DISSERTATION

REVEALING AND ANALYZING THE SHARED STRUCTURE OF DEEP FACE EMBEDDINGS

Submitted by

David G. McNeely-White

Department of Computer Science

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Summer 2022

Doctoral Committee:

Advisor: J. Ross Beveridge

Nathaniel Blanchard
Michael Kirby
Chris Peterson

ABSTRACT

REVEALING AND ANALYZING THE SHARED STRUCTURE OF DEEP FACE EMBEDDINGS

Deep convolutional neural networks trained for face recognition are found to output face embeddings which share a fundamental structure. More specifically, one face verification model's embeddings (i.e. last–layer activations) can be compared directly to another model's embeddings after only a rotation or linear transformation, with little performance penalty. If only rotation is required to convert the bulk of embeddings between models, there is a strong sense in which those models are learning the same thing. In the most recent experiments, the structural similarity (and dissimilarity) of face embeddings is analyzed as a means of understanding face recognition bias. Bias has been identified in many face recognition models, often analyzed using distance measures between pairs of faces. By representing groups of faces as groups, and comparing them as groups, this shared embedding structure can be further understood. Specifically, demographic-specific subspaces are represented as points on a Grassmann manifold. Across 10 models, the geodesic distances between those points are expressive of demographic differences. By comparing how different groups of people are represented in the structure of embedding space, and how those structures vary with model designs, a new perspective on both representational similarity and face recognition bias is offered.

ACKNOWLEDGEMENTS

First, I would like to thank the CSU Computer Science department faculty, staff, and students, who tirelessly foster an environment for learning and research. Special thanks to the past and current members of the vision group, who were among the first to welcome me into the computer vision community. I would also like to thank Martha Palmer and the members of the RAMFIS and iSAT teams at CU Boulder for your unflinching support and inclusion. Thanks also to my teammates at Amazon Science, for supporting me in this final year of grad school.

Next, I am extremely thankful for my friends and family. You all offered your sympathies (and condolences) at every opportunity, keeping me from feeling overwhelmed by the rigors of graduate school. Though I feel far from home, your support kept me going. Special thanks to Ben Sattelberg for his instrumental support both as a collaborator and friend.

I would also like to thank my committee members for their continued support in this process, freely offering new perspectives and ideas which are critical to this process. Our meetings consistently gave me encouragement and inspiration. I am especially grateful for Professor Ross Beveridge's service as my advisor. Ross is an exceptional scientist, professor, advisor, colleague, and (despite my best efforts) friend. Without your keen guidance and relentless encouragement, I would still be doubting myself and my work. Moreover, I may have never finished! In all seriousness, in the relentless world of academic research, Ross is a beacon of goodwill. I am honored to have worked with him.

Finally, I cannot overstate the support of my wife and closest friend, the soon-to-be-Dr. Katherine McNeely-White. Whether it be bouncing research ideas, commiserating about incoherent reviewers, or simply decompressing after another long week, you were undaunted by my requests for support. In good times and in bad, you are there for me. Soon we will look back on this time with sanguine affection, but I'll never forget how little I would've accomplished without you. Kat, you are my rock. I can't wait for our next journey together.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

The deep learning revolution is well underway. Deep neural networks (DNNs) have facilitated major leaps in performance on tasks from many domains. They have been applied to image- and video-based problems with great success on many tasks including image classification [10], object detection [11], semantic segmentation [12], human pose estimation [13], and face recognition [14]. Deep models have become increasingly available for public use via cloud computing platforms [15–18]. This allows non-expert end users to easily integrate various DNN-based models into their applications.

In comparison with other techniques, deep models are often difficult to explain. This is in part due to the size and complexity of deep models, which may consist of billions of parameters fit using thousands of GPU hours using millions of training samples and stochastic techniques. Still, many researchers have proposed methods for peering into these black boxes. For example, it is now commonplace for researchers to offer ablation studies when attempting to justify a new architecture or training method, testing for performance changes between incremental modifications (e.g. [19–21]). Others offer qualitative techniques such as activation visualization (GradCAM [22]) or nonlinear projection of outputs to fewer dimensions which can be plotted (t-SNE [23], UMAP [24]). In the interest of quantitative comparisons beyond performance, some choose to measure the similarity between learned representations. These measures can be abstract, but generally are presented alongside some insights they support, such as: CNNs tend to learn features "bottom-up" [2], and early layers tend to be similar across tasks [1]. These representational similarity techniques are most similar to those studied here. The work presented in this thesis finds the last-layer representations of various face recognition models trained for the same task are highly similar, adhering to a shared structure despite their differences in architecture or training dataset. Subsequently, a method for analyzing this structure is studied as a means for better understanding face recognition model bias.

For clarity, representations, features, embeddings, and activations are considered equivalent terms here, referring to the outputs of hidden units (or layers) of deep neural networks. These outputs may be fed to another bank of hidden units, or treated as the final output of the model and used for a given task. Over the course of model training, a representation is learned which distinguishes some inputs from others in a way which is useful for the modeling task, as measured by an objective function. In the models studied here, this objective involves converting the representation of the model into a prediction, and comparing that prediction against ground truth labels for training or evaluation. As model designs, training methods, and training datasets differ, the hidden representations learned by models may also differ. Efforts to understand how representations vary across different models not only study which models are objectively better, as commonly measured by task performance, but also shed light on the structure of information extracted by neural networks. In Section 2.1, notable such efforts to measure, analyze, and utilize representational similarity are covered in detail.

This thesis covers my efforts to add to the growing body of research aimed at understanding deep models by comparing their learned representations. Where other works may focus on how representations change between different neural network layers, different steps in training, or between models trained for different tasks, the work here is concerned with the last-layer representations of models of different architecture, trained for the same task. By architecture, I mean the configuration of learned nonlinear functions which extract information from the input which is useful to the modeling task. These architectures vary wildly, from the elegant residual skip-connections of ResNet [25] to the systematically-refined "cell structures" discovered using neural architecture search [26]. Some architectures will perform better than others, require fewer training steps, or consume less computing power, as they are each trained from different initial parameters, many learning within a different parameter space. Despite their differences, however, many architectures are compared and found to extract very similar information in the early efforts covered here.

Specifically, a linear mapping was fit such that the last-layer activations of one CNN could be swapped with the other with little change in overall image classification performance. In the early stages, the last-layer activations of two deep convolutional neural networks (CNNs) trained for ImageNet [27] image classification, Inception-v4 [28] and ResNet-152 [25], were found to have a linear relationship with one another [29]. This finding surprised me and also my advisors, since each model is taking a unique path through a different parameter space, yet arriving on a very similar target. Since linear mappings are lossless and structure-preserving (when full-rank), the results of this first experiment suggest that differences in CNN architecture have little impact on the content of the final representation learned. Encouraged by this finding, this experiment was expanded to include all 90 unique pairings of 10 ImageNet CNNs. Now covering a much broader array of architectures, the same linear-interchangeability phenomenon was observed. All models studied up to this point were trained for ImageNet image classification, chosen for its relative popularity in the deep neural modeling community. However, a common architectural component associated with all such models (the final linear classification layer) drew the significance of the results of those experiments into question. This work was originally presented in my Master's thesis [30], summarized in Chapter 3.

To avoid the pitfalls of those previous efforts, the new contributions presented in this thesis concern models trained for face recognition. Face recognition is of interest for a few key reasons. First, evaluation of face recognition systems is typically carried out with faces unseen during training, i.e. open-set evaluation. In contrast, image classification systems are typically evaluated using images of the same classes used during training. In image classification, convolutional layer outputs are converted from abstract embeddings to class-wise confidence scores via linear classifier, and this shared class-wise score space is what drew the previous results into question. Though face models may be trained using a linear classification layer, they are subsequently evaluated on new faces, meaning that classification layer is no longer relevant. Instead, direct comparisons of those pre-classification representations are emphasized. Second, the face recognition community offers many datasets, especially evaluation datasets with established protocols and metadata. A detailed

description of recent face recognition modeling techniques and evaluation protocols is covered in Section 3.2. Finally, face recognition has the potential to create massive social impacts, especially with regards to fairness. Indeed, many have found demographic bias in face recognition. In a recent massive comparison of commercial face recognition systems, dark-skinned women's faces were consistently recognized at lower rates than others' faces [31]. Many have analysed demographic bias in biometrics, covered in greater detail in Section 2.2. In short, demographic bias in face recognition is commonly observed, and is concerning for its potential to reinforce institutional bias and inequality among demographic groups, as discussed in [32].

In Chapter 4, the representations produced by face recognition models of different architecture and training protocol are compared, revealing a fundamental similarity between CNNs trained on the same task, in line with the results of previous experiments using ImageNet models. These models may vary by architecture, loss function, or training dataset, yet produce highly similar face embeddings. This fundamental similarity can be thought of as a canonical embedding space, a compelling and consequential finding which may have implications for other CNN models and modeling tasks. One implication with regards to security is described, which has already been applied to extract the identity of individuals thought to be obscured with embeddings in a federally-funded research program (DARPA AIDA).

Evidence of a shared embedding structure for models trained on the same task invites investigation into the nature of this structure. Each model's instance of this shared, task-driven structure may have many interesting properties, dependent upon model designs. In other words, for the task of separating faces of different people, though the bulk of the learned representation may be shared between distinct models, fine yet meaningful deviations may be driven by model designs. Since face recognition datasets often provide face attributes alongside identification labels, a unique bias analysis opportunity is presented. Learned representations may be analyzed for the purposes of understanding demographic structure, with implications for face recognition bias.

In Chapter 5 a method for comparing subsets of embeddings as linear subspaces, or points on a Grassmann manifold, is studied. Prior methods for measuring face recognition bias compare

embeddings using pairwise distances in the same or similar way as the model is evaluated [33–35]. These methods express meaningful biases in the form of lower recognition rates and different distance distributions for certain people groups. What is hidden by aggregates of distances, though, is how bias is or is not also structural in nature. As evidence is found that face representations conform to a shared structure, comparisons between face representations which are sensitive to structure are encouraged. Using geodesic distance between points on a Grassmann, face demographics are represented as groups, and compared as groups. Comparing information about faces in this way complements the trends of previous experiments, further solidifying the idea that face models converge to a common representation.

Though the model designer has great incentive to design better models, and a social duty to reduce bias, the interactions between model designs and learned biases can be complex. In one study, deep neural networks were found to suppress difficult, yet predictive information in favor of easy, less predictive information (i.e. shortcuts) [8]. Along similar lines, 3 times more female faces than male were necessary to prevent gender bias [36]. From another perspective, successful bias-mitigation efforts involve the obfuscation of demographic information [37–39]. So, some find that different demographics require different amounts of information, while others find success by removing demographics altogether.

As the relationships between our semantic understanding of faces and the incentives of a deep neural network are sure to be complex, this research is intended to offer a new perspective on face embedding structure, with a focus on demographic bias. This is surely an enormous task, but these initial results are encouraging. Face structure can be broken down, and this structure is consistent across models, in line with the results of previous efforts.

Chapter 2

Related Work

Many have compared models and their learned representations for the purposes of understanding or improving them. These are covered in Section 2.1. As mentioned prior, many seek to estimate, understand, and mitigate bias in face recognition models. These are covered in Section 2.2.

2.1 Representational Similarity

One straightforward method to compare representations is by measuring their correlation after some sort of alignment. Many works have taken this approach, varying the method of calculating correlation or alignment. Often included alongside these proposed metrics are effects observed when measuring representational similarity between different models, at different stages of training, or using different datasets. While many provide justification for their technique, even including examples and comparing other methods, there is no clear consensus on what qualities a representational similarity metric *should* have. This issue is remedied in many works by applying the similarity measure to replicate or reveal a representational phenomenon, providing grounding to an otherwise potentially speculative technique.

In this section, the activation values produced by the neurons of a neural network are considered to be their representation, as opposed to their weights. Also again, representation, feature, and activation all refer to the same thing: the output of some layer in a deep neural network after feeding it an input, in contrast to the output of the final layer, or the weights themselves. Though the new work in this thesis only studies the last-layer representations, many works covered here study other layers as well.

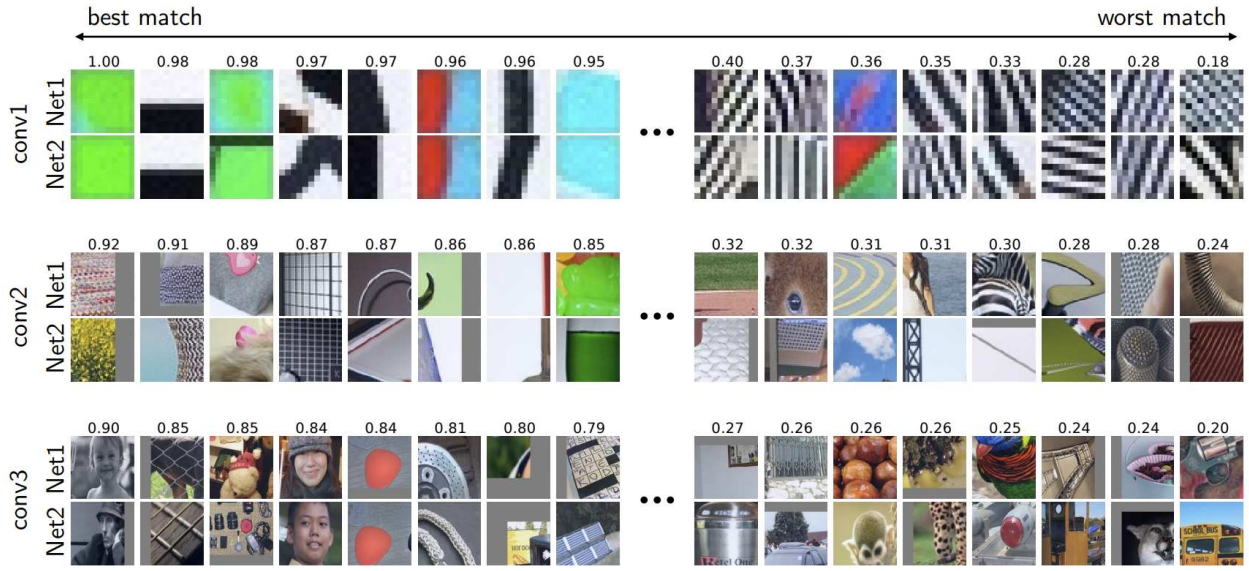


Figure 2.1: The eight best and worst features chosen by bipartite matching in [5], as measured by correlation. Taken from [5].

2.1.1 Correlation-Based Metrics

What Should Similarity Be Invariant To?

Kornblith *et al.*2019

Natural Alignment

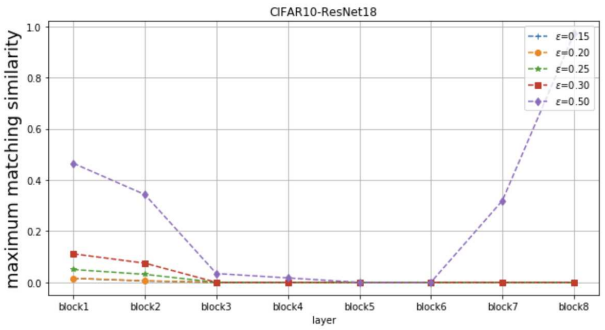
In 2015, Li *et al.*looked for axis/neuron alignment between convolutional neural networks (CNNs) trained on the ImageNet[1] [40] by pairing neurons with maximal correlation [5]. They train a variant of AlexNet [40] on ImageNet [27] using 4 different random initializations. Then for each network, activations generated from the ImageNet validation set are extracted from all layers, and correlations are computed between all pairs of neurons from the same layer of different nets. Correlation is computed by taking the inner product of mean-centered, standardized activations. Notably, when comparing activations produced by convolutional layers, they choose a point in each convolutional filter (channel) at random. Correlation scores are then used to compute a

[1]For reference, ILSVRC2012, commonly referred to as ImageNet, is a large-scale image classification dataset containing 1.3 million images of various sizes across 1000 class labels.

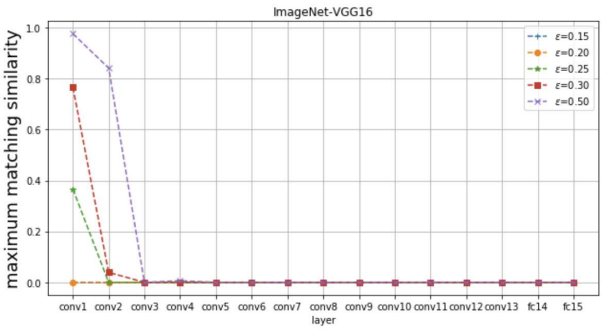
bipartite *match* or *semi-match* between neurons from the same layer of different networks. In other words, they find one-to-one correspondences for each neuron in network A, layer i by finding the neuron in network B, layer i which has the highest correlation score, optionally reusing neurons in network B (a *semi-match*). They report the mean inter-model correlation at each layer after performing these alignments, and offer examples which activate matched filters such as in Figure 2.1. Additionally, the authors use a *mutual information* technique to score pairs of neurons, reporting observations are "largely similar to that of using correlation similarity." Later, they find many-to-many correspondences by spectral and hierarchical clustering techniques, providing examples of filters which are clustered.

This work is one of the first efforts to measure representational similarity between deep CNNs, providing evidence that "some features are learned repeatedly in multiple networks, but other rare features are not always learned." However, while they report correlation values and provide qualitatively-similar examples, it is not immediately clear how similar one should expect representations to be.

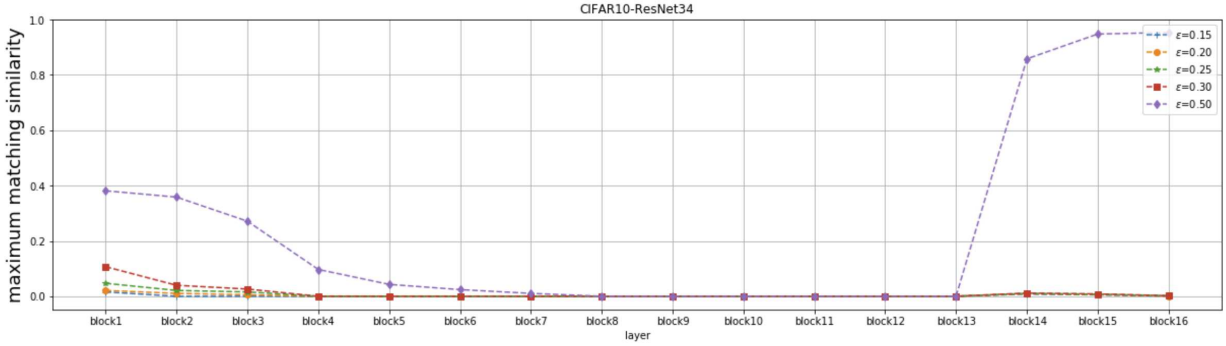
In 2018, Wang *et al.*built upon this neuron alignment effort by searching for minimal subsets of neurons between networks which represent each other, in a similar effort to the previous work by Li *et al.* [5, 6]. In contrast to the previous work, however, they measure similarity by finding subsets of neurons $\hat{A} \in \mathcal{A}$ and $\hat{B} \in \mathcal{B}$ such that every neuron $\hat{A}$ can be expressed by a linear combination of neurons $\hat{B}$ within some tolerance ϵ. As in [5], they represent neurons as the activation values produced by each layer of a deep convolutional neural network. Their work provides comprehensive theory on the properties of various kinds of matching subsets of neurons, and multiple algorithms for finding them. Similar to Li *et al.* [5], they perform comparisons between nets of identical architecture and training dataset but different random initialization. For convolutional layers, they "randomly sample d from $h \times w \times d$ outputs to form an activation vector for several times [sic], and average the maximal matching similarity." A closer inspection of their published code reveals that these "neurons" used for similarity measure are randomly sub-selected from all inputs and all spatial positions, using the same randomly chosen input examples and positions for



(a) CIFAR10-ResNet18



(b) ImageNet-VGG16



(c) CIFAR10-ResNet34

Figure 2.2: Maximal matching similarities of different architectures trained on different datasets at various matching tolerance ϵ, taken from [6].

each channel. This random subsampling method is very similar (even equivalent, given more detail) to the method described by Li *et al.* [5]. Using variants of VGG [41] and ResNet [42][2] trained on either CIFAR-10 [43][3] or ImageNet [27], they report similarity between the same layer of different networks as the proportion of neurons belonging to a matching subset out of all neurons in both layers. In summary, they report near-zero similarity for all layer-wise comparisons of the networks they study, except for layers near the input or output compared with a high tolerance. For these results, please refer to Figure 2.2. In other words, this work found that the proportion of neurons in a given layer of one network which can be expressed as a linear combination of neurons in the same layer of a different network is typically very small.

At face value, this work presents a compelling case for incompatibility between intermediate layers of deep image classification CNNs. However, as will be discussed in the next chapter, multiple other works produce contradictory results–that a linear transformation is adequate to transfer performance between models which differ not only in random initialization, but in architecture or training dataset as well [7, 29, 44]. While Wang *et al.*provide rigorous definition of their method, it is not immediately clear what the significance of their linear combinations of neurons is. As before, there is no clear baseline for comparing similarity measures aside from the upper and lower bounds of total linear dependence or independence. More importantly, both Wang *et al.*and Li *et al.*make some key assumptions which significantly narrow the scope of representational similarity measured.

First, for each filter in a layer, both works randomly sub-sample over the spatially-sensitive activations of all input samples. By comparing activations at spatially-sensitive points, these methods are built on the implicit assumption that similarities in activations be spatially aligned. This assumption is reasonable when representations come from the same model architecture, layer depth, and input images. However, a small change in image resolution, filter size, or filter stride will

[2]ResNet is a popular CNN architecture for its use of "shortcut connections" also known as residual or skip connections. Simply put, these explicitly pass an unchanged representation from early layers to later layers, and are now present in one way or another in nearly every subsequent CNN architecture.

[3]For reference, CIFAR is a common image classification benchmark dataset, made of 60k 32x32 color images split into 10 classes.

likely destroy such alignment. Moreover, certain areas of images may contain more relevant features for classification than others, further biasing this method which weighs all random positions and samples equally. Other representational similarity works such as SVCCA (discussed in the following section) note the same, that such spatially-sensitive features are only valid when they come from the same layer of the same architecture.

Second, both of these works focus specifically on the activations of neurons, and methods for matching similar neurons to others (or collections thereof). Prior investigative works have found such "natural bases" defined by individual neurons are semantically indistinguishable from random bases in activation space [45]. What's more, for the sake of interpretability some have worked to remove this effect, producing CNNs which semantically separate visual information by neuron or filter, so-called "disentangled representations" [46]. By anchoring their metrics in pairings or subsets which operate on subspaces defined by neurons rather than on relationships between entire activation spaces, both methods rely upon the assumption that independent features naturally align with neurons–i.e. with weight vectors. Without a compelling argument or clear evidence for representational alignment along neurons, this assumption likely produces an arbitrarily narrow and biased measure of similarity.

In the following subsection, techniques for aligning representations along arbitrary correlated bases are explored, including methods for comparing the activations of different convolutional layers or different images. While it is still unclear what these metrics *should* measure, there is a sense in which the following results contradict those covered so far, since they tend to indicate more and more frequent levels of similarity.

SVD Alignment

In all works discussed in this subsection, samples of representations from either source, X and Y, are centered.

In 2017, Raghu *et al.*adapt canonical correlation analysis (CCA) to measuring similarity between the representations of deep neural networks, naming their technique singular vector canonical correlation analysis (SVCCA) [2]. For reference, CCA consists of finding linear transforma-

tions W_X, W_Y such that the correlation between two sets of activations X and Y are maximized, after which correlation between each component in X and Y is reported. Aptly named, SVCCA first uses singular value decomposition (SVD) to reduce the dimensionality of both matrices before applying CCA to measure correlation.

For reference, SVD decomposes a real (or complex) matrix (e.g., of activations) X into the product of three matrices $X = U\Sigma V$. For a decomposition of a matrix $X \in \mathbb{R}^{m \times n}$, the matrices U and V are both orthonormal transformations, and the matrix $\Sigma \in \mathbb{R}^{m \times n}$ is a diagonal matrix with values corresponding to the $p = min\{m, n\}$ singular values of X. For $k < p$, a reduced k-dimension matrix X' can be calculated as $X' = U\Sigma' V$, where Σ' is produced by multiplying $p - k$ values along the diagonal of Σ by zero. Since the values of Σ are typically sorted (i.e. by increasing variance) the latter $p - k$ values of Σ are typically chosen for multiplication so that the $p - k$ dimensions of X with the least variance are those removed, and the k dimensions of maximum variance are retained. Raghu *et al.*demonstrate that these dimensions of low variance have little impact on performance and contain primarily noise [2].

To validate the use of SVD for dimension reduction, they reduce the dimension of features from the penultimate layer of a 7-layer CNN trained on CIFAR-10 from 512 to 25, reporting minor performance penalty. As a sanity check, if they instead either choose k random neurons or k neurons of maximum activation as means of dimension reduction, far more dimensions are required to meet the same performance as dimension reduction by SVD. By first reducing the dimension of data using SVD before CCA, the authors argue that greater similarity distinctions can be expressed than with CCA alone. They provide a supporting example of this in which correlated, low variance features would produce high correlation with CCA alone, but are instead first removed by SVD so that calculations of similarity be restricted to dimensions of high variance. Such behavior may be desirable when seeking to determine whether two representations simply contain correlated information, or contain correlated *high variance* information. Admittedly this situation seems contrived, and the authors are relatively unclear on the benefits of combining SVD

and CCA beyond statements such as, "Both SVD and CCA have important properties for analysing network representations and SVCCA consequently benefits greatly from being a two-step method."

To compute similarity scores for two representations X and Y (e.g. the outputs of hidden layers from two different networks), the authors first apply SVD to find subspaces $X' \in X$ and $Y' \in Y$ which each express 99% of variance of the original representations. Then, they use CCA to compute linear transformations U and V such that the correlation of resulting subspaces $\tilde{X} = UX'$ and $\tilde{Y} = VY'$ is maximized. The output of CCA is a series of correlation-maximizing orthogonal bases U for X and V for Y, and the correlations between both transformed representations $\boldsymbol{\rho}$. The *SVCCA similarity* is computed as the mean of the first p correlations [4].

As mentioned in the prior subsection, they describe inherent incompatibilities introduced by comparing the activations of convolutional layers when there are differences in model architecture, layer depth, or input image. To deal with different layers or inputs, they propose to compare convolutional layer activations by treating each position of each filter as a separate neuron, searching for linear combinations of these which are high variance and correlated. This approach is certainly more generalizable than relying on spatially-aligned features, but produces computationally large spaces to decompose, align, and correlate. In contrast, other methods use global average pooling, averaging across all spatial locations within each filter. To reduce the size of computation without losing precision, the authors make use of a discrete fourier transform (DFT), for which they provide ample theoretical support in context of translation-invariant CNN features.

