AVERAGED KERNELS ARE BETTER KERNELS

Anonymous authors
Paper under double-blind review

ABSTRACT

If multiple artists are asked to draw a circle by hand, each one will produce something slightly imperfect. Yet, the average of their sketches can look strikingly close to ideal. We investigate whether representations produced by separately trained models adhere to a similar principle. Precisely, we study the average model *kernel*: the Gram matrix computed from all model embeddings. Using metrics of representational similarity, i.e., Centered Kernel Alignment (CKA) and Mutual k-Nearest Neighbors (MKNN), we show that the average kernel increases in similarity with the kernel of a stronger model. We demonstrate this for vision models trained on different datasets, skewed datasets, models of different architectures, and even models trained on different tasks. Empirically, we even find that the similarity landscape with respect to teacher kernels is a convex basin. Finally, we show how averaging can be directly translated into improved model performance by using DiffKNN, a differentiable version of MKNN, to optimize a student network for representational similarity with the average kernel. We observe that the resulting student model increases in task accuracy with the number of models being averaged. More broadly, our findings hint at a more general framing that can be given for model-merging techniques, in which models can be thought to lie in the same loss basin with respect to their representations.

1 INTRODUCTION

Many ensembling techniques exploit the fact that averaging across models can outperform any single model instance. Model averaging can be done at the output level, e.g. bagging (Breiman, 1996), or at the weight level, e.g., model souping (White, 2016; Wortsman et al., 2022), or at the representation level, e.g., multiple-distillation (Allen-Zhu & Li, 2020). The explanation given for the effectiveness of this type of averaging is that it reduces variance in weight-space and increases the chance for a student to encounter useful features. These methods work best when models are already aligned in weight space or representation space, but they cannot be applied for models trained separately on different datasets or with different architectures.

In this work, we suggest an alternative explanation for why averaging works, with an interpretation motivated by the observation that independent models tend to converge in representation as they improve Huh et al. (2024); Conwell et al. (2022); Han et al. (2023); Moschella et al. (2022); Norelli et al. (2023). We then show how this can be used to extend the notion of model averaging to a broader range of architectural settings than covered by weight averaging. Recent works, namely (Huh et al., 2024), posit that as models improve, they converge to the same 'platonic' representation $\boldsymbol{K}^*$. More precisely, for a model f and dataset with elements x_i, the kernel is the matrix $\boldsymbol{K}$ of all dot products between model embeddings $\phi_i = f(x_i)$, such that $\boldsymbol{K}_{ij} = \langle \phi_i, \phi_j \rangle$. The similarity structures of different models can be compared using measures such as *Mutual k-Nearest Neighbors* (MKNN) or *Centered Kernel Alignment* (CKA) which apply comparisons across the kernel matrices from two different models. Works such as Huh et al. (2024) make the observation that as models improve on task performance, their kernels converge in representational similarity. Of course, in Plato's philosophy, platonic forms are not material and do not exist in the same sense that physical objects exist. Plato asserts, for example, that there are no perfect circles in nature. However, we investigate averaging as a method that can be seen as a method for approaching the ideal. For illustration, imagine asking many people to draw a circle. They will each draw something imperfect, but the circles may be averaged together to smooth out individual mistakes and achieve something closer to the platonic ideal.

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.

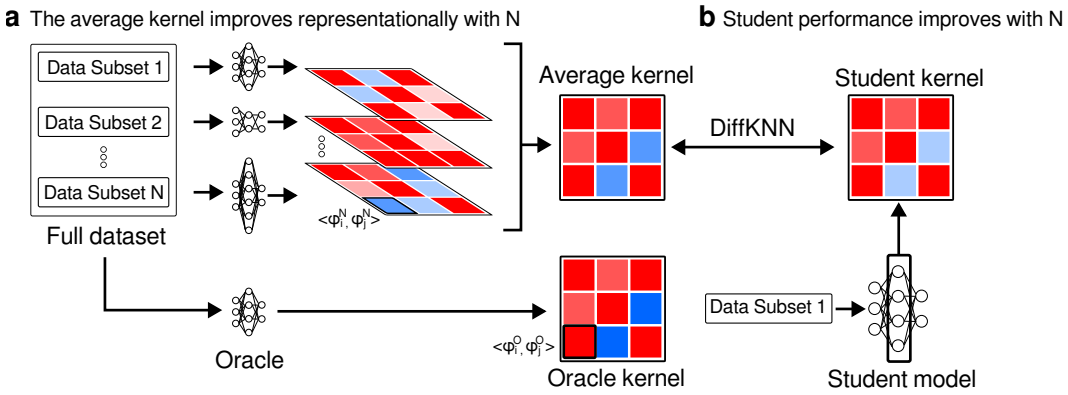

Figure 1: **Schematic overview**: (**a**) In our experiments, we average together the kernels from many different independently trained models. We do this for ensembles that include models trained on different data subsets, skewed data subsets, models of different architectures, and models trained on different tasks. To quantify the strength of the average kernel, we compare it to the kernel of an *oracle*, a model trained on more data, or a stronger architecture. We observe that, structurally, the average kernel approaches the oracle kernel as N increases. (**b**) We leverage this observation to train a student model that optimizes for embeddings that representationally match the average kernel. The resulting student performance is greater than the performance of its individual teachers. This confirms that the better quality of the average kernel can actually be translated into better task ability. This optimization is made possible with the use of DiffKNN, a differentiable version of MKNN.

Could it be that averaging works because every model imperfectly instantiates the platonic representation? In this work, we make the empirical observation that as model kernels are averaged together, they produce a mean kernel that trends towards that of a stronger model. In the limit, our theoretical motivation is the notion that each model can be seen as providing a noisy sample of the platonic kernel $\boldsymbol{K}^*$. We simulate this by training an 'oracle' model f^O to high performance on a full, balanced dataset, and separately training a population of relatively data-impoverished models. We empirically find that averaging the kernels of the weak models together results in a better kernel, which increases in representational similarity to $\boldsymbol{K}^O$ as N grows. In fact, we find that this observation holds, even for pretrained models of different architectures and models trained on different tasks. Additionally, we make the observation that, empirically, the landscape of similarity to f^O is often convex with respect to weightings between the ensemble's constituent kernels. This hints at a broader framing that could be given for model merging techniques: while prior works such as Git Re-Basin Ainsworth et al. (2023) show that networks are in the same loss basin modulo permutation, our results support a view that many models of different tasks and architectures could likewise lie in the same loss basin when viewed at the level of representations.

Finally, we connect our observations to model performance. Building on methods of kernel distillation He & Ozay (2022); Tian et al. (2019); Tung & Mori (2019), we use differentiable versions of CKA and MKNN to train a student model and optimize its embeddings so that they are representationally close to that of the average kernel. Across models of different checkpoints and architectures, we find that this procedure results in increased student accuracy, confirming the causal connection between kernel quality and task performance. Moreover, we observe these effects, even when the student distribution is poor, i.e., small and heavily imbalanced. In application, similar methods could be particularly useful in federated learning settings with heterogeneous architectures. For example, separate institutions like hospitals could train local models on siloed data and share model access or weights for evaluation on a public transfer set, allowing for the creation of a student model that aggregates knowledge without directly pooling private datasets. We anticipate that such use cases will become more common in the future, since many institutions are willing to make their models open-weight, but the training schemes and data are often closed or opaque.

Our contributions:

1. We present empirical evidence that the average kernel improves in quality, becoming more similar to a high-performing oracle kernel, as the number of averaged kernels, N, grows. We demonstrate that this holds even for heterogeneous architectures.

2. We give a theoretical framing that provides intuition for why kernel averaging is effective, showing that under independent kernel noise around a shared target representation, kernel averaging recovers the target neighborhood structure as ensemble size grows.
3. Closing the loop, we show that a student model can even be trained to optimize its embeddings for representational similarity with the average kernel, using standard representational distillation methods. This improves performance, demonstrating a causal relationship between model performance and average kernel structure.

2 PRIOR WORKS

Convergence of representations The Platonic Representation Hypothesis (PRH) and related works (Huh et al., 2024; Conwell et al., 2022; Han et al., 2023; Moschella et al., 2022; Norelli et al., 2023) make the observation that as models improve, their representations become more similar (Huh et al., 2024), as measured by MKNN. They posit that all models are moving towards a common 'platonic' representation. Implicit in their findings is a pathway to platonic representations, namely, training better models from larger amounts of high-quality data. In this work, we present findings that corroborate PRH in so far as we find evidence for the converse: averaging kernels both improves the representations and also improves performance (see Section 6).

Model averaging Many of the earliest approaches for model-merging were simple weight-averaging methods (White, 2016; Wortsman et al., 2022; Matena & Raffel, 2022; Yadav et al., 2023). But naive weight-averaging is only possible when models start from the same pre-trained checkpoint, since feed-forward layers are symmetric with respect to neuron permutation. Some approaches attempt to resolve this by finding this permutation via explicit matching (Wang et al., 2020; Ainsworth et al., 2023). This makes the restrictive assumption that such a matching between neurons exists, which limits the representations that a student may learn, especially when a student's architecture differs from a teacher's. In this work, we explore whether knowledge transfer is possible without matching. Another way of doing ensembling is multiple distillation. We build on this work and present a broader framing for why such averaging may work. Many multiple-distillation approaches require models to use the same embedding space (Liu et al., 2020). In contrast, we consider averaging at the *kernel* level, since kernels can be compared even if the underlying embedding spaces differ.

Guidance Prior works have explored the notion of guiding networks using representational similarity as an auxiliary objective (He & Ozay, 2022; Subramaniam et al., 2025a;b). We extend Subramaniam et al. (2025b), since we focus on knowledge transfer from an *ensemble* of models rather than a single model. Compared to He & Ozay (2022), we offer a complementary explanation of why kernel averaging is effective as a strategy and explain when it fails. During training, we optimize for alignment based on MKNN, which avoids the pitfalls of raw kernel correlation, i.e., Centered Kernel Alignment (CKA); e.g., bias from diagonal elements of the kernel. We share similar motivations to Yu et al. (2025), although we do not learn an adapter between models, and instead use representational similarity measures for direct alignment.

3 KERNEL AVERAGING INTUITION

Here, we provide a brief intuitive motivation for why it can be expected that ensembling kernels should lead to better representations and performance. Consider a dataset of M elements. A trained model f gives an embedding for the dataset elements $f(i) = \phi_i$. This induces a distance between elements via the inner-product between model embeddings: $\boldsymbol{K}^f_{i,j} := \langle \phi_i, \phi_j \rangle$. For some chosen k, let $\mathcal{S}^f_k$ be indices of the k elements nearest to x_i under this distance.

Definition: The *mutual k nearest neighbors* distance between two models f and g is defined as the overlap between their nearest neighbor sets under each model's distance function:

$$m_{nn}(f, g) := \frac{1}{M} \left| \mathcal{S}^f_k \cap \mathcal{S}^g_k \right|$$

Observation 1: Let $\mathcal{S}^*(i)$ be the nearest neighbors of x_i under the platonic representation $\phi^* = f^*(x_i)$. Assume each model learns a noisy version of the platonic representation, and errors are not correlated between models. For a particular model, let the distance $\boldsymbol{K}_{i,j}$ be a random variable with

mean $\boldsymbol{K}^*_{i,j} := \langle \phi^*_i, \phi^*_j \rangle$. Given N samples $\boldsymbol{K}^k_{i,j} \sim \boldsymbol{K}_{i,j}$, let $\overline{\boldsymbol{K}}_{i,j} = \frac{1}{N} \sum_k \boldsymbol{K}^k_{i,j}$. Let f^* and $\bar{f}$ be models that induce kernels $\boldsymbol{K}^*$ and $\overline{\boldsymbol{K}}$ respectively. Then $\lim_{N\to\infty} P\left(m_{nn}(\bar{f}, f^*) = 1\right) = 1$.

