypes to determine which prototypes are present in the image. Based on the activations of prototypes, the model recognizes the image. Hence, it becomes important to generate similar activations of prototypes, for a given input, to recognize an image like the teacher. Patch-Prototype Correspondence loss helps to achieve this objective. It mimics the local representations of the teacher for which prototypes become active. Unlike [37], which mimics the entire feature map of a teacher for knowledge transfer, we propose to mimic local representations of the teacher that activate prototypes.

Since this is the first such effort, to validate whether we have achieved our objective, we propose three new metrics: (i) ***Average number of Active Patches (AAP)***, which determines the average number of local representations which are active for the model. It is used to evaluate the Patch-Prototype Correspondence, with a motive of bringing this value of the student close to the teacher; (ii) ***Average Jaccard Similarity of Active Patches with Teacher (AJS)***, which determines the overlap of the active local representations of the student to the teacher. It is calculated for a pair of models, namely, teacher and student. The higher its value, the closer the student is to the teacher. It is also used to evaluate the Patch-Prototype Correspondence; and (iii) ***Prototype Matching Score (PMS)***, which evaluates how close the prototypes of the student are w.r.t. the teacher. It is used to evaluate the transfer of Global Explanations.

We summarize our contributions as follows:

- To the best of our knowledge, we present the first attempt to transfer *interpretability* from a prototypical teacher to a student model.
- We propose two novel losses, *Global Explanation loss and Patch-Prototype Correspondence loss* for the knowledge transfer. We show that with our approach, the final layer decision module of a teacher can be used for the student directly as is, without relearning.
- We propose three evaluation metrics to determine the faithfulness of the student to the teacher in terms of interpretability.
- We perform a comprehensive suite of experiments on benchmark datasets which show the effectiveness of our method.

2. Related Works

2.1. Interpretability

Interpretability for machine learning models can be provided as post-hoc explanations or as self-explanations. While the former gives intuition about the trained black-box model [8, 19, 21, 22, 32, 55], the latter tries to understand the complex decision process by modifying architecture during training [2, 38, 54]. Existing interpretable models can be divided into four major types [26], namely, gradient-based, model-approximation based, conceptual interpretation and example-based methods. Our focus in this paper is on model-approximation and example-based methods. Either globally or locally, the model-approximation methods approximate the representations using a self-explanatory model such as linear models and decision trees. Local models like LIME [36] focus on local similarity neighbourhood. While global models like soft decision trees [17, 24, 43], adaptive neural trees [45] approximate the entire deep neural models. On the other hand, example-based methods [11, 18] compare input images with exemplar images to interpret a single input image. Since exemplars are too specific, prototypical models [4, 20] approximate the model within a set of prototypes. These learned prototypes do not focus much on the decision process; their capacity is limited interpretability. By combining model approximation methods with prototype-based models, performance and interpretability can be handled.

Recently proposed models like ProtoPNet [6], ProtoTree [33] use the above concept to improve interpretability and focus on model's decision process. ProtoPNet learns class-specific prototypes representing parts of a class approximated by a linear model for decision making. ProtoTree learns class-agnostic prototypes approximated by a decision tree making the architecture hierarchical. Our proposed work focuses on transferring the interpretable knowledge stored in these prototypes to a shallower network.

2.2. Interpretable Knowledge Distillation

A lot of works focus on the design and development of small models applicable in a resource-constraint deployment. Knowledge Distillation is one such model compression method to improve the performance of small student models using a large teacher model. It distils the dark knowledge by mimicking the logits [3], or soft labels [14] from teacher to student. Survey papers on knowledge distillation covering distillation strategies, student-teacher architectures and the recent findings can be viewed to get the background on this setup [10, 50]. Even though a plethora of literature for knowledge distillation is present, very few works have been proposed to focus on the interpretability aspect of knowledge distillation.

Interpretability in knowledge distillation is commonly achieved by transferring the teacher's dark knowledge into interpretable tree-based models in one form or the other. Post-hoc interpretations were obtained for the dark knowledge by varying the inputs to trees such as matching logits [30], soft targets or using different types of trees architecture viz soft decision trees [17], vanilla decision trees , adaptive neural trees [45], Neural backed decision trees [49], Gradient boosting trees [5] and generalized additive models [31]. Instead of using representations from the deep neural networks, Tree-Network-Tree architecture [27] attempts to learn a tree-based model in the input space to extract the decision path and form an embedding representation. This is further used to learn a neural network whose soft labels are used to distill knowledge into another tree-based model. This three-step process helps in making the model interpretable due to the extractable decision paths from the distilled tree. Most of the attempts above focus on relieving the tension between accuracy and interpretability. Due to the constraints on input/weight space, the knowledge could not be completely distilled into the student, or the distilled models could not be used to their full capability.

Another line of thought is to use visualization methods for interpretation. DarkSight [52] uses a simple interpretable classifier such as Naive Bayes' as a student model to mimic the dark knowledge. Applying low dimensional representation to data and jointly optimizing on model compression objective, the visualizations obtained on the network's predictions provides interpretability.

