COMPARING NEURAL NETWORK REPRESENTATIONS WITH BAYESIAN PRIOR PREDICTIVE DISTRIBUTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

How best to compare internal representations of neural networks remains a highly relevant unresolved question in both machine learning and neuroscience, even though numerous similarity measures have been proposed. Here, we propose and extensively evaluate a new measure based on a Bayesian perspective on linear readouts called BayesCompare. BayesCompare compares the prior predictive distributions of linear readouts from the representations using metrics or divergences for probability distributions. We evaluate BayesCompare with a suite of experiments using 6 popular architectures trained on ImageNet-1k, comparing BayesCompare against a wide range of existing methods, including Representational Similarity Analysis (RSA) and Centered Kernel Alignment (CKA). First, we quantify each method's ability to retrieve the corresponding layer from 5 instances of the same architecture. All methods are able to retrieve layers to some extent, but some metrics within BayesCompare improve layer retrieval over existing methods. Second, we confirm that BayesCompare preserves the gradual change between representations throughout the network resulting from incremental operations of layers, to a similar extent as the existing methods. Third, we measure the methods' consistency under different random image sets and set sizes. All BayesCompare metrics provide highly reliable results even with only 50 input images, whereas RSA shows the lowest consistency. We also analyze the effects of rank transformation and assess the agreement between all metrics. BayesCompare metrics are closely aligned with each other, but distinct from existing methods, suggesting that BayesCompare captures a different aspect of representational similarity. Overall, these results establish BayesCompare as an effective way of measuring representational similarity while being easy and fast to compute and give strong guidance on which variants one should use. We provide a Python package to apply BayesCompare.

1 INTRODUCTION

Deep learning models are highly successful for machine learning applications and as models of brain processing, motivating efforts to better understand their internal processing. In particular, comparing internal representations to each other can help us understand the learning dynamics of networks [34], the effects of architectural or training decisions [41, 33, 36], network robustness [18, 6, 17], and enable training networks with knowledge distillation [56, 38]. Although various methods have been proposed to compare representations, the debate over which methods should be used is ongoing.

Here, we propose and extensively test BayesCompare, a Bayesian approach to compare neural network representations based on linear readouts [44]. Building upon [44], this work provides the full framework and extensive evaluation of the purely theoretical analysis. Linear readouts (or linear probes) are well-established and widely used tools in both machine learning [2, 5] and as encoding models in neuroscience [35]. BayesCompare applies a Bayesian treatment to this readout from the neural network, which results in a prior predictive distribution, which we can compare between representations. BayesCompare has a single free parameter controlling the signal-to-noise ratio, which ultimately determines the scale of the distances. Additionally, we can choose a metric or divergence for comparing the prior predictive distributions.

To test BayesCompare against existing methods for comparing DNN representations and to guide the choice of signal-to-noise ratio and comparison metric, we require criteria for what makes a good

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

comparison method. Here, we consider three such criteria: First, we evaluate *layer retrieval*, which is the most commonly used criterion in the literature [25, 40, 34, 10]. This criterion evaluates a method's ability to match layers between instances of the same architecture, trained with only minor variations, such as different random seeds or learning rates. Second, [23] proposes to evaluate *layer monotonicity*, which tests whether the incremental changes made by consecutive layers yield gradual changes in the tested dissimilarity. Third, going beyond the tests proposed in the literature, we test the *consistency* across image sets. This notion of consistency is similar to the reliability criterion for metrics in neuroscience, which measures how stable distances are across noisy observations [54]. However, our consistency measures whether a method yields similar results for different random sets of inputs and different set sizes.

In extensive tests on a range of six modern vision network architectures trained on ImageNet, we find that BayesCompare yields favorable results across all three criteria, while all existing methods perform substantially worse in at least one of the criteria. Additionally, we find that rank-transforming dissimilarities profoundly affects results, highlighting that monotone transformations of the dissimilarities are not benign. Our results demonstrate that BayesCompare offers a distinct perspective on comparisons between representations and improvements over the existing approaches.

2 Related Work

Many methods for comparing representations in deep neural networks (DNNs) have been proposed, which we can split into similarity-based methods and methods based on mappings between representations, even though there are important connections between these two approaches [8, 9, 14].

The classic similarity-based method from neuroscience is Representational Similarity Analysis (RSA), which computes a Representational Dissimilarity Matrix (RDM) and compares this matrix between representations [26]. RSA contains many variations which use different measures to compare the original activation patterns and the RDMs, respectively [54] and has recently been extended with more statistically efficient methods [9, 45]. The default method from machine learning is the kernel-based Centered Kernel Alignment (CKA) [25], of which the linear version is by far most commonly used. An additional classical measure of relatedness from information theory is the distance correlation [49]. To explicitly ignore the scale of distances in favor of a more topological view, the Jaccard coefficient between neighborhood graphs can be used to compare representations [55]. Finally, Normalized Bures Similarity (NBS) [51] is a dissimilarity-based method which is equivalent [14] to orthogonal Procrustes analysis [58], a fitting-based method.

The most classical fitting-based methods are based on canonical correlation analysis [13, 40, 34], which employs linear models to map between representations. Later methods aim to reduce the flexibility of the mapping, to better discriminate representations: Generalized shape metrics [58] measure the distance after one of the representations is transformed to minimize the distance. As acceptable transformations, the set of rotations is most commonly chosen to yield orthogonal Procrustes analysis. The SoftMatching distance goes even further and allows only a softened version of permutation to align representations [22]. Looking at differences between readouts instead, GULP provides another method for comparing representations [5], which also has a similarity-based equivalent when the readout is expressed using kernel methods.

Aiming to reduce the methods zoo, some recent efforts concentrated on benchmarking and analyzing comparison methods instead of proposing new methods [10, 7, 23, 3]. These comparisons are typically run with smaller networks that we use here, but reveal some consistent results: First, CCA-based metrics are suboptimal for comparing DNN representations [25, 23, 3], because methods that allow less flexible mappings generally perform better. In the most recent large-scale evaluations, Jaccard distance of neighborhood graphs emerged as the best-performing metric for vision models in [23].

3 BayesCompare Framework

Suppose that we are interested in the representation in a deep neural network layer with k neurons for a list of n stimuli. The activities of these neurons can be formed into a matrix $\mathbf{X} \in \mathbb{R}^{n \times k}$. A linear readout function f_β maps these activities $\mathbf{X}$ to a vector of scalar outputs for each stimulus $\mathbf{y}$ with the

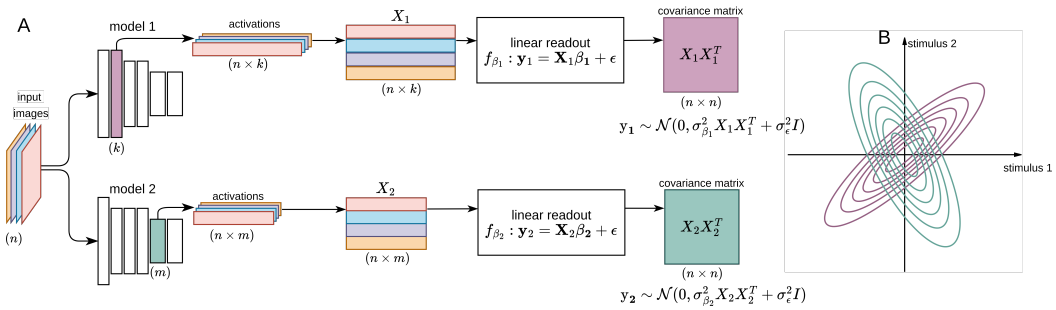

Figure 1: Illustration of the BayesCompare method for comparing representations from two neural networks. A) Predictive prior distributions are obtained from model activations, using a list of stimuli with n images. B) Example prior predictive distributions, $\mathbf{y}_1$ and $\mathbf{y}_2$, if $n = 2$ images were used to obtain the representations.

following relation:

$$\mathbf{y} = \mathbf{X}\beta + \epsilon \qquad \epsilon \sim \mathcal{N}(0, \sigma^2_\epsilon I), \tag{1}$$

where the noise ϵ is assumed to be iid. Gaussian. Adopting a Bayesian view on this linear readout, we define a zero-mean Gaussian prior over the weights $\beta \sim \mathcal{N}(0, \sigma^2_\beta I)$. With this prior choice, the linear readout function becomes ℓ_2 regularized. This ridge regression model predicts a Gaussian distribution for the observed outcomes,

$$\mathbf{y} \sim \mathcal{N}(0, \sigma^2_\beta \mathbf{X}\mathbf{X}^T + \sigma^2_\epsilon I). \tag{2}$$

We call this joint distribution over the outputs for a given list of inputs the prior predictive distribution. We can compute this prior predictive distribution for each representation, as shown in Figure 1. Then, to compare model representations, we compute the pairwise distance between layers' prior predictive distributions using a distance or divergence.

Metrics of BayesCompare. To compute the distance between the prior predictive distributions, we consider 6 metrics, where 4 are proper distances: Jensen-Shannon Distance, the square root of Jensen-Shannon Divergence (JSD)[48, 12], Total Variation Distance (TVD), 2-Wasserstein [19, 39] and Hellinger (Jeffreys-Matusita) [16, 37], while symmetrized Kullback-Leibler (KL) [27] and Bhattacharyya [4, 20] are divergences. See the Appendix A.2 for the definitions.

Parameters of Readout Function. There are two parameters in our ridge-regularized linear readout function to be set: σ^2_β and σ^2_ϵ. To make our method independent from the overall scale of the representations, we rescale the predictions such that the average variance of a single prediction is 1 and its total variance corresponds to the number of images n, i.e. such that $\mathrm{tr}\left(\sigma^2_\beta \mathbf{X}\mathbf{X}^T + \sigma^2_\epsilon I\right) = n$. This yields $\sigma^2_\beta = n(1 - \sigma^2_\epsilon)\left(\mathrm{tr}\left(\mathbf{X}\mathbf{X}^T\right)\right)^{-1}$.