The authors apply this method to multiple models in various ways, revealing interesting trends. When measuring SVCCA similarity across time-steps during training of a basic 5-layer CNN and a 30-layer ResNet trained on CIFAR-10, they observe that early layers converge more quickly than later layers for both networks. This data is provided in Figure 2.3. They apply this finding by "freeze training" both CNNs, in which layers are progressively frozen from the bottom up during

[4]It should be noted that SVCCA projects X and Y into fewer dimensions *before* CCA is computed, not to be confused with the standard use of SVD in the calculation of orthogonal bases of X and Y. Inspecting the code published alongside [2], SVD is indeed computed on both sets of activations before CCA and within CCA, in the computation of *SVCCA similarity*.

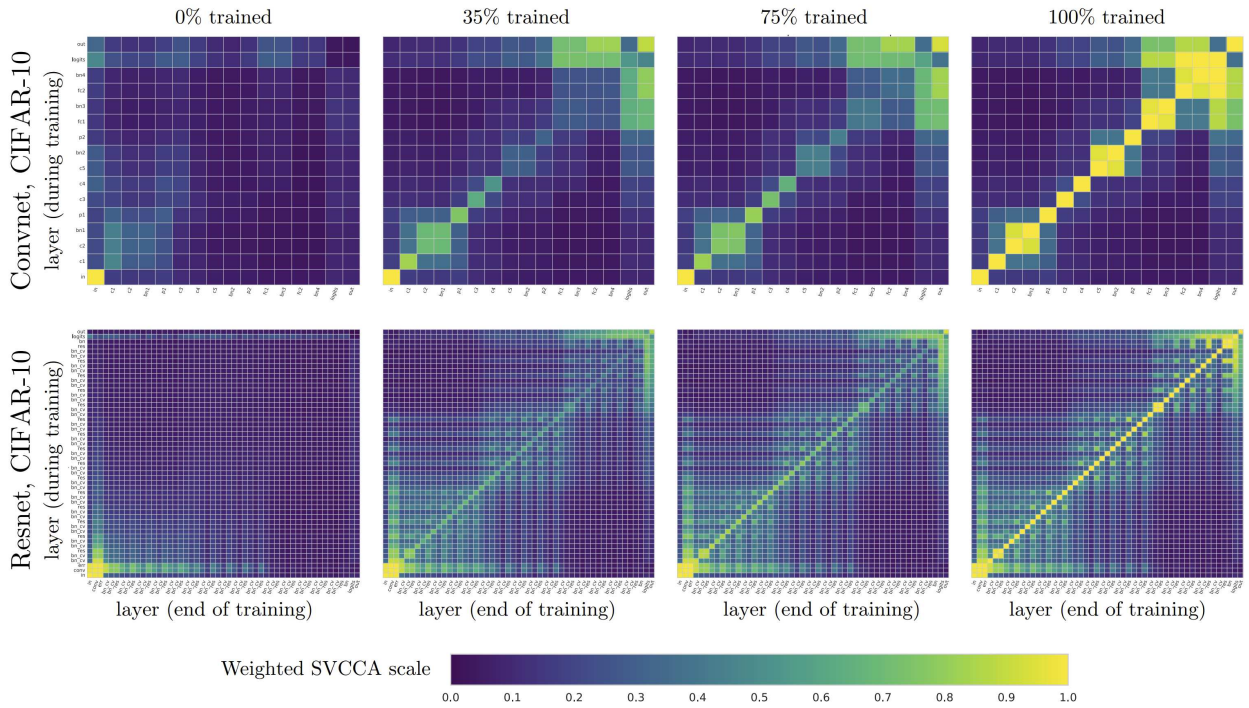


Figure 2.3: SVCCA similarity matrices of trained and untrained layers of ConvNet (a plain 5-layer CNN, first row) and ResNet (second row), measured at different training steps (column-wise). Taken from [2].

training, producing modest improvements in test performance and training speed. Next, they use SVCCA similarity to compare intermediate layer activations with neurons corresponding to a given class logit (a single neuron of the final softmax output layer), demonstrating that SVCCA similarity is expressive of semantic differences between classes. Finally, they also compare between representations from different architectures, revealing generally low similarity except in early layers. This last result is somewhat at odds with later representational similarity efforts ([1, 3]), which find consistent similarity across intermediate representations of different architectures trained on CIFAR-10.

Later, Morcos *et al.*present a refinement of this technique named projection weighted canonical correlation analysis (PWCCA) [3]. By viewing the sorted singular vectors (orthogonal bases) produced by CCA over the course of training, they found that the order of smaller singular vectors varies considerably, increasing variability correlation scores that they hypothesize can be attributed to noise. To address this, they replace the mean correlation presented by [2] with a weighted mean which they call *projection weighting*. These weights, $\tilde{\alpha}_i$ (one per orthogonal basis produced by

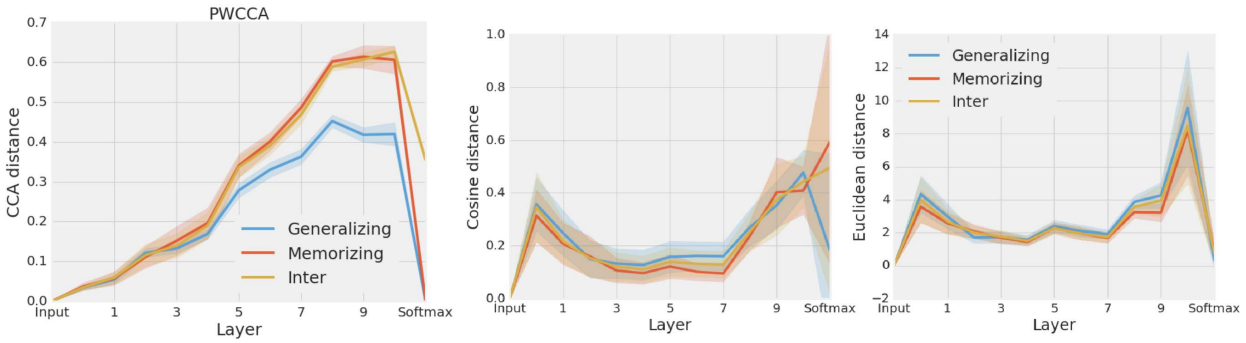


Figure 2.4: Various distance measurements (y-axis labels) between representations produced by models trained on true labels (Generalizing), a fixed random permutation (Memorizing), or between either (Inter). Taken from [3].

CCA) are computed as the sum of absolute covariance between neuron activations $x_j \in X$ and each CCA bases $u_i \in U$, or

$$\tilde{\alpha}_i = \sum_j |\langle h_i, z_j \rangle|.$$

Weights are normalized such that $\Sigma_i \alpha_i = 1$, allowing (asymmetric) *PWCCA distance* to be computed as

$$d(X, Y) = 1 - \sum_i \alpha_i \rho_i.$$

They apply this similarity measure to uncover and replicate multiple phenomena on 11-layer CNNs trained on CIFAR-10. By training a model using a fixed random permutation of labels (as in [47]), they reveal that such "memorizing" networks learn representations which are less similar than "generalizing" networks trained using true labels. In addition, they observe that this effect is not expressed using Euclidean and cosine distance as similarity scores. These measures can be viewed in Figure 2.4. Next, they compare the similarity of models with progressively wider convolutional layers (number of convolutional filters). They found wider networks tend to learn more similar representations, also finding tight correlation between similarity and test accuracy, even when similarity is computed on the training set. Later, they train 200 networks with identical architecture from different random initializations and with different learning rates, and measure the similarity between all pairs of models (using activations produced by the 8th layer of each model). For these 200 initializations and learning rates, they find 5 distinct similarity clusters. When mea-

suring the cumulative robustness of each model to filter removal/ablation, similar clusters emerge. Finally, they use a recurrent neural network (RNN, in contrast to a *feedforward* network such as a CNN) trained on language modelling tasks. For reference, an RNN is fit using sequential data such as a time series or text corpus, modelling data seen previously using a hidden state. They find that PWCCA reveals RNN layers are learned "bottom-up," as was revealed for CNNs by SVCCA [2], though SVCCA is not expressive of this effect for RNNs. When comparing RNN hidden representations at different steps in the sequence of data, however, they find that PWCCA is no more expressive of changes in hidden representation than cosine or Euclidean distance.

PWCCA provides a substantial improvement over previous representational similarity efforts in many regards. Instead of comparing convolutional layer activations in a spatially-sensitive manner which is subject to problems of alignment, resolution, and computational size, they average over all spatial locations for each filter. As stated previously, this method is commonly used on the final convolutional layer before classification, known as global average pooling. Further, Morcos *et al.*provide other similarity measures for comparison including the measure which inspired them (SVCCA [2]), and naive Euclidean or cosine distance. In addition, they also include the standard deviation for their measures when appropriate. By including experimental controls, baselines, and uncertainty measurements, this effort stands out as a more justified improvement in representational similarity methods.

In the most recent correlation-based representational similarity work covered here, Kornblith *et al.*begin with an analysis of the necessary properties of a representational similarity index, followed by a comprehensive overview of previous methods in the representational similarity literature and covered here prior [1]. Through the lens of transformation invariance, they categorize each method and describe the strengths, weaknesses, and applications of each category. In particular, they provide motivation for a similarity metric which is not invariant to invertible linear transformation. This property is differentiated from CCA and linear regression techniques (such as my own, covered in the following section). Using a simple proof they demonstrate the necessity for a greater number of samples than sample dimensions for any such invariant similarity measures, which is

problematic when comparing very large convolutional layers trained on smaller datasets. Additionally, they cite multiple works suggesting that neural network training is not invariant to invertible linear transformation of either inputs or activations, arguing that neural networks consistently learn representations with similar principal components and feature scales, and that arbitrary linear transformations may obscure or even invert this information. Beyond invertible linear transformation invariance, they argue that a similarity matrix should be invariant to orthogonal transformation since "training with fixed orthogonal transformations of activations yields the same distribution of training trajectories as untransformed activations." They also decide to include invariance to isotropic (uniform) scaling, which is not explicitly argued for, but intuitively allows for capturing similarity between features which have the same shape but different size. Importantly, they note that if they were to include invariance anisotropic (non-uniform) scaling with invariance to orthogonal transformation, this would provide invariance to invertible linear transformation.

In a departure from previous efforts, they choose to first compute the similarity between centered samples X and *itself* as XX^T, and then compare this inter-example similarity structure with the structure produced by centered samples Y as YY^T. These self-similarity matrices are also known as representational similarity matrices (RSM) and commonly used in neuroscience literature. Conveniently, the inner product between these two inter-example similarity structures after flattening them both is equal to the sum of squared inner products between pairs of examples across representations, or

$$\langle \mathrm{vec}(XX^T), \mathrm{vec}(YY^T) \rangle = ||X^T Y||_F^2$$

when the linear kernel is used (i.e. the dot product). To maintain invariance to isotropic scaling, they normalize this score as

$$\frac{||X^T Y||_F^2}{||X^T X||_F^2 ||Y^T Y||_F^2}$$

which they note has been discovered thrice prior [48–50]. In the most recent rediscovery, this method is extended to any positive semi-definite kernel and called centered kernel alignment

(CKA). In subsequent analyses they include CKA similarity measures using both a linear kernel and a radial basis function (RBF) kernel.

Besides discussing motivating variances and invariances for their representational similarity metric, the authors also define the differences between CKA and other metrics in the field. Indeed, each of the metrics covered prior are included in this section of [1]. In particular, they note that SVCCA is invariant to invertible linear transformation so long as the same subspaces are found with SVD, and that PWCCA similarity is very similar to a summary statistic used for linear regression. They go on to state that linear CKA resembles CCA weighted by the amount of variance explained by the singular vectors of X and Y (i.e. singular values–see equations 13 and 14 of [1]).

To ground their reasoning and chosen metric, they provide a representational similarity sanity test. This test involves comparing CKA, CCA, SVCCA, PWCCA, and linear regression metrics on their ability to correctly identify representations produced by the same layer of identical architecture CNNs trained on the same data with different random initializations. For exact formulations of each summary metric, see Table 1 of [1]. CKA using either a linear or RBF kernel outperforms all others by a wide margin on this test. Their illustration of the layer-wise similarity structure between models trained from different random intialization as revealed by different metrics is provided in Figure 2.5.

The remainder of the paper consists of experiments revealing or replicating representational similarity phenomena using linear CKA. For all of their measurements, they compare all combinations of layers either within or across models, producing a 2D heat map of similarity. While it is not explicitly stated in [1], attempts to replicate their results make use of global average pooling to compare the outputs of convolutional layers.

In their first experiment, they compare deep CNNs trained on CIFAR-10 with varying depth (number of convolutional layers). They find that increasing depth up to a point increases performance, after which depth is "pathological" as performance is degraded and groups of consecutive layers learn highly similar representations. The same effect is not observed when testing ResNets ([42]). Next, they use CKA to identify layers of corresponding depth across different architec-

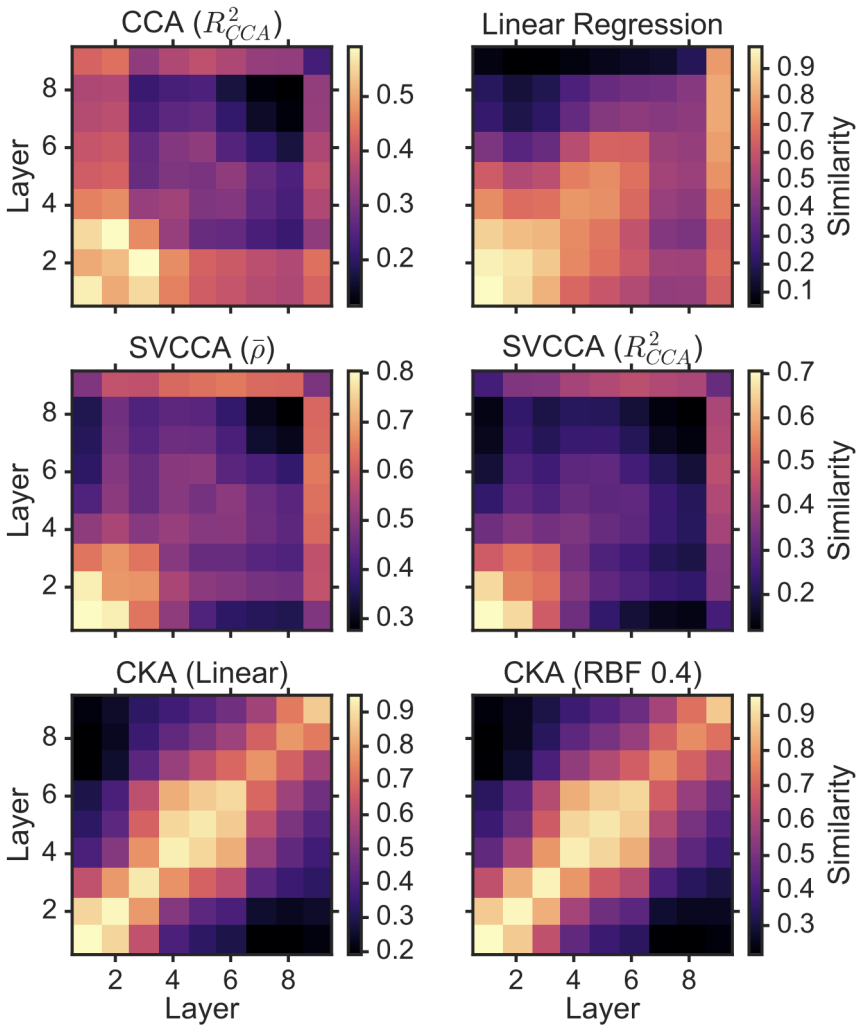


Figure 2.5: Comparison of various similarity metrics. Each grid corresponds to the similarity between layers of identical CNNs trained on the same data from different random initializations. Taken from [1].

tures, finding linear CKA is capable of comparing representations across architectures, while other similarity metrics fail to reveal cross-architecture correspondence. When measuring similarity between models of increasing width, they find layers "become more similar to each other and to wide networks as width increases." When comparing models trained on CIFAR-10 against those trained on CIFAR-100, they find models trained on different datasets learn similar representations especially in early layers. Finally, they compare two identical CNNs trained from different random initializations by measuring the similarity of each RSM XX^T and YY^T with each eigenvector of XX^T, u_X^i. This leads them to conclude that the shared subspace of each RSM is spanned primarily by the largest eigenvectors.

In comparison to other methods, the work by Kornblith *et al.*provides far more motivation, both within context of related efforts and without. Just as CCA-based measures have revealed more detail than neuron alignment measures, CKA produces far finer expressions of similarity than many prior works. By (re)defining the properties a deep neural network similarity measure *ought* to have, and providing both mathematical definitions of and empirical evidence for the shortcomings of other measures, this work serves as a model for future efforts in the field.

Despite both the ample and growing empirical and theoretical support of these methods, however, they are so far limited to analysis of representations. In the following section, methods which measure similarity by the performance of CNNs on transformed or interchanged representations are covered. By shifting the focus of representational similarity from relative measures of correlation to concrete measures of performance, the following works provide further, grounded insight of the properties of deep neural networks.

2.1.2 Performance-Based Metrics

> What does it mean by two representations being different?
>
> Feng *et al.*2020

Instead of measuring the similarity of different representations directly, some reveal insight by swapping or otherwise altering the representations themselves and measuring the impact on model performance. Relying on the performance of the model provides an implicit baseline to compare against, in contrast to the relative differences revealed by correlation-based methods. In a broader sense, measuring similarity by changes in performance also requires fewer assumptions to be made about the meaning of representational similarity. While it may be reasonable to assume that features which are uncorrelated are distinct, it is still an assumption that must be tested (e.g. by a "sanity check" as in [1, 4]). On the other hand, previous works have shown that some similarity effects may be hidden by performance metrics alone [2]. Regardless, as long as there are still new characteristics of neural networks to be uncovered, there will probably be many ways to do so.

What follows is a broader set of experiments attempting to characterize and understand the relationships between models by relying on performance. In an ideal world, both correlation and performance would be provided as in many of the works covered prior. By focusing on the source of representations over the comparison, however, these works provide ample insight into the nature of deep neural networks.

Franken-CNNs

The work by Lenc and Vedaldi combines many efforts to characterize and understand the invariances and covariances of deep CNNs trained for image classification [7]. They study 3 deep CNNs (an AlexNet [40] variant, VGG [41], and ResNet [42]) trained on ILSVRC2012 for image classification. Their analysis is divided into two broad efforts.

By rotating, flipping, or scaling images fed to these CNNs, the first effort reveals how such transformations can be either inverted in later layers (revealing equivariance), or have little to no

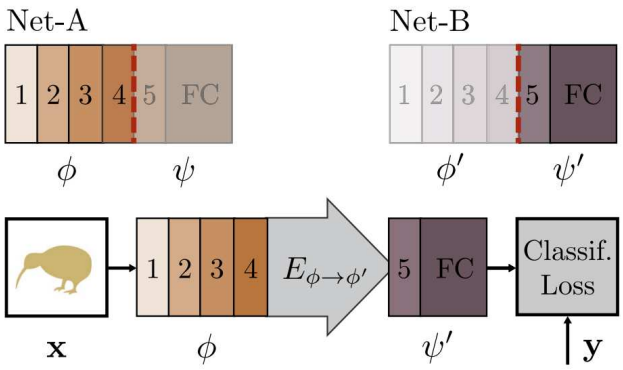


Figure 2.6: Method for creating "Franken-CNNs" for studying the coverage or equivalence of one network's representation with another's, by use of a stitching layer or mapping. Taken from [7].

effect on representations (invariance). For images which have been scaled or horizontally flipped, attempting to learn equivariant mappings of intermediate CNN representations does not improve performance over leaving features unchanged, which they attribute to the implicit existence of such transformations in natural images. They continue, "For vertical flips and rotations, however, the learned equivariant mapping substantially reduce [sic] the error."

The second effort consists of an analysis of the coverage and equivalence of features from different CNNs[5]. By producing "Franken-CNNs" made of the first m layers of CNN A and the last n layers of CNN B stitched together with an affine transformation, they study whether the features from one deep CNN are equivalent to another. An illustration of the method used to construct these "Franken-CNNs" is provided in Figure 2.6. In essence, if features from one CNN are compatible with another CNN after simple mapping (e.g. affine transformation), then the representations of each CNN must contain overlapping information. To fit these affine mappings, they feed training data to the "Franken-CNN" and backpropagate classification loss, freezing all weights except those in the affine mapping. They fit affine maps between all combinations of layers of two CNNs, and measure the performance of each "Franken-CNN." Similarity is measured as the performance penalty induced by mapping features between CNNs, relative to the performance of either CNN alone.

[5]The bulk of equivalence experiments and results are present only in the version of this article published in the *International Journal of Computer Vision* (i.e., please do not refer to preprints hosted by arXiv or Oxford University, nor the Open Access version hosted by the Computer Vision Foundation).

To map between convolutional layers of different spatial dimension, they use nearest-neighbor interpolation for down-sampling, and bilinear interpolation for up-sampling. First, they consider mappings between one CNN and itself, essentially either skipping or repeating layers using affine maps. They find that features mapped through repeated layers preserve more performance than skipped layers, which they summarize as "the deep layers of a neural network cover the earlier layer [sic], but not vice-versa." They note that there is a strong correlation between the differences in layer resolution and mapped feature performance. Next, they compare models which have the same architecture, but different dataset. They train AlexNets on ILSVRC2012, MIT Places [51], and a combination of both, before fitting affine maps between the features produced by each layer of each model and the same layer of another model. To be clear, features are mapped between the same layer of different models, rather than between all combinations of layers. They find accuracy is preserved when mapping between the first convolutional layers, and steadily decreases as mappings are fit to later layers. This experiment reveals a similar pattern as other representational similarity efforts–that early layers produce rather task-independent features, while later layers specialize to the task. Finally, they compare models trained on the same dataset (ILSVRC2012) but of different architecture. In this case, mappings are fit between all combinations of layers of AlexNet, VGG16, and ResNet50. For most mappings between layers of similar depth, they report modest performance penalty induced by mapping, suggesting a broad degree of coverage between CNNs of different architecture. In the case of VGG16 mapped to AlexNet, they find generally poor performance, and in the case of VGG16 mapped to ResNet50, all mappings produce <10% accuracy.

Focusing on their latter effort, Lenc and Vedaldi provide evidence for a fundamental overlap between the features extracted by different image classification CNNs. This work provides a foundation for my later efforts covered in the next chapter.

Franken-CNN works suggest that there is a fundamental way in which the features of CNNs trained for the same task tend to converge towards similar representations. One possible interpretation of this effect is that models trained on the same task tend to extract the same features–that the task determines which target features are most useful and perhaps suppresses those which are not

useful. Even still, further work is required to understand why and when representations overlap. For example, recall in the previous section that similarity measured by PWCCA and linear CKA is correlated with the performance achieved on the model's target task. This correlation is also observed in [7] and the next chapter, and may obscure other factors impacting similarity.

The following works focus on variations in modelling task to study the properties of deep CNNs, finding varied and interesting effects.

Task Swapping

The work by Feng *et al.*presents a representational similarity metric called transferred discrepancy (TD) [4]. This metric defines the difference between two models as the mean distance between their predictions on a new task. Predictions are obtained by first retraining each model's linear classifier for a new set of labels corresponding to the new task, Training a model on one task before replacing its linear classifier with one trained to predict labels for a new task is a common method within the field of transfer learning, called *fixed feature extraction*. Indeed, others have measured CNN feature transferability, finding that the performance on ILSVRC2012 is highly correlated to the transferred performance on a variety of tasks [52].

To elaborate, their features are always produced by the penultimate layer of a CNN, typically the spatially-pooled output of the final convolutional layer before being converted to classification predictions by a linear transformation. Then, new task predictions are produced by fitting new linear classifiers to both sets of features, with the objective of minimizing a loss function against a new label set (e.g multinomial logistic regression, i.e. softmax regression). By using a different label set and optionally different input images than those used to train either feature-producing CNN, this method allows one to measure the difference in transferability of each CNN's features to a new task.

The work by Feng *et al.*differs by presenting TD as a representational similarity metric, offering extensive theoretical analysis. In contrast to other efforts to characterize representations by their relative performance, such as those covered in this section prior [7, 44], this work focuses on the development of a representational similarity metric akin to SVCCA, PWCCA, and CKA [1–3].

Within this analysis, they show TD's invariance to orthogonal transformation and isotropic scaling, its behavior as the number of samples compared approaches infinity, and conditions under which TD is related to maximum match [6], CCA [2, 3], and linear CKA [1]. Notably, they state that a CCA goodness-of-fit measure (the mean of squared correlations, R^2_{CCA}) is equivalent to TD when averaged across "all linearly realizable tasks in the linear probing setting, which makes CCA downstream-task agnostic." While comparing representations by their performance on a new task is far from new, the detailed comparison with correlation-based measures provides new grounding for using performance to measure representational similarity.

They begin with a "sanity check," comparing TD, linear CKA, and CCA on their ability to express differences in features produced by models trained on different tasks. They train 32-layer ResNets [42] on CIFAR-10, CIFAR-5, CIFAR-2, and SVHN (**training tasks**), before evaluating each model on CIFAR-10 (**downstream task**). CIFAR-n is produced by grouping the labels of CIFAR-10 [43] into n categories, using the same, complete set of images. The Street View House Numbers dataset (SVHN) consists of images of house numbers taken from Google Street View images [53]. After training, they feed images from the training and test sets of CIFAR-10 to each, extracting penultimate layer activations from each model. Linear classifiers are fit using each model's features generated from CIFAR-10 *training* images and labels, after which TD is computed by comparing the predictions of each of those classifiers using CIFAR-10 *test* images and labels. Features generated from the CIFAR-10 test images are also compared directly using CCA (R^2_{CCA}) and linear CKA similarity scores. This procedure is repeated 10 times, and the average is reported. The results of this experiment show that models trained on variants of CIFAR are more similar to each other than to models trained on SVHN. Each metric is also expressive of relative differences in CIFAR variants, with greater similarity reported for smaller changes in n (i.e. features from models trained on CIFAR-2 are more similar to CIFAR-5 than CIFAR-10). The authors conclude that the results of this experiment show that each metric is expressive of task-driven similarity, though they note differences in the magnitudes of different scores. This and the following experiment's results are provided in Table 2.1.

Table 2.1: Comparisons of TD with CKA [1] and CCA [2, 3]. TD is evaluated on the downstream task. The top table refers to the "sanity check," and the bottom refers to the experiment on models of different initialization. Taken from [4].

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

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

In a similar experiment, they compare representations produced by models trained from different random initializations. Specifically, they train a pair of models on CIFAR-2, and another pair on CIFAR-5 (**training tasks**). TD is measured as before, using CIFAR-10, CIFAR-5, and CIFAR-2 as **downstream tasks**. This results in 6 TD scores (one for each pair and each downstream task). Since each CIFAR-n consists of only a change in labels, leaving images unchanged, they report only one CCA and CKA score per pair. This leads them to conclude that TD is more expressive than CCA and CKA, and "models trained with different initializations can capture different features." These results do show that TD is again expressive of the similarity of CIFAR variants. However, conclusions drawn from the methodological incompatibilities of TD with CKA and CCA are dubious at best. Further, since they only report one score for each model pair, it's not clear how random initialization has altered the features captured.

