Selection of General Convolutional Layers with Projection Correlation

Anonymous Authors[1]

Abstract

This paper studies the transfer learning problem in image classification applications. A phase transition phenomenon has been empirically validated: the convolutional layer shifts from general to specific with respect to the target task as its depth increases. The paper suggests measuring the generality of convolutional layers through an easy-to-compute and tuning-free quantity named projection correlation. The non-asymptotic upper bounds for the estimation error of the proposed generality measure has been provided. Based on this generality measure, the paper proposes a forward-adding-layer-selection algorithm to select general layers. The algorithm aims to find a cut-off in the pre-trained model according to where the phase transition from general to specific happens. Then, we propose to transfer only the general layers as specific layers can cause overfitting issues and hence hurt the prediction performance. The proposed algorithm is computationally efficient and can consistently estimate the true beginning of phase transition under mild conditions. Its superior empirical performance has been justified by various numerical experiments.

1. Introduction

Recent researches in transfer learning (Yosinski & Lipson, 2014; Long & Jordan, 2015; Long et al., 2016; Azizpour et al., 2015; Mou et al., 2016) revealed an interesting phenomenon: when we train convolutional neural network (CNN) models on image datasets, the features generated by low-level layers are prone to represent some common visual notions like shapes, edges, geometric symmetry, effects of lighting, and so on. For instance, the features in the first layer of different models tend to be common filters or color blobs (Yosinski & Lipson, 2014; Lenc & Vedaldi, 2015; Huh et al., 2016; Zeiler & Fergus, 2014). This phenomenon is observed not only across various datasets, but also with distinct tasks including image classification (Krizhevsky et al., 2012), feature extraction (Mao & Jain, 1995; Lawrence et al., 1997), objective recognition (Hinton et al., 2012), image segmentation (Long et al., 2015; Ronneberger et al., 2015), image understanding (Mahendran & Vedaldi, 2015; Maninis et al., 2016) and so on. Therefore, it is suitable to call the first layer a GENERAL LAYER. On the other hand, it is well known that the high-level layers (especially the last one) of CNN models are task specific and data dependent. Thus, we call the last-layer a SPECIFIC LAYER. If the first layer is general and the last layer is specific, then there must be a phase transition from general to specific somewhere in the network. We refer to Figure 1 for a toy example to visualize the feature maps in each layer of CNN. This phase transition phenomenon sheds some light on the over-fitting issue (Azizpour et al., 2015; N. Papernot & Swami, 2016; Geng et al., 2016; Brock et al., 2017) in transfer learning as including the layers that are not general to the target task can hurt the performance.

To avoid over-fitting, some literature (Yosinski & Lipson, 2014; Jean et al., 2016) proposed to fit a sequence of nested CNN models on the target task, each of which includes an additional layer from the pre-trained model. Then the model that minimizes the target loss function (e.g., prediction or classification error) is selected as the final model in transfer learning. Though straightforward, this prediction-based scheme has several limitations. First, this scheme requires one to fit every model in the sequence, which can be computationally expensive if the pre-trained model has a large number of layers or each model fitting is slow. Second, the generality of layer is indirectly measured by the cost function of target task. The fairness of this measurement is subject to the tuning of the CNN models. For example, the prediction performance of a CNN model may be simultaneously affected by multiple hyper-parameters like the number of feature maps per layer, the size of local receptive fields, pooling size, learning rate, and so on.

To overcome the aforementioned limitations, we propose to measure the generality of a layer in the pre-trained model with respect to a target response through an easy-to-compute and tuning-free quantity named projection correlation. The

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

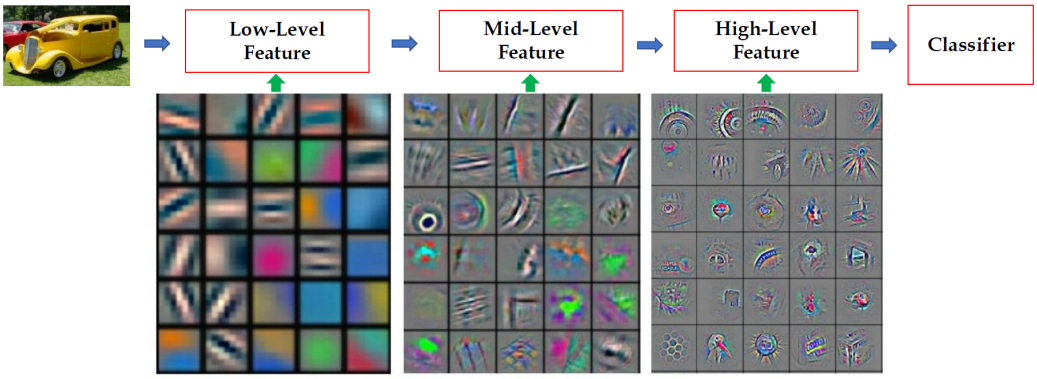

Figure 1. Visualization of feature maps in low, mid, and high level layers of CNN.

projection correlation, proposed in (Zhu et al., 2017), is a measure of functional dependence between two random vectors which enjoys several nice probability properties. The projection correlation equals zero if and only if the two random vectors are independent and is invariant to orthogonal transformations. In this paper, we further demonstrate the projection correlation does not depend on the dimensions and moment conditions of the two random vectors. Moreover, projection correlation is an ideal measurement of generality of convolutional layers in transfer learning as it does not require specifying any regression/classification model and is insensitive to the correlations and moment conditions of the dataset. In addition, the estimation of projection correlation is computationally efficient and free of tuning parameters. The theoretical analysis exhibits the estimation of projection correlation enjoys a non-asymptotic exponential type upper bound. This guarantees that, we can recover the beginning of phase transition in transfer learning with high probability without training any CNN model on the target dataset. The only condition required is the minimum signal strength of phase transition does not converge to zero too fast as the sample size diverges.