With constant σ_ϵ, predictive distributions become more dissimilar as the number of images grows. This is not desirable, as we prefer finding similar values with different sample sizes and need to keep distances within an intermediate range so that bounded metrics like JSD, TVD, and Hellinger remain informative. Therefore, we increase the noise variance, σ^2_ϵ, to counteract this scale change due to the number of images. We set $\sigma^2_\epsilon = \frac{bn}{1+bn}$, introducing a signal-to-noise tradeoff parameter b, which determines the overall sensitivity of the analysis. This corresponds to a constant signal variance and a proportional relationship between noise variance and the number of images used after normalization. As measurement variance typically decreases inversely with the number of repeated measurements, this corresponds to the expected behavior if we planned measurements with a fixed budget distributed equally over the images. With these settings, the variance of the prior predictive distribution, $\Sigma_\mathbf{y}$ becomes:

$$\Sigma_\mathbf{y} = \frac{1}{1+bn} \frac{n\mathbf{X}\mathbf{X}^T}{\mathrm{tr}(\mathbf{X}\mathbf{X}^T)} + \frac{bn}{1+bn} I. \tag{3}$$

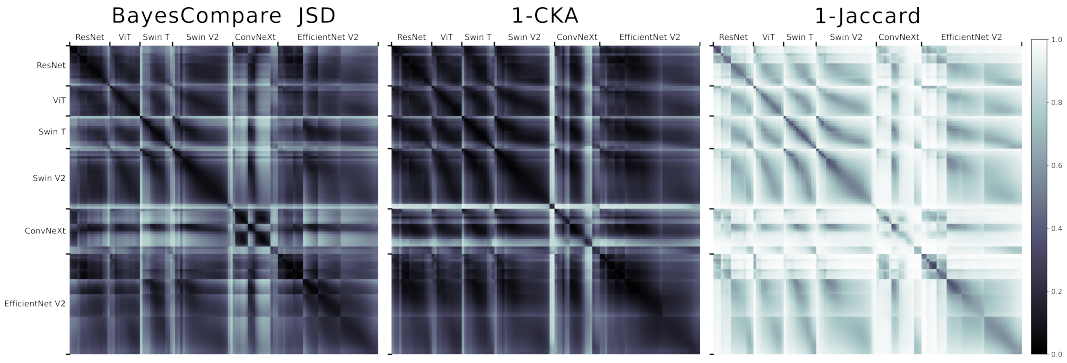

Figure 2: Averaged distance matrices for JSD (at $b = 0.01$), CKA and Jaccard over 5 model instances for all models, based on the 1000 images used in Experiment 1.

We explore how different values of b affect the performance of our metrics in finding similar representations in Experiment 1 and provide guidance on choosing this value (see Appendix A.6 Fig. 9 for example comparison matrices). The second experiment guides metric selection based on their ability to retain gradual representational changes across neighboring layers. In our third experiment, we investigate the effects of different n values on the consistency of the metrics. There, we confirm that the trade-off term provides a more stable outcome for our metrics, as shown in Figure 5.

Equivalent Representations. As we use divergences or distances to compare prior predictive distributions, representations are equivalent if and only if they induce the same prior predictive distribution. As all predictions are zero-mean Gaussian distributions, this is true if and only if the covariance matrices of the prior predictive distributions are equal, i.e. if and only if their trace-normalized covariance matrices are equal. Thus, BayesCompare is invariant to overall rescaling and to unitary transformations of the representations, see Appendix A.3. In contrast to existing methods like CKA or RSA, BayesCompare is not invariant to shifts of the representation, as the predicted variances for responses preserve the information how far each representation was away from the origin.

4 Evaluations and Results

4.1 Experimental Setup

In our experiments, we consider six widely used neural network architectures for image classification: ResNet-50 [15], Vision Transformer (ViT-B/16) [11], Swin Transformer Tiny [29], Swin Transformer V2 Small [30], ConvNeXt Tiny [31] and EfficientNet V2 Small [50]. We train 4 instances of each network on ImageNet-1k [43] with minor modifications in their base training recipes, such as different image augmentation policies or learning rates, in addition to different random initializations (see Table 2 in Appendix A.5 for details). An additional instance for each architecture is a pretrained model sourced from TorchVision (BSD-3-Clause License) [32]. Validation performance of the different instances is similar.

We compute distance matrices, which contain selected layers of all instances of all the models (see Appendix A.6), using BayesCompare and existing methods. For all experiments, we use randomly chosen unlabeled images from MS COCO (CC BY 4.0) [28] (unlabeled2017) as inputs for obtaining representations.

Comparison Methods We compare our methods against the following methods: *RSA* with an Euclidean distance RDM compared with a Spearman rank correlation; *Normalized Bures Similarity*, which is equivalent to orthogonal Procrustes analysis; linear *Centered Kernel Alignment*; *distance correlation* of Euclidean distances; *Jaccard* similarity, which performed best in recent vision benchmarks [23]; and a kernel-based implementation of *GULP*. We restrict ourselves to metrics that can be computed from kernel or distance matrices. This excludes primarily the CCA-based methods,

which are already known to perform poorly in DNN comparisons [25, 23] and the SoftMatching distance, which struggles with the high dimensionality of intermediate layer representations anyway. For methods that compute a similarity score, we evaluate 1 - similarity scores rather than the similarity scores themselves. In the Appendix A.1, we provide definitions of all metrics, as well as the hyperparameters used.

Figure 2 shows the distance matrices for JSD in BayesCompare, CKA and Jacard as examples, where the distances are averaged over the 5 instances for each architecture. It is evident that BayesCompare produces sensible results. In particular, both JSD and CKA show high similarities (equivalently small distances) for the self-similarity of models and a smooth decrease in similarity towards more distant layers. Also, both methods find some correspondences between architectures. However, as an example, the two metrics evaluate ConvNeXt representations differently. While BayesCompre finds most ConvNeXt layers to be highly distinct from all other architectures, CKA finds similarities for most ConvNeXt layers and other architectures. This result demonstrates that BayesCompare is substantially different from CKA. Jaccard, on the other hand, diverges from both metrics and finds much higher distances even between the corresponding layers of other instances (the diagonal).

4.2 EXPERIMENT 1: *Layer Retrieval*

Corresponding layers should be very similar across instances of the same network that differ only in slight changes in their training recipes (e.g., different initialization or augmentation policies). In particular, all other layers should be less similar, so that choosing the most similar layer retrieves the corresponding layer. We call the extent to which this is true the *layer retrieval ability* of a metric.

Evaluation Method. We quantify layer retrieval ability using the silhouette score (SS) [53, 42]. The silhouette score measures how well a class is clustered compared to the distance to other classes. It ranges from -1 to 1, with higher values indicating better clustering. Here, one class is the corresponding layers from the 5 instances of the same architecture, while the other class is all the other layers, including those of the same architecture.

The amount of additive noise on the covariance matrices determines the scale of the distances (see Section 3). Therefore, in this experiment, we sweep a wide range of values for the noise level parameter in Eq. 3: $b = [0.05, 0.02, 0.01, 0.005, 0.002]$. Since we use a randomly chosen 1000-image set to obtain representations in this experiment ($n = 1000$), these b values correspond to the noise variance values $\sigma_\epsilon^2 = [0.98, 0.95, 0.91, 0.83, 0.67]$, respectively.

Results. All tested metrics are able to discriminate corresponding layers from the other layers decently, evidenced by all silhouette scores being greater than or equal to 0.75, see Figure 3 A. However, KL divergence, Bhattacharyya and JSD measures yield particularly high scores, showing that BayesCompare can improve layer retrieval over existing methods. We confirm these results with statistical tests on different image subsets and find these differences between metrics are significant, as shown in Appendix A.7.

To observe the effects of noise level on discrimination performance of BayesCompare metrics, we compute the average of the silhouette score over all layers and models, per noise level (Fig. 3 B). Although KL divergence and Bhattacharyya show a slight decrease in the average silhouette score as the noise level decreases, they remain the best discriminating metrics. The average silhouette score decrease is more evident for JSD, TVD and Hellinger. This is because these metrics are bounded by 1. As noise decreases, self-distances start to grow while the distances to other layers cannot increase further, which decreases the silhouette score (see the effects of noise variance on the JSD distance matrix in the Appendix A.6 Fig. 9).

4.3 EXPERIMENT 2: *Preservation of Layer Monotonicity*

As layers in a neural network perform sequential operations, their representations typically change gradually, resulting in higher similarity between neighboring layers. Conversely, the similarity between layers decreases as the distance between them increases [23]. This phenomenon is called layer monotonicity, and an effective metric should accurately capture this gradual change. The same 1000-image random set as in Experiment 1 is used for this experiment.

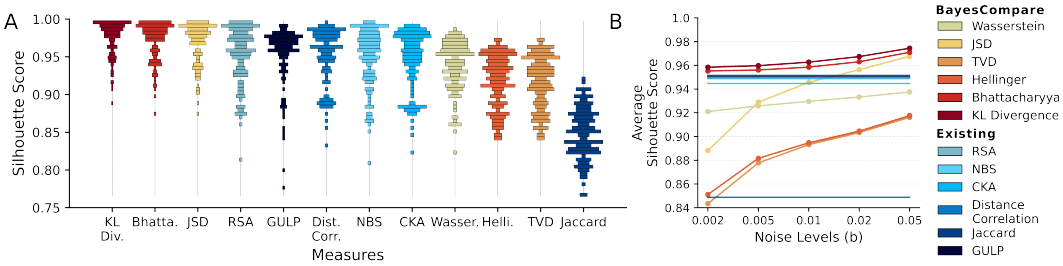

Figure 3: *Layer retrieval performance of metrics:* A: Silhouette score histogram for each measure aggregated over all models, shown for noise level $\sigma_\epsilon = 0.98$, each datapoint corresponds to a DNN layer, B: Average silhouette score over layers and models per noise level.