In their final set of experiments, they test the effect of various popular training techniques on TD. For each of these modelling strategies, they train models on CIFAR-5 and transfer to CIFAR-10. They find that random flips and crops of input images produces better TD scores, as well as large learning rates, learning rate decay, and adversarial training.

The work by Feng *et al.*serves as a useful analysis of representational similarity metrics, providing grounding for performance-based metrics. However, the results they present are very narrow in

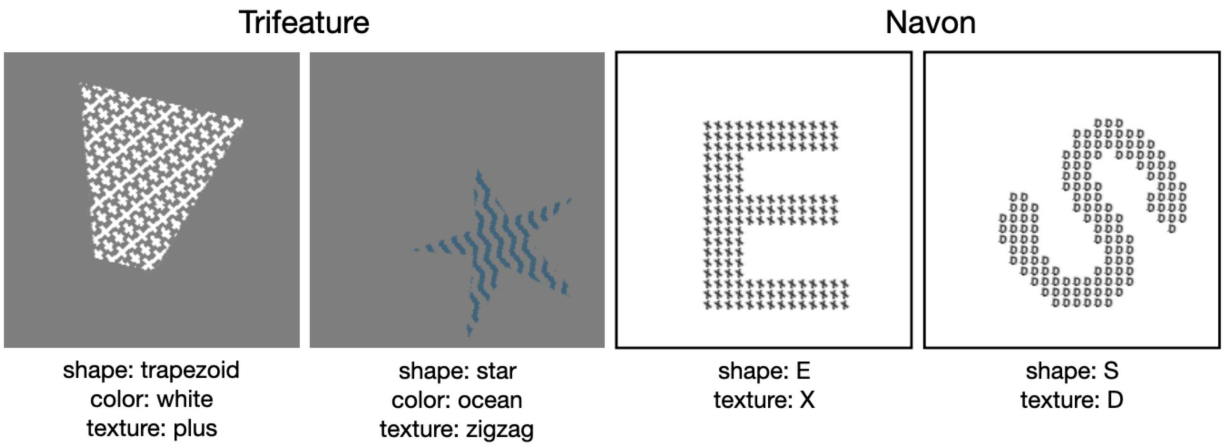


Figure 2.7: Examples from each dataset used in [8].

scope, even seeming contrived when used for comparison with other metrics. With more datasets and generally finer-grained results, a clearer picture could be produced of the differences between their methods and others.

In *What shapes feature representations?* Hermann and Lampinen use synthetic datasets to shed light on many learning effects of deep neural networks [8]. This work begins with a comprehensive review of related efforts, placing their work among many efforts to study the relationships between tasks used for deep learning, the effect of dataset statistics and task difficulty on deep neural networks, and the differences between trained and untrained model representations. Their work centers around two synthetic datasets, named Trifeature and Navon. By variably correlating attributes used to generate images, various tendencies of CNNs to learn to extract certain features over others are revealed.

In the Navon dataset (adapted from [54]), each image is generated and labeled using shape and texture attributes as seen in Figure 2.7. Images from the Trifeature dataset are similarly generated and labeled, with the addition of a color attribute. Datasets are generated using a single **target** attribute as the classification label, holding out subsets of non-label attributes for validation. For example, if a dataset were generated for texture classification, then the validation set may contain squares, triangles, and circles while the training set would not, so that texture classification remain generalized. Normalized images are then used to train an AlexNet [40] or ResNet50 [42]. Finally,

a linear classifier (or probe) is used to test the "decodability" of image attributes at many model layers. Decodability refers simply to the accuracy of such a linear probe in predicting image attributes from intermediate layer features. They provide a baseline measure of image attribute decodability using different target attributes, finding target attributes are enhanced while non-target attributes are suppressed compared to an untrained model. For example, they find that models trained to recover the type of shape in the input image tend to suppress the color and texture relative to an untrained model.

In their first experiment, they vary the degree to which one image attribute is correlated with another, and measure the decodability of each image attribute at various layers. When two attributes are perfectly correlated, they find color is preferred over shape, and shape is preferred over texture. What's more, even when both attributes are equally predictive such as this, non-preferred attributes are less decodable (suppressed) compared to untrained representations. When the correlation between attributes is reduced, the same effect is observed.

Next, they study multi-layer perceptrons (MLP) trained on non-visual binary tasks to study learning trends in modelling task difficulty. They feed 32-element binary vectors to 5-layer MLPs, where half of the vector is decodable by a single linear transformation, and the other half is produced by XOR which requires an MLP [55]. The model is trained to predict a label which is probabilistically determined by these two features, with varying predictivity of either. When the linear feature is not predictive, the model learns to extract the label from the non-linear XOR feature. However, the easier linear feature can suppress the harder non-linear feature, even when the harder feature is more predictive. This suggests a tendency for deep learning methods to trade off predictivity for ease of learning.

While they only use toy problems for their experiments, this paper is a model for the study of representational similarity. Superficially, some results may be intuitive. For example, the tendency for models to learn easy features even when harder (yet learnable) features are more predictive may be described by the presence of local optima during (non-convex) optimization. Still, some of these results suggest behavior that is much less counterintuitive, offering many exciting research

questions. Why do models suppress predictive information? If texture is preferred less than color and shape, what similar hierarchy of attributes exists for natural data? Even when possible explanations exist, works such as these offer many opportunities to further understand deep nets.

The next work by Gygli *et al.*builds upon efforts to analyze and apply representational similarity effects by training architectures which produce *compatible* representations [9]. Specifically, they seek a general method for the creation of a library of compatible network components which are specialized for different tasks. This motivation aligns with the findings of many previously revealed representational phenomena, namely that models broadly converge to similar solutions, revealing a degree of waste in the resources used to train and retrain models on existing tasks. The paper begins with an extremely comprehensive review of related work in the fields of representational similarity analysis, feature distillation, multi-task learning (MTL), continual learning (CL), unsupervised domain adaptation, and transfer learning.

In essence, they propose multiple strategies for building compatible components, before applying combinations of those strategies to train compatible network components. These components are then tested for compatibility by unsupervised domain adaptation, cross-architecture evaluation, and transfer learning. In this work, the two components they define are the feature extractor $f()$ containing all convolutional and pooling layers, and the target task head $h()$ containing the final classification layer. The three strategies they propose are compatibility through self-supervision (RP, for rotation prediction), through discriminating common classes (DCC), and through identical initial weights (IIW).

In the first, they add an auxiliary task which requires no external labels, commonly referred to as self-supervision. They choose rotation prediction, in which an auxiliary task head must predict the random rotation applied to an input image. DCC is achieved by computing the sum of losses on two tasks, relying upon common class labels across tasks. IIW involves simply starting the training process from the same initial weights instead of a new random initialization, relying upon identical architectures.

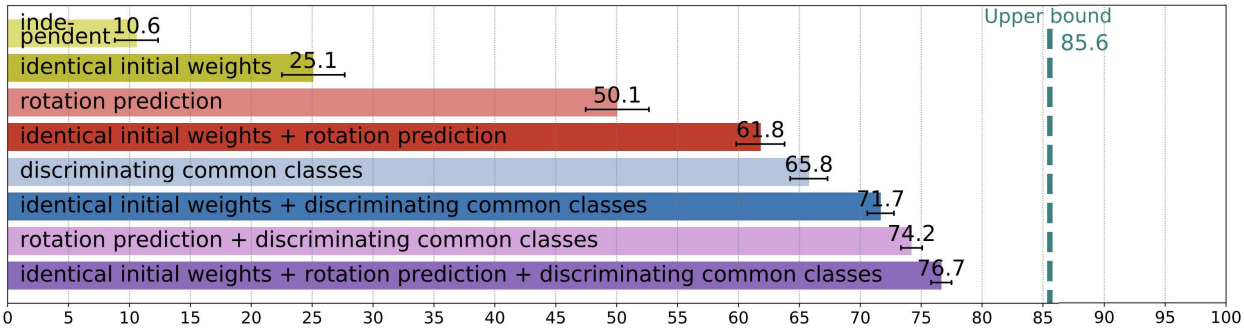


Figure 2.8: Recombination accuracy for each different compatibility strategy. Numbers refer to the mean accuracy over 10 runs, with standard deviations indicated by horizontal error bars. The upper bound is determined by training and evaluating a network without recombination. Taken from [9].

To test for compatibility, they train ResNet56 ([42]) CNNs with some combination of the 3 strategies on different datasets (CIFAR-10 and STL-10, a very similar image classification dataset with 9 out of 10 classes overlapping [43, 56]). After training compatibility is measured by swapping the task head for each model and measuring their performance on the test set of the dataset used to train the feature extractor. Combining all strategies produces the highest accuracy in this way, indicating that each strategy improves compatibility. The accuracy for each combination is depicted in Figure 2.8. Interestingly, they find that starting from models pre-trained to another task produces worse performance, suggesting that compatibility should be encouraged early in training. They also measure the correlation between features without optimal transformation, finding features are highly correlated and aligned even when compared to networks trained independently and compared *with* optimal transformation (as in CCA).

Next, they apply these techniques for the purposes of unsupervised domain adaptation, cross-architecture compatibility, and more efficient transfer learning. By training a model on CIFAR-10 and including an auxiliary RP task head, then training the model to perform rotation prediction on a new dataset without updating the RP task head, they produce a model which is compatible with both the original dataset and the new dataset. Using this method, they meet the state-of-the-art performance on unsupervised domain adaptation between CIFAR-10 and STL-10, though not for the reverse. For cross-architecture compatibility, they begin by training a ResNet-56 [42], a Wide ResNet-56 [57], and a MobileNet V2 [58] using a single DCC head on 5 of the 10 classes of

CIFAR-10. By freezing all feature extractors before adding the remaining 5 classes and retraining the DCC head, they produce test accuracies on each model using the same head, reporting only a few percent lost over training independently. Finally, they apply a similar method to unsupervised domain adaptation to achieve high performance transfer-learned models with only 5 epochs of fine-tuning.

While this method is a departure from previous methods focused solely on characterizing CNNs, they still provide insight *while* applying that insight to solve problems. Often the results of representational similarity efforts, though enlightening, can be difficult to apply. In reality, this can even make it difficult to publish such findings. Though efforts without application have still shed light on deep CNNs, applications such as these provide a natural grounding to the phenomena they reveal.

2.2 Bias in Face Recognition

This section provides an overview of the methods used to analyze and mitigate bias in face recognition models. However, the methods used in face recognition are not explicitly covered here. For such an overview, refer to Section 3.2.

2.2.1 Bias Estimation

The authors of a recent survey find many reports of demographically unfair/biased biometrics systems [35]. They note that the vast majority of literature they survey uses *ad hoc* methodologies when discussing algorithmic fairness, citing recent efforts by Howard*et al.* [34] to introduce "differential performance" and "differential outcome" as standard terms for measuring demographic bias. Differential performance refers to comparisons independent of decision thresholds, and differential outcomes refers to comparisons regarding a specific decision threshold. In summary, they find sex is most frequently studied among demographic covariates, followed by race, then by age. Further, face recognition systems are found to be most consistently poor for women, and for the very young or old. For performance comparisons between races, they did not observe a single race

as more challenging, observing the algorithm's country of origin as a major factor. Across the studies they survey, they note inconsistencies in experiment design and bias definition, alongside a lack of control for dataset size, distribution, and confounding covariates (e.g. pose and illumination). Though it contains no original face bias experiments itself, this work places efforts to measure bias in biometrics systems in perspective, especially the methods commonly used.

In the largest experiment of its kind, the National Institute of Standards and Technology (NIST) "has conducted tests to quantify demographic differences for nearly 200 face recognition algorithms from nearly 100 developers, using four collections of photographs with more than 18 million images of more than 8 million people" [31]. These tests measured the performance of commercial face recognition systems on United States government domestic mugshots, immigration application photos, visa application photos, and border crossing photos, finding varying degrees of demographic bias. The results of these tests are extensive, with a few broader takeaways. False negative rates, the proportion of cases where an algorithm falsely predicts two images depict a *different* face, "often vary by factors below 3" between demographic groups. False positive rates, or the proportion of cases where an algorithm falsely predicts two images depict the *same* face, "vary by factors of 10 to beyond 100 times." These rates differ between demographic groups, depending on the algorithm's country of origin and photo source. Some see variations based on algorithm country of origin as evidence for an other-race effect (ORE) [59], a phenomenon where people tend to better recognize faces of their own racial group, which has been researched extensively [60]. For high quality application photos, "false positive rates are highest in West and East African and East Asian people, and lowest in Eastern European individuals." Broadly, false positive rates are higher for women than men, and for the oldest and youngest individuals. Some have challenged these broader findings, stating that the most accurate algorithms do not exhibit significant demographic bias [61]. Still, as of writing it is the largest of its kind, and reveals relevant trends for anyone studying demographic bias in face recognition.

Using the same dataset studied here, Lu*et al.* [33] compared a range of demographic and non-demographic face image covariates. They use a fusion of 4 CNN-based models to produce sim-

ilarity scores for pairs of face images of IJB-B [62] and IJB-C [63]. These similarity scores can be used to make a same/different decision based on a threshold, after which the rate of correctly predicted matched (true positive rate) and incorrectly predicted matches (false positive rate) can be determined. The decision threshold is varied to produce different true/false positive rates, as in receiver operating characteristic (ROC) analysis. This process is standard for comparing face recognition models, and also covered in Section 3.2. To analyze the effects of covariates, separate ROC curves can be drawn for pairs belonging to different covariate groups. The yaw and roll of the face relative to the camera reduced recognition performance, especially when faces of very different orientation are compared. Further, male faces were more easily distinguished than female faces, medium-age faces more than older and younger, lighter skin more than darker, indoor images than outdoor, and faces with mustaches over other facial hair styles. This work is highly relevant, since the same covariates are studied here, albeit using a different comparison method and many newer face recognition models.

2.2.2 Bias Mechanism Analysis

Though neural network face recognition bias is apparent and problematic, its nature is not clearly understood. More broadly, the ways in which deep models interact with any complex data are often difficult to interpret, as mentioned in Chapter 1. By measuring model bias as some quality of the dataset is varied, however, these works shed light on the root causes of face recognition bias.

In Vangara *et al.*, the embedding distributions of Caucasian and African American faces were found to differ [64]. Essentially, they find that a fixed decision threshold will produce worse facial recognition performance for African American faces, where subgroup-specific thresholds will lead to equal or near-equal performance (in this case, false positive and false negative rates). Their results depend on the model used for embedding faces, but still presents subgroup-specific thresholds as an important topic for understanding face recognition bias. Besides demonstrating the need for these subgroup-specific thresholds, they also find little difference in the d-prime statistic between "imposter" and "genuine" image embedding similarity score distributions for each group, suggest-

ing the groups are equally distributed for the model studied. For reference, d-prime measures the distance between two distributions–essentially a measure of how separated two distributions are, with the difference between imposter and genuine distributions corresponding to the ability of the model to separate genuine images (of the same person) from imposter images (of different people). Finally, they note that their evaluation dataset, MORPH [65], contains fewer ICAO-compliant[6] images of African American faces than Caucasian faces, arguing that this may contribute to bias.

Albiero *et al.*similarly find that images of different gender also produce different distributions in embedding space, further motivating the use of subgroup-specific thresholds [67]. Unlike Vangara *et al.*, however, they use more recent, higher-performance models, and more evaluation datasets (MORPH [65], Notre Dame [68], AFD [69]). Further, they observe that images of male faces are more easily separated then female faces, as measured by the d-prime statistic. They investigate whether these differences are due to gender-correlated face image covariates such as face expression, head pose, forehead occlusion (due to hairstyle), and facial makeup. They find that each covariate indeed negatively impacts the separability of female faces over male, though the separability imbalance is not completely remedied once covariate distributions are made equivalent. When balancing for gender in model training datasets, they again find a persisting imbalance between the distribution separability of male and female faces. In a follow-up analysis, Qiu *et al.*find that roughly 3 times more female faces than male are needed during training to obtain gender-equal performance [36]. Intrinsic differences in morphology due to gender are proposed as an explanation in [70], finding that the eyes and lips are most useful for classifying a face as female, while the nose and brow are most useful for classifying a face as male.

In [71], subgroup-specific thresholds are also studied. They evaluate subgroup-specific thresholds for 8 groups (all unique combinations of 4 ethnicity groups and 2 genders), using a new benchmark dataset named Balanced Faces in the Wild (BFW, after LFW [72]). This dataset is a subset of the VGGFace2 [73] dataset, and balanced for gender, ethnicity, and images per individ-

[6]ICAO refers to an image quality standard used by the International Civil Aviation Organization for face images in travel documents, defined in ISO/IEC standard 19794-5 [66].

ual. In comparison to single-threshold face verification, their subgroup-specific method results in improved face verification performance for all groups (though marginally for some).

2.2.3 Bias Mitigation

Producing face models which are unbiased is clearly challenging, as certain groups seem to be intrinsically more difficult to model. This does not ease the burden of any model designer, however. This subsection describes model designs efforts to reduce or eliminate bias.

During training, face recognition models often add an angular margin term to the typical softmax cross-entropy loss, further encouraging the model to de-correlate face images of different people during training. In [74], subgroup-specific margins are learned using a reinforcement learning (RL) technique, where the RL agent controls non-Caucasian loss margins with the goal of balancing inter- and intra-class distances between Caucasians and non-Caucasians. They provide two new training datasets sampled from MS-Celeb-1M [75] and online celebrity images: BUPT-BalancedFace (equally racially balanced) and BUPT-GlobalFace (balanced with the racial distribution of the real world). They train ResNet-34 [25] models with and without their RL-based subgroup-specific margins for each of: Normface [76], Cosface [77], and Arcface [78]. Evaluating using the Racial Faces in the Wild (RFW[7] [39]) dataset, their method improves average face verification performance and fairness (as measured by standard deviation) when combined with any other method. Their method does not produced totally unbiased results, however, with Caucasians remaining the most easily recognized.

In [39], an information maximization technique named IMAN is presented alongside a new racially-balanced evaluation dataset, RFW. This technique is inspired by those within unsupervised domain adaptation (UDA) research, where a model learns features which are invariant to domain (in this case, ethnicity). They break down model training into multiple steps, first training a network to maximally-discriminate a source domain (Caucasian faces), followed by clustering to generate pseudo-labels which are used in classifying target-domain images. They add an additional

[7]Note that they remove overlapping faces from RFW from their BUPT-* training datasets, necessary .

step where mutual information is maximized between the target domain and the classification output. They train ResNet-34 [25] models using this method or Arcface [78], and evaluate each on IJB-A [79], GBU [80], and a new dataset named Racial Faces in the Wild (RFW). RFW is derived from MS-Celeb-1M [75], and inspired by the difficulty of the "ugly" subset of GBU, consisting of equal-sized Caucasian, Asian, Indian, and African subsets in a similar format as LFW. Their method outperforms Arcface on the datasets tested, but again does not deliver equal performance across all races in RFW.

In [38], an adversarial technique named DebFace is implemented for disentangling 4 face attributes from learned representations (gender, age, race, and identity). This method consists of a CNN which predicts one embedding per attribute, all of which are fed to each of 4 attribute-specific classifiers. Adversarial loss is applied such that each classifier can predict its single unique attribute. They also train a baseline model which uses the same architecture, but no adversarial loss, and evaluate both models using a combination of multiple datasets. By removing demographic information (gender, age, race) from identity representations, DebFace suffers from weakened average performance. However, each representation is less biased towards other attributes than BaseFace, even showing near-equal identification performance between genders.

In [37], another adversarial technique is applied for reducing bias, along with a new ethnicity-balanced training dataset, DiveFace. Their method feeds the outputs of a pre-trained face recognition CNN to a second model which is trained to remove information predictive of a given attribute (e.g. gender, ethnicity). They use a ResNet-50 [25] model trained on VGGFace2 [73], then use the DiveFace dataset to train their second adversarial model, before evaluating on LFW [72] and CelebA [81]. After adversarial projection, face features were found to offer far reduced gender, ethnicity, attractiveness, and expression (smiling/not smiling) classification accuracy, with reduced identification performance. This method shows promise for reducing bias in face embeddings, but is limited in evaluation, showing mediocre identification performance on easy datasets.

While the methods covered show promise, the challenge to produce effective, high-performance bias reduction methods remains. In summary, though the models may be incentivized to learn less-

biased representations, bias remains. These methods are relevant to the work presented in this thesis, that an underlying representational similarity persists despite architectural changes. Future works might consider how bias-reduction techniques do or do not impact this underlying similarity.

Chapter 3

Closed-set Linear Equivalence

This chapter summarizes the work first presented in my Master's thesis. Since then, small refinements have been made, but the overall story is largely unchanged.

In this effort, the goal was to compare CNNs which vary in architecture, by their last-layer embeddings. The models studied are all trained for ImageNet (ILSVRC2012 [27]) image classification, where last-layer CNN embeddings are converted into image predictions by a final linear classification layer (after the last CNN layer). By swapping last-layer, pre-classifier embeddings between CNNs after a simple mapping, a fundamental similarity is observed. In essence, each model produces embeddings which can be classified by any other model's linear classification layer, after simple mapping. Though the simple mapping (a linear transformation matrix) can be fit empirically using embeddings from different networks, they can be fit far more precisely using either networks' linear classifier. Though such classifier-based mappings draw the results of the empirically-calculated mappings into question, a fundamental similarity between models of different architecture is still observed.

3.1 Classifier Based Linear Maps

To begin, ten ImageNet-trained networks were obtained which differ in architecture. Each network and its associated weights was obtained from TensorFlow Hub[8] with the exception of Inception-v4, which was obtained from the TensorFlow-Slim GitHub repository[9]. See Table 3.1 for details of the networks and their respective top-1 single-crop classification accuracies on the 50,000 ILSVRC2012 validation samples as reported by Google and reproduced by us[10].

[8]https://tfhub.dev/

[9]https://github.com/tensorflow/models/tree/master/research/slim

[10]A crop size of 331x331 was used to minimize the difference in features encoded into each network's feature vectors, except in the case of MobileNet-v2 which is only available pretrained with a maximum crop size of 224x224. This

Table 3.1: Classification accuracies of the 10 CNNs studied on the ILSVRC2012 validation set, both reported by Google and verified independently (with respective crop sizes). Also included for reference are the number of dimensions used in each CNN's feature space, and the total number of parameters present in the model.

	Reported		Observed			
CNN	Acc.	Crop	Acc.	Crop	Dimension	# Params
Inception-v1 [82]	69.8%	224x224	71.1%	331x331	1024	5M
Inception-v2 [83]	73.9%	224x224	73.9%	331x331	1024	11M
MobileNet-v2-1.4-224 [58]	74.9%	224x224	74.6%	224x224	1792	7M
ResNet-v1-152 [25]	76.8%	224x224	78.8%	331x331	2048	60M
ResNet-v2-152 [42]	77.8%	224x224	78.7%	331x331	2048	60M
Inception-v3 [83]	78.0%	299x299	78.9%	331x331	2048	24M
Inception-v4 [28]	80.1%	299x299	80.4%	331x331	1536	43M
Inception-ResNet-v2 [28]	80.4%	299x299	81.2%	331x331	1536	56M
NASNet-Large [84]	82.7%	331x331	82.7%	331x331	4032	89M
PNASNet-Large [26]	82.9%	331x331	82.9%	331x331	4320	86M

These networks' architectures vary widely, from the simple residual connections of ResNets [25, 42], the hand-built modules of Inception nets [28, 82, 83], the resource-constrained design of MobileNet-v2 [58], to the heavily optimized NASNet and PNASNet [26,84]. Due to the significant methodological differences used for constructing these networks, it seems reasonable to assume they will partition the space used to assign class labels differently.

3.1.1 Classifier Based Metrics

To better lay out this approach, a format that clearly establishes the intended meaning by outlining features/embeddings and their connection to object labeling follows.

Each network presented in Table 3.1 may be considered a function, $v : \mathbb{R}^{w \times h \times 3} \rightarrow \mathbb{R}^{1000}$. The domain consists of $w \times h$ pixel RGB images and the range is the $\mathbb{R}^{1000}$ label space; each dimension represents a network activation corresponding to one of the 1,000 ILSVRC2012 object labels. The function v can be further divided into two parts:

$$v(x) = \mathbf{C}f(x). \tag{3.1}$$

means most networks actually perform slightly better than reported by Google, although MobileNet-v2 performs marginally worse.

Here, $f : \mathbb{R}^{w \times h \times 3} \rightarrow \mathbb{R}^d$ is a highly nonlinear function, mapping from the input image space to a d-dimensional feature/embedding space. The matrix $\mathbf{C} \in \mathbb{R}^{1000 \times d}$ is a linear classifier that transforms a feature vector into class label activations. The argmax of these activations is typically used to determine the predicted class. Note that many traditional neural network systems include some form of bias that would appear to complicate this definition, but by mapping to projective space (always appending a 1 to vector f), the last column of $\mathbf{C}$ can be used to represent the bias.

The goal here is to measure the extent to which a linear mapping converts between the features of two networks. Given two networks, v_A and v_B defined by

$$\begin{aligned} v_A(x) &= \mathbf{C}_A f_A(x) \\ v_B(x) &= \mathbf{C}_B f_B(x), \end{aligned} \tag{3.2}$$

with features of system v_A, $f_A \in \mathbb{R}^{d_A}$, and features of system v_B, $f_B \in \mathbb{R}^{d_B}$, the goal is to measure the extent to which the functions f_A and f_B behave similarly. These functions are unlikely to have identical output or be directly comparable, but the extent to which they have a linear relationship can be investigated. Specifically, does there exist a matrix $\mathbf{M}_{A \rightarrow B} \in \mathbb{R}^{d_B \times d_A}$ such that

$$f_B(\cdot) \approx \mathbf{M}_{A \rightarrow B} f_A(\cdot). \tag{3.3}$$

Naively, calculating distance between the features $f_B(x)$ and $f_A(x)$ for each sample in the ILSVRC2012 dataset using something like Euclidean distance appears tempting, but for a variety of reasons it is not helpful. Most obvious is the problem of possible representational permutations. It is well understood that two networks of identical architecture may permute feature dimensions, yet be in all other ways be equivalent. Additionally, even when setting aside the permutation problem it is also known that distances in high dimensional spaces can sometimes obfuscate results [85]. Further, comparing feature spaces is complicated by possible differences in the number of dimensions from one network to the next.

It is important to mention that a promising avenue to work around such problems is to employ methods such as canonical correlation analysis [3, 86]. However, because such methods are invariant to linear transforms, they give insight into similarity while not serving the goal of constructing and understanding the $\mathbf{M}_{A\to B}$ mappings.

Due to these potential issues with common distance metrics, a useful proxy is to measure similarity is the top-1 accuracy of network combinations: what accuracy is attained by the system using the linear mapping:

$$v_{A\to B}(x) = \mathbf{C}_B \mathbf{M}_{A\to B} f_A(x). \tag{3.4}$$

As shown in Section 3.1.4, under some conditions when $\mathbf{C}_A$ and $\mathbf{C}_B$ are known, a matrix $\mathbf{M}_{A\to B}$ may be calculated directly from them. However, the broader case where such knowledge is not available and hence $\mathbf{M}_{A\to B}$ must be estimated empirically is explored first.

