TRANSFERRED DISCREPANCY: QUANTIFYING THE DIFFERENCE BETWEEN REPRESENTATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Understanding what information neural networks capture is an essential problem in deep learning, and studying whether different models capture similar features is an initial step to achieve this goal. Previous works sought to define metrics over the feature matrices to measure the difference between two models. However, different metrics sometimes lead to contradictory conclusions, and there has been no consensus on which metric is suitable to use in practice. In this work, we propose a novel metric that goes beyond previous approaches. Recall that one of the most practical scenarios of using the learned representations is to apply them to downstream tasks. We argue that we should design the metric based on a similar principle. For that, we introduce the transferred discrepancy (TD), a new metric that defines the difference between two representations based on their downstream-task performance. Through an asymptotic analysis, we show how TD correlates with downstream tasks and the necessity to define metrics in such a task-dependent fashion. In particular, we also show that under specific conditions, the TD metric is closely related to previous metrics. Our experiments show that TD can provide fine-grained information for varied downstream tasks, and for the models trained from different initializations, the learned features are not the same in terms of downstream-task predictions. We find that TD may also be used to evaluate the effectiveness of different training strategies. For example, we demonstrate that the models trained with proper data augmentations that improve the generalization capture more similar features in terms of TD, while those with data augmentations that hurt the generalization will not. This suggests a training strategy that leads to more robust representation also trains models that generalize better.

1 INTRODUCTION

Deep neural networks have achieved great success in many real-world applications, such as image classification (Krizhevsky et al., 2012), speech recognition (Hinton et al., 2012), and natural language processing (Devlin et al., 2018). It is generally agreed in the community that deep learning captures better representations than the previous hand-crafted feature engineering, which contributes to a significant performance improvement (Bhardwaj et al., 2018). Therefore, it is worthwhile to investigate what features[1] a neural network learns in practice, which helps us understand the nature of deep learning. As an initial step towards this challenging task, many people study how different the features learned by different neural networks are.

Recent works (Li et al., 2015; Raghu et al., 2017; Wang et al., 2018; Morcos et al., 2018; Kornblith et al., 2019; Liang et al., 2020) have proposed several metrics measuring the difference between a pair of features learned by two different models. Quite confusingly though, two metrics that both seem reasonable can even draw opposite conclusions on the same issue. For example, Wang et al. (2018) used the maximum match under linear transformations as the evaluation metric and found that the representations of two neural networks trained from different random initializations were utterly different. In sharp contrast, Kornblith et al. (2019) measured with the central kernel alignment (CKA) and concluded that the models capture almost identical features. The contradiction caused by using different metrics is rooted in the disagreement towards the following question (Kornblith et al., 2019):

[1]Without any confusion, we use the terms *feature* and *representation* interchangeably.

What does it mean for two representations to be different?

Existing metrics measure the difference directly by feature values, which we believe may not be the best way. In representation learning, the quality of features learned by a neural network is hardly evaluated on their values, but rather on the performances they achieve when applied to downstream tasks (Devlin et al., 2018; Kolesnikov et al., 2019; Chen et al., 2020). Given a trained feature extractor, we train an additional output head on the features of the data for each downstream task and consider one feature (or feature extractor) better than the other if it achieves higher downstream-task performance. Such a way of evaluation has been widely adopted and proven useful in computer vision (Chen et al., 2020) and natural language processing (Devlin et al., 2018).

Based on the above discussion, we argue that a reasonable representation difference metric should also be designed by incorporating the downstream tasks. To achieve this, we propose a new metric which takes the downstream task as an input, and refer to it as the transferred discrepancy (TD). Given two feature extractors and a set of downstream tasks, we train an output head on top of each feature extractor per task, and define the TD metric as the difference between the predictions over the downstream task data. The more different the two feature extractors are, the more likely they may lead to different predictions on the same downstream task data, and the higher TD value they will achieve. We analyze the theoretical properties of the TD metric under the linear probing setting, where the downstream tasks are limited to linear regression. Under this setting, we prove that the TD metric is invariant under reasonable transformations and analyze its asymptotic limits. Furthermore, we show that by properly selecting downstream tasks, the TD metric is closely related to existing metrics such as the maximum match, canonical correlation analysis (CCA), and CKA.

Apart from theoretical analysis, we conduct extensive experiments and demonstrate that the TD metric can reveal faithful information for various downstream tasks in practice. Importantly, we observe that the features learned by models trained from different initializations are not quite the same according to their performances on downstream tasks. Furthermore, we study a quantity called *TD robustness*, which is defined as the difference between the predictions of two models trained from different initializations on the same downstream task. We investigate how TD robustness varies upon changes in factors such as data augmentation methods and training strategies. Remarkably, we find that TD robustness is closely connected with the quality of the representation. For instance, data augmentation methods that are proven to improve the quality of the representation (Shorten & Khoshgoftaar, 2019; Chen et al., 2020) are also observed to increase TD robustness while transformations that harm the quality lower it. Such a relationship is also observed for other factors, providing a new perspective on how various factors in deep learning affect the representation the model learns.

2 Related Work

Many previous works try to understand whether two neural networks with drastically different parameters but similar high performances learn similar representations (Li et al., 2015; Raghu et al., 2017; Wang et al., 2018; Morcos et al., 2018; Kornblith et al., 2019; Liang et al., 2020). Using a neural network's hidden states as features, these works obtain a feature matrix over a set of samples for each model, and evaluate the correlation between the two feature matrices with some metrics taken from matrix theory. For example, Wang et al. (2018) measured the size of the intersection between two matrices' row spaces. Raghu et al. (2017) applied CCA to large singular vectors of the two matrices, which is further improved by Morcos et al. (2018). Kornblith et al. (2019) computed the norms of the generalized cross-correlation operator between two matrices. However, all of these works cast the quantification of the difference between two models as a matrix correlation problem, while the practical usage of the feature is largely ignored.

The proposed TD is motivated by representation learning. In representation learning, the quality of the learned representations is often evaluated based on their performance on downstream tasks. In computer vision, the representation of images can be pre-trained using SimCLR (Chen et al., 2020), and such learned representations can help the model training on twelve downstream classification tasks. Another widely known milestone in natural language processing is the BERT model (Devlin et al., 2018). Thus, we think it is more reasonable to define whether two representations are similar based on their performances on downstream tasks as well.

3 TRANSFERRED DISCREPANCY (TD)

Let $\mathcal{X}$ be the input space, and $\mathbf{x}_1, \cdots, \mathbf{x}_n$ be n i.i.d. samples from an underlying distribution p_{data} defined on $\mathcal{X}$. Let $\Phi(\cdot)$ and $\Phi'(\cdot)$ be the two feature extractors, typically neural networks. Let $\mathbf{z}_i = \Phi(\mathbf{x}_i) \in \mathbb{R}^p$ and $\mathbf{z}'_i = \Phi'(\mathbf{x}_i) \in \mathbb{R}^{p'}$ be the feature of $\mathbf{x}_i$ extracted by $\Phi(\cdot)$ and $\Phi'(\cdot)$ respectively, $i = 1, \cdots, n$. Note that the dimensions of the features, p and p', are not necessarily equal, and we assume that $p \leq p'$ without loss of generality. Denote $Z = (\mathbf{z}_1, \cdots, \mathbf{z}_n) \in \mathbb{R}^{p \times n}$ and $Z' = (\mathbf{z}'_1, \cdots, \mathbf{z}'_n) \in \mathbb{R}^{p' \times n}$ as the feature matrices. Our goal is to quantify the difference between the two feature extractors $\Phi(\cdot)$ and $\Phi'(\cdot)$.

As previously discussed, in representation learning, the feature extractor is designed and trained to improve the model's performance on downstream tasks. Thus, we argue that the difference between $\Phi(\cdot)$ and $\Phi'(\cdot)$ should be evaluated based on their corresponding performances on downstream tasks. In practice, a downstream task can be specified by a label vector $Y = (y_1, \cdots, y_n)$, where y_i is the label of $\mathbf{x}_i$. y_i can be a categorical value for classification tasks or a numeric value for regression tasks. Let $h_W(\mathbf{z}_i)$ and $h'_{W'}(\mathbf{z}'_i)$ be the output heads built upon the two feature extractors, W and W' are the learnable parameters. Given a loss function $\ell(\hat{y}, y)$, the parameters W and W' are obtained by minimizing the empirical risks:

$$\begin{cases} \hat{W} = \arg\min_{W} \dfrac{1}{n} \displaystyle\sum_{i=1}^{n} \ell(h_W(\mathbf{z}_i), y_i), \\ \hat{W}' = \arg\min_{W'} \dfrac{1}{n} \displaystyle\sum_{i=1}^{n} \ell(h'_{W'}(\mathbf{z}'_i), y_i). \end{cases} \tag{1}$$

After $\hat{W}$ and $\hat{W}'$ are obtained, the difference between Z and Z' is measured by the divergence between $h_W(Z)$ and $h'_{W'}(Z')$. Let $d(u, v)$ be a symmetric divergence function. The difference between Z and Z' on the task with label Y is defined as[2]:

$$\text{TD}(Z, Z'; Y) = \frac{1}{n} \sum_{i=1}^{n} d(h_{\hat{W}}(\mathbf{z}_i), h'_{\hat{W}'}(\mathbf{z}'_i)). \tag{2}$$

As we quantify the difference between representations from a feature-transferring perspective, we name it *transferred discrepancy*. The TD metric defined on one downstream task may be insufficient to measure the difference, so we further define the transferred discrepancy on a family of tasks. Suppose $\mathcal{S}$ is a set of label vectors with each vector Y as a task. The TD measured on set $\mathcal{S}$ is:

$$\text{TD}(Z, Z'; \mathcal{S}) = \max_{Y \in \mathcal{S}} \text{TD}(Z, Z'; Y). \tag{3}$$

The TD metric can reveal the difference between Z and Z'. Intuitively, when Z and Z' are very similar, $\hat{W}$ and $\hat{W}'$ will be similar and $h_{\hat{W}}(\mathbf{z})$ and $h_{\hat{W}'}(\mathbf{z})$ will not differ much from each other on every example in the downstream task. Thus, Z and Z' will have a small TD value. On the contrary, when Z and Z' are different features, the prediction $h_{\hat{W}}(\mathbf{z})$ and $h_{\hat{W}'}(\mathbf{z})$ are more likely different and the value of TD will be large. Although the TD metric is designed from a practitioner's perspective and is subject to the downstream tasks, we will show that it has nice theoretical properties and close relation with previous metrics in the next section.

4 THEORETICAL PROPERTIES OF THE TD METRIC

In this section, we theoretically analyze the TD metric under the *linear probing* setting. We show that TD is invariant to orthogonal transformation and isotropic scaling (4.1), and further investigate its asymptotic behavior as n approaches infinity (4.2). Finally, we demonstrate that under certain conditions, the TD metric and three previous metrics, maximum match, CCA, and CKA, depend on the same statistics (4.3).

[2]For ease of understanding, Eqn (2) defines the metric over the training data. Generally, one can also quantify this distance on unseen data, e.g., the test set in the downstream task, to take the generalization into account. In our experiments, we use TD metrics over the test data and find the value is similar to that over the training data.