Based on this generality measure, we propose a forward-adding-layer-selection algorithm to select general layers to improve the transfer learning performance. The algorithm aims to find a cut-off in the pre-trained model according to the estimated location of the phase transition from general to specific. Then, we advocate transferring only the general layers and ignoring the specific ones that do not improve the transfer learning performance. Our method selects a parsimonious model in a computationally efficient way. The extensive numerical experiments show that the transfer learning models selected by our method are identical to the ones selected by the exhaustive search.

The rest of paper is organized as follows. Section 2 introduces the projection correlation and its properties. In Section 3, we propose to measure the generality of convolutional layers with the projection correlation. In Section 4, we introduce a forward-adding-layer-selection algorithm to select general layers. The Section 5 provides various numerical experiments. Due to the limitation of space, we defer the proofs of theoretical results to a supplemental material.

2. Projection correlation

To begin with, we introduce the projection correlation and its properties to pave the way for the proposed generality measure. Let $X \in \mathbb{R}^p$ and $Y \in \mathbb{R}^q$ be two random vectors. The projection correlation is elicited by the following independence testing problem.

$$H_0 : X \perp\!\!\!\perp Y \quad \text{verse} \quad H_1 : \text{otherwise.}$$

The null hypothesis holds if and only if $U = \alpha^{\mathrm{T}} X$ and $V = \beta^{\mathrm{T}} Y$ are independent for all unit vectors α and β. Let $F_{U,V}(u, v)$ be the joint distribution of $(U,\ V)$, we can define the squared projection covariance as follows

$$\mathrm{Pcov}(X, Y)^2 = \iiint \mathrm{cov}^2\{I(\alpha^{\mathrm{T}} X \le u),\ I(\beta^{\mathrm{T}} Y \le v)\} \, dF_{U,V}(u, v)\, d\alpha\, d\beta. \quad (1)$$

Furthermore, we define the projection correlation between X and Y as the square root of

$$\mathrm{PC}(X, Y)^2 = \frac{\mathrm{Pcov}(X, Y)^2}{\mathrm{Pcov}(X, X)\mathrm{Pcov}(Y, Y)}, \quad (2)$$

with convention $0/0 = 0$.

In general $0 \le \mathrm{PC}(X, Y) \le 1$, testing whether X and Y are independent amounts testing whether $\mathrm{PC}(X, Y) = 0$. The projection correlation is a measure of dependence between two random vectors and enjoys some appealing properties.

Let X and Y be two random vectors with continuous marginal and joint probability distributions, $\mathrm{PC}(X,Y)=0$ if and only if X and Y are independent. Note that this property does not hold in general without the assumption that (X,Y) is jointly continuous. When X and Y are two dependent discrete random variables that are constructed in a similar fashion as in (Hoeffding, 1948), one can show that $\mathrm{PC}(X,Y)=0$.

Though intuitive, the definition of projection covariance in (1) is not straightforward in terms of calculation. (Zhu et al., 2017) gives an explicit formula for the squared projection covariance in (1).

Let $(X_1,Y_1),\ \ldots,\ (X_5,Y_5)$ be 5 independent random copies of (X,Y),

$$\begin{aligned}
&\mathrm{Pcov}(X,Y)^2 = S_1+S_2-2S_3\\
&= E\left[\arccos\left\{\frac{(X_1-X_3)'(X_4-X_3)}{\|X_1-X_3\|\|X_4-X_3\|}\right\}\arccos\left\{\frac{(Y_1-Y_3)'(Y_4-Y_3)}{\|Y_1-Y_3\|\|Y_4-Y_3\|}\right\}\right]\\
&+E\left[\arccos\left\{\frac{(X_1-X_3)'(X_4-X_3)}{\|X_1-X_3\|\|X_4-X_3\|}\right\}\arccos\left\{\frac{(Y_2-Y_3)'(Y_5-Y_3)}{\|Y_2-Y_3\|\|Y_5-Y_3\|}\right\}\right]\\
&-2E\left[\arccos\left\{\frac{(X_1-X_3)'(X_4-X_3)}{\|X_1-X_3\|\|X_4-X_3\|}\right\}\arccos\left\{\frac{(Y_2-Y_3)'(Y_4-Y_3)}{\|Y_2-Y_3\|\|Y_4-Y_3\|}\right\}\right],
\end{aligned}$$

where $\|\cdot\|$ is the L_2 norm. Equation (2) shows that the projection covariance only depends on the vectors of form $(X_k-X_l)/\|X_k-X_l\|$ and $(Y_k-Y_l)/\|Y_k-Y_l\|$ whose second moments are unity. This gives us the intuition that the projection covariance is free of the moment conditions on (X,Y) which are usually required by some other measurements, such as distance correlation (Li et al., 2012).

In the second half of this section, we introduce an estimator of projection correlation. Let $\boldsymbol{X}=(X_1,\ \cdots,X_n)^{\mathrm{T}}$ and $\boldsymbol{Y}=(Y_1,\ \cdots,Y_n)^{\mathrm{T}}$ be an observed sample of (X,Y). The sample projection variance and covariance of $\boldsymbol{X}$ and $\boldsymbol{Y}$ can be calculated as

$$\begin{aligned}
\widehat{\mathrm{Pcov}}(\boldsymbol{X},\boldsymbol{Y})^2 &= n^{-3}\sum_{k,l,r=1}^{n}A_{klr}B_{klr},\\
\widehat{\mathrm{Pcov}}(\boldsymbol{X},\boldsymbol{X}) &= \left\{n^{-3}\sum_{k,l,r=1}^{n}A_{klr}^2\right\}^{1/2},\\
\widehat{\mathrm{Pcov}}(\boldsymbol{Y},\boldsymbol{Y}) &= \left\{n^{-3}\sum_{k,l,r=1}^{n}B_{klr}^2\right\}^{1/2},
\end{aligned}$$