Observation 2: Even for finite N, an additional model kernel to the ensemble can still be useful. In particular, let $I\left(\boldsymbol{K}^*_{i,j}; \overline{\boldsymbol{K}}_{i,j}\right)$ be the mutual information between $\boldsymbol{K}^*_{i,j}$ and $\overline{\boldsymbol{K}}_{i,j}$. Then, $I\left(\boldsymbol{K}^*_{i,j}; \frac{1}{N}\sum_k^N \boldsymbol{K}^k_{i,j}\right) \leq I\left(\boldsymbol{K}^*_{i,j}; \boldsymbol{K}^{k+1}_{i,j}, \frac{1}{N}\sum_{k=1}^N \boldsymbol{K}^k_{i,j}\right)$.

Observation 3: Finally, even if the errors do not follow a normal distribution or are correlated, and the best achievable mutual nearest neighbor distance $m_{NN}(f, f^*)$ is p, then $\lim_{N\to\infty} P(m_{nn}(\bar{f}, f^*) = p) = 1$.

Proofs: Given in Appendix Section A.5.

4 Approach

Here, we outline how we train students for representational similarity with the ensemble average. An overview of our training approach can be seen in Figure 1 and Supplementary Figure 10. Briefly, we are given an ensemble of models and a dataset. Each model induces a neighborhood around each dataset element in embedding space. We average across models to obtain the average neighborhood. We optimize a student model to produce internal embeddings that give rise to this average neighborhood.

Formally, given a dataset $\mathcal{D} = \{x_1, ..., x_M\}$ of M elements, and an ensemble of N models $\{f_1, ..., f_N\}$, each model, f_k, induces a distance between dataset elements via the dot product between model embeddings: $\boldsymbol{K}^{f_k,l}_{ij} = \langle \phi^{k,l}_i, \phi^{k,l}_j \rangle$, where $\phi^{k,l}_i = f_k(x_i)_l$ is the output of f_k at layer l. Elsewhere, our notation will omit the index by layer l, and it can be assumed that results are averaged across layers. Under the distance defined by this dot product, let $\mathcal{S}^k_n(i)$ be the indices of the n closest elements to ϕ_i. We average distances across models and obtain an average neighborhood. We then study the quality of the resulting average kernel (see Section 6).

Guidance To optimize for representational similarity with the ensemble kernel, we build on the work of Subramaniam et al. (2025b). There, a student model is trained to optimize representational similarity with a teacher model. We use similar methods here. During a forward pass, we obtain the final logits as well as the intermediate outputs of each layer of the network. The final logits are used to compute the task loss, and the intermediate outputs are concatenated per layer, and tuned to match the average layer kernel of the teachers. We train student models using a diverse set of representational similarity metrics, to optimize towards the average kernel. Namely, we use a differentiable version of MKNN (DiffKNN) Subramaniam et al. (2025a), as well as Centered Kernel Alignment (CKA) Kornblith et al. (2019), and Unbiased Centered Kernel Alignment (UCKA) Song et al. (2007). Below, we give the objective function according to each metric, starting with the more familiar CKA.

CKA First, define the Hilbert-Schmidt Independence criterion (HSIC) for models f and g with kernels $\boldsymbol{K}^f, \boldsymbol{K}^g \in \mathbb{R}^{M\times M}$ as $HSIC(\boldsymbol{K}^f, \boldsymbol{K}^g) = \text{tr}(\tilde{\boldsymbol{K}^f}\tilde{\boldsymbol{K}^g})$, where $\tilde{\boldsymbol{K}} = \boldsymbol{HKH}$, and $\boldsymbol{H}$ is the centering matrix, $\boldsymbol{H} = \boldsymbol{I} - \frac{1}{M}\boldsymbol{1}\boldsymbol{1}^T$. Then, the CKA between f and g is:

$$\mathcal{L}_{CKA} = \frac{HSIC(\boldsymbol{K}^f, \boldsymbol{K}^g)}{\sqrt{HSIC(\boldsymbol{K}^f, \boldsymbol{K}^f) * HSIC(\boldsymbol{K}^g, \boldsymbol{K}^g)}} \tag{1}$$

Unbiased-CKA We also explore using Unbiased-CKA (UCKA) Song et al. (2007), which removes bias due to the diagonal. The formula is the same as in Equation (1), but unbiased $HSIC_U$ is used instead of $HSIC$:

$$HSIC_U(\boldsymbol{K}^f, \boldsymbol{K}^g) = \frac{1}{m(m-3)}\left[\text{tr}(\tilde{\boldsymbol{K}}^f\tilde{\boldsymbol{K}}^g) + \frac{\boldsymbol{1}^T\tilde{\boldsymbol{K}}^f\boldsymbol{1}\boldsymbol{1}^T\tilde{\boldsymbol{K}}^g\boldsymbol{1}}{(m-1)(m-2)} - \frac{2\boldsymbol{1}^T\tilde{\boldsymbol{K}}^f\tilde{\boldsymbol{K}}^g\boldsymbol{1}}{m-2}\right] \tag{2}$$

where

$$\tilde{\boldsymbol{K}}^f := \boldsymbol{K} - \text{diag}(\boldsymbol{K}^f), \qquad \tilde{\boldsymbol{K}}^g := \boldsymbol{K} - \text{diag}(\boldsymbol{K}^g), \tag{3}$$

so that $\tilde{\boldsymbol{K}}^f_{ii} = \tilde{\boldsymbol{K}}^g_{ii} = 0$ and $\tilde{\boldsymbol{K}}^f_{ij} = \boldsymbol{K}^g_{ij}$, $\tilde{\boldsymbol{K}}^f_{ij} = \boldsymbol{K}^g_{ij}$ for $i \neq j$.

DiffKNN Briefly, for f_a and f_b, the differentiable mutual k nearest neighbors (DiffKNN) is given as:

$$\mathcal{L}_{\text{MKNN}} = \frac{1}{M} \sum_{i=1}^{M} D_{KL}\Big(P_{i\setminus *} \Big\| Q_{i\setminus *}\Big) \tag{4}$$

where

$$P_{ij} = \frac{\exp(\boldsymbol{K}_{ij}^{f_a})\, \mathbf{1}\{j \in \mathcal{S}_k^{f_a}(i)\}}{\sum_{j' \in \mathcal{S}_k^{f_a}(i)} \exp(\boldsymbol{K}_{ij'}^{f_a})} \tag{5}$$

and Q_{ij} is defined similarly for f_b. See Subramaniam et al. (2025b) or Appendix A.3 for full description.

Training In the case of any given measure, $\mathcal{L}_{CKA}$, $\mathcal{L}_{UCKA}$, $\mathcal{L}_{MKNN}$, we average the representational similarity across layers and teachers to obtain $\mathcal{L}_{\text{sim}}$. We train jointly on the task objective and the similarity objective, which are computed by batch. The final objective is then: $\mathcal{L} = \mathcal{L}_{CE} + \mathcal{L}_{sim}$ During training, we take the log of the CKA and UCKA loss so that it has the same range as the cross-entropy component.

5 METHODOLOGY

Experiments We find empirical evidence for the observations stated in Section 3 in two stages. First, we want to determine whether averaging the kernels of many models together produces a better kernel. We test this by training a population of impoverished vision models on small subsets of a full dataset. We then measure the representational similarity, e.g., MKNN, with an oracle vision model, which has been trained on a much larger dataset. Second, we test whether student models can actually be optimized towards this average kernel to obtain improved performance. We train models from scratch on small subsets of data as a baseline. We then train models on the same data but jointly optimize for representational similarity with the ensemble average kernel. We then study how task performance varies as the number of teachers in the ensemble grows.

We experiment with two settings: one in which the models in the ensemble are all trained on the same distribution of data and one in which the data distributions are all different. We call these settings *balanced* and *specialist*, respectively. The distributions of specialist datasets all share the same support, but each distribution is severely skewed towards a few classes (see Figure 11). For ImageNet100, approximately 99.5% of each specialist split consists of examples from its majority classes. Specialists do not overlap in the classes for which they 'specialize'. Note that this means that for the specialist experiments, the test set represents a distribution shift.

Data Unlike in language tasks, there is a very natural mapping between individual inputs (images) and embeddings (hidden states) in the vision domain, so we perform experiments on vision classification tasks. We run two groups of experiments. A small-scale experiment is run on CIFAR10 Krizhevsky et al. (2009), for which we train ResNet20 models (He et al., 2016), as implemented by Idelbayev. We also run experiments in which we train ResNet18 models (He et al., 2016) on a subset of ImageNet (Krizhevsky et al., 2012), with 100 classes selected according to Van Gansbeke et al. (2020). For the purpose of the experiment, we take the test set from the ImageNet validation set and the CIFAR10 test set. The test set is common to all splits. The validation set is different per split, and follows the distribution of the train set. Teacher kernels are taken from the checkpoint that achieves the best validation loss.

Multi-architecture and multi-task Outside of the small model setting, we investigate kernel averaging when architecture and training task are varied more significantly. In the first setting, *multi-architecture*, we average together pretrained models of different architectures that shared a common training task. In the second setting, *multi-task*, we average together models of different training tasks and architectures. For each setting, we use publicly available checkpoints. In the multi-architecture case, we average together ResNet-18, ResNet-50, ResNet-101 (He et al., 2016), ViT-B/16, ViT-B/32 (Dosovitskiy et al., 2020), ConvNeXt-Tiny, and ConvNeXt-Small (Liu et al., 2022). See Section A.1 for details. In the multi-task case, we average together ViT-B-MAE (trained with MAE) (Dosovitskiy et al., 2020), CLIP ViT-B/16 (contrastive language-image training) (Radford et al., 2021), ResNet-50 (He et al., 2016), DEIT-Base Distilled (distilled ViT) (Touvron et al., 2021), DINOv2 Base (discriminative self-supervised pretraining) (Oquab et al., 2023), and CNXV2-Base (fully-convolutional

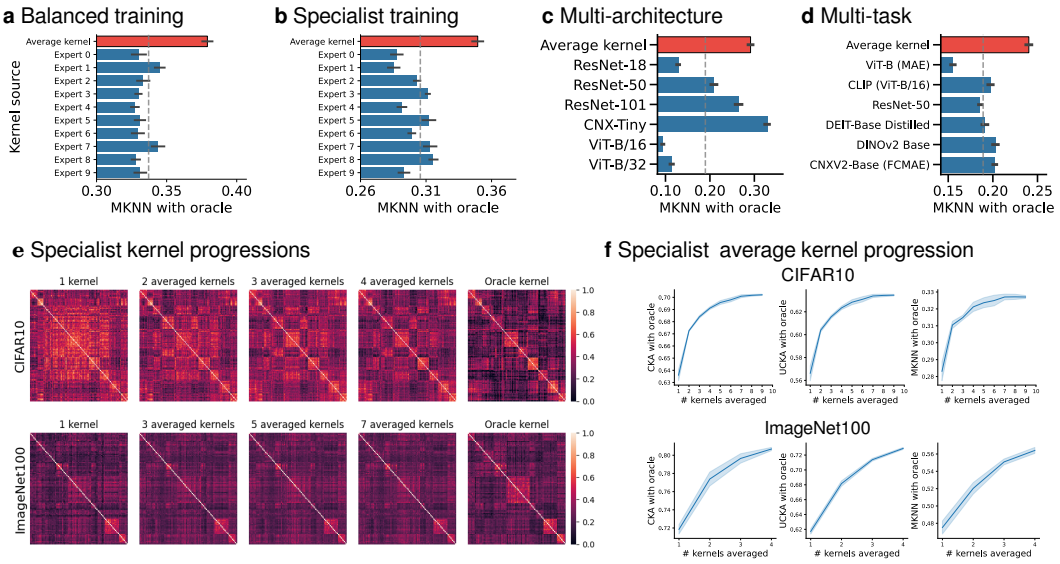

Figure 2: **Averaging kernels improves representational similarity with a more accurate model.** We separately train independent models on disjoint (**a**) balanced and (**b**) specialist data distributions. We compare with an oracle model, which has been trained on the full, balanced dataset. In both cases, we see that the average kernel (red) is closer to the oracle than any of the individual models (blue). Notably, the sum is greater than its parts, in the sense that the average kernel quality (red) is greater than the average of individual kernel qualities (grey, dashed). We see that this is even the case for (**c**) vision models of different architectures and (**d**) models trained on different tasks! We do a deep dive on models trained on specialist distribution data, i.e., each model is trained on a different, lopsided data distribution, and we observe that (**e**) the average model kernels approach the oracle (rightmost) as N increases. These trends are quantified in (**f**) using CKA, UCKA, and MKNN. Shading shows standard error across permutations ($n = 5$).