One close work to our proposed method attempts to mimic the decision-process of the teacher in a layer-wise manner, instead of the teacher's output [42]. They construct a decision tree using agglomerative clustering on the layers of the teacher model and use it to train the student model. Knowing the decision process in each layer makes the student model interpretable.

Our proposed work differs from the all above methods in transferring the dark knowledge (i) stored in the form of prototypes (ii) even without re-learning the decision module of the student. We also retain the faithfulness to the teacher in terms of interpretability. It also mimics the decision process made by the teacher, hence the student model can be trained to its full capacity without forgoing the performance. In terms of interpretability, the model is inherently interpretable due to the usage of an interpretable model for training a teacher.

3. Proto2Proto: Proposed Methodology

Prototypical methods [6, 33, 39] learn a tuple of prototypes $\boldsymbol{P} = (p_i)_{i=1}^{m}$, where $p_i \in \mathbb{R}^d$, which are representations of the most important regions in the training dataset. They facilitate a refined recognition of images. These methods usually contain four modules: backbone, add-on, comparison and decision module. The input image x is first

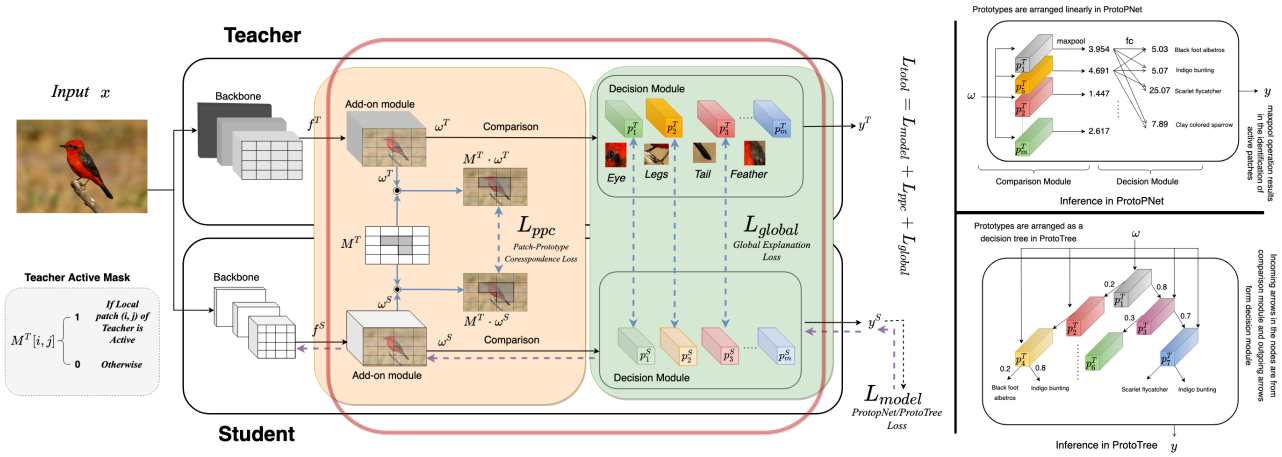

Figure 2. Our proposed architecture for training a "Proto2Proto" Student model via Knowledge Distillation. It shows information flow between teacher and student, as well as the inference processes (**black** arrows: forward information flow, **purple** arrows: backpropagation in student, **blue** arrows: loss terms introduced in our work for teacher-student alignment).

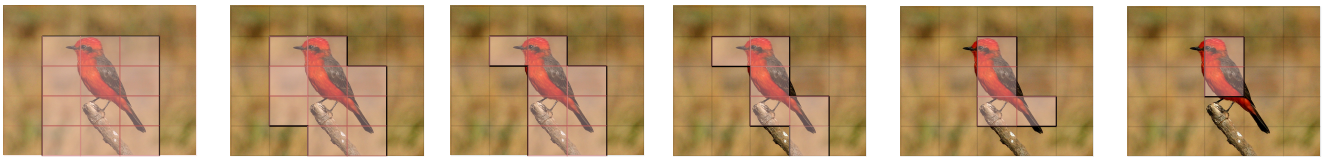

Figure 3. Effect on *Active Patches* by varying τ using L_2 as a metric. The binary activation mask M is upscaled and overlayed on the image for visualization, for different values of τ. The highlighted part of the image shows the active patches.