3.1.2 Empirically Calculated Mapping

Although Euclidean distance is potentially a poor measure of similarity between spaces as a whole, it may be used as optimization metric to efficiently construct empirical $\mathbf{M}_{A\to B}$ mappings. Calculating the mapping is a ridge regression problem over all the 1.3 million images in the training set, X, using the Euclidean norm $||\cdot||_2$ and the Frobenius norm $||\cdot||_F$:

$$\underset{\tilde{\mathbf{M}}_{A\to B}}{\text{minimize}} \sum_{x_i \in X} ||\tilde{\mathbf{M}}_{A\to B} f_A(x_i) - f_B(x_i)||_2 \; + \; ||\tilde{\mathbf{M}}_{A\to B}||_F \tag{3.5}$$

The resulting matrix $\tilde{\mathbf{M}}_{A\to B}$ minimizes total point-wise Euclidean distance between the feature spaces.

$\tilde{\mathbf{M}}_{A\to B}$ is then used for pairs of networks to construct the mapped networks $v_{A\to B}$ as defined by Equation 3.4. The recognition accuracy for the mapped networks is then calculated over the ILSVRC2012 test set. This process is completed for each pairwise combination of the 10 pre-trained CNNs and the results are summarized in Table 3.1. One thing to emphasize here is that

Table 3.2: Classification accuracies of 121 inter-CNN linear maps. Each cell represents a single instance of Equation 3.4 with system v_A corresponding to the **row CNN** and system v_B corresponding to the **column CNN**. The number in large font in each cell indicates the accuracy of this hybrid CNN. Diagonal elements correspond to identity mappings, so they are the original network accuracies from Table 3.1. The number in small font indicates the percent change from the original unmapped row/source CNN (i.e. the value in that row which belongs to the diagonal). The darker the shade of red, the greater the performance penalty introduced by the mapping, relative to the feature extractor's own classifier. The last row and column, gray background, presents comparisons with a random untrained control CNN that is the PNASNet Large architecture using randomly initialized weights.

		Target (classifier)										
Source (feature extractor)		Inception V1	Inception V2	MobileNet V2 1.4 224	ResNet V1 152	ResNet V2 152	Inception V3	Inception V4	Inception ResNet V2	NASNet Large	PNASNet Large	PNASNet Large*
	Inception V1	71.06% 0.0%	62.86% -11.6%	68.16% -4.1%	68.45% -3.7%	68.29% -3.9%	67.38% -5.2%	66.75% -6.1%	65.81% -7.4%	64.97% -8.6%	65.41% -8.0%	0.08% -99.9%
	Inception V2	69.87% -5.8%	73.94% 0.0%	72.66% -1.7%	72.75% -1.6%	72.67% -1.7%	72.34% -2.2%	72.01% -2.6%	71.55% -3.2%	70.98% -4.0%	71.26% -3.6%	0.08% -99.9%
	MobileNet V2 1.4 224	68.13% -9.5%	66.17% -11.3%	74.60% 0.0%	71.59% -4.0%	71.21% -4.5%	70.63% -5.3%	69.94% -6.2%	69.47% -6.9%	69.16% -7.3%	69.72% -6.5%	0.06% -99.9%
	ResNet V1 152	73.94% -6.5%	73.36% -6.9%	76.92% -2.4%	78.78% 0.0%	77.01% -2.2%	76.45% -3.0%	76.08% -3.4%	75.84% -3.7%	75.25% -4.5%	75.12% -4.6%	0.08% -99.9%
	ResNet V2 152	74.74% -5.3%	74.35% -5.5%	77.16% -2.0%	77.81% -1.1%	78.70% 0.0%	76.70% -2.5%	76.35% -3.0%	76.06% -3.4%	75.39% -4.2%	75.45% -4.1%	0.07% -99.9%
	Inception V3	75.59% -4.4%	75.57% -4.2%	77.91% -1.2%	77.92% -1.2%	77.63% -1.6%	78.88% 0.0%	77.63% -1.6%	77.48% -1.8%	77.10% -2.3%	77.17% -2.2%	0.07% -99.9%
	Inception V4	78.28% -2.7%	78.50% -2.4%	79.84% -0.7%	79.75% -0.8%	79.57% -1.0%	79.78% -0.8%	80.39% 0.0%	79.66% -0.9%	79.63% -0.9%	79.60% -1.0%	0.08% -99.9%
	Inception ResNet V2	79.61% -1.9%	79.63% -1.9%	80.72% -0.5%	80.77% -0.5%	80.52% -0.8%	80.71% -0.6%	80.82% -0.4%	81.16% 0.0%	80.78% -0.5%	80.70% -0.6%	0.07% -99.9%
	NASNet Large	80.96% -2.1%	81.30% -1.6%	82.44% -0.3%	82.34% -0.4%	82.24% -0.5%	82.42% -0.3%	82.61% -0.1%	82.58% -0.1%	82.66% 0.0%	82.65% -0.4%	0.08% -99.9%
	PNASNet Large	81.15% -2.2%	81.40% -1.9%	82.62% -0.4%	82.51% -0.5%	82.47% -0.6%	82.72% -0.3%	82.80% -0.2%	82.77% -0.2%	82.84% -0.1%	82.94% 0.0%	0.07% -99.9%
	PNASNet Large*	2.50% -97.0%	1.85% -97.8%	4.71% -94.3%	3.66% -95.6%	3.95% -95.2%	4.47% -94.6%	4.82% -94.2%	3.48% -95.8%	5.41% -93.5%	5.47% -93.4%	0.07% -99.9%

neither the ground truth image labels nor the $\mathbf{C}_A$ and $\mathbf{C}_B$ matrices are used in when estimating the linear mappings $\tilde{\mathbf{M}}_{A \to B}$.

One additional control system is also considered. The control is simply the PNASNet-Large network using only the initial randomly generated weights. This control is added to provide some backdrop against which to empirically assess the relative difference between the mapped networks designed and trained to achieve high recognition accuracy.

3.1.3 Classifier Based Mapping Results

Table 3.2 presents the performance of all combinations of CNN features and back-end classifiers. All accuracies are reported using the ILSVRC2012 validation set, which was unseen by all CNNs and mappings during training. In addition, a randomly-initialized PNASNet-Large, denoted by *, is also included.

When comparing column-wise (feeding different features into the same classifier), accuracies are sometimes *increased* when compared to the classifier CNN's own features. That is, even though the mappings were trained to reproduce the classifier CNN's less expressive features, some additional helpful information is passed through the mapping. As an example, see ResNet-v2-152's column and unmapped accuracy of 78.70%. This unmapped accuracy is taking ResNet-v2's features and passing them into ResNet-v2's classifier. When instead fed a linear transformation of PNASNet's features, ResNet-v2's classifier produces 82.46% accuracy. This is an increase in the ResNet-v2 classifier's accuracy of 3.76%! Of course, when analyzing the effects of mappings on the *source* CNN's performance (i.e. the fashion that cells were shaded in Table 3.2), **no feature vectors actually become more discriminative** when mapped. Additionally, the sharpest reductions in accuracy occur for the lowest-accuracy networks. This suggests that as architectures become complex enough to solve the ILSVCR2012 dataset well, they are converging to similar solutions (as predicted by Roeder *et al.* [87]).

The central question is whether the features learned by each trained CNN studied are equivalent. In every case, the percent change in classification accuracy introduced by using another CNN's features is no worse than -11.6% (and in most cases, much better). Indeed, the median percent change over all mappings is only -1.90%. This strongly suggests that the features learned by one CNN are also learned by every other, though some have learned somewhat superior variations.

One additional item of note is that the PNASNet-Large with randomly initialized weights improves from the baseline random accuracy when mapped to the features of other networks. To be clear, the improvement is dramatic from one perspective, jumping by as much as two orders of magnitude. However, it is lackluster at best from another perspective, reaching only about 5

percent accuracy in the best case. This finding lends credence to the view that even just the CNN architecture itself, independent of learned weights, encodes useful information. Indeed, this property of networks has been explored in more depth in a number of different contexts including fast architecture search — see [88–91].

3.1.4 Analytic Mapping

An important caveat to this accuracy-based comparison is that there is a vacuous sense in which the accuracy of the mapped systems can be high while similarity of the feature spaces may be low.

Lemma 3.1.1. *Consider a mapping $\mathbf{C}_A \in \mathbb{R}^{d \times d_A}$ and a mapping $\mathbf{C}_B \in \mathbb{R}^{d \times d_B}$ that is tall (i.e. $d < d_b$) and of full row rank (i.e. rank d). Then, there exists a matrix $\mathbf{M}_{A \to B}$ such that*

$$\mathbf{C}_A = \mathbf{C}_B \mathbf{M}_{A \to B} \tag{3.6}$$

where $\mathbf{M}_{A \to B}$ can be defined as

$$\mathbf{M}_{A \to B} = \mathbf{C}_B^{+} \mathbf{C}_A. \tag{3.7}$$

Here $\mathbf{C}_B^{+}$ is the Moore-Penrose inverse of $\mathbf{C}_B$.

This isn't necessarily an astonishing fact — it is simply a statement that any matrix with full row rank has a right inverse and as such we can write

$$\mathbf{C}_B \mathbf{M}_{A \to B} = \mathbf{C}_B \mathbf{C}_B^{+} \mathbf{C}_A = I \mathbf{C}_A = \mathbf{C}_A \tag{3.8}$$

However, it has the following unfortunate consequence:

Theorem 3.1.2. *Given two systems,*

$$\begin{aligned} v_A(x) &= \mathbf{C}_A f_A(x) \\ v_B(x) &= \mathbf{C}_B f_B(x) \end{aligned} \tag{3.9}$$

as defined by Equation 3.1, if $\mathbf{C}_B$ *is of full rank then then there exists a matrix* $\mathbf{M}^*_{A\to B}$ *such that for all* $x \in \mathbb{R}^{w\times h\times 3}$*,*

$$\mathbf{C}_A f_A(x) = \mathbf{C}_B \mathbf{M}^*_{A\to B} f_A(x). \tag{3.10}$$

where $\mathbf{M}^*_{A\to B}$ *can be defined as*

$$\mathbf{M}^*_{A\to B} = \mathbf{C}_B^+ \mathbf{C}_A. \tag{3.11}$$

Here $\mathbf{C}_B^+$ *is the Moore-Penrose inverse of* $\mathbf{C}_B$*.*

All ILSVRC2012 networks considered here have classifiers that are of full rank.

A few consequences of this theorem:

- For any two linear-classifier based neural networks, there exists a linear mapping such that $v_{A\to B}$ has accuracy exactly equal to that of v_A
- This linear mapping and accuracy is independent of f_B and the accuracy is independent of $\mathbf{C}_B$.

This means that although high linear similarity in feature space implies high accuracy after mapping, one cannot simply say that high accuracy after a linear mapping implies high linear similarity.

As an example, consider a system which when given an image generates a random feature vector, and which uses a fixed full rank classifier. This system has random performance on ILSVRC2012, and features from a real neural network cannot be mapped to its changing "features" in any meaningful sense. However, when "mapping" to its features using the above analytic mapping, the performance of the source network is achieved.

This does not necessarily mean this analytic mapping does not transform between feature spaces well when given features from two networks. It just means that it may not (and clearly does not, in situations like the random classifier), and that the metric being used does not always indicate when the metric may succeed or fail.

To avoid this issue using empirical mappings, computation of the matrix $\tilde{\mathbf{M}}_{A\to B}$ is performed without knowledge of the classifiers and entirely between the two feature spaces. Additionally, the Euclidean distance between feature spaces is used for the optimization target, rather than the result-

ing mapped network accuracy. This means that such a mapping is taking advantage of similarity in feature space when being calculated, rather than accuracy.

Additionally, the random control system described above is included to highlight the distinction being drawn here. In particular, the poor performance as seen in Table 3.2 when mapping to this control system demonstrates that the optimization problem being solved to map between feature spaces likely does not fall prey to the potential issues in the analytic solution shown to exist by Theorem 3.1.2.

Perhaps the simplest way to summarize what has been seen so far, is to recognize the following asymmetry of implications. To start, when Theorem 3.1.2 is applicable there **always exists** an exact linear mapping between two CNNs. However, the existence of a linear mapping is not always a sufficient condition for concluding features are similar. The introduction of the random control addresses this latter point. This distinction also helps spur interest in studying linear mappings between CNNs in contexts where there is no common classifier, and as such Theorem 3.1.2 does not apply.

3.2 Face Recognition

Face recognition modeling offers an exciting area to expand the analyses covered in the previous section. These models are very similar in many ways, but differ in key areas which enable clearer conclusions. Most importantly, face models are evaluated on faces unseen during training, using direct comparison of embeddings rather than a linear classifier. Though linear classifiers are still commonly used during training, they are removed during evaluation. This section provides an overview of the differences between image classification and face recognition.

Specifically, face recognition is performed using aligned images of faces, first produced by a face detection model (e.g. MTCNN [92]). Detected, aligned faces are fed to a CNN to produce an embedding vector, which is converted into a prediction using a linear classifier followed by softmax. Training loss is calculated using the same cross-entropy loss as in image classification, with an added angular margin penalty which encourages faces of different people (negative examples) to

be de-correlated from the faces of the same person (positive examples). The method of computing this angular margin may differ (e.g. Facenet [93], Normface [76], Sphereface [94], Cosface [77], Arcface [78]). As angular loss does not typically operate on embedding magnitude, face vectors are typically first unit-normalized. There is no single training dataset used to train face recognition models, with different datasets leading to different performance (e.g. CASIA-WebFace [95], VGGFace2 [73], MS-Celeb-1M [75], Glint360k [20]).

What's more, evaluation is commonly performed using smaller datasets which do not overlap with popular large training datasets (e.g. LFW [72], IJB-C [63], AgeDB [96], CFP [97], WebFace260M [14]). Many datasets prescribe specific image pairs for performing 1:1 face verification, with IJB-C providing "templates", or groups of faces to first be aggregated before being compared using 1:1 verification. Face verification typically consists of computing distances for each pair prescribed. These distances are also called match scores. Match scores can be converted to same/different judgements by a simple threshold, after which those judgements can be compared to ground truth image pair labels.

The rate at which the same/different judgements agree with labels can be reported, as is common with LFW [72]. Other metrics can also be reported, such as the rate at which same judgements are correct (true positive rate, or TPR), or similarly the rate at which same judgements are incorrect (false positive rate, or FPR). Equivalent to TPR is true accept rate (TAR) and true match rate (TMR), and equivalent to FPR is false accept rate (FAR) and false match rate (FMR). Some choose to also report the rate at which different judgements are correct, or true negative rate (TNR, also true non-match rate, TNMR). Analogously, some also report the reate at which different judgements are incorrect (FNR, FNMR). For reference $TPR = 1 - FNR$ and $TNR = 1 - FPR$.

These rates can be reported at different thresholds for a more complete performance profile, with IJB-C results commonly consisting of TAR (i.e. TPR) at different fixed FAR (i.e. FPR). Alternatively, a receiver operating characteristic (ROC) curve can be shown, which is the TPR as a function of FPR at different thresholds. A summary of this ROC curve is the area under the curve, also the ROC AUC.

Face recognition is an excellent domain for measuring non-performance metrics as well, as many datasets include attributes about the face or image, such as age, gender, ethnicity, face orientation, or lighting conditions. As mentioned in Section 2.2, bias analyses often include many ROC curves for different subgroups of people, highlighting different performance characteristics for different demographics, for example. Indeed, those plots have motivated many to consider different optimal thresholds for different subgroups [64, 67, 71].

Chapter 4

Open-set Linear Equivalence

As discussed in the previous chapter, face verification is an excellent task domain for facilitating linear correspondence studies. In this chapter, similar experiments are conducted using models which may vary in architecture, training dataset, or angular loss function. By comparing models which are not tied together by a common architectural feature, even completely distinct in training data, these experiments offer clear evidence of a fundamental shared similarity structure–a canonical embedding space. Additionally, this chapter includes a sensitivity analysis regarding the number of images necessary to fit a mapping, mappings constrained to rotation, and a discussion of the security implications of interchangeable embeddings. While linear correspondence between closed-set image classifiers is certainly compelling, by establishing linear correspondence between open-set image classifiers, even stronger evidence is established that a fundamental similarity in learned representations exists between modern deep CNNs.

Ten independently trained face-recognition models of various pedigree and performance were selected for study. Models were selected to include a variety of performances, training datasets, CNN architectures, and angular loss functions.

Table 4.3 lists the training datasets, architectures, loss functions, performance on IJB-C, and GitHub source for the ten models. The training datasets used are described in Table 4.1, ranging

Table 4.1: The datasets used in these experiments. The first five datasets were used to train networks used in our experiments. The final dataset, **IJB-C**, is a test dataset used to test mappings between feature spaces.

Dataset	# individuals	# images/video frames
VGGFace2 [73]	9.1 K	3.3 M
CASIA-WebFace [95]	10.5 K	0.5 M
MS1M [75]	100 K	10 M
MS1MV2[11] [78]	85 K	5.8 M
Glint360K [20]	360 K	17 M
IJB-C [63]	**3.5 K**	**148.8 K**

Table 4.2: Computational size and complexity of CNN backbones studied.

Model	# parameters	Flops
MobileNet-v2 1.4 224 [58]	6.1 M	1.2 B
64-CNN [94]	23 M	3.5 B
Inception-ResNet-v1 [28]	24 M	11 B
ResNet-50 [42]	25.6 M	7.2 B
ResNet-100 [42]	44.5 M	13.4 B

Table 4.3: Configuration and accuracy of each model. A shortened name is provided for later reference. Note that these accuracy values are calculated by internal verification and may differ slightly from the stated values for each model's source publication (when available).
*Sources not associated with original publication.

				IJB-C (TAR @ FAR)						
Short Name	**Training Dataset**	**CNN Architecture**	**Angular Loss Function**	**1e-1**	**1e-2**	**1e-3**	**1e-4**	**1e-5**	**1e-6**	**Source**
M2_R100_A	MS1MV2	ResNet100 [42]	ArcFace [78]	0.991	0.984	0.975	0.963	0.945	0.898	[98]
V_R50_A	VGGFace2	ResNet50 [42]	ArcFace [78]	0.994	0.984	0.963	0.928	0.875	0.744	[98]
G_R100_P1.0	Glint360k	ResNet100 [42]	PartialFC (r=1.0) [20]	0.993	0.988	0.981	0.973	0.960	0.912	[98]
G_R100_P0.1	Glint360k	ResNet100 [42]	PartialFC (r=0.1) [20]	0.992	0.987	0.981	0.974	0.961	0.872	[98]
M1_R50_A	MS1M	ResNet50 [42]	ArcFace [78]	0.979	0.954	0.918	0.861	0.782	0.701	[99]*
M1_MB2_A	MS1M	MobileNetV2 [58]	ArcFace [78]	0.981	0.940	0.869	0.766	0.629	0.503	[99]*
V_IR1_C	VGGFace2	InceptionResNetV1 [28]	Center Loss [100]	0.990	0.967	0.908	0.808	0.681	0.518	[101]*
C_IR1_C	CASIA-WebFace	InceptionResNetV1 [28]	Center Loss [100]	0.981	0.929	0.832	0.697	0.534	0.408	[101]*
M1_64S_PFE	MS1M	64-CNN+PFE [19, 94]	AM-Softmax [102]	0.985	0.970	0.942	0.872	0.757	0.610	[103]
C_64S_PFE	CASIA-WebFace	64-CNN+PFE [19, 94]	AM-Softmax [102]	0.982	0.949	0.889	0.798	0.678	0.530	[103]

from 500,000 images in CASIA-WebFace [95] to 17 million images in Glint360K [20]. The relative size and complexity of each CNN backbone is listed in Table 4.2, ranging from MobileNet's 6 million parameters to ResNet-100's 44.5 million parameters. For more details on models and datasets, please refer to their respective publications and code sources.

Using four of the worst-performing models, initial experiments were carried out on LFW (see Appendix A). While these experiments yielded promising results, performance on LFW is highly-saturated using modern CNNs. This motivated the use of IJB-C, a more recent public face benchmark dataset which focuses on unconstrained media, and presents a much greater challenge. IJB-C includes both still images and video frames along with many pre-defined evaluation protocols. These experiments are focused on 1:1 verification, which includes pairs of multi-image templates and a broad range of difficulties. Performance on IJB-C is also typically reported as true acceptance rates at fixed false acceptance rates (TAR @ FAR), which allowed finer measurement of the relative quality of mappings between networks as expressed through ever greater intolerance for false matches.

4.1 Model Evaluation

All 10 models were downloaded pre-trained from their respective sources, and evaluated on IJB-C using the 1:1 verification protocol. Each model was evaluated using its own source repository's preprocessing steps including face detection and cropping. These may differ in crop size, aspect ratio, or similarity transform target, however, the same set of IJB-C images were used in all cases.

Each model was then passed preprocessed images to produce an associated set of embeddings. Embeddings were all evaluated in the same fashion, using evaluation code adapted from Jia Guo and Jiankang Deng's InsightFace project on GitHub [78, 98]. For all 10 networks here, the dimensionality of the feature space is 512. However, with the Probabilistic Face Embeddings (PFE) architecture (models M1-64S-PFE and C-64S-PFE) [19], the output of a second uncertainty module was concatenated to features yielding 1024 dimensions. This uncertainty module consisted of

a network with two fully-connected layers with input and output dimensions of 512, equal to the dimension of the output of the base CNN model.

IJB-C 1:1 verification consists of generating many templates, each corresponding to one or more images. In cases where video frames are present in a template, features belonging to the same video are first aggregated by simple vector average. A single template vector was then calculated as an L2-normalized sum of all image and video features within that template. Template pairs were scored by the inner product (dot product), equivalent to cosine similarity since all templates are unit-length. Finally, a list of template pairs was used for ROC analysis to determine true acceptance rates at fixed false acceptance rates (TAR @ FAR). In the case of PFE, this differed from their mean likelihood score, treating σ simply as an additional feature, allowing a uniform distance measure across all models to be maintained for later cross-model comparisons. The performance of each model at 6 FARs is provided in Table 4.3.

4.1.1 Calculating Mappings

The extent to which a linear map converts between the features of two networks is the chief interest of this study. Let X_E and X_V be the 11,856 enrollment and 457,519 verification images belonging to the IJB-C 1:1 Verification protocol. For a source network f_A and target network f_B, a matrix $\mathbf{M}_{A\to B} \in \mathbb{R}^{d_A \times d_B}$ was fit such that

$$f_B(X_E) \approx \mathbf{M}_{A\to B} f_A(X_E) \tag{4.1}$$

for all input images $x \in \mathbb{R}^{w\times h\times 3}$. Essentially, this approach seeks a mapping which minimizes the distance between pairs of points in feature space corresponding to the same image, up to differences in preprocessing. The result of $\mathbf{M}_{A\to B} f_A(X_E)$ is also explicitly unit-normalized so that it corresponds to the output of the models we studied.

Two methods were used to calculate mappings, both using pairs of embeddings generated from the IJB-C 1:1 verification enrollment set. To elaborate, these 11,856 images were passed to both models to generate 11,856 pairs of embeddings.

Linear mappings were computed by solving the ordinary least squares regression problem over image pairs:

$$\text{minimize} \sum_{i=1}^{m} ||\tilde{\mathbf{M}}_{A \to B} f_A(x) - f_B(x)||_2. \tag{4.2}$$

Rotation mappings were computed using the methods developed by Wahba and Kabsch for finding the optimal rotation for minimizing the distances between two sets of points [104, 105]. Simply put, this algorithm consists of computing the singular value decomposition of the cross-covariance matrix of two sets of points $f_A(X_E)$ and $f_B(X_E)$, followed by recomposition with all singular values set to 1. To ensure no flips or mirroring, the last singular value (corresponding dimension of least variance) is optionally set to -1. In other terms, we calculated rotation mappings as:

$$\begin{aligned} f_A(X_E)^T f_B(X_E) &= U \Sigma V_h \\ M_{A \to B} &= U I' V_h \end{aligned} \tag{4.3}$$

where

$$I' = \text{diag}\left([1 \quad 1 \quad \dots \quad 1 \quad \det(U) * \det(\mathrm{V_h})]\right).$$

This produced a linear mapping matrix with the additional constraint of being orthogonal and having determinant 1, a rotation. Note that $f_A(X_E)$ and $f_B(X_E)$ would typically be centered first to find the optimal rotation axes, but points were left in their original translation (on the unit hypersphere), so that embeddings are rotated about the origin.

4.1.2 Evaluating Mappings

A natural method for mapping evaluation is to measure impact on performance using a validation dataset of faces unseen during training of any models. Essentially, mapped features were produced from the mapping's source network, and evaluated against features generated by the mapping's target network. As in Section 4.1, templates were generated from collections of embeddings, except each template in a pair was generated by a different network.

To be precise, target model templates were calculated from embeddings in the verification set, $f_B(X_V)$, and source model templates were calculated from mapped embeddings generated from the same images $\mathbf{M}_{A\to B} f_A(X_V)$. As in the previous section, template match scores are computed as the inner product, equivalent to cosine similarity when templates are unit-length. ROC analysis is performed to produce true accept rates at various false accept rates (TAR @ FAR), which may be compared to unmapped model performance.

4.2 Cross-CNN Mapping Results

Mapping evaluation results are summarized in Figure 4.1. Hatched bars along the diagonal show the same TARs listed in Table 4.3. Off-diagonal elements contain TARs produced by cross-CNN evaluation as described in Section 4.1.2, with the row label indicating the source model, and the column label indicating the target model. These labels correspond to each model's "Short Name" in Table 4.3. Rotation mapping evaluations are in Figure 4.2.

For the bulk of cross-CNN comparisons, linear mappings seem to convert embeddings effectively and with little performance penalty. When looking for poor mapping performance, Partial FC loss using 10% label subsampling (G_R100_P0.1) seems to produce features which are more difficult or dissimilar. In contrast, the same model trained using all of Glint360K's labels (G_R100_P1.0) consistently produced high performance when mapped (though not necessarily the highest). This suggests that label subsampling, while impacting single-model verification performance very little, has a relatively large effect on embedding space similarity. The contrast in mapping performance between these two models which themselves are highly compatible and near identical in training setting prompts further investigation, as explained in Section 4.4.