where for $k,l,r=1,\ \cdots,n$,

$$\begin{aligned}
&a_{klr}=\arccos\left\{\frac{(X_k-X_r)'(X_l-X_r)}{\|X_k-X_r\|\|X_l-X_r\|}\right\},\\
&a_{klr}=0 \text{ if } k=r \text{ or } l=r,\\
&\bar{a}_{k\cdot r}=n^{-1}\sum_{l=1}^{n}a_{klr},\quad \bar{a}_{\cdot lr}=n^{-1}\sum_{k=1}^{n}a_{klr},\quad \bar{a}_{\cdot\cdot r}=n^{-2}\sum_{k=1}^{n}\sum_{l=1}^{n}a_{klr},\\
&A_{klr}=a_{klr}-\bar{a}_{k\cdot r}-\bar{a}_{\cdot lr}-\bar{a}_{\cdot\cdot r},\\
&b_{klr}=\arccos\left\{\frac{(Y_k-Y_r)'(Y_l-Y_r)}{\|Y_k-Y_r\|\|Y_l-Y_r\|}\right\},\\
&b_{klr}=0 \text{ if } k=r \text{ or } l=r,\\
&\bar{b}_{k\cdot r}=n^{-1}\sum_{l=1}^{n}b_{klr},\quad \bar{b}_{\cdot lr}=n^{-1}\sum_{k=1}^{n}b_{klr},\quad \bar{b}_{\cdot\cdot r}=n^{-2}\sum_{k=1}^{n}\sum_{l=1}^{n}b_{klr},\\
&B_{klr}=b_{klr}-\bar{b}_{k\cdot r}-\bar{b}_{\cdot lr}-\bar{b}_{\cdot\cdot r}.
\end{aligned}$$

Then the sample projection correlation between $\boldsymbol{X}$ and $\boldsymbol{Y}$ is defined as the square root of

$$\widehat{\mathrm{PC}}(\boldsymbol{X},\boldsymbol{Y})^2=\frac{\widehat{\mathrm{Pcov}}(\boldsymbol{X},\boldsymbol{Y})^2}{\widehat{\mathrm{Pcov}}(\boldsymbol{X},\boldsymbol{X})\widehat{\mathrm{Pcov}}(\boldsymbol{Y},\boldsymbol{Y})}. \tag{3}$$

The following theorem provides exponential-type deviation inequalities for sample projection covariance and correlation.

Theorem 2.1. *For any $0<\varepsilon<1$, as long as $n\geq 10\pi^2/\varepsilon$, there exists positive constants c_1 and c_2, such that*

$$\begin{aligned}
&\Pr\left\{|\widehat{\mathrm{Pcov}}(\boldsymbol{X},\boldsymbol{Y})^2-\mathrm{Pcov}(X,Y)^2|>\varepsilon\right\}\\
&\qquad\leq c_1\exp\{-c_2n\varepsilon^2\},\\
&\Pr\left\{|\widehat{\mathrm{PC}}(\boldsymbol{X},\boldsymbol{Y})^2-\mathrm{PC}(X,Y)^2|>\varepsilon\right\}\\
&\qquad\leq 5c_1\exp\{-c_2n\varepsilon^2\gamma\},
\end{aligned}$$

where

$$\gamma=\min\{\gamma_x^3\gamma_y^3/64M^4,\gamma_x^2\gamma_y^2/64M^4,\gamma_x\gamma_y/4\},$$

$\gamma_x=\mathrm{Pcov}(X,X)^2$, $\gamma_y=\mathrm{Pcov}(Y,Y)^2$ *and* $M=2\pi^2$.

Remark 2.2. *The above exponential inequalities do not depend on the dimensionality and moment conditions of both random vectors. The probability that the estimation errors exceed the threshold ε decays exponentially with sample size n which guarantees good finite sample performance of the proposed estimators. This novel non-asymptotic results may be of independent interest to audience.*

3. Measure the generality of layers

Consider the following transfer learning problem. Let $\{\boldsymbol{X}^s,\boldsymbol{Y}^s\}$ be a source dataset where we pre-trained a CNN model of J layers. Denote $\boldsymbol{\Phi}^j(\cdot),j=1,\ldots,J$ the features learned in the jth layer. When the jth layer has multiple feature maps, the function $\boldsymbol{\Phi}^j(\cdot)$ can be defined as an aggregation of feature maps. For example, we can define $\boldsymbol{\Phi}^j(\cdot)$ as the vectorization of all feature maps in the jth convolutional (or pooling) layer. Our framework also allows $\boldsymbol{\Phi}^j(\cdot)$ to be defined as other aggregation methods of feature maps.

Suppose that, we want to transfer the pre-trained model to a target task with a pair of input and output probability distributions $\{X^t,Y^t\}$. We propose to measure the generality of

the layers in the pre-trained model with respect to the target by the projection correlation. To be specific, we suggest to use the population projection correlation

$$\omega_j = \mathrm{PC}(\mathbf{\Phi}^j(X^t), Y^t), \quad j = 1, \ldots, J,$$

as a measure of generality between the jth layer in the pre-trained model and the target task realized on $\{X^t, Y^t\}$. Given an observed target dataset $\{\boldsymbol{X}^t, \boldsymbol{Y}^t\}$ follows distributions $\{X^t, Y^t\}$, we can estimate ω_j by

$$\widehat{\omega}_j = \widehat{\mathrm{PC}}(\mathbf{\Phi}^j(\boldsymbol{X}^t), \boldsymbol{Y}^t), \quad j = 1, \ldots, J.$$

Further, we use $j^* = \underset{1 \le j \le J}{\operatorname{argmax}} \omega_j$ to denote the beginning of phase transition from general to specific in transfer learning. When $j^* = J$, there is no phase transition happens in the transfer learning. This can be true when all the layers in the pre-trained model are "helpful" to the target task. In practice, we can estimate j^* by $\widehat{j} = \underset{1 \le j \le J}{\operatorname{argmax}} \widehat{\omega}_j$.