MAE) (Woo et al., 2023). Each is pretrained and taken from Wightman (2019). When it comes to training in both the multi-architecture and multi-task case, we train the student model on a balanced subset of the data. We emphasize that for all cases, especially the mixed task settings, we cannot directly employ techniques like logit ensembling or multiple-distillation since all models are trained on entirely different tasks and therefore, have different logit spaces. Similarly, techniques that use hidden-state adapters (Csiszárik et al., 2021) which train a linear layer to transform one representation to another are also prohibitively expensive in comparison to kernel averaging with respect to memory usage.

6 Results

Averaging kernels In short, we find that kernels can be averaged to obtain stronger representations, i.e., closer representational similarity to the oracle. We observe that the *sum is greater than its parts*, in the sense that the quality of the average kernel is greater than the average kernel quality. This holds for the balanced, specialist, multi-architecture, and multi-task models (see Figure 2a-d). We observe that as we average together more of these kernels, we obtain a kernel that trends closer to the oracle's kernel, as measured by CKA, UCKA, and MKNN (see Figure 2)e-f. A single specialist kernel has trouble producing representations that clearly distinguish between classes. These distinctions become clearer as more specialist kernels are averaged.

Moreover, we observe something stronger. It appears that we can take a weighted average among kernels and observe that the landscape with respect to oracle-similarity is a basin. Precisely, for convex weights, $\sum_i w_i = 1$, we measure the similarity between the oracle kernel and $\sum_i w_i \cdot \boldsymbol{K}_i$, and we find that the highest quality kernels are mixtures between teachers and that the landscape of this similarity appears approximately convex (see Figure 3).

Student training We find that we can in fact train a student model to jointly optimize for a classification objective and representational similarity with an ensemble of models. And we find

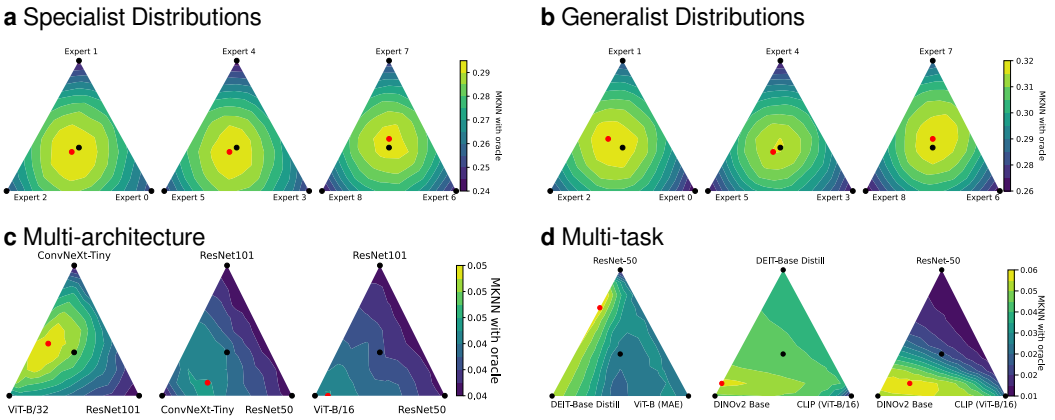

Figure 3: **Similarity landscapes appear convex with respect to constituent kernels**. Each vertex of each triangle represents a specialist model's kernel. Each interior point of a triangle shows the oracle similarity with the kernel of a convex weighted average of constituent kernels, as measured by MKNN. For models trained on (**a**) specialist and (**b**) generalist subsets of ImageNet100, the weighting between three teacher models that yields the highest similarity with the oracle (red) is usually close to the barycenter (black), and the landscape appears highly convex. This is less the case for multi-architecture and multi-task ensembles, although they also exhibit convexity.

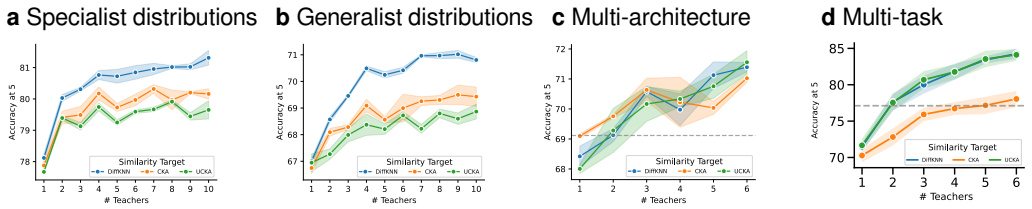

Figure 4: **Optimizing for representational similarity with larger ensembles results in better performance.** Averaging kernels across more teachers leads to better student task accuracy for (**a**) specialist, (**b**) generalist, (**c**) multi-architecture, (**d**) and multi-task ensembles. Shading shows standard error across seeds ($n = 3$). DiffKNN (blue) is empirically observed to be the best target for representational similarity. In (**c**) and (**d**), since models are more heterogeneous, for comparison, we plot the performance of a student guided by the single strongest teacher using DiffKNN (gray, dashed).

that training with guidance from an ensemble of teachers results in stronger test performance. As teachers are added to the ensemble, we find improvement on the task accuracy and loss to be roughly monotonic (see Figure 4). This closes the loop between averaging and performance, showing that kernel averaging does indeed induce model representations that result in higher task accuracy. Again, this holds for specialist, generalist, multi-architecture, and multi-task ensembles. We find that optimizing for representational similarity using DiffKNN results in the strongest performance. We speculate that that this is because MKNN imposes a weaker and more local constraint during training by encouraging neighborhood agreement rather than global kernel correlation. This affects a much more sparse set of kernel matrix elements. In comparison, CKA and UCKA encourage correlations across the entire kernel matrix to be the same. This is a stronger requirement, which may waste training time on regions of the kernel where correlations are ultimately unimportant.

We observe that as more teachers are used to guide the student, the representation quality of the student increases, i.e., its similarity with the oracle kernel increases. This can be seen qualitatively (see Figure 5a) and quantitatively (Figure 5b). As in fig. 3, we can plot a landscape with respect to convex weightings of teachers. The resulting accuracy landscape is rougher than the landscapes observed in fig. 3, due to the noise inherent in training, but it remains true that the best combination of teachers is better than any individual teacher.

Crucially, it should be noted that these improvements occur even when the student data is impoverished or skewed. In fact, we show that optimizing for representational similarity with the teacher ensemble

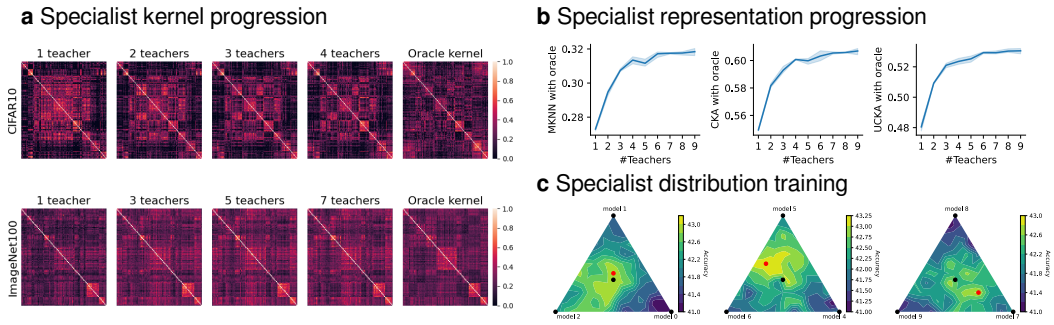

Figure 5: **Student kernels improve as more teachers are added**. As in fig. 2e, the student kernel (**a**) can be seen to grow in similarity with the oracle (rightmost) as the number of teachers increases. This is seen quantitatively by the trends shown in MKNN, UCKA, and CKA (**b**). Shading shows standard error across seeds ($n = 3$). The accuracy landscape (**c**) reveals that performance is stronger for weighted combinations of teachers than for any individual teacher, although the basins are rougher than in Figure 3 due to noise during training.

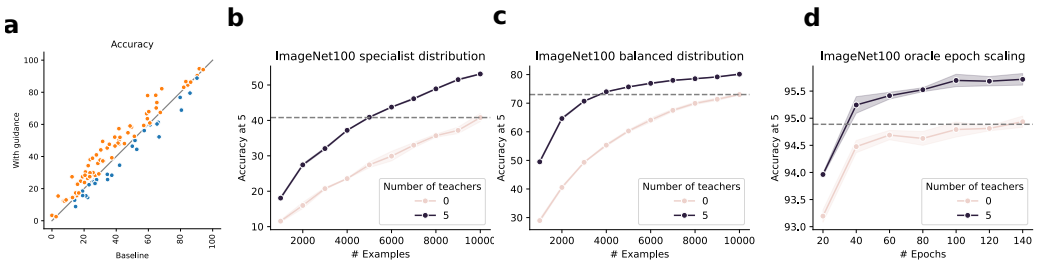

Figure 6: **Training guided by the average kernel increases generalization, data efficiency, and compute efficiency.** In (**a**) each point represents performance on an ImageNet100 class by the baseline (x-axis) and the guided student (y-axis). The student, which is trained on specialist distribution, nevertheless increases in performance on unseen classes (orange). Guidance from the average kernel allows for greater sample efficiency on (**b**) specialist and (**c**) generalist data, as well as (**d**) increased performance at equivalent epochs. That is, the guided model is able to achieve the baseline model's final performance (gray, dashed) with fewer samples and epochs.

can improve generalization for students, even when students are guided by kernels computed on lopsided distributions (See Figure 6a). Guidance using the average kernel also improves the sample efficiency (Figure 6b-c) and epoch efficiency (Figure 6d) of the student model. We find that the guided model achieves the performance of the baseline with only $\approx 40\%$ and $\approx 50\%$ of the samples for the balanced and specialist distributions respectively. A guided model trained on the full dataset, only requires $< 30\%$ of the epochs to achieve the same performance as the baseline.

Limitations Observation 1 holds only if errors in the kernel are not correlated among models. If models are all misaligned in a correlated way, then their average will be as well. Along these lines, we observe that accuracy plateaus as a function of ensemble size. This suggests that there exists representational knowledge that requires large scale data to learn, which cannot be obtained by averaging alone. We observe that our training is most successful when teachers are very different; see the larger gains for specialist training compared with balanced training (Table 1).

Comparison with ensembling Although our intention is not to propose a state-of-the-art model-merging method, we note that optimizing for similarity with the average kernel *could* be used as a type of model merging. To this end, we show how it compares to straightforward output ensembling, i.e., voting. We give results for prediction ensembling in Table 1. For this, each model in the ensemble is run, and the output prediction is decided by majority voting. Prediction ensembling obtains a very good performance, which makes sense since it requires N times the parameters and N times the inference time, as compared to running a single model. Despite this, our method of model merging via averaged representational similarity produces accuracies that are very close, especially when DiffKNN is used. For example, a single student trained with DiffKNN achieves an acc@1 of 84.29 on balanced CIFAR10, while the prediction ensemble system achieves an acc@1 of 86.38.