Even when constrained to only rotate embedding spaces, cross-CNN performance is still at or near single-CNN performance at a FAR of 0.1. While this FAR is very weak, these results provide a demonstration that mapping between face verification CNNs is possible using linear or rotation maps. Interestingly, some mappings exceed the performance of their target or source model (but not both). Compared with linear maps, rotation produce a higher penalty in almost all cases, and

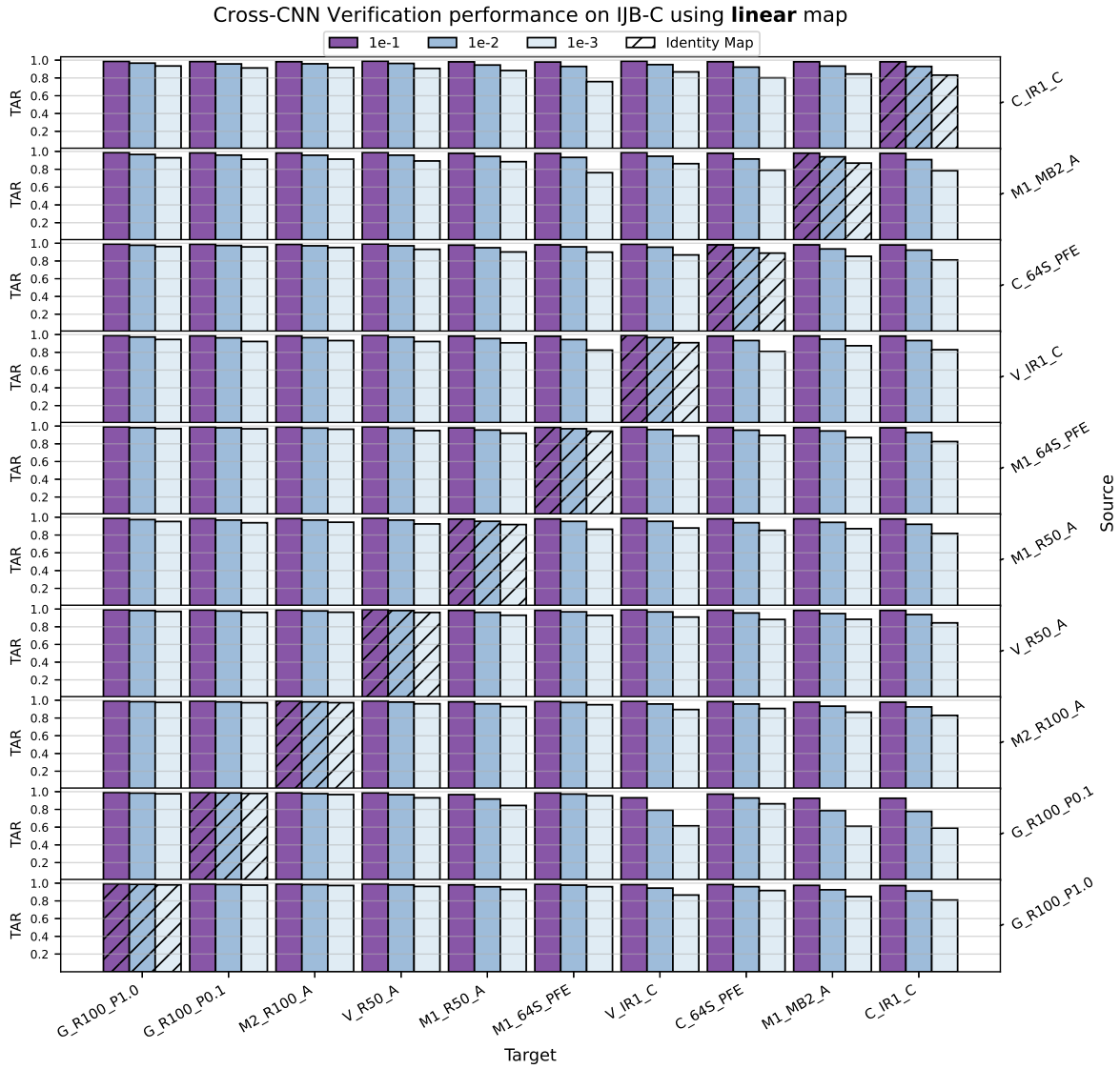


Figure 4.1: Linear maps reveal consistent overlap between feature spaces of distinct CNNs. Bars indicate the TAR on IJB-C 1:1 verification at the FAR indicated by the bar color. Hatched bars correspond to the unmodified performance of each model (also in Table 4.3). Models are sorted according to unmodified TAR at a FAR of 0.01. Off-diagonal bars correspond to the accuracy obtained when comparing features across networks, with the "Source" model's features mapped by linear transformation to approximate the "Target" model's features. Models are referred to by their "Short Name" listed in Table 4.3. Best viewed in color.

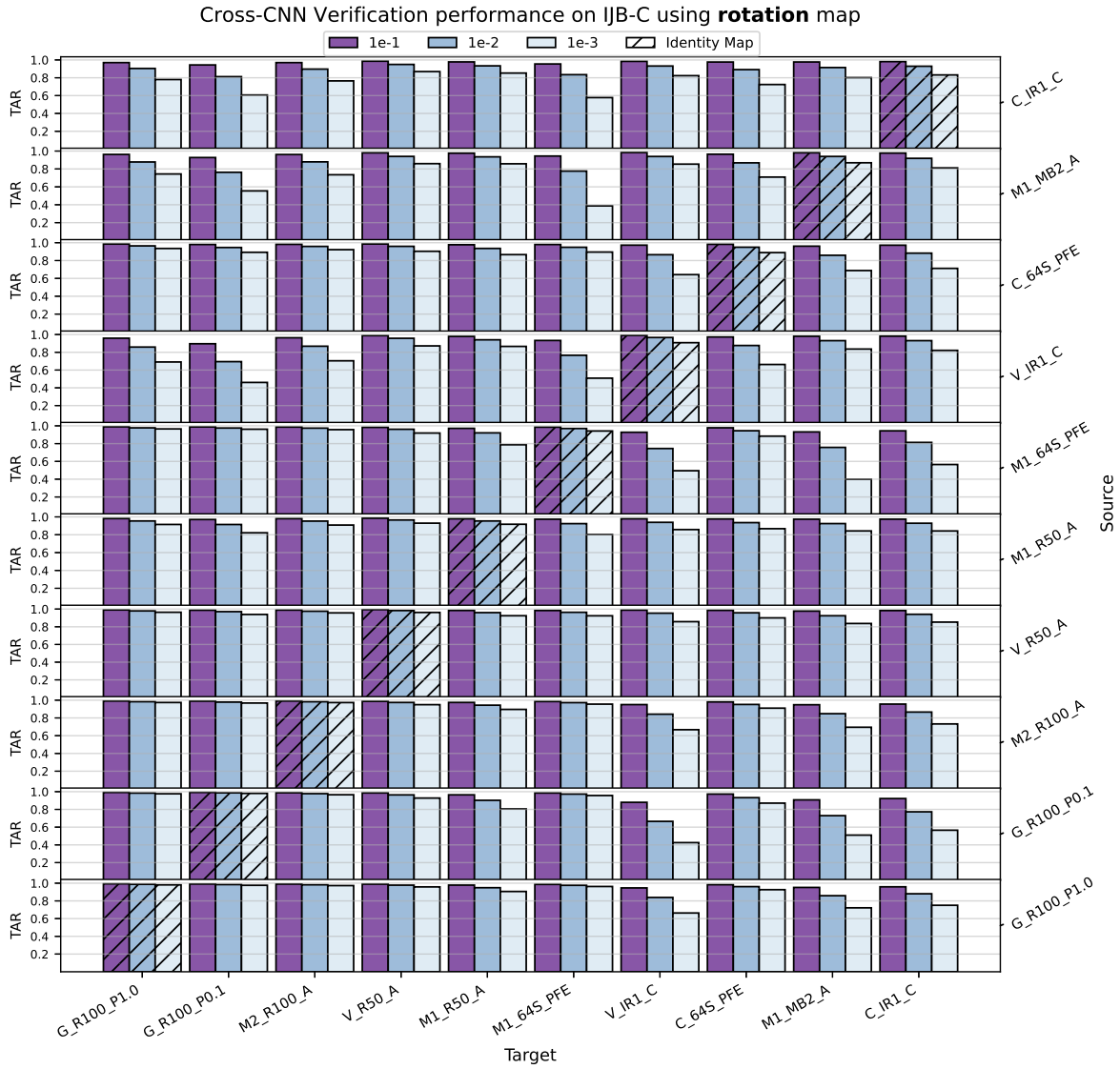


Figure 4.2: Rotation maps also reveal consistent overlap between feature spaces of distinct CNNs. See the caption of Figure 4.1 for figure description.

the penalty was nearly symmetric when source and target are swapped, due to the structure of finding an orthonormal solution. The extra constraint of rotation maps further reveals the nature of cross-CNN relationships.

High TARs were maintained for the most successful networks as the FAR is decreased to 1e-3, though networks with less representational complexity have less success in representing the more complex networks. Smaller FARs are listed with our complete results in Figures B.1 and B.2, including some failure cases which indicated an exciting direction for future work, as discussed in Section 4.4.

4.2.1 Sensitivity to number of images

To better understand the complexity of mappings between embedding spaces, mappings were also fit and evaluated using a variable number of paired examples. Specifically, a random subset of the 11,856 images in the IJB-C 1:1 verification enrollment set were selected, and mappings were fit as described before using a smaller set of images selected at random. Then, mappings were evaluated as before, by comparing the mapped embeddings of one network to the unmapped embeddings of another on the IJB-C 1:1 verification set, after both sets of embeddings have been aggregated into templates. Average performance was collected over 3 repetitions of this experiment using independent random subsets. The resulting TARs at a FAR of 0.01 using all (11,856), 1024, or 256 samples to fit said mappings is illustrated in Figure 4.3. Rotation mappings were used, as they have fewer degrees-of-freedom and thus require fewer examples. For more sample sizes, see Figure B.3 in Appendix B.

Though some mappings performed worse with fewer examples, many provided near unchanged performance. As before, the relative difference in model pairs seem to generally reduce mapping performance. Since Figure 4.3 is sorted by single-model performance model pairs closer to the top-left and bottom-right of this figure are further from each other in relative performance, and also seem to produce generally poorer performance when mapped.

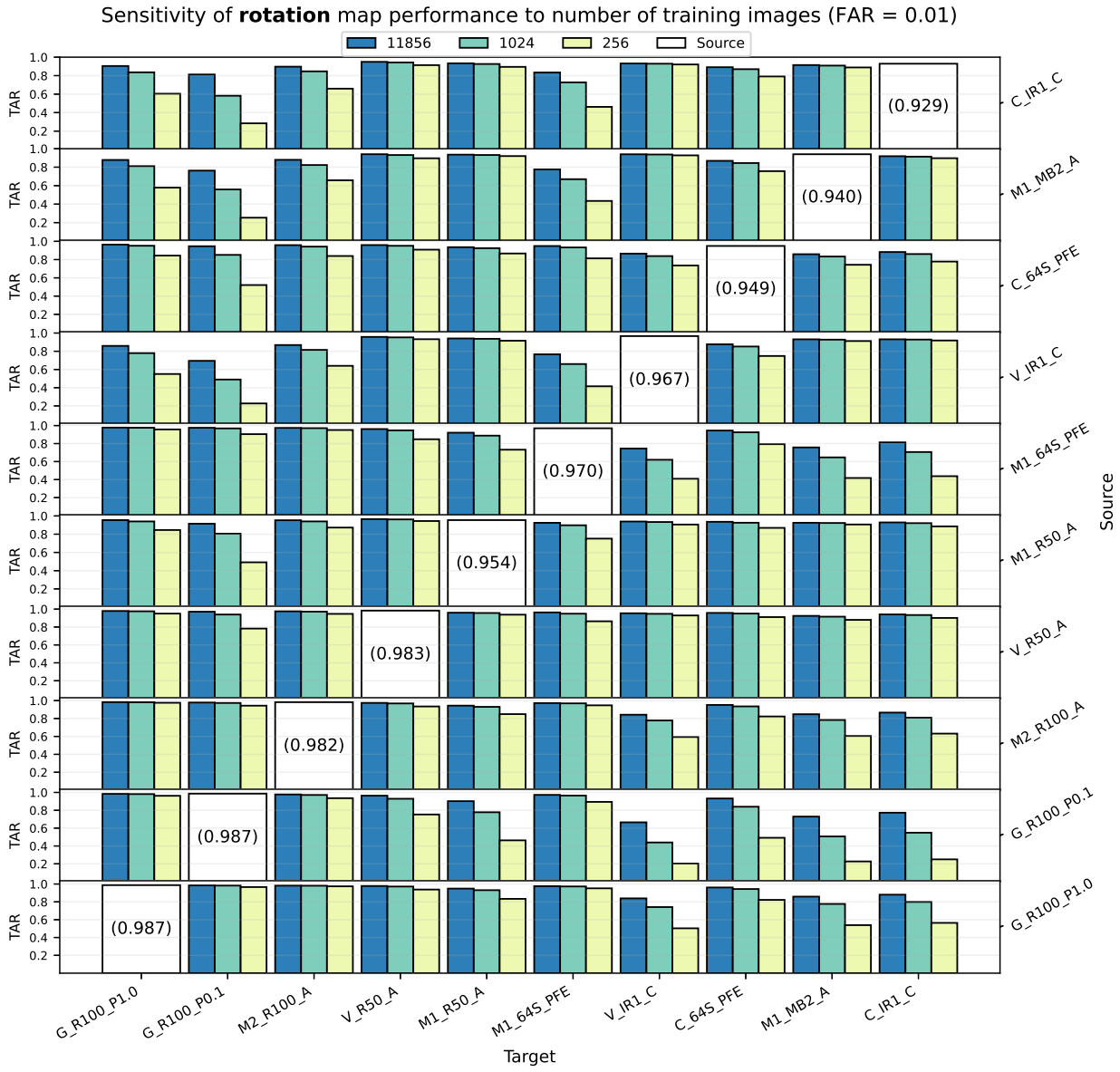


Figure 4.3: Each bar represents the TAR at a FAR of 0.01 achieved by comparing "Source" CNN features to "Target" CNN features after rotation mapping. TAR is shown for rotation mappings computed from 256, 1024 and 11,856 pairs of corresponding embeddings.

These results are broadly encouraging, further confirming the presence of fundamental similarity between embedding spaces. On the other hand, cases where performance is heavily impacted when using fewer examples reveal the relative difficulty of certain CNN pairs over others.

4.3 Security Implications

Anyone handling embeddings from an operational face recognition system based upon existing neural networks must ask themselves this question:

What risks might ensue if embeddings from my system become available to others?

In light of what has been presented, risks include naming an individual associated with an "anonymous" embedding and possibly even enabling an impersonation attack.

To recap and setup for explaining these risks, the experiments above demonstrated that two face embedding systems (Systems A and B) are likely to produce the same embeddings, differing only by a fixed linear transformation. Further, these linear transformations can be determined using relatively few (hundreds) paired embeddings. In other words, if embeddings from System A can be obtained, and their corresponding embeddings in System B discovered (e.g. by also obtaining the faces or identities they represent), then the general linear mapping between embeddings spaces can be calculated. Note also that to establish a mapping between systems, the paired embeddings themselves do not actually need to be labeled. To be clear, they must be of the same person, but the actual identity of the person is not itself used.

Now assuming the mapping between embedding spaces has been determined, how might it be used? Two examples are highlighted. The first is the risk that, even though those responsible for System A might hope an unlabeled embedding is anonymous, there is a clear path to discovering the withheld identity. If System B is tied to a database of labeled faces containing the identity of the “anonymous” embedding from System A, then a simple identity query to System B with the mapped embedding will return the associated identity.

For the second example, consider the “Template Reconstruction Attack” described in Mai et al [106]. In this scenario, an attacker gains access to a face verification model (System A) and

embeddings corresponding to a target individual. Then, the attacker trains a face reconstruction model using System A which converts embeddings into 3-D face representations. Once embeddings can be reconstructed into faces, the attacker can reconstruct the face of a target individual and impersonate them when authenticating using the compromised system (System A). Why the "Template Reconstruction Attack" approach is important here is because, when combined with this work, direct access to System A may no longer be necessary. Instead, a single embedding from System A can be mapped into System B's embedding space and then the reconstruction process proceeds using System B.

The first example is an approach actually put into practice in the context of processing face embeddings from two major participants in the DARPA AIDA program. Each of these participants generated embeddings from a set of images and video frames, associating embeddings with random unique identifiers. Embeddings were provided alongside other document extractions in a non-reversible fashion to simulate conditions where primary sources need remain anonymous (e.g. whistleblowers). The goal of our team was to combine such multi-source extractions into a knowledge base for downstream participants. By pairing embeddings from different AIDA participants with the same unique identifier (thus the same source image and face), we were able to fit linear mappings between the embeddings of unknown models. Ultimately, this allowed us to create correspondences between faces which we had never seen, embedded by models we could not access. The second example, the possibility for enabling an impersonation attack, has not yet been tried and doing so is beyond the scope of this work. That said, the idea is well founded and clearly worthy of further exploration.

In closing, note that these vulnerabilities depend upon the reliability of the mapping. As demonstrated in Figure 4.3, some mappings perform poorly, especially when using fewer examples. Still, from the perspective of security, even a low-reliability vulnerability is worth consideration. More to the point, the goal in this paper has not been to present detailed end-to-end attack models. Instead, the emphasis is that the embeddings — despite being the result of extremely non-linear upstream processes — are not alone sufficient to hide identity. In summary, face em-

beddings and their associated identities should be treated as sensitive information and stored as such. Securing face embeddings against the vulnerability we describe here could be accomplished by an invertible, sufficiently nonlinear transformation applied before storage. For example, strong encryption of embedding vectors should be adequate to guard against the risks cited above.

4.4 Discussion

I am confident that the results presented here are strong evidence for a fundamental similarity between common CNN-based face recognition systems. Though only 10 models were studied, these results suggest that the similarity structure of face CNN embeddings remains stable despite changes in architecture, dataset, and loss. Although some information does not transfer in these linear or rotation maps, clearly the bulk of information does. Where one might expect a new combination and configuration of CNN layers and training examples to produce new learned representations, our work demonstrates that embeddings produced by CNN models trained for the same task approximate a **canonical** space, which each model appears to converge upon.

Generally, the performance of the poorest-performing model in a pair seems to provide an upper bound of cross-CNN performance. However, certain cases perform worse, suggesting certain models have greater compatibility than others, or even that some information is not captured by poorer-performing models. For example, take G_R100_P0.1 (ResNet100 trained on Glint360k using Partial FC loss) which performs comparatively well on its own, and in many cross-CNN mappings. This model is already somewhat distinct in that it uses a random subset of 10% of identities for computing softmax loss during training [20]. Take note, however, of the target models which produce poor mapping performance, V_IR1_C, C_IR1_C, and M1_MB2_A. While these models perform worse already, they seem to have greater incompatibility with G_R100_P0.1 than others. Roughly speaking, if model A is compatible with model B, and model B is compatible with model C, then why isn't A compatible with C? One hypothesis is that, despite evidence for broad task-driven similarity between CNNs, certain design features of these CNNs do impact finer aspects of their representation. In other words, it may be that subsampling identities during

training as with Partial FC biases the resulting model away from a representation found using InceptionResNet or MobileNet architectures.

While results were demonstrated using 90 pairs of CNNs, it's unclear how small variations in preprocessing or model implementation may contribute to mapping performance. While many architectures and results for a large face dataset were included, finer characterization of these relationships requires carefully controlling for architecture and training details. Such analysis will likely come at great resource cost, so a systematic exploration with increased granularity and controls is left to future works.

Further, this method may be explicitly lossy when features are represented with different numbers of dimensions, producing non-square mapping matrices (i.e. non-zero nullity per the rank–nullity theorem). The work of Gong *et al.* [107] offers insight along these lines by providing evidence for a far reduced "intrinsic dimensionality" produced by common face verification CNNs. This suggests that while rectangular linear transformations indeed project information into fewer dimensions, mapping between two sets of face embeddings may require far fewer dimensions than the maximum rank of a rectangular matrix.

4.5 Conclusion

The existence of performance-preserving linear mappings between face recognition CNNs which vary in training and construction suggests this phenomenon depends primarily–if not solely–upon the modeling task. Subsequently, the existence of task-dependent canonical embedding spaces suggests there is a high degree of redundancy in the training procedure of bespoke CNN-based models for a common task, as the bulk of the learned representation is unchanged. Consider the computational effort required to train and develop these models, given that they each converge to highly similar solutions. Perhaps there is a richer training signal to-be-developed which takes advantage of this seemingly universal similarity structure, efficiently guiding training towards solutions found by previous models. Regardless, these results seem to discourage efforts to optimize

CNN architectures for the purpose of improved performance, so long as the modeling task (i.e. the dataset) provides the biggest impact on the content of modeling output.

More broadly, if the modeling task most significantly impacts the structure of embedding space, it prompts further investigations into the relationship between these spaces and the modeling tasks which produced them. Others have investigated the relationships between modeling tasks, finding where learned representations of one task help or hurt performance on another task [108, 109]. Besides the impact on performance, cross-task representations could also be studied by their interactions in embedding space structure. Clearly, more work is necessary to understand how these structures are organized, as our own work depends upon performance as a downstream measure of similarity. Still, if researchers want models to learn fundamentally new representations, this work suggests they need new tasks.

Chapter 5

Analyzing Face Embeddings using Subspace Angles

As discussed in Chapters 1 and 2, deep model bias is prevalent and complex. In Chapters 3 and 4, evidence was found suggesting models trained on the same task learn to extract information in fundamentally similar ways as represented in their output or embedding spaces. In essence, models trained with different datasets and/or different architectures produce outputs (i.e. embedding vectors) which can be interchanged by a structure-preserving mapping. This suggests that a common output space is learned by models which are trained for face recognition, and perhaps other tasks. In this chapter, the structure of face embeddings is further investigated in an effort to better understand how information in the face image is represented in embedding space. This is an attempt to build on the findings of previous chapters, offering a new perspective into how deep face recognition models encode bias. Bias is defined as a meaningful difference in the performance or outcomes of the model with respect to specific input image *covariates*. For the face images studied here, these covariates include both demographic information (age, skin tone, gender) and non-demographic information (lighting, occlusion, orientation). Demographic biases are of particular interest for their potential to disenfranchise certain demographic groups, especially those which are already subject to institutional or systemic biases (see [32] for notable examples).

Estimating the degree to which a face model is biased is commonly performed by measuring the relative performance the model achieves between different covariate groups, such as faces of different age, skin tone, or gender (see Section 2.2 for an overview and notable examples). Specifically, images are fed to a model which produces high-dimensional embeddings, after which those embeddings can be compared using an embedding distance measure, such as cosine distance. Subsequently, a decision threshold is used to convert distances (i.e. match scores) into same/different judgements which are scored against ground truth labels. Typically, the rate at which faces of the same person are below the threshold (the true positive rate, TPR) is reported at different fixed false positive rates (FPR), determined using the same match scores but different decision thresholds.

Though methods for choosing groups of faces and comparing match scores may differ slightly, this ROC analysis framework describes the bulk of face recognition model bias analysis [35].

The method described in the following section is proposed as an alternative to pairwise similarity metrics (i.e. match scores). Simply put, when different images of the same person are presented to a good face-recognition CNN, the output embeddings will not identically match, but they will be similar. Embeddings are nothing more than points or vectors, though CNN-based face embeddings tend to be in very high dimension space. Embeddings produced by images of the same person will be closer to each other than to embeddings of new people. Face verification studies (such as in the previous chapter) will measure the rates at which embeddings of the same person are closer to each other than to embeddings of new people.

The new work being presented here is investigating a different question: should randomly-chosen embeddings of female faces be any closer or farther than randomly-chosen embeddings of male faces? What about other covariates like age, race, or lighting? Pairwise comparisons have been used to investigate these questions, as covered in Section 2.2, painting a clear picture that modern face recognition models are indeed often demographically biased, e.g. distinguishing male faces more accurately than female faces. However, these comparisons do not study how bias is represented beyond the relative discriminability of each covariate groups' embeddings. Instead, the rates at which embeddings of the same person are closer or further to embeddings of different people are simply split by the respective covariate being studied. When these rates differ in aggregate, one can say that faces with attribute A tend to be harder or easier to recognize by the model than faces with attribute B. This is not to say that these studies are not valuable, only that a more direct and efficient comparison may yield a clearer picture of the ways in which biases are encoded into deep neural face recognizers. To get to the bottom of the question posed, a method is needed which compares how embedding groups interact and overlap, not just how well-separated each group is within itself (i.e. how well each group's *identities* are separated).

The new research presented here also builds upon the findings of the previous chapter: models trained for face recognition produce fundamentally similar embedding spaces. The methodology

used represents face image covariates as linear subspaces, and compares them using principal angles. A generalization of linear subspaces is used to facilitate direct and efficient comparison, called the Grassmann manifold. Within this abstraction, groups of embeddings are represented as the linear subspaces they each span, or points on a Grassmann manifold. To investigate the question asked previously, these groups can be selected based on attributes of the face or image used to produce them, such as whether the embedding represents a face of a person who is male or female. As an aside, gender is represented as a binary variable here since this is how gender is annotated in the datasets studied. The separation between these groups can then be measured as the length of the geodesic between their representative points on a Grassmann, or its prerequisite principal angles as described in the following section. Face covariate groups which produce embeddings in distinct subspaces will produce longer geodesics, and conversely covariates which produce embeddings in overlapping subspaces will produce shorter geodesics.

It is not immediately clear what to expect from these measurements, or how bias is exhibited by subspace separation. In two bias mitigation efforts, information which is predictive of different demographics was removed from embeddings, reducing the differences in performance between demographics (and overall, slightly) [37, 38]. If bias-mitigated embeddings are not predictive of demographics, then they surely span overlapping subspaces, since separation could be used for prediction. The converse is not necessarily true, however, as there may be unbiased embeddings which remain predictive of some facial covariate, thus well-separated in embedding space. Instead, it may be that subspaces are distinct, yet still distribute faces of different covariate groups unequally. Indeed, there is growing evidence that faces of different demographics produce different distributions in embedding space. In a bias mitigation effort by Wang *et al.*, demographic bias was reduced by learning specific loss margins (essentially the penalty for wrongly matching two faces) for each demographic group [74]. In another bias analysis, Albiero *et al.*found roughly 3 times more female faces than male faces were necessary during training to achieve gender-equal performance [36]. If the model needs unequal incentives or unequal proportions of training samples to achieve equal performance, perhaps unbiased face representations do not distribute informa-

tion equally. It is also possible that subspaces are overlapping in some dimensions, but distinct in others, perhaps highlighting unequal distributions. Withholding further speculation, more work is clearly needed to understand how face recognition bias is represented in embedding space.

5.1 Measuring embedding subspace similarity

In this experiment, Grassmann manifolds will be utilized for comparisons of embedding subspaces. From a practical standpoint, representing face embedding groups as points on a Grassmann manifold facilitates comparison based on the respective regions of embedding space that they span– their respective subspaces. In more precise terms, the Grassmann manifold $Gr(k, V)$ is a space that parameterizes all k-dimensional linear subspaces of the n-dimensional vector space V. Then, any given point $P \in Gr(k, V)$ represents a specific linear subspace of V. These points can be defined as the space spanned by collections of face embeddings $P = span(Y)$.

