Newer is not always better: Rethinking transferability metrics, their peculiarities, stability and performance

Shibal Ibrahim[1], **Natalia Ponomareva**[2], **and Rahul Mazumder**[1]

[1]Massachusetts Institute of Technology, Cambridge, MA, USA
`{shibal, rahul}@mit.edu`
[2]Google Research, New York, NY, USA
`{nponomareva}@google.com`

Abstract

Fine-tuning of large pre-trained image and language models on small customized datasets has become increasingly popular for improved prediction and efficient use of limited resources. Fine-tuning requires identification of best models to transfer-learn from and quantifying transferability prevents expensive re-training on *all* of the candidate models/tasks pairs. In this paper, we show that the statistical problems with covariance estimation drive the poor performance of H-score [1] — a common baseline for newer metrics — and propose shrinkage-based estimator. This results in up to 80% absolute gain in H-score correlation performance, making it competitive with the state-of-the-art LogME measure by [26]. Our shrinkage-based H-score is $3-55$ times faster than LogME. Additionally, we look into a less common setting of target (as opposed to source) task selection. We highlight previously overlooked problems in such settings with different number of labels, class-imbalance ratios etc. for some recent metrics e.g., NCE [24], LEEP [18] that misrepresented them as leading measures. We propose a correction and recommend measuring correlation performance against relative accuracy in such settings. We support our findings with $\sim$ 65,000 (fine-tuning trials) experiments.

1 Introduction

Transfer learning (TL) is a set of techniques of using abundant somewhat related source data $p(X^{(s)}, Y^{(s)})$ to ensure that a model can generalize well to the target domain, defined as either little amount of labelled data $p(X, Y)$ (supervised), and/or a lot of unlabelled data $p(X)$ (unsupervised TL). TL is most commonly achieved via fine-tuning. Fine-tuning (FT) is a process of adapting a model trained on source data by using target data to do several optimization steps (for example, SGD) that update the model parameters. FT is becoming increasing popular because large models like ImageNet [12], Bert [5] etc. are released by companies and are easily modifiable. Training such large models from scratch is often prohibitively expensive for the end user. In this paper, we are primarily interested in effectively measuring transferability before training of the final model begins. Given a source data/model, a **transferability measure** (TM) quantifies how much knowledge of source domain/model is transferable to the target model. If a measure is capable of efficiently and accurately measuring transferability across arbitrary tasks, the problem of transfer learning is greatly simplified by using the measure to search over candidate sources and targets.

Related Work Negative Conditional Entropy (NCE) [24] uses pre-trained source model and evaluates conditional entropy between target pseudo labels (source models' assigned labels) and real target labels. Log Expected Empirical Predictor (LEEP) [18] modifies NCE by using soft predictions from the source model. [3] propose representing each output class by the mean of all images from that class and computing Earth Mover's distance between the centroids of the source classes and target classes. [15] proposed $\mathcal{N}$LEEP that fits a Gaussian mixture model on the target feature embeddings and computes the LEEP score between the probabalistic assignment of target features to different clusters and the target labels. [10] proposed TransRate — a computationally-friendly surrogate of mutual information (using coding rate) between the target feature embeddings and the target labels. [1] introduced H-score that takes into account inter-class feature variance and feature redundancy. [26] proposed LogME that considers an optimization problem rooted in Bayesian statistics to maximize the marginal likelihood. [4] introduced LFC to measure in-class similarity of target feature embeddings.

Transferability setup We consider two FT scenarios based on existing literature. (i) *Source Model Selection (SMS)*: For a particular target data/task, this regime aims to select the "optimal" source model (or data) to transfer-learn from, from a collection of candidate models/data. (ii) *Target Task Selection (TTS)*: For a particular (source) model, this regime aims to find the most related target data/task. In addition, we consider two FT strategies: (a) *Linear FT*: All layers except for the penultimate layer are frozen. Only the weights of the head classifier are re-trained while fine-tuning. (b) *Nonlinear FT*: Any arbitrary layer can be designated as a feature extractor, up to which all the layers are frozen; all subsequent (nonlinear) layers along with head are re-trained.

Contributions Our contributions are three-fold: (i) We show that H-score, a baseline for newer measures, suffers from instability due to poor covariance estimation. We propose shrinkage-based estimation of H-score with regularized covariance estimation techniques. We show 80% absolute increase over the original H-score and show superior performance in 9/15 cases against all newer TMs across various FT scenarios. (ii) We present a fast implementation of our estimator that is $3-55$ times faster than state-of-the-art LogME measure. Unlike LogME, our estimator is tractable even for really high-dimensional feature embeddings $\sim 10^5$. (iii) We identify problems with 3 other measures (NCE, LEEP and $\mathcal{N}$LEEP) in target task selection when either the number of classes or the class imbalance varies across candidate target tasks. We propose measuring correlation against relative target accuracy (instead of vanilla accuracy) in such scenarios. 65,000 FT experiments with ImageNet models and different regimes constructed from CIFAR-100/CIFAR-10 validates our proposals.
This paper is organized as follows. Section 2 addresses shortcomings of the pioneer TM (H-Score) that arise due to limited target data. Section 3 identifies problems with recent NCE, LEEP and $\mathcal{N}$LEEP metrics and proposes corrections. Finally, Section 4 presents a meta study of all metrics.

2 Improved estimation of H-score for limited target data

H-score [1] is the pioneer measure, often used as a baseline for newer measures that often demonstrate the improved performance. It characterizes discriminatory strength of feature embedding for classification: $\mathrm{H}(f) = \mathrm{tr}({\mathbf{\Sigma}^{f}}^{-1}\mathbf{\Sigma}^{z})$, where d is the embedding dimension, $\boldsymbol{f}_i = h(\boldsymbol{x}_i) \in \mathbb{R}^d$ is the target feature embeddings when the feature extractor $h(.)$ from the source model is applied to the target sample $\boldsymbol{x}_i$, $\boldsymbol{F} \in \mathbb{R}^{n_t \times d}$ denotes the target feature matrix, $Y \in \{1, \cdots, C\}$ are the target labels, $\mathbf{\Sigma}^f \in \mathbb{R}^{d\times d}$ denotes the sample feature covariance matrix of $\boldsymbol{f}$, $\boldsymbol{z} = \mathrm{E}\left[\boldsymbol{f}|Y\right] \in \mathbb{R}^d$ and $\boldsymbol{Z} \in \mathbb{R}^{n_t \times d}$ denotes the target conditioned feature matrix, $\mathbf{\Sigma}^z \in \mathbb{R}^{d\times d}$ denotes the sample covariance of $\boldsymbol{z}$.