Notations	Description
$f(x)$	Backbone features for input x
$\omega(x)$	Add-on features (matched against the prototypes) for input x
$\boldsymbol{P}$	A tuple of prototypes
p	Single prototype
M	Binary mask of active patches
$\mathscr{L}(x)$	Set of local patches for input x
$l_{ij}(x)$	ij^{th} local patch for input x
τ	Distance threshold to determine an active patch
C	Number of classes

Table 1. Notations defined in the paper. The notations are superscripted by T or S to denote teacher or student, respectively.

passed on to the backbone module to obtain a feature map $f(x)$ which is then passed on to an add-on module to obtain a feature map $\omega(x) \in \mathbb{R}^{H*W*d}$. The add-on module outputs HW local representations or patches of dimension d. We denote them by a set $\mathscr{L}(x) = \{l_{11}(x), l_{12}(x), ...l_{HW}(x)\}$, where $l_{ij}(x) \in \mathbb{R}^d$. Note that the dimensionality of prototypes is same as local representations, i.e., d. The comparison module then compares $\mathscr{L}(x)$ with $\boldsymbol{P}$ using similarity scores. The scores between the prototypes and their nearest patch are then passed on to the decision module. In ProtoPNet [6], the decision module is a fully connected layer with dimensionality $|\boldsymbol{P}| * C$, where C is the number of classes. Also, each prototype belongs to a particular class which makes the weights of the decision module sparse. ProtoPShare [39] improves on ProtoPNet and additionally merges semantically similar prototypes using novel data-dependent merge-pruning algorithm. In ProtoTree [33], the decision module is a decision tree where the prototypes are arranged as nodes of the decision tree. For clarity, the notations with descriptions are mentioned in Table 1.

A typical information flow in prototypical models is explained as follows: a) **Inference in ProtoPNet :-** Input image x is passed through CNN to obtain a set of features $\mathscr{L}(x)$. For each prototype p, we compute its L_2 distance with set $\mathscr{L}(x)$, the distances are then inverted to get similarity scores and the maximum similarity score is then computed using maxpooling. $|\boldsymbol{P}|$ similarity scores thus obtained are then fed to fully connected layer of size $|\boldsymbol{P}| * C$ to get the output logits. b) **Inference in ProtoTree :-** Here, prototypes are arranged as internal nodes of a *soft, binary* decision tree (user-provided structure). Maximum similarity scores, obtained as above are then normalized to $[0, 1]$ which act as probability values. For each prototype node, routing is done based on comparing max similarity score with pre-set threshold values. Leaf nodes store the classification predictions.

In our setup, we transfer the knowledge of a trained prototypical teacher model to a student model not only to improve the performance of the student in terms of accuracy but also to bring it closer to the teacher in terms of interpretability. We denote a *learned* tuple of prototypes of teacher model by $\boldsymbol{P^T}$ and a *learnable* tuple of prototypes of student model by $\boldsymbol{P^S}$. The global interpretability of the teacher is contained in its prototypes since they implicitly represent the parts/regions the model focuses on to make the decisions. For input image x, the local interpretability depends on the pair $(\mathscr{L}^T(x), \boldsymbol{P^T})$, since the decisions are taken based on the distance between $\boldsymbol{P^T}$ and local representations $\mathscr{L}^T(x)$. In order to transfer this knowledge, it is necessary that the student agrees with the teacher on local representations and prototypes. To achieve this, we propose two losses: *Global Explanation loss* and *Patch-Prototype Correspondence loss*. We first define active patches to determine which local representations are associated with the prototypes. Next, we define Patch-Prototype Correspondence loss, which forces the student to agree with the teacher on active local representations. Finally, we define Global Explanation loss which forces the student to agree with the teacher on the prototypes.

Active Patch: In comparison module, each prototype is associated with the closest local patch and only the associated local patches are involved in the decision making. We call such patches as *active*. For input x, let $k(x) \in \mathscr{L}(x)$ be a local patch. We define a function $Active$ for local patch $k(x)$ as below.

$$\begin{aligned} &\text{If } \exists p \in \boldsymbol{P} \text{ s.t. } D(k(x), p) = D^* \leq \tau \\ &\text{where } D^* = \min_{i,j} D(l_{ij}(x), p) \\ &\text{and } i, j \in \{1, 2, ...H\}, \{1, 2, ...W\} \end{aligned} \tag{1}$$

$$Active(k(x), \tau, \boldsymbol{P}) = \begin{cases} 1 & \text{if Eq. (1) is satisfied} \\ 0 & \text{Otherwise} \end{cases} \tag{2}$$

where τ is a distance threshold and $\boldsymbol{P}$ is a tuple of prototypes as defined earlier. The local patch $k(x)$ is said to be active if there exists a prototype $p \in \boldsymbol{P}$ such that the distance of p from $k(x)$ is minimum among all local patches $\mathscr{L}(x)$ and less than τ. The hyperparameter τ controls the maximum distance allowed to define an active patch. Fig. 3 shows the effect of τ on the number of active patches. As observed, the number of active patches decrease with the value of τ. We refer to τ as τ_{train} during training and τ_{test} during testing.