To compare two points $P_1, P_2 \in Gr(k, V)$, the length of the shortest path along $Gr(k, V)$ connecting them (a geodesic) can be measured. This geodesic distance is the two-norm of the vector of principal angles between these two subspaces,

$$dist(P_1, P_2) = \left(\sum_{i=1}^{p} \theta_i^2 \right)^{1/2} \tag{5.1}$$

[110]. To obtain these principal angles for two sets of face embeddings Y_1, Y_2, orthonormal bases for each are first computed. Here singular value decomposition (SVD) is used,

$$\begin{aligned} U_1 \Sigma_1 V_1 &= Y_1 \\ U_2 \Sigma_2 V_2 &= Y_2. \end{aligned} \tag{5.2}$$

Then, SVD is applied again on the cross-covariance of these bases,

$$U_d, \Sigma_d, V_d = SVD(U_1^T U_2). \tag{5.3}$$

The values $\sigma_i \in \Sigma_d$ are the cosine of the principal angles between the spaces spanned by Y_1, Y_2, also the canonical correlations [111]. These are converted to angles $\theta_i = arccos(\sigma_i)$, and used in the above formula for $dist(P_1, P_2)$. These angles can also be analyzed individually.

For illustration, imagine taking a unit vector from the subspace which spans all male face embeddings, and another from the subspace which spans all female face embeddings. These two vectors need not be represented in the original set of embeddings used to generate them, and may instead be a linear combination of either set of original embeddings. For the first principal angle, the two vectors are chosen such that they maximally correlate. If the two subspaces overlap, this angle will be zero. The next principal angle is found in the same way, with the added constraint that the maximally-correlating vectors be orthogonal to those chosen for the prior principal angle. In other words, the space spanned by the two vectors measured for the first principal angle (i.e. a hyperplane) is taken out of consideration for subsequent principal angles. If the subspaces overlap along multiple orthogonal directions (i.e. dimensions), this angle will again be zero. This process is repeated, measuring the angle between vectors of maximum correlation which are orthogonal to all previously-chosen vectors, for a specified number of iterations, or until all directions spanned by either subspace have been exhausted.

The description above accurately captures the essential quality of principal angles, but computationally they are solved in terms of a generalized eigenvalue problem (e.g. see the appendix of [2]), computed using the steps listed in the previous paragraph. Still, the angles represent those illustrated–the angle between the closest linear combinations of face embeddings in either space (and orthogonal to prior combinations).

Now that principal angles are obtained, they may be analyzed. In essence, these angles describe how similar two sets of images are in embedding space. If many angles are close to zero degrees, the spaces are highly overlapping, meaning the model represents those faces in a similar manner. If many angles are close to ninety degrees, the spaces are minimally overlapping, indicating that the model represents those faces differently. In the context of face recognition model embeddings, the

profile of angles between subspaces depicts how similarly the model represents faces from either subspace.

5.1.1 Addressing noise through dimension reduction

As the CNNs studied operate on natural images subject to pixel noise, the baseline similarity measured as proposed is likely to be generally high and inexpressive. If for no other reason, this is because pixel noise will span the space given sufficient samples. Essentially, though the model's task requires it to extract a useful signal, its outputs still contain some of the noise present in any given set of natural images. If each subset spans all dimensions, then they will each have perfect similarity as measured by principal angles. So, a dimensionality reduction process is included, where dimensions of low variance are removed with the hope of removing noise. Thankfully, researchers have found that dimension-reduced face embeddings of some CNNs can still be used to perform accurate face verification, using linear dimension reduction by principal components analysis from 512 to 128 dimensions with negligible performance change [107].

Reducing dimensionality in the calculations above is performed by removing columns of the orthonormal bases U_1 and U_2 generated using SVD as described in Equation 5.2. For reference, each of the models here produces embeddings in 512 dimensions. The number of remaining columns is either fixed at 64, for example, or can be determined using a target variance proportion. As an example of the latter, if the first 59 squared values along the diagonal of Σ_1 sum to 99% of all squared values along the diagonal of Σ_1, then only those first 59 columns of U_1 are used to calculate principal angles. As the number of values of Σ_1 expressing a given proportion of variance may differ from the respective number of values of Σ_2, the number of columns removed from U_1 and U_2 may also differ, meaning the matrix product $\hat{U_1}^T \hat{U_2}$ cannot be computed. In this case, the maximum of the two numbers of columns is kept for both U_1 and U_2.

This work includes results for different levels of dimension reduction, to test the notion that dimensionality reduction is favorable for the noisy, sparse, high-dimensional embeddings studied here. If there is meaningful separation to be found between subspaces, projecting them into

fewer dimensions expressing the greatest variance will facilitate greater distinction than the original high-dimension low-variance ambient space. Dimension reduction efforts must be conducted carefully, as too few dimensions removed may not remove enough noise to meaningfully separate the respective spans of each subset.

5.1.2 Baseline similarity estimation

Though this method is motivated by the concept of comparing subspaces, it is ultimately an empirical measure based upon samples of embeddings. As such, determining baseline similarity is also an empirical task. After all, even when two faces are highly distinct regarding their covariates, they still both likely have eyes, a nose, etc. Therefore, a baseline level of similarity is first measured.

This baseline should represent the "average face", and thus not be biased towards any specific covariate group. In other words, a roughly equal number of male and female faces should be present in each random subset, along with other covariates like age, skin tone, etc. Specifically, many random, non-overlapping embedding subsets will be gathered such that covariate groups are roughly equally represented. Then, the principal angles they produce can be determined as described previously. Many principal angles can be gathered from new random subsets, and aggregated to produce a baseline similarity profile for faces which is independent of covariate differences.

5.1.3 Embedding generation and selection

With a method for baseline-aware, noise-insensitive subspace comparison defined, now the subspaces themselves can be constructed. This experiment builds upon the previous chapter using the same 10 models for study. Recall that these models are trained on one of many datasets, using one of many architectures and losses. Filtering to those with annotations, 141,226 images and video frames were obtained from the IARPA Janus Benchmark – C (IJB-C, [63]) dataset. Images are preprocessed (face detection, cropping, similarity transformation, and normalization) in the same manner as each model was trained, before being fed to that model. Each model consists of a

deep convolutional neural network, using many convolutional layers to transform the input image into an output embedding vector and then unit-normalized, as is standard for each.

There are several relevant face covariates included for the face images in the IJB-C dataset. This allows embeddings to be grouped according to different labels, such as female vs male. These covariates are the age, gender, and skin tone of the person pictured, along with whether the image was captured indoors or outdoors, the orientation of the face relative to the camera, and which parts of the face are occluded in the image [63]. Gender is provided as a binary variable, female or male. Here, age is grouped into the same categorical ranges as [33], namely 0-19, 20-34, 35-49, 50-64, and 65+. Skin tone is provided as one of 6 types within the Fitzpatrick scale used by dermatologists [112]. Demographic covariates (age, gender, skin tone) are of particular interest to the research community, and to the greater public concerned with bias in face recognizers. Non-demographic covariates (facial hair, indoor/outdoor, yaw, roll, occlusion) are of less interest, but still present an opportunity to better understand embedding space structure. Unfortunately, as there are relatively few images annotated within the facial hair, yaw, roll, and occlusion covariate values, they are not included in this analysis.

5.1.4 A Sanity Check

Prior works have successfully revealed many ways in which face recognition models are biased (see Section 2.2). In this work, groups of faces are represented as groups and compared as groups, with the hope that such Grassmann-based measures will provide further insights on this bias. To facilitate clear comparisons between these two methods, pairwise distance metrics are also included here.

In short, pairwise angles will be computed for every unique combination (i.e. the Cartesian product) of two sets of embeddings, and averaged. The same sets will be used as discussed in prior subsections, both for establishing baselines and for comparing covariates. These averaged distances can be compared alongside geodesic distances, allowing validation of either as a more expressive metric for comparing sets of embeddings.

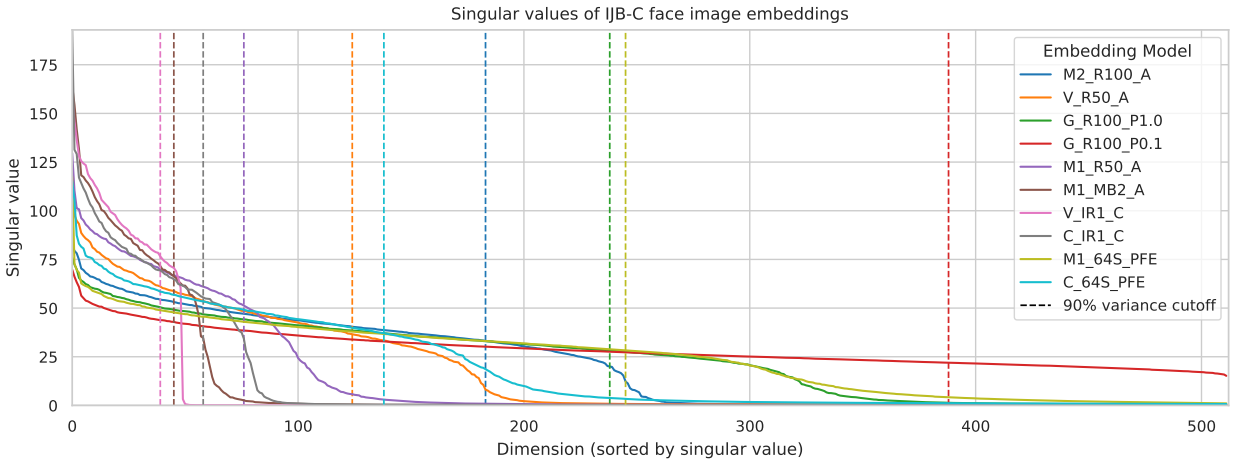


Figure 5.1: The 10 models studied cover a broad range of singular value distributions.

5.2 Results

5.2.1 Effects of dimension reduction

As mentioned previously, some dimension reduction is necessary to prevent noise in each subspace from spanning the ambient space. Though each model produces embeddings vectors in R^{512}, some express information more "compactly" than others. This is illustrated by the singular values of each model, displayed in Figure 5.1 using the short model names listed in Table 4.3. This figure includes a vertical dashed line for each model, showing the number of dimensions necessary to express 90% of its embedding space variance. The variety of dimensional "utilization" has implications for measuring geodesic distance between points on a Grassmann manifold, as each manifold $Gr(k, n)$ is parameterized by the dimension of the subspaces and ambient space, k and n.

Two methods of dimension reduction are considered, fixed and variance-proportional. For both of these methods, the singular value decomposition of a subset of each model's embeddings is used to determine which dimensions are removed, i.e. those with the smallest variance. Again, dimensions in which the data varies the least are removed with the intuition that such dimensions contain a greater proportion of pixel noise, reducing the expressivity of subspace angles. With fixed dimension reduction, each model's embeddings are projected to the first N dimensions, where N is

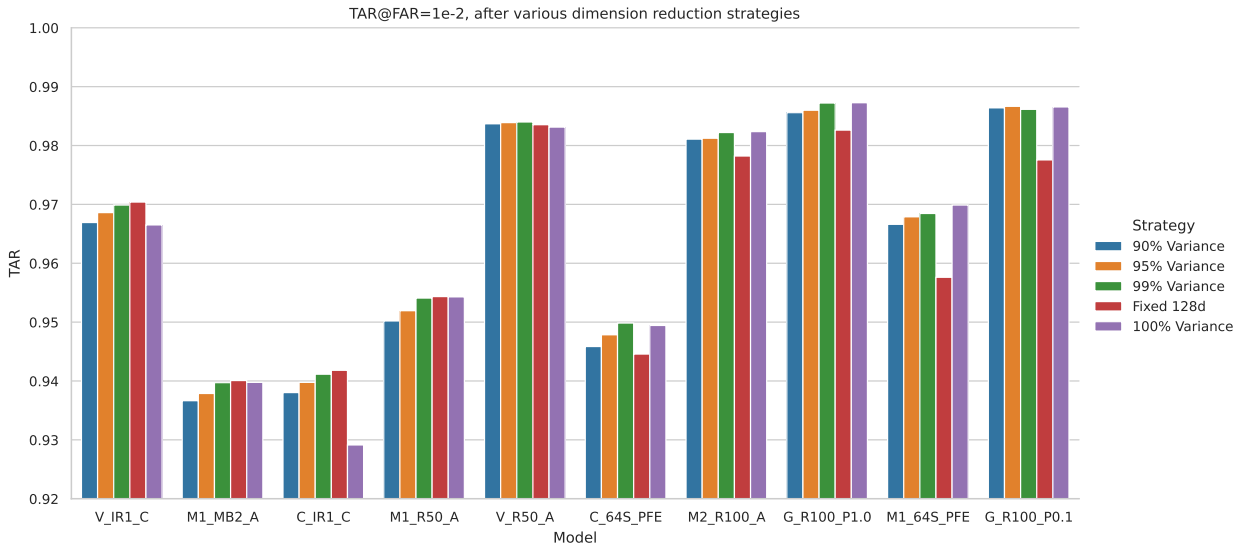


Figure 5.2: Reducing dimensionality of features to a fixed number (here 128) results in non-uniform changes in performance, while variance-proportional dimension reduction produces more uniform, consistent changes. The IJB-C 1:1 verification TAR is reported at FAR=1e-2. Models are sorted by the proportion of variance retained at 128 dimensions.

fixed for all models. With variance-proportional dimension reduction, the proportion of variance is fixed for all models, thus N may vary between models. Essentially, fixed dimension reduction yields embeddings which are all the same dimensionality, while variance-proportional yields embeddings which contain the same proportion of their original embeddings' variance.

Naturally, fixing the number of dimensions to 128 leads to different proportions of variance retained for each model. Compared to variance-proportional, fixed dimension reduction has a less uniform effect on performance, as illustrated in Figure 5.2. For the first few models, the red bar (performance at 128 dimensions) is above the others (variance-proportional dimension reduction), while the trend is reversed for the latter models. In the interest of preventing the "compactness" (i.e. relative proportion of variance) of each model's embedding space from confounding other experimental controls, a variance-proportional strategy is chosen.

5.2.2 Summarized results

As there are 10 models, 4 covariates, and many combinations of covariate values therein, these experiments have produced extensive results. Each covariate value combination (e.g. Female

Table 5.1: Configuration and accuracy of each model, including after dimension reduction retaining 90% of variance. A shortened name is provided for later reference. For full performance and model sources, refer to Table 4.3.

				IJB-C (TAR @ FAR=1e-2)		Number of Dimensions	
Short Name	**Training Dataset**	**CNN Architecture**	**Angular Loss Function**	100% Variance	90% Variance	100% Variance	90% Variance
M2_R100_A	MS1MV2	ResNet100 [42]	ArcFace [78]	0.984	0.981	512	183
V_R50_A	VGGFace2	ResNet50 [42]	ArcFace [78]	0.963	0.984	512	124
G_R100_P1.0	Glint360k	ResNet100 [42]	PartialFC (r=1.0) [20]	0.988	0.986	512	238
G_R100_P0.1	Glint360k	ResNet100 [42]	PartialFC (r=0.1) [20]	0.987	0.986	512	388
M1_R50_A	MS1M	ResNet50 [42]	ArcFace [78]	0.954	0.950	512	76
M1_MB2_A	MS1M	MobileNetV2 [58]	ArcFace [78]	0.940	0.937	512	45
V_IR1_C	VGGFace2	InceptionResNetV1 [28]	Center Loss [100]	0.967	0.967	512	39
C_IR1_C	CASIA-WebFace	InceptionResNetV1 [28]	Center Loss [100]	0.929	0.938	512	58
M1_64S_PFE	MS1M	64-CNN+PFE [19, 94]	AM-Softmax [102]	0.970	0.967	512	245
C_64S_PFE	CASIA-WebFace	64-CNN+PFE [19, 94]	AM-Softmax [102]	0.949	0.946	512	138

vs Male) yields N principal angles when comparing subspaces of R^N. Principal angles can be aggregated into geodesic distances along the Grassmann manifold $Gr(k, n)$, though there are still many distances to compare given the number of models and covariates. Further, as discussed previously, a pairwise distance metric is also included for comparison, doubling the number of distances to analyze. With this in mind, 4 models are highlighted here, with the remainder in Appendix C. These 4 models are selected such that none of them use the same training data, neural network architecture, nor angular margin loss. For the names, configuration, and performance of these models, see Table 5.1.

See Figure 5.3 for the geodesic distances between face embedding subspaces organized by embedding model and covariate. Each colored cell corresponds to the geodesic distance between two subspaces spanned by different face embeddings. Cells are grouped according to their prerequisite embedding model (by figure column) and covariate (by figure row). The upper-rightmost cell in each grid is annotated in italics with the geodesic distance between subspaces spanned by random faces, with no shared identities. All other cells are annotated according to comparisons between face embeddings belonging to specific covariate values (e.g. Female vs. Female, Female vs. Male) again with no shared identities. Cells along the diagonal represent comparisons between subspaces of the same covariate value, but different identities. Each distance is the result of 7 random samples of up to 2,000 face embeddings, with the standard deviation annotated in parentheses. Cells corre-

sponding to the 0-19 age group are the exception to this, as only 1,170 embeddings are available for comparison. For the remaining models, see Figure C.1 in Appendix C.

At first glance, Figure 5.3 illustrates a few broader trends for those hoping to measure bias using geodesic distance between covariate-valued points on the Grassmann. Scanning left to right (comparing models), each column spans a narrow range of the color bar, which seems to be related to the number of embedding dimensions. Scanning top to bottom (comparing covariates), each row includes a similar relative pattern, where colored cells along the diagonal are near the baseline distance.This figure presents values along a single color scale in an effort to present results as plainly and objectively as possible.

Interestingly, cells in the bottom-left tend to set the upper bound for each model. For covariates with more than 2 values (age range and skin tone), the trend is more consistent, that the further a cell is from the diagonal (i.e. the larger the difference in face age or skin tone), the larger the geodesic distance. These trends support the notion that the studied geodesic distance measure is expressive of semantic differences in demographics. Comparing face image embeddings captured indoors or outdoors in the last row, distance measures are least absolutely distinct from random baseline, though still slightly different as seen by their numeric annotations.

Recall that aggregated pairwise angles (i.e. inverse cosine of cosine similarity) are also studied for comparison to subspace angle distances. More precisely, the average angle between all combinations of two groups of embeddings is used in place of geodesic distance between the subspaces those two groups span. These angles are depicted in Figure 5.4, which is in the same format as the previous Figure 5.3. The same procedure is followed, including reporting the mean and standard deviation of the angles between 7 repeated random samples. For consistency, pairwise angles are reported between the same dimension-reduced embeddings as with geodesic distances, though the effect of noise on pairwise angles is not as large of a concern as evidenced by the lack of dimension reduction during typical face verification performance evaluation.

Again scanning column by column, the dimensionality of each model's embeddings seems to affect the range of the distances observed. Though the random baseline is shifted, the relative

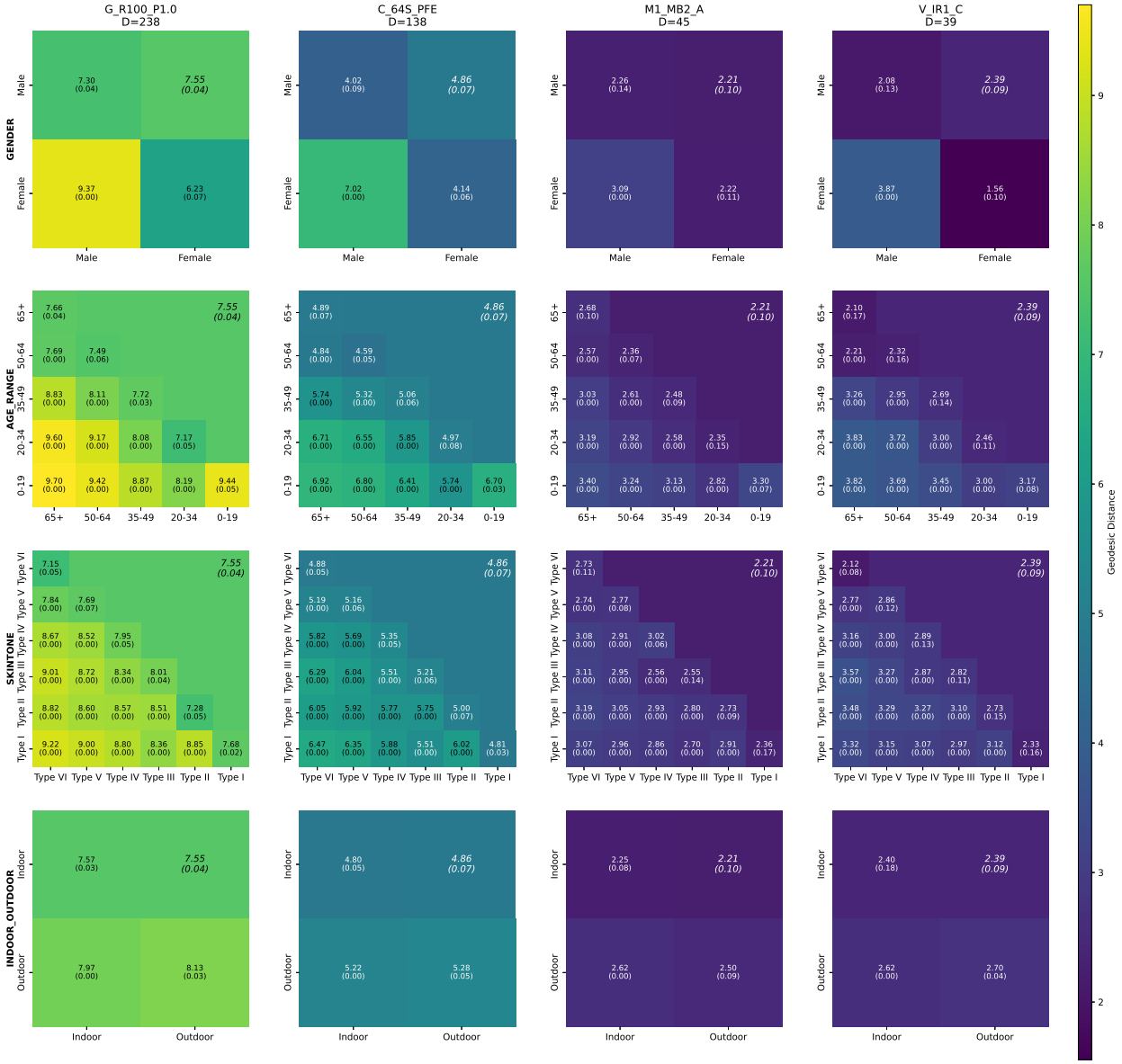


Figure 5.3: Geodesic distances express differences in gender, age range, skin tone, and lighting (best viewed in color, magnified). Distances are calculated according to the method described in Section 5.1, without any normalization or standardization. This figure is organized as a grid of grids, with each colored cell corresponding to a geodesic distance between subspaces spanned by IJB-C face embeddings. Each figure row corresponds to a different covariate, while each figure column corresponds to a different face embedding model. In each grid, the upper triangular cell(s) are colored according to each model's baseline geodesic distance between random sets of faces belonging to different individuals. The diagonal of each inner grid depicts the distance between subspaces of the same covariate value (e.g. Female vs. Female), but different individuals. The lower triangular cell(s) depict the distance between subspaces of **different** covariate values (e.g. Female vs. Male). The closer a cell is to the bottom left corner, the more semantically distinct the subspaces are. Each cell is the result of 7 random samples, with the standard deviation annotated in parentheses below the mean. Each sample involves comparing subspaces consisting of approximately 2,000 embeddings (except those involving the 0-19 age group, which includes only 1,170 embeddings). (Nav table)

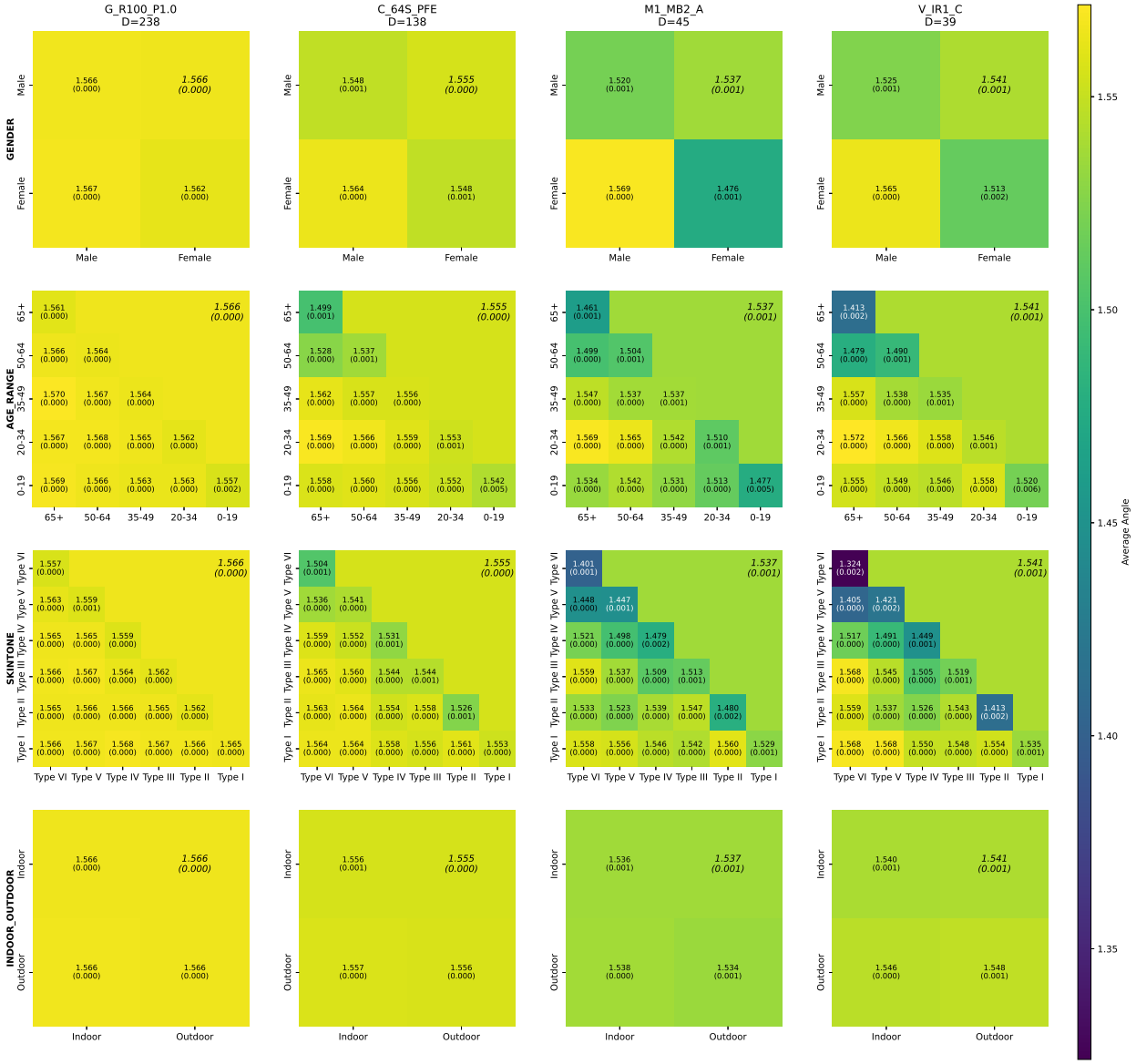


Figure 5.4: Pairwise angles express some differences, though less consistently and clearly than geodesic distances (best viewed in color, magnified). Angles are calculated according to the method described in Subsection 5.1.4, without any normalization or standardization. This figure is organized in the same manner as Figure 5.3, as a grid of grids, with each colored cell corresponding to the mean pairwise angle between two groups of IJB-C face embeddings. (Nav table)

spread of pairwise distances also seems to be dependent on the embedding dimensionality, perhaps due to the "curse of dimensionality." Comparing rows, female faces, those with an age over 65, or those with darkest skin tone are consistently closer to faces of the same demographic than random baseline. This is in contrast to geodesic distance measures, which tend to be bounded below by values near the random baseline.