4.1 TRANSFORMATION INVARIANCE

The linear probing setting is widely studied in literature (Alain & Bengio, 2016; Oord et al., 2018; Hjelm et al., 2018) and applied in practice (Chen et al., 2020; Anand et al., 2019). In this setting, a linear model is trained on top of a feature extractor, and its performance is used as a proxy for the quality of the features. Define $h_W(\mathbf{z}) = W\mathbf{z} + b$ and $h'_{W'}(\mathbf{z}') = W'\mathbf{z}' + b'$, where $W \in \mathbb{R}^p, W' \in \mathbb{R}^{p'}$ and $b, b' \in \mathbb{R}$. Let $\ell(\hat{y}, y) = (\hat{y} - y)^2$ be the square loss and $d(u, v) = (u - v)^2$ be the squared distance. Following Kornblith et al. (2019), we assume that both Z and $Z^\top$ have been preprocessed to center the rows and Y is centered[3] . Besides, we assume that we have enough data such that $n > \max\{p, p'\}$, and both empirical covariance matrices $ZZ^\top$ and $Z'Z'^\top$ are invertible[4] . Under these assumptions, the optimization problem (1) can be rewritten as

$$\begin{cases} \hat{W}, b = \underset{W,b}{\arg\min} \dfrac{1}{n} \displaystyle\sum_{i=1}^{n} (W\mathbf{z}_i + b - y_i)^2, \\ \hat{W}', b' = \underset{W',b'}{\arg\min} \dfrac{1}{n} \displaystyle\sum_{i=1}^{n} (W'\mathbf{z}'_i + b' - y_i)^2. \end{cases} \tag{4}$$

As Z, Z' are all row-centered, problem (4) has a simple closed-form solution: $\hat{b} = \hat{b}' = 0$ and $\hat{W} = YZ^\top(ZZ^\top)^{-1}, \quad \hat{W}' = YZ'^\top(Z'Z'^\top)^{-1}$. Plugging this solution into Eqn (2) yields

$$\mathrm{TD}(Z, Z'; Y) = \frac{1}{n} \left\| Y[Z^\top(ZZ^\top)^{-1}Z - Z'^\top(Z'Z'^\top)^{-1}Z'] \right\|_2^2. \tag{5}$$

A reasonable metric should have some basic properties. Kornblith et al. (2019) proposed that a similarity metric of features should have two invariance properties, the invariance to isotropic scaling, and the invariance to orthogonal transformation. It is straightforward to see that (1). For any $\beta, \beta' \in \mathbb{R}^+$, $\mathrm{TD}(Z, Z'; Y) = \mathrm{TD}(\beta Z, \beta' Z'; Y)$. (2). For any unitary matrices $Q \in \mathbb{R}^{p \times p}$ and $Q' \in \mathbb{R}^{p' \times p'}$, $\mathrm{TD}(Z, Z'; Y) = \mathrm{TD}(QZ, Q'Z'; Y)$. Thus, Eqn (5) derives that TD is invariant to isotropic scalings and orthogonal transformations. In fact, such transformation invariance holds not only for the square loss function but also for many other choices of ℓ as long as ℓ is strongly convex. We leave the discussion in Appendix A.4.

4.2 CONVERGENCE ANALYSIS

Since $\mathbf{x}_1, \cdots, \mathbf{x}_n$ are independently sampled from p_{data}, it is natural to require that as n goes to infinity, $\mathrm{TD}(Z, Z'; \mathcal{S})$ converges to a value which represents the difference of two feature extractors over the data distribution on downstream task set $\mathcal{S}$. We denote this value by $\mathrm{TD}(\Phi(p_{\mathrm{data}}), \Phi'(p_{\mathrm{data}}); \mathcal{S})$. Denote the joint feature distribution for Z and Z' as $(P, P') := (\Phi(p_{\mathrm{data}}), \Phi'(p_{\mathrm{data}}))$ where P and P' are corresponding marginal distributions. Since the rows of Z and Z' are centered, we have $\mathbb{E}_{\mathbf{z} \sim P}[\mathbf{z}] = 0$ and $\mathbb{E}_{\mathbf{z}' \sim P'}[\mathbf{z}'] = 0$. Let the covariance matrix of the joint distribution (P, P') be $\begin{bmatrix} A & B \\ B^\top & C \end{bmatrix}$. We also assume that the covariance matrices A and C are invertible. Since we seek to define TD on downstream tasks, the following *representative task set* contains all possible tasks associated with Z and Z'. All proofs can be found in the Appendix A.

Representative Task Set Define $\mathcal{S}^* = \{Y = \boldsymbol{\alpha} A^{-\frac{1}{2}} Z + \boldsymbol{\alpha}' C^{-\frac{1}{2}} Z' : \boldsymbol{\alpha} \in \mathbb{R}^p, \boldsymbol{\alpha} \in \mathbb{R}^{p'}, \|\boldsymbol{\alpha}\|_2 \leq 1, \|\boldsymbol{\alpha}'\|_2 \leq 1\}$. This set contains all tasks linearly realizable by Z and Z' thus covers a wide range of tasks. $A^{-\frac{1}{2}}$ and $C^{-\frac{1}{2}}$ are used to normalize Z and Z' so that they have the same level of contribution to $\mathcal{S}^*$. For any downstream task Y in $\mathcal{S}^*$, the following theorem rigorously states how the TD metric depends on Y and the representations Z, Z'.

Theorem 1. *Suppose $A, B,$ and C are defined as above. Denote $D = A^{-\frac{1}{2}} B C^{-\frac{1}{2}}$. For downstream task $Y = \boldsymbol{\alpha} A^{-\frac{1}{2}} Z + \boldsymbol{\alpha}' C^{-\frac{1}{2}} Z'$, we have*

$$\mathrm{TD}(Z, Z'; Y) \xrightarrow{a.s.} \boldsymbol{\alpha}(I_p - DD^\top)\boldsymbol{\alpha}^\top + \boldsymbol{\alpha}'(I_{p'} - D^\top D)\boldsymbol{\alpha}'^\top + 2\boldsymbol{\alpha}(D - DD^\top D)\boldsymbol{\alpha}'^\top, \tag{6}$$

[3]Note that this assumption only simplifies the proof. Without such a preprocessing, we can get similar results.

[4]In practice we have $n > \max\{p, p'\}$ in most if not all cases. Even if $ZZ^\top$ is not invertible, adding a tiny noise to Z makes $ZZ^\top$ invertible.

Theorem 1 indicates that in this setting, the transferred discrepancy primarily depends on the task (i.e., α and α') and the matrix D which leverages the covariance matrix B between Z and Z'. Here we show two special cases where Eqn 6 has a simpler form for better understandings:

- TD for linearly correlated features: If $p = p'$ and there exists a unitary matrix $Q \in \mathbb{R}^{p\times p}$ such that (z, z') drawn from (P, P') satisfies $z' = Qz$, i.e., Z and Z' are linearly correlated. We have $D = Q^\top$, so $\text{TD}(Z, Z'; \boldsymbol{\alpha}A^{-\frac{1}{2}}Z + \boldsymbol{\alpha}'C^{-\frac{1}{2}}Z') \xrightarrow{a.s.} 0$.
- TD for independent features: If P and P' are independent, then $B = 0$ and $D = 0$. Consequently, $\text{TD}(Z, Z'; \boldsymbol{\alpha}A^{-\frac{1}{2}}Z + \boldsymbol{\alpha}'C^{-\frac{1}{2}}Z') \xrightarrow{a.s.} 2$.

Next, we provide the convergence analysis of the TD metric on a set of tasks. We first study the asymptotic limit for TD on the *representative task set*, and then extend the analysis to the *restricted task sets*. Both results show that TD highly relates to the singular value distribution of matrix D.

Theorem 2. *Under the notations in Theorem 1, denote $\sigma_1 \geq \cdots \geq \sigma_p \geq 0$ be the singular values of D. If $p = p'$, or $p < p'$ and $(1 - \sigma_p)(1 + \sigma_p)^2 \geq 1$, we have a closed-form limit of TD on the representative task set $\mathcal{S}^*$:*

$$\text{TD}(Z, Z'; \mathcal{S}^*) \xrightarrow{a.s.} \max_{j=1,\cdots,p} 2(1 - \sigma_j)(1 + \sigma_j)^2. \tag{7}$$

The result shows that the TD metric on the representative task set is closely related to the singular values of D. For example, if σ_p is small, there exists a task $Y \in \mathcal{S}^*$ that is close to the row space of one feature matrix but nearly orthogonal to the row space of the other. Two representations will have drastically different performance on Y, and $\text{TD}(Z, Z'; \mathcal{S}^*)$ is large. If σ_p is close to 1, so do all the singular values, and any task $Y \in \mathcal{S}^*$ will be close to both the row spaces of Z and Z'. Thus, the performance is similar, and $\text{TD}(Z, Z'; \mathcal{S}^*)$ is small. When there are both large and small singular values, the difference between the performance varies a lot on diverse tasks, and evaluating the difference on specific tasks may be helpful. The following corollary gives a convergence guarantee for a more practical case for subsets of the representation task set:

Restricted Task Set We study smaller task sets called restricted task sets that allow us to jointly consider $\sigma_1, \cdots, \sigma_p$ instead of σ_p alone. Particularly, we construct a cascade of sets $\mathcal{S}_1 \subset \cdots \subset \mathcal{S}_p \subset \mathcal{S}$, and the two representations are considered more similar if they have similar performance on a larger task set. Define $\mathcal{S}_r = \{Y = \boldsymbol{\alpha}A^{-\frac{1}{2}}Z + \boldsymbol{\alpha}'C^{-\frac{1}{2}}Z' : \|\boldsymbol{\alpha}\|_2 \leq 1, \|\boldsymbol{\alpha}'\|_2 \leq 1, \boldsymbol{\alpha}\mathbf{u}_j = 0 \text{ and } \boldsymbol{\alpha}'\mathbf{v}_j = 0 \text{ if } j > r\}$, $r = 1, \cdots, p$, where $\mathbf{u}_j$ and $\mathbf{v}_j$ are the j-th singular vectors of U, V, in $D = U\Sigma V^\top$ respectively. When $p = p'$, we have $\mathcal{S}_p = \mathcal{S}^*$. The following corollary gives the asymptotic limits of the TD metric on the restricted task sets:

Corollary 1. *Under the notations in Corollary 2 and for the restricted task set $\mathcal{S}_r$ defined as above, we have*

$$\text{TD}(Z, Z'; \mathcal{S}_r) \xrightarrow{a.s.} \max_{j=1,\cdots,r} 2(1 - \sigma_j)(1 + \sigma_j)^2. \tag{8}$$

We conduct experiments to empirically analyze the singular values of D for some Z and Z' trained on the Cifar-10 dataset (Krizhevsky & Hinton, 2009). Results are shown in A.5.

4.3 Connection With Previous Metrics

In this section, we show that our TD metric has a close connection with previous metrics by showing the matrix D defined in section 4.2 is also the key to derive previous metrics. Previous works lead to contradictive conclusions by using D in different ways. We only discuss the relationship with CCA in the main body and leave the discussions on maximum match and CKA in Appendix B.