We hypothesize that the sub-optimal performance of H-Score (compared to that of more recent metrics) for measuring transferability in many of the evaluation cases, e.g., in [18], is due to lack of robust estimation of H-Score — see Fig. 1 for a synthetic example showing the non-reliability of empirical H-score over various sample sizes when compared with its population version. Given that many of the deep learning models in the context of TL have high-dimensional feature embedding space — typically larger than the number of target samples — the estimation of the two covariance matrices in H-score becomes challenging: the sample covariance matrix of the feature embedding has a large condition number[1] in small data regimes. In many cases, it cannot even be inverted. [1] used a pseudo-inverse of the covariance matrix $\mathbf{\Sigma}^f$. However, this method of estimating a precision matrix can be sub-optimal as inversion can amplify estimation error [13]. We propose to use well-conditioned shrinkage estimators from rich literature in statistics on the estimation of high-dimensional covariance matrices [20]. Such shrinkage estimators can offer significant gain in the performance of H-score in predicting transferability, making it a leading TM.

[1]Condition number of a positive semidefinite matrix A, is the ratio of its largest and smallest eigenvalues.

Proposed Transferability Measure We propose the following shrinkage based H-score:

$$\mathrm{H}_\alpha(f) = \mathrm{tr}\left(\boldsymbol{\Sigma}_\alpha^{f^{-1}} \cdot (1-\alpha)\boldsymbol{\Sigma}^z\right), \tag{1}$$

Estimating $\boldsymbol{\Sigma}_\alpha^f$: While there are several possibilities to obtain a regularized covariance matrix [20], we present an approach that considers a linear operation on the eigenvalues of the sample version of the feature embedding covariance matrix. Similar ideas of using well-conditioned plug-in covariance matrices are used in the context of discriminant analysis [8]. In particular, we improve the conditioning of the covariance matrix by considering its weighted convex combination with a scalar multiple of the identity matrix: $\boldsymbol{\Sigma}_\alpha^f = (1-\alpha)\boldsymbol{\Sigma}^f + \alpha\sigma\boldsymbol{I}_d$, where $\alpha \in [0,1]$ is the shrinkage parameter and σ is the average variance computed as $\mathrm{tr}(\boldsymbol{\Sigma}^f)/d$. Note that the inverse of $\boldsymbol{\Sigma}_\alpha^f$ can be computed for every α, by using the eigen-decomposition of $\boldsymbol{\Sigma}^f$. The shrinkage parameter controls the bias and variance trade-off; the optimal α needs to be selected. This distribution-free estimator is well-suited for our application as the explicit convex linear combination is easy to compute and makes the covariance estimates well-conditioned and more accurate [13, 2, 21].

Understanding $(1-\alpha)\boldsymbol{\Sigma}^z$: The scaling factor $(1-\alpha)$ can be understood in terms of ridge-regularized covariance estimation:

$$1/(1+\lambda)\cdot\boldsymbol{\Sigma}^z = \mathrm{argmin}_{\hat{\boldsymbol{\Sigma}}}\, ||\hat{\boldsymbol{\Sigma}} - \boldsymbol{\Sigma}^z||_2^2 + \lambda||\hat{\boldsymbol{\Sigma}}||_2^2, \tag{2}$$

where $\lambda \geq 0$ is the ridge penalty. Choosing $\lambda = \alpha/(1-\alpha)$, it becomes clear that $(1-\alpha)\boldsymbol{\Sigma}^z$ is the regularized covariance.

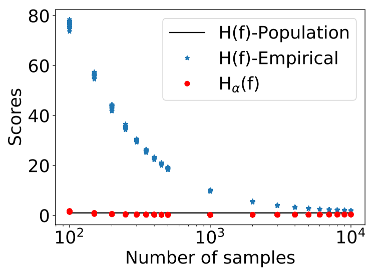

Figure 1: Stability of original $\mathrm{H}(f)$ and the shrinkage-based $\mathrm{H}_\alpha(f)$ with respect to number of samples. The original H-Score is ~ 75 times larger than the population version of the H-Score (estimated with a sample size of 10^6). In contrast, the shrinkage-based H-Score is significantly more reliable.

Choice of $\boldsymbol{\alpha}$ [13] proposed a covariance matrix estimator that minimizes mean squared error loss between the shrinkage based covariance estimator and the true covariance matrix. The optimization with respect to α considers the following objective:

$$\begin{aligned} &\min_{\alpha,v}\;\; \mathrm{E}[||\boldsymbol{\Sigma}^* - \boldsymbol{\Sigma}||^2] \\ &\quad \text{s.t. } \boldsymbol{\Sigma}^* = (1-\alpha)\boldsymbol{\Sigma}^f + \alpha v I, \quad \mathrm{E}[\boldsymbol{\Sigma}^f] = \boldsymbol{\Sigma}. \end{aligned} \tag{3}$$

where $||\boldsymbol{A}||^2 = \mathrm{tr}(\boldsymbol{A}\boldsymbol{A}^T)/d$. This optimization problem permits a closed-form solution for the optimal shrinkage parameter, which is given by:

$$\begin{aligned} \alpha^* &= \mathrm{E}[||\boldsymbol{\Sigma}^f - \boldsymbol{\Sigma}||^2]/\mathrm{E}[||\boldsymbol{\Sigma}^f - (\mathrm{tr}(\boldsymbol{\Sigma})/d)\cdot\boldsymbol{I}_d||^2] & (4)\\ &\simeq \min\{(1/n_t^2)\textstyle\sum_{i\in[n_t]}||\boldsymbol{f}_i\boldsymbol{f}_i^T - \boldsymbol{\Sigma}^f||^2/||\boldsymbol{\Sigma}^f - (\mathrm{tr}(\boldsymbol{\Sigma}^f)/d)\cdot\boldsymbol{I}_d||^2, 1\}. & (5) \end{aligned}$$

where (5) defines a valid estimator (not dependent on true covariance matrix) for practical use. For proof, we refer the readers to Section 2.1 and 3.3 in [13].

