DEPARA: Deep Attribution Graph for Deep Knowledge Transferability

Jie Song[1]*, Yixin Chen[1]*, Jingwen Ye[1], Xinchao Wang[2], Chengchao Shen[1], Feng Mao[3], and Mingli Song[1]
[1]Zhejiang University, [2]Stevens Institute of Technology
[3]Alibaba Group

Abstract

Exploring the intrinsic interconnections between the knowledge encoded in PRe-trained Deep Neural Networks (PR-DNNs) of heterogeneous tasks sheds light on their mutual transferability, and consequently enables knowledge transfer from one task to another so as to reduce the training effort of the latter. In this paper, we propose the DEeP Attribution gRAph (DEPARA) to investigate the transferability of knowledge learned from PR-DNNs. In DEPARA, nodes correspond to the inputs and are represented by their vectorized attribution maps with regards to the outputs of the PR-DNN. Edges denote the relatedness between inputs and are measured by the similarity of their features extracted from the PR-DNN. The knowledge transferability of two PR-DNNs is measured by the similarity of their corresponding DEPARAs. We apply DEPARA to two important yet understudied problems in transfer learning: pre-trained model selection and layer selection. Extensive experiments are conducted to demonstrate the effectiveness and superiority of the proposed method in solving both these problems. Code, data and models reproducing the results in this paper are available at `https://github.com/zju-vipa/DEPARA`.

1. Introduction

Driven by massive labeled data [5] and the developing advanced deep models [9], the field of artificial intelligence has made remarkable progress in recent years. However, in real-world scenarios we often encounter the dilemma where limited labeled training data are available for addressing our problems at hand. The common practice in this situation is transferring the pre-trained models, which are open sourced by dedicated researchers or industries, to solve our own problems. Yet, along this road comes up another problem: faced with countless PR-DNNs of various layers, which model and which layer of it should be transferred to benefit the target task most? Currently the model selection is usually done blindly by adopting the ImageNet pre-trained models [21, 15] and the layer selection is usually conducted heuristically. However, the ImageNet pre-trained models will not always produce satisfactory performances for all the tasks, especially when the task is significantly different from the one defined by ImageNet [2, 28]. Likewise, the heuristically selected layer may also perform sub-optimally, as the optimal layer for being transferred depends on various factors such as task relatedness and the amount of the target data.

To tackle the aforementioned problems, we need to explore and reveal the underlying transferability among deep knowledge from PR-DNNs of heterogeneous tasks. Recently, Zamir *et al*. [33] did the pioneering work towards this direction. They proposed a fully computational approach, termed *taskonomy*, to measure the task transferability. However, there are three unneglected limitations in taskonomy tremendously hampering its real-world application. The first is its prohibitively expensive cost in computation. For computing the pairwise relatedness for a given task dictionary, the computation cost will grow quadratically with the number of the tasks, which will be excessively expensive when the number of tasks becomes large. The second limitation is that it adopts transfer learning to model the relatedness between tasks, which still requires a moderately large amount of labeled data to train the transfer models. Lastly, taskonomy only consider the transferability across different models or tasks while ignoring the transferability across different layers, which we argue is also important for a transfer to be successful.

The main obstacle standing in the way of measuring the transferability learned from different PR-DNNs is the "black-box" nature of deep models. As the knowledge (*e.g.*, features) learned from different PR-DNNs is unexplainable and actually in different embedding space, it is very tricky to compute the transferability directly. In this paper, to derive the transferability of knowledge encoded in PR-DNNs, we propose the DEeP Attribution gRAph (DEPARA) to represent the knowledge learned in PR-DNNs. In DEPARA,

*Equal contribution.

nodes correspond to the inputs and are represented by their vectorized attribution maps [25, 3, 24] with regards to the outputs of the PR-DNN. Edges denote the relatedness between inputs and are measured by their similarity in the embedding space of the PR-DNN (as seen in Figure 1). As the DEPARAs of different PR-DNNs are defined on the same set of inputs, they are actually in the same embedding space and thus the knowledge transferability of two PR-DNNs is directly measured by the similarity of their corresponding DEPARAs. More similar DEPARAs indicate that more correlated knowledge is learned from different PR-DNNs, thus the knowledge transferability to each other is higher.

The proposed method requires no human annotations, imposes no constraints on architectures and is several-magnitude times faster than taskonomy. Meanwhile, beyond model selection, it can also be easily adopted to the layer selection problem in transfer learning. Extensive experiments conducted demonstrate the effectiveness of DEPARA for quantifying the deep knowledge transferability.

To sum up, we made the following three main contributions: (1) We introduce the challenging, important yet under-studied deep knowledge transferability problem where only PR-DNNs are provided without any labeled data. (2) We propose the DEPARA, an efficient and effective method for deriving the transferability of the knowledge learned from PR-DNNs. To our knowledge, this is the first work to address the pre-trained model selection and the layer selection problems simultaneously. (3) Extensive experiments are conducted to demonstrate the effectiveness of DEPARA in solving both the model and the layer selection problems in transfer learning.

2. Related Work

2.1. Knowledge Transferability

Transferring PR-DNNs to new tasks is an active research topic. Razavian *et al*. [20] demonstrated that features extracted from deep neural networks could be used as generic image representations to tackle the diverse range of visual tasks. Yosinski *et al*. [31] investigated the transferability of deep features extracted from every layer of deep neural networks. Azizpour *et al*. [2] studied several factors affecting the transferability of deep features. Recently, the effects of pre-training datasets for transfer learning are also studied [12, 8, 11, 28]. Albeit many heuristics are found by these works, none of them explicitly quantify the transferability among different tasks and layers to provide a principled way for model and layer selection. Zamir *et al*. [33] proposed a fully computational approach to measure the task relatedness. Dwivedi and Roig [6] adopted representation similarity analysis for efficient task taxonomy. Song *et al*. [26] utilized the similarity of attribution maps to quantify the model transferability. However, the layer selection problem is still omitted in these works. In this paper, we propose DEPARA to address both the model and the layer selection problems in transfer learning.