Specialist Training

Training	CIFAR10		ImageNet100	
	Acc@1	Acc@5	Acc@1	Acc@5
Baseline	55.68 ± 0.68	94.37 ± 0.11	39.92 ± 0.02	65.99 ± 0.22
DiffKNN	65.31 ± 0.39	96.92 ± 0.06	44.59 ± 0.28	70.81 ± 0.09
CKA	64.24 ± 0.18	96.93 ± 0.17	42.94 ± 0.22	69.43 ± 0.36
UCKA	63.87 ± 0.20	96.38 ± 0.12	42.43 ± 0.36	68.87 ± 0.28
Prediction Ensemble	70.62 ± 0.31	96.89 ± 0.075	56.49 ± 0.33	71.52 ± 0.28
Oracle	91.50 ± 0.09	99.67 ± 0.02	84.08 ± 0.10	95.61 ± 0.03

Balanced Training

Training	CIFAR10		ImageNet100	
	Acc@1	Acc@5	Acc@1	Acc@5
Baseline	81.52 ± 0.53	98.78 ± 0.08	52.98 ± 0.26	76.95 ± 0.32
DiffKNN	84.29 ± 0.06	99.14 ± 0.02	57.44 ± 0.07	81.31 ± 0.23
CKA	82.62 ± 0.11	98.98 ± 0.08	56.28 ± 0.26	80.16 ± 0.16
UCKA	83.69 ± 0.27	99.04 ± 0.05	55.73 ± 0.20	79.65 ± 0.29
Prediction Ensemble	86.38 ± 0.03	95.34 ± 0.08	64.72 ± 0.28	79.45 ± 0.16
Oracle	91.50 ± 0.09	99.67 ± 0.02	84.08 ± 0.10	95.61 ± 0.03

Table 1: **Test performance of students** Student test performance on specialist (top) and balanced (bottom) datasets, reported as accuracy on the top 1 and top 5 predicted classes. The baseline (first row) is trained on a single data subset and not optimized for representational similarity with any teachers. For DiffKNN, CKA, and UCKA, performance is reported for a student trained to optimize for representational similarity with all teachers, $n = 10$ for ImageNet100 and $n = 5$ for CIFAR10. For comparison, we also report the performance of a prediction ensembling approach; i.e., running each model and selecting an output based on majority vote. Even though the single student uses $N\times$ fewer parameters at inference time, it is still able to recover a good portion of the performance. Standard error is computed among seeds ($n = 3$).

7 Conclusion

Our findings, especially Figure 3, resolve an important question about the nature of averaging kernels. Prior to running our experiments, it could have been speculated that averaging, as opposed to voting, would be an incorrect approach to aggregating knowledge from different kernels. For example, this would be the case if all models were in different basins of the loss landscape due to their different specializations. However, the landscapes shown in Figure 3 suggest that all teacher kernels reside in the same convex basin, so that averaging is a well-motivated operation. This finding can be viewed in the same light as the observation made in Wortsman et al. (2022), in which fine-tuned weights of the same model were found to lie in the same loss basin. At the very least, it is safe to say that, empirically, we observe that a convex combination of kernels is superior to any individual model kernel.

More broadly, our findings suggest a more general framing to the story behind model soup Wortsman et al. (2022), and Git Re-Basin Ainsworth et al. (2023). Whereas the existence of model soups suggests that fine-tuned weights of the same checkpoint may all exist in the same loss basin, Git Re-Basin shows that many models, even models trained from different initializations, may exist in the same basin, modulo permutation of neurons. We extend this. For models that have been trained on different data distributions, our findings hint that these models can be considered to be in the same basin, once considered at the representation level. For future investigations, it can be explored whether this holds for models trained on very different data, perhaps different modalities. We have started this investigation in Appendix A.9 and see improvements from averaging kernels across modalities. Future work can explore this more thoroughly across diverse modalities.

AI USE STATEMENT

AI was not used in any part of this paper. All code, figures, and text were written by authors and validated by the authors.

ETHICS STATEMENT

We study the quality of averaged model kernels and explore how model merging may be accomplished with models that have been trained on different initializations. This has the potential to make the benefits of machine learning models more widely available, since it is common to come across open-weight models in the wild. This method gives users the ability to combine a diverse set of models, even if they do not have the resources to train such models themselves from scratch. We do not anticipate that this ability has any unique potential to be used for societal harm.

REPRODUCIBILITY STATEMENT

Code is included and open-sourced as part of this work. This work employs open-source models and datasets to aid reproducibility. All models used employ small amounts of compute and results can be validated on early-generation GPUs. We describe all experimental procedures to reproduce results and use standard hyperparameter settings for all training in the paper.

REFERENCES

Samuel Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. Git re-basin: Merging models modulo permutation symmetries. In *The Eleventh International Conference on Learning Representations*, 2023.

Zeyuan Allen-Zhu and Yuanzhi Li. Towards understanding ensemble, knowledge distillation and self-distillation in deep learning. *arXiv preprint arXiv:2012.09816*, 2020.

Shaden Naif Alshammari, John R Hershey, Axel Feldmann, William T Freeman, and Mark Hamilton. I-con: A unifying framework for representation learning. In *The Thirteenth International Conference on Learning Representations*, 2025.

Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

Colin Conwell, Jacob S Prince, Kendrick N Kay, George A Alvarez, and Talia Konkle. What can 1.8 billion regressions tell us about the pressures shaping high-level visual representation in brains and machines? *BioRxiv*, pp. 2022–03, 2022.

Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Algorithms for learning kernels based on centered alignment. *The Journal of Machine Learning Research*, 13:795–828, 2012.

Nello Cristianini, John Shawe-Taylor, Andre Elisseeff, and Jaz Kandola. On kernel-target alignment. *Advances in neural information processing systems*, 14, 2001.

Adrián Csiszárik, Péter Kőrösi-Szabó, Akos Matszangosz, Gergely Papp, and Dániel Varga. Similarity and matching of neural network representations. *Advances in Neural Information Processing Systems*, 34:5656–5668, 2021.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

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.

Yena Han, Tomaso A Poggio, and Brian Cheung. System identification of neural systems: If we got it right, would we know? In *International conference on machine learning*, pp. 12430–12444. PMLR, 2023.

Bobby He and Mete Ozay. Feature kernel distillation. In *International Conference on Learning Representations*, 2022.

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.

Minyoung Huh, Brian Cheung, Tongzhou Wang, and Phillip Isola. The platonic representation hypothesis. *ICML*, 2024.

Yerlan Idelbayev. Proper ResNet implementation for CIFAR10/CIFAR100 in PyTorch. `https://github.com/akamaster/pytorch_resnet_cifar10`. Accessed: 2026-01-15.

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.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Alex Krizhevsky et al. Learning multiple layers of features from tiny images. 2009.

Yuang Liu, Wei Zhang, and Jun Wang. Adaptive multi-teacher multi-level knowledge distillation. *Neurocomputing*, 415:106–113, 2020.

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.

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

Michael S Matena and Colin A Raffel. Merging models with fisher-weighted averaging. *Advances in Neural Information Processing Systems*, 35:17703–17716, 2022.

Luca Moschella, Valentino Maiorca, Marco Fumero, Antonio Norelli, Francesco Locatello, and Emanuele Rodolà. Relative representations enable zero-shot latent space communication. *arXiv preprint arXiv:2209.15430*, 2022.

Antonio Norelli, Marco Fumero, Valentino Maiorca, Luca Moschella, Emanuele Rodola, and Francesco Locatello. Asif: Coupled data turns unimodal models to multimodal without training. *Advances in Neural Information Processing Systems*, 36:15303–15319, 2023.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Mannat Singh, Quentin Duval, Kalyan Vasudev Alwala, Haoqi Fan, Vaibhav Aggarwal, Aaron Adcock, Armand Joulin, Piotr Dollár, Christoph Feichtenhofer, Ross Girshick, et al. The effectiveness of mae pre-pretraining for billion-scale pretraining. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 5484–5494, 2023.

Le Song, Alex Smola, Arthur Gretton, Karsten M Borgwardt, and Justin Bedo. Supervised feature selection via dependence estimation. In *Proceedings of the 24th international conference on Machine learning*, pp. 823–830, 2007.

Vighnesh Subramaniam, Colin Conwell, Boris Katz, Andrei Barbu, and Brian Cheung. Network of theseus (like the ship). *arXiv preprint arXiv:2512.04198*, 2025a.

Vighnesh Subramaniam, David Mayo, Colin Conwell, Tomaso Poggio, Boris Katz, Brian Cheung, and Andrei Barbu. Training the untrainable: Introducing inductive bias via representational alignment. *Advanced in neural information processing systems*, 38, 2025b.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. *arXiv preprint arXiv:1910.10699*, 2019.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pp. 10347–10357. PMLR, 2021.

Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1365–1374, 2019.

Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Scan: Learning to classify images without labels. In *European conference on computer vision*, pp. 268–285. Springer, 2020.

Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. In *International Conference on Learning Representations*, 2020.

Tom White. Sampling generative networks. *arXiv preprint arXiv:1609.04468*, 2016.

Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019.

Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. Convnext v2: Co-designing and scaling convnets with masked autoencoders. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16133–16142, 2023.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pp. 23965–23998. PMLR, 2022.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems*, 36:7093–7115, 2023.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. In *The Thirteenth International Conference on Learning Representations*, 2025.

A Appendix

A.1 Multi-architecture and Multi-task settings

Multi-architecture All networks are trained on ImageNet-1K (Deng et al., 2009). All pretrained checkpoints are taken from TorchVision (maintainers & contributors, 2016)

1. ResNets (He et al., 2016): ResNet-18, ResNet-50, ResNet-101. These networks are identical to what is described in He et al. (2016).
2. ViTs (Dosovitskiy et al., 2020): ViT-B/16, ViT-B/32. These networks are identical to what is described in Dosovitskiy et al. (2020).
3. ConvNeXt (Liu et al., 2022): ConvNeXt-Tiny, ConvNeXt-Small. These networks are identical to what is described in Liu et al. (2022).

When reporting averaging, we use the best-performing network, ConvNeXt-Small, as our oracle. This network achieves the highest top-1 ImageNet accuracy.

Multi-Task: We use the following networks for our mixed-task setting. Each is pretrained and taken from Wightman (2019).

1. ViT-B-MAE (Dosovitskiy et al., 2020): A ViT trained with MAE rather than supervised training with ImageNet-1K (Singh et al., 2023).
2. CLIP (ViT-B/16) (Radford et al., 2021): A ViT trained with contrastive language-image pretraining.
3. ResNet-50 (He et al., 2016): Same weights and setting as multi-architecture.
4. DEIT-Base Distilled (Touvron et al., 2021): ViT trained using a data-efficient distillation procedure on ImageNet-1K.
5. DINOv2 Base (Oquab et al., 2023): Pretrained ViT, trained using discriminative self-supervised pretraining.
6. CNXV2-Base (FCMAE) (Woo et al., 2023): ConvNeXt-V2 trained using a fully-convolutional masked autoencoder (FCMAE) loss on ImageNet-22K.

A.2 Training details

For UCKA and CKA, we report performances when guidance is performed on the second-to-last layer of the network; we find that this results in better accuracy than guidance on all layers.

For ImageNet100, the oracle was trained with 100K examples. Each teacher/student/baseline was trained on a tenth of data (10K examples). Data subset construction is described in section 5.2 and A.5, and we will add these details there. Models were initialized with Kaiming initialization, trained for 200 epochs, using an SGD optimizer with weight decay of 1e-4 and a learning rate of 0.1.

During training, since we are mainly interested in testing whether *any* representational similarity guidance is useful, we use an equal weighting between L_{sim} and L_{CE}.