Additional Discussion The covariance $\boldsymbol{\Sigma}^z$ can not be shrunk independently of $\boldsymbol{\Sigma}^f$ in the estimation of $\mathrm{H}_\alpha(f)$— the two covariances are coupled by the law of total covariance:

$$\boldsymbol{\Sigma}^f = \mathrm{E}[\boldsymbol{\Sigma}^{f_Y}] + \boldsymbol{\Sigma}^z. \tag{6}$$

where $\boldsymbol{f}_Y$ denotes the feature embedding of target samples that belong to class $Y \in \mathcal{Y}$ and $\boldsymbol{\Sigma}^{f_Y} = \mathrm{Cov}(\boldsymbol{f}|Y)$ denotes the class-conditioned covariances. We can write

$$\begin{aligned} (1-\alpha)\boldsymbol{\Sigma}^f &= (1-\alpha)\mathrm{E}[\boldsymbol{\Sigma}^{f_Y}] + (1-\alpha)\boldsymbol{\Sigma}^z, \\ \boldsymbol{\Sigma}_\alpha^f = (1-\alpha)\boldsymbol{\Sigma}^f + \alpha\frac{\mathrm{tr}(\boldsymbol{\Sigma}^f)}{d}\boldsymbol{I}_d &= (1-\alpha)\mathrm{E}[\boldsymbol{\Sigma}^{f_Y}] + \alpha\frac{\mathrm{tr}(\boldsymbol{\Sigma}^f)}{d}\boldsymbol{I}_d + (1-\alpha)\boldsymbol{\Sigma}^z. \end{aligned} \tag{7}$$

Comparing (7) with (6), we see that the same shrinkage parameter α should be used when using shrinkage estimators, to preserve law of total covariance. The first two terms on the right side in (7) can be understood as shrinkage of class-conditioned covariances to the average (global) variance. The third term in (7) (e.g. $(1-\alpha)\boldsymbol{\Sigma}^z$) can then be understood as ridge shrinkage as in (2).

We first provide validation of shrinkage-based estimation of H-Score on synthetic classification data. We generated 1 million 1000-dimensional features with 10 classes using Sklearn multi-class dataset generation function [19]. Number of informative features is set to 500 with rest filled with random noise. We visualize the original and the population version of the H-score and the shrinkage-based H-Score for different sample sizes in Fig. 1. We observe that the original H-Score becomes highly unreliable as the number of samples decreases. In contrast, the shrunken estimation of H-Score is highly stable and has a small error when compared with the population H-Score.

We provide an efficient implementation for small target data ($C \leq n_t < d$) in Supplement Section S2.1 that leads to $3 - 55$ times faster computation than the state-of-the-art LogME measure (see Table S2 in Supplement Section S2.1.2).

3 A closer look at NCE, LEEP and $\mathcal{N}$LEEP measures

We pursue a deeper investigation of some of the newer metrics (reported to be superior to H-Score) and bring to light what appear to be some overlooked issues with these metrics in Target Task Selection (TTS) scenario. TTS has received less attention than Source Model Selection (SMS). We are the first to highlight these problematic aspects with NCE, LEEP and $\mathcal{N}$LEEP, which can potentially lead to the misuse of these metrics in measuring transferability. These measures are sensitive to the number of target classes (C) and tend to be smaller when C is larger (see Fig. 2[Left]). Therefore, use of these measures for target tasks with *different* C will most likely select the task with a smaller C. However, in practice, transferring to a task with a smaller C is not always easier; for example, reframing a multiclass classification into a set of binary tasks can create more difficult to learn boundaries [6]. Furthermore, the measures are also problematic if two candidate target tasks have different imbalances in their classes even if C is the same. The measures would predict higher transferability for imbalanced data regimes over balanced settings (see Fig. 2[Right]). However, imbalanced datasets are typically harder to learn. If these measures are correlated against vanilla accuracy, which tends to be higher as the imbalance increases e.g. for binary classification, the measures would falsely suggest they are good indicators of performance. Earlier work erroneously showed good correlation of these metrics against vanilla accuracy to show dominance of such metrics in TTS with different C [18, 23] and imbalance [23].

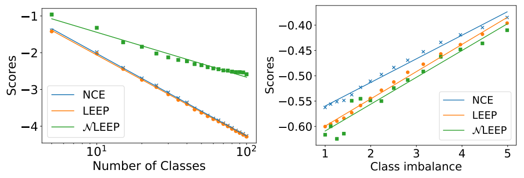

Figure 2: Relation of NCE, LEEP & $\mathcal{N}$LEEP to [Left] number of classes (log-scale) and [Right] class imbalance, $max(n_1, n_2)/min(n_1, n_2)$, for VGG19 on CIFAR100. For [Left], we randomly select 2-100 classes. For [Right], we randomly select 2 classes and vary the class imbalances.

Here, we propose a method to ameliorate the shortcomings of NCE, LEEP and $\mathcal{N}$LEEP to prevent misuse of these measures, so that they lead to more reliable conclusions. We propose to standardize the metrics by the entropy of the target label priors, leading to the following definitions:

$$\text{n-NCE} = 1 + \text{NCE}/\text{H}(Y), \quad \text{n-LEEP} = 1 + \text{LEEP}/\text{H}(Y), \quad \text{n-}\mathcal{N}\text{LEEP} = 1 + \mathcal{N}\text{LEEP}/\text{H}(Y).$$

We provide additional motivation (and proof) for these definitions in Supplement Section S3. For scenarios where candidate target tasks have different C, we propose an alternative evaluation criteria (*relative* accuracy) instead of vanilla accuracy — see Section 4 for more details. We provide empirical validation of the proposed normalization to these measures in Table 2 in Section 4. We also show that our proposed $\text{H}_\alpha(f)$ is the leading metric even in these scenarios.

