Transferability Between Regression Tasks

Cuong N. Nguyen[1] **Phong Tran**[2,3] **Lam Si Tung Ho**[4] **Vu Dinh**[5]
Anh T. Tran[3] **Tal Hassner**[6] **Cuong V. Nguyen**[1]

[1]Florida International University [2]MBZUAI [3]VinAI Research
[4]Dalhousie University [5]University of Delaware [6]Meta AI

Abstract

Transfer learning has been a widely used technique to adapt a deep learning model trained for one task to another when there is a data distribution shift between these tasks. To improve the effectiveness of transfer learning and to understand relationships between tasks, we consider the problem of transferability estimation between regression tasks and propose two novel transferability estimators that are simple, computationally efficient, yet effective and theoretically grounded. We test our proposed methods extensively in various challenging, practical scenarios and show they significantly outperform existing state-of-the-art regression task transferability estimators in both accuracy and efficiency.

1 Introduction

Transfer learning [15, 21] has been a popular technique to adapt a deep learning model trained for one task to another when there is a data distribution shift between these tasks. To improve transfer learning and to understand task relationships, transferability estimation [3, 18, 24] has recently been studied. The aim of *transferability estimation* is to derive a computationally efficient metric (a score) that can predict the effectiveness of transferring a deep learning model from source to target tasks. This problem has recently gained attention as a means for model and task selection [3, 6, 18, 24, 29] that can potentially improve the performance and reduce the cost of transfer learning. Transferability estimators were also developed and used for checkpoint ranking [12, 14] and few-shot learning [23].

Nearly all existing methods consider only the transferability of classification tasks [3, 8, 13, 14, 17, 18, 22, 24] with very few designed for regression tasks [13, 29]. Furthermore, those few methods proposed for estimating regression model transferability have so far been either complex and inefficient [29] or ineffective [13], as we will show in our experiments. Compared to these previous work, we offer *efficient and theoretically grounded* transferability estimation for regression problems.

Specifically, we propose two families of simple transferability estimators that utilize the penalized mean squared error (MSE) of a linear regression model computed from the source and target training sets. The first family, *Linear MSE*, uses features extracted from a model trained on the source task (the *source model*), while the second family, *Label MSE*, uses the *dummy* labels obtained from the source model. In special cases where the source and target data share the inputs, the Label MSE estimators can be computed efficiently from true labels without a source model. Besides simplicity, our methods also have theoretical properties relating to the expected risk of the target model.

We conduct experiments on two real-world keypoint detection datasets, CUB-200-2011 [26] and OpenMonkey [28], to show the advantages of our methods. The results clearly demonstrate that the proposed methods outperform recently published, state-of-the-art (SotA) transferability estimators for regression problems, such as LogME [29] and TransRate [13], in both effectiveness and efficiency.

• **Other Related Work.** Besides transferability estimation, our paper is also related to previous work that developed theoretical bounds for transfer learning [2, 4, 5, 16, 25, 27]. These bounds are

Workshop on Distribution Shifts, 36th Conference on Neural Information Processing Systems (NeurIPS 2022).

usually hard to compute in practice and thus unsuitable for transferability estimation. Some previous transferability estimators have theoretical bounds on the empirical loss [18, 24], but they were shown for classification and did not bound the expected loss. Our work, on the other hand, focuses on regression and proves generalization bounds for the expected loss.

2 Transferability Between Regression Tasks

• **Transfer learning settings.** Consider a source training set $\mathcal{D}_s = \{(x_i^s, y_i^s)\}_{i=1}^{n_s}$ and a target training set $\mathcal{D}_t = \{(x_i^t, y_i^t)\}_{i=1}^{n_t}$ consisting of n_s and n_t examples respectively, where $x_i^s, x_i^t \in \mathbb{R}^d$ are d-dimensional input vectors, $y_i^s \in \mathbb{R}^{d_s}$ is a d_s-dimensional source label vector, and $y_i^t \in \mathbb{R}^{d_t}$ is a d_t-dimensional target label vector. Note that we allow multi-output regression tasks where $d_s, d_t > 1$.