Patch-Prototype Correspondence Loss: Romero *et al.* [37] proposed to mimic the local representations of the teacher as hints to the student to improve the performance of the student. We propose to transfer only the active local representations of the teacher's add-on layer to the student to improve its interpretability as well. We identify the active patches of the teacher for input x, and represent it as a binary activation mask M^T which is defined as:

$$\begin{aligned} M^T(x)[i,j] = Active(l_{ij}(x), \tau_{train}, \boldsymbol{P^T}) \\ \forall i, j \in \{1, 2, ...H\}, \{1, 2, ...W\} \end{aligned} \tag{3}$$

We now define the Patch-Prototype Correspondence loss as:

$$L_{ppc} = \frac{1}{N} \sum_{n=1}^{N} \left\| M^T(x_n) \cdot [\omega^T(x_n) - \omega^S(x_n)] \right\|_2 \tag{4}$$

where N is the number of training images, $\omega^T(x_n)$ and $\omega^S(x_n)$ are the outputs of the add-on layer of teacher and student respectively, for input image x_n.

Global Explanation Loss: To achieve our objective, the student should agree with the teacher on the prototypes. Global Explanation loss forces the prototypes of the student model to be close to that of the teacher model. We define it as:

$$L_{global} = \frac{1}{m} \sum_{i=1}^{m} D(p_i^T, p_i^S) \tag{5}$$

where D is a distance metric (cosine, euclidean, etc.) and m is the number of prototypes. We term this as the Global Explanation loss, because the prototypes are globally interpretable, i.e, without any input we can tell which region they are focusing on. We use Euclidean distance as a metric in our experiments.

Model Loss: It denotes the corresponding losses of the prototypical part methods [6, 33]. For ProtoPNet, we refer to L_{model} as L_{ppnet} and for ProtoTree, L_{ptree}.

The total loss is given by:

$$L_{total} = L_{model} + \lambda_{global} L_{global} + \lambda_{ppc} L_{ppc} \tag{6}$$

where λ_{global} and λ_{ppc} are the hyperparameters which are used to balance the losses.

3.1. Evaluating Interpretability Transfer

Since this is the first such effort on interpretability transfer on prototypical networks, we also propose three evaluation metrics to determine the proximity of the student model to the teacher model in terms of interpretability. Note that the aim of our evaluation metrics is not to evaluate the interpretability of individual models but to evaluate interpretability transfer between teacher and student models.

Average number of Active Patches (AAP): We had earlier stated that the student should agree with the teacher on local representations and prototypes. To achieve agreement of the former, we introduced L_{ppc} loss. AAP is one of the

metrics which evaluates L_{ppc}. We define the average number of active patches of model m as:

$$\text{AAP}(\tau_{test}, \boldsymbol{P}) = \frac{1}{N}\sum_{n=1}^{N}\sum_{h=1}^{H}\sum_{w=1}^{W} Active(l_{hw}(x_n), \tau_{test}, \boldsymbol{P}) \tag{7}$$

where N is the number of images, τ_{test} is the distance threshold, $\boldsymbol{P}$ is a tuple of prototypes of model m (ideally, the notation should be $\boldsymbol{P^m}$, for simplicity we ignore m), and $Active$ is defined in Eq. (2). The closer the value of AAP of student to that of teacher, the better the interpretability transfer.

Average Jaccard Similarity of Active Patches with Teacher (AJS): For AAP score, we counted the number of active patches. Here, we determine the overlap of the active patches of student with teacher to find out the similarity between them. AAP score is calculated for individual models whereas AJS is calculated for a (student, teacher) pair. We assign a unique identifier to all active patches of an image. For image x, $\mathcal{A}(x) = \{..., id_{ij}(x), ...\}$, where $id_{ij}(x) = \textit{UNIQUE-ID}(ij)$ and $Active(l_{ij}(x), \tau_{test}, \boldsymbol{P}) = 1$. We define Average Jaccard Similarity between active patches of student $\boldsymbol{S}$ and teacher $\boldsymbol{T}$ as follows:

$$\text{AJS}(\boldsymbol{S}, \boldsymbol{T}) = \frac{1}{N}\sum_{n=1}^{N}\frac{\left|\mathcal{A}^T(x_n) \cap \mathcal{A}^S(x_n)\right|}{\left|\mathcal{A}^T(x_n) \cup \mathcal{A}^S(x_n)\right|} \tag{8}$$

where N is the number of images. $\mathcal{A}^T$ and $\mathcal{A}^S$ are the active patches of teacher and student, respectively. Note that $\text{AJS}(\boldsymbol{T}, \boldsymbol{T}) = 1$ serves as the target upper bound for the student model.

Prototype Matching Score (PMS): Now, we define a metric to evaluate L_{global}. The intuition is to compute a matching score to measure the closeness between prototypes of teacher and student. To compute this score, the exact correspondence between teacher & student prototypes is required, which is unknown. Hence, we use Hungarian matching algorithm (HMA) to match the prototypes. Note that HMA may not be required to match teacher prototypes with our student as there will be a sequential mapping between the two (Eq. (5)) but is needed for baseline student.