4 Empirical Evaluation & Conclusion

We evaluate existing TMs and our proposed modifications in various FT regimes and data settings. We are inspired by [18] who consider TTS and SMS. The regimes highlight important aspects of TMs, e.g., dataset size, number of target classes, and feature dimension etc. Some of these aspects have been overlooked when evaluating TMs, leading to improper conclusions.

Evaluation criteria TMs are often evaluated by how well they correlate with the test accuracy after FT the model on target data. Following [24, 18, 10], we used Pearson correlation. We include additional results with rank correlations (e.g., Spearman) in Supplement Section S7. We argue that considering correlation with vanilla accuracy is flawed in some scenarios. In particular, for TTS, it is wrong to compare target tasks based on accuracy when C is different e.g 5 vs 10 classes. In such a case, task with 10 classes will more likely have lower test accuracy compared to that for task with 5 classes. It is more appropriate to consider the gain in accuracy achieved by the model over it's random baseline. Hence we propose relative accuracy (for balanced classes): $\frac{\text{Accuracy}-1/C}{1/C}$. This measure is more effective in capturing the performance gain achieved by the same model in transferring to two domains with different C. This also highlights the limitation of NCE, LEEP and $\mathcal{N}$LEEP which are sensitive to C and tend to have smaller values with higher C; these measures do not provide useful information about how hard these different tasks when evaluated on the original accuracy scale. Correlations marked with asterisks (*) in Tables 1, 2, 3 are not statistically significant (p-value > 0.05). Hyphen (-) indicates the computation ran out of memory or was really slow.

Target Task Selection We consider Small balanced (S-B) target data, small imbalanced (S-IB) and Large Balance (L-B). See additional details in Supplement Section S6. We compare our proposed $\mathrm{H}_\alpha(f)$ against the original H-Score by [1]. Table 1 demonstrates 80% absolute gains in correlation performance of $\mathrm{H}_\alpha(f)$ over $\mathrm{H}(f)$, making it a leading metric in many cases in small target data regimes. Table 2 shows how various supervised TMs perform in TTS when the number of target classes **varies**. $\mathrm{H}_\alpha(f)$ dominates the performance in both cases, surpassing all supervised TMs.

Table 1: Correlation comparison of our proposed $\mathrm{H}_\alpha(f)$ with supervised TMs against FT accuracy.

Fine-tuning	Target Data	Model	Regime	$\mathrm{H}(f)$	$\mathrm{H}_\alpha(f)$	NCE	LEEP	$\mathcal{N}$LEEP	TransRate	LFC	LogME
Linear	CIFAR-100	VGG19	S-B	-0.138*	0.807	0.656	0.647	0.809	0.564	0.759	**0.848**
			S-IB	0.030*	**0.771**	0.573	0.625	0.703	0.462	0.473	0.748
		ResNet50	S-B	0.034*	**0.865**	0.663	0.684	0.807	0.273	0.773	0.833
			S-IB	-0.103	0.785	0.560	0.569	0.699	0.437	0.518	**0.819**
	CIFAR-10	VGG19	S-B	0.004*	0.671	0.523	0.596	0.612	0.415	0.437	**0.735**
			S-IB	0.091*	0.808	0.746	0.817	0.830	0.287	0.320	**0.886**
		ResNet50	S-B	-0.291	**0.733**	0.427	0.444	0.611	-0.019*	0.565	0.705
			S-IB	0.170*	**0.893**	0.656	0.708	0.752	0.279	0.005*	0.832
Nonlinear	CIFAR-100	VGG19	S-B	0.165*	**0.729**	0.575	0.589	0.674	-0.029*	0.700	-
			S-IB	0.032*	0.487	0.487	0.542	**0.551**	0.480	0.173	-

Table 2: Correlation against *relative* accuracy for L-B case with different C across target tasks.

Model	$\mathrm{H}(f)$	$\mathrm{H}_\alpha(f)$	NCE	n-NCE	LEEP	n-LEEP	$\mathcal{N}$LEEP	n-$\mathcal{N}$LEEP	TransRate	LogME
VGG19	0.876	**0.971**	-0.949	0.655	-0.947	0.661	-0.932	0.945	0.681	0.968
ResNet50	0.950	**0.979**	-0.950	-0.736	-0.950	-0.728	-0.936	-0.626	0.562	0.959

Source Model Selection We select 9 small to large ImageNet models. We evaluate SMS for FT under small sample setting. We sample 50 images per class from all classes available in the original train split of CIFAR-100/CIFAR-10. We designate 10 samples per class for hyperparameter tuning.

Table 3: Correlation of proposed $\mathrm{H}_\alpha(f)$ without/with Random Projection (RP) for FT in SMS.

Regime	Target	$\mathrm{H}(f)$	$\mathrm{H}_\alpha(f)$[No RP]	$\mathrm{H}_\alpha(f)$[RP]	NCE	LEEP	$\mathcal{N}$LEEP	TransRate	LogME
Linear	CIFAR-100	-0.190*	0.024*	**0.859**	0.825	0.839	0.852	-0.204*	0.705
	CIFAR-10	0.276*	0.277*	**0.939**	0.938	0.936	0.938	0.311*	0.923
Nonlinear	CIFAR-100	-0.108*	0.125*	0.879	0.967	0.976	**0.977**	-	-

For SMS, the feature dimensions vary significantly across source models (see Table S3 in Supplement Section S5). In such scenarios, dimensionality reduction is useful for H-score for more meaningful comparison across different models. We apply random projection (RP) on target feature embeddings to project to 128-dimensional space. This provides the gains of proposed $\mathrm{H}_\alpha(f)$ in SMS as well for small samples as given in Table 3, making it again a leading metric in SMS.