From the source dataset $\mathcal{D}_s$, we train a deep learning model (w^*, h^*) consisting of a feature extractor w^* and a regression head h^* that minimizes the empirical MSE loss. That is, $w^*, h^* = \operatorname{argmin}_{w,h} \mathcal{L}(w, h; \mathcal{D}_s)$, where $w : \mathbb{R}^d \to \mathbb{R}^{d_r}$ is a feature extractor network, $h : \mathbb{R}^{d_r} \to \mathbb{R}^{d_s}$ is a source regression head network, and $\mathcal{L}(w, h; \mathcal{D}_s) = \frac{1}{n_s} \sum_{i=1}^{n_s} \|y_i^s - h(w(x_i^s))\|^2$ is the empirical MSE loss of the whole model (w, h) on $\mathcal{D}_s$, with $\|\cdot\|$ being the ℓ_2 norm. In practice, we usually consider the source model as a whole (e.g., a ResNet model [11]) and use its first l layers from the input (for some chosen number l) as the feature extractor w. The regression head h is the remaining part of the model from the l-th layer to the output layer, and the prediction for any input x is $h(w(x))$.

After training the optimal source model (w^*, h^*), we perform transfer learning to the target task by freezing the optimal feature extractor w^* and re-training a new regression head k^* using the target dataset $\mathcal{D}_t$, also by minimizing the empirical MSE loss: $k^* = \operatorname{argmin}_k \mathcal{L}(w^*, k; \mathcal{D}_t) = \operatorname{argmin}_k \left\{ \frac{1}{n_t} \sum_{i=1}^{n_t} \|y_i^t - k(w^*(x_i^t))\|^2 \right\}$, where $k : \mathbb{R}^{d_r} \to \mathbb{R}^{d_t}$ is a target regression head network that may have a different architecture than that of h. In general, the regression heads h and k may contain multiple layers and are not necessarily linear. This transfer learning setting is usually called *head re-training* [9, 19, 20] and will be used in our theoretical analysis. In our experiments, we also consider another transfer learning setting called *fine-tuning*, where we fine-tune the trained feature extractor w^* on the target set in addition to re-training k^* [1, 7, 10].

• **Transferability Estimators.** A transferability estimator measures the effectiveness of transfer learning given a pair of source and target tasks. Intuitively, we can use the MSE of the transferred target model $\mathcal{L}(w^*, k^*; \mathcal{D}_t)$ as an estimator. However, that means we need to run the actual transfer learning algorithm, which could be expensive if the network architecture of the target regression heads (e.g., k and k^*) is complex. A simple way to reduce this computational cost is to approximate $\mathcal{L}(w^*, k^*; \mathcal{D}_t)$ using an ℓ_2 regularized linear regression (or Ridge regression) head; that is, we consider the family of Linear MSE estimators below. In our definitions, $\|\cdot\|_F$ is the Frobenius norm.

Definition 1. *A **Linear MSE** transferability estimator between a source dataset $\mathcal{D}_s$ and a target dataset $\mathcal{D}_t$, with a regularization parameter $\lambda \geq 0$, is defined as: $\mathcal{T}_\lambda^{\text{lin}}(\mathcal{D}_s, \mathcal{D}_t) := \min_{A,B} \left\{ \frac{1}{n_t} \sum_{i=1}^{n_t} \|y_i^t - Aw^*(x_i^t) - B\|^2 + \lambda \|A\|_F^2 \right\}$, where $A \in \mathbb{R}^{d_r \times d_t}$ and $B \in \mathbb{R}^{d_t}$.*