Training for the ResNet18 models was done on a machine with 8 NVIDIA TITAN RTX GPUs with 24GB. Training for larger models was done on a single H100.

A.3 Differentiable Mutual Nearest Neighbors

We adapt the appendices of Subramaniam et al. (2025a) and present them here for ease of reference.

For Differentiable MNN (D-MNN), we use Huh et al. (2024) and Alshammari et al. (2025) to reformulate nearest neighbors into a differentiable form.

Let b be the mini-batch size and anchors $i \in \{1, \dots, b\}$. Two encoders f_1, f_2 produce features $\phi_i = f_1(x_i) \in \mathbb{R}^d$, $\psi_i = f_2(y_i) \in \mathbb{R}^d$. Define $\mathcal{J}_i = \{1, \dots, b\} \setminus \{i\}$. For $k \in \mathbb{N}$, let $s_{f_1}(i), s_{f_2}(i) \subset \mathcal{J}_i$ be the index sets of the k nearest neighbors of i under $\{\phi_j\}$ and, $\{\psi_j\}$, respectively.

k-Nearest Neighbors: As defined by Huh et al. (2024), the overlap score for anchor i is

$$g_{NN}(\phi_i, \psi_i) = \frac{|s_{f_1}(i), s_{f_2}(i)|}{k} \tag{6}$$

and the batch score is the average over i.

From overlap to probabilities: We introduce hard conditional distributions uniform on the kNN sets:

$$p_{f_1}(j \mid i) = \frac{\mathbf{1}\{j \in s_{f_1}(i)\}}{k}, \qquad q_{f_2}(j \mid i) = \frac{\mathbf{1}\{j \in s_{f_2}(i)\}}{k}, \quad j \in \mathcal{J}_i. \tag{7}$$

Then

$$\sum_{j \in \mathcal{J}_i} p_{f_1}(j \mid i)\, q_{f_2}(j \mid i) = \frac{|s_{f_1}(i) \cap s_{f_2}(i)|}{k^2}, \quad \Rightarrow \quad g_{NN}(\phi_i, \psi_i) = k \sum_{j \in \mathcal{J}_i} p_{f_1}(j \mid i)\, q_{f_2}(j \mid i). \tag{8}$$

Differentiable conditional neighborhoods: Let $\tau > 0$ be a temperature. Define pairwise similarities either by

$$s_{ij}^f = -\frac{\|\phi_i - \phi_j\|_2^2}{\tau} \quad \text{or} \quad s_{ij}^f = \frac{\phi_i^\top \phi_j}{\tau}, \qquad j \in \mathcal{J}_i, \tag{9}$$

and analogously s^g_{ij} from $\{\psi_j\}$. Let $T^f_k(i) \subset \mathcal{J}_i$ and $T^g_k(i) \subset \mathcal{J}_i$ denote top-k index selections under s^f_{ij} and s^g_{ij}, respectively. We define masked, normalized conditionals over the top-k:

$$P_{ij} \equiv p_{f_1}(j \mid i) = \frac{\exp(s^{f_1}_{ij})\,\mathbf{1}\{j \in T^{f_1}_k(i)\}}{\sum_{j' \in T^{f_1}_k(i)} \exp(s^{f_1}_{ij'})}, \qquad Q_{ij} \equiv q_{f_2}(j \mid i) = \frac{\exp(s^{f_2}_{ij})\,\mathbf{1}\{j \in T^{f_2}_k(i)\}}{\sum_{j' \in T^{f_2}_k(i)} \exp(s^{f_2}_{ij'})}. \tag{10}$$

(Any differentiable soft-top-k operator may be used; with an exact mask and $\tau \to 0$, P_{ij} and Q_{ij} converge to the hard uniforms.)

D-MNN alignment and limit: We define the differentiable alignment for anchor i as the inner product

$$g_{\text{soft}}(\phi_i, \psi_i) \;=\; \sum_{j \in \mathcal{J}_i} P_{ij}\, Q_{ij}, \qquad \bar{g}_{\text{soft}} \;=\; \frac{1}{b}\sum_{i=1}^{b} m_{\text{soft}}(\phi_i, \psi_i). \tag{11}$$

Under the exact top-k mask and $\tau \to 0$, $P_{ij}, Q_{ij} \to \frac{1}{k}\mathbf{1}\{j \in s_{f_1}(i)\}, \frac{1}{k}\mathbf{1}\{j \in s_{f_2}(i)\}$, so that $k\, g_{\text{soft}}(\phi_i, \psi_i) \to g_{NN}(\phi_i, \psi_i)$ by equation 8.

Training objective: Excluding the anchor, we write $P_{i\setminus *} = \{P_{ij}\}_{j \in \mathcal{J}_i}$ and similarly for Q. When P (from f) guides Q (from g), we minimize

$$\Delta_{\text{D-MNN}} \;:=\; \mathcal{L}_{KL} \;=\; \frac{1}{b}\sum_{i=1}^{b} D_{KL}\Big(P_{i\setminus *} \,\Big\|\, Q_{i\setminus *}\Big). \tag{12}$$

This KL aligns conditional neighborhood distributions. Most importantly, before computing g, rows of $\boldsymbol{A}$ and $\boldsymbol{B}$ are ℓ_2-normalized. For our setting in this paper, we have to tune the temperature for D-MNN.

A.4 Centered Kernel Alignment

To compare representations, we use a representation similarity metric, $\mathcal{M}$, which corresponds to centered kernel alignment (CKA) (Kornblith et al., 2019; Cortes et al., 2012; Cristianini et al., 2001) in our setting. We specifically consider linear CKA.

CKA uses kernel functions on mean-centered representations to compute representational similarity matrices, which are then compared via the Hilbert-Schmidt Independence Criterion (HSIC). More specifically, suppose we have two sets of representations $\boldsymbol{A} \in \mathbb{R}^{b \times d_1}$ and $\boldsymbol{B} \in \mathbb{R}^{b \times d_2}$. We first compute the Gram matrices for each set of representations

$$\boldsymbol{K} = \boldsymbol{A}\boldsymbol{A}^T, \boldsymbol{L} = \boldsymbol{B}'\boldsymbol{B}'^T \tag{13}$$

We center the Gram matrices by introducing a matrix, H, where $H = \boldsymbol{I}_b - \frac{1}{b}\mathbf{1}\mathbf{1}^T$.

$$\tilde{\boldsymbol{K}} = \boldsymbol{H}\boldsymbol{K}\boldsymbol{H}, \tilde{\boldsymbol{L}} = \boldsymbol{H}\boldsymbol{L}\boldsymbol{H} \tag{14}$$

We compute the HSIC on the Gram matrices.

$$HSIC(\boldsymbol{K}, \boldsymbol{L}) = \text{tr}(\tilde{\boldsymbol{K}}, \tilde{\boldsymbol{L}}) \tag{15}$$

Finally, we define our linear CKA metric as:

$$g_{\text{CKA}}(\boldsymbol{A}, \boldsymbol{B}') := \text{CKA}(\boldsymbol{K}, \boldsymbol{L}) = \frac{HSIC(\boldsymbol{K}, \boldsymbol{L})}{\sqrt{HSIC(\boldsymbol{K}, \boldsymbol{K}) * HSIC(\boldsymbol{L}, \boldsymbol{L})}} \tag{16}$$

In our setting, we consider representational *dissimilarity* and aim to minimize the dissimilarity between representations from our target network and guide network. We define this as:

$$\Delta_{\text{CKA}}(\boldsymbol{A}, \boldsymbol{B}) = 1 - g(\boldsymbol{A}, \boldsymbol{B}) \tag{17}$$

Linear CKA ranges from 0 to 1 (very different representations). Because of this, we take the complement by subtracting the linear CKA from 1 to represent dissimilarity. Most importantly, before computing g, rows of $\boldsymbol{A}$ and $\boldsymbol{B}$ are ℓ_2-normalized.

A.4.1 UNBIASED CKA

We cover the unbiased HSIC (Song et al., 2007) here.

We start with our Gram matrices, $\boldsymbol{K}, \boldsymbol{L}$. An unbiased estimator for HSIC then involves forming zero-diagonal copies of both matrices.

$$\tilde{\boldsymbol{K}} \coloneqq \boldsymbol{K} - \operatorname{diag}(\boldsymbol{K}), \qquad \tilde{\boldsymbol{L}} \coloneqq \boldsymbol{L} - \operatorname{diag}(\boldsymbol{L}), \tag{18}$$

so that $\tilde{\boldsymbol{K}}_{ii} = \tilde{\boldsymbol{L}}_{ii} = 0$ and $\tilde{\boldsymbol{K}}_{ij} = \boldsymbol{K}_{ij}$, $\tilde{\boldsymbol{L}}_{ij} = L_{ij}$ for $i \neq j$. Let $\mathbf{1} \in \mathbb{R}^m$ be the all-ones vector. The unbiased estimator of HSIC is

$$HSIC_U(K, L) = \frac{1}{m(m-3)} \left[\operatorname{tr}(\tilde{\boldsymbol{K}}\tilde{\boldsymbol{L}}) + \frac{\operatorname{tr}(\tilde{\boldsymbol{K}}\tilde{\boldsymbol{L}})}{(m-1)(m-2)} - \frac{2}{m-2} \operatorname{tr}(\tilde{\boldsymbol{K}}\tilde{\boldsymbol{L}}) \right], \tag{19}$$

which is well-defined for $m \geq 4$. The last term corresponds to the unbiased correction.

A.5 PROOF OF OBSERVATION 1

Proof: For ease of discussion, consider a single element of the dataset x_i. Let $\boldsymbol{K}_{min} := \min_{j \neq k}(\boldsymbol{K}^*_{i,j} - \boldsymbol{K}^*_{i,k})$ be the smallest gap between distances to x_i as given by the platonic representations. Let $r_{\boldsymbol{K}}(j)$ be the ranking of element x_j among the neighbors of x_i, ordered by distance, so that $r_{\boldsymbol{K}}(j) < r_{\boldsymbol{K}}(k)$ iff $\boldsymbol{K}_{i,j} < \boldsymbol{K}_{i,k}$. By the law of large numbers, $\lim_{N\to\infty} P(|\boldsymbol{K}_{i,j} - \overline{\boldsymbol{K}}_{i,j}| < \boldsymbol{K}_{min}) = 1$. Then, by the union bound, $\lim_{N\to\infty} P(r_{\overline{\boldsymbol{K}}}(i) = r_{\boldsymbol{K}}(i)) = 1$, which implies that the neighborhoods will be equal. □

This comes from the chain rule for mutual information and the non-negativity of mutual information:

$$I\left(\boldsymbol{K}^*_{i,j}; \boldsymbol{K}^{N+1}_{i,j}, \frac{1}{N}\sum_{k}^{N} \boldsymbol{K}^k_{i,j}\right) = I\left(\boldsymbol{K}^*_{i,j}; \frac{1}{N}\sum_{k=1}^{N} \boldsymbol{K}^k_{i,j}\right) + I\left(\boldsymbol{K}^*_{i,j}; \boldsymbol{K}^{N+1}_{i,j} \;\middle|\; \frac{1}{N}\sum_{k=1}^{N} \boldsymbol{K}^k_{i,j}\right)$$

If it can be assumed that $D_{i,j} \sim \mathcal{N}(\boldsymbol{K}^*_{i,j}, \sigma)$, then the sample mean is a sufficient statistic for $\boldsymbol{K}^*_{i,j}$ and $I\left(\boldsymbol{K}^*_{i,j}; \frac{1}{N}\sum_{k=1}^{N} \boldsymbol{K}^k_{i,j}\right) \leq I\left(\boldsymbol{K}^*_{i,j}; \frac{1}{N+1}\sum_{k}^{N+1} \boldsymbol{K}^k_{i,j}\right)$.