Conclusion To summarize, regularized covariance estimation in H-Score (pioneer TM) makes it a leading metric (9/15 cases) against newer measures across various FT scenarios and data settings. With fast implementation, our estimator can be $3-55$ times quicker than state-of-the-art LogME measure. Standardization of NCE, LEEP and $\mathcal{N}$LEEP and correlation against relative accuracy in target selection can provide for more conclusive evaluation about performance of such measures.

Acknowledgments

The research was conducted while Shibal Ibrahim was at Google, and part of the writing was done during his time at MIT. At MIT, Shibal Ibrahim and Rahul Mazumder acknowledge research funding from the Office of Naval Research [Grant ONR-N000141812298].

References

[1] Y. Bao, Y. Li, S. Huang, L. Zhang, L. Zheng, A. Zamir, and L. Guibas. An information-theoretic approach to transferability in task transfer learning. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 2309–2313, 2019.

[2] Y. Chen, A. Wiesel, Y. C. Eldar, and A. O. Hero. Shrinkage algorithms for mmse covariance estimation. *IEEE Transactions on Signal Processing*, 58(10):5016–5029, 2010.

[3] Y. Cui, Y. Song, C. Sun, A. Howard, and S. J. Belongie. Large scale fine-grained categorization and domain-specific transfer learning. *CoRR*, abs/1806.06193, 2018.

[4] A. Deshpande, A. Achille, A. Ravichandran, H. Li, L. Zancato, C. Fowlkes, R. Bhotika, S. Soatto, and P. Perona. A linearized framework and a new benchmark for model selection for fine-tuning, 2021.

[5] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[6] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting (With discussion and a rejoinder by the authors). *The Annals of Statistics*, 28(2):337 – 407, 2000.

[7] I. Guyon. Design of experiments for the nips 2003 variable selection benchmark. 2003.

[8] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[10] L.-K. Huang, Y. Wei, Y. Rong, Q. Yang, and J. Huang. Frustratingly easy transferability estimation. *ArXiv*, abs/2106.09362, 2021.

[11] S. Kornblith, J. Shlens, and Q. V. Le. Do better imagenet models transfer better? In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2656–2666, 2019.

[12] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 25, 01 2012.

[13] O. Ledoit and M. Wolf. A well-conditioned estimator for large-dimensional covariance matrices. *Journal of Multivariate Analysis*, 88(2):365–411, Feb. 2004.

[14] H. Li, P. Chaudhari, H. Yang, M. Lam, A. Ravichandran, R. Bhotika, and S. Soatto. Rethinking the hyperparameters for fine-tuning. *CoRR*, abs/2002.11770, 2020.

[15] Y. Li, X. Jia, R. Sang, Y. Zhu, B. Green, L. Wang, and B. Gong. Ranking neural checkpoints. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2663–2673, June 2021.

[16] D. Mahajan, R. B. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. van der Maaten. Exploring the limits of weakly supervised pretraining. *CoRR*, abs/1805.00932, 2018.

[17] A. W. Max. Inverting modified matrices. In *Memorandum Rept. 42, Statistical Research Group*, page 4. Princeton Univ., 1950.

[18] C. V. Nguyen, T. Hassner, M. Seeger, and C. Archambeau. Leep: A new measure to evaluate transferability of learned representations, 2020.

[19] F. Pedregosa, G. Varoquaux, A. Gramfort, et al. Scikit-learn: Machine learning in python. *J. Mach. Learn. Res.*, 12(null):2825–2830, Nov. 2011.

[20] M. Pourahmadi. *High-dimensional covariance estimation: with high-dimensional data*, volume 882. John Wiley & Sons, 2013.

[21] J. Schäfer and K. Strimmer. A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical Applications in Genetics and Molecular Biology*, 4, 2005.

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

[23] Y. Tan, Y. Li, and S. Huang. OTCE: A transferability metric for cross-domain cross-task representations. *CoRR*, abs/2103.13843, 2021.

[24] A. Tran, C. Nguyen, and T. Hassner. Transferability and hardness of supervised classification tasks. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1395–1405, 2019.

[25] N. X. Vinh, J. Epps, and J. Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *Journal of Machine Learning Research*, 11(95):2837–2854, 2010.

[26] K. You, Y. Liu, J. Wang, and M. Long. Logme: Practical assessment of pre-trained models for transfer learning. In *ICML*, 2021.

Supplementary Material

S1 Acronyms

Table S1: List of acronyms used in the paper.

Terms	**Acronyms**
Transferability Measure	TM
Target Task Selection	TTS
Source Model Selection	SMS
Fine-tuning	FT
Dimensionality Reduction	DR
Principal Component Analysis	PCA
Random Projection	RP
Small-Balanced	S-B
Small-Imbalanced	S-IB
Large-Balanced	L-B

S2 Supervised Transferability metrics

S2.1 Efficient Computation for small target data

For small target data ($C \leq n_t < d$), the naive implementation of $\mathrm{H}_\alpha(f)$ can be very slow. We propose an optimized implementation for our shrinkage-based H-Score that exploits diagonal plus low-rank structure of $\mathbf{\Sigma}^f_\alpha$ for efficient matrix inversion and the low-rank structure of $\mathbf{\Sigma}^z$ for faster matrix-matrix multiplications. We assume $\boldsymbol{F}$ (and correspondingly $\boldsymbol{Z}$) are centered. The optimized computation of $\mathrm{H}_\alpha(f)$ is given by:

$$\mathrm{H}_\alpha(f) = (1-\alpha)/(n_t\alpha\sigma) \cdot \left(\|\boldsymbol{R}\|_F^2 - (1-\alpha) \cdot \mathbf{vec}\,(\boldsymbol{G})^T\, \mathbf{vec}\left(\boldsymbol{W}^{-1}\boldsymbol{G}\right)\right), \tag{S1}$$

where $\boldsymbol{R} = \left[\sqrt{n_1}\bar{\boldsymbol{f}}_{Y=1}, \cdots, \sqrt{n_C}\bar{\boldsymbol{f}}_{Y=C}\right] \in \mathbb{R}^{d\times C}$, $\boldsymbol{G} = \boldsymbol{F}\boldsymbol{R} \in \mathbb{R}^{n_t\times C}$, $\boldsymbol{W} = n_t\alpha\sigma\boldsymbol{I}_n + \boldsymbol{F}\boldsymbol{F}^T \in \mathbb{R}^{n_t\times n_t}$. The derivation is provided in the Supplement Section S2.1.1. We make a timing comparison of our optimized implementation of $\mathrm{H}_\alpha(f)$ against the computational times of the state-of-the-art LogME measure and demonstrate $3 - 55$ times faster computation (see Table S2 in Supplement Section S2.1.2).