The prototypes across models are distributed in different spaces, a direct comparison between the two using a distance metric is not trivial. Hence, we determine the similarity of prototypes across models in terms of local patches which activates the corresponding prototype. In AJS, we maintained a list of active patches for a given input image. Here, we maintain a list of active patches across all the images, for each prototype, to determine the similarity of prototypes of two models. We use *Modified-Jaccard-Similarity* as a distance metric to compare the prototypes of the teacher and the student. Algorithm 1 summarizes the overall idea. Please refer Supplementary material for details on *Modified-Jaccard-Similarity*.

Algorithm 1: Prototype Matching Score (PMS)

Input: T - Teacher, S - Student, D - Test/Val Dataset
Output: Prototype Matching Score (PMS)

$numSamples = |D|$
Initialize Teacher Prototype list, $Q^T = \{q_1^T, ..., q_m^T\}$, where $q_i^T = \phi\ \forall i \in \{1, 2, ...m\}, \left|\boldsymbol{P^T}\right| = m$
Initialize Student Prototype list, $Q^S = \{q_1^S, ..., q_m^S\}$, where $q_i^S = \phi\ \forall i \in \{1, 2, ...m\}, \left|\boldsymbol{P^S}\right| = m$
for $i = 1 : numSamples$ **do**
  for $j = 1 : m$ **do**
    $k^T = \text{argmin}_{h,w}\ D(l_{hw}^T(x_i), p_j^T)$
    $q_j^T = q_j^T \cup \text{UNIQUE-ID}(k^T, x_i)$
    $k^S = \text{argmin}_{h,w}\ D(l_{hw}^S(x_i), p_j^S)$
    $q_j^S = q_j^S \cup \text{UNIQUE-ID}(k^S, x_i)$
  end
end
score-matrix = *Modified-Jaccard-Similarity*(Q^T, Q^S)
matching-scores = *Hungarian*(score-matrix)
return *Average*(matching-scores)

4. Experimental Results

We demonstrate our proposed method and perform experiments on the fine-grained classification benchmark datasets CUB-200-2011 [48] and Stanford Cars [25]. To have fair comparison, our experimental setting is similar to the setup in ProtoPNet [6] and ProtoTree [33]. We have experimented with various architectures like ResNet [12] and VGG [40]. For each setting, we have the results of teacher, baseline student and Proto2Proto student (Ours). The models are evaluated on interpretability using AAP, AJS and PMS as defined in Section 3.1. Apart from this, we also evaluate all the models based on top-1 accuracy since we do not want to compromise on accuracy in favour of interpretability. The architectures ResNet50, and VGG19 were used for the teacher model and ResNet18, Resnet34, and VGG11 were used for the student model. Hyperparameter details can be found in the Supplementary Material.

Table 2 shows the performance of our proposed approach. Our student outperforms the baseline student in all our experiments. It is able to achieve on par or even exceed the teacher's performance in some cases. For example, VGG19 $\rightarrow$ VGG11 (KD), our method gives a 5.83% absolute increment in accuracy compared to the baseline student. A similar trend is observed for ResNet architectures on CUB and Stanford Cars. The evaluation metrics AJS will be 1 for the teacher model since the model is similar to itself. This metric shows how close the student model is to the teacher model. As seen from the results, our proposed student model is closer to the teacher in all settings compared to the baseline student. A similar argument holds for PMS, how close the student prototypes are with the teacher

Datasets	Method	Setting	AAP	AJS (↑)	PMS (↑)	Top-1 Accuracy (↑)
CUB	ProtopNet	VGG19 (Teacher)	29.10	1.0	1.0	77.97
	ProtopNet	VGG11 (Student)	37.92	0.58	0.36	71.62
	Ours	VGG19 → VGG11 (KD)	29.29	0.73 **(+0.15)**	0.81 **(+0.45)**	77.45 **(+5.83)**
	ProtopNet	Resnet50 (Teacher)	20.24	1.0	1.0	79.22
	ProtopNet	Resnet18 (Student)	39.77	0.42	0.18	75.47
	Ours	Resnet50 → Resnet18 (KD)	19.23	0.71 **(+0.29)**	0.74 **(+0.56)**	79.80 **(+4.33)**
	ProtopNet	Resnet50 (Teacher)	20.24	1.0	1.0	79.22
	ProtopNet	Resnet34 (Student)	18.17	0.30	0.16	78.31
	Ours	Resnet50→ Resnet34 (KD)	19.33	0.73 **(+0.43)**	0.79 **(+0.63)**	79.89 **(+1.6)**
CARS	ProtopNet	Resnet50 (Teacher)	29.22	1.0	1.0	85.31
	ProtopNet	Resnet18 (Student)	32.67	0.45	0.14	79.96
	Ours	Resnet50 → Resnet18 (KD)	29.61	0.62 **(+0.17)**	0.72 **(+0.58)**	84.00 **(+4.04)**
	Prototree	Resnet50 (Teacher)	21.35	1.0	1.0	85.70
	Prototree	Resnet18 (Student)	23.60	0.46	0.12	77.87
	Ours	Resnet50 → Resnet18 (KD)	21.55	0.59 **(+0.13)**	0.65 **(+0.53)**	81.50 **(+3.63)**