2.2. Deep Model Attribution

Attribution refers to assigning importance scores to the inputs for a specified output. Existing attribution methods can be mainly divided into two groups, including perturbation- [34, 35, 36] and gradient-based methods [25, 3, 24, 27, 23, 1]. Perturbation-based methods compute the attribution of an input feature by making perturbations, *e.g.*, removing, masking or altering, to individual inputs or neurons and observe the impact on later neurons. In contrast, backpropagation-based methods calculate the attributions for all input features in one or few forward and backward pass through the network, which renders them more efficient. In this paper, we directly adopt existing attribution methods for transferability. Devising more advanced attribution method for our problem is left to future work.

2.3. Deep Knowledge Representation

How to represent the knowledge encoded in PR-DNNs is vital for knowledge reusing. Hinton *et al*. [10] viewed the soft predictions of a trained teacher model as the knowledge for knowledge distillation. Following their work, some other forms of knowledge are proposed to facilitate student learning. For example, Romero *et al*. [22] proposed to adopt intermediate representations learned by the teacher as hints to improve the final performance of the student. Zagoruyko and Komodakis [32] utilized the attention of the teacher model to guide the learning of the student. Recently, the relation of input instances learned from the trained deep models is also found a kind of useful knowledge [4, 16, 14, 29, 17]. For example, Chen *et al*. [4] utilized cross sample similarities to accelerate deep metric learning. Park *et al*. [16] leveraged mutual relations of data examples for knowledge distillation. In this paper, we propose DEPARA to represent the deep knowledge, which enables us easily quantify the knowledge transferability.

3. Deep Knowledge Transferability

3.1. Notation and Problem Setup

Assume there are N PR-DNNs available, denoted by $M = \{m_1, m_2, ..., m_N\}$. Each model in M can be viewed to be composed of a number of nonlinear functions: $m_i := f^i_{L_i} \circ \cdots \circ (f^i_2 \circ f^i_1)$, where f denotes the basic nonlinear function, L_i denotes the number of nonlinear functions in m_i, and the symbol $\circ$ denotes the function composition operation. Note that no constraints are imposed on the architectures of models in M, so the number of nonlinear functions in these PR-DNNs may be different. The task handled by m_i is denoted by t_i, and all the tasks in-

volved in M are collectively denoted by the task dictionary T, $T = \{t_1, t_2, ..., t_N\}$. For task t_i, we adopt $P_i(x, y)$ to denote the joint data distribution of the corresponding data domain. In this paper, the term *deep knowledge* refers to the embedding space learned by PR-DNNs. The embedding space produced after f_k^i in m_i is denoted by $\mathcal{F}_k^i$. Given M without any labeled data, we investigate the transferability, which is defined in the next section, between different $\mathcal{F}$s for facilitating task selection and layer selection in transfer learning.

3.2. Definition of Transferability

An intuitive description of transferability is "*how well a deep ConvNet representation can be transferred to the target task*" [31, 2]. Here we introduce a more rigorous definition to facilitate addressing the model and the layer selection problems in transfer learning. Assume there is a deep knowledge pool denoted by $\Omega = \{\mathcal{F}^{(1)}, \mathcal{F}^{(2)}, ...\}$[1]. Note that in this pool any two knowledge items $\mathcal{F}^{(i)}$ and $\mathcal{F}^{(j)}$ may be produced from different models or layers. The transferability of $\mathcal{F}^{(i)}$ to task t_j, denoted by $\mathcal{T}_{\mathcal{F}^{(i)} \to t_j}$, is defined as the ascending rank of $\mathcal{F}^{(i)}$ among Ω for solving the target task. Here the rank is computed based on the standard empirical risk. Formally, let D be the target data randomly sampled from P_j, *i.e.*, $D = \{(x_1, y_1), (x_2, y_2), ...\}$. $\mathcal{F}^{(i)}(D)$ denotes the embeddings of D in $\mathcal{F}^{(i)}$, then

$$\mathcal{T}_{\mathcal{F}^{(i)} \to t_j}(\Omega, D) := ascending\_rank(\mathcal{R}_{P_j}(h_{\mathcal{F}^{(i)}(D)}); \Omega). \tag{1}$$

$h_{\mathcal{F}^{(i)}(D)}$ is the hypothesis produced on $\mathcal{F}^{(i)}(D)$. $\mathcal{R}$ denotes the standard expected risk:

$$\mathcal{R}_{P_j}(h) := \mathbb{E}_{x,y \sim P_j}[\ell_j(h(x), y)], \tag{2}$$

where ℓ_j is the objective function of task t_j. Detailed descriptions of $ascending\_rank$ is provided in the supplementary material. If the transferability of every $\mathcal{F}$ in Ω to task t_j is known, we can directly select the $\mathcal{F}$ which ranks first for solving the target task t_j. Note that when every $\mathcal{F}$ in Ω comes from a different PR-DNN, the definition of transferability can be used for model selection. If all $\mathcal{F}$s in Ω come from different layers of the same PR-DNN, the definition can be used for layer selection in transfer learning.

The transferability defined above is intuitively straightforward. However, the computation is expensive for measuring the transferability between every pair of tasks in the task dictionary. What is worse, it needs labeled data for all the tasks involved. To bypass these problems, We propose DEPARA to approximate the defined transferability without any labeled data. We argue two factors must be considered simultaneously for computing the transferability:

1. **Inclusiveness**: for a transfer to be successful, $\mathcal{F}$ produced by the PR-DNN of the source task should be inclusive of sufficient information for solving the target task. Inclusiveness is an intuitively straightforward and fundamental ingredient of transferability. However, since $\mathcal{F}$ is highly nonlinear and unexplainable, it is very challenging to directly measure the inclusiveness of $\mathcal{F}$ for solving the target task.