S2.1.1 Derivation of optimized implementation for $\mathrm{H}_\alpha(f)$

We derive an optimized computation for our proposed shrinkage-based H-score $\mathrm{H}_\alpha(f)$ for small target data ($C \leq n_t < d$) as follows:

$$\begin{aligned}
\mathrm{H}_\alpha(f) &= \mathrm{tr}\left(\mathbf{\Sigma}_\alpha^{f^{-1}} \cdot (1-\alpha)\mathbf{\Sigma}^z\right) = \frac{(1-\alpha)}{n_t} \cdot \mathrm{tr}\left(\left(\alpha\sigma\boldsymbol{I}_d + \frac{(1-\alpha)}{n_t}\boldsymbol{F}^T\boldsymbol{F}\right)^{-1}\boldsymbol{Z}^T\boldsymbol{Z}\right), \\
&= \frac{(1-\alpha)}{n_t\alpha\sigma} \cdot \mathrm{tr}\left(\left(\boldsymbol{I}_d + \frac{(1-\alpha)}{n_t\alpha\sigma}\boldsymbol{F}^T\boldsymbol{F}\right)^{-1}\boldsymbol{R}\boldsymbol{R}^T\right), \\
&= \frac{(1-\alpha)}{n_t\alpha\sigma} \cdot \mathrm{tr}\left(\left(\boldsymbol{I}_d - (1-\alpha)\cdot\boldsymbol{F}^T\left(n_t\alpha\sigma\boldsymbol{I}_n + \boldsymbol{F}\boldsymbol{F}^T\right)^{-1}\boldsymbol{F}\right)\boldsymbol{R}\boldsymbol{R}^T\right), && \text{(S2)} \\
&= \frac{(1-\alpha)}{n_t\alpha\sigma} \cdot \left(\mathrm{tr}\left(\boldsymbol{R}^T\boldsymbol{R}\right) - (1-\alpha)\cdot\mathrm{tr}\left(\boldsymbol{G}^T\boldsymbol{W}^{-1}\boldsymbol{G}\right)\right), && \text{(S3)} \\
&= \frac{(1-\alpha)}{n_t\alpha\sigma} \cdot \left(\|\boldsymbol{R}\|_F^2 - (1-\alpha)\cdot\mathbf{vec}\,(\boldsymbol{G})^T\,\mathbf{vec}\left(\boldsymbol{W}^{-1}\boldsymbol{G}\right)\right), && \text{(S4)}
\end{aligned}$$

where $\boldsymbol{R} = \left[\sqrt{n_1}\bar{\boldsymbol{f}}_{Y=1}, \cdots, \sqrt{n_C}\bar{\boldsymbol{f}}_{Y=C}\right] \in \mathbb{R}^{d\times C}$, $\boldsymbol{G} = \boldsymbol{F}\boldsymbol{R} \in \mathbb{R}^{n_t\times C}$, $\boldsymbol{W} = n_t\alpha\sigma\boldsymbol{I}_n + \boldsymbol{F}\boldsymbol{F}^T \in \mathbb{R}^{n_t\times n_t}$, (S2) follows by Woodbury matrix identity ($(\boldsymbol{I}+\boldsymbol{U}\boldsymbol{V})^{-1} = \boldsymbol{I} - \boldsymbol{U}(\boldsymbol{I}+\boldsymbol{V}\boldsymbol{U})^{-1}\boldsymbol{V}$) [17] and (S3) and (S4) follow by trace properties.

S2.1.2 Timing comparison between LogME and $\mathrm{H}_\alpha(f)$

We empirically investigate the computational times of $\boldsymbol{H_\alpha(f)}$ when computed via our optimized implementation in (S1). For this exercise, we generate synthetic multi-class classification data using Sklearn [19] multi-class dataset generation function that is adapted from [7]. We investigate different values for number of samples (n_t), feature dimension (d) and number of classes (C). For data generation, we set number of informative features to be 100 with the rest of the features filled with random noise. Table S2 demonstrates a significant computational advantage of $\boldsymbol{H_\alpha(f)}$ over LogME. We observe $3 - 55$ times faster computational times. LogME seems intractable both with respect to memory and time for $d \sim 10^5$ as exposed by the nonlinear settings in Table 1 and 3.

Table S2: Timing comparison of LogME and our shrinkage-based H-score. All times are in ms.

n_t	d	$\|\mathcal{Y}\| = C$	LogME	$\mathrm{H}(f)$	$\mathrm{H}_\alpha(f)$
500	500	50	150	95	**20**
500	1000	50	300	200	**66**
500	5000	50	39100	9680	**1400**
500	10000	50	296000	80000	**5280**
500	1000	10	252	170	**63**
500	1000	100	358	202	**69**
100	1000	50	305	248	**19**
1000	1000	50	333	173	**116**

S3 Normalization of NCE, LEEP and $\mathcal{N}$LEEP

Our proposed definition for normalized NCE in Section 3 ensures the normalized NCE is bounded between $[0, 1]$. For proof, see Section S3.1. n-NCE is in fact equivalent to normalized mutual information and has been extensively used to measure correlation between two different labelings/clustering of samples [25]. Given the similar behavior of LEEP and $\mathcal{N}$LEEP different C and class imbalance as that for NCE (shown in Fig. 2), we suggest similar normalization. However, this normalization does not ensure boundedness of n-LEEP score (and by extension n-$\mathcal{N}$LEEP) in the range $[0, 1]$ as in the case of n-NCE.