We use a regularizer in the estimator to avoid overfitting when $\mathcal{D}_t$ is small. Previous work such as LogME [29] prevents overfitting by taking a more complicated and expensive Bayesian approach. Here we argue that a simple regularization can tackle this issue effectively and more efficiently. Given a pre-trained feature extractor w^*, we can compute $\mathcal{T}_\lambda^{\text{lin}}(\mathcal{D}_s, \mathcal{D}_t)$ efficiently by solving a Ridge regression problem. If the target head is restricted to only linear regression model, the non-regularized Linear MSE $\mathcal{T}_0^{\text{lin}}(\mathcal{D}_s, \mathcal{D}_t)$ is equal to the MSE of the transferred target model (w^*, k^*) on $\mathcal{D}_t$. If the target head has more than one layer with a non-linear activation function, $\mathcal{T}_\lambda^{\text{lin}}(\mathcal{D}_s, \mathcal{D}_t)$ can be regarded as using a regularized linear model to approximate this non-linear head.

Although Linear MSE scores can be computed efficiently, they may still be expensive if the feature vectors $w^*(x_i^t)$ are high-dimensional. Furthermore, in many cases, we need to compute and compare the scores for several pairs of datasets, resulting in high computational costs. To further reduce the costs, we propose the Label MSE transferability estimators below, which replaces $w^*(x_i^t)$ by the "dummy" source label $z_i = h^*(w^*(x_i^t))$. Using dummy labels from the trained source model (w^*, h^*) is a technique previously used in the LEEP transferability scores for classification tasks [18].

Definition 2. *A **Label MSE** transferability estimator between a source dataset $\mathcal{D}_s$ and a target dataset $\mathcal{D}_t$, with a regularization parameter $\lambda \geq 0$, is defined as: $\mathcal{T}_\lambda^{\text{lab}}(\mathcal{D}_s, \mathcal{D}_t) := \min_{A,B} \left\{ \frac{1}{n_t} \sum_{i=1}^{n_t} \|y_i^t - Az_i - B\|^2 + \lambda \|A\|_F^2 \right\}$, where $A \in \mathbb{R}^{d_s \times d_t}$, $B \in \mathbb{R}^{d_t}$, and $z_i = h^*(w^*(x_i^t)), \forall i$.*

Table 1: Correlation coefficients when transferring from OpenMonkey to CUB-200-2011.

Transfer setting	Label-based method				Feature-based method			
	LabLogME	LabTransRate	LabMSE0	LabMSE1	LogME	TransRate	LinMSE0	LinMSE1
Head re-training	0.958	0.165	**0.992***	**0.992***	0.969	0.121	0.982	0.988*
Half fine-tuning	0.706	0.392	**0.882***	**0.882***	0.870	0.304	0.865	0.872*
Full fine-tuning	0.691	0.410	**0.870***	**0.870***	0.861*	0.311	0.854	0.861*

From our definitions, the lower the Linear MSE or Label MSE, the better the transferability. Since the size of z_i is usually much smaller than that of $w^*(x_i^t)$ (i.e., $d_s \ll d_r$), computing the Label MSE is usually cheaper than computing the Linear MSE. We prove novel properties of Label MSE below.

• **Theoretical Results.** We prove some theoretical properties (in the form of generalization bounds) for the Label MSE with ReLU feed-forward neural networks. We first make the standard assumption that the source data and the target data are drawn iid from the unknown distributions $\mathbb{P}(X^s, Y^s)$ and $\mathbb{P}(X^t, Y^t)$ respectively. Given any model (w, k) for the target task, the expected risk of (w, k) is defined as $\mathcal{R}(w, k) = \mathbb{E}_{(x^t,y^t)\sim\mathbb{P}(X^t,Y^t)}\{\|y^t - k(w(x^t))\|^2\}$. Our main theoretical result is Theorem 1 below. In this theorem, L is the number of layers of the ReLU feed-forward neural network (w, k), and we assume the number of hidden nodes and parameters in each layer are upper bounded by H and $M \geq 1$ respectively.