2. **Accessibility**: $\mathcal{F}$ should be sufficiently abstracted and easily re-purposed to the target task so that the target task can be well solved with limited human supervision. Without the requirement of accessibility, $\mathcal{F}$ produced by shallower layers will be more likely of higher transferability as $\mathcal{F}$ from shallower layers tend to be of higher inclusiveness than that from higher layers. Measuring the accessibility of $\mathcal{F}$ is also a challenging problem due to the black-box nature of deep models.

3.3. Deep Attribution Graph

An illustrative diagram of the DEPARA is depicted in Figure 1. Formally, assume there is a set of randomly sampled unlabeled data points $D_p = \{x_1, x_2, ..., x_n\}$. D_p is called *probe data* in this paper. The probe data are first fed to the PR-DNN to obtain their features, *i.e.*, the outputs of the specific layer, after a forward pass. Then the attribution maps are produced by a backward pass. The back-propagation rule depends on the adopted attribution methods [1]. In DEPARA, each node corresponds to a data point in probe data and its feature is the vectorized attribution map of this data point. The edge between two nodes denotes the relatedness of the two data points and are measured by their similarity in the embedding space of the PR-DNN. For $\mathcal{F}_k^i$ from m_i, a DEPARA symbolized by $\mathcal{G}_k^i(D_p) = (\mathcal{V}_k^i, \mathcal{E}_k^i)$ can be obtained, where $\mathcal{V}$ and $\mathcal{E}$ denote the nodes and the edges, respectively. $\mathcal{G}_k^i(D_p)$ indicates the DEPARA is defined on D_p. More detailed descriptions of the nodes and the edges are provided as follows.

3.3.1 Nodes

The nodes in $\mathcal{G}_k^i$ are collectively denoted by $\mathcal{V}_k^i = \{v_{k,1}^i, v_{k,2}^i, ..., v_{k,n}^i\}$, where $v_{k,m}^i$ is the attribution of x_m with regards to $\mathcal{F}_k^i(x_m)$. In this paper, we adopt Gradient*Input [24] for attribution. Gradient*Input refers to a first-order Taylor approximation of how the output would change if the input was set to zero, which implies the importance of the input w.r.t the output. Mathematically, for the i-th element $x_{(i)}$ in x, its attribution score $v_{(i)}$ with respect to $\mathcal{F}$ is computed as:

$$v_{(i)} := x_{(i)} * \frac{\partial \|\mathcal{F}(x)\|^2}{\partial x_{(i)}}, \tag{3}$$

where $\|\cdot\|$ denotes ℓ_2 norm.

[1]Note that we use $\mathcal{F}^{(i)}$ to denote the i-th item in Ω, and $\mathcal{F}^i$ to denote the knowledge produced by m_i.

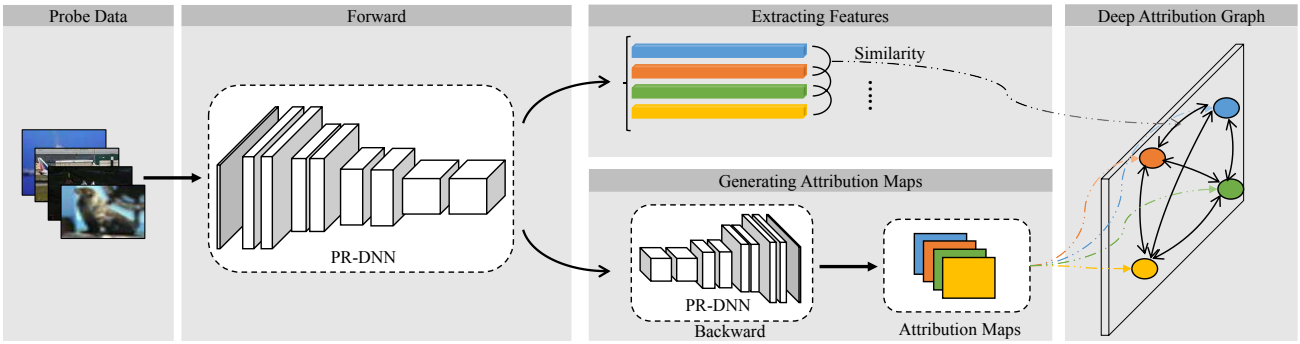


Figure 1. The illustrative diagram of the procedure for constructing the deep attribution graph.

The nodes are devised for measuring the inclusiveness of $\mathcal{F}$. The intuition is that for $\mathcal{F}^i(x_m)$ and $\mathcal{F}^j(x_m)$ of the same input x_m but produced by two PR-DNNs m_i and m_j, if they produce more similar attributions (*i.e.*, they focus on the more similar regions on the input), they are more likely to contain correlated information and be transformed to each other. Otherwise, they focus on different input dimensions so that being less correlated to each other.

3.3.2 Edges

The edges in $\mathcal{G}_k^i$ are collectively denoted by $\mathcal{E}_k^i = \{e_{k,11}^i, e_{k,12}^i, ..., e_{k,nn}^i\}$, where $e_{k,pq}^i$ is the edge of the p-th node and the q-th node and denotes the similarity of corresponding inputs in the embedding space $\mathcal{F}_k^i$. Formally,

$$e_{k,pq}^i := cosine\_sim(\mathcal{F}_k^i(x_p), \mathcal{F}_k^i(x_q)). \tag{4}$$

We adopt cosine similarity to define the edge because it is insensitive to the length of $\mathcal{F}(\cdot)$. Note that we assume there exists an edge between every pair of nodes in $\mathcal{V}_k^i$, so that $\mathcal{G}_k^i$ is actually a fully connected graph. Furthermore, as $\mathcal{G}_k^i$ is devised to be undirected, $e_{k,pq}^i = e_{k,qp}^i$ for any p and q.