S3.1 Normalization of NCE

NCE [24] evaluates conditional entropy between target pseudo labels $Z^{(t)}$ (source model's assigned labels) and actual target labels, as given by:

$$\text{NCE}(Y^{(t)}|Z^{(t)}) = \frac{1}{n_t}\sum_{i=1}^{n_t} \log p_{Y^{(t)}|Z^{(t)}}(y_i|z_i) \tag{S5}$$

The conditional entropy of the target labels conditioned on the dummy labels (source model' labels on target data) is:

$$H(Y|Z) = -\sum_{y\in Y, z\in Z} p_{Y,Z}(y,z) \log p_{Y|Z}(y|z) \tag{S6}$$

It holds that $0 \leq H(Y|Z) \leq H(Y)$ (see appendix S3.1.1). Negative Conditional Entropy (NCE) is given by NCE $= -H(Y|Z)$ and it holds that $0 \leq -\text{NCE} \leq H(Y)$. We normalize the NCE as follows:

$$0 \geq \frac{\text{NCE}}{H(Y)} \geq -1 \tag{S7}$$

$$0 \leq 1 + \frac{\text{NCE}}{H(Y)} \leq 1 \tag{S8}$$

where in (S7) we use the fact that entropy is greater than 0 for any practical classification task.

S3.1.1 Proof of bounds on conditional entropy

$$
\begin{aligned}
H(Y|Z) &= -\sum_{y\in\mathcal{Y},z\in\mathcal{Z}} p_{Y,Z}(y,z)\log p_{Y|Z}(y|z) && \text{(S9)}\\
&= \sum_{z\in\mathcal{Z}} p_Z(z)\left[-\sum_{y\in\mathcal{Y}} p_{Y|Z}(y|z)\log p_{Y|Z}(y|z)\right] && \text{(S10)}\\
&= \sum_{z\in\mathcal{Z}} p_Z(z)H(Y|Z=z) && \text{(S11)}\\
&\geq 0 && \text{(S12)}
\end{aligned}
$$

where (S12) holds because $H(Y|Z=z)\geq 0$ and holds with equality if and only if Y is a deterministic function of Z.

$$
\begin{aligned}
H(Y|Z) &= -\sum_{y\in\mathcal{Y},z\in\mathcal{Z}} p_{Y,Z}(y,z)\log p_{Y|Z}(y|z) && \text{(S13)}\\
&= -\sum_{y\in\mathcal{Y},z\in\mathcal{Z}} p_{Y,Z}(y,z)\log \frac{p_{Y,Z}(y,z)}{p_Z(z)} && \text{(S14)}\\
&= \sum_{y\in\mathcal{Y},z\in\mathcal{Z}} p_Y(y)\frac{p_{Y,Z}(y,z)}{p_Y(y)}\log \frac{p_Z(z)}{p_{Y,Z}(y,z)} && \text{(S15)}\\
&\leq \sum_{y\in\mathcal{Y}} p_Y(y)\ \log\sum_{z\in\mathcal{Z}} \frac{p_{Y,Z}(y,z)}{p_Y(y)}\frac{p_Z(z)}{p_{Y,Z}(y,z)} && \text{(S16)}\\
&= -\sum_{y\in\mathcal{Y}} p_Y(y)\log p_Y(y) && \text{(S17)}\\
&= H(Y) && \text{(S18)}
\end{aligned}
$$

where (S16) holds by Jensen inequality.

S4 Fine-tuning with hyperparameter optimization

Current practices for FT typically involve a selection of values for hyperparameters when retraining the model on target data. Given that the target datasets are typically small in the transfer learning scenarios, the typical strategy is to adopt the default hyperparameters for training large models while using smaller initial learning rate and fewer epochs for FT. It has been believed that adhering to the original hyperparameters for FT with small learning rate prevents catastrophic forgetting of the originally learned knowledge or features. Many studies have used fixed hyperparameters (e.g. learning rate, momentum and weight decay, number of epochs) for FT. However, the choice of hyperparameters is not necessarily optimal for FT on target tasks. Earlier work has reported that the performance is sensitive to the default hyperparameter selection, in particular learning rate, momentum (for stochastic gradient descent), weight decay and number of epochs [16, 11, 14]. The optimal choice of these parameters is not only target data dependent but also sensitive to the domain similarity between the source and target datasets [14]. Therefore, in order to ensure the target task accuracy (against which the correlation of transferability metrics is measured) is optimal, we repeat the FT exercise for 100 trials of hyperparameter settings. We employ Adam for FT experiments and optimize over batch size, learning rate, number of epochs and weight decay (L2 regularization on the classifier head). We select the space of these hyperparameters based on existing literature on FT, e.g. the learning rate is varied in the range $[1e-1, 1e-5]$, the number of epochs between $\{25, 50, 75, 100, 125, 150, 175, 200\}$, the batch size between $\{32, 64, 128\}$ and the weight decay in the range $[1e-6, 1e-2]$.

Table S3: Feature extraction layer in ImageNet models for nonlinear FT. The names are from pre-trained ImageNet models in Tensorflow Keras https://keras.io/api/applications/.

Models	Linear Fine-Tuning		Nonlinear Fine-Tuning	
	Embedding Layer	d	Embedding Layer	d
VGG19	penultimate	4096	block3_pool	$28 \times 28 \times 256 = 200,704$
ResNet50	penultimate	2048	conv2_block3_out	$28 \times 28 \times 256 = 200,704$
ResNet101	penultimate	2048	conv2_block3_out	$28 \times 28 \times 256 = 200,704$
DenseNet121	penultimate	1024	pool2_pool	$28 \times 28 \times 128 = 100,352$
DenseNet201	penultimate	1920	pool2_pool	$28 \times 28 \times 128 = 100,352$
Xception	penultimate	2048	add_6	$14 \times 14 \times 728 = 142,688$
InceptionV3	penultimate	2048	mixed4	$12 \times 12 \times 768 = 110,592$
MobileNet	penultimate	1024	conv_pw_6_relu	$14 \times 14 \times 512 = 100,352$
EfficientNetB0	penultimate	1280	block3a_activation	$28 \times 28 \times 144 = 112,896$

S5 Feature Embedding layers for linear and nonlinear Finetuning

S6 Target Task Selection Experiments setup

This evaluation regime is motivated by task transfer policy learning in robotics/reinforcement learning. Under this regime, transferability measures can be used to greedily optimize a task transfer policy given a collection of tasks. For instance, a robot has to automatically select which new object to pick up. Given that the robot has learned to pick up a few objects before, it would be beneficial for the robot to optimally select the most transferable source/task object pair and improve it's maneuvering ability throughout the process in a highly efficient manner. TMs can also shed light on the relatedness of different tasks in reinforcement learning setups for better understanding.