Canonical Correlation Analysis (CCA) CCA measures the relationship between Z and Z' by finding two bases of their row spaces such that when projected onto these bases, the correlation between the two matrices is maximized. Formally, for $1 \leq j \leq p$, define the maximum correlation coefficient ρ_i by the following optimization problem:

$$\begin{aligned} \rho_j = \max_{\mathbf{w}_j, \mathbf{w}'_j} \quad & \frac{\text{cov}(\mathbf{w}_j^\top Z, \mathbf{w}_j'^\top Z')}{\sqrt{\text{var}(\mathbf{w}_j^\top Z)\text{var}(\mathbf{w}_j'^\top Z')}}, \\ \text{subject to} \quad & \text{cov}(\mathbf{w}_k^\top Z, \mathbf{w}_j^\top Z) = 0,\ \text{cov}(\mathbf{w}_k'^\top Z', \mathbf{w}_j'^\top Z') = 0, \forall k < j. \end{aligned} \tag{9}$$

Then the summary statistics of CCA is defined as $R^2_{CCA} := \frac{\sum_{j=1}^{p} \rho_j^2}{p}$. Equivalently, let $Q = Z^\top (ZZ^\top)^{-\frac{1}{2}}$ and $Q' = Z'^\top (Z'Z'^\top)^{-\frac{1}{2}}$, then $R^2_{CCA} = \frac{\|Q'^\top Q\|_F^2}{p}$. Since data are independently sampled from the distribution, we have $\mathbb{E}_{Z,Z'}[Q'^\top Q] = D^\top$. By $\|D\|_F^2 = \sum_{j=1}^{p} \sigma_j^2$, we have:

$$\mathbb{E}_{Z,Z'}[R^2_{CCA}] = \frac{\sum_{j=1}^{p} \sigma_j^2}{p}. \tag{10}$$

Moreover, take $\hat{A} = \frac{ZZ^\top}{n}$ and $\hat{C} = \frac{Z'Z'^\top}{n}$ as the empirically estimators of A and C. CCA is directly related to TD on an empirical representative task set $\hat{\mathcal{S}}^*$ with $Y = \boldsymbol{\alpha}\hat{A}^{-\frac{1}{2}} Z + \boldsymbol{\alpha}'\hat{C}^{-\frac{1}{2}} Z'$ as below:

Theorem 3. *Let α and α' be uniformly distributed on the unit ball and $\hat{A}, \hat{C}$ defined above. If $p = p'$,*

$$\mathbb{E}_{\alpha,\alpha'}[\text{TD}(Z, Z'; Y = \boldsymbol{\alpha}\hat{A}^{-\frac{1}{2}} Z + \boldsymbol{\alpha}'\hat{C}^{-\frac{1}{2}} Z')] = \frac{2p}{p+2}(1 - R^2_{CCA}). \tag{11}$$

This result reveals the connection between CCA and our proposed metric. CCA is equivalent to averaging TD on all linearly realizable tasks in the linear probing setting, which makes CCA downstream-task agnostic. However, when evaluated on a subset of tasks, or several tasks of interest, CCA cannot provide fine-grained information, while the TD metric is always faithful to the downstream tasks. In the next section, we conduct extensive experiments to study the performance of the TD metric in practical applications.

5 Experiments

In this section, we empirically compare one TD induced metric, TD_{cls}, with CCA and CKA for classification tasks. We directly use the CKA with linear kernel since it performs similarly with CKA using other kernels (Kornblith et al., 2019). For convenience, we define the three metrics as below:

- $D_{CCA}(Z, Z') = 1 - R^2_{CCA}$
- $D_{CKA}(Z, Z') = 1 - S_{CKA}(Z^\top Z, Z'^\top Z') = 1 - \frac{\|ZZ'^\top\|_F^2}{\|ZZ^\top\|_F \|Z'Z'^\top\|_F}$
- $\text{TD}_{cls}(Z, Z'; Y) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{\arg\max h(\mathbf{z}_i) \neq \arg\max h'(\mathbf{z}'_i)}$

where $\arg\max h(\mathbf{z})$ denotes the output prediction for $\mathbf{z}$. It is easy to see that TD_{cls} measures the fraction of different predictions between two representations on the downstream task Y. We also try some other distance metrics and obtain similar conclusions. The results are in Appendix D. We will firstly show that the TD metric is valid with a sanity check, and then investigate whether models that are trained using different initializations learn similar features. Lastly, we will define TD robustness and study the TD robustness of different training strategies. We will also establish a connection between TD robustness and the quality of the representation.

5.1 A Sanity Check

In this section, we demonstrate the validity of the TD metric by a simple sanity check: given three feature extractors Φ_1, Φ_2, Φ_3, such that Φ_1 and Φ_2 capture similar features than Φ_1 and Φ_3 by some prior knowledge, we check whether the value of the TD metric is consistent with the prior.

To build this sanity check, we train Φ_1 and Φ_2 using similar datasets, but train Φ_3 using a completely different dataset. Specifically, we design Cifar-2 and Cifar-5 by grouping Cifar-10's labels into 2 and 5 groups respectively. Therefore, Cifar-2, Cifar-5, and Cifar-10 datasets share the same input data but use slightly different labels. These three datasets serve as candidates for training Φ_1 and Φ_2. Φ_3 is trained using the SVHN dataset. All feature extractors use ResNet32 architecture, and the classification head is removed after training. For simplification, we use the Cifar-10 task as the downstream task. We apply Φ_1, Φ_2, Φ_3 to the training data, and train a classification head for Cifar-10 using logistic regression. After training, we apply the three classifiers to the test set and calculate TD_{cls}. For a fair comparison, both D_{CCA} and D_{CKA} are also calculated on the test samples. Each experiment is conducted for ten times. All details can be found in Appendix C.

We compare the difference between pairs of models using different metrics and list all results in Table 1. It can be seen that both D_{CCA} and D_{CKA} have smaller values in the first three rows compared with the other rows, which indicates that the features learned from Cifar-2/5/10 are more similar, but are very different from the features learned from SVHN. TD_{cls} also has a similar trend: the values of TD_{cls} in the first three rows are smaller than 0.5 while the values in the other rows are all larger than 0.8. Thus, all these three metrics are reasonable and consistent with our prior knowledge.

Table 1: The sanity check result.

Model 1	Model 2	Downstream	D_{CCA}	D_{CKA}	TD_{cls}
Cifar-10	Cifar-5	Cifar-10	0.7642	0.3024	0.2158
Cifar-10	Cifar-2		0.8323	0.5330	0.4958
Cifar-5	Cifar-2		0.8030	0.4530	0.4944
Cifar-10	SVHN	Cifar-10	0.9218	0.9713	0.8001
Cifar-5	SVHN		0.9793	0.9710	0.8049
Cifar-2	SVHN		0.9163	0.9782	0.8163

5.2 Does Initialization Affect Learned Features?

In this section, we address the problem widely studied by previous works (Wang et al., 2018; Kornblith et al., 2019; Morcos et al., 2018): whether models trained from different random initializations learn different features. We train two ResNet32 networks on the training set of Cifar-5/2 using the same setting (dataset, algorithm, hyperparameters, etc.) but with different initializations. We use three downstream tasks, Cifar-10/5/2. D_{CCA} D_{CKA} are directly computed over the test set. For TD, we first train the output heads of the models on the training data and then compute TD_{cls} on the test data. In Table 2, we repeat the experiment ten times and report the average values for each row.

Table 2: Similarity of the models learned with different initializations.

Model 1	Model 2	Downstream	D_{CCA}	D_{CKA}	TD_{cls}
Cifar-5	Cifar-5	Cifar-10	0.6961	0.0835	0.2139
		Cifar-5			0.0442
		Cifar-2			0.0109
Cifar-2	Cifar-2	Cifar-10	0.6931	0.0402	0.3745
		Cifar-5			0.2631
		Cifar-2			0.0164

It can be seen that TD_{cls} provides more information than CCA and CKA. Since CCA and CKA are downstream-task-agnostic, they can only present one scalar value for each pair of models. However, TD_{cls} outputs different values for different downstream tasks. For example, the two models trained on Cifar-5 have very consistent predictions on Cifar-2 and Cifar-5, but their predictions are much more different on Cifar-10: they disagree on 21.39% of the Cifar-10 test samples. It can be also observed that the two models trained on Cifar-2 behave similarly on the Cifar-2 downstream task but quite differently on Cifar-5 and Cifar-10.

We find that the two feature extractors trained from different initializations do not learn the same features. First of all, we show that our experimental setting is reasonable since the downstream tasks have a close connection with the upstream tasks. To verify this, we find that the model trained from the Cifar-5 task can reach 80% accuracy on the Cifar-10 downstream task. This result indicates that for those models, the features can be transferred from one task to the other. However, in this reasonable setting, the disagreements between representations are high, i.e., 21.39% for two Cifar-5 models and 37.45% for two Cifar-2 models. Such difference indicates that the features of these models are not the same when evaluated on Cifar-10. Our finding is consistent with the results in Morcos et al. (2018) that models trained with different initializations can capture different features, but differs from the results in Kornblith et al. (2019).

5.3 TD Robustness and Its Application to Training Strategy Evaluation

Deep learning practitioners design different training strategies in the hope of enhancing the quality of features captured by the model. In this section, we build a connection between the quality of the

learned representations and the *TD robustness* of the implemented training strategy. A training strategy is said to be TD-robust if it leads to models with consistent predictions on the same downstream task when trained from different initializations. We study the effect of different factors on TD robustness. Specifically, we focus on three factors: data augmentation, learning rate schedules and adversarial training. Experimental details are left in the Appendix C.

Data Augmentation We study three augmentation methods: random flipping, random cropping, and adding Gaussian noise. Random flipping and cropping are widely used in practice and believed helpful to generalization (Shorten & Khoshgoftaar, 2019). Gaussian additive noise is used for learning smooth classifiers (Cohen et al., 2019).

Table 3: The effect of difference factors on the TD robustness.

Factors	Configuration	Upstream	Downstream	TD_{cls}
Data Augmentation	Without augmentation	Cifar-5	Cifar-10	0.2812
	Random flipping			0.2739
	Random flipping + croping			**0.2150**
	Random flipping + croping + Gaussian additive noise			0.2542
LR Schedule	Small LR + without decay	Cifar-5	Cifar-10	0.2521
	Large LR + without decay			0.2770
	Large LR + with decay			**0.2313**
Std / Adv training	Standard Training	Cifar-5	Cifar-10	0.2150
	Adversarial Training			**0.1823**

We report the results in Table 3. It can be seen that using cropping and flipping leads to more similar representations, and cropping has a significant impact on TD_{cls}. On the other hand, adding Gaussian noise has the opposite effect and makes representations less similar. This result aligns well with previous works that show random flipping and cropping can improve the quality of the representation, but Gaussian noise hurts the quality (Chen et al., 2020).

Learning Rate Schedule Practically, a learning rate decay scheduler is usually used for the sake of finding the local minima. We here show that models using a learning-rate decay schedule also learn more similar features across different initializations. The result in Table 3 shows that the "Large LR + with decay" schedule leads to more similar representations than the other two schedules. This observation coincides with the fact that learning rate decay helps improve the quality of the representation in practice, and validates the theoretical finding in Li et al. (2019).