Theorem 1. *For any $\lambda \geq 0$ and $\delta > 0$, with probability at least $1 - \delta$, we have: $\mathcal{R}(w^*, k^*) \leq \mathcal{T}_\lambda^{\text{lab}}(\mathcal{D}_s, \mathcal{D}_t) + C(d, d_t, M, H, L, \delta)/\sqrt{n_t}$.*

This theorem shows that the expected risk is upper bounded by the Label MSE plus a complexity term $C(d, d_t, M, H, L, \delta)/\sqrt{n_t}$ that depends on the architecture of the target network and the target dataset (i.e., the input/output dimensions and the dataset size). When the complexity term is small (e.g., when n_t is large), the Label MSE will be a tighter bound for the expected risk.

• **Setting with Shared Inputs.** When the source and target data have the same inputs, i.e., $x_i^s = x_i^t = x_i$ and $n_s = n_t = n$, we can compute the Label MSE directly from the labels without training the source model (w^*, h^*) or computing the dummy labels. Formally, we can use the following new definition for the Label MSE: $\widehat{\mathcal{T}}_\lambda^{\text{lab}}(\mathcal{D}_s, \mathcal{D}_t) := \min_{A,B}\left\{\frac{1}{n}\sum_{i=1}^n \|y_i^t - Ay_i^s - B\|^2 + \lambda\|A\|_F^2\right\}$, where we train a Ridge regression model directly from the corresponding label pairs (y_i^s, y_i^t). This new transferability estimator is *less expensive* since we do not need to train the source model. We can also prove similar generalization bounds with $\widehat{\mathcal{T}}_\lambda^{\text{lab}}(\mathcal{D}_s, \mathcal{D}_t)$.

3 Experiments

We evaluate our transferability estimators on the keypoint regression tasks using two large-scale public datasets: (1) **CUB-200-2011** [26]: This dataset contains 11,788 bird images with 15 labeled keypoints indicating 15 different parts of a bird body, and (2) **OpenMonkey** [28]: This dataset contains over 100,000 monkey images in natural contexts, annotated with 17 body keypoints.

We use ResNet34 [11] as the model backbone due to its good performance. Following [18, 24], we investigate how well our transferability estimators correlate (using Pearson correlation) with the *test* MSE of the target model obtained from actual transfer learning. We consider three transfer learning algorithms: (1) **head re-training**: We fix all layers of the source model up until the penultimate layer and re-train the last fully-connected (FC) layer; (2) **half fine-tuning**: We fine-tune the last convolutional block and all the FC layers of the source model (around half of the parameters in the network), while keeping all other layers fixed; and (3) **full fine-tuning**: We fine-tune the whole source model using the target training set.

For our methods, we consider $\lambda = 0$ (**LinMSE0** and **LabMSE0**) and $\lambda = 1$ (**LinMSE1** and **LabMSE1**). We compare with two recent SotA baselines for regression: **LogME** [29] and **TransRate** [13]. Besides their usual versions, we also consider the label-based versions where these estimators are computed from dummy labels and target labels (**LabLogME** and **LabTransRate**). As in [13], we divide the target label values into 5 equal-sized bins to compute TransRate and LabTransRate.

• **General Transfer Between Two Different Domains.** We first consider the general case where source models are trained on OpenMonkey and then transferred to CUB-200-2011. Specifically, we

Table 2: Correlation coefficients when transferring between tasks with shared inputs. The last row shows the average running time (in milliseconds) on CUB-200-2011.

Dataset	Transfer setting	Label-based method				Feature-based method			
		LabLogME	LabTransRate	LabMSE0	LabMSE1	LogME	TransRate	LinMSE0	LinMSE1
Open Monkey	Head re-training	0.890	0.666	0.973*	0.973*	0.646	0.711	0.949	**0.981***
	Half fine-tuning	0.615	0.340	0.754*	0.754*	0.391	0.488	**0.893***	0.787
	Full fine-tuning	0.569	0.269	0.705*	0.705*	0.352	0.439	**0.852***	0.739
CUB-200-2011	Head re-training	0.547	0.008	0.916*	0.916*	0.889	0.029	0.921	**0.932***
	Half fine-tuning	0.400	0.006	0.536*	0.536*	0.560	0.006	**0.628***	0.607
	Full fine-tuning	**0.120***	0.001	0.056	0.056	0.099	0.100*	0.097	0.093
Average running time (ms)		3.55	4.11	2.87	**2.58***	112.99	94.57	107.48	27.41*