Evaluation Method. To measure the preservation of layer monotonicity, we compute Kendall's τ coefficient [21] between the distance of two layers in the network and their distance according to the different representational similarity measures. A value of $\tau = 1$ indicates perfect alignment between the vectors. For the physical distance between two layers in the network, we create a Toeplitz matrix of the same shape as the distance matrix, where the elements are pairwise absolute index differences. For the representational distance matrix, we use the self-similarity matrix of an architecture averaged over its instances. Then, we compute the τ_b statistic between the representational distance matrix and the physical distance matrix, per model architecture.

Results. Layer monotonicity varies strongly with the model architecture, and ConvNeXt has the lowest layer monotonicity score among all networks, for all metrics, as shown in Fig. 4. All distances within BayesCompare yield very similar results, matching existing methods as well, which sometimes score slightly higher and sometimes slightly lower. The only exception is GULP, which almost always scores worse. Overall, we can say that most metrics preserve the gradual representational change across layers for most networks.

4.4 Experiment 3: *Consistency across Image Sets and Sizes*

For results obtained with an image set to be generalizable to the population of all images, different random image sets should yield similar comparison results, ideally with as few images as possible. Furthermore, the number of images used should not systematically change results. To test these aspects, we compute representational similarities between the models using 100 randomly sampled MS COCO subsets per set size ($n = [50, 100, 200, 500, 1000]$), totaling 500 image sets and compare results between samples.

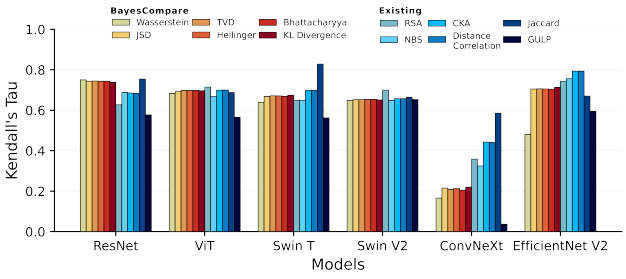

Figure 4: *Preservation of layer monotonicity:* Kendall's Tau-b is used for measuring how much each metric preserves the gradual change in representational similarity between neighboring layers in the network.

Evaluation Method. To assess consistency independent of the metric's scale, we use the intraclass correlation coefficient (ICC) [47, 24] for comparisons of distances between image sets. We also investigate how the mean and variance of the distances change with the number of images. For this, we compute the mean and variance for each element in the distance matrix over the random sets, for each set size. Then, we report the average of the mean and variance over the elements. Ideally, the mean should remain constant, while the variance should decrease as more images are used for computations.

Results. As expected, for all metrics, the consistency of distances increases with increasing number of images, as shown in Figure 5 A. Nevertheless, the effect of the number of images on different metrics differs in severity. All BayesCompare metrics yield highly reliable results even with a set size of $n = 50$, with only JSD slightly less consistent, perhaps due to sampling errors in the computation. GULP also shows highly consistent results throughout. CKA and Distance correlation perform considerably worse but stabilize for sets of 500 or 1000 images. Finally, Jaccard and RSA yield substantially less consistent results.

For most metrics, the mean distance changes little with the number of images, within $\pm 10\%$ of the value at 50 images, see Figure 5 C. However, GULP's average distance increases substantially with sample size. Similarly, the mean distance of distance correlation, CKA and NBS increases, although not as high as GULP. Among the subtly changing metrics, Jaccard and Wasserstein show a slight increase in mean with increasing sample size, while all other BayesCompare metrics show a small decrease in their mean. The metric that shows the smallest change across sample sizes is RSA.

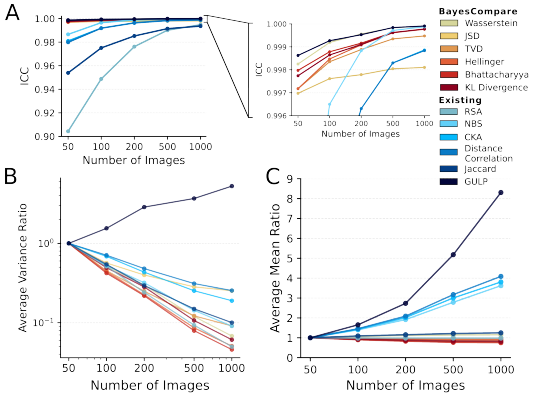

Figure 5: *Consistency across Image Sets and Sizes:* A: ICC scores as a measure of consistency across random sets of different sizes, B: Average variance of distances for different set sizes divided by the variance at $n = 50$, C: Average mean distance for different set sizes divided by the mean for $n = 50$.

All metrics, except GULP, show the expected behavior: the variance decreases with a larger sample size, as illustrated in Figure 5 B. The variance change with respect to $n = 50$ was at least 25% at $n = 1000$ for all other metrics. CKA, distance correlation and JSD showed the smallest change; Bhattacharyya, Hellinger and RSA showed the greatest change, relatively.

4.5 Effects of Rank Transformation

To diminish the effects of scale and boundedness, we apply a rank transformation to the distance matrices before subjecting them to our experiments. In this way, we could compare only the rank order that the metrics assign to each distance.

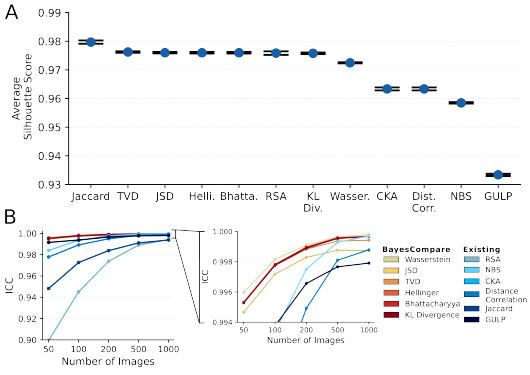

Figure 6: Results after rank transformation. A) Mean silhouette scores over 100 random image sets, with $n = 1000$ after rank transformation, error bars show standard deviation, and B) the ICC scores after rank transformation.

First, we evaluate the silhouette score for layer retrieval after rank transformation. The average silhouette scores of all metrics increase to more than 0.93, and the differences between metrics get smaller. To identify whether this small difference was meaningful, we computed average silhouette scores from the rank-transformed distance matrices of Experiment 3, per metric and randomized image set, using only the matrices with $n = 1000$. Then, we use the Wilcoxon signed-rank test to compare pairwise differences between the metrics' average silhouette scores. The pairwise differences among BayesCompare's JSD, Hellinger, and Bhattacharyya metrics are not significant. Additionally, there is no significant difference between KL divergence and RSA, nor between CKA and distance correlation. All other pairs of metrics differ significantly from each other ($p < 0.05$, Bonferroni corrected). These results show that Jaccard has the highest silhouette scores after rank transformation, closely followed by TVD. GULP performed the worst in terms of average silhouette scores after rank transformation.

Second, we repeat the ICC computation of Experiment 3 with rank-transformed distance matrices. Wasserstein performs the best for all set sizes, coming just ahead of Bhattacharyya, KL divergence

and TVD. All BayesCompare metrics score higher than existing metrics. JSD does not perform as well as other BayesCompare metrics for $n \geq 200$, probably due to numerical approximation errors becoming relevant. Among the existing metrics, NBS performs best, catching up with BayesCompare $n = 500$ or $n = 1000$. All other existing methods remain lower than the BayesCompare methods, except for JSD. Jaccard and RSA still show noticeably poorer consistency after rank transform.

4.6 Relations between Metrics

The high similarities in performance of metrics after rank transformation prompt us to investigate the (dis)agreement between metrics. To understand the relationship between metrics, we compute the rank correlation coefficient (Spearman's ρ_a) between each pair of metrics, using the distance matrices obtained for Experiment 1.

The rank correlation coefficients show that BayesCompare metrics yield highly similar results after rank transformation, as shown in Figure 7. Only Wasserstein gives slightly different results ($\rho_a \geq 0.94$). Among the existing metrics, we found that RSA is the most different from BayesCompare metrics, followed by Jaccard. Meanwhile, GULP is the most similar metric to the BayesCompare metrics. This result shows that BayesCompare metrics capture a different aspect of representational similarity from existing methods.

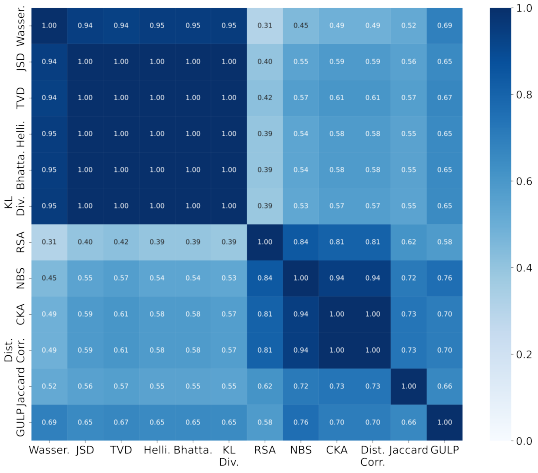

Figure 7: Rank correlation, Spearman's ρ_a, between metrics. Computed using the distance matrices of Experiment 1 for noise level $\sigma_\epsilon = 0.98$.