A.6 PROOF OF OBSERVATION 2

This comes from the chain rule for mutual information and the non-negativity of mutual information:

$$I\left(\boldsymbol{K}^*_{i,j}; \boldsymbol{K}^{N+1}_{i,j}, \frac{1}{N}\sum_{k}^{N} \boldsymbol{K}^k_{i,j}\right) = I\left(\boldsymbol{K}^*_{i,j}; \frac{1}{N}\sum_{k=1}^{N} \boldsymbol{K}^k_{i,j}\right) + I\left(\boldsymbol{K}^*_{i,j}; \boldsymbol{K}^{N+1}_{i,j} \;\middle|\; \frac{1}{N}\sum_{k=1}^{N} \boldsymbol{K}^k_{i,j}\right)$$

If it can be assumed that $D_{i,j} \sim \mathcal{N}(\boldsymbol{K}^*_{i,j}, \sigma)$, then the sample mean is a sufficient statistic for $\boldsymbol{K}^*_{i,j}$ and $I\left(\boldsymbol{K}^*_{i,j}; \frac{1}{N}\sum_{k=1}^{N} \boldsymbol{K}^k_{i,j}\right) \leq I\left(\boldsymbol{K}^*_{i,j}; \frac{1}{N+1}\sum_{k}^{N+1} \boldsymbol{K}^k_{i,j}\right)$.

A.7 PROOF OF OBSERVATION 3

Proof: The proof is similar to the proof of observation 1. Suppose that the average neighborhood is a biased estimator of the true neighborhood under the platonic representation. Then, the best achievable g_{MKNN} is p. That is, there is at most p fraction of the overlap between the average neighborhood and the true neighborhood. Let x_i be some element in this overlap. Then by the same argument as made in the proof of observation 1, the rank of x_i, $r_{\overline{\boldsymbol{K}}(x_i)}$ will be correct with probability 1 as $N \to \infty$.

A.8 SPECIALIST DATASETS

Classes in CIFAR10 specialist splits

Split	Classes
1	airplane, automobile
2	bird, cat
3	deer, dog
4	frog, horse
5	ship, truck

Classes in ImageNet100 specialist splits

Split	Classes
1	cougar, giant schnauzer, dalmatian, Staffordshire bullterrier, coyote, grey fox, schipperke, guenon, leopard, Norfolk terrier
2	toy terrier, cairn, proboscis monkey, guinea pig, Scotch terrier, Doberman, collie, Border collie, flat-coated retriever, Arctic fox
3	American black bear, Shetland sheepdog, bluetick, wood rabbit, Irish setter, Border terrier, tusker, trimaran, wreck, bicycle-built-for-two
4	recreational vehicle, unicycle, table lamp, file, banana, rapeseed, organ, upright, electric guitar, oboe
5	hatchet, plane, lawn mower, robin, water ouzel, goose, American coot, oystercatcher, albatross, box turtle
6	sea snake, diamondback, sidewinder, wing, hand blower, microphone, analog clock, computer keyboard, thresher, printer
7	vending machine, neck brace, scorpion, ringlet, washer, steel arch bridge, stupa, planetarium, obelisk, totem pole
8	shoe shop, tobacco shop, dam, picket fence, honeycomb, mashed potato, shower curtain, ashcan, manhole cover, bagel
9	knee pad, crash helmet, pop bottle, trifle, guacamole, bucket, broom, mitten, envelope, cowboy hat
10	gown, maze, pickelhaube, Windsor tie, purse, sleeping bag, stretcher, maillot, stole, dumbbell

Table 2: Classes in specialist splits

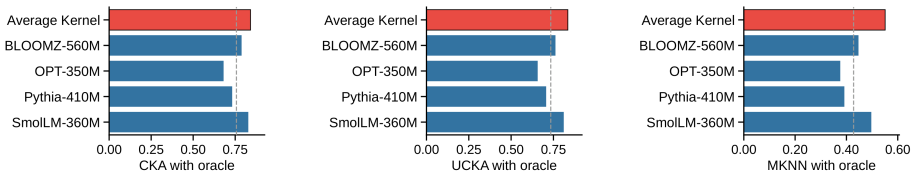

Figure 7: **Alignment of language models with a language oracle.** CKA, UCKA, and MKNN are shown from left to right. The average kernel (red) achieves higher alignment than every individual language model (blue) across all three metrics, showing that averaging also improves agreement with an oracle within the same modality. Dashed lines indicate the mean score across the four individual models.

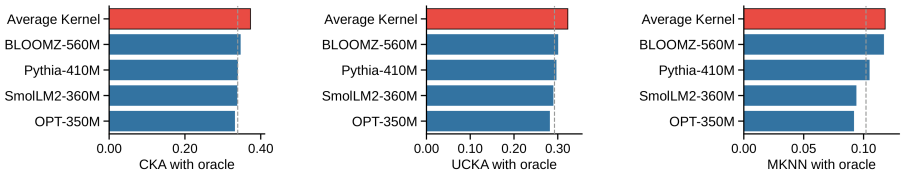

Figure 8: **Alignment of language models with a vision oracle.** CKA, UCKA, and MKNN are shown from left to right. The average kernel (red) achieves higher alignment than any individual language model (blue) across all three metrics, although the improvement over the best individual model is small for MKNN. Dashed lines indicate the mean score across the four individual models.

A.9 Multimodal Kernel Averaging

We study kernel averaging in richer settings beyond our mixed-task setting. First, we apply kernel averaging to language models, pretrained on different datasets. Our experiment uses Wikipedia captions extracted from Huh et al. (2024). Our oracle model is Qwen 3 0.6B (Yang et al., 2025). See Figure 7. As in our other settings, we find that the average kernel significantly out-performs other model kernels when compared to the oracle kernel. This is prominently shown in MKNN, where we improve by 0.06.

We then apply kernel across multimodal settings. In on setting, we average together kernels from language models and compare them with a vision oracle. And in the other setting, we average together vision kernels and compare them with a language oracle. As with our language setting, we borrow the experimental setting from Huh et al. (2024), which uses image-text pairs from Wikipedia. For each setting the images are fed to the vision models and the corresponding text are fed to the language model. We compute representational similarity across all pairs of layers and take a max over scores. Our set of language models include BLOOMZ-560M, Pythia-410M, SmolLM2-360M, and OPT-350M. The vision oracle we compared these language models with is DINOv2 (ViT-L/14). See Figure 8. Across a majority of settings, the average kernels of the language models are more similar to the vision oracle than the individual models. Likewise, our set of vision models include ViT-B/16 (AugReg), ViT-L/16 (AugReg), ViT-B/16 (MAE), ViT-L (MAE). The language oracle we compare

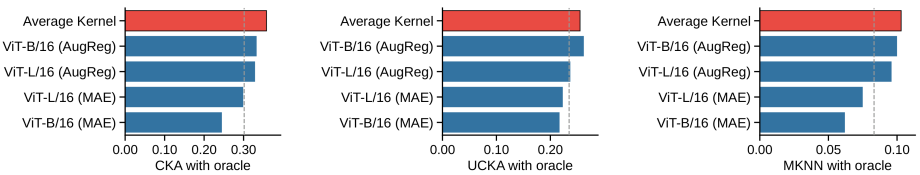

Figure 9: **Alignment of vision models with a language oracle.** CKA, UCKA, and MKNN are shown from left to right. The average kernel (red) achieves the highest alignment under CKA and MKNN, while ViT-B/16 (AugReg) retains the highest UCKA. Individual vision models are shown in blue, with dashed lines indicating their mean score.

these vision models with is Qwen-3-0.6B. See Figure 9. Across a majority of settings, the average kernels of the vision models are more similar to the language oracle than the individual models.

A.10 ADDITIONAL FIGURES

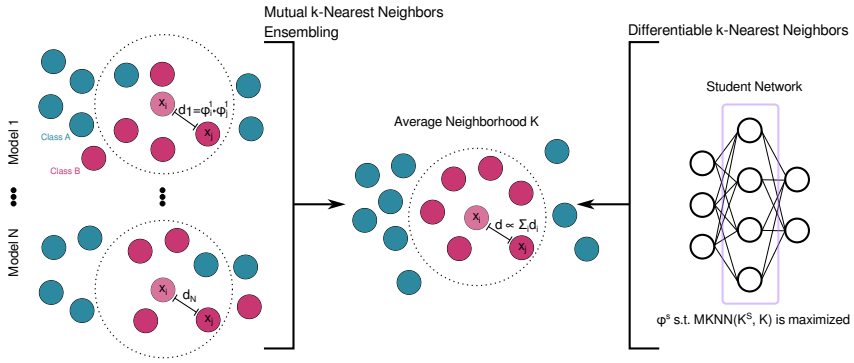

Figure 10: **Schematic overview of our approach**: (*Left*) Given an ensemble of models, each model gives rise to a neighborhood around every dataset element x_i. For individual models, neighborhoods may be imperfect; that is, the closest elements to x_i may be semantically incorrect. (*Center*) However, averaging kernels across models can smooth out these mistakes and produce distances that give rise to more correct neighborhoods. (*Right*) We can then use measures such as Differentiable k-Nearest Neighbors (DiffKNN) to train student models such that their representations ϕ are optimized towards producing this average neighborhood.

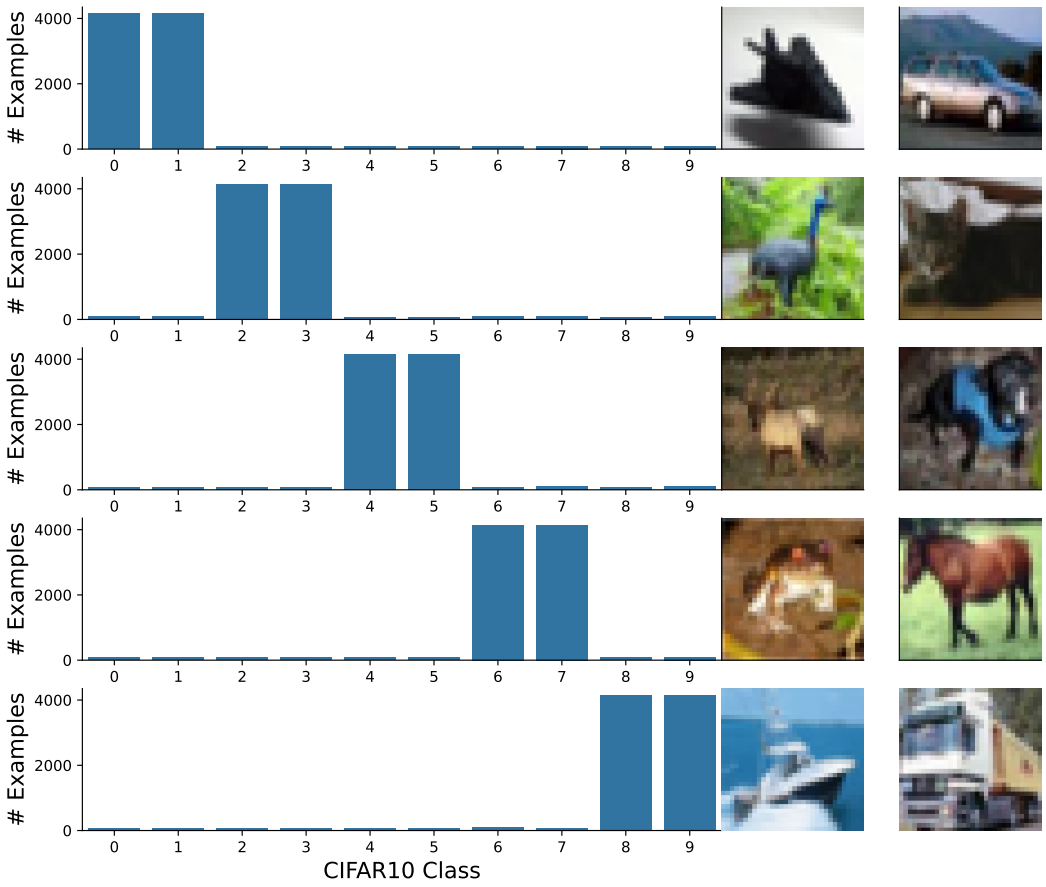

Figure 11: **CIFAR10 specialist splits**. To study the benefit of combining knowledge from models that have been trained on different data distributions, we divide the datasets into *specialist* sub-datasets, each of which is separately skewed ($> 99\%$) towards two classes. Shown here are the distributions of the specialist splits (left) and dataset samples (right) for CIFAR10. Similarly, we divide ImageNet100 into 10 specialist splits, which are each skewed to a separate set of 10 classes.