train a source model for each of the 17 OpenMonkey keypoints and transfer them to each of the 15 CUB-200-2011 keypoints, resulting in a total of 255 final models. Since each keypoint consists of x and y positions, all source and target tasks in this experiment have two dimensional labels. The actual MSEs of these models are computed on the respective test sets and then used to calculate the Pearson correlation coefficients with the transferability estimators. In this experiment, the label-based estimators are computed from the dummy labels and the actual target labels.

Table 1 shows the results for this experiment.[1] From the results, TransRate and LabTransRate perform poorly in this regression setting, while our methods are equal or better than LogME and LabLogME in all cases, especially when using $\lambda = 1$ (LinMSE1) or dummy labels (LabMSE0 and LabMSE1). We observe that adding a regularizer in LabMSE1 does not significantly change its correlation coefficient compared to LabMSE0. This is because we scaled the labels to $[0, 1]$ as a pre-processing step, so a regularizer with $\lambda = 1$ has negligible effect on the scores. It is also surprising that LabMSE0 and LabMSE1 are better than LinMSE0 and LinMSE1. One possible explanation for this phenomenon is that the dummy labels (i.e., body parts of monkeys) give more information about the target labels (i.e., body parts of birds) than the extracted features.

• **Transfer Between Tasks with Shared Inputs.** In this experiment, we consider the setting where the source and target tasks have the same inputs. Since each image in our datasets contains multiple keypoints, we can use any two different keypoints on the same dataset as source and target tasks. In total, we can construct 210 source-target pairs for CUB-200-2011 and 272 pairs for OpenMonkey. The labels for all source and target tasks are also 2-dimensional real values. We repeat the previous experiment with these source-target pairs for each dataset. The main difference in this experiment is that we use the *true* source labels (instead of dummy labels) when computing LabLogME, LabTransRate, LabMSE0, and LabMSE1. Thus, these estimators can be computed without any source models in this setting, and hence incurring very low computational costs.

Results for this experiment are in Table 2. Here both versions of TransRate perform poorly on CUB-200-2011, while TransRate is slightly better than LogME on OpenMonkey. In the first 5 settings of Table 2, LabMSE0 and LabMSE1 both outperform LabLogME and LabTransRate, while LinMSE0 and LinMSE1 both outperform LogME and TransRate. In these settings, the LinMSE methods achieve the best correlations overall. In the last setting (full fine-tuning on CUB-200-2011), all methods perform poorly.

• **Efficiency of Our Methods.** In the last row of Table 2, we compare the average running time of different methods in the shared inputs setting on CUB-200-2011. We can observe that LabMSE0, LabMSE1, LinMSE0, and LinMSE1 are all faster than the corresponding label-based or feature-based baselines. Overall, LabMSE1 and LinMSE1 achieve the best running time among the label-based and feature-based methods respectively.

4 Conclusion

We proposed Linear MSE and Label MSE, two families of simple but effective transferability estimation methods for regression, and proved novel theoretical bounds for the expected risk of the target models using these estimators. Our experiments showed that the proposed methods are superior to recent, relevant SotA methods in terms of both effectiveness and efficiency.

[1] In our tables, bold numbers indicate the best results in each row, while asterisks (*) indicate the best results among the corresponding label-based or feature-based methods.

References

[1] Pulkit Agrawal, Ross Girshick, and Jitendra Malik. Analyzing the performance of multilayer neural networks for object recognition. In *ECCV*, 2014.

[2] Kamyar Azizzadenesheli, Anqi Liu, Fanny Yang, and Animashree Anandkumar. Regularized learning for domain adaptation under label shifts. In *ICLR*, 2019.