Among the existing methods, distance correlation and CKA turned out to be the same after rank transformation, indicating a non-linear relationship between these metrics. This finding is consistent with the known relation between these two metrics [46]. The similarities between the CKA-distance correlation pair, NBS and RSA are also high ($\rho_a \geq 0.81$). These high similarities between those four metrics confirm the established close relationships between them [14, 57]. On the other hand, Jaccard and GULP show a separation from this group of metrics.

These results are consistent with the findings in Section 4.5, where we found no significant differences between some of the metric pairs in terms of the average silhouette scores after rank transformation. The finding here that CKA and distance correlation are the same, up to a non-linear transform, explains the results in the previous section. Similarly, the finding of $\rho_a = 1$ between Bhattacharyya and Hellinger is consistent with the fact that they are related by a non-linear transform; see Appendix A.2.1. Table 1 summarizes the results of all experiments.

5 Discussion

We establish BayesCompare as an effective way of measuring representational similarity. Comparing the prior predictive distributions for linear readouts from DNN representations yields well-interpretable and quick-to-compute metrics. BayesCompare performs better than all other tested methods at layer retrieval, yields highly stable results across random image choices—even with small set sizes—and matches the order of layers well, as we confirm in extensive experiments with various ImageNet trained networks.

Of the other comparison methods, Jaccard distance is the only one that clearly beats our BayesCompare methods in any experiment. It yields a higher silhouette score after rank transformation and a higher rank correlation with the distance in the network. These effects strongly rely on the use of a rank transform, though, as Jaccard distance considers layers to be quite different from themselves across instances and thus yields very bad silhouette scores without the rank transform. Also, Jaccard distance estimates are quite variable across image samples and Jaccard distance is not differentiable as it relies on a hard neighborhood classification. Thus, the range of practical situations for which

Jaccard distance is preferable is quite limited. None of the other methods has a clear advantage over BayesCompare and each has some disadvantage: RSA yields comparable silhouette scores, but substantially higher variance of results across image samples. NBS, CKA, and distance correlation yield slightly worse results across the board. Finally, GULP yields inadequate layer recovery after rank transform and yields increasing values with higher variance if more images are used for evaluation, which impairs interpretability.

Rank transforming distances before running evaluations turned out to be quite informative, even though rank transforming distances is rarely useful for applications, where we typically want our distances to be continuous values that are independent of the other representations we compare to and ideally form a proper metric. Without the rank transformation, final non-linearities on the distances have strong effects on performance metrics. For example, Hellinger and Bhattacharyya distances yield substantially different results, while they can be transformed into each other exactly. Also, computing rank correlations to investigate the relations between metrics revealed the high similarities among BayesCompare methods and the non-linear relationship between CKA and distance correlation.

Our evaluations also provide guidance on which variants of BayesCompare one should use. Wasserstein is the only metric that yields a substantially different ranking of the distances, but performs worse than the others and should thus usually be avoided. For applications where the numerical size of the distance values matters, KL divergence and Bhattacharyya with the highest level of noise perform best ($b = 0.05$, corresponding to $\sigma_\epsilon = 0.98$ for $n = 1000$). If a true metric is desired, JSD provides a similarly good alternative. To aid future applications of the BayesCompare methods, we release a Python package (MIT License) that computes our methods efficiently for representations from neural networks implemented in PyTorch.

Limitations. Although we use a diverse set of vision network architectures, we consider only models trained with supervised learning for an object classification task. Furthermore, we used a set of selected layers from the architectures to construct a reasonably sized distance matrix. BayesCompare can be used in any setup where a linear probe can be used on the internal representations. Therefore, we expect these results to generalize to networks trained for different tasks, using different training methods, and with different layers, but this remains untested.

Future Directions. For establishing BayesCompare as a representational similarity metric, we kept the prior for our weights as a simple zero-mean Gaussian. A potential avenue for future work is to investigate the use of alternative priors for the weights and exploration of different readout functions. With BayesCompare, it is possible to perform a wide range of layer-specific analyses of networks, a topic that was rather overlooked in the literature. A direction for future work is to investigate commonalities between similar layer types in different architectures and assess the performance of representations in terms of predicting the biological neural activations.

Table 1: Summary of experimental results and evaluations. 'r.t.' stands for rank transformation. For Experiment 3, mean change and variance change are reported for $n = 1000$ with respect to $n = 50$ in terms of percent change.

Metric	EXP1 (avg. SS)	EXP1 (avg. SS - after r.t.)	EXP2 (avg. τ)	EXP3 (avg. ICC)	EXP3 (change of mean %)	EXP3 (var. change %)	EXP3 (avg. ICC - after r.t.)
Wasserstein	.9296	.9725	.5617	.9993	15.1	-93.2	**.9985**
JSD	.9456	.9760	.6130	.9977	-21.6	-74.7	.9975
TVD	.8931	.9762	.6137	.9986	-10.9	-90.8	.9981
Hellinger	.8947	.9760	.6142	.9988	-8.36	-94.9	.9983
Bhattacharyya	.9584	.9760	.6125	.9990	-24.7	**-95.4**	.9983
KL Divergence	**.9626**	.9758	.6157	.9989	-17.8	-93.9	.9982
RSA	.9519	.9759	.6325	.9628	**-1.4**	-95.0	.9603
NBS	.9486	.9585	.6221	.9963	262.1	-90.9	.9948
CKA	.9447	.9633	.6627	.9933	280.7	-81.1	.9918
Distance Correlation	.9495	.9633	.6621	.9930	308.9	-74.6	.9918
Jaccard	.8480	**.9797**	**.6984**	.9798	25.0	-90.0	.9779
GULP	.9510	.9333	.4983	**.9994**	730.7	427.0	.9955

AI USE STATEMENT

We have not used generative AI tools for any of the required disclosure tasks.

Additionally, we used generative AI tools to polish writing, to identify relevant literature and to create or edit software codes. We have reviewed all AI-assisted work. For instance, we verified all LLM-generated code, checked the relevant literature suggestions provided by LLMs, and processed the writing suggestions before including them in the main text. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Ethics statement

This study uses only openly available datasets (ImageNet1k and MSCOCO) and is in compliance with the ICLR Code of Ethics in every respect.

Reproducibility statement

We explain in detail our experimental setup and the models we trained and used in Section 4, Evaluations and Results and in Appendix A.4 Table 2.

We provide the code for our model training and experiments, along with data specifications, in the supplemental materials, to enable replication of the results. Also, BayesCompare is available as an open-source Python library.

References

[1] Beatrice Acciaio, Daniel Bartl, Anne Grass, Songyan Hou, and Gudmund Pammer. The geometry of the adapted bures–wasserstein space. *arXiv preprint arXiv:2602.00623*, 2026.

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

[3] Itamar Avitan and Tal Golan. Model–behavior alignment under flexible evaluation: When the best-fitting model isn't the right one. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=kE4XEY7Bbc.

[4] Anil Bhattacharyya. On a measure of divergence between two statistical populations defined by their probability distribution. *Bulletin of the Calcutta Mathematical Society*, 35:99–110, 1943.

[5] Enric Boix-Adsera, Hannah Lawrence, George Stepaniants, and Philippe Rigollet. Gulp: a prediction-based metric between representations. *Advances in Neural Information Processing Systems*, 35:7115–7127, 2022.

[6] Christian Cianfarani, Arjun Nitin Bhagoji, Vikash Sehwag, Ben Zhao, Heather Zheng, and Prateek Mittal. Understanding robust learning through the lens of representation similarities. *Advances in Neural Information Processing Systems*, 35:34912–34925, 2022.

[7] Nathan Cloos, Guangyu Robert Yang, and Christopher J Cueva. A framework for standardizing similarity measures in a rapidly evolving field. *arXiv preprint arXiv:2409.18333*, 2024.

[8] Jörn Diedrichsen and Nikolaus Kriegeskorte. Representational models: A common framework for understanding encoding, pattern-component, and representational-similarity analysis. *PLOS Computational Biology*, 13(4):e1005508, April 2017. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1005508. URL http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1005508.

[9] Jörn Diedrichsen, Eva Berlot, Marieke Mur, Heiko H. Schütt, Mahdiyar Shahbazi, and Nikolaus Kriegeskorte. Comparing representational geometries using whitened unbiased-distance-matrix similarity. *Neurons, Behavior, Data analysis, and Theory*, 5(3), 2021. doi: 10.51628/001c.27664. URL https://doi.org/10.51628/001c.27664. Number: 3.

[10] Frances Ding, Jean-Stanislas Denain, and Jacob Steinhardt. Grounding representation similarity through statistical testing. *Advances in Neural Information Processing Systems*, 34:1556–1568, 2021.

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[12] D.M. Endres and J.E. Schindelin. A new metric for probability distributions. *IEEE Transactions on Information Theory*, 49(7):1858–1860, July 2003. ISSN 0018-9448. doi: 10.1109/TIT.2003.813506. URL http://ieeexplore.ieee.org/document/1207388/. Number: 7.

[13] David R. Hardoon, Sandor Szedmak, and John Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural Computation*, 16(12):2639–2664, 2004. doi: 10.1162/0899766042321814.

[14] Sarah E Harvey, Brett W Larsen, and Alex H Williams. Duality of bures and shape distances with implications for comparing neural representations. In *Proceedings of UniReps: the First Workshop on Unifying Representations in Neural Models*, pp. 11–26. PMLR, 2024.

[15] 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.

[16] Ernst Hellinger. Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen. *Journal für die reine und angewandte Mathematik*, 1909(136):210–271, 1909.

[17] Blaine Hoak, Kunyang Li, and Patrick McDaniel. Alignment and adversarial robustness: Are more human-like models more secure? *arXiv preprint arXiv:2502.12377*, 2025.

[18] Haydn T Jones, Jacob M Springer, Garrett T Kenyon, and Juston S Moore. If you've trained one you've trained them all: inter-architecture similarity increases with robustness. In *Uncertainty in Artificial Intelligence*, pp. 928–937. PMLR, 2022.