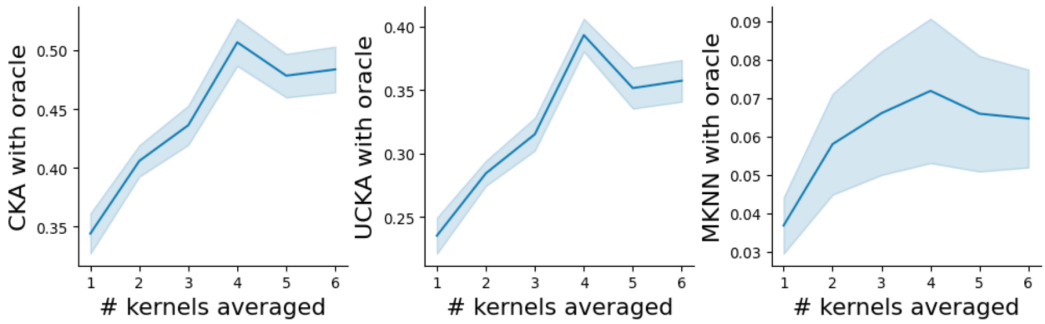

Figure 12: **Averaging kernels from different network architectures improves representational similarity with a more accurate model.** Each kernel is constructed using a different network architecture using the ImageNet-100 dataset where the order corresponds to ResNet-18, ViT-B/32, ResNet-50, ResNet-101, ViT-B/16, and ConvNeXt-Tiny. The oracle model corresponds to the best performing network based on ImageNet-1K performance, ConvNeXt-Small. The same metrics as Figure 2 are calculated using the same method.

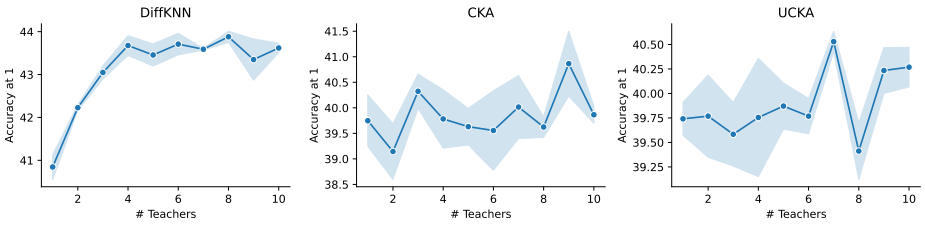

Figure 13: **Accuracy as a function of ensemble size for specialist ImageNet100 teachers** Same results as Figure 4, but for teachers with specialist distributions on ImageNT100. Bands show standard error across seeds ($n = 3$).

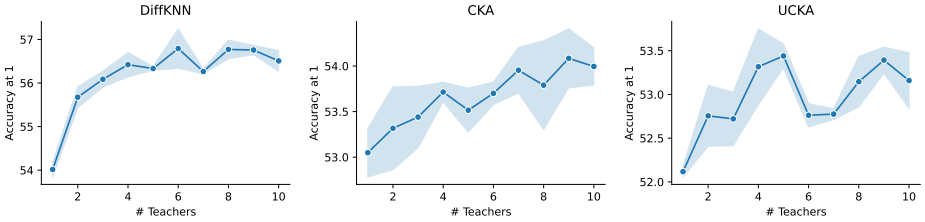

Figure 14: **Accuracy as a function of ensemble size for balanced ImageNet100 teachers** Same results as Figure 4, but for teachers with balanced distributions on ImageNET100. Bands show standard error across seeds ($n = 3$).

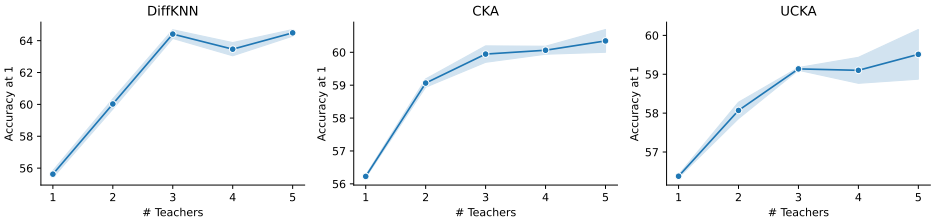

Figure 15: **Accuracy as a function of ensemble size for specialist CIFAR10 teachers** Same results as Figure 4, but for teachers with specialist distributions on CIFAR10. Bands show standard error across seeds ($n = 3$).

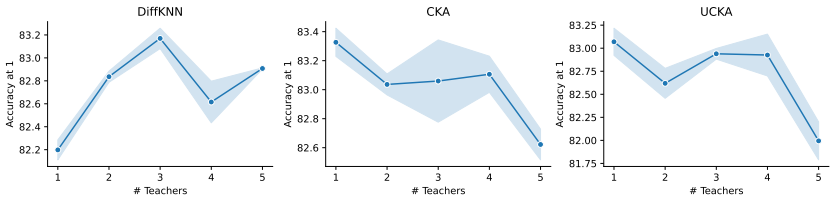

Figure 16: **Accuracy as a function of ensemble size for balanced CIFAR10 teachers** Same results as Figure 4, but for teachers with balanced distributions on CIFAR10. Bands show standard error across seeds ($n = 3$).

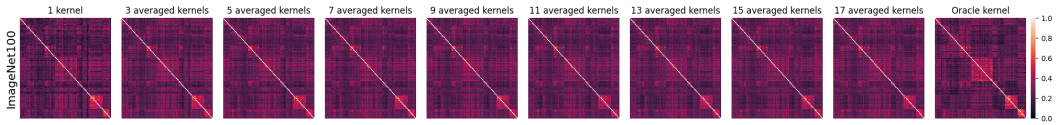

Figure 17: **The progression of averaging all 10 specialist teachers.** Teachers were trained on ImageNet100.

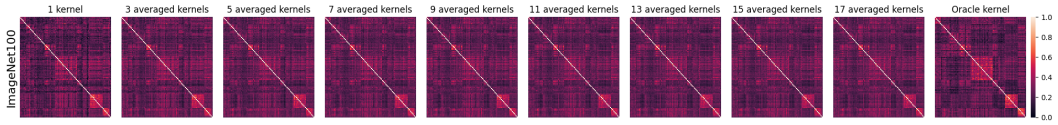

Figure 18: **The progression of averaging all 10 balanced teachers.** Teachers were trained on ImageNet100.

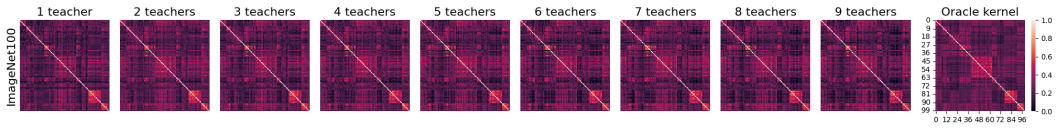

Figure 19: **The progression of a student kernel, progressively trained using all 10 specialist teachers.** Teachers were trained on ImageNet100.

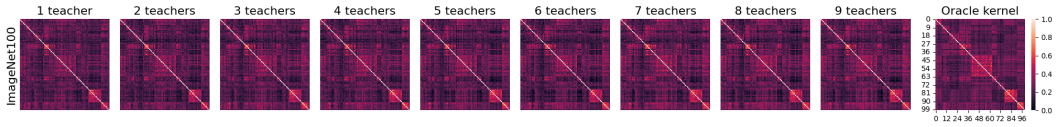

Figure 20: **The progression of a student kernel, progressively trained using all 10 balanced teachers.** Teachers were trained on ImageNet100.

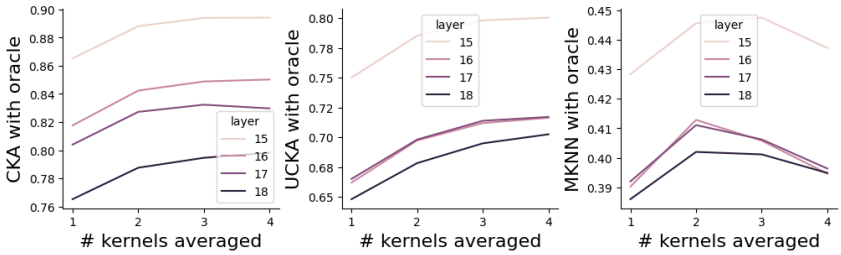

Figure 21: **CIFAR10 similarity trends for specialist teachers.** Trends shown for last 4 network layers.

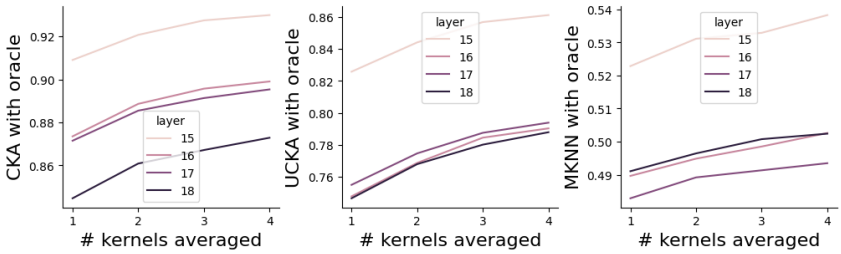

Figure 22: **CIFAR10 similarity trends for balanced teachers.** Trends shown for last 4 network layers.

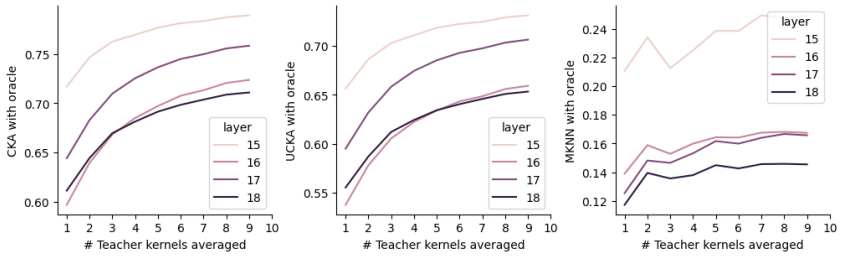

Figure 23: **ImageNet100 similarity trends for specialist teachers.** Trends shown for last 4 network layers.

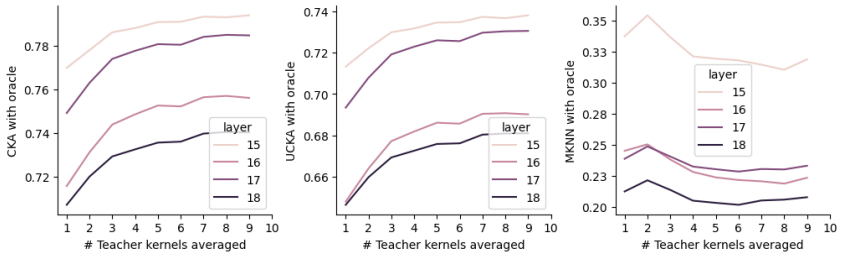

Figure 24: **ImageNet100 similarity trends for balanced teachers.** Trends shown for last 4 network layers.