Adversarial Training Recently, Tsipras et al. (2019) and Ilyas et al. (2019) showed that adversarial training helps neural networks learn features that align better with human perceptions and Santurkar et al. (2019) used robust network to boost performance in synthesis tasks. We find that adversarial training is more TD-robust than standard training, as shown in Table 3. Our result suggests that adversarial training may capture features with better qualities even though it lowers the accuracy.

TD Robustness Versus the Quality of the Representation Our experimental results demonstrate a strong connection between TD robustness and the quality of the representation: a training strategy leading to better qualities produces better TD robustness. This connection makes TD a useful tool for investigating the effect of different factors on representations and designing new training strategies.

6 Conclusion and Future Work

In this work, we propose the Transferred Discrepancy, a metric that quantifies the difference between two representations using the difference between their performance on the same set of downstream tasks. Our theoretical analysis founds a solid basis for TD and reveals the connection between TD and previous metrics. We also conduct extensive experiments to study how different training factors affect the difference between two representations trained from different initializations. We believe that using downstream-task performances to study the difference between learned features is promising, and in the future we will investigate more factors such as normalization and dropout. At the same time, we would also like to extend the current setting to study deep transfer learning.

REFERENCES

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

Ankesh Anand, Evan Racah, Sherjil Ozair, Yoshua Bengio, Marc-Alexandre Côté, and R Devon Hjelm. Unsupervised state representation learning in atari. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8769–8782. Curran Associates, Inc., 2019.

Anurag Bhardwaj, Wei Di, and Jianing Wei. *Deep Learning Essentials: Your hands-on guide to the fundamentals of deep learning and neural network modeling*. Packt Publishing Ltd, 2018.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *ICML 2019 : Thirty-sixth International Conference on Machine Learning*, pp. 1310–1320, 2019.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Brandon Tran, and Aleksander Madry. Learning perceptually-aligned representations via adversarial robustness. *arXiv preprint arXiv:1906.00945*, 2019.

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

Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, 29(6):82–97, 2012.

R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*, pp. 125–136, 2019.

Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 1920–1929, 2019.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3519–3529, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Handbook of Systemic Autoimmune Diseases*, 1(4), 2009.

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

Yixuan Li, Jason Yosinski, Jeff Clune, Hod Lipson, and John Hopcroft. Convergent Learning: Do different neural networks learn the same representations? *arXiv e-prints*, art. arXiv:1511.07543, November 2015.

Yuanzhi Li, Colin Wei, and Tengyu Ma. Towards explaining the regularization effect of initial large learning rate in training neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 11669–11680. Curran Associates, Inc., 2019.

Ruofan Liang, Tianlin Li, Longfei Li, Jing Wang, and Quanshi Zhang. Knowledge consistency between neural networks and beyond. In *International Conference on Learning Representations*, 2020.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Ari Morcos, Maithra Raghu, and Samy Bengio. Insights on representational similarity in neural networks with canonical correlation. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 31*, pp. 5727–5736. Curran Associates, Inc., 2018.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 6076–6085. Curran Associates, Inc., 2017.

Shibani Santurkar, Andrew Ilyas, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Image synthesis with a single (robust) classifier. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 1262–1273. Curran Associates, Inc., 2019.

Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):60, 2019.

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

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*, 2019.

Liwei Wang, Lunjia Hu, Jiayuan Gu, Zhiqiang Hu, Yue Wu, Kun He, and John Hopcroft. Towards understanding learning representations: To what extent do different neural networks learn the same representation. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 31*, pp. 9584–9593. Curran Associates, Inc., 2018.

A ASYMPTOTIC LIMITS OF TRANSFERRED DISCREPANCY

A.1 PROOF OF THEOREM 1

Suppose the covariance matrix of the joint distribution is $(\Phi(p_{data}), \Phi'(p_{data}))$. By the law of large numbers, as $n \to \infty$, $\frac{ZZ^\top}{n} \to A$, $\frac{ZZ'^\top}{n} \to B$ and $\frac{Z'Z'^\top}{n} \to C$ almost surely. Denote $D = A^{-\frac{1}{2}}BC^{-\frac{1}{2}}$. For downstream task $Y = \boldsymbol{\alpha}A^{-\frac{1}{2}}Z + \boldsymbol{\alpha}'C^{-\frac{1}{2}}Z'$, we have

$$
\begin{aligned}
\mathrm{TD}(Z, Z'; Y) =& \frac{1}{n}\Big\|\boldsymbol{\alpha}A^{-\frac{1}{2}}[ZZ^\top(ZZ^\top)^{-1}Z - ZZ'^\top(Z'Z'^\top)^{-1}Z'] \\
&+\boldsymbol{\alpha}'A^{-\frac{1}{2}}[Z'Z^\top(ZZ^\top)^{-1}Z - Z'Z'^\top(Z'Z'^\top)^{-1}Z']\Big\|_2^2 \\
\xrightarrow{a.s.}& \frac{1}{n}\left\|\boldsymbol{\alpha}A^{-\frac{1}{2}}(Z - BC^{-1}Z') + \boldsymbol{\alpha}'C^{-\frac{1}{2}}(B^\top A^{-1}Z - Z')\right\|_2^2 \\
=& \frac{1}{n}\left[\left\|\boldsymbol{\alpha}A^{-\frac{1}{2}}(Z - BC^{-1}Z')\right\|_2^2 + \left\|\boldsymbol{\alpha}'C^{-\frac{1}{2}}(B^\top A^{-1}Z - Z')\right\|_2^2\right. \\
&\left.+ 2\boldsymbol{\alpha}A^{-\frac{1}{2}}(Z - BC^{-1}Z')(B^\top A^{-1}Z - Z')^\top C^{-\frac{1}{2}}\boldsymbol{\alpha}'^\top\right] \\
\xrightarrow{a.s.}& \boldsymbol{\alpha}(I_p - DD^\top)\boldsymbol{\alpha}^\top + \boldsymbol{\alpha}'(I_{p'} - D^\top D)\boldsymbol{\alpha}'^\top + 2\boldsymbol{\alpha}(D - DD^\top D)\boldsymbol{\alpha}'^\top,
\end{aligned}
\tag{12}
$$

Thus, we have the results in Theorem 1. It shows that the difference between Z's and Z''s performance primarily depends on the matrix D and the task Y $(\boldsymbol{\alpha}, \boldsymbol{\alpha}')$. □

A.2 PROOF OF THEOREM 2

When evaluated on the *representative task set*, $\mathcal{S}^* = \{Y = \boldsymbol{\alpha}A^{-\frac{1}{2}}Z + \boldsymbol{\alpha}'C^{-\frac{1}{2}}Z' : \boldsymbol{\alpha} \in \mathbb{R}^p, \boldsymbol{\alpha} \in \mathbb{R}^{p'}, \|\boldsymbol{\alpha}\|_2 \le 1, \|\boldsymbol{\alpha}'\|_2 \le 1\}$, the transferred discrepancy is

$$
\begin{aligned}
\mathrm{TD}(Z, Z'; \mathcal{S}^*) =& \sup_{\|\boldsymbol{\alpha}\|_2 \le 1, \|\boldsymbol{\alpha}'\|_2 \le 1} \mathrm{TD}(Z, Z'; Y\boldsymbol{\alpha}A^{-\frac{1}{2}}Z + \boldsymbol{\alpha}'C^{-\frac{1}{2}}Z') \\
\xrightarrow{a.s.}& \lim_{n\to\infty} \sup_{\|\boldsymbol{\alpha}\|_2 \le 1, \|\boldsymbol{\alpha}'\|_2 \le 1} \mathrm{TD}(Z, Z'; Y\boldsymbol{\alpha}A^{-\frac{1}{2}}Z + \boldsymbol{\alpha}'C^{-\frac{1}{2}}Z').
\end{aligned}
\tag{13}
$$

The TD metric is a polynomial function of $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}'$ and the limit only affect the coefficients, and $\mathcal{A} = \{(\boldsymbol{\alpha}, \boldsymbol{\alpha}') | \|\boldsymbol{\alpha}\|_2 \le 1, \|\boldsymbol{\alpha}'\|_2 \le 1\}$ is a compact set in $\mathbb{R}^{p+p'}$. Thus, the $\lim$ and the $\sup$ in Eqn (13) is interchangeable. According to Theorem 1, there is

$$
\begin{aligned}
\mathrm{TD}(Z, Z'; \mathcal{S}^*) \xrightarrow{a.s.}& \sup_{(\boldsymbol{\alpha}, \boldsymbol{\alpha}) \in \mathcal{A}} \lim_{n\to\infty} \mathrm{TD}(Z, Z'; Y\boldsymbol{\alpha}A^{-\frac{1}{2}}Z + \boldsymbol{\alpha}'C^{-\frac{1}{2}}Z') \\
=& \sup_{(\boldsymbol{\alpha}, \boldsymbol{\alpha}) \in \mathcal{A}} [\boldsymbol{\alpha}(I_p - DD^\top)\boldsymbol{\alpha}^\top + \boldsymbol{\alpha}'(I_{p'} - D^\top D)\boldsymbol{\alpha}'^\top + 2\boldsymbol{\alpha}(D - DD^\top D)\boldsymbol{\alpha}'^\top].
\end{aligned}
\tag{14}
$$

Let the singular values of D be $\sigma_1 \ge \cdots \ge \sigma_p$, and the SVD of D be $D = U\Sigma V^\top$. Since the covariance matrix $\begin{bmatrix} A & B \\ B^\top & C \end{bmatrix} \succeq 0$, its schur complement $A - BC^{-1}B^\top \succeq 0$. Thus,

$$
A - BC^{-1}B^\top = A^{\frac{1}{2}}\left(I_p - A^{-\frac{1}{2}}BC^{-1}B^\top A^{-\frac{1}{2}}\right)A^{\frac{1}{2}} = A^{\frac{1}{2}}\left(I_p - DD^\top\right)A^{\frac{1}{2}} \succeq 0, \tag{15}
$$

which implies that $1 \ge \sigma_1 \ge \cdots \ge \sigma_p \ge 0$. Denote $\alpha U = \boldsymbol{\beta} = (\beta_1, \cdots, \beta_p)$ and $\alpha' V = \boldsymbol{\beta}' = (\beta'_1, \cdots, \beta'_{p'})$. Then $\|\boldsymbol{\beta}\|_2 \le 1$, $\|\boldsymbol{\beta}'\|_2 \le 1$, and

$$
\mathrm{TD}(Z, Z'; \mathcal{S}^*) \xrightarrow{a.s.} \sup_{\boldsymbol{\beta}, \boldsymbol{\beta}'}[\sum_{j=1}^{p}(1 - \sigma_j^2)(\beta_j^2 + \beta_j'^2 + 2\sigma_j\beta_j\beta'_j) + \sum_{k=p+1}^{p'}\beta_k'^2]. \tag{16}
$$