Table 2. Results of Proto2Proto student (Ours) on ProtoPNet [6], ProtoTree [33] for multiple architectures like ResNet, VGG experimented on CUB and Stanford Cars. Evaluated performance using Top-1 Accuracy and interpretability using metrics AAP, AJS and PMS

prototypes. The value will be 1.0 for the teacher model itself. For example, for CUB, ResNet50 → ResNet18 (KD), the increase is 0.56 in absolute values between baseline student and proposed student. The strength of our proposed student comes in the interpretability aspect along with performance. This is observed in all the experimental results as the metric AAP, AJS, and PMS are improving for our student compared to the baseline student model. For example, in ResNet50 → ResNet34 (KD), even though the performance difference is much less than other models, AJS and PMS metrics shows an increase of 0.43 and 0.63 in absolute values, respectively. Hence the proposed model is interpretable without forgoing accuracy. Additional results on the above experiments can be found in the supplementary material.

Method/Setting	$\|\boldsymbol{P}\|$	AAP	AJS (↑)	PMS (↑)	Acc. (↑)
Baseline + PShare	960	10.69	0.20	0.13	70.15
Ours + PShare	960	13.69	0.33	0.70	74.01
			(+0.13)	**(+0.57)**	**(+3.86)**

Table 3. Results of pruning ResNet18 baseline student and our student model pruned using ProtoPShare [39] (PShare) on Cars

4.1. Ablation studies

Ablation on losses We perform experiments with various combinations of losses to demonstrate the effectiveness of our approach. Table 4 summarizes our results. As evident, adding the individual losses improves the performance on all metrics and all three losses combined performs even better. We additionally perform experiments by reusing the teacher's decision module for the student. It significantly boosts interpretability scores and accuracy. Notice that we get a minor benefit in interpretability scores by removing L_{ppnet}. However, the accuracy is more with L_{ppnet}. Hence, we use the last setting in our experiments.

L_{ppnet}	L_{ppc}	L_{global}	Reuse	AAP	AJS	PMS	Accuracy
✓				39.77	0.42	0.18	75.47
✓		✓		31.26	0.49	0.31	75.19
✓	✓			30.59	0.54	0.61	77.63
✓	✓	✓		**20.89**	0.70	0.69	78.11
	✓	✓	✓	19.29	**0.72**	**0.76**	79.44
✓	✓	✓	✓	19.23	0.71	0.74	**79.80**
			Teacher	20.24	1.0	1.0	79.20

Table 4. Performance of ResNet18 student on different losses on CUB dataset using ResNet50 teacher . The column "Reuse" indicates whether we use teachers decision module for student.

Ablation on pruning ProtoPShare [39] introduced a novel prototype-merging strategy to join semantically similar prototypes. They significantly reduced the number of prototypes of ProtoPNet model without much drop in accuracy. In Table 3, we summarize the results of pruning. We prune the baseline student and Proto2Proto student from 2000 to 960 prototypes, using ProtoPshare. We observe that our Proto2Proto model significantly performs better than baseline model even after pruning.

Figure 4. Comparison of prototypes (each prototype projected to the nearest latent training image patch) of Teacher and Student (Baseline and Proto2Proto). The Student prototypes are matched with that of Teacher's via Hungarian Matching Algorithm.

Comparison with KD methods Since our focus is on adding interpretability via knowledge distillation, to demonstrate efficacy of our method, we have compared against Relational Knowledge distillation (RKD) [34] and Hint Loss [37]. Baseline student is trained with ProtoTree/ProtoPNet loss *only*. Our method performs significantly well on interpretability scores as compared to the existing KD methods. In terms of accuracy, we get a minor improvement over RKD but on interpretability scores, we perform significantly better. This shows that our method performs on par with existing KD methods and enjoys interpretability for distilled dark knowledge. Additional abalation studies can be found in supplementary.

Setting	AAP	AJS (↑)	PMS (↑)	Acc. (↑)
Teacher	29.22	1.0	1.0	85.31
Baseline Student	32.67	0.45	0.14	79.96
Hint [37]	28.13	0.48	0.15	81.52
RKD [34]	30.85	0.53	0.27	83.31
Ours	**29.61**	**0.62**	**0.72**	**84.00**

Table 5. Comparison with state-of-the-art Knowledge Distillation methods on Cars with Resnet50 (Teacher) and Resnet18 (Student)

5. Visualization

Figure 4 shows the prototypes learnt by the teacher, baseline student and Proto2Proto student. As discussed before, the prototypes of our student model are closer to the teacher model compared to the baseline student. For example, the prototype 1 of teacher focuses on the windshield of the car and prototype 2 focuses on the bonnet, which are also captured by the corresponding prototypes of Proto2Proto student model. The baseline student model, however, focuses on the background instead of the windshield and bonnet. As shown in Figure 1, the top-5 activated prototypes of our proposed student are similar to that of teacher's as they focuses on the wings of the bird compared to the baseline student which focuses on the neck for a given test image. This shows that our Proto2Proto model can mimic the teacher well, and the resulting prototypes retain faithfulness to the teacher compared to the baseline student. For visualizing the prototypes for ProtoPNet decision module, similar approach of using highly activated train image patch of the learnt prototype and upsampling it as described in [6] is used. Please refer Supplementary Material for more visualizations on ProtoTree/ProtoPNet for CUB and Cars.