The edges are devised to uncover the accessibility of transferability. If the embedding space $\mathcal{F}_k^i$ produced after f_k^i of m_i can be easily transferred (*i.e.*, of high accessibility) to another embedding space $\mathcal{F}_l^j$ produced after f_l^j of m_j, $\mathcal{F}_k^i$ and $\mathcal{F}_l^j$ should be similar in topological structure. Otherwise, it will consume a large amount of labeled data and training time to rebuild the embedding space $\mathcal{F}_l^j$ on top of $\mathcal{F}_k^i$, which violates the definition of high accessibility. The edges in $\mathcal{G}$ can be viewed as a representation of the topological structure in the embedding space. Two embedding spaces of the similar topological structure should produce similar edges in $\mathcal{G}$ for the same set of probe data.

3.4. Task Transferability

Here we adopt DEPARAs to quantify the transferability among different tasks in T, a goal similar to taskonomy [33]. However, in our problem only PR-DNNs of corresponding tasks are provided. We assume no labeled data are available for any task.

Before constructing DEPARAs for the tasks in T, two issues must be resolved. The first is that for task t_i, which embedding space $\mathcal{F}$ (*i.e.*, layer) of m_i should we choose to best represent the knowledge needed for task t_i. In this paper, we viewed all PR-DNNs in an encoder-decoder architecture. The encoder extracts compact features and the decoder makes predictions using the features from the decoder. We adopt the embedding space learned by the encoder, denoted as $\mathcal{F}_e^i$, to represent the knowledge of t_i. Thus the knowledge pool can be denoted by $\Omega = \{\mathcal{F}_e^1, \mathcal{F}_e^2, ..., \mathcal{F}_e^N\}$. The second is that we need a set of probe data which are shared among all the tasks for probing the topological structure of $\mathcal{F}$ and constructing the DEPARAs. In this paper, the probe data are randomly sampled. More details about how the probe data are obtained are provided in the experiment section and the supplementary material.

According to Eq. (3) and (4), for each task t in T, a DEPARA $\mathcal{G}_e$ is obtained on the probe data D_p. The transferability of $\mathcal{F}_e^i$ to task t_j is approximated by the descending rank of $\mathcal{F}_e^i$ in Ω based on the graph similarity:

$$\mathcal{T}_{\mathcal{F}_e^i \to t_j}(\Omega, D_p) \approx descending\_rank(s(\mathcal{G}_e^i, \mathcal{G}_e^j); \Omega, D_p), \tag{5}$$

where $s(\cdot)$ is the similarity function. $s(\mathcal{G}_e^i, \mathcal{G}_e^j) = s(\mathcal{V}_e^i, \mathcal{V}_e^j) + \lambda s(\mathcal{E}_e^i, \mathcal{E}_e^j)$. For nodes, we adopt the cosine similarity function: $s(\mathcal{V}_e^i, \mathcal{V}_e^j) = \frac{1}{n}\sum_{k=1}^{n} \frac{v_{e,k}^i \cdot v_{e,k}^j}{\|v_{e,k}^i\| \cdot \|v_{e,k}^j\|}$. For edges, the similarity is defined to be Spearman correlation coefficient: $s(\mathcal{E}_e^i, \mathcal{E}_e^j) = 1 - \frac{6\sum d_k^2}{n^3 - n}$, where d_k is the difference between the ranks of the k-th elements of $\mathcal{E}_e^i$ and $\mathcal{E}_e^j$. λ is the trade-off hyper-parameter. Detailed descriptions of $descending\_rank$ are given in the supplementary material.

3.5. Layer Transferability

As aforementioned, deep models are usually composed of many nonlinear functions or layers. For a PR-DNN $m_i = f_{L_i}^i \circ \cdots \circ (f_2^i \circ f_1^i)$, actually L_i different embedding spaces can be obtained, which can be denoted by

$\Omega_i = \{\mathcal{F}_1^i, \mathcal{F}_2^i, ..., \mathcal{F}_{L_i}^i\}$. However, in task transferability described above as well as taskonomy [33], only one embedding space $\mathcal{F}_e^i$ from the encoder is considered and all other learned knowledge is ignored. It may lead to suboptimal performance as reusing $\mathcal{F}_e^i$ can not guarantee to be optimal for different target tasks.

Here we consider the layer selection problem which is also important in transfer learning: for a source task t_i, which layer of its PR-DNN should be choosed to benefit the target task t_j most? The layer selection problem can be viewed as selecting $\mathcal{F}^i$ from Ω_i which benefits the target task t_j most. We adopt $\mathcal{F}_e^j$ produced by the encoder of m_j to denote the knowledge essential to task t_j, as $\mathcal{F}_e^j$ is usually the most compact. The layer selection is conducted by

$$k = \arg\max_k s(\mathcal{G}_k^i, \mathcal{G}_e^j). \tag{6}$$

With k computed from Eq. (6), we adopt $\mathcal{F}_k^i$ for transferring the PR-DNN m_i to the target task t_j.

4. Experiments

We first validate the proposed method for task transferability, then show its effectiveness for layer selection.

4.1. Task Transferability on Taskonomy Models

4.1.1 Pre-trained Models

Here we adopt PR-DNNs released by taskonomy [33] to validate the effectiveness of DEPARA for task transferability. Twenty PR-DNNs are selected in this experiment, each of which is for a single-image task. As all taskonomy models naturally follow an encoder-decoder architecture, we directly use the output of the encoder for constructing the DEPARA. Taskonomy measures the task transferability by the performance of transfer learning. We adopt its results to evaluate our method.

4.1.2 Probe Data

Following [26], we construct the probe data by randomly sampling $1,000$ images in the validation set of taskonomy data. We try using more data, but no obvious improvement in performance is observed in our experiment. Additionally, we also adopt Indoor Scene [19] and COCO [13], which are very different from taskonomy data, as the probe data for computing the transferability of taskonomy tasks. For more details, please refer to the supplementary material.