When $p = p'$, Cauchy-Schwarz inequality guarantees that $2\beta_j\beta'_j \leq \beta_j^2 + \beta_j'^2$, where the equation holds when $\beta_j = \beta'_j$. It follows that

$$\begin{aligned} \mathrm{TD}(Z, Z'; \mathcal{S}^*) &\xrightarrow{a.s.} \sup_{\boldsymbol{\beta}, \boldsymbol{\beta}'} \sum_{j=1}^{p} (1-\sigma_j)(1+\sigma_j)^2(\beta_j^2 + \beta_j'^2) \\ &= \max_{j=1,\cdots,p} 2(1-\sigma_j)(1+\sigma_j)^2. \end{aligned} \tag{17}$$

When $p < p'$, if there exists an i such that $(1-\sigma_j)(1+\sigma_j)^2 \geq 1$, then the optimal $\boldsymbol{\beta}$ satisfies $\beta'_{p+1} = \cdots = \beta'_{p'} = 0$, so we can achieve the same result as (17). Thus, we prove the results in Theorem 2. □

A.3 Proof of Corollary 1

Here, we use the same notation as the previous section. Based on the results in Theorem 2, we now consider restricted task sets. In $\mathcal{S}_r = \{Y = \boldsymbol{\alpha} A^{-\frac{1}{2}} Z + \boldsymbol{\alpha}' C^{-\frac{1}{2}} Z' : \|\boldsymbol{\alpha}\|_2 \leq 1, \|\boldsymbol{\alpha}'\|_2 \leq 1, \boldsymbol{\alpha}\mathbf{u}_j = 0 \text{ and } \boldsymbol{\alpha}'\mathbf{v}_j = 0 \text{ if } j > r\}$, we have $\beta_j = \beta'_j = 0$ for $j > r$. It yields

$$\begin{aligned} \mathrm{TD}(Z, Z'; \mathcal{S}_r) &\xrightarrow{a.s.} \sup_{\boldsymbol{\beta}, \boldsymbol{\beta}'} \sum_{j=1}^{r} (1-\sigma_j)(1+\sigma_j)^2(\beta_j^2 + \beta_j'^2) \\ &= \max_{j=1,\cdots,r} 2(1-\sigma_j)(1+\sigma_j)^2. \end{aligned} \tag{18}$$

which concludes the proof. □

A.4 The Invairance Properties of the TD metric

For a general loss function $\ell(h_W(\mathbf{z}), y)$ that is Lipschitz, strongly convex in $h_W(\mathbf{z})$ and satisfies $\ell(y, y) = 0$, we will show that it also satisfy the two invariance properties mentioned in Section 4. When training the output head's parameter $\hat{W}$ from

$$\hat{W} = \arg\min_{W} \frac{1}{n} \sum_{i=1}^{n} \ell(h_W(\mathbf{z}_i), y_i)), \tag{19}$$

there exists only one optimal $h_W(\mathbf{z}_i)$ due to the convexity of the loss function ℓ. Denote the output set of $h_W(\mathbf{Z})$ as $S_{\mathbf{z}} := \{h_{\hat{W}}(\mathbf{z}) | \forall W\}$. It is easy to see that, if there is a linear transformation directly applied to $\mathbf{z}$ by $h_W(\mathbf{z})$, the output set will be invariant to isotropic scaling and orthogonal transformation. $S_{\mathbf{z}} = S_{\beta\mathbf{z}}$ for all β and $S_{\mathbf{z}} = S_{Q\mathbf{z}}$ for all unitary matrix Q. Although the optimization algorithm may find different W, the output $h_W(\mathbf{z}_i)$ remains unchanged for each data. Therefore, the TD metric defined as $\mathrm{TD}(Z, Z'; Y) = \frac{1}{n}\sum_{i=1}^{n} d(h_W(\mathbf{z}_i), h_{W'}(\mathbf{z}'_i))$ is invariant to isotropic scaling and orthogonal transformation.

A.5 Empirical Analysis of the Distribution of Singular Values

Previous theoretical analysis shows that in the linear probing setting, the TD metric is closely related to the distribution of singular values of $D = A^{-\frac{1}{2}} B C^{-\frac{1}{2}}$. If all the singular values are large, then the two representations will have similar performance on all the tasks in the representative set. If there exists a small singular value, then there exists a direction that is close to one representation matrix's row space but nearly orthogonal to the other representation matrix's row space. On a task $y \in \mathcal{S}^*$ in this direction, these two representations will have drastically different performance. When all the singular values are small, these two representations will perform differently on all tasks in the representative task set. Now, we empirically look at the distribution of singular values of D in practice.

We train two ResNet32 models on the Cifar-10 training set with different random initializations (i.e. seeds), extract the representation matrix on the Cifar-10 test set, and calculate the empirically

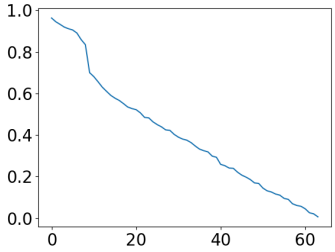

Figure 1: Singular values of D

estimator $\hat{D} = (ZZ^\top)^{-\frac{1}{2}} ZZ'^\top (Z'Z'^\top)^{-\frac{1}{2}}$. The training hyperparameters are the same as in C.1. We plot the distribution of the singular values of $\hat{D}$ in Figure 1. It can be seen that the singular values larger than 0.6 take only a small fraction among all the singular values. Most singular values are small, and there exist singular values close to 0. Thus, the two representations will have similar performance on the task related to the large singular value and have drastically different performance on the task related to the small singular value. Therefore, a universal similarity index may not reveal all the difference, and one may consider evaluating the two representations on a set of downstream tasks of interest. When evaluated on the restricted task set we proposed in Section 4.2, the TD is 0.288 on $\mathcal{S}_1$ and 0.649 on $\mathcal{S}_5$. When using TD in practice, the task set can also be selected as various tasks we want to deal with using the pre-trained feature extractor.

In Section 4.3, Theorem 3 provides that the CCA index obtains a universal index by taking the average over all tasks in the representative set or over the squares of the singular values. Since most of the singular values are small and the performance is different on most tasks, the CCA value shall be large and varies little for different models. In Section D's experimental results, the CCA value is always larger than 0.6 and remains stable when trained with different techniques of data augmentation and different training strategies.

B Transferred Discrepancy versus the Maximum Match, CCA and CKA

B.1 Main Results for the Maximum Match and CKA

In Section 4.3 we discussed the relationship between CCA and TD. Here we present our main results for the maximum match and CKA, which show that both of them are closely related to the matrix D.

Maximum Match Max-match measures the intersection between two subspaces of the row spaces of Z and Z'. Denote the row vectors of $A^{-\frac{1}{2}}Z$ by $Z_{row} = \{z^{(1)}, \cdots, z^{(p)}\}$ and the row vectors of $C^{-\frac{1}{2}}Z'$ by $Z'_{row} = \{z'^{(1)}, \cdots, z'^{(p')}\}$. Let $\hat{Z}_{row}$ and $\hat{Z}'_{row}$ be the largest subspaces of Z_{row} and Z'_{row} such that they are ϵ-close to each other ($\epsilon \geq 0$). Here closeness means that for any $z^{(i)} \in \hat{Z}_{row}$, $\min_{z' \in \text{span}(\hat{Z}'_{row})} \left\| z^{(i)} - z' \right\|_2 \leq \sqrt{n}\epsilon$ and vice versa. The max-match similarity index is defined as

$$S_{max-match}(\epsilon) = \frac{|\hat{Z}_{row}| + |\hat{Z}'_{row}|}{p + p'} \tag{20}$$

Compared with the original definition in Wang et al. (2018), we normalize Z and Z' by $A^{-\frac{1}{2}}$ and $C^{-\frac{1}{2}}$, so that they are at the same scale with respect to ϵ. We also add $\sqrt{n}$ in the definition of ϵ-closeness so that the result is meaningful as $n \to \infty$. The following theorem provides the relationship between $S_{max-match}$ and D:

Theorem 4. *Let $s = \mathbb{E}_{Z,Z'}[S_{max-match}(\epsilon)]$. Denote the singular values of D by $\sigma_1 \geq \cdots \geq \sigma_p$. Let k be the largest integer such that*

$$\sigma_1^2 + \cdots + \sigma_k^2 \geq k(1-\epsilon^2), \tag{21}$$

Then we have

$$s \leq \frac{2k}{p+p'}. \tag{22}$$

(Wang et al., 2018) found that the maximum match similarity is very small when $\epsilon \leq 0.3$. Figure 1 shows that large singular values account for a very small fraction of all singular values of D. Thus, it is a natural consequence of Theorem 4 that $S_{max-match}(\epsilon)$ should be very small. This metric investigate the two feature matrix under linear transformation, and Liang et al. (2020) extended it to non-linear transformation using several convolution and activation layers.

Centered Kernel Alignment (CKA) Kornblith et al. (2019) proposed CKA based on dot product and its extension in reproducing Hilbert spaces. Particularly, the linear CKA similarity index is defined as

$$S_{CKA}(Z^\top Z, Z'^\top Z') = \frac{\|Z'Z^\top\|_F^2}{\|ZZ^\top\|_F \|Z'Z'^\top\|_F}. \tag{23}$$

Theorem 5. *The expectation of CKA is*

$$\mathbb{E}[\text{CKA}(Z^\top Z, Z'^\top Z^\top)] = \frac{\mathbf{tr}(DCD^\top A)}{\sqrt{\mathbf{tr}(A^2)}\sqrt{\mathbf{tr}(C^2)}}. \tag{24}$$

This result is straightforward from the definition. It indicates that the CKA gets a universal similarity index by re-weighting the matrix D with matrices A and C.

B.2 Lemma for Theorem 3

Lemma 1. *If $\mathbf{x} \in \mathbb{R}^p$ is uniformly distributed in the unit ball, then $\mathbb{E}\mathbf{x}_1^2 = \frac{1}{p+2}$.*

Proof We directly compute this expectation:

$$\mathbb{E}\mathbf{x}_1^2 = \frac{\int_{-1}^1 \int_{\mathbf{x}_2^2+\cdots+\mathbf{x}_p^2 \leq 1-\mathbf{x}_1^2} \mathbf{x}_1^2 dV d\mathbf{x}_1}{\int_{-1}^1 \int_{\mathbf{x}_2^2+\cdots+\mathbf{x}_p^2 \leq 1-\mathbf{x}_1^2} dV d\mathbf{x}_1}. \tag{25}$$

Denote the volume of the unit ball in $\mathbb{R}^p$ by V_p. We have $\int_{\mathbf{x}_2^2+\cdots+\mathbf{x}_p^2 \leq t^2} dV = t^{p-1}V_{p-1}$. Denote the Beta function by $B(P,Q)$. For the numerator, let $t^2 = 1-\mathbf{x}_1^2$, and we have

$$\begin{aligned}
\int_{-1}^1 \int_{\mathbf{x}_2^2+\cdots+\mathbf{x}_p^2 \leq 1-\mathbf{x}_1^2} \mathbf{x}_1^2 dV d\mathbf{x}_1 &= \int_{-1}^1 t^{p-1} V_{p-1} \mathbf{x}_1^2 dV d\mathbf{x}_1 \\
&= V_{p-1} \int_{-1}^1 (1-\mathbf{x}_1^2)^{\frac{p-1}{2}} \mathbf{x}_1^2 d\mathbf{x}_1 \\
&= V_{p-1} \left(\int_{-1}^1 (1-\mathbf{x}_1^2)^{\frac{p-1}{2}} d\mathbf{x}_1 - \int_{-1}^1 (1-\mathbf{x}_1^2)^{\frac{p+1}{2}} d\mathbf{x}_1 \right) \\
&= V_{p-1} \left(\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \cos^p \theta d\theta - \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \cos^{p+2} \theta d\theta \right) \\
&= V_{p-1} \left(B(\frac{p+1}{2}, \frac{1}{2}) - B(\frac{p+3}{2}, \frac{1}{2}) \right) \\
&= \frac{1}{p+2} V_{p-1} B(\frac{p+1}{2}, \frac{1}{2}).
\end{aligned} \tag{26}$$