6. Conclusions and Future Work

Recent approaches have focused on designing implicit interpretable models. However, they lack the capacity to transfer knowledge in terms of interpretability. We proposed a novel framework to transfer the interpretability of teacher to student, to aid the student in making decisions like teacher. Further, we proposed three evaluation metrics to demonstrate the efficacy of our approach and reported significant performance improvement across all metrics on our student model. Extending our approach to other tasks like continual learning, transfer learning, can be an excellent future direction.

Broader Impact and Limitations. Our work has no known social detrimental impact. In the current setup, we require the teacher and student to have same number of prototypes. However, the number of parameters added by the prototypes are often very less as compared to the feature extractor. Also, as shown in Section 4.1, existing pruning strategies work on our student as well which partially addresses this issue. The proposed Prototype Matching Score uses jaccard similarity to compare the prototypes of different models. A more straightforward way would be to use a distance metric (Euclidean, cosine, etc.). However, it is non-trivial to apply such a distance metric in our setting. Prototypical models are usually ensembled at the logits layer to obtain improved performance. Such a setup will require to evaluate interpretability between ensemble of teachers and ensemble of students. A further study is required to apply the proposed evaluation metrics in such a setup.

Acknowledgements. This work has been partly supported by the funding received from DST, Govt of India, through the ICPS program as well as a Google Research Scholar Award. We thank the anonymous reviewers for their valuable feedback that improved the presentation of this paper.

References

[1] Sungsoo Ahn, Shell Xu Hu, Andreas Damianou, Neil D Lawrence, and Zhenwen Dai. Variational information distillation for knowledge transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9163–9171, 2019. 1

[2] David Alvarez Melis and Tommi Jaakkola. Towards robust interpretability with self-explaining neural networks. *Advances in neural information processing systems*, 31, 2018. 3

[3] Lei Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? *arXiv preprint arXiv:1312.6184*, 2013. 3

[4] Jacob Bien and Robert Tibshirani. Prototype selection for interpretable classification. *The Annals of Applied Statistics*, 5(4):2403–2424, 2011. 3

[5] Zhengping Che, Sanjay Purushotham, Robinder Khemani, and Yan Liu. Distilling knowledge from deep networks with applications to healthcare domain. *arXiv preprint arXiv:1512.03542*, 2015. 3

[6] Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This looks like that: Deep learning for interpretable image recognition. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 1, 2, 3, 4, 5, 6, 7, 8

[7] Yiheng Chi, Abhiram Gnanasambandam, Vladlen Koltun, and Stanley H Chan. Dynamic low-light imaging with quanta image sensors. In *European Conference on Computer Vision*, pages 122–138. Springer, 2020. 1

[8] Amirata Ghorbani, James Wexler, James Y Zou, and Been Kim. Towards automatic concept-based explanations. *Advances in Neural Information Processing Systems*, 32, 2019. 3

[9] Abhiram Gnanasambandam and Stanley H Chan. Image classification in the dark using quanta image sensors. In *European Conference on Computer Vision*, pages 484–501. Springer, 2020. 1

[10] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021. 3

[11] Yash Goyal, Ziyan Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. Counterfactual visual explanations. In *International Conference on Machine Learning*, pages 2376–2384. PMLR, 2019. 3

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1, 6

[13] Byeongho Heo, Minsik Lee, Sangdoo Yun, and Jin Young Choi. Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3779–3787, 2019. 1

[14] Geoffrey E. Hinton, Oriol Vinyals, and J. Dean. Distilling the knowledge in a neural network. *ArXiv*, abs/1503.02531, 2015. 1, 3

[15] Guanzhe Hong, Zhiyuan Mao, Xiaojun Lin, and Stanley H Chan. Student-teacher learning from clean inputs to noisy inputs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12075–12084, 2021. 1

[16] Ming Hong, Yuan Xie, Cuihua Li, and Yanyun Qu. Distilling image dehazing with heterogeneous task imitation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3462–3471, 2020. 1

[17] Ozan Irsoy, Olcay Taner Yıldız, and Ethem Alpaydın. Soft decision trees. In *Proceedings of the 21st international conference on pattern recognition (ICPR2012)*, pages 1819–1822. IEEE, 2012. 3

[18] Rajiv Khanna, Been Kim, Joydeep Ghosh, and Sanmi Koyejo. Interpreting black box predictions using fisher kernels. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 3382–3390. PMLR, 2019. 3

[19] Been Kim, Rajiv Khanna, and Oluwasanmi O Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. *Advances in neural information processing systems*, 29, 2016. 3