Denote $\Delta_j \equiv \mathrm{PC}(\mathbf{\Phi}^j(\boldsymbol{X}^t), \boldsymbol{Y}^t)^2 - \mathrm{PC}(\mathbf{\Phi}^{j+1}(\boldsymbol{X}^t), \boldsymbol{Y}^t)^2$, $j = 1, \ldots, J-1$, the gap between two consecutive layers. In order to consistently detect the beginning of phase transition, the following assumption requires the minimum signal strength of phase transition does not converge to zero too fast as n diverges.

Assumption 1 (Minimum signal strength)**.** *Let $\mathcal{S} = \{j : \Delta_j > 0\}$ be the set of index of all positive gaps. For some $c_3 > 0$ and $0 \le \tau < 1/2$, we assume that $\min_{k \in \mathcal{S}} \Delta_k = \Delta_{k^*} > c_3 n^{-\tau}$.*

The following theorem shows that the proposed forward-adding-layer-selection algorithm, detailed in Section 4 below, can consistently estimate the beginning of phase transition, i.e. j^*.

Theorem 3.1 (location consistency of phase transition)**.** *Under Assumption 1, we have*

$$\Pr\{\widehat{j} = j^*\} > 1 - c_1' \exp\{-c_2' n^{1-2\tau}\gamma\},$$

where γ is given in Theorem 2.1, c_1' and c_2' are two positive constants.

4. Forward adding selection of layers

Given the generality measure introduced in Section 3, we propose the following forward-adding-layer-selection algorithm to find a cut-off k in $\{1, \ldots, J\}$, such that $\{1, \ldots, k\}$ are general layers which will improve the learning of the target task while $\{k+1, \ldots, J\}$ are specific layers which will not benefit the target task.

The proposed forward-adding-layer-selection algorithm saves computational cost from two aspects. First, we have a warm start k_0 based on the ranking of projection correlations. Second, the procedure stops when the target loss function stops decreasing. The intuition of the algorithm is as follows. The algorithm starts the transfer learning with the layers that are most likely to be general to the target task. The algorithm adds one new layer each time and stops when the new layer no longer improves the prediction performance. We illustrate the difference between proposed forward selection algorithm and prediction-based algorithm in Figure 2.

Algorithm 1 Forward adding selection of general layers

1. Calculate the projection correlation between $\mathbf{\Phi}^j(\boldsymbol{X}^t)$ and $\boldsymbol{Y}^t$, i.e. $\widehat{\omega}_j = \widehat{\mathrm{PC}}\left(\mathbf{\Phi}^j(\boldsymbol{X}^t), \boldsymbol{Y}^t\right)$, for $j = 1, \ldots J$.
2. Set the initial value of cut-off as $k_0 = \operatorname{argmax}_j \widehat{\omega}_j$ which is the layer index corresponding to the largest sample projection correlation.
3. Train a model on the target task whose first k_0 layers are transferred from the pre-trained model. Denote e_0 the target loss function associated with this model.
4. Train an updated model on the target task with the first $k_0 + 1$ layers being transferred from the pre-trained model. Denote e_1 the target loss function associated with this updated model.
5. If $e_1 \ge e_0$, stop and set $k = k_0$. If $e_1 < e_0$, set $k_0 = k_0 + 1$, $e_0 = e_1$ and repeat step 4 until stop.

5. Numerical Results

5.1. Experiment on CIFAR-10 dataset

In this experiment, we study a transfer learning problem for image classification. The dataset of interest is the CIFAR-10 dataset [1] which consists of 60,000 32x32 colour images in 10 classes, with 6000 images per class. To mimic two representative scenarios in transfer learning, we split the CIFAR-10 dataset into a source dataset $\{\boldsymbol{X}^s, \boldsymbol{Y}^s\}$ and a target dataset $\{\boldsymbol{X}^t, \boldsymbol{Y}^t\}$, according to one of the following two cases.

1. Imbalance case: 10 classes are equally and randomly divided into two groups, one belongs to the source dataset and the other one belongs to the target dataset. In other words, both $\{\boldsymbol{X}^s, \boldsymbol{Y}^s\}$ and $\{\boldsymbol{X}^t, \boldsymbol{Y}^t\}$ have 5 **non-overlapping** classes[2], each of which has 6,000 images.

[1] https://www.cs.toronto.edu/kriz/cifar.html

[2] In our experiment, the source dataset contains automobile, truck, deer, dog and horse classes; while the target dataset contains airplane, bird, ship, cat and frog classes

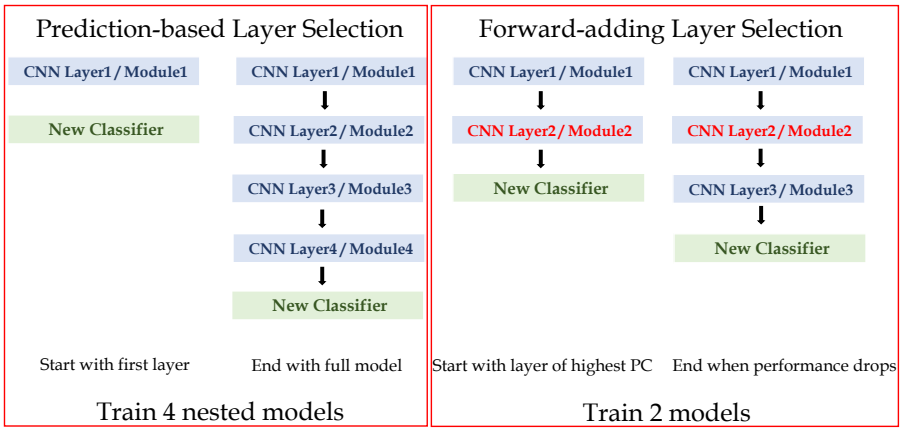

Figure 2. Comparison of prediction-based layer selection and forward-adding-layer-selection.

2. Balance case: each class is equally and randomly divided into two subsamples, one belongs to the source dataset and the other one belongs to the target dataset. In other words, both $\{\boldsymbol{X}^s, \boldsymbol{Y}^s\}$ and $\{\boldsymbol{X}^t, \boldsymbol{Y}^t\}$ has 10 classes, each of which has 3,000 images.