Similarly, the denominator is equal to $V_{p-1}B(\frac{p+1}{2}, \frac{1}{2})$. Thus, $\mathbb{E}\mathbf{x}_1^2 = \frac{1}{p+2}$. □

B.3 Proof of Theorem 3

Suppose $\hat{D}$ is defined as the empirical estimator of D, i.e. $\hat{D} = (Z'Z'^\top)^{-\frac{1}{2}}Z'Z^\top(ZZ^\top)^{-\frac{1}{2}}$. Denote the singular values of $\hat{D}$ as $1 \geq \hat{\sigma}_1 \geq ... \geq \hat{\sigma}_p \geq 0$. Similar to Eqn (6) and Eqn (12), we have

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{\alpha},\boldsymbol{\alpha}'}\left[\text{TD}(Z, Z'; Y = \boldsymbol{\alpha}\hat{A}^{-\frac{1}{2}}Z + \boldsymbol{\alpha}'\hat{C}^{-\frac{1}{2}}Z')\right] \\
=&\mathbb{E}_{\boldsymbol{\alpha},\boldsymbol{\alpha}'}\left[\boldsymbol{\alpha}(I_p - \hat{D}\hat{D}^\top)\boldsymbol{\alpha}^\top + \alpha'(I_{p'} - \hat{D}^\top\hat{D})\alpha'^\top + 2\alpha(\hat{D} - \hat{D}\hat{D}^\top\hat{D})\alpha'^\top\right] \\
=&\mathbb{E}_{\boldsymbol{\beta},\boldsymbol{\beta}'}\left[\sum_{j=1}^{p}(1-\hat{\sigma}_j^2)(\beta_i^2 + \beta_i'^2 + 2\hat{\sigma}_i\beta_i\beta_i') + \sum_{k=p+1}^{p'}\beta_k'^2\right].
\end{aligned} \tag{27}$$

Here we also change the variable $\boldsymbol{\alpha}, \boldsymbol{\alpha}'$ to $\boldsymbol{\beta}, \boldsymbol{\beta}'$ with $\boldsymbol{\beta} = \boldsymbol{\alpha}\hat{U}$ and $\boldsymbol{\beta}' = \boldsymbol{\alpha}'\hat{V}$. $\hat{U}$ and $\hat{V}$ are the orthogonal matrices in $\hat{D}$'s singular value decomposition $\hat{D} = \hat{U}\hat{\Sigma}\hat{V}^\top$. Since $\boldsymbol{\alpha}$ is uniformly distributed in the unit ball in $\mathbb{R}^p$, so does $\boldsymbol{\beta}$. Similarly, $\boldsymbol{\beta}'$ is uniformly distributed in the unit ball in $\mathbb{R}^{p'}$. Thus, $\mathbb{E}\beta_j\beta_j' = 0$. It follows by Lemma 1 that $\mathbb{E}\beta_j^2 = \frac{1}{p+2}$ for all $j \in [p]$, and $\mathbb{E}\beta_k'^2 = \frac{1}{p'+2}$ for all $k \in [p']$. Therefore,

$$\mathbb{E}_{\boldsymbol{\alpha},\boldsymbol{\alpha}'}[\text{TD}(Z, Z'; Y = \boldsymbol{\alpha}\hat{A}^{-\frac{1}{2}}Z + \boldsymbol{\alpha}'\hat{C}^{-\frac{1}{2}}Z')] = \frac{p - \sum_{j=1}^{p}\hat{\sigma}_j^2}{p+2} + \frac{p' - \sum_{j=1}^{p}\hat{\sigma}_j^2}{p'+2}. \tag{28}$$

Similar to Eqn (10), $R_{CCA}^2 = \frac{\|\hat{D}\|_F^2}{p} = \frac{\sum_{j=1}^{p}\hat{\sigma}_j^2}{p}$. Thus, we have

$$\mathbb{E}_{\alpha,\alpha'}[\text{TD}(Z, Z'; Y = \boldsymbol{\alpha}\hat{A}^{-\frac{1}{2}}Z + \boldsymbol{\alpha}'\hat{C}^{-\frac{1}{2}}Z')] = \frac{2p}{p+2}(1 - R_{CCA}^2). \tag{29}$$

This results shows that the calculation of R_{CCA} is equivalent to averaging on the TD over all linearly realizable tasks. This averaging produces a downstream-task-agnostic metric but dismissing many information. Consequently, the CCA varies little in all the experiments and is insensitive to the change of training strategies. □

B.4 Lemma for Theorem 4

Lemma 2. *Suppose that $A \in \mathbb{R}^{n\times n}$ is a symmetric semi-positive definite matrix. Let the eigenvalues of A be $0 \leq \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$. For a constant $\epsilon \geq 0$, let k be the largest integer such that $\lambda_1 + \cdots + \lambda_k \leq k\epsilon$. Suppose that $\mathbf{v}_1, \cdots, \mathbf{v}_m \in \mathbb{R}^n$ are unit vectors orthogonal to each other, i.e. $\|\mathbf{v}_i\|_2 = 1$ for all i and $\mathbf{v}_i\mathbf{v}_j^\top = 0$ for all $i \neq j$. If $\mathbf{v}_i A\mathbf{v}_i^\top \leq \epsilon$ for all $i = 1, \cdots, m$, then $m \leq k$.*

Proof There exists a unitary matrix Q such that $A = Q\Lambda Q^\top$, where $\Lambda = \text{diag}\{\lambda_1, \lambda_2, \cdots, \lambda_n\}$. Let $\mathbf{u}_i = Q\mathbf{v}_i$, $i = 1, \cdots, m$. Due to the property of unitary matrices, $\mathbf{u}_i$ are unit vectors orthogonal to each other. Denote $\mathbf{u}_i = (\mathbf{u}_i^1, \mathbf{u}_i^2, \cdots, \mathbf{u}_i^n)$. Then, we have

$$\epsilon \geq \mathbf{v}_i A\mathbf{v}_i^\top = \mathbf{u}_i\Lambda\mathbf{u}_i^\top = \sum_{j=1}^{n}\lambda_j(\mathbf{u}_i^j)^2. \tag{30}$$

Let $x_j = \sum_{i=1}^{m}(\mathbf{u}_i^j)^2$. Since $\mathbf{u}_i$ are unit vectors orthogonal to each other, we have $0 \leq x_j \leq 1$ and $x_1 + \cdots + x_n = m$. Thus,

$$m\epsilon \geq \sum_{i=1}^{m}\sum_{j=1}^{n}\lambda_j(\mathbf{u}_i^j)^2 = \sum_{j=1}^{n}\lambda_j x_j \geq \lambda_1 + \cdots + \lambda_m. \tag{31}$$

Therefore, by the definition of k we have $m \leq k$. □

B.5 PROOF OF THEOREM 4

Let $\hat{Z}_{row}$ and $\hat{Z'}_{row}$ be the two largest subspaces described in the theorem. For each $z^{(i)} \in \hat{Z}_{row}$, there exists an $\mathbf{a}_i \in \mathbb{R}^{p'}$ such that $\left\| z^{(i)} - \mathbf{a}_i C^{-\frac{1}{2}} Z' \right\|_2 \leq \sqrt{n}\epsilon$. Let $1_i = (0, \cdots, 0, 1, 0, \cdots, 0)$ be a p-dimensional row vector whose i^{th} element is 1 and the rest are 0. Thus, $\mathbf{b}_i = (1_i, -\mathbf{a}_i)$ satisfy

$$\mathbf{b}_i \begin{bmatrix} A^{-\frac{1}{2}} Z Z^\top A^{-\frac{1}{2}} & A^{-\frac{1}{2}} Z Z'^\top C^{-\frac{1}{2}} \\ C^{-\frac{1}{2}} Z' Z^\top A^{-\frac{1}{2}} & C^{-\frac{1}{2}} Z' Z'^\top C^{-\frac{1}{2}} \end{bmatrix} \mathbf{b}_i^\top = \| z^{(i)} - \mathbf{a}_i C^{-\frac{1}{2}} Z' \|_2^2 \leq n\epsilon^2. \tag{32}$$

By taking the expectation we have

$$\epsilon^2 \geq \mathbf{b}_i \begin{bmatrix} \boldsymbol{I}_p & D \\ D^\top & \boldsymbol{I}_{p'} \end{bmatrix} \mathbf{b}_i^\top = 1_i (I_p - DD^\top) 1_i^\top + \| \mathbf{a}_i - 1_i D \|_2^2 \tag{33}$$

Thus, for every $z^{(i)} \in \hat{Z}_{row}$ we have

$$1_i (I_p - DD^\top) 1_i^\top \leq \epsilon^2 \tag{34}$$

The eigenvalues of $I_p - DD^\top$ are $0 \leq 1 - \sigma_1^2 \leq \cdots \leq 1 - \sigma_p^2$. Since 1_i are unit vectors orthogonal to each other, using Lemma 2 we have $|\hat{Z}_{row}| \leq k$. Similarly, for each $z'^{(j)} \in \hat{Z'}_{row}$,

$$1_j (I_{p'} - D^\top D) 1_j^\top \leq \epsilon^2. \tag{35}$$

Consequently, $|\hat{Z'}_{row}| \leq k$. Thus, $s \leq \frac{2k}{p+p'}$. □

C EXPERIMENTAL SETTINGS

All experimental settings are listed in this section and results are reported in Section D.

C.1 SETTINGS OF SECTIONS 5.1 AND 5.2

The Design of Upstream Tasks and Downstream Tasks We design several tasks based on the CIFAR-10 dataset and the SVHN dataset. For the CIFAR-10 dataset, we manually group the labels to design three different tasks: Cifar-10, Cifar-5 and Cifar-2, each of which can be used either as an upstream task or as a downstream task. The Cifar-10 task uses the original CIFAR-10 labels, and Cifar-5 and Cifar-2 are built by semantically regrouping those 10 classes into 5 and 2 categories, as shown in Table 4. Figure 2 provides a visualization of the categories of Cifar-5 and Cifar-10. For the SVHN dataset, we directly use its original labels.

Table 4: Two tasks induced from CIFAR-10: Cifar-2 task(first table) and Cifar-5 task(second table).

Cifar-2 Category	Classes
Man-made transport	Airplane, Automobile, ship, truck
animals	Bird, cat, deer, dog, horse, frog

Cifar-5 Category	Classes
Cars	Automobile, truck
Large mammals	Deer, horse
Medium mammals	Cat, dog
Large transport	Ship, airplane
Non-mammals	Frog, bird

Architecture We use ResNet-32 (He et al., 2016) for all tasks.