4.1.3 Evaluation Metric

We adopt two evaluation metrics, P@K and R@K, which are widely used in information retrieval, to compare the task transferability constructed from our method with that from taskonomy. Each target task is viewed as a query, and its top-5 source tasks which produce the best transferring performances in taskonomy are regarded as relevant to the query. We adopt the Precision-Recall (PR) curve to demonstrate the overall performance of the proposed method.

4.1.4 Visualization Results across Tasks

Here we visualize some nodes in $\mathcal{V}$ and edges in $\mathcal{E}$ of DEPARA to provide a better perceptual understanding of the proposed method. Results are shown in Figure 2. It can be seen that some tasks produce similar attribution maps and instance relationships, while some others not. For example, Rgb2depth produces highly similar attribution maps and relational graph with Rgb2mist. However, their results are dissimilar with that of Autoencoder. Actually, Rgb2depth and Rgb2mist are proved in taskonomy of high transferability to each other, while their transferability to Autoencoder is relatively low. Furthermore, taskonomy adopts agglomerative clustering to categorize the tasks into four groups: 3D, 2D, geometric, and semantic tasks. From Figure 2, we can see that our method tends to produce relatively similar nodes and edges within each group of tasks. Although some exceptions may occur, the results become more credible as we aggregate results of more nodes and edges.

4.1.5 Task Transferability Results

In this section, we evaluate the proposed method by the task transferability obtained from taskonomy. To better understand the results, we introduce a baseline using Random Ranking, which indicates the task transferability is randomly determined. To make ablation study of the proposed method, we introduce three variants of our method. DEPARA-$\mathcal{V}$: only the nodes in DEPARA are utilized for task transferability; DEPARA-$\mathcal{E}$: only the edges are used; DEPARA: the full version of our method using both nodes and edges, where λ is tuned by randomly sampling a small subset of all the PR-DNNs. Additionally, we also introduce another competitor: Representation Similarity Analysis (RSA) proposed by [7]. Here we adopt PR curve to compare the performance of all the aforementioned methods. To further demonstrate the similarity between the task transferability obtained by our method and that from taskonomy, the task similarity tree produced by DEPARA is also depicted in Figure 3. The task similarity tree from taskonomy and some other more results are provided in the supplementary material. From these results, we can conclude that: (1) The proposed method produces task transferability highly similar to that of taskonomy. As our method is much more efficient[2] than taskonomy, it is an effectual substitute for taskonomy when human annotations are unavailable or the

[2]The proposed method takes about 20 GPU hours on one Quadro P5000 card for pre-trained taskonomy models while taskonomy takes thousands of GPU hours on the cloud for 20 tasks.

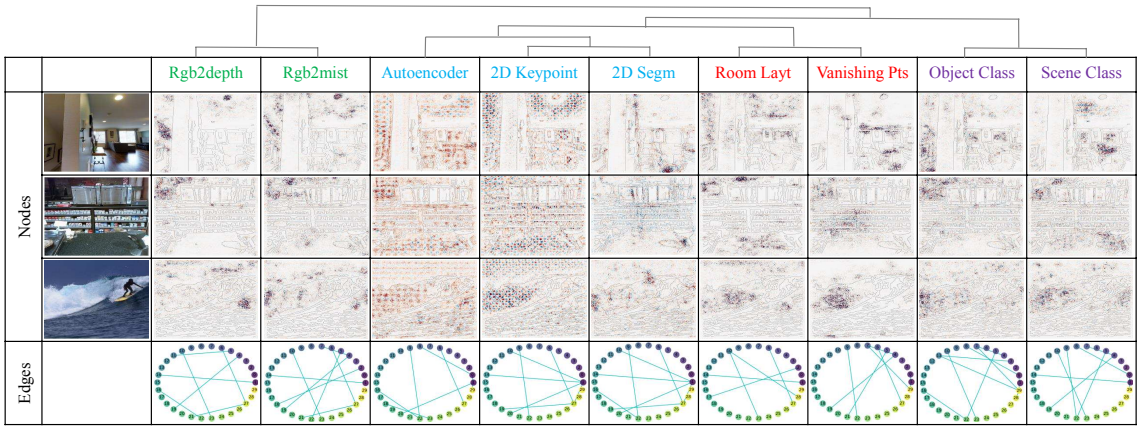
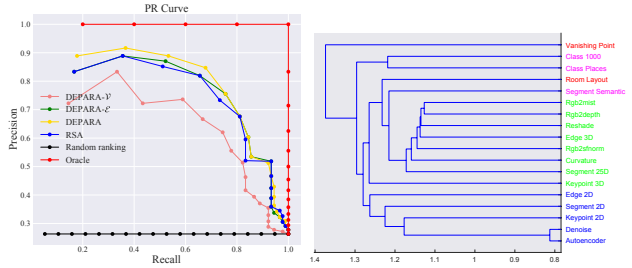


Figure 2. Visualization of some examples of the nodes and the edges of DEPARA. For the nodes, we visualize three examples from taskonomy data, Indoor Scene and COCO, respectively. For the edges, we randomly sample 30 nodes from taskonomy data and show their interconnections. Note that some weak connections are omitted for better visualization. Here we select two 3D tasks, three 2D tasks, two geometric tasks, and two semantic tasks for visualization. The task similarity tree derived from taskonomy is depicted above task names.



(a) PR curve. (b) Task similarity tree.

Figure 3. PR curve and the task similarity tree obtained on probe data randomly sampled from taskonomy data.

task library T is large in size. (2) DEPARA outperforms RSA [7], which demonstrates its superiority over the state-of-the-art. Actually, DEPARA-$\mathcal{E}$ and RSA yield comparable performance, as they are quite similar in method. (3) DEPARA outperforms DEPARA-$\mathcal{V}$ and DEPARA-$\mathcal{E}$ by a considerable margin, which implies the essentiality of both the nodes and the edges in DEPARA for measuring the knowledge transferability. For more results and interesting observations, please refer to the supplementary material.