[19] Leonid V Kantorovich. Mathematical methods of organizing and planning production. *Management science*, 6(4):366–422, 1960.

[20] Ravi Kashyap. The perfect marriage and much more: Combining dimension reduction, distance measures and covariance. *Physica A: Statistical Mechanics and its Applications*, 536:120938, 2019.

[21] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1-2):81–93, 1938.

[22] Meenakshi Khosla and Alex H Williams. Soft matching distance: A metric on neural representations that captures single-neuron tuning. In *Proceedings of UniReps: the First Workshop on Unifying Representations in Neural Models*, pp. 326–341. PMLR, 2024.

[23] Max Klabunde, Tassilo Wald, Tobias Schumacher, Klaus Maier-Hein, Markus Strohmaier, and Florian Lemmerich. Resi: A comprehensive benchmark for representational similarity measures. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=PRvdO3nfFi.

[24] Terry K Koo and Mae Y Li. A guideline of selecting and reporting intraclass correlation coefficients for reliability research. *Journal of chiropractic medicine*, 15(2):155–163, 2016.

[25] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International conference on machine learning*, pp. 3519–3529. PMlR, 2019.

[26] Nikolaus Kriegeskorte, Marieke Mur, and Peter A Bandettini. Representational similarity analysis-connecting the branches of systems neuroscience. *Frontiers in systems neuroscience*, 2:249, 2008.

[27] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[28] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer, 2014.

[29] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021.

[30] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12009–12019, 2022.

[31] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11976–11986, 2022.

[32] TorchVision maintainers and contributors. Torchvision: Pytorch's computer vision library. `https://github.com/pytorch/vision`, 2016.

[33] Johannes Mehrer, Nikolaus Kriegeskorte, and Tim Kietzmann. Beware of the beginnings: intermediate and higherlevel representations in deep neural networks are strongly affected by weight initialization. In *Conference on Cognitive Computational Neuroscience*, 2018.

[34] Ari Morcos, Maithra Raghu, and Samy Bengio. Insights on representational similarity in neural networks with canonical correlation. *Advances in neural information processing systems*, 31, 2018.

[35] Thomas Naselaris, Kendrick N Kay, Shinji Nishimoto, and Jack L Gallant. Encoding and decoding in fmri. *Neuroimage*, 56(2):400–410, 2011.

[36] Thao Nguyen, Maithra Raghu, and Simon Kornblith. Do wide and deep networks learn the same things? uncovering how neural network representations vary with width and depth. *arXiv preprint arXiv:2010.15327*, 2020.

[37] Leandro Pardo. *Statistical inference based on divergence measures*. Chapman and Hall/CRC, 2018.

[38] Nikolaos Passalis and Anastasios Tefas. Learning deep representations with probabilistic knowledge transfer. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 268–284, 2018.

[39] Gabriel Peyré and Marco Cuturi. *Computational optimal transport: With applications to data science*. Now Foundations and Trends, 2019.

[40] Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Narain Sohl-Dickstein. Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. In *Neural Information Processing Systems*, 2017. URL `https://api.semanticscholar.org/CorpusID:23890457`.

[41] Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? *Advances in neural information processing systems*, 34:12116–12128, 2021.

[42] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

[43] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

[44] Heiko H. Schütt. Bayesian comparisons between representations. In *8th Annual Conference on Cognitive Computational Neuroscience*, 2025. URL https://openreview.net/forum?id=BEzJu8VsJ6.

[45] Heiko H Schütt, Alexander D Kipnis, Jörn Diedrichsen, and Nikolaus Kriegeskorte. Statistical inference on representational geometries. *eLife*, 12:e82566, 2023. ISSN 2050-084X. doi: 10.7554/eLife.82566. URL https://doi.org/10.7554/eLife.82566.

[46] Dino Sejdinovic, Bharath Sriperumbudur, Arthur Gretton, and Kenji Fukumizu. Equivalence of distance-based and rkhs-based statistics in hypothesis testing. *The annals of statistics*, pp. 2263–2291, 2013.

[47] Patrick E Shrout and Joseph L Fleiss. Intraclass correlations: uses in assessing rater reliability. *Psychological bulletin*, 86(2):420, 1979.

[48] Robin Sibson. Information radius. *Zeitschrift für Wahrscheinlichkeitstheorie und Verwandte Gebiete*, 14:149–160, 1969. URL https://api.semanticscholar.org/CorpusID:198179065.

[49] Gábor J. Székely, Maria L. Rizzo, and Nail K. Bakirov. Measuring and testing dependence by correlation of distances. *The Annals of Statistics*, 35(6):2769 – 2794, 2007.

[50] Mingxing Tan and Quoc Le. EfficientNetV2: Smaller models and faster training. In *International conference on machine learning*, pp. 10096–10106. PMLR, 2021.

[51] Shuai Tang, Wesley J Maddox, Charlie Dickens, Tom Diethe, and Andreas Damianou. Similarity of neural networks with gradients. *arXiv preprint arXiv:2003.11498*, 2020.

[52] JohnMark Taylor and Nikolaus Kriegeskorte. Extracting and visualizing hidden activations and computational graphs of pytorch models with torchlens. *Scientific Reports*, 13(1):14375, 2023.

[53] Imran Thobani, Javier Sagastuy-Brena, Aran Nayebi, Rosa Cao, and Daniel LK Yamins. Inter-animal transforms as a guide to model-brain comparison. In *ICLR 2024 Workshop on Representational Alignment*, 2024.

[54] Alexander Walther, Hamed Nili, Naveed Ejaz, Arjen Alink, Nikolaus Kriegeskorte, and Jörn Diedrichsen. Reliability of dissimilarity measures for multi-voxel pattern analysis. *NeuroImage*, 137:188–200, August 2016. ISSN 10538119. doi: 10.1016/j.neuroimage.2015.12.012. URL https://linkinghub.elsevier.com/retrieve/pii/S1053811915011258.

[55] Chenxu Wang, Wei Rao, Wenna Guo, Pinghui Wang, Jun Liu, and Xiaohong Guan. Towards understanding the instability of network embedding. *IEEE Transactions on Knowledge and Data Engineering*, 34(2):927–941, 2022.

[56] Lin Wang and Kuk-Jin Yoon. Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks. *IEEE transactions on pattern analysis and machine intelligence*, 44(6):3048–3068, 2021.

[57] Alex H Williams. Equivalence between representational similarity analysis, centered kernel alignment, and canonical correlations analysis. In *UniReps: 2nd Edition of the Workshop on Unifying Representations in Neural Models*, 2024. URL https://openreview.net/forum?id=zMdnnFasgC.

[58] Alex H Williams, Erin Kunz, Simon Kornblith, and Scott Linderman. Generalized shape metrics on neural representations. *Advances in neural information processing systems*, 34:4738–4750, 2021.

A Appendix

A.1 Representational Similarity Measures

The activities of a DNN layer with k neurons for a list of n stimuli are formed into a matrix $\mathbf{X} \in \mathbb{R}^{n \times k}$. Then, the kernel matrix for this layer and the stimulus set is defined as $\mathbf{K_x} = \mathbf{X}\mathbf{X}^T$. Based on this, the double-centered kernel matrix is:

$$\bar{\mathbf{K}}_\mathbf{x} = \mathbf{C}\mathbf{X}\mathbf{X}^T\mathbf{C}, \quad \text{where} \quad \mathbf{C} = \mathbf{I}_n - \frac{1}{n}\mathbf{1}\mathbf{1}^T. \tag{4}$$

Here, $\mathbf{I}_n$ is the $n \times n$ identity matrix and $\mathbf{1}\mathbf{1}^T$ is an $n \times n$ matrix of ones.

The following sections are the implementations of representational similarity measures between two hypothetical layers l_x and l_y. The kernel matrices for layer l_x are defined above. Similarly, for the layer l_y with m neurons and for the same list of stimuli n, the activation matrix is $\mathbf{Y} \in \mathbb{R}^{n \times m}$ and the kernel matrix is $\mathbf{K_y} = \mathbf{Y}\mathbf{Y}^T$. The double-centered kernel matrix for layer l_y follows the same relations shown in Eq. 4 and will be denoted as $\bar{\mathbf{K}}_\mathbf{y}$.

Representational Similarity Analysis (RSA). [26] We used RSA with Euclidean distance for inner similarity and Spearman's rank correlation for outer similarity. First, the squared Euclidean distance matrix from each kernel matrix is obtained with the following relation, shown only for $\mathbf{K_x}$:

$$\mathbf{D}_{\mathbf{x}ij}^2 = \mathbf{K}_{\mathbf{x}ii} + \mathbf{K}_{\mathbf{x}jj} - 2\mathbf{K}_{\mathbf{x}ij} \tag{5}$$

The upper-triangular entries of the Euclidean distance matrix are rank-transformed and put into vectors $\mathbf{d_x}$ and $\mathbf{d_y}$,

$$\mathbf{d_x} = \operatorname{rank}\left[(\mathbf{D}_{\mathbf{x}ij})_{i<j}\right], \quad \mathbf{d_y} = \operatorname{rank}\left[(\mathbf{D}_{\mathbf{y}ij})_{i<j}\right] \tag{6}$$

Then, Spearman's rank correlation (ρ_a) is computed between the Euclidean distance vectors, with the following:

$$\rho_a(\mathbf{d_x}, \mathbf{d_y}) = \frac{12\tilde{\mathbf{d}}_\mathbf{x}^T\tilde{\mathbf{d}}_\mathbf{y}}{r^3 - r}, \tag{7}$$

where $\tilde{\mathbf{d}}_\mathbf{x} = \mathbf{d_x} - \frac{1}{r}\sum_{i=0}^{r} i$ and $\tilde{\mathbf{d}}_\mathbf{y} = \mathbf{d_y} - \frac{1}{r}\sum_{i=0}^{r} i$ and r is the dimension of the vectors.