We currently evaluate target task selection regime on visual classification tasks with both VGG19 [22] and ResNet50 [9] models on subsets of CIFAR-100/CIFAR-10 data under three different dataset regimes following [18]. In all 3 cases outlined below, 20% of the samples from the randomly generated subsets is designated as validation set for hyperparameter tuning to find the model with with optimal validation accuracy. We use all examples in the original test set for evaluating out-of-sample accuracy performance on the target data. *Both* training and validation samples in the subsets are used for computation of transferability metrics and we report the correlation of these measures against the (relative) test accuracies for the randomly generated subsets.

- *Small-Balanced Target Data:* We make a random selection of 5 classes from CIFAR-100/CIFAR-10 and sample 50 samples per class from the original train split, out of which we designate 10 samples per class for validation. We repeat this exercise 50 times (with a different selection of 5 classes), fine-tune the model for each selection (100 hyperparameter tuning trials per selection to find finetuned model with optimal validation accuracy) and evaluate performance of those optimal models in terms of test accuracy. We then evaluate rank correlations of TMs across the 50 experiments with random selection of 5 sub-classes.
- *Small-Imbalanced Target Data:* We make 50 random selections of 2 classes from CIFAR-100/CIFAR-10, sample between $30 - 60$ samples from the first class and sample $5\times$ the number of samples from the second class. This makes for a binary imbalanced classification task. We again measure performance of transferability measures against optimal target test accuracy.
- *Large-Balanced Target Data with different number of classes:* We randomly select 2-100 classes from CIFAR-100 and include all samples from the chosen classes (500 samples per class). This constructs a range of large balanced dataset target task selection cases. We evaluate correlation of TMs with relative target test accuracy across the variable number of target classes.

S7 Spearman Rank Correlation Performance of supervised Transferability Measures

We present rank correlation performance of all supervised TMs across various FT scenarios (target task selection and source model selection), FT strategies (linear and nonlinear) in various data regimes. Table S4 combines setups in Tables 1, 2, and 3 and presents *Spearman* correlation performance of $\mathrm{H}_\alpha(f)$ against supervised TMs. Correlations marked with asterisks (*) are not statistically significant (p-value > 0.05). Hyphen (-) indicates the computation ran out of memory on 128GB RAM and/or was really slow.

Table S4: Spearman correlation comparison of supervised TMs. Larger correlations indicate better identifiability as quantified by TM. We compared our proposed $\mathrm{H}_\alpha(f)$ against original $\mathrm{H}(f)$ and state-of-the-art measures. For L-B regimes in the table we correlate against *relative* accuracy. For other rows, we use vanilla accuracy.

FT scenario	FT strategy	Target Data	Model	Regime	$\mathrm{H}(f)$	$\mathrm{H}_\alpha(f)$	n-NCE	n-LEEP	n-$\mathcal{N}$LEEP	TransRate	LFC	LogME
Target Task Selection	Linear	CIFAR-100	VGG19	S-B	-0.19*	0.77	0.67	0.67	0.81	0.56	0.72	**0.86**
				S-IB	-0.07*	0.71	0.64	0.63	0.72	0.40	0.52	**0.79**
				L-B	0.96	**0.97**	0.50	0.44	0.95	0.91	0.88	0.96
			ResNet50	S-B	0.13*	0.80	0.63	0.65	0.78	0.19*	0.69	**0.82**
				S-IB	-0.10*	0.76	0.57	0.58	0.69	0.43	0.53	**0.81**
				L-B	0.98	**1.00**	-0.89	-0.86	-0.74	0.90	0.92	0.99
		CIFAR-10	VGG19	S-B	0.06*	0.57	0.49	0.49	0.55	0.30	0.30	**0.65**
				S-IB	0.21*	0.72	0.76	0.85	0.85	0.32	0.41	**0.86**
			ResNet50	S-B	-0.31	**0.60**	0.28	0.29	0.51	0.02*	0.46	0.59
				S-IB	0.35	**0.76**	0.64	0.69	0.72	0.25*	-0.10*	**0.76**
	Nonlinear	CIFAR-100	VGG19	S-B	-0.00*	**0.76**	0.61	0.62	0.71	0.02*	0.71	-
				S-IB	0.03*	0.59	0.62	0.62	**0.68**	0.47	0.16*	-
Source Model Selection	Linear	CIFAR-100	-	Small	0.30*	**0.88**	0.83	0.83	0.80	0.35*	0.81	0.83
		CIFAR-10	-	Small	0.07*	0.88	0.93	0.92	0.92	0.07*	0.72	**0.95**
	Nonlinear	CIFAR-100	-	Small	0.052*	**0.96**	0.93	0.93	0.93	0.29	0.88	-

With respect to Spearman correlations in the table above, our shrinkage-based H-score $\mathrm{H}_\alpha(f)$ leads in 7/15 cases and LogME leads in 8/15 cases. In terms of Pearson correlations, $\mathrm{H}_\alpha(f)$ leads in 9/15 cases (Tables 1, 2, 3) and LogME leads in 4/15 cases. Additionally, LogME seems to be intractable with respect to memory and computational speed for nonlinear settings where feature dimension is large ($d \sim 10^5$). Our efficient implementation for $\mathrm{H}_\alpha(f)$ provides a $3 - 55$ times computational advantage over LogME.

S8 Experimental code and type of resources

We use Tensorflow Keras for our implementation. Imagenet checkpoints (Resnet and VGG) come from Keras https://keras.io/api/applications/. For experiments, we use 2 P100 GPUs per model, 15GB RAM per GPU