To investigate the effects of different types of probe data, we also evaluate the proposed method with probe data from Indoor Scene and COCO. The task-wise P@K and R@K results, as well as the average results of the proposed method and some competitors, are provided in Table 1. It can be seen that although the data from Indoor Scene and COCO are quite different from taskonomy data, the proposed method still produces the task transferability of which the task-wise topological structure is highly similar to the one obtained by taskonomy. It indicates that the proposed method is insensitive to the randomly sampled probe data. Furthermore, the proposed method consistently outperforms DEPARA-$\mathcal{V}$, DEPARA-$\mathcal{E}$ and RSA on all the datasets, which again verifies the effectiveness and superiority of the proposed method.

4.2. Layer Selection in Transfer Learning

4.2.1 Experimental Settings

We adopt Syn2Real-C [18] dataset to validate the effectiveness of DEPARA for layer selection. In Syn2Real-C, the source and the target data are from different domains, but of the same 12 object categories. The source domain contains $152,397$ synthetic images and the target domain consists of $55,388$ images cropped from the Microsoft COCO dataset. In this paper, we use the data from the source and the target domain to train two domain-specific models. The ultimate goal is improving the performance on the target domain.

We consider two pre-trained models for being transferred to the target: (1) the model trained on the source domain (DNN-Source); (2) the deep model pre-trained on ImageNet (DNN-ImageNet). We adopt the architecture of VGG-19 for both models. DNN-Source is trained from scratch. The initial learning rate is set to be 0.01 and decayed to 0.001 after 50 epochs. We set weight decay to be 0.0005 and momentum to be 0.9. DNN-Source is trained for 80 epochs totally. For DNN-ImageNet, we directly adopt the pre-trained weights provided by TORCHVISION. To compute the transferability of DNN-Source and DNN-ImageNet to classification task on the target domain, we also trained the DNN-Target from scratch on the target data alone.

4.2.2 Performance of DEPARA for Layer Selection

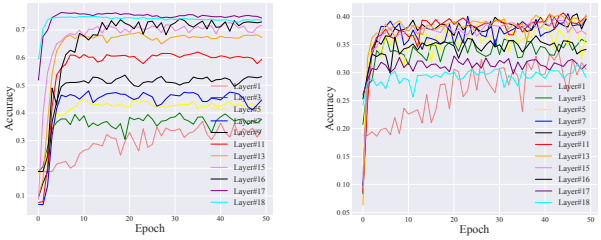
Here we show that DEPARA can pick out the layers which yield near the highest performance when transferred to the

Table 1. Task-wise similarity between the result from the DEPARA and that from taskonomy. The average results are shown on the right. For a better comparison, average results of DEPARA-$\mathcal{V}$, DEPARA-$\mathcal{E}$ and RSA are also provided.

		AutoEnco	Curvature	Denoise	Edge 2D	Edge 3D	Keypts 2D	Keypts 3D	Reshade	RGB2Depth	RGB2Mist	RGB2Norm	RoomLayt	Segmt 25D	Segmt 2D	VanishPts	SegmtSemt	Class1000	DEPARA	DEPARA-$\mathcal{V}$	DEPARA-$\mathcal{E}$	RSA
Taskonmy	P@1	1.0	0.0	1.0	1.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**0.88**	0.71	0.82	0.82
	P@5	1.0	0.6	1.0	0.4	0.8	0.8	0.8	0.8	0.8	0.8	0.6	0.8	0.8	0.8	0.8	0.4	0.8	**0.75**	0.68	**0.75**	0.73
	R@5	1.0	0.6	1.0	0.4	0.8	0.8	0.8	0.8	0.8	0.8	0.6	0.8	0.8	0.8	0.8	0.4	0.8	**0.75**	0.68	**0.75**	0.73
IndoorScn	P@1	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**1.00**	0.82	**1.00**	**1.00**
	P@5	1.0	0.6	1.0	0.6	0.6	1.0	1.0	1.0	0.8	0.8	0.8	0.8	0.8	1.0	0.8	0.6	0.6	**0.81**	0.72	0.78	0.79
	R@5	1.0	0.6	1.0	0.6	0.6	1.0	1.0	1.0	0.8	0.8	0.8	0.8	0.8	1.0	0.8	0.6	0.6	**0.81**	0.72	0.78	0.79
COCO	P@1	1.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0	1.0	**0.82**	**0.82**	0.76	**0.82**
	P@5	1.0	0.6	0.8	0.8	0.6	1.0	1.0	0.8	0.8	1.0	1.0	0.8	1.0	0.8	0.4	0.6	0.6	**0.80**	0.78	0.65	0.69
	R@5	1.0	0.6	0.8	0.8	0.6	1.0	1.0	0.8	0.8	1.0	1.0	0.8	1.0	0.8	0.4	0.6	0.6	**0.80**	0.78	0.65	0.69

target task. To this end, we exhaustively conduct the transfer learning for every layer in the pre-trained VGG-19. For each layer transferred to the target task, the current layer and all the layers previous to this layer are fixed and all the layers following the current layer are fine-tuned. As transfer learning usually happens when the target data is scarce, we conduct the experiments in two modes: (1) 0.1-T: 10% of the target data are used; (2) 0.01-T: only 1% of the target data are used. In both modes, the pre-trained VGG-19 is further trained for 50 epochs on target data. To select the transferred layer, we simply set λ to be 1 for both DNN-ImageNet and DNN-Source in 0.1-T mode. In 0.01-T mode, as the target data becomes scarcer, the accessibility becomes more important in transferability. Thus we set λ to be 10 in 0.01-T mode.