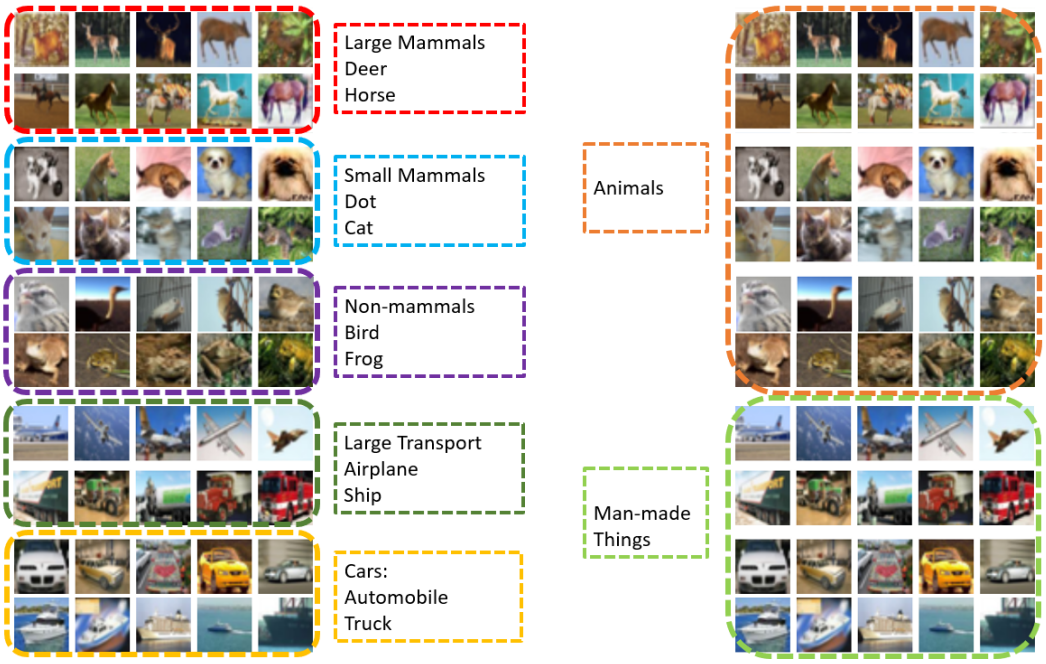

Figure 2: A visualization of Cifar-2 (left) and Cifar-5 (right).

Upstream Task Training (Feature Extractor) For any upstream task, we train the model for 200 epochs with SGD and the batch size is set to 128. The learning rate is initially set to 0.1 and decayed by 0.1 at epochs 60 and 120. The momentum is 0.9, and the weight decay rate is 5e-4. Unless explicitly stated, random cropping and random horizontal flipping are applied for data augmentation. After training, we remove the network's last fully-connected layer (output head) and take the remaining network as the feature extractor.

Downstream Task Training For any downstream task, we add a linear layer with softmax on top of the feature extractor as the output head. We freeze the feature extractor and only fine-tune the output head using the training set of the downstream task. The output head is trained for 50 epochs with SGD and the batch size is set to 128. All other hyperparameters are set to the same value as the upstream task training. The model's performance on the test set is reported.

More TD Metrics for Evaluation Recall that the TD metric is defined as $d(h_{W_1}(\mathbf{z}), h_{\hat{W}_2}(\mathbf{z}))$ in (2), where $d(\cdot, \cdot)$ is a distance metric. We consider two TD metrics defined with different $d(\cdot, \cdot)$:

- Soft Distance: $\text{TD}_{soft}(Z, Z') = \mathcal{W}_1(h_{W_1}(Z), h_{W_2}(Z')) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}\|h_{W_1}(Z_i) - h_{W_2}(Z_i')\|_1$,
- Hard Distance: $\text{TD}_{hard}(Z, Z') = \frac{1}{n}\sum_{i=1}^{n}\mathbf{1}_{\arg\max h_{W_1}(Z_i) = \arg\max h_{W_2}(Z_i')}$.

TD_{hard} is used in the paper as TD_{cls}, which directly measures the fraction of samples on which the two models have different predictions. TD_{soft} defined with the l_1 distance is equivalent to the Wasserstein distance between the two probability densities. Both TD_{hard} and TD_{soft} are metrics between [0,1], and small numbers indicate more similar representations. Although TD_{soft} and TD_{hard} are based on different distance metrics, they exhibit similar trends in most experiments.

Tasks Based on CIFAR-100 We also conduct experiments with the CIFAR-100 dataset, in which each sample has two levels of labels: a fine label (100 classes) and a coarse label (20 superclasses) as listed in Table 5. The two levels of original labels produce two tasks: C 100 and C 20. We further design two tasks, C 10 and C 4, by semantically regrouping the samples based on the labels' relationship extracted from the WordTree in ImageNet (Krizhevsky et al., 2012). Details are shown in Table 6.

Table 5: Two original labelings from CIFAR-100: 100 classes and 20 superclasses.

Superclass (C 20 Category)	Classes
aquatic mammals	beaver, dolphin, otter, seal, whale
fish	aquarium fish, flatfish, ray, shark, trout
flowers	orchids, poppies, roses, sunflowers, tulips
food containers	bottles, bowls, cans, cups, plates
fruit and vegetables	apples, mushrooms, oranges, pears, sweet peppers
household electrical devices	clock, computer keyboard, lamp, telephone, television
household furniture	bed, chair, couch, table, wardrobe
insects	bee, beetle, butterfly, caterpillar, cockroach
large carnivores	bear, leopard, lion, tiger, wolf
large man-made outdoor things	bridge, castle, house, road, skyscraper
large natural outdoor scenes	cloud, forest, mountain, plain, sea
large omnivores and herbivores	camel, cattle, chimpanzee, elephant, kangaroo
medium-sized mammals	fox, porcupine, possum, raccoon, skunk
non-insect invertebrates	crab, lobster, snail, spider, worm
people	baby, boy, girl, man, woman
reptiles	crocodile, dinosaur, lizard, snake, turtle
small mammals	hamster, mouse, rabbit, shrew, squirrel
trees	maple, oak, palm, pine, willow
vehicles 1	bicycle, bus, motorcycle, pickup truck, train
vehicles 2	lawn-mower, rocket, streetcar, tank, tractor

Table 6: CIFAR-100-induced task: C 4 (first table) with four classes and C 10 (second table) with 10 classes.

C 4 Category	Superclass
mammals	Aquatic mammals, large carnivores, large omnivores and herbivores, small mammals, medium-sized mammals, people
Non-mammals	Fish, Reptiles, insects, non-insect invertebrates,
Man-made things	Vehicles 1, vehicles 2, Food containers, household electrical devices, household furniture, Large man-made outdoor things
Natural things and plants	Trees, flowers, fruit and vegetables, Large natural outdoor scenes

C 10 Category	Superclass
Aquatic animals	Aquatic mammals, fish
Large animals	large carnivores, large omnivores and herbivores
Medium and small mammals	small mammals, medium-sized mammals
Vehicles	Vehicles 1, vehicles 2.
Other animals	Reptiles, insects, non-insect invertebrates
People	people
Plants	Trees, flowers, fruit and vegetables
Household	Food containers, household electrical devices, household furniture
Large man-made outdoor things	Large man-made outdoor things
Large natural outdoor scenes	Large natural outdoor scenes

C.2 Settings of Section 5.3

Data Augmentation We study three data augmentation techniques: random flipping, random cropping (padding = 4), and adding Gaussian noise ($\sigma = 0.1$).

Learning Rate Schedule We train the feature extractor for 100 epochs with three learning rate schedules. In the "Small LR + without decay" schedule, the learning rate is fixed at 0.01; in the "Large LR + without decay" schedule, the learning rate is fixed at 0.1; and in the "Large LR + with decay" schedule, the learning rate is 0.1 in the first 50 epochs and 0.01 in the last 50 epochs.

Adversarial training Adversarial training (Madry et al., 2017) is one of the successful training method to improve the robustness of neural networks to adversarial attacks. The main idea to add adversarial examples into the training set during training to improve robustness. We further investigat how adversarial training affect TD robustness. We use PGD-7 to generate adversarial examples and set the perterbation to $\frac{8}{255}$ and step size to $\frac{2}{255}$ during training. When trained on CIFAR-10, this setting of the adversarial training can achieve 79.22% clean accuracy and 48.23 % robustness under the same level of attacks.

Apart from the three factors above, we also study how other factors affect TD robustness.

Batch Size We train the feature extractor with different batch sizes. We set the batch size to 32, 64, 128, 256, and 512, following the common practice of setting the batch size to a power of 2.

Architectures We investigate how the model's width and depth affect the TD robustness. For the ResNet models (He et al., 2016), we study models with different depths including ResNet20, ResNet32, ResNet44, ResNet56 and ResNet110, and models with different widths (Wide ResNet) such as 2xResNet32, 5xResNet32, and 10xResNet32. We also experiments on VGG models (Simonyan & Zisserman, 2014) including VGG 13-bn, VGG 16-bn, and VGG 19-bn.

Upstream Tasks We study how the choice of upstream task affects the representation learned by the feature extractor. We use the four tasks based on the CIFAR-100 dataset: C 4/10/20/100.

D Experimental Results

D.1 A Sanity Check

The sanity check results are reported in Section 5.1 in Table 7, including TD_{soft} for each row. It is clear that the first three rows have smaller D_{CCA}, D_{CKA} and TD_{hard} values than the other rows, which indicates that the features learned from Cifar-2/5/10 are more similar, but are very different from the features learned from SVHN.

Table 7: The sanity check results

Model 1	Model 2	Downstream	D_{CCA}	D_{CKA}	TD_{soft}	TD_{hard}
Cifar-10	Cifar-5	Cifar-10	0.7642	0.3024	0.3344	0.2158
Cifar-10	Cifar-2		0.8323	0.5330	0.6717	0.4958
Cifar-5	Cifar-2		0.8030	0.4530	0.5057	0.4944
Cifar-10	SVHN		0.9218	0.9713	0.8528	0.8001
Cifar-5	SVHN		0.9793	0.9710	0.7521	0.8049
Cifar-2	SVHN		0.9163	0.9782	0.5000	0.8163

D.2 Does Initialization Affect Learned Features?

In this section, we further study whether models trained from different random initializations learn similar representations using a variety of downstream tasks. The results are reported in Table 8.

As shown by the results, TD_{soft} and TD_{hard} exhibit similar trends and output different values for different downstream tasks. When we train the model on Cifar-5, both models provide consistent predictions on Cifar-2, but the variance gets much bigger on Cifar-10. Furthermore, it can be observed that from C 4 to C 100, as the task becomes more difficult, the difference between the two representations increases. On C 4, the two models trained on Cifar-5 disagree on 32.77% of the test samples. When evaluated on C 100, however, the same two models disagree on as much as 80.05% of the data. The variation of TD as demonstrated in the table implies that a reasonable metric measuring the difference between two representations should take the downstream tasks into consideration.