Normalized Bures Similarity (NBS). [51] Normalized Bures Similarity is computed with the double-centered kernel matrices as follows,

$$s_{\mathrm{NBS}}(\bar{\mathbf{K}}_\mathbf{x}, \bar{\mathbf{K}}_\mathbf{y}) = \frac{\operatorname{tr}\left[\left(\bar{\mathbf{K}}_\mathbf{x}^{1/2}\bar{\mathbf{K}}_\mathbf{y}\bar{\mathbf{K}}_\mathbf{x}^{1/2}\right)^{1/2}\right]}{\sqrt{\operatorname{tr}(\bar{\mathbf{K}}_\mathbf{x})\operatorname{tr}(\bar{\mathbf{K}}_\mathbf{y})}}. \tag{8}$$

For a more stable numerical computation, we implemented the following equivalent expression:

$$s_{\mathrm{NBS}}(\bar{\mathbf{K}}'_\mathbf{x}, \bar{\mathbf{K}}'_\mathbf{y}) = \operatorname{tr}\left[\left((\bar{\mathbf{K}}'_\mathbf{x})^{1/2}\bar{\mathbf{K}}'_\mathbf{y}(\bar{\mathbf{K}}'_\mathbf{x})^{1/2}\right)^{1/2}\right], \tag{9}$$

where the kernel matrices are trace normalized beforehand: $\bar{\mathbf{K}}'_\mathbf{x} = \frac{\bar{\mathbf{K}}_\mathbf{x}}{\operatorname{tr}(\bar{\mathbf{K}}_\mathbf{x})}$ and $\bar{\mathbf{K}}'_\mathbf{y} = \frac{\bar{\mathbf{K}}_\mathbf{y}}{\operatorname{tr}(\bar{\mathbf{K}}_\mathbf{y})}$.

Centered Kernel Alignment (CKA). [25] CKA computes the similarity between two double-centered kernel matrices as follows,

$$s_{\mathrm{CKA}}(\bar{\mathbf{K}}_\mathbf{x}, \bar{\mathbf{K}}_\mathbf{y}) = \frac{\operatorname{tr}(\bar{\mathbf{K}}_\mathbf{x}\bar{\mathbf{K}}_\mathbf{y})}{\sqrt{\operatorname{tr}(\bar{\mathbf{K}}_\mathbf{x}^2)\operatorname{tr}(\bar{\mathbf{K}}_\mathbf{y}^2)}}. \tag{10}$$

Distance Correlation. [49] Distance correlation is computed by:

$$s_{\text{DistCorr}}(\bar{\mathbf{K}}_{\mathbf{x}}, \bar{\mathbf{K}}_{\mathbf{y}}) = \frac{\text{dCov}^2(\bar{\mathbf{K}}_{\mathbf{x}}, \bar{\mathbf{K}}_{\mathbf{y}})}{\sqrt{\text{dCov}^2(\bar{\mathbf{K}}_{\mathbf{x}}, \bar{\mathbf{K}}_{\mathbf{x}})\,\text{dCov}^2(\bar{\mathbf{K}}_{\mathbf{y}}, \bar{\mathbf{K}}_{\mathbf{y}})}}, \tag{11}$$

where the distance covariance is defined as,

$$\text{dCov}^2(\bar{\mathbf{K}}_{\mathbf{x}}, \bar{\mathbf{K}}_{\mathbf{y}}) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} \bar{\mathbf{K}}_{\mathbf{x}ij}\bar{\mathbf{K}}_{\mathbf{y}ij}. \tag{12}$$

Note that distance correlation computed with double-centered kernel matrices is equivalent to its computation with double-centered Euclidean distance matrices.

GULP. [5] For the GULP under kernel ridge-regression, we followed the proof in [5] that shows the equivalent expression for $d_\lambda(\mathbf{X}, \mathbf{Y})$ in terms of kernel functions associated with $\mathbf{X}$ and $\mathbf{Y}$.

$$\begin{aligned} d_\lambda(\mathbf{X}, \mathbf{Y})^2 = &\operatorname{tr}\left((\mathbf{K}_{\mathbf{x}} + \lambda\mathbf{I})^{-1}\mathbf{K}_{\mathbf{x}}(\mathbf{K}_{\mathbf{x}} + \lambda\mathbf{I})^{-1}\mathbf{K}_{\mathbf{x}}\right) + \\ &\operatorname{tr}\left((\mathbf{K}_{\mathbf{y}} + \lambda\mathbf{I})^{-1}\mathbf{K}_{\mathbf{y}}(\mathbf{K}_{\mathbf{y}} + \lambda\mathbf{I})^{-1}\mathbf{K}_{\mathbf{y}}\right) \\ &- 2\operatorname{tr}\left((\mathbf{K}_{\mathbf{x}} + \lambda\mathbf{I})^{-1}\mathbf{K}_{\mathbf{x}}(\mathbf{K}_{\mathbf{y}} + \lambda\mathbf{I})^{-1}\mathbf{K}_{\mathbf{y}}\right). \end{aligned} \tag{13}$$

In our implementations, we set $\lambda = 0.01$.

Jaccard Similarity. [55] Jaccard is a neighborhood-based measure. For each pair of elements in the activation matrix, it computes the ratio between the number of shared k nearest neighbors (k-NN) (the intersection) and the total number of unique neighbors in both sets (the union). We used cosine similarity between elements for finding the k-NNs, obtained from the kernel matrices as follows,

$$\mathbf{D}_{\mathbf{x}ij}^{\text{CosSim}} = \frac{\mathbf{K}_{\mathbf{x}ij}}{\sqrt{\mathbf{K}_{\mathbf{x}ii}\mathbf{K}_{\mathbf{x}jj}}}. \tag{14}$$

Then, Jaccard is computed for a stimulus i, with the k-NNs of $\mathbf{X}$ and $\mathbf{Y}$, $\mathcal{N}_k^{\mathbf{X}}(i)$ and $\mathcal{N}_k^{\mathbf{Y}}(i)$, respectively.

$$s_{\text{Jaccard}}(\mathbf{K}_{\mathbf{x}}, \mathbf{K}_{\mathbf{y}}, i) = \frac{|\mathcal{N}_k^{\mathbf{X}}(i) \cap \mathcal{N}_k^{\mathbf{Y}}(i)|}{|\mathcal{N}_k^{\mathbf{X}}(i) \cup \mathcal{N}_k^{\mathbf{Y}}(i)|}. \tag{15}$$

We set $k = 10$, in our implementations.

A.2 BayesCompare Framework

To keep the notation consistent with the main text, we will denote the covariances of the prior predictive distributions for two hypothetical layers we are comparing as $\mathbf{\Sigma}_1$ and $\mathbf{\Sigma}_2$. For these layers, the activation matrices are $\mathbf{X}_1 \in \mathbb{R}^{n\times k}$ and $\mathbf{X}_2 \in \mathbb{R}^{n\times m}$

$$\tilde{\mathbf{K}}_{\mathbf{1}} = \mathbf{X}_1\tilde{\mathbf{C}}_{\mathbf{1}}\mathbf{X}_1^T \quad \text{where} \quad \tilde{\mathbf{C}}_{\mathbf{1}} = \mathbf{I}_k - \frac{1}{k}\mathbf{1}\mathbf{1}^T \in \mathbb{R}^{k\times k} \tag{16}$$

$$\tilde{\mathbf{K}}_{\mathbf{2}} = \mathbf{X}_2\tilde{\mathbf{C}}_{\mathbf{2}}\mathbf{X}_2^T \quad \text{where} \quad \tilde{\mathbf{C}}_{\mathbf{2}} = \mathbf{I}_m - \frac{1}{m}\mathbf{1}\mathbf{1}^T \in \mathbb{R}^{m\times m} \tag{17}$$

Here, normalization matrices $\tilde{\mathbf{C}}_{\mathbf{1}}$ and $\tilde{\mathbf{C}}_{\mathbf{2}}$ remove the mean across features from the activation matrices. In the following sections, the definitions of divergences are given for general normal distributions with non-zero means and variances. However, in our implementations, we set the mean

terms to zero and use the trace-normalized and noise-added covariance matrices indicated with $\mathbf{\Sigma}$ that are obtained from the kernel matrices with the following,

$$\mathbf{\Sigma}_1 = \frac{1}{1+bn}\frac{n\tilde{\mathbf{K}}_\mathbf{1}}{\mathrm{tr}(\tilde{\mathbf{K}}_\mathbf{1})} + \frac{bn}{1+bn}\mathbf{I}_n \quad \text{and} \quad \mathbf{\Sigma}_2 = \frac{1}{1+bn}\frac{n\tilde{\mathbf{K}}_\mathbf{2}}{\mathrm{tr}(\tilde{\mathbf{K}}_\mathbf{2})} + \frac{bn}{1+bn}\mathbf{I}_n \tag{18}$$

A.2.1 DISTANCES

Jensen Shannon Divergence. [48, 12] JSD corresponds to mutual information between samples and their generating distribution labels. The approximation used for the implementation is:

$$d^2_{\text{JSD}}(P_1, P_2) \approx \frac{1}{K}\sum_{i=1}^{K}\log_2\frac{p_1(x_i^{(1)})}{p_1(x_i^{(1)}) + p_2(x_i^{(1)})} + \frac{1}{K}\sum_{i=1}^{K}\log_2\frac{p_2(x_i^{(2)})}{p_1(x_i^{(2)}) + p_2(x_i^{(2)})} - 1, \tag{19}$$

where $x_i^{(1)}$ and $x_i^{(2)}$ are samples of the two distributions. For all evaluations the square root of the Jensen Shannon Divergence, which is a metric and therefore sometimes called Jensen Shannon Distance.