Results are listed in Table 2. We can see that: (1) The proposed method can successfully pick out the layers which yield the highest performance when transferred to the target. For example, for DNN-ImageNet in 0.01-T mode, $\#15$, $\#16$, $\#17$ and $\#18$ layers yield the highest transferring performance among all the layers. Our method successfully picks out these layers as they produce the highest DEPARA similarity. Actually, the average Spearman's correlation between the similarity and the accuracy is 0.913 for all the results shown in Table 2, implying that the similarity of DEPARA is a good indicator for layer selection in transfer learning. (2) For different trained models, the layers which yield the highest transferring performance differ. Furthermore, as the size of the target data varies, the best-performing layer may also change. For example, in 0.1-T mode for DNN-Source, $\#3$, $\#5$ and $\#7$ layers yield the highest performance. However, in 0.01-T mode the highest-performing layers are $\#11$, $\#12$ and $\#13$. By appropriately setting λ, the proposed method can still pick out the best layers for different amounts of target data. (3) Surprisingly, DNN-ImageNet yields much higher transferring performance than DNN-Source. The similarity of $\mathcal{E}$ in some layers of DNN-ImageNet is significantly higher than that



(a) DNN-ImageNet. (b) DNN-Source.

Figure 4. The test accuracy curves of different layers during the fine-tuning period in 0.01-T mode.

of DNN-Source, which implies that the embedding space learned on ImageNet is more suitable for solving the target task. The DNN-Source, albeit trained on the same task as the target, learned quite a different embedding space due to the large difference between the source and the target domain. Thus it produces relatively worse performance when transferred to the target data. (4) Trained from scratch on the target data, VGG-19 achieves 61.74% and 32.27% accuracy in 0.1-T and 0.01-T mode, respectively. Comparing these figures to the results in Table 2, we can see that some layers produce worse performance when transferred to the target data than they are trained from scratch. This phenomenon is known as *negative transfer* [30]. Negative transfer occurs especially when the PR-DNN is trained on a quite different domain (like DNN-Source) or for an unrelated task to the target task. For DNN-Source, most layers produce negative transfer when transferred to the target data. All these results imply the importance of both the model selection and the layer selection in transfer learning. Some other interesting observations from Table 2 are provided in the supplementary material.

In Figure 4, we depict the test accuracy curves of different layers when transferred to the target data. The results further demonstrate the layers selected by the proposed method are more suitable for being transferred to the tar-

Table 2. Layer-wise transferring performance of DNN-ImageNet and DNN-Source transferred to the target domain. SIM denotes the similarity between the DEPARAs of the specific layer and the target task. ACC denotes the accuracy on target test data. For space consideration, we omit the 2-nd, 4-th, 6-th and 8-th layers of VGG-19. Darker color indicates higher values.

			CONVOLUTIONAL LAYERS												FC LAYERS	
			#1	#3	#5	#7	#9	#10	#11	#12	#13	#14	#15	#16	#17	#18
DNN-ImageNet	SIM	$\mathcal{V}$	0.45	0.45	0.48	0.52	0.55	0.55	0.55	0.55	0.54	0.54	0.54	0.54	0.53	0.52
		$\mathcal{E}$	0.16	0.01	0.20	0.03	0.35	0.32	0.14	0.15	0.50	0.43	0.77	0.78	0.81	0.81
		$\lambda = 1$	0.61	0.46	0.68	0.55	0.90	0.87	0.69	0.70	1.04	0.97	1.31	1.32	1.34	1.33
		$\lambda = 10$	2.05	0.55	2.48	0.82	4.05	3.75	1.95	2.05	5.54	4.84	8.24	8.34	8.63	8.62
	ACC (%)	0.1-T	60.74	63.78	69.23	69.77	73.36	74.89	76.86	77.11	79.50	76.89	81.15	80.81	80.71	79.21
		0.01-T	34.03	37.71	40.16	44.67	53.06	58.11	59.35	63.08	67.24	68.50	71.72	72.85	74.33	73.54
DNN-Source	SIM	$\mathcal{V}$	0.60	0.60	0.55	0.53	0.50	0.50	0.50	0.49	0.48	0.48	0.48	0.47	0.46	0.45
		$\mathcal{E}$	0.06	0.11	0.15	0.17	0.18	0.18	0.19	0.19	0.20	0.17	0.15	0.11	0.10	0.09
		$\lambda = 1$	0.66	0.71	0.70	0.70	0.68	0.68	0.69	0.67	0.68	0.65	0.63	0.58	0.56	0.54
		$\lambda = 10$	1.20	1.70	2.05	2.23	2.30	2.30	2.40	2.39	2.48	2.18	1.98	1.57	1.46	1.35
	ACC (%)	0.1-T	49.84	61.92	62.72	62.28	59.81	60.24	58.49	54.03	54.21	52.67	52.15	48.54	41.50	36.10
		0.01-T	30.58	35.49	37.20	39.47	39.64	39.63	40.07	40.11	40.37	39.04	36.88	34.13	31.40	29.13

get than other layers. From Figure 4, it can be seen that the selected layers converge much faster than other layers when re-trained for the target task. For example, for the PR-DNN DNN-ImageNet, the proposed method picks out the #15, #16, #17, #18 layers for being transferred. The final accuracy also tends to be higher than that of other layers. Furthermore, layers in DNN-ImageNet produce more smooth test accuracy curves than DNN-Source, which indicates that the embedding space learned by DNN-ImageNet are more easily adapted to the target task. The embedding space learned by DNN-Source, however, is quite different in topological structure (as indicated by the low similarity of edges in DEPARA) from the one learned on the target data. When adapted to the target data, it will be largely destroyed and rebuilt for the target, thus the test accuracy curves oscillate and the transferring performance is poor.