Table 8: Similarity of the models learned with different initializations.

Model 1	Model 2	Downstream	D_{CCA}	D_{CKA}	TD_{soft}	TD_{hard}
Cifar-5	Cifar-5	Cifar-10	0.6961	0.0835	0.1617	0.2139
		Cifar-5			0.0200	0.0442
		Cifar-2			0.0124	0.0109
		C 100	0.8003	0.4210	0.3113	0.8005
		C 20			0.2599	0.6377
		C 10			0.2347	0.5010
		C 4			0.1845	0.3277
		SVHN	0.8663	0.6940	0.1765	0.5312
Cifar-2	Cifar-2	Cifar-10	0.6931	0.0402	0.1489	0.3745
		Cifar-5			0.1281	0.2631
		Cifar-2			0.0168	0.0164
		C 100	0.7533	0.3036	0.1762	0.7651
		C 20			0.1595	0.6018
		C 10			0.1485	0.4259
		C 4			0.1243	0.2644
		SVHN	0.8330	0.6305	0.1296	0.4157

Now we try to answer whether the features learned by two models trained from different initializations are similar. We want to emphasize that the question depends on how we evaluate the features, i.e., which downstream task set we choose. Among all the tasks, Cifar-2/5/10 are the most correlated. The models trained on Cifar-5 can achieve 80% accuracy on Cifar-10, indicating that the features can be successfully transferred. However, even on such a correlated task, the two models disagree on 21.39% of the test samples. Thus, the features captured by models trained from different initializations are not the same. The TD metric allows a practitioner to choose the downstream tasks of their interest, and both TD_{hard} and TD_{soft} have clear practical meanings: TD_{hard} shows the fraction of samples that two models disagree on while TD_{soft} measures the difference between predictions in terms of the likelihood indicated by the softmax output.

D.3 TD Robustness and Its Application to Training Strategy Evaluation

Data Augmentation We calculate TD robustness under various configurations of data augmentation's techniques, including random flipping (F), random cropping (C), and Gaussian additive noise (G). Full results are reported in Table 9.

Table 9: The effect of data augmentation: We study random flipping (F), random cropping (C) and adding Gaussian noise (G). +/- indicates whether the data augmentation method is applied.

Upstream	Downstream	F	C	G	D_{CCA}	D_{CKA}	TD_{soft}	TD_{hard}
Cifar-5	Cifar-10	-	-	-	0.7677	0.0866	0.2199	0.2812
		+	-	-	0.7552	0.0887	0.1692	0.2739
		-	+	-	0.7306	0.0758	0.1614	0.2260
		+	+	-	0.7285	0.0833	0.1593	0.2150
		-	-	+	0.7783	0.1310	0.1940	0.3107
		+	-	+	0.7668	0.1319	0.1900	0.2790
		-	+	+	0.7603	0.1370	0.2051	0.2773
		+	+	+	0.7500	0.1319	0.1976	0.2542

In the results, TD_{soft} and TD_{hard} share similar trends. The results show that both random flipping and random cropping have positive effects on TD robustness while the Gaussian additive noise decrease TD robustness. For TD_{hard}, random cropping has a more significant effect than random flipping. Applying random cropping can reduce the value of TD_{hard} from 0.2812 to 0.2260, while flipping only reduces it to 0.2739. For TD_{soft}, random flipping and random cropping have comparable effects.

On the other hand, both TD_{hard} and TD_{soft} increase when Gaussian noise is added. The results reveal that TD robustness matches previous understandings that flipping and cropping help learn a good representation while Gaussian noise may harm the generalization (Shorten & Khoshgoftaar, 2019; Chen et al., 2020).

Learning Rate Schedule Full results are reported in Table 10. All four metrics show that the models trained with the "Large LR + with decay" schedule capture more similar features than the models trained with the other two schedules. These observations coincide with the fact that initialing with a large learning rate and then decaying it in the middle of training helps improve the quality of the representation. A theoretical analysis of this phenomenon can be found in Li et al. (2019).

Table 10: The effect of learning rate schedule.

Upstream	Downstream	Schedule	D_{CCA}	D_{CKA}	TD_{soft}	TD_{hard}
Cifar-5	Cifar-10	Small LR + without decay	0.7739	0.1502	0.2118	0.2521
		Large LR + without decay	0.7417	0.2855	0.2470	0.2770
		Large LR + with decay	0.7264	0.1232	0.1896	0.2313

Adversarial training In Table 11, we report the results for standard training and adversarial training. When the training strategy changes from standard training to adversarial training, TD_{hard} decreases from 0.2139 to 0.1823, and TD_{soft} drops from 0.1617 to 0.0883. It can be concluded that although adversarial training lowers the classification accuracy of the model, it improves TD robustness and makes models trained from different initializations learn more similar features. This conclusion coincides with the widespread belief that adversarial training helps models capture features that align better with human perception (Tsipras et al., 2019; Ilyas et al., 2019; Engstrom et al., 2019).

Table 11: The difference between training with standard methods and adversarial training.

Upstream	Downstream	Training	D_{CCA}	D_{CKA}	TD_{soft}	TD_{hard}
Cifar-5	Cifar-10	Standard	0.6961	0.0835	0.1617	0.2139
		Adversarial	0.6339	0.0717	0.0883	0.1823

Batch Size The results on how batch size affects TD robustness are reported in Table 12. All four metrics verify that batch size 128 is the optimal setting for the default learning rate schedule. In practice, batch size 128 achieves the highest accuracy, and batch size 256 decreases the average accuracy by 1%.

Table 12: The effect of batch size.

Upstream	Downstream	Batch Size	D_{CCA}	D_{CKA}	TD_{soft}	TD_{hard}
Cifar-5	Cifar-10	32	0.7422	0.1484	0.2201	0.2858
		64	0.7244	0.1271	0.1772	0.2436
		128	0.6961	0.0835	0.1617	0.2139
		256	0.7054	0.0690	0.1704	0.2142
		512	0.7252	0.0744	0.1646	0.2214

Number of Training Epochs We plot TD robustness during training in Figure 3. The two red vertical lines indicate the epochs when the learning rate is decayed with factor 0.1. The plot discloses the properties of training that the performance curve does not manifest. For instance, after learning rate decay, all curves of four metrics drop significantly and then rise up again, indicating that the models are overfitting the training samples at a lower learning rate.

Architectures We also experiment with different architectures, including ResNet and VGG of different depths and widths, and report the results in Table 13. For ResNet models, when the depth

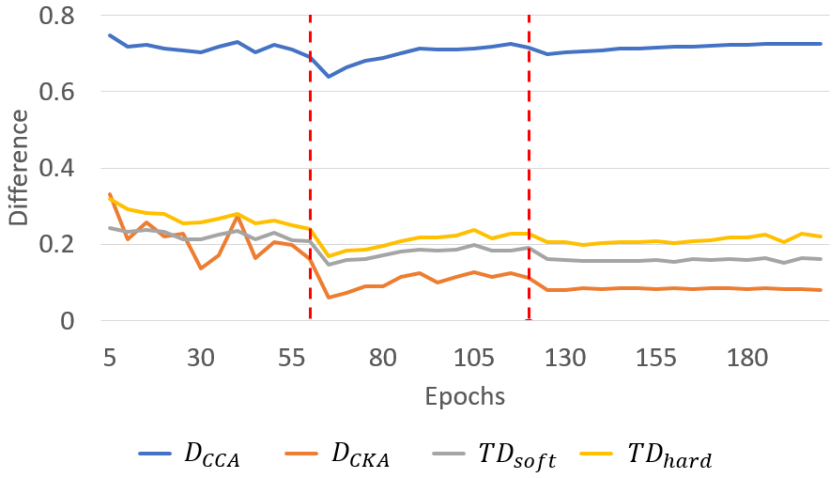

Figure 3: The changes in difference metrics during training.

increases from 20 to 110, TD_{hard} also decreases steadily, and the disagreement decreases from 22.53% to 19.76%. TD_{soft} also shares similar trends, but the minimum point occurs for ResNet56 models. D_{CCA} first goes up and then down from ResNet32 to ResNet 44 and to ResNet56. D_{CKA} indicates that ResNet56 models capture similar features than other models. In practice, it is also believed that increasing the depth is helpful for the quality of the representation and the model's generalization, aligning with the trends of TD_{hard}. In our experiments, when transferred to Cifar-10, the ResNet110 trained on Cifar-5 can reach 83% average accuracy while the number for ResNet20 is only 76%. The increased depth helps learn better features for transferring to Cifar-10.

As we use ResNet of different widths, both TD_{hard} and TD_{soft} decreases when the width increases. Simply doubling the width from 1xResNet32 to 2xResNet32 results in a significant reduction in TD_{hard}. Intuitively, when the width increases, the representation's dimension also increases, and more features are learned. In this case, two representations are more likely to have similar features. Although it is widely believed that the depth has a more significant impact on the performance than the width, our experiments show that the width may be more important for the similarity between two representations than the depth.

Table 13: The effect of model architectures.

Upstream	Downstream	Architecture	D_{CCA}	D_{CKA}	TD_{soft}	TD_{hard}
Cifar-5	Cifar-10	ResNet20	0.7414	0.0982	0.1617	0.2253
		ResNet32	0.6961	0.0835	0.1617	0.2139
		ResNet44	0.7023	0.0736	0.1583	0.2021
		ResNet56	0.6997	0.0710	0.1561	0.2015
		ResNet110	0.6887	0.0736	0.1627	0.1976
		1xResNet32	0.6961	0.0835	0.1617	0.2139
		2xResNet32	0.6917	0.0473	0.1315	0.1580
		5xResNet32	0.7038	0.0349	0.1082	0.1530
		10xResNet32	0.7125	0.0422	0.1285	0.1478
		VGG13	0.7411	0.0415	0.1297	0.3753
		VGG16	0.7767	0.0419	0.1408	0.4567
		VGG19	0.7974	0.0455	0.1179	0.3838

Upstream Tasks Using which upstream task to train the feature extractor is the core question in representation learning. In our experiment, we study four upstream tasks: C 4/10/20/100. We evaluate the results on Cifar-10 and Cifar-5 and report the results in Table 14. From C 4 to C 100, the upstream task becomes more difficult and contains more information. In practice, the models trained on harder

Table 14: The effect of different Upstream tasks

Upstream	Downstream	D_{CCA}	D_{CKA}	TD$_{soft}$	TD$_{hard}$
C 100	Cifar-10	0.7379	0.4352	0.3028	0.3340
C 20		0.8221	0.5287	0.3307	0.4362
C 10		0.8153	0.4750	0.3242	0.4386
C 4		0.8016	0.4002	0.3072	0.4953
C 100	Cifar-5	0.7379	0.4352	0.2124	0.2169
C 20		0.8221	0.5287	0.2511	0.2807
C 10		0.8153	0.4750	0.2472	0.2673
C 4		0.8016	0.4002	0.2541	0.3179

upstream tasks have higher accuracy on both the Cifar-5 and Cifar-10 downstream tasks. TD$_{soft}$, TD$_{hard}$, and D_{CCA} all show that the models trained on the C 100 task learn the most similar features in the table while D_{CKA} outputs the smallest number for models trained with C 4.