For each case, we pre-train a six layers CNN on the source dataset using 25,000 images (equally distributed across classes). The architecture of the network is illustrated as in Figure 3. The network is trained by RMSProp optimizer (Tieleman & Hinton, 2012) with learning rate equals 0.001 and decay equals 0.9. The classification accuracy, measured over the rest 5,000 test images, is 89.6% and 79.7% for the **Imbalance** and **Balance** cases, respectively. This pre-trained CNN model is not the record keeper of this dataset in terms of classification accuracy. Nevertheless, its classic architecture gives us a straightforward illustration of the phase transition phenomenon in transfer learning.

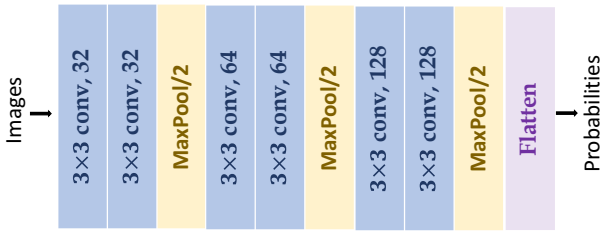

Figure 3. Architecture of CNN in CIFER-10 experiment.

We notice that the **Balance** case is more like a classic training-testing splitting as $\{\boldsymbol{X}^s, \boldsymbol{Y}^s\}$ and $\{\boldsymbol{X}^t, \boldsymbol{Y}^t\}$ follows approximately the same joint probability distribution. Therefore, we expect the full model pre-trained on the source dataset should be general to the target dataset. On the other hand, the **Imbalance** case is designed to mimic some real challenges in transfer learning: the marginal distributions of inputs and outputs (and hence their joint distribution) are different in the source dataset and the target dataset. Moreover, transferring the full pre-trained model can cause over-fitting issues. Intuitively, we expect to observe a phase transition phenomenon in the **Imbalance** case but not in the **Balance** case.

Denote, $\boldsymbol{\Phi}^j(\cdot)$ the vectorization of feature maps in the jth convolutional layer in the pre-trained model, $j = 1, \ldots, 6$. In practice, the target dataset is usually of limited sample size and may be insufficient to train a deep CNN model. To address this concern, let $\{\widetilde{\boldsymbol{X}}^t, \widetilde{\boldsymbol{Y}}^t\}$ be a random subsample of size 500, 1000, and 1500 drawn from the target dataset, respectively. We measure the generality of convolutional layers with respect to the target dataset by

$$\widehat{\omega}_j = \widehat{\mathrm{PC}}\left(\boldsymbol{\Phi}^j(\widetilde{\boldsymbol{X}}^t), \widetilde{\boldsymbol{Y}}^t\right), \quad \text{for} \quad j = 1, \ldots 6.$$

We calculate the sample mean and sample standard deviation of $\widehat{\omega}_j$ over 20 replications. The results versus j for **Imbalance** and **balance** cases are presented in panel (A) and (B) of Figure 4, respectively. For the **Imbalance** case, we observe that the phase transition phenomenon happens between the 4th and 5th layers. On the contrary, we do not observe any phase transition in the **Balance** case.

Next, we validate our findings by assessing the classification performance of 7 nested models, $\mathcal{M}_0, \ldots, \mathcal{M}_6$, on the target dataset. Each of the 7 models has the same architecture as in Figure 3 and is trained over 150 epochs. For the model $\mathcal{M}_j$, $j = 0, \ldots, 6$, we transfer the first j layers from the pre-trained model, while the rest $6 - j$ layers are initialized randomly and trained using 25,000 images (equally distributed across classes) drawn from the target dataset. In $\mathcal{M}_0$, all layers are trained with the target dataset. In $\mathcal{M}_6$, all layers are transferred from the pre-trained model. In $\mathcal{M}_1, \ldots, \mathcal{M}_5$, the models are partially transferred from the pre-trained CNN model.

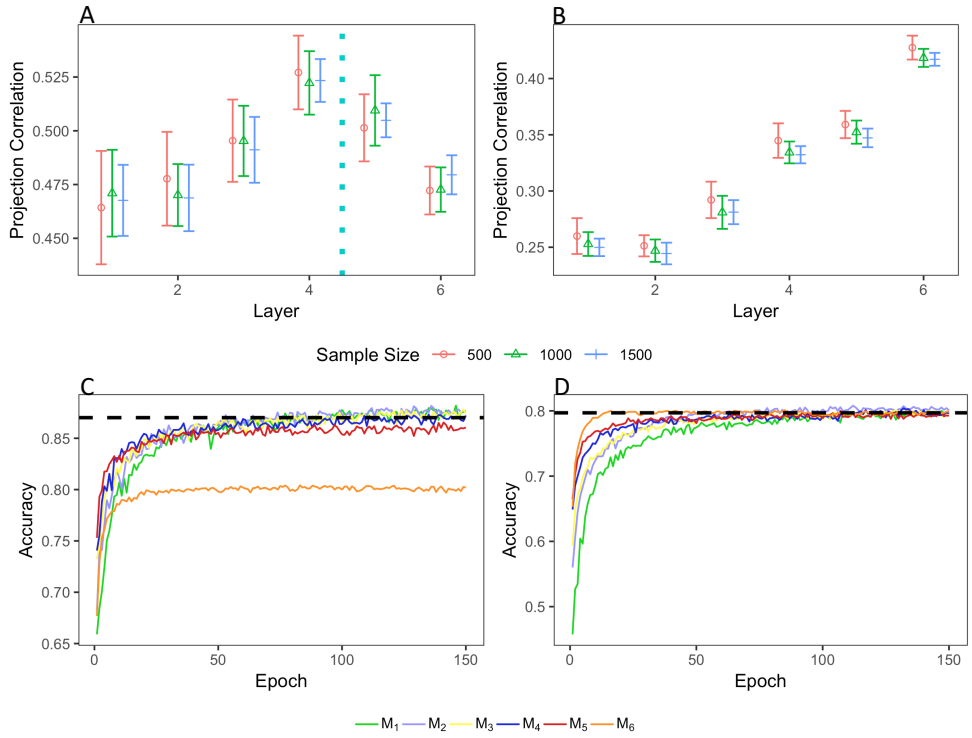

Figure 4. (A) **Imbalance case**: projection correlation versus layers with sample sizes 500, 1000 and 1500; (B) **Balance case**: projection correlation versus layers with sample sizes 500, 1000 and 1500; (C) **Imbalance case**: classification accuracy on the target dataset versus the number of epochs for 7 transferred models; (D) **Balance case**: classification accuracy on the target dataset versus the number of epochs for 7 transferred models.