5. Discussions and Conclusions

In this paper, we propose the DEPARA to investigate the transferability of knowledge encoded in PR-DNNs. We adopt DEPARA to handle two important yet under-studied problems in transfer learning: measuring the transferability across tasks for pre-trained model selection, and measuring the transferability across layers for layer selection. Extensive experiments are conducted to show its effectiveness in solving both these two problems in transfer learning. We summarize the advantages and the limitations of the proposed method. We hope it could make the contributions of this paper clearer and inspire us to study further.

Advantages. (1) Unlike taskonomy [33] which requires a large amount of labeled data, the proposed method quantifies the task transferability with only pre-trained models available. (2) As no training is involved, the computation cost of the proposed method grows nearly linearly with the size of the task dictionary, which is significantly more efficient than taskonomy. (3) The proposed method solves not only the model selection, but also the layer selection problem. As far as we know, we are the first to simultaneously tackle the model and the layer selection problems in transfer learning. (4) The proposed method imposes no constraints on the model architectures and are insensitive to the probe data. (5) This paper introduces a rigorous definition of knowledge transferability. Meanwhile, two vital ingredients, including inclusiveness and accessibility, are introduced for better approximating the transferability.

Limitations. (1) This paper directly adopts the existing attribution method, Gradient*Input [24], for quantifying transferability. However, different attribution methods may affect the proposed method in some way. In future work, more studies are needed to investigate the effects of different attribution methods on the proposed method. (2) The optimal trade-off between the nodes and the edges of DEPARA for knowledge transferability is proved to be dependent on the probe data and the amount of the target data. In this paper, the trade-off hyper-parameter λ is set via cross-validation or empirically. However, more study is needed to uncover the relationship between λ and its influencing factors. (3) The probe data used in the proposed method is randomly sampled. Although different probe data are shown to produces effective task-wise topological structures, they still affect the final performance to some degree. More investigation is needed to study how to construct the probe data for better measuring the transferability across different tasks, models and layers.

Acknowledgments. This work is supported by National Key Research and Development Program (2018AAA0101503), National Natural Science Foundation of China (61976186), Key Research and Development Program of Zhejiang Province (2018C01004), and the Major Scientific Research Project of Zhejiang Lab (No. 2019KD0AC01).

References

[1] Marco B Ancona, Enea Ceolini, Cengiz Oztireli, and Markus H. Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. In *ICLR*, 2018.

[2] Hossein Azizpour, Ali Sharif Razavian, Josephine Sullivan, Atsuto Maki, and Stefan Carlsson. Factors of transferability for a generic convnet representation. *TPAMI*, 38:1790–1802, 2014.

[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.

[4] Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. Darkrank: Accelerating deep metric learning via cross sample similarities transfer. In *AAAI*, 2017.

[5] Jia Deng, Wei ping Dong, Richard Socher, Li-Jia Li, Kehui Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[6] Kshitij Dwivedi and Gemma Roig. Representation similarity analysis for efficient task taxonomy & transfer learning. In *CVPR*, 2019.

[7] Kshitij Dwivedi and Gemma Roig. Representation similarity analysis for efficient task taxonomy & transfer learning. In *CVPR*, June 2019.

[8] Kaiming He, Ross B. Girshick, and Piotr Dollár. Rethinking imagenet pre-training. In *ICCV*, 2019.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CVPR*, pages 770–778, 2015.

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

[11] Mi-Young Huh, Pulkit Agrawal, and Alexei A. Efros. What makes imagenet good for transfer learning? *ArXiv*, abs/1608.08614, 2016.

[12] Simon Kornblith, Jonathon Shlens, and Quoc V. Le. Do better imagenet models transfer better? *ArXiv*, abs/1805.08974, 2018.

[13] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[14] Yufan Liu, Jiajiong Cao, Bing Li, Chunfeng Yuan, Weiming Hu, Yangxi Li, and Yunqiang Duan. Knowledge distillation via instance relationship graph. In *CVPR*, June 2019.

[15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

[16] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *CVPR*, June 2019.

[17] Baoyun Peng, Xiao Jin, Jiaheng Liu, Shunfeng Zhou, Yichao Wu, Yu Liu, Dongsheng Li, and Zhaoning Zhang. Correlation congruence for knowledge distillation. In *ICCV*, 2019.

[18] Xingchao Peng, Ben Usman, Kuniaki Saito, Neela Kaushik, Judy Hoffman, and Kate Saenko. Syn2real: A new benchmark forsynthetic-to-real visual domain adaptation. *CoRR*, abs/1806.09755, 2018.

[19] Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *CVPR*, 2009.

[20] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: An astounding baseline for recognition. *CVPR Workshops*, pages 512–519, 2014.

[21] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *TPAMI*, 39:1137–1149, 2015.

[22] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *CoRR*, abs/1412.6550, 2014.

[23] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *ICML*, 2017.

[24] 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.

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

[26] Jie Song, Yixin Chen, Xinchao Wang, Chengchao Shen, and Mingli Song. Deep model transferability from attribution maps. In *NeurIPS*, pages 6179–6189. Curran Associates, Inc., 2019.

[27] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *ICML*, 2017.

[28] Antonio Torralba and Alexei A. Efros. Unbiased look at dataset bias. *CVPR*, pages 1521–1528, 2011.

[29] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *ICCV*, 2019.

[30] Zirui Wang, Zihang Dai, Barnabas Poczos, and Jaime Carbonell. Characterizing and avoiding negative transfer. In *CVPR*, June 2019.

[31] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *NIPS*, 2014.

[32] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR*, 2016.

[33] Amir R. Zamir, Alexander Sax, William Shen, Leonidas J. Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *CVPR*, June 2018.

[34] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *ECCV*, 2014.

[35] Jian Zhou and Olga G. Troyanskaya. Predicting effects of noncoding variants with deep learning–based sequence model. *Nature Methods*, 12:931–934, 2015.

[36] Luisa M. Zintgraf, Taco Cohen, Tameem Adel, and Max Welling. Visualizing deep neural network decisions: Prediction difference analysis. In *ICLR*, 2017.