Looking at the annotated numeric values in either set of heatmaps, one can continue to find interesting trends hidden by near-imperceptible color differences. In the interest of comparing results in a model-agnostic fashion using this color map style, a normalized iteration of both figures is also presented. These figures are in the same format, illustrating the same values, with two minimal changes. First, each value is divided by the random baseline similarity, with the assumption that the separation of subspaces of random faces is a reasonable "yardstick" for a given model's embeddings. Next, a diverging color map is used, centered around the each model's scaled baseline distance, i.e. 1.0. This second iteration is presented in Figures 5.5 and 5.6 (as well as for the other 6 models in Figures C.3 and C.4 in Appendix C).

In both figures, the trends observed previously are emphasized. Specifically, geodesic distances tend to increase with demographic distinction (i.e. as cells are further from each grid's diagonal). What's more, this format highlights sub-baseline geodesic distances for faces of the same gender, similar to the sub-baseline pairwise angles found for faces aged 65+ or having skin tone Type VI. Though the two measures don't necessarily emphasize the same groups, sub-baseline distances may be of interest for bias research. This intuition requires further confirmation, however, as it is not yet understood whether the ideal bias-mitigation strategy is to remove demographic information (as in [37–39]), or to compensate for groups which are more difficult to model (as suggested by [36, 67, 70]).

Though proportional measures emphasize the trends observed earlier, they do not produce the same results for each model. This is not necessarily expected, as the relative distances between subspaces may not scale linearly with variance proportion. Worse, two models which are equal in their dimension-wise distribution of variance and produce equal baselines may still be able to

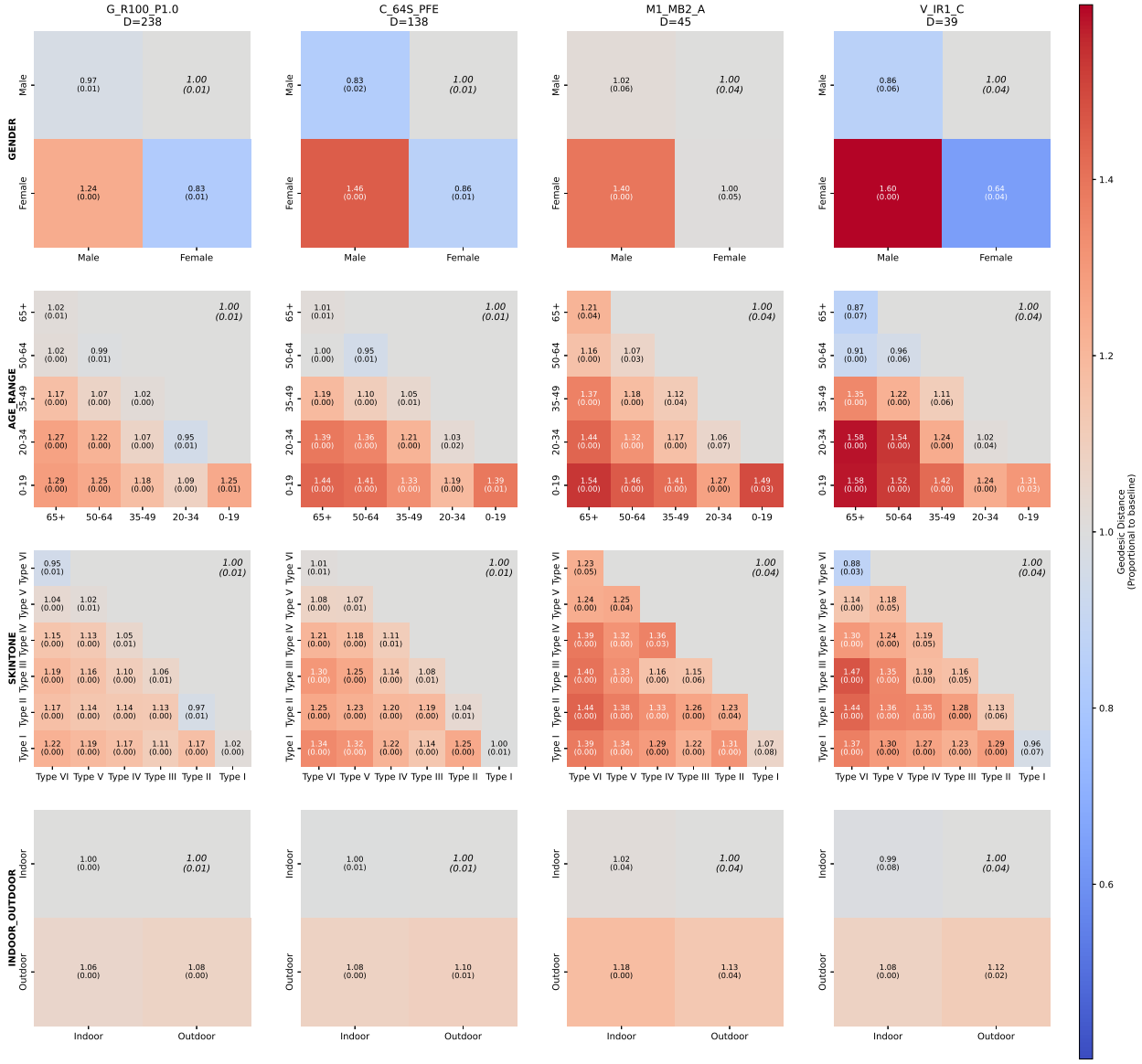


Figure 5.5: Proportional geodesic distances highlight consistency between models (best viewed in color, magnified). The data and format of this figure are the same as Figure 5.3, except each value has been divided by its model's random baseline geodesic distance. This means each data point now represents the proportional change from baseline, with 1.0 representing no change. The color map is centered such that 1.0 corresponds to gray, and values above or below correspond to red and blue, respectively. Each cell is again annotated, here using the proportional distance and proportional standard deviation. (Nav table)

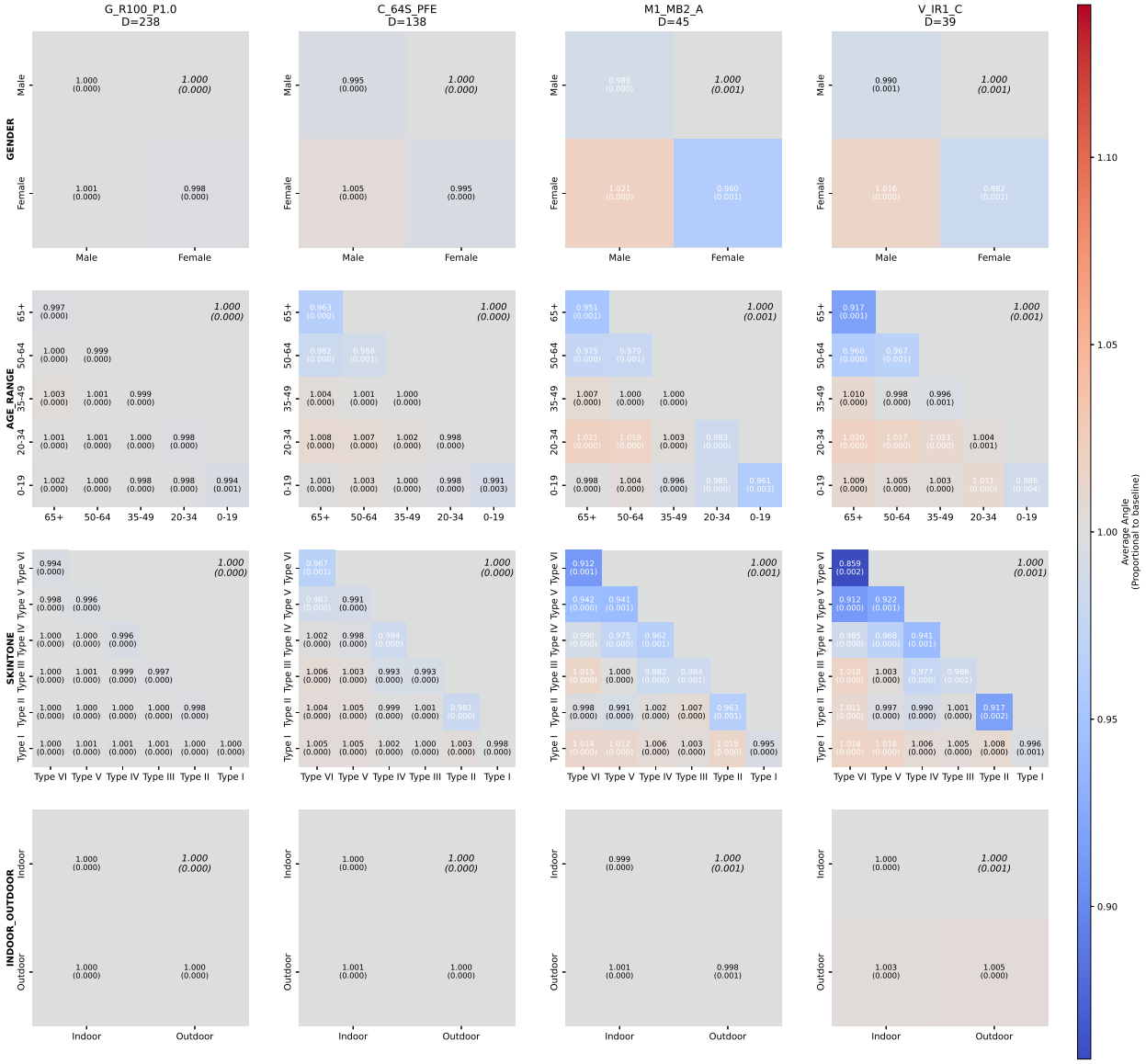


Figure 5.6: Proportional pairwise angles highlight inconsistency between models (best viewed in color, magnified). The data and format of this figure are the same as Figure 5.4, except each value has been divided by its model's random baseline average pairwise angle. Scaling pairwise angles (linearly) does not produce the same cross-model consistency as scaling geodesic distances. In other words, the scale of pairwise angles seems to change at a different rate than the scale of geodesic distances with respect to dimensionality. (Nav table)

produce different embedding structures. In other words, while the results of the previous chapter support the existence of canonical face embeddings, determining which embeddings are nearest to canonical structure remains a challenge. Therefore, determining an appropriate strategy for normalization is difficult, as it relies on assumptions regarding which embedding space details are meaningful or not. Further normalization efforts[12] are thus outside the scope of this work.

5.2.3 Principal angle curves

Recall that each of the colored cells of the previous figures represents the geodesic distance on a Grassmann manifold between two points. The geodesic distance is calculated using the principal angles between those subspaces which are represented as points on a Grassmann. In the interest of illustrating results with minimal assumptions, the principal angles used to calculate geodesic distances are also presented. In Figure 5.7, selected principal angle curves are displayed in a similar format to previous "heatmaps," with each figure column corresponding to a different embedding model, and each row corresponding to a different covariate.

Each line is an "expanded" view of the geodesic distances in the figures presented previously. The **dotted** line depicts the principal angles between two sets of faces which are randomly selected, but contain no identities in common–an expanded view of the previous baselines. Similarly, the **dashed** line depicts principal angles between two subspaces spanned by faces of the same covariate value (e.g. Female vs. Female), corresponding the the first and last values along the diagonal of each grid in the previous figures. Finally, **solid** red line depicts the principal angles between subspaces spanned by faces which are most semantically distinct[13] within a given covariate (e.g. darkest skin tone vs. lightest). Due to the limited space in this format, each plot contains only these 4 lines. Each line is the average of 7 repeated random samples within these conditions, with the area spanned by 1 standard deviation shaded in gray. For the remaining 6 models, see Figure C.5 in Appendix C.

[12]To be clear, variance-proportional dimension reduction and baseline-proportional distance scaling certainly rely upon assumptions regarding what information is meaningful.

[13]The 0-19 age range is excluded to emphasize same-size embedding subsets.

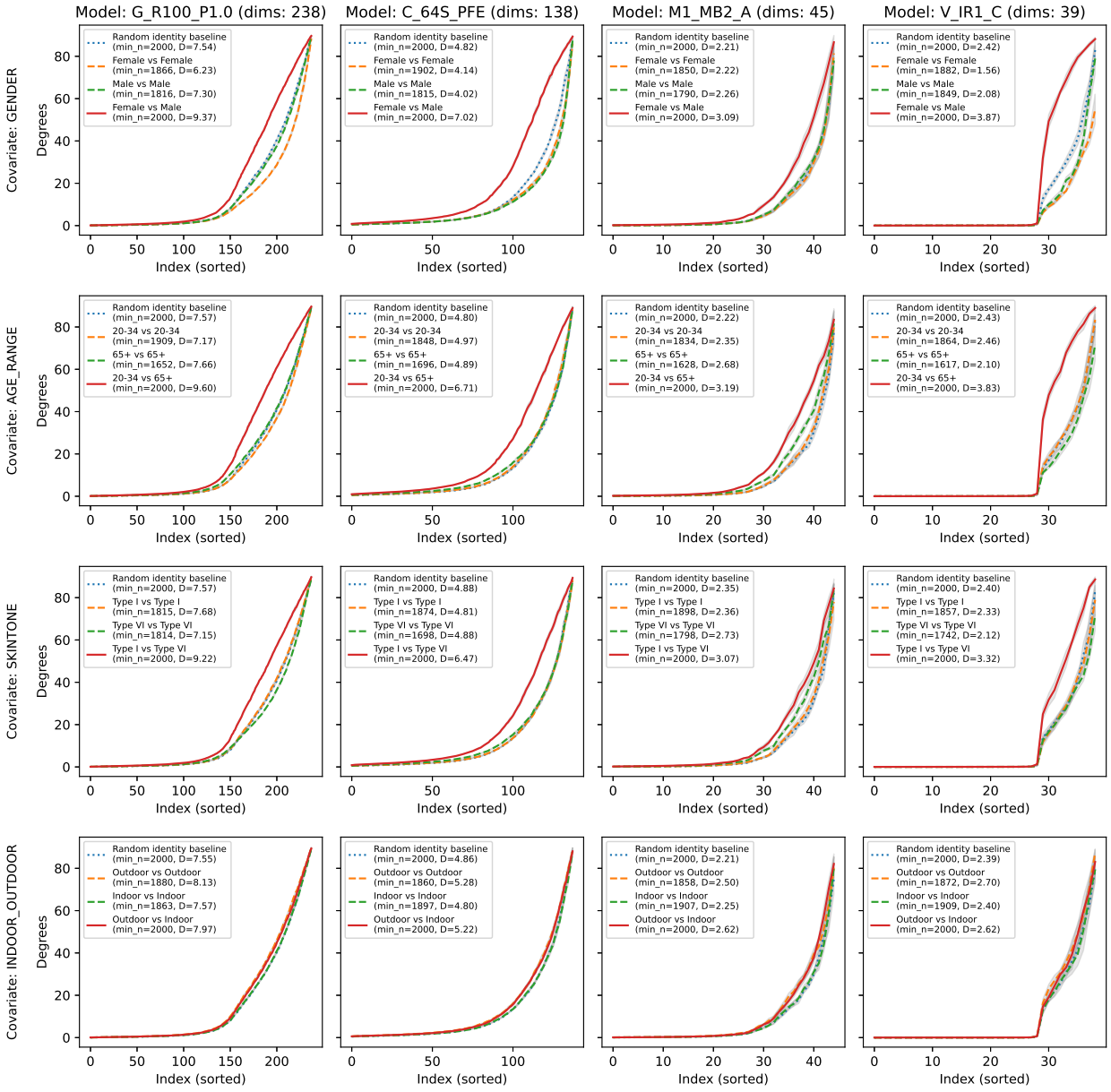


Figure 5.7: Principal angles show distinction between face embedding subspaces spanned by different covariate groups. These four selected models use different training datasets, architectures, and loss functions. Dotted lines show the principal angles between randomly selected faces of different identity. Dashed lines show the same, except all faces belong to a specific covariate group. Solid lines show the principal angles between faces of different covariate groups. Many more combinations exist than are depicted here, with the emphasis instead on those comparisons which are most distinct (i.e. the corners of the previous colored grid figures). Angles are converted to degrees here, where the geodesic distance and pairwise angles were calculated using radians. Each line is the mean of 7 measurements between repeated random samples, with the gray shaded area representing 1 standard deviation. (Nav table)

At a glance, this figure illustrates a similar trend as the previous ones: face embeddings subspaces of distinct demographic groups tend to be further from those of the same demographic group, or a random baseline. The shape of the principal angle curves, however, offers finer details. Each principal angle curve starts near 0 degrees, remains relatively flat, then rises to end at 90 degrees. This suggests much of embedding space is still spanned by any sufficiently large set of faces, even after dimension reduction. Conversely, this means there are some subspaces which are quite sensitive to face identity and/or demographics. Interestingly, one model "V_IR1_C" (last column) produces principal angle curves with a sharp jump, even when comparing subspaces spanned by random faces, where the other models produce smooth curves which diverge sooner. For another example of this behavior on the other extreme of dimensionality, see model "G_R100_P0.1" in Figure C.5.

Looking at subspace separation angle by angle, it is not immediately clear which angles (i.e. which dimensions) are most important for the modeling task. To better understand which principal angles are within the most expressive dimensions, the principal angle curves for the gender covariate are displayed at different levels of variance-proportional dimension reduction in Figure 5.8. The figure is in the same format as the previous principal angle curves, except each row corresponds to a different proportion of variance kept when performing dimension reduction, and all plots concern the gender covariate.

Scanning row by row, the number of near-zero principal angles is consistently reduced as the amount of variance retained is reduced. This suggests that the dimensions of least variance are also those which lead to the smallest principal angles. This is supported by the earlier notion that dimension reduction is necessary to remove pixel noise, and that dimensions where noise is present will not be expressive of subspace separation.

5.3 Conclusion

What's most interesting about this work is that it has not been done before, and the results were not necessarily easy to predict. This work applies a method for describing groups of embeddings as

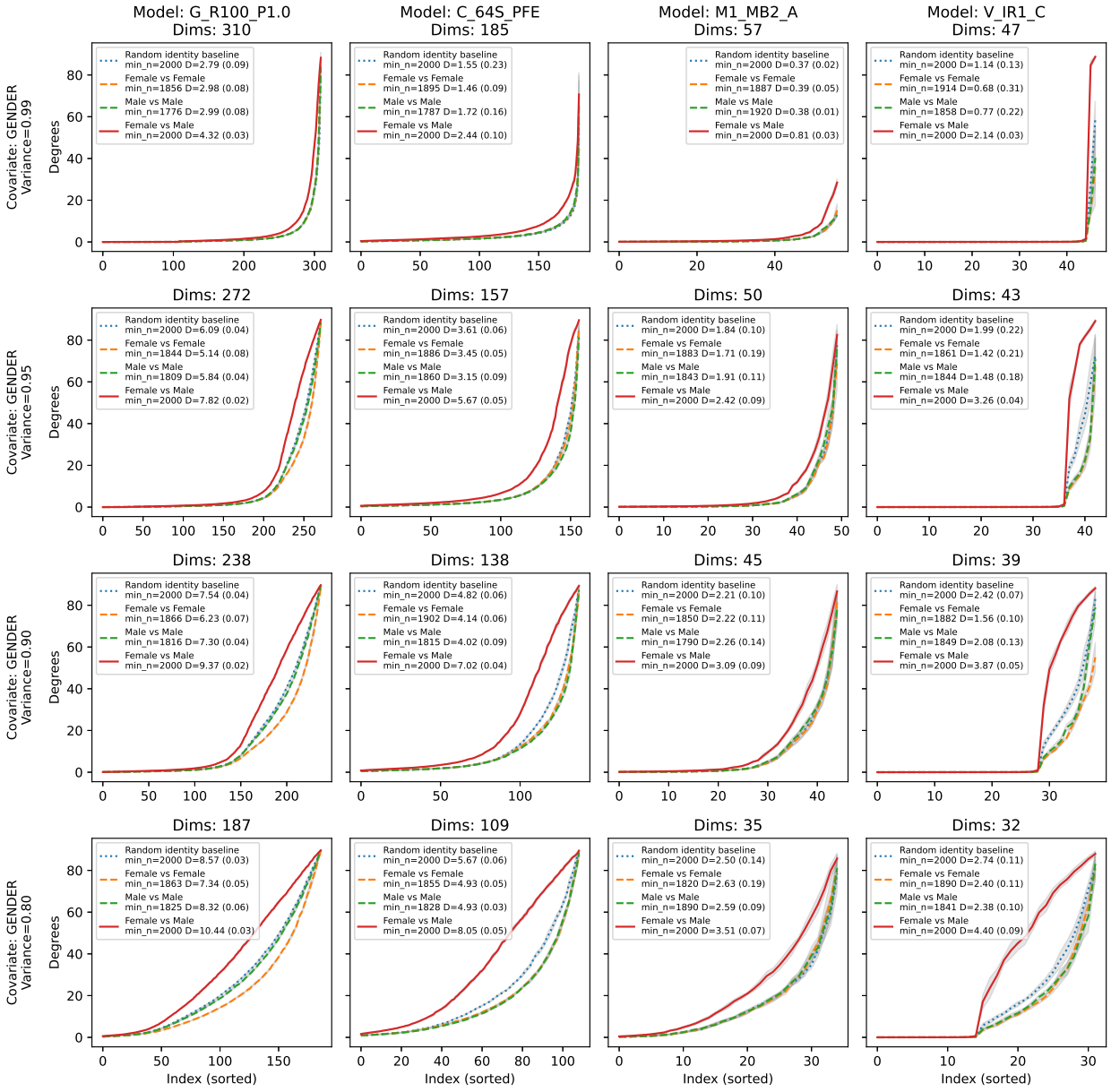


Figure 5.8: Low-variance dimensions lead to more near-zero principal angles, confirming that dimensions which are most expressive of different subspaces are also those containing the greatest proportion of variance. The format of this figure is the same as Figure 5.7, except each row shows a different proportion of variance removed during dimension reduction, instead of a different covariate. This figure only includes comparisons within the gender covariate. (Nav table)

groups, and comparing them as groups (i.e. subspaces). When applied to face embeddings, faces of different demographics are distinguished, even when pairwise distances are inconsistent. Face embeddings are structural in nature and this method confirms that demographic-specific subspaces are descriptive of that structure. Furthermore, the consistency in relative geodesic distances among different models further supports the implications of the previous chapter–that there is fundamental similarity in the structure of learned face representations.

Though the results of this work are extensive, it is difficult to make conclusions that are not subjective. Still, the results of this chapter suggest that demographic information is learned in a consistent way, as evidenced by each model's tendency to further separate face embedding subspaces of further skin tone, age, and gender[14]. Regardless, there are many ways in which this work could be extended to better understand the relationship between bias and model design.

First and foremost, more measures of bias need to be included to validate how bias is related to subspace angles. Though subspace angles are expressive of demographic differences, it is not clear how subspace angles relate to the rate at which models produce biased outcomes. A performance-based bias measure would be sufficient, as is common in other bias analyses.

Second, though the models chosen here cover a wide array of designs, bias-mitigated models would be of special interest for future works. Some bias-mitigation efforts rely on demographic obfuscation, which would likely lead to inexpressive demographic subspace angles (e.g. [37]). In contrast, others not necessarily studying bias rely on effective modeling of auxiliary information in order to make predictions which are well-informed of factors which confound the model task. In one work, an auxiliary quality estimator is used to make predictions that are sensitive to uncertainty [19], and in another the pose of the head in the image is linked to the embedding space, facilitating superior pose-aware predictions [113]. It would be interesting to investigate whether subspace angles also can provide some auxiliary information for producing better-informed models, especially for the purposes of controlling bias.

[14]As gender is annotated using a binary variable, "further" can be replaced with "different."

Surprisingly, another major source of limitation is the dataset. Though IJB-C includes other covariates (pose, occlusion, facial hair), the number of samples within each prevents their study using subspace angles. Even within the covariates considered, only 1170 images are available within the 0-19 age group, where the next group (20-34) contains over 36k images. Further, this dataset is not constructed such that each covariate group is roughly equally distributed among other covariates (e.g. female faces tend to be younger than male faces). There are many other demographic-annotated datasets for consideration when studying bias, such as MORPH [65] and RFW [39]. For non-demographic information such as occlusion, color standardization, or head pose, systematic image augmentation is also an option, as in [113].

Finally, as mentioned previously, significant work is necessary to understand how to fairly and objectively compare measures between different models. This is a difficult task, since it requires a solid understanding of the nature of embedding structure as it relates to ambient dimensionality, the distribution of variance across dimensions, the structural feature being measured, and probably far more. In that sense, efforts like this one are in their infancy.

Regardless, these experiments are new and the results are encouraging. Geodesic distances between subspaces spanned by face embeddings representing different ages, genders, or skin tones are larger than a random baseline, and are larger-still as differences in age and skin tone increase. These results are consistent for a wide array of face model designs, supporting the previous results that face models converge to a common learned representation. As the shared structure of learned representations is better understood, better bias-mitigated models can be designed.

Chapter 6

Conclusion

A key takeaway from this work is that it is hard to make good assumptions about the learned representations of deep convolutional neural nets. Deep learning research can be challenging for any scientist to navigate, since the significance of many advances is measured chiefly by their respective improvements to benchmark dataset performance. The breakout AlexNet CNN trained for ImageNet image classification achieved 63.3% accuracy [40], whereas a recent state-of-the art CNN achieves 90% (EfficientNet trained using Meta Pseudo Labels [114]). The way in which these models are trained has certainly changed, but a reasonable assumption regarding the advances made in the last decade of research is that architecture is key to building better CNNs. Yet, the results presented in this thesis suggest that the structure of learned representations is mostly unchanged by differences in architecture.

The earliest efforts along these lines were covered in my Master's thesis. They reveal that there is a yet-unknown set of architectures which consist of distinct combinations of operations, yet do not each learn to extract fundamentally distinct information. This was determined by testing for the interchangeability between the last-layer embeddings of 10 different ImageNet-trained CNN models. In order to successfully swap representations between networks, only a linear mapping was necessary in order to preserve a high level of performance between all 90 heterogeneous pairs. A link between these models threw those results into question, however. Essentially, each model was learning to map the same images to the same set of labels, and this created a correspondence in the precursor to label predictions–the embeddings studied for interchangeability. This vacuous sense of representational similarity challenged the idea that CNNs are learning the same thing **despite** their differences, instead highlighting the shared set of training targets (ground truth labels) as the primary driver of last-layer representational similarity.