The performance of each model is assessed by the classification accuracy of over 5,000 test images in the target dataset. We present the classification accuracy versus the number of epochs for **Imbalance** and **Balance** cases in panel (C) and (D) of Figure 4, respectively. In panel (C) and (D), the black dashed lines represent the classification performance of $\mathcal{M}_0$. We use $\mathcal{M}_0$ as a benchmark to represents the scenario that we have a large enough target dataset to train a model by itself. The performances of the other 6 models are plotted as colored lines. If $\mathcal{M}_j$, $j = 1, \ldots, 6$, performs as well as $\mathcal{M}_0$, then the first j layers are general to the target dataset as they perform as well as the ones fully trained on the target dataset. Panel (C) justified our finding of phase transition in the **Imbalance** case. The lines corresponding to $\mathcal{M}_1, \ldots, \mathcal{M}_4$ converge to the benchmark while the lines corresponding to $\mathcal{M}_5$ and $\mathcal{M}_6$ are worse than the benchmark. On the other hand, all colored lines in Panel (D) converge to the benchmark which indicates no phase transition in the **Balance** case.

Last but not least, we justify the effectiveness of projection correlation by checking the trend of projection correlation during the training process of $\mathcal{M}_3, \mathcal{M}_4$, and $\mathcal{M}_5$ in the **Imbalance** case. Figure 5 (A), (B), (C) show how the projection correlations of layers 4,5,6 evolve during the training processes. The projection correlations converge fast in all three models. The projection correlations of Layer 6 increase after training. However, if we only train Layer 6 ($\mathcal{M}_5$), the final projection correlation of Layer 6 is lower compared with the other two models. Moreover, training Layer 5 increases its projection correlation, while training Layer 4 does not help too much. Therefore, the phase transition phenomenon happens between Layer 4 and Layer 5.

5.2. Experiment on Caltech101 dataset

In this experiment, we explore the proposed forward-adding-layer-selection algorithm on transfer learning between ImageNet [3] and Caltech101 images [4]. Besides, we compare our method with the prediction-based algorithm as well as the fine-tuning method (Howard & Ruder, 2018). The fine-tuning method is a popular transfer learning scheme and has achieved encouraging results. Our layer selection approach provides a complement to the fine-tuning method. We consider two deep learning models Inception-V3 (Szegedy et al., 2016) and Densenet-121(Huang et al., 2017). The pre-

[3] http://www.image-net.org/
[4] www.vision.caltech.edu/Image_Datasets/Caltech101/

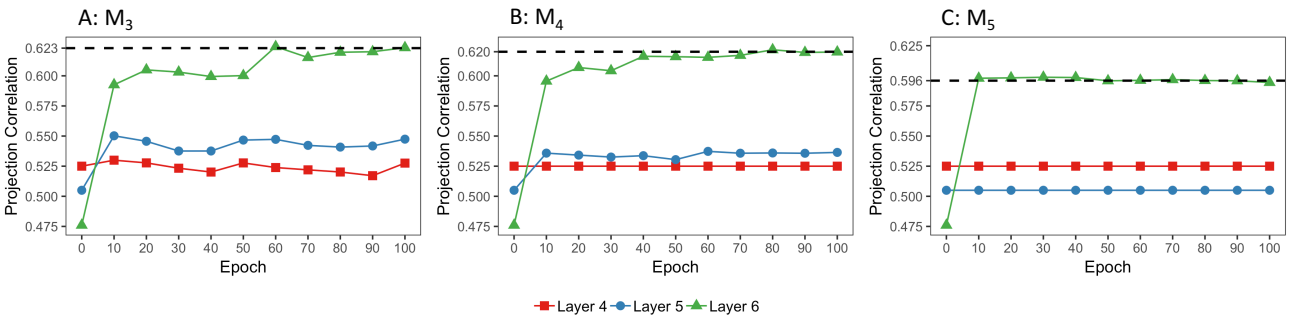

Figure 5. Projection correlations of layers 4,5,6 during the training processes of $\mathcal{M}_3$, $\mathcal{M}_4$, and $\mathcal{M}_5$ in the **Imbalance** case.

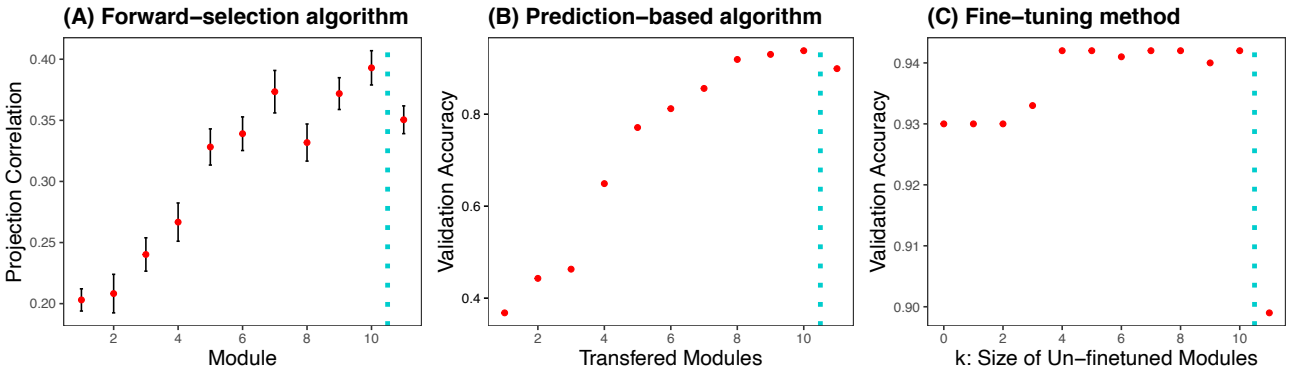

Figure 6. Selection of General Modules of Inception-V3. The cyan dashed lines indicate the cut-off between general and specific modules estimated by each method.