[3] Yajie Bao, Yang Li, Shao-Lun Huang, Lin Zhang, Lizhong Zheng, Amir Zamir, and Leonidas Guibas. An information-theoretic approach to transferability in task transfer learning. In *ICIP*, 2019.

[4] Shai Ben-David and Reba Schuller. Exploiting task relatedness for multiple task learning. *Learning theory and kernel machines*, 2003.

[5] John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman. Learning bounds for domain adaptation. In *NeurIPS*, 2007.

[6] Daniel Bolya, Rohit Mittapalli, and Judy Hoffman. Scalable diverse model selection for accessible transfer learning. In *NeurIPS*, 2021.

[7] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *BMVC*, 2014.

[8] Aditya Deshpande, Alessandro Achille, Avinash Ravichandran, Hao Li, Luca Zancato, Charless Fowlkes, Rahul Bhotika, Stefano Soatto, and Pietro Perona. A linearized framework and a new benchmark for model selection for fine-tuning. *arXiv:2102.00084*, 2021.

[9] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*, 2014.

[10] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[12] Jiaji Huang, Qiang Qiu, and Kenneth Church. Exploiting a zoo of checkpoints for unseen tasks. In *NeurIPS*, 2021.

[13] Long-Kai Huang, Junzhou Huang, Yu Rong, Qiang Yang, and Ying Wei. Frustratingly easy transferability estimation. In *ICML*, 2022.

[14] Yandong Li, Xuhui Jia, Ruoxin Sang, Yukun Zhu, Bradley Green, Liqiang Wang, and Boqing Gong. Ranking neural checkpoints. In *CVPR*, 2021.

[15] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. In *ICML*, 2017.

[16] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation: Learning bounds and algorithms. In *COLT*, 2009.

[17] Cuong N Nguyen, Lam Si Tung Ho, Vu Dinh, Tal Hassner, and Cuong V Nguyen. Generalization bounds for deep transfer learning using majority predictor accuracy. In *ISITA*, 2022.

[18] Cuong V Nguyen, Tal Hassner, Matthias Seeger, and Cedric Archambeau. LEEP: A new measure to evaluate transferability of learned representations. In *ICML*, 2020.

[19] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *CVPR*, 2014.

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

[21] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In *ICANN*, 2018.

[22] Yang Tan, Yang Li, and Shao-Lun Huang. OTCE: A transferability metric for cross-domain cross-task representations. In *CVPR*, 2021.

[23] Xinyi Tong, Xiangxiang Xu, Shao-Lun Huang, and Lizhong Zheng. A mathematical framework for quantifying transferability in multi-source transfer learning. In *NeurIPS*, 2021.

[24] Anh T Tran, Cuong V Nguyen, and Tal Hassner. Transferability and hardness of supervised classification tasks. In *ICCV*, 2019.

[25] Nilesh Tripuraneni, Michael Jordan, and Chi Jin. On the theory of transfer learning: The importance of task diversity. In *NeurIPS*, 2020.

[26] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The Caltech-UCSD Birds-200-2011 Dataset. *Technical Report*, 2011. `https://authors.library.caltech.edu/27452/`.

[27] Boyu Wang, Jorge Mendez, Mingbo Cai, and Eric Eaton. Transfer learning via minimizing the performance gap between domains. In *NeurIPS*, 2019.

[28] Yuan Yao, Abhiraj Abhiraj Mohan, Eliza Bliss-Moreau, Kristine Coleman, Sienna M Freeman, Christopher J Machado, Jessica Raper, Jan Zimmermann, Benjamin Y Hayden, and Hyun Soo Park. OpenMonkeyChallenge: Dataset and Benchmark Challenges for Pose Tracking of Non-human Primates. *bioRxiv*, 2021. `http://openmonkeychallenge.com/`.

[29] Kaichao You, Yong Liu, Jianmin Wang, and Mingsheng Long. LogME: Practical assessment of pre-trained models for transfer learning. In *ICML*, 2021.