[20] Been Kim, Rajiv Khanna, and Oluwasanmi O Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. *Advances in neural information processing systems*, 29, 2016. 3

[21] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, pages 2668–2677. PMLR, 2018. 3

[22] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR, 2017. 3

[23] Nikos Komodakis and Sergey Zagoruyko. Paying more attention to attention: improving the performance of convolutional neural networks via attention transfer. In *ICLR*, 2017. 1

[24] Peter Kontschieder, Madalina Fiterau, Antonio Criminisi, and Samuel Rota Bulo. Deep neural decision forests. In *Proceedings of the IEEE international conference on computer vision*, pages 1467–1475, 2015. 3

[25] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013. 6

[26] Peter Cho-Ho Lam, Lingyang Chu, Maxim Torgonskiy, Jian Pei, Yong Zhang, and Lanjun Wang. Finding representative interpretations on convolutional neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1345–1354, 2021. 1, 3

[27] Jiawei Li, Yiming Li, Xingchun Xiang, Shu-Tao Xia, Siyi Dong, and Yun Cai. Tnt: An interpretable tree-network-tree learning framework using knowledge distillation. *Entropy*, 22(11):1203, 2020. 3

[28] Zachary C Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57, 2018. 1

[29] Xuan Liu, Xiaoguang Wang, and Stan Matwin. Improving the interpretability of deep neural networks with knowledge distillation. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 905–912. IEEE, 2018. 1

[30] Xuan Liu, Xiaoguang Wang, and Stan Matwin. Improving the interpretability of deep neural networks with knowledge distillation. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 905–912. IEEE, 2018. 3

[31] Yin Lou, Rich Caruana, and Johannes Gehrke. Intelligible models for classification and regression. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 150–158, 2012. 3

[32] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017. 3

[33] Meike Nauta, Ron van Bree, and Christin Seifert. Neural prototype trees for interpretable fine-grained image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14933–14943, 2021. 1, 2, 3, 4, 5, 6, 7

[34] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3967–3976, 2019. 1, 8

[35] Baoyun Peng, Xiao Jin, Jiaheng Liu, Dongsheng Li, Yichao Wu, Yu Liu, Shunfeng Zhou, and Zhaoning Zhang. Correlation congruence for knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5007–5016, 2019. 1

[36] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016. 3

[37] 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, 8

[38] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019. 3

[39] Dawid Rymarczyk, Łukasz Struski, Jacek Tabor, and Bartosz Zieliński. Protopshare: Prototypical parts sharing for similarity discovery in interpretable image classification. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '21, page 1420–1430, New York, NY, USA, 2021. Association for Computing Machinery. 2, 3, 4, 7

[40] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1, 6

[41] Jie Song, Haofei Zhang, Xinchao Wang, Mengqi Xue, Ying Chen, Li Sun, Dacheng Tao, and Mingli Song. Tree-like decision distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13488–13497, June 2021. 1

[42] Jie Song, Haofei Zhang, Xinchao Wang, Mengqi Xue, Ying Chen, Li Sun, Dacheng Tao, and Mingli Song. Tree-like decision distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13488–13497, June 2021. 3

[43] Alberto Suárez and James F Lutsko. Globally optimal fuzzy decision trees for classification and regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(12):1297–1311, 1999. 3

[44] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*, 2017. 1

[45] Ryutaro Tanno, Kai Arulkumaran, Daniel Alexander, Antonio Criminisi, and Aditya Nori. Adaptive neural trees. In *International Conference on Machine Learning*, pages 6166–6175. PMLR, 2019. 3

[46] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *International Conference on Learning Representations*, 2019. 1

[47] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1365–1374, 2019. 1

[48] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 6

[49] Alvin Wan, Lisa Dunlap, Daniel Ho, Jihan Yin, Scott Lee, Henry Jin, Suzanne Petryk, Sarah Adel Bargal, and Joseph E Gonzalez. Nbdt: Neural-backed decision trees. *arXiv preprint arXiv:2004.00221*, 2020. 3

[50] Lin Wang and Kuk-Jin Yoon. Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 3

[51] Guodong Xu, Ziwei Liu, Xiaoxiao Li, and Chen Change Loy. Knowledge distillation meets self-supervision. In *European Conference on Computer Vision*, pages 588–604. Springer, 2020. 1

[52] Kai Xu, Dae Hoon Park, Chang Yi, and Charles Sutton. Interpreting deep classifier by visual distillation of dark knowledge. *arXiv preprint arXiv:1803.04042*, 2018. 3

[53] Chuanguang Yang, Zhulin An, Linhang Cai, and Yongjun Xu. Hierarchical self-supervised augmented knowledge distillation. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 1217–1223. International Joint Conferences on Artificial Intelligence Organization, 8 2021. Main Track. 1

[54] Chih-Kuan Yeh, Been Kim, Sercan Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. On completeness-

aware concept-based explanations in deep neural networks. *Advances in Neural Information Processing Systems*, 33:20554–20565, 2020. 3

[55] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014. 3