trained Inception-V3 and Densenet-121 models [5] contain 11 inception modules and 4 dense blocks respectively. To save space, we refer to (Szegedy et al., 2016) and (Huang et al., 2017) for detailed introductions of these architectures. For very deep networks like Inception, DenseNet, ResNet (He et al., 2016), etc., we suggest computing the projection correlation using the last layer of each module/block to save computational time.

First, we use Algorithm 1 to select the general modules/blocks from the pre-trained Inception-V3 or Densenet-121 networks with respect to the target Caltech101 dataset. We calculate the sample mean and sample standard deviation of the projection correlation of each module/block over 20 replications. Each sample has 500 randomly selected images from the Caltech101 dataset. The results are presented in the panel (A) of Figure 6 and panel (A) of Figure 7 respectively. In comparison, the prediction-based algorithm consists of many nested transferred models where $\mathcal{M}_j$, transfers the first j modules/blocks from the pre-trained networks, and trains the new classifiers. The classifiers are trained using the Adam optimizer (Kingma & Ba, 2014) with the default learning rate of 0.001, the first decay rate of 0.9, and the second decay rate of 0.999. We stop the training process after 100 epochs. We present the classification accuracy on the Caltech101 dataset of each model in the panel (B) of Figure 6 and Figure 7 respectively. For the fine-tuning method, we split the pre-trained models into 3 groups: Module/Block $1-k$, Module/Block $(k+1)-11$ or 4, and the classifier. We set the learning rates of the three groups to be 0, 10^{-4}, and 10^{-3}. When $k=0$, we fine-tune all modules/blocks. When $0<k<11$ or 4, we freeze the first k modules/blocks and fine-tune $(k+1)$ to 11 or 4 modules/blocks. The classification accuracy of each fine-tuning method are shown in the panel (C) of Figure 6 and Figure 7 respectively.

According to Figure 6, all three methods suggest us transfer the first 10 modules of the Inception-V3 model. In addition, the pattern of projection correlation gives a good estimate of the pattern of classification accuracy in the prediction-based algorithm and fine-tuning method. Besides, according to Figure 7, the three methods unanimously suggest transferring all 4 blocks of the Densenet-121 model. Despite the similar performance, the computational cost of the projection correlation based method is much smaller than the other two competitors. For the prediction-based algorithm, each replication involves training 11 transferred models $\mathcal{M}_j, j=1,\cdots,11$. The total computational time is around

[5] Pre-trained model weights can be downloaded at https://keras.io/api/applications/.

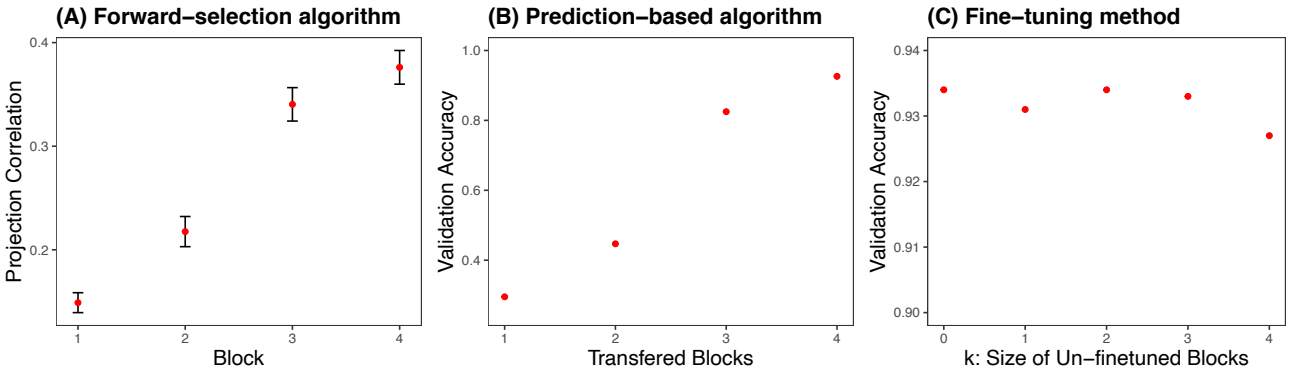

Figure 7. Selection of General Blocks of DenseNet-121.

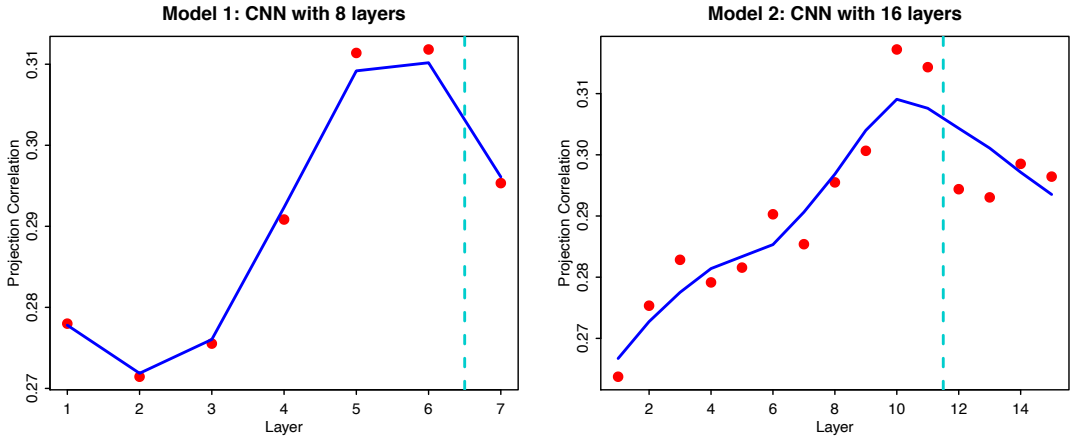

Figure 8. Layer selection in transfer learning for satellite imagery.