Total Variation Distance. Total Variation Distance (TVD) is defined as the largest absolute difference between the probabilities that two distributions assign to any given event. The sampled estimation used in the implementation for BayesCompare is:

$$d_{\text{TVD}}(P_1, P_2) \approx \frac{1}{2K}\sum_{i=1}^{K}\max\left(0, 1 - \frac{p_2(x_i^{(1)})}{p_1(x_i^{(1)})}\right) + \frac{1}{2K}\sum_{i=1}^{K}\max\left(0, 1 - \frac{p_1(x_i^{(2)})}{p_2(x_i^{(2)})}\right). \tag{20}$$

For both TVD and JSD, we set the sample size $K = 10000$.

2-Wasserstein. [19, 39]The 2-Wasserstein distance measures the minimum cost required to transport one probability distribution into another. When computed between two multivariate Gaussians, $P_1 \sim \mathcal{N}(\mu_1, \Sigma_1)$ and $P_2 \sim \mathcal{N}(\mu_2, \Sigma_2)$, it has a closed-form solution as follows:

$$d^2_{\text{Wasserstein}}(P_1, P_2) = \|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|_2^2 + \mathrm{tr}\left(\mathbf{\Sigma}_1 + \mathbf{\Sigma}_2 - 2\left(\mathbf{\Sigma}_1^{1/2}\mathbf{\Sigma}_2\mathbf{\Sigma}_1^{1/2}\right)^{1/2}\right). \tag{21}$$

It is known that the Bures distance and the 2-Wasserstein distance of zero-mean Gaussians are identical, simply because the formulae to compute them are [1]. However, BayesCompare only trace-normalizes the Gram matrices, whereas methods we compare from the literature either double-center or transform to distances. Thus, the metrics cannot be computed from each other.

Hellinger (Jeffreys-Matusita). [16, 37] The Hellinger distance measures the Euclidean distance between the square-root representations of two probability distributions, $\sqrt{P_1}$ and $\sqrt{P_2}$. Its closed-form solution for two multivariate Gaussians is:

$$d^2_{\text{Hellinger}}(P_1, P_2) = 1 - \frac{\left(|\mathbf{\Sigma}_1|\,|\mathbf{\Sigma}_2|\right)^{1/4}}{\left|\frac{\mathbf{\Sigma}_1+\mathbf{\Sigma}_2}{2}\right|^{1/2}}\exp\left(-\frac{1}{8}\left(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\right)^T\left(\frac{\mathbf{\Sigma}_1+\mathbf{\Sigma}_2}{2}\right)^{-1}\left(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\right)\right).$$

It is directly related to the Bhattacharyya distance with the following non-linear relation:

$$d^2_{\text{Hellinger}}(P_1, P_2) = 1 - \exp\left(-d_{\text{Bhattacharyya}}(P_1, P_2)\right). \tag{22}$$

A.2.2 DIVERGENCES

Kullback–Leibler Divergence. [27] The KL divergence quantifies how much one probability distribution differs from another by measuring the expected information lost when P_1 is used to approximate P_2. The KL divergence does not satisfy the symmetry and triangle inequality, therefore,

it is not a proper metric. However, in this implementation, the following symmetrized KL divergence with the closed-form solution for Gaussians is used:

$$d_{\mathrm{KL}}\left(P_1, P_2\right) = \frac{1}{2}\ln\frac{|\boldsymbol{\Sigma}_2|}{|\boldsymbol{\Sigma}_1|} + \frac{1}{2}\operatorname{tr}\left(\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\Sigma}_1\right) - \frac{n}{2} + \frac{1}{2}\left(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1\right)^T \boldsymbol{\Sigma}_2^{-1}\left(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1\right), \tag{23}$$

$$d_{\mathrm{KL}}^{\mathrm{sym}}\left(P_1, P_2\right) = \frac{d_{\mathrm{KL}}\left(P_1, P_2\right) + d_{\mathrm{KL}}\left(P_2, P_1\right)}{2}, \tag{24}$$

where n is the dimension of the multivariate Gaussians.

Bhattacharyya. [4, 20] Bhattacharyya distance measures the logarithmic overlap between two probability distributions based on the geometric mean of their probabilities. Bhattacharyya distance is not a proper metric, as it does not satisfy the triangle inequality. With multivariate Gaussian distributions, it can be computed as follows:

$$d_{\mathrm{Bhattacharyya}} = \frac{1}{8}\left(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\right)^T \left(\frac{\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2}{2}\right)^{-1}\left(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\right) + \frac{1}{2}\ln\left(\frac{\left|\frac{\boldsymbol{\Sigma}_1+\boldsymbol{\Sigma}_2}{2}\right|}{\sqrt{|\boldsymbol{\Sigma}_1|\,|\boldsymbol{\Sigma}_2|}}\right). \tag{25}$$

A.3 Equivalence Classes

BayesCompare metrics are invariant to certain transformations. These are listed here in detail. In the definitions, $\mathbf{X} \in \mathbb{R}^{n\times k}$ is the activations of a DNN layer with k neurons for a list of n stimuli and the kernel matrix for it is $\mathbf{K}_{\mathbf{X}} = \mathbf{X}\mathbf{X}^T$.

Orthogonal transformations. Since BayesCompare metrics rely on kernel matrices, they are invariant to orthogonal transformations, including permutations and rotations of feature space. For an orthogonal transformation, $\mathbf{Q} \in \mathbb{R}^{k\times k}$, $\mathbf{Q}\mathbf{Q}^T = \mathbf{I}$. Then, for the transformed activations, $\hat{\mathbf{X}} = \mathbf{X}\mathbf{Q}$, the corresponding kernel matrix is,

$$\mathbf{K}_{\hat{X}} = \hat{\mathbf{X}}\hat{\mathbf{X}}^T = \mathbf{X}\mathbf{Q}\mathbf{Q}^T\mathbf{X}^T = \mathbf{X}\mathbf{X}^T = \mathbf{K}_{\mathbf{X}} \tag{26}$$

This set of equivalences is common to all metrics based on kernel matrices, such as CKA and RSA.

Scale invariance. Kernel matrices are trace-normalized before computing distances. Therefore, two metrics can yield the same distance if and only if $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2$, $\boldsymbol{\Sigma}$ as in Eq. 3, which is equivalent to $\frac{\tilde{\mathbf{K}}_1}{\operatorname{tr}(\tilde{\mathbf{K}}_1)} = \frac{\tilde{\mathbf{K}}_2}{\operatorname{tr}(\tilde{\mathbf{K}}_2)}$. For a scaled activation $\mathbf{X}' = c\mathbf{X}$, kernel matrix is $\mathbf{K}_{\mathbf{X}'} = \mathbf{X}'\mathbf{X}'^T = c^2\mathbf{X}\mathbf{X}^T$. Then,

$$\frac{\mathbf{K}_{\mathbf{X}'}}{\operatorname{tr}(\mathbf{K}_{\mathbf{X}'})} = \frac{c^2\mathbf{X}\mathbf{X}^T}{\operatorname{tr}(c^2\mathbf{X}\mathbf{X}^T)} = \frac{\mathbf{X}\mathbf{X}^T}{\operatorname{tr}(\mathbf{X}\mathbf{X}^T)} = \frac{\mathbf{K}_{\mathbf{X}}}{\operatorname{tr}(\mathbf{K}_{\mathbf{X}})} \quad \text{for} \quad c \in \mathbb{R}\setminus\{0\} \tag{27}$$

This invariance is also common with CKA. BayesCompare is, however, sensitive to the distance of individual patterns to the origin. CKA removes this information with double-centering, and RSA removes it by converting the Gram matrix to distances. Hence, the BayesCompare measures have a smaller equivalence class than those of the methods in the literature.

A.4 Resources for Experiments

To obtain activations from neural networks and compute distance matrices with different methods, we used an HPC cluster node with 2 AMD Rome CPUs, each with 64 cores @ 2.6 GHz and the node has 512 GB of RAM. Depending on the number of images and the size of the architecture, the covariance matrix extraction changes between 2 minutes and a couple of hours per architecture (including 5 instances). Distance computation also ranges between around 10 minutes and a couple of hours, depending on the method used. Therefore, all three experiments can be run in 4 to 5 days with similar computing resources.

A.5 Models and Training Variations

Table 2: Trained models with differences in their training recipes, other than different random initializations, Top-1 accuracies on ImageNet1k validation set, and selected layers from each model. Resnet50 instances are trained using four Tesla V100 SXM2 graphics cards (32GB) and all other model instances are trained using four NVIDIA Ampere graphics cards (40GB) on one HPC cluster node. The intermediate layer activations are obtained using TorchLens (ver. 0.1.36, GPL-3.0 License) [52]. Training durations per model instance were as follows: ResNet:∼1 day, ViT:∼11 days, Swin:∼1 day, Swin V2:∼4 days, ConvNeXt:∼2 days, EfficientNet:∼4 days.