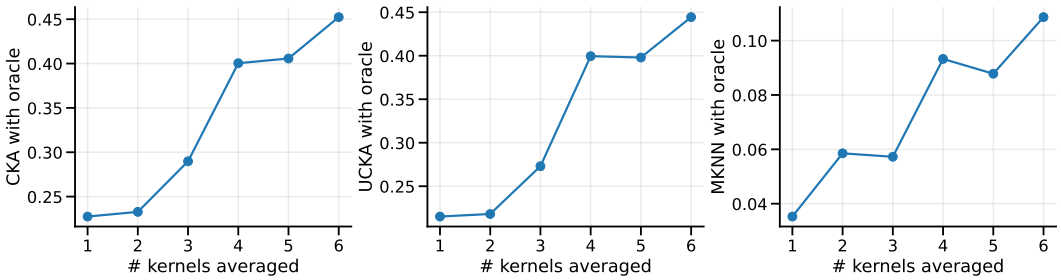

Figure 25: **Multitask averaging**: Order of models is: ViT-B/16 (MAE), ViT-B/16 (CLIP), ResNet-50 (ImageNet), DeIT-Base, DINOv2-Base, and ConvNeXt-Base (FCMAE pretrained) and the oracle model is ConvNeXt-XL (IN-1K trained).

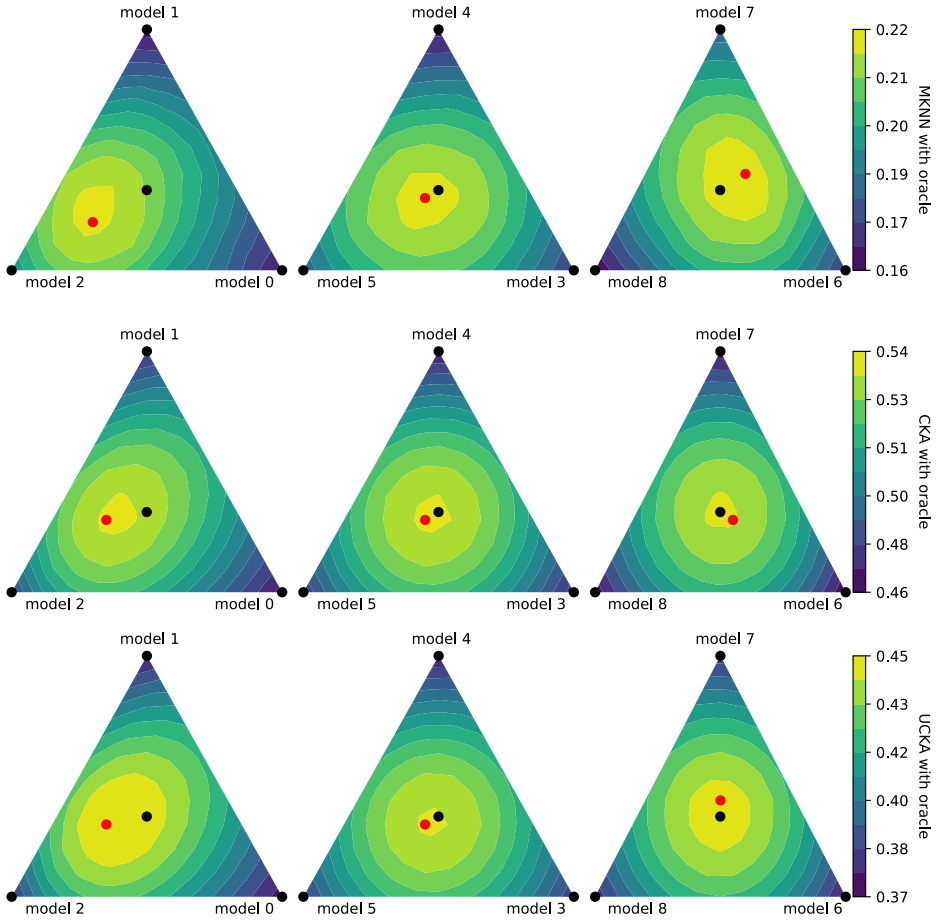

Figure 26: **Similarity landscape appears convex with respect to constituent kernels**. Each vertex of each triangle represents a specialist model's kernel. Each interior point of a triangle shows the oracle similarity with the kernel of a convex weighted average of constituent kernels. The weighting between three teacher models that yields the highest similarity with the oracle (red) is usually close to the barycenter (black), and the landscape appears highly convex.

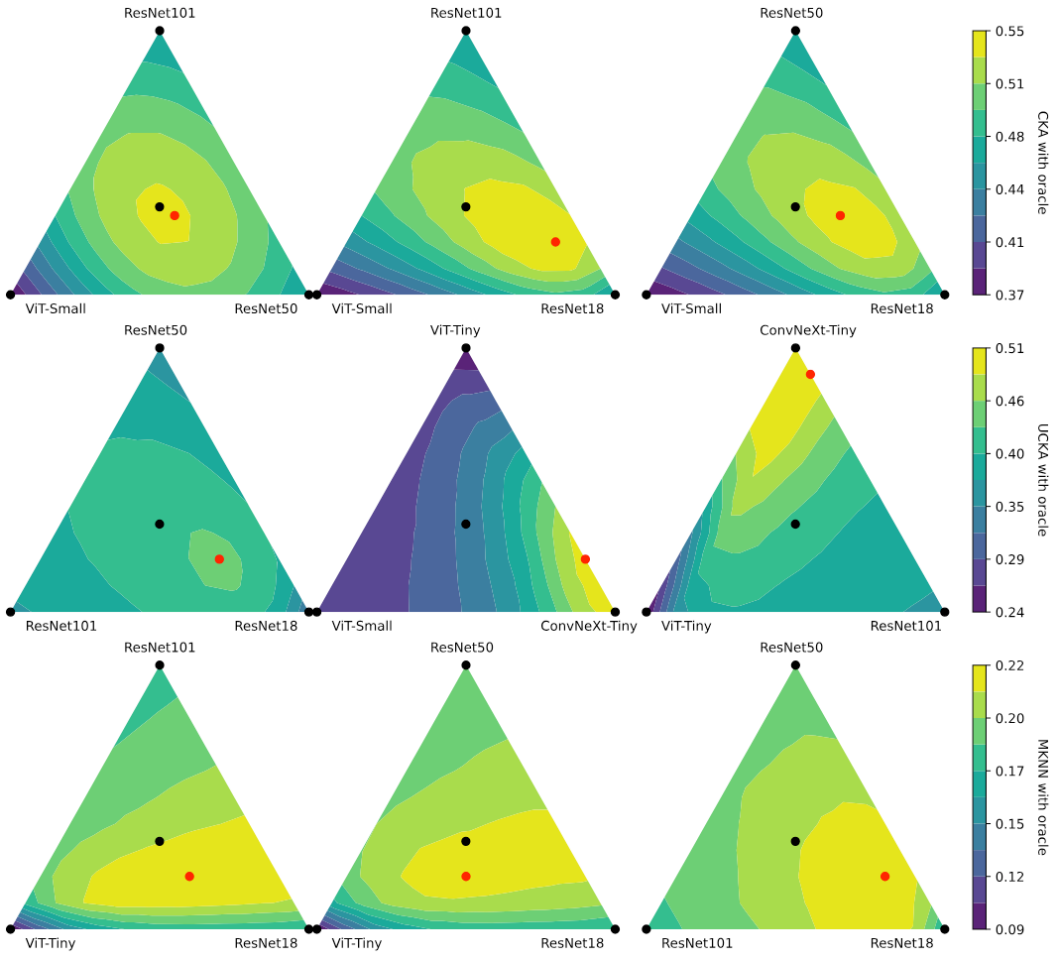

Figure 27: **Similarity landscape of different network architectures appears convex with respect to constituent kernels.** Each vertex of each triangle represents a kernel from a *different* model network architecture. Similar to Figure 3, we see that weighting of the kernels between teacher models that yields the highest similarity with the oracle (red) is usually close to the barycenter (black) and the landscape appears highly convex. Our results are also influenced by model performance by the pretrained networks selected.

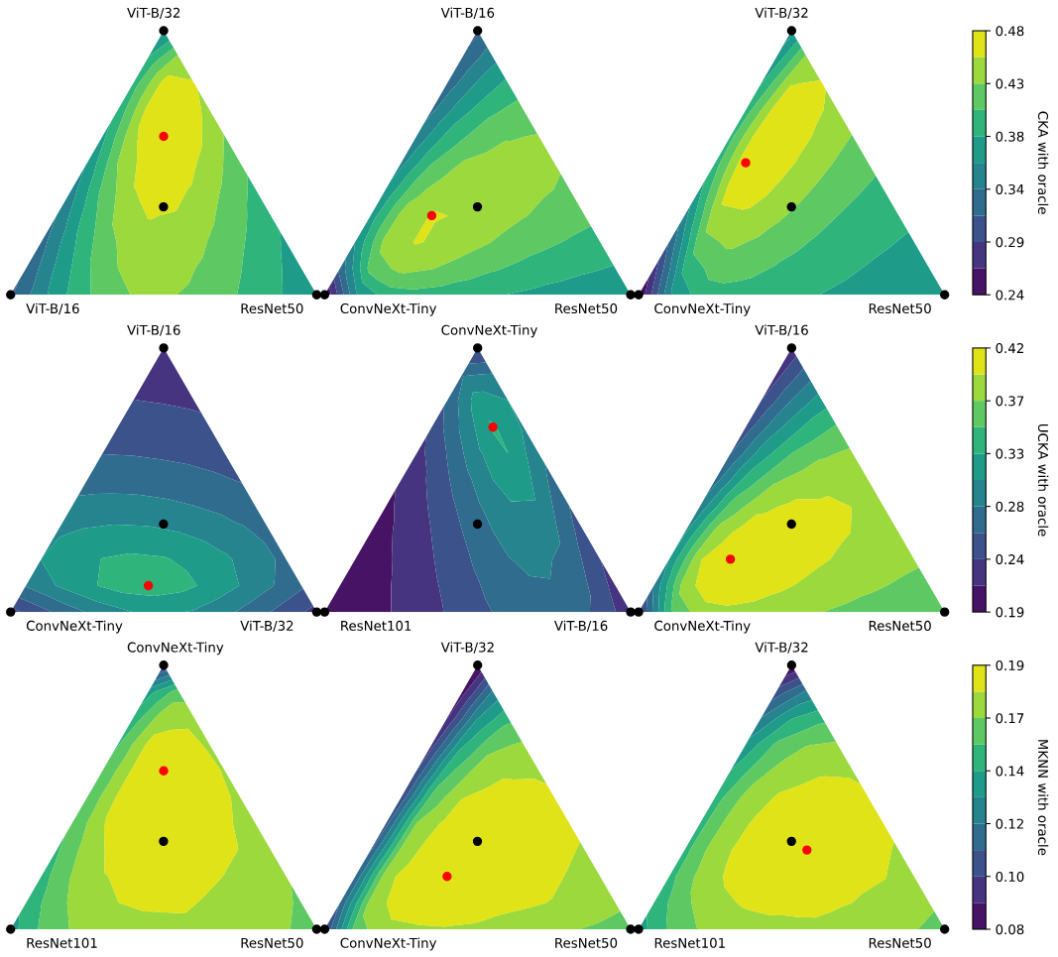

Figure 28: **Controlling for accuracy still yields convex landscapes when averaging kernels across multiple architectures.** To test the role of performance in our multiarchitecture experiments, we train each network architecture to the same accuracy on a different split of ImageNet-100 and then reproduce Figure 27. We can see that networks are still convex and the best performing weighted kernel (red) is close to the barycenter (black).