In response, a similar set of experiments was conducted using facial recognition models. Beyond architectural differences, these models are not necessarily trained on the same dataset, nor

using the exact same loss. Because training datasets may differ, there is no explicitly-similar architectural component between such models as in those studied prior. Since these models are evaluated by direct comparison of their embeddings (rather than classifier scores as in ImageNet), it is even easier to measure the success of an effort to convert between embedding spaces.

Between these models with arguably fewer similarities, the same fundamental representational similarity phenomenon was observed. By use of a linear mapping, the face embeddings of one model could be converted into those of another model while retaining high face verification performance. Even when restricting that mapping to only perform rotations, cross-model face verification could be performed using the embeddings of models which were trained on a different dataset, using a different architecture, and a (perhaps marginally) different loss target. If only rotation is required to convert the bulk of embeddings between models, there is a strong sense in which those models are learning the same thing.

Architecture is clearly not the only driving force behind the structure of learned representations. Within the architectures studied, architecture seems to not even be the primary driver of learned representation. This is not to say that architecture is meaningless, only that there are likely other factors which have a greater impact on the structure of learned representations. One potential explanation for this phenomenon is that models trained to perform the same task are targeting the same information, thus arranging information in a highly similar way. One way to extend this work would be to consider the role of task in shaping representations, but such a multi-task effort is beyond the scope of this thesis and left for future work (see Taskonomy [108] for relevant results regarding cross-task relationships). Besides looking for the mechanisms which impact this similarity between tasks, another natural step is to consider what factors impact representational similarity within task. Indeed, in each of the experiments discussed so far, some models' representations were more easily interchanged than others. Fortunately, within face recognition modeling there are those working to understand the structure of embedding space, especially as it varies due to model design. Therefore, the final experiments in this thesis regard the content of face embedding similarity for another worthy cause: bias in face recognition.

Face recognition bias is a compelling and worthwhile topic on its own, but this research also targets how the structure necessary to separate faces is dependent on how the model was trained, and even the architecture. Since this thesis is concerned with structural similarity, groups of faces are represented as subspaces, and compared as subspaces. By comparing groups as subspaces, this method is grounded in the shared representational structure revealed prior. By comparing how the faces of different people (and different imaging conditions) are represented in the structure of embedding space, and how those structures vary with the designs of the model, a new perspective on face recognition bias is offered. Specifically, the results of these last experiments support the notion that face embeddings are similarly structured, and that demographics are members of that structure. Face models are known to be biased, and (recently) known to produce embeddings which adhere to a common structure, so measuring bias in that structure offers an exciting new perspective.

Paradoxically, the experiments which motivate the latter experiments suggest that differences in CNN architecture do little to change the structure of the learned representation. Yet, the goals of the latter experiments are to use representational structure to separate and qualify model designs, including architecture. In truth, the relationships between a neural network's learned representation, its architecture, and its modeling task are complex, requiring further effort. Regardless, the work here offers key insights on this topic: that architecture is not the primary driver of learned representations, and that understanding the structure of learned representations is possible given the right tools.

Bibliography

[1] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. *arXiv preprint arXiv:1905.00414*, pages 3519–3529, 2019.

[2] Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. In *Advances in Neural Information Processing Systems*, pages 6076–6085, 2017.

[3] Ari Morcos, Maithra Raghu, and Samy Bengio. Insights on representational similarity in neural networks with canonical correlation. In *Advances in Neural Information Processing Systems*, pages 5727–5736, 2018.

[4] Yunzhen Feng, Runtian Zhai, Di He, Liwei Wang, and Bin Dong. Transferred discrepancy: Quantifying the difference between representations. *arXiv preprint arXiv:2007.12446*, 2020.

[5] Yixuan Li, Jason Yosinski, Jeff Clune, Hod Lipson, and John E Hopcroft. Convergent learning: Do different neural networks learn the same representations? In *FE@ NIPS*, pages 196–212, 2015.

[6] Liwei Wang, Lunjia Hu, Jiayuan Gu, Yue Wu, Zhiqiang Hu, Kun He, and John Hopcroft. Towards understanding learning representations: To what extent do different neural networks learn the same representation. *arXiv preprint arXiv:1810.11750*, 2018.

[7] Karel Lenc and Andrea Vedaldi. Understanding image representations by measuring their equivariance and equivalence. *International Journal of Computer Vision*, 127(5):456–476, May 2019.

[8] Katherine L Hermann and Andrew K Lampinen. What shapes feature representations? exploring datasets, architectures, and training. *arXiv preprint arXiv:2006.12433*, 2020.

[9] Michael Gygli, Jasper Uijlings, and Vittorio Ferrari. Towards reusable network components by learning compatible representations, 2020.

[10] Zihang Dai, Hanxiao Liu, Quoc V Le, and Mingxing Tan. Coatnet: Marrying convolution and attention for all data sizes. *arXiv preprint arXiv:2106.04803*, 2021.

[11] Mengde Xu, Zheng Zhang, Han Hu, Jianfeng Wang, Lijuan Wang, Fangyun Wei, Xiang Bai, and Zicheng Liu. End-to-end semi-supervised object detection with soft teacher. *arXiv preprint arXiv:2106.09018*, 2021.

[12] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

[13] Adrian Bulat, Jean Kossaifi, Georgios Tzimiropoulos, and Maja Pantic. Toward fast and accurate human pose estimation via soft-gated skip connections. In *2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020)*, pages 8–15. IEEE, 2020.

[14] Zheng Zhu, Guan Huang, Jiankang Deng, Yun Ye, Junjie Huang, Xinze Chen, Jiagang Zhu, Tian Yang, Jiwen Lu, Dalong Du, et al. Webface260m: A benchmark unveiling the power of million-scale deep face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10492–10502, 2021.

[15] Inc. Amazon Web Services. Artificial intelligence services, 2021.

[16] Google LLC. Ai & machine learning products | google cloud, 2021.

[17] Microsoft Corporation. Cognitive services—apis for ai solutions | microsoft azure, 2021.

[18] IBM Corporation. Ibm watson products, 2021.

[19] Yichun Shi and Anil K Jain. Probabilistic face embeddings. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6902–6911, 2019.

[20] Xiang An, Xuhan Zhu, Yang Xiao, Lan Wu, Ming Zhang, Yuan Gao, Bin Qin, Debing Zhang, and Fu Ying. Partial fc: Training 10 million identities on a single machine. In *Arxiv 2010.05222*, 2020.

[21] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *arXiv preprint arXiv:2104.14294*, 2021.

[22] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[23] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[24] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[25] 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*, pages 770–778, 2016.

[26] Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy. Progressive neural architecture search. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 19–34, 2018.

[27] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[28] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[29] David McNeely-White, J. Beveridge, and Bruce Draper. Inception and resnet features are (almost) equivalent. *Cognitive Systems Research*, 59, 10 2019.

[30] David G McNeely-White. Same data, same features: Modern imagenet-trained convolutional neural networks learn the same thing. Master's thesis, Colorado State University, 2020.

[31] Patrick Grother, Mei Ngan, and Kayee Hanaoka. *Face Recognition Vendor Test (FVRT): Part 3, Demographic Effects*. National Institute of Standards and Technology, 2019.

[32] Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on fairness, accountability and transparency*, pages 77–91. PMLR, 2018.

[33] Boyu Lu, Jun-Cheng Chen, Carlos D Castillo, and Rama Chellappa. An experimental evaluation of covariates effects on unconstrained face verification. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, 1(1):42–55, 2019.

[34] John J Howard, Yevgeniy B Sirotin, and Arun R Vemury. The effect of broad and specific demographic homogeneity on the imposter distributions and false match rates in face recognition algorithm performance. In *2019 IEEE 10th International Conference on Biometrics Theory, Applications and Systems (BTAS)*, pages 1–8. IEEE, 2019.

[35] Pawel Drozdowski, Christian Rathgeb, Antitza Dantcheva, Naser Damer, and Christoph Busch. Demographic bias in biometrics: A survey on an emerging challenge. *IEEE Transactions on Technology and Society*, 1(2):89–103, 2020.

[36] Vítor Albiero, Kai Zhang, and Kevin W Bowyer. How does gender balance in training data affect face recognition accuracy? In *2020 IEEE International Joint Conference on Biometrics (IJCB)*, pages 1–10. IEEE, 2020.

[37] Aythami Morales, Julian Fierrez, Ruben Vera-Rodriguez, and Ruben Tolosana. Sensitivenets: Learning agnostic representations with application to face images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(6):2158–2164, 2020.

[38] Sixue Gong, Xiaoming Liu, and Anil K Jain. Debface: De-biasing face recognition. *arXiv preprint arXiv:1911.08080*, 2019.

[39] Mei Wang, Weihong Deng, Jiani Hu, Xunqiang Tao, and Yaohai Huang. Racial faces in the wild: Reducing racial bias by information maximization adaptation network. In *Proceedings of the ieee/cvf international conference on computer vision*, pages 692–702, 2019.

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

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

[42] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

[43] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[44] David McNeely-White, Ben Sattelberg, Nathaniel Blanchard, and Ross Beveridge. Exploring the interchangeability of CNN embedding spaces. *arXiv preprint arXiv:2010.02323v4*, 2020.

[45] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[46] Quanshi Zhang, Ying Nian Wu, and Song-Chun Zhu. Interpretable convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8827–8836, 2018.

[47] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

[48] Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Algorithms for learning kernels based on centered alignment. *The Journal of Machine Learning Research*, 13(1):795–828, 2012.

[49] Paul Robert and Yves Escoufier. A unifying tool for linear multivariate statistical methods: The rv-coefficient. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 25(3):257–265, 1976.

[50] Ledyard R Tucker. A method for synthesis of factor analysis studies. Technical report, Educational Testing Service Princeton Nj, 1951.

[51] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[52] Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? *arXiv preprint arXiv:1805.08974*, 2018.

[53] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop*, 2011.

[54] Katherine Hermann, Ting Chen, and Simon Kornblith. The origins and prevalence of texture bias in convolutional neural networks. *Advances in Neural Information Processing Systems*, 33, 2020.

[55] Marvin Minsky and Seymour A Papert. *Perceptrons: An introduction to computational geometry*. MIT press, 2017.

[56] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011.

[57] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[58] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[59] Jacqueline G Cavazos, P Jonathon Phillips, Carlos D Castillo, and Alice J O'Toole. Accuracy comparison across face recognition algorithms: Where are we on measuring race bias? *IEEE transactions on biometrics, behavior, and identity science*, 3(1):101–111, 2020.

[60] Christian A Meissner and John C Brigham. Thirty years of investigating the own-race bias in memory for faces: A meta-analytic review. *Psychology, Public Policy, and Law*, 7(1):3, 2001.

[61] Michael McLaughlin and Daniel Castro. The critics were wrong: Nist data shows the best facial recognition algorithms are neither racist nor sexist. Technical report, Information Technology and Innovation Foundation, 2020.

[62] Cameron Whitelam, Emma Taborsky, Austin Blanton, Brianna Maze, Jocelyn Adams, Tim Miller, Nathan Kalka, Anil K Jain, James A Duncan, Kristen Allen, et al. Iarpa janus benchmark-b face dataset. In *proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 90–98, 2017.

[63] Brianna Maze, Jocelyn Adams, James A Duncan, Nathan Kalka, Tim Miller, Charles Otto, Anil K Jain, W Tyler Niggel, Janet Anderson, Jordan Cheney, et al. Iarpa janus benchmark-c: Face dataset and protocol. In *2018 International Conference on Biometrics (ICB)*, pages 158–165. IEEE, 2018.

[64] Kushal Vangara, Michael C King, Vitor Albiero, Kevin Bowyer, et al. Characterizing the variability in face recognition accuracy relative to race. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[65] Karl Ricanek and Tamirat Tesafaye. Morph: A longitudinal image database of normal adult age-progression. In *7th international conference on automatic face and gesture recognition (FGR06)*, pages 341–345. IEEE, 2006.

[66] Information technology — Biometric data interchange formats — Part 5: Face image data, November 2011.

[67] Vitor Albiero, Krishnapriya KS, Kushal Vangara, Kai Zhang, Michael C King, and Kevin W Bowyer. Analysis of gender inequality in face recognition accuracy. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision Workshops*, pages 81–89, 2020.

[68] P Jonathon Phillips, Patrick J Flynn, Todd Scruggs, Kevin W Bowyer, Jin Chang, Kevin Hoffman, Joe Marques, Jaesik Min, and William Worek. Overview of the face recognition

grand challenge. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 947–954. IEEE, 2005.

[69] Zhangyang Xiong, Zhongyuan Wang, Changqing Du, Rong Zhu, Jing Xiao, and Tao Lu. An asian face dataset and how race influences face recognition. In *Pacific Rim Conference on Multimedia*, pages 372–383. Springer, 2018.

[70] Vidya Muthukumar, Tejaswini Pedapati, Nalini Ratha, Prasanna Sattigeri, Chai-Wah Wu, Brian Kingsbury, Abhishek Kumar, Samuel Thomas, Aleksandra Mojsilovic, and Kush R Varshney. Understanding unequal gender classification accuracy from face images. *arXiv preprint arXiv:1812.00099*, 2018.

[71] Joseph P Robinson, Gennady Livitz, Yann Henon, Can Qin, Yun Fu, and Samson Timoner. Face recognition: too bias, or not too bias? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–1, 2020.

[72] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[73] Qiong Cao, Li Shen, Weidi Xie, Omkar M Parkhi, and Andrew Zisserman. Vggface2: A dataset for recognising faces across pose and age. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, pages 67–74. IEEE, 2018.

[74] Mei Wang and Weihong Deng. Mitigating bias in face recognition using skewness-aware reinforcement learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9322–9331, 2020.

[75] Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In *European conference on computer vision*, pages 87–102. Springer, 2016.

[76] Feng Wang, Xiang Xiang, Jian Cheng, and Alan Loddon Yuille. Normface: L2 hypersphere embedding for face verification. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 1041–1049, 2017.

[77] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5265–5274, 2018.

[78] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2019.

[79] Brendan F Klare, Ben Klein, Emma Taborsky, Austin Blanton, Jordan Cheney, Kristen Allen, Patrick Grother, Alan Mah, and Anil K Jain. Pushing the frontiers of unconstrained face detection and recognition: Iarpa janus benchmark a. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1931–1939, 2015.

[80] P Jonathon Phillips, J Ross Beveridge, Bruce A Draper, Geof Givens, Alice J O'Toole, David Bolme, Joseph Dunlop, Yui Man Lui, Hassan Sahibzada, and Samuel Weimer. The good, the bad, and the ugly face challenge problem. *Image and Vision Computing*, 30(3):177–185, 2012.

[81] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pages 3730–3738, 2015.

[82] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2016.

[83] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[84] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8697–8710, 2018.

[85] Richard Bellman. A markovian decision process. *Journal of mathematics and mechanics*, pages 679–684, 1957.

[86] Harold Hotelling. Relations between two sets of variates. In *Breakthroughs in statistics*, pages 162–190. Springer, 1992.

[87] Geoffrey Roeder, Luke Metz, and Diederik P Kingma. On linear identifiability of learned representations. *arXiv preprint arXiv:2007.00810*, 2020.

[88] Andrew M Saxe, Pang Wei Koh, Zhenghao Chen, Maneesh Bhand, Bipin Suresh, and Andrew Y Ng. On random weights and unsupervised feature learning. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pages 1089–1096, 2011.

[89] Nicolas Pinto, David Doukhan, James J DiCarlo, and David D Cox. A high-throughput screening approach to discovering good forms of biologically inspired visual representation. *PLoS Comput Biol*, 5(11):e1000579, 2009.

[90] Kevin Jarrett, Koray Kavukcuoglu, Marc'Aurelio Ranzato, and Yann LeCun. What is the best multi-stage architecture for object recognition? In *2009 IEEE 12th international conference on computer vision*, pages 2146–2153. IEEE, 2009.

[91] Raja Giryes, Guillermo Sapiro, and Alex M Bronstein. Deep neural networks with random gaussian weights: A universal classification strategy? *IEEE Transactions on Signal Processing*, 64(13):3444–3457, 2016.

[92] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE signal processing letters*, 23(10):1499–1503, 2016.

[93] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[94] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 212–220, 2017.

[95] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. Learning face representation from scratch. *arXiv preprint arXiv:1411.7923*, 2014.

[96] Stylianos Moschoglou, Athanasios Papaioannou, Christos Sagonas, Jiankang Deng, Irene Kotsia, and Stefanos Zafeiriou. Agedb: the first manually collected, in-the-wild age database. In *proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 51–59, 2017.

[97] Soumyadip Sengupta, Jun-Cheng Chen, Carlos Castillo, Vishal M Patel, Rama Chellappa, and David W Jacobs. Frontal to profile face verification in the wild. In *2016 IEEE winter conference on applications of computer vision (WACV)*, pages 1–9. IEEE, 2016.

[98] Jia Guo and Jiankang Deng. Insightface: 2d and 3d face analysis project, April 2021.

[99] Kuan-Yu Huang. Arcface unofficial implemented in tensorflow 2.0+ (resnet50, mobilenetv2)., June 2020.

[100] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *European conference on computer vision*, pages 499–515. Springer, 2016.

[101] David Sandberg. Face recognition using tensorflow, April 2018.

[102] Feng Wang, Jian Cheng, Weiyang Liu, and Haijun Liu. Additive margin softmax for face verification. *IEEE Signal Processing Letters*, 25(7):926–930, 2018.

[103] Yichun Shi. Probabilistic face embeddings, August 2019.

[104] Grace Wahba. A least squares estimate of satellite attitude. *SIAM review*, 7(3):409–409, 1965.

[105] Wolfgang Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 32(5):922–923, 1976.

[106] Guangcan Mai, Kai Cao, Pong C Yuen, and Anil K Jain. On the reconstruction of face images from deep face templates. *IEEE transactions on pattern analysis and machine intelligence*, 41(5):1188–1202, 2018.

[107] Sixue Gong, Vishnu Naresh Boddeti, and Anil K Jain. On the intrinsic dimensionality of image representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3987–3996, 2019.

[108] Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3712–3722, 2018.

[109] Amir R Zamir, Alexander Sax, Nikhil Cheerla, Rohan Suri, Zhangjie Cao, Jitendra Malik, and Leonidas J Guibas. Robust learning through cross-task consistency. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11197–11206, 2020.

[110] Yung-Chow Wong. Differential geometry of grassmann manifolds. *Proceedings of the National Academy of Sciences of the United States of America*, 57(3):589, 1967.

[111] Andrew V Knyazev and Merico E Argentati. Principal angles between subspaces in an a-based scalar product: algorithms and perturbation estimates. *SIAM Journal on Scientific Computing*, 23(6):2008–2040, 2002.

[112] Thomas B Fitzpatrick. Soleil et peau. *J Med Esthet*, 2:33–34, 1975.

[113] Xiaolong Yang, Xiaohong Jia, Dihong Gong, Dong-Ming Yan, Zhifeng Li, and Wei Liu. Larnet: Lie algebra residual network for face recognition. In *International Conference on Machine Learning*, pages 11738–11750. PMLR, 2021.

[114] Hieu Pham, Zihang Dai, Qizhe Xie, and Quoc V Le. Meta pseudo labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11557–11568, 2021.

Appendix A

Pilot experiment on LFW

LFW verification consists of generating pairs of embeddings for 6,000 predefined image pairs. Then, accuracy was measured by performing 10-fold cross-validation as specified by the *Unrestricted, labeled outside data* protocol documented in LFW [72]. Accuracy on LFW is highly-saturated by modern CNNs, so four of the worst-performing models were evaluated. For this performance, see the hatched bars of Figure A.1.

Linear mappings were fit using the same method described in Section 4.1.1, and evaluated using the same method described in Section 4.1.2, instead reporting the mean accuracy obtained on each of the ten cross-validation folds. More precisely, one fold was held out for testing while the remaining nine are used to find a linear mapping. Then, the held out fold was used for cross-CNN evaluation. This process was repeated ten times, once for each fold. The mean accuracy obtained on held out folds is reported in Figure A.1.

Sensitivity experiments can also be carried out as in Section 4.2.1, using a random subset of embedding pairs. For each fold, a mapping is fit on a random subset of training fold embeddings, and evaluated on the remaining test fold. The mean accuracy obtained at different sizes of training subsets is reported in Figure A.2.

Though these experiments are carried out on an easier dataset, they suggest that this linear correspondence phenomenon is not necessarily dependent on evaluation method or dataset.

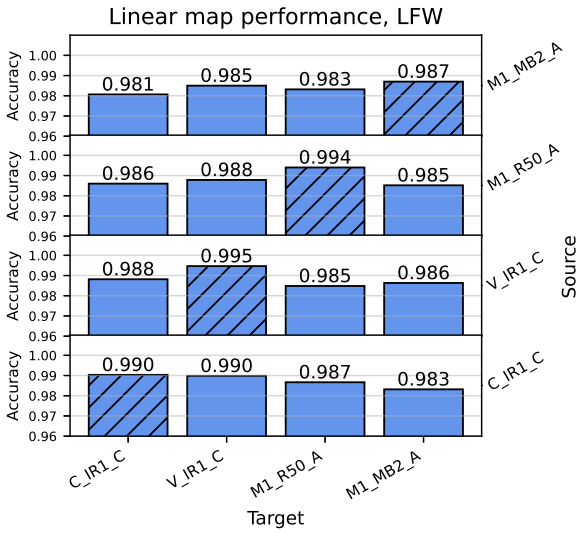


Figure A.1: Linear mappings are sufficient for converting between the features of different CNNs with minimal drop in performance. Cross-hatched bars indicate the performance when evaluating a model against its own embeddings, without mapping.

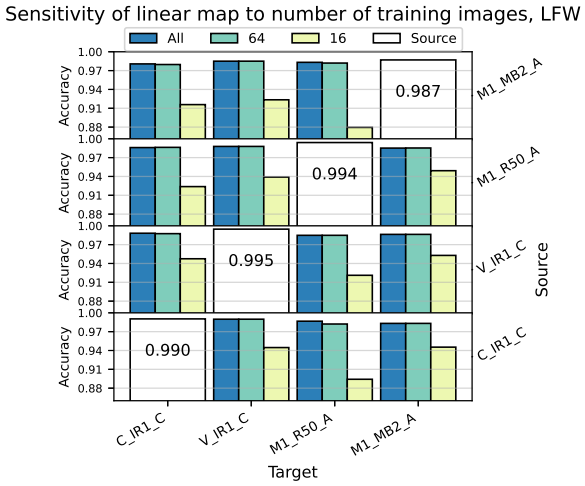


Figure A.2: When using a reduced number of images, linear mappings still convert between the features of different CNNs with high performance.

Appendix B

Extended Face Mapping Results

Figures containing the full results from the cross-CNN IJB-C 1:1 verification experiments are provided using a linear map (Figure B.1), a rotation map (Figure B.2), and a restricted training sample size using a rotation map (Figure B.3).

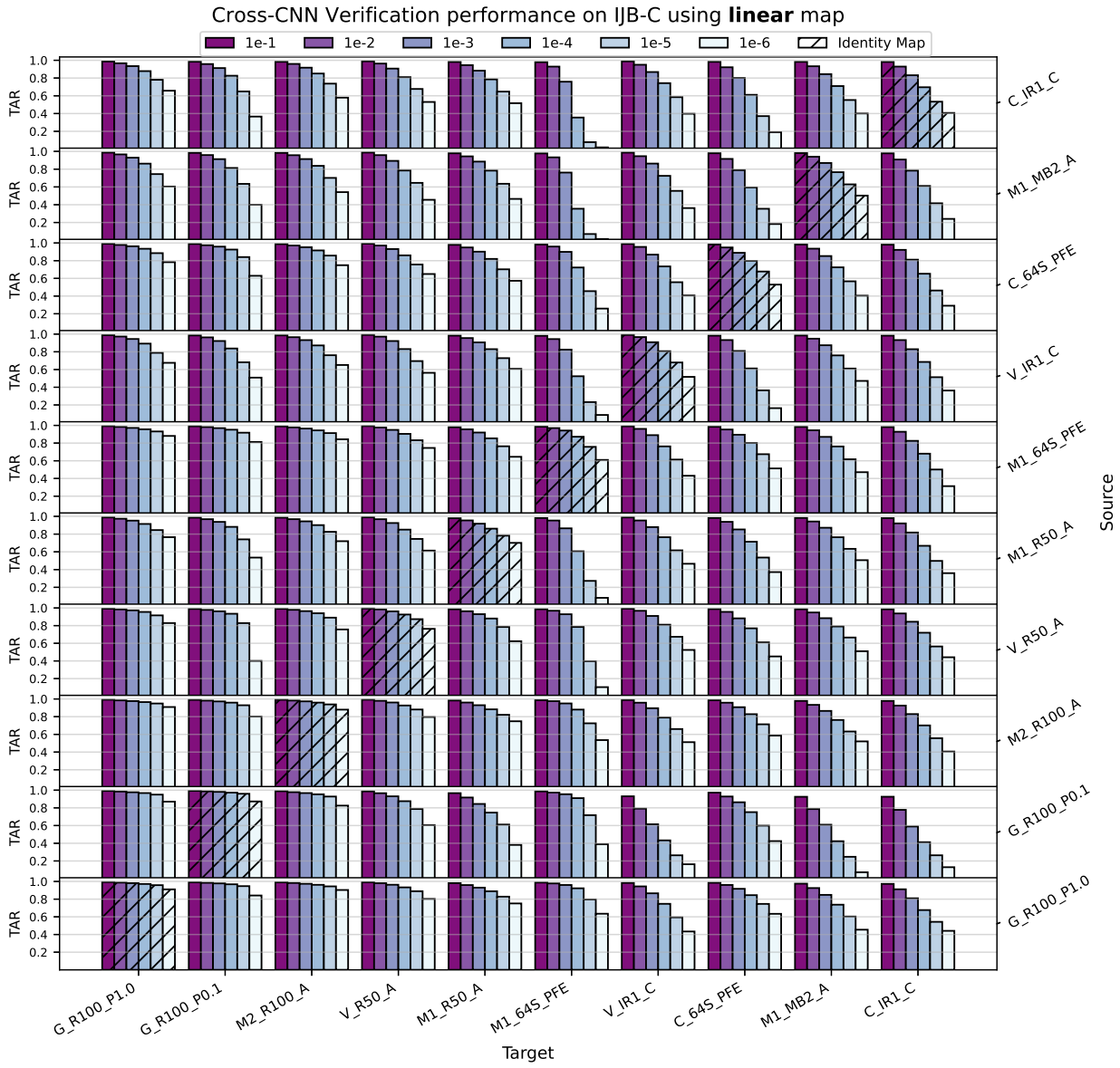


Figure B.1: Full **linear** mapping performance results, in the same format as Figure 4.1.