Model Architecture	Training Variety	Accuracy (Top-1)	Selected Layer
ResNet50 [15]	-	76.01	Last ReLU in each block
	-	76.38	
	Learning Rate and Augmentation Policy	76.52	
	Learning Rate and Augmentation Policy	76.78	
	Pre-trained by PyTorch	80.90	
ViT-B/16 [11]	-	80.73	Last Layernorm in each block
	-	80.69	
	Augmentation Policy	80.59	
	Augmentation Policy	79.01	
	Pre-trained by PyTorch	81.00	
Swin Transformer Tiny [29]	-	81.45	First Layernorm in each block
	-	81.51	
	Augmentation Policy	81.53	
	Augmentation Policy and Interpolation	81.08	
	Pre-trained by PyTorch	81.50	
Swin Transformer V2 Small [30]	-	83.69	First Add in each block
	-	83.53	
	Augmentation Policy	83.45	
	Augmentation Policy and Interpolation	83.22	
	Pre-trained by PyTorch	83.70	
ConvNeXt Tiny [31]	-	81.31	The Add in each block
	Training Duration	81.27	
	Augmentation Policy	81.44	
	Augmentation Policy and Interpolation	81.12	
	Pre-trained by PyTorch	82.52	
EfficientNet V2 Small [50]	-	83.76	The Add in each block if exists, Batchnorm otherwise
	-	83.65	
	Augmentation Policy	83.42	
	Augmentation Policy and Interpolation	83.35	
	Pre-trained by PyTorch	84.20	

A.6 DISTANCE MATRICES

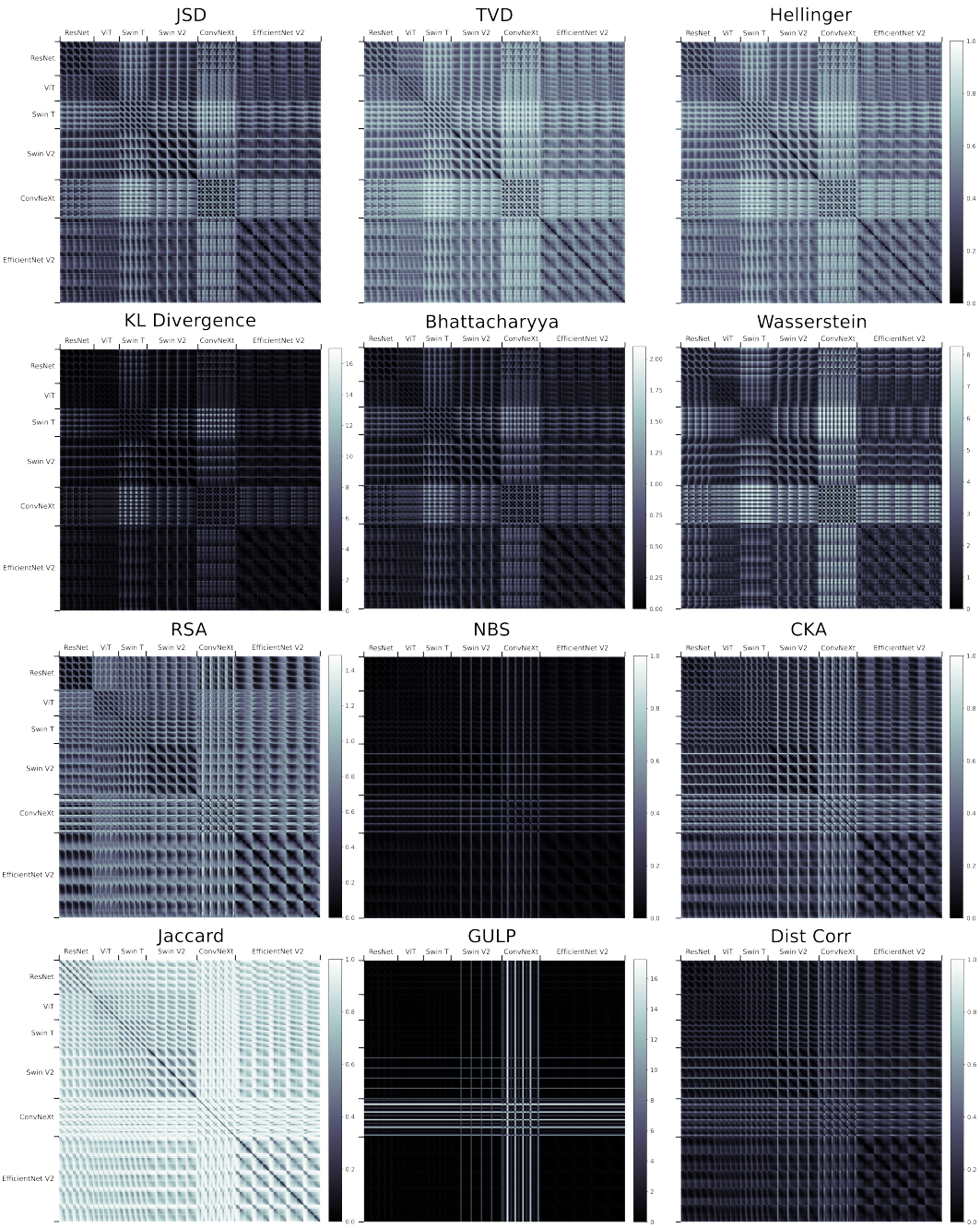

Figure 8: Distance matrices for all methods and all instances of all architectures. There are 5 instances for each architecture. $b = 1/100$ noise level is shown for BayesCompare metrics, obtained for Experiment 1.

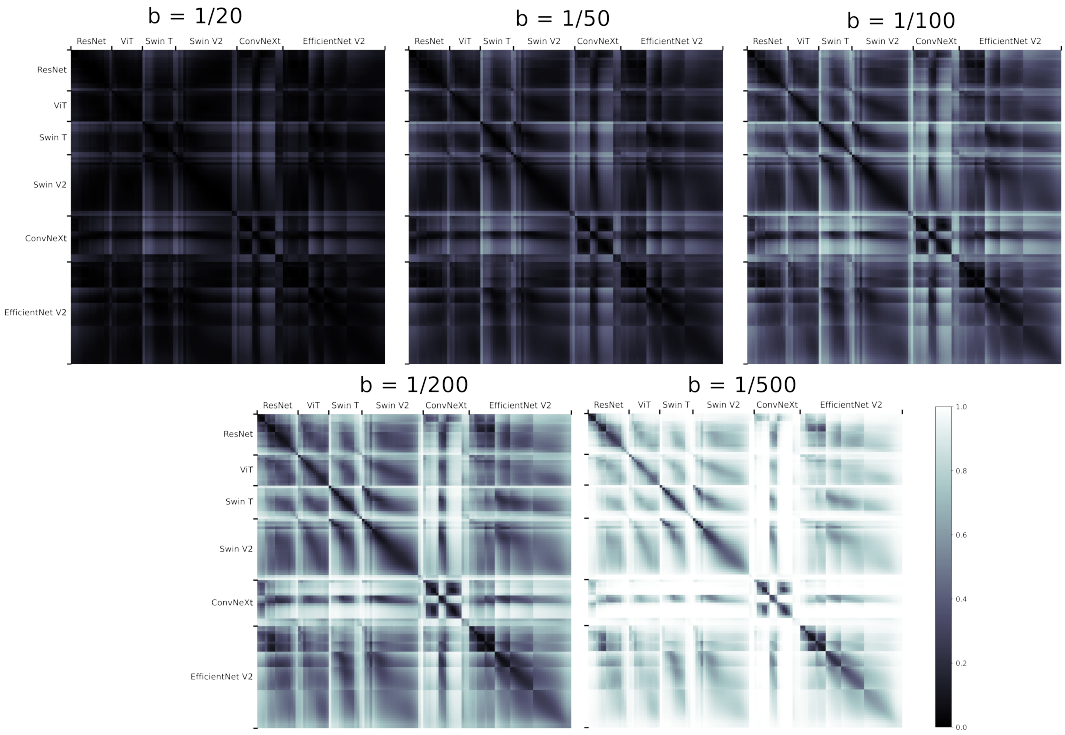

Figure 9: Effects of additive noise on JSD, one of the bounded metrics in BayesCompare, shown on the distance matrices averaged over instances, obtained for Experiment 1.

A.7 Statistical Test Results for Experiment 1

We computed a Wilcoxon signed-rank test over the silhouette scores obtained for 100 different 1000-image subsets of MS COCO. In other words, we have computed the statistical tests on the silhouette scores obtained from the distance matrices for Exp 3 (only those with $n = 1000$ and $b = 0.05$). We applied this test both on the unranked and ranked distance matrices, and the result of the latter is shown in Fig. 6A.

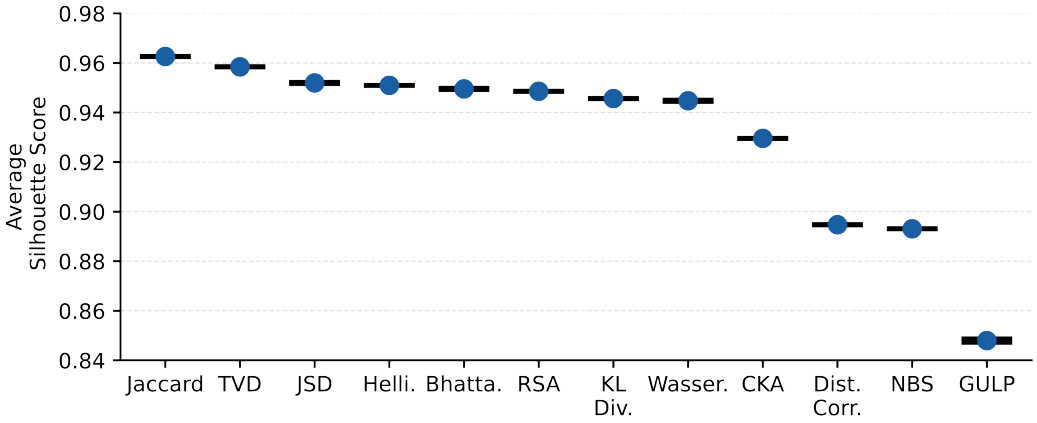

Figure 10: Mean silhouette scores over 100 random image sets with with $n = 1000$ and $b = 0.05$, without rank transformation. Error bars show standard deviation. All pairs of metrics are significantly different from each other ($p < .05$, Bonferroni corrected).