770 minutes per replication.[6]. In contrast, the running cost of our method is around 3 minutes per replication.

5.3. Learning poverty with satellite imagery

High-resolution satellite imagery has become an important and reliable data source of landscape features that can be correlated with economic activities. Recent research (Jean et al., 2016) proposed to use luminosity at night (nightlights) in satellite imagery as a noisy but easily obtained passive measure of poverty to fill the data gap in Africa. The traditional data collection effort, like survey, are facing various economical and political difficulties. The target dataset contains 643 satellite imagery in Uganda which is insufficient for a sophisticated CNN model. The goal is to learn the nightlights intensity through a CNN model. Given the limited sample size of the target dataset, the researchers of (Jean et al., 2016) propose to transfer an 8 layers CNN model pre-trained on ImageNet dataset to the target dataset. With a certain amount of engineering efforts (model comparison and fine-tuning), the researchers propose to transfer the first 6 convolutional layers from the pre-trained CNN to learn the satellite imagery data of Uganda. The estimated nightlights explained up to 75% of the variation in local-level economic outcomes.

Here we demonstrate the effectiveness of our method in layer selection for this interesting transfer learning application. We use two CNN models pre-trained on the ImageNet database. The first model is an 8 layers CNN, VGG8, with architecture as in (Chatfield et al., 2014). The second model is a 16 layers CNN, VGG16, with architecture as in (Simonyan & Zisserman, 2014). For both models, we use projection correlation to measure the generality of convolutional layers with respect to the task dataset (i.e. nightlights). The results are presented in Figure 8. For model 1, our method successfully selects the first 6 convolutional layers as general layers which is in line with the findings in (Jean et al., 2016). For model 2, our method suggests to transfer the first 11 convolutional layers.

[6]All models are trained on an Nvidia Tesla P100 GPU.

References

Azizpour, H., Razavian, A. S., Sullivan, J., Maki, A., and Carlsson, S. Factors of transferability for a generic convnet representation. *IEEE transactions on pattern analysis and machine intelligence*, 38(9):1790–1802, 2015.

Brock, A., Lim, T., Ritchie, J. M., and Weston, N. Smash: one-shot model architecture search through hypernetworks. *arXiv preprint arXiv:1708.05344*, 2017.

Chatfield, K., Simonyan, K., Vedaldi, A., and Zisserman, A. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*, 2014.

Geng, M., Wang, Y., Xiang, T., and Tian, Y. Deep transfer learning for person re-identification. *arXiv preprint arXiv:1611.05244*, 2016.

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

Hinton, G., Deng, L., Yu, D., Dahl, G., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Kingsbury, B., et al. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal processing magazine*, 29, 2012.

Hoeffding, W. A non-parametric test of independence. *The Annals of Mathematical Statistics*, 19(4):546–557, 1948.

Howard, J. and Ruder, S. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.

Huh, M., Agrawal, P., and Efros, A. A. What makes imagenet good for transfer learning? *arXiv preprint arXiv:1608.08614*, 2016.

Jean, N., Burke, M., Xie, M., Davis, W. M., Lobell, D. B., and Ermon, S. Combining satellite imagery and machine learning to predict poverty. *Science*, 353(6301):790–794, 2016.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Lawrence, S., Giles, C. L., Tsoi, A. C., and Back, A. D. Face recognition: A convolutional neural-network approach. *IEEE transactions on neural networks*, 8(1): 98–113, 1997.

Lenc, K. and Vedaldi, A. Understanding image representations by measuring their equivariance and equivalence. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 991–999, 2015.

Li, R., Zhong, W., and Zhu, L. Feature screening via distance correlation learning. *Journal of the American Statistical Association*, 107(499):1129–1139, 2012.

Long, M., C. Y. W. J. and Jordan, M. Learning transferable features with deep adaptation networks. In *In Proceedings of the 32nd International Conference on International Conference on Machine Learning (ICML)*, volume 13, pp. 97–105, 2015.

Long, J., Shelhamer, E., and Darrell, T. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440, 2015.

Long, M., Zhu, H., Wang, J., and Jordan, M. I. Unsupervised domain adaptation with residual transfer networks. In *Advances in Neural Information Processing Systems*, pp. 136–144, 2016.

Mahendran, A. and Vedaldi, A. Understanding deep image representations by inverting them. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5188–5196, 2015.

Maninis, K.-K., Pont-Tuset, J., Arbeláez, P., and Van Gool, L. Deep retinal image understanding. In *International conference on medical image computing and computer-assisted intervention*, pp. 140–148. Springer, 2016.

Mao, J. and Jain, A. K. Artificial neural networks for feature extraction and multivariate data projection. *IEEE transactions on neural networks*, 6(2):296–317, 1995.

Mou, L., Meng, Z., Yan, R., Li, G., Xu, Y., Zhang, L., and Jin, Z. How transferable are neural networks in nlp applications? *arXiv preprint arXiv:1603.06111*, 2016.

N. Papernot, P. McDaniel, S. J. M. F. Z. B. C. and Swami, A. The limitations of deep learning in adversarial settings. In *IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 372–387, 2016.

Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

Tieleman, T. and Hinton, G. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4 (2):26–31, 2012.

Yosinski, J., C. J. B. Y. and Lipson, H. How transferable are features in deep neural networks? In *In Advances in neural information processing systems (NIPS)*, pp. 3320–3328, 2014.

Zeiler, M. D. and Fergus, R. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pp. 818–833. Springer, 2014.

Zhu, L., Xu, K., Li, R., and Zhong, W. Projection correlation between two random vectors. *Biometrika*, 104(4):829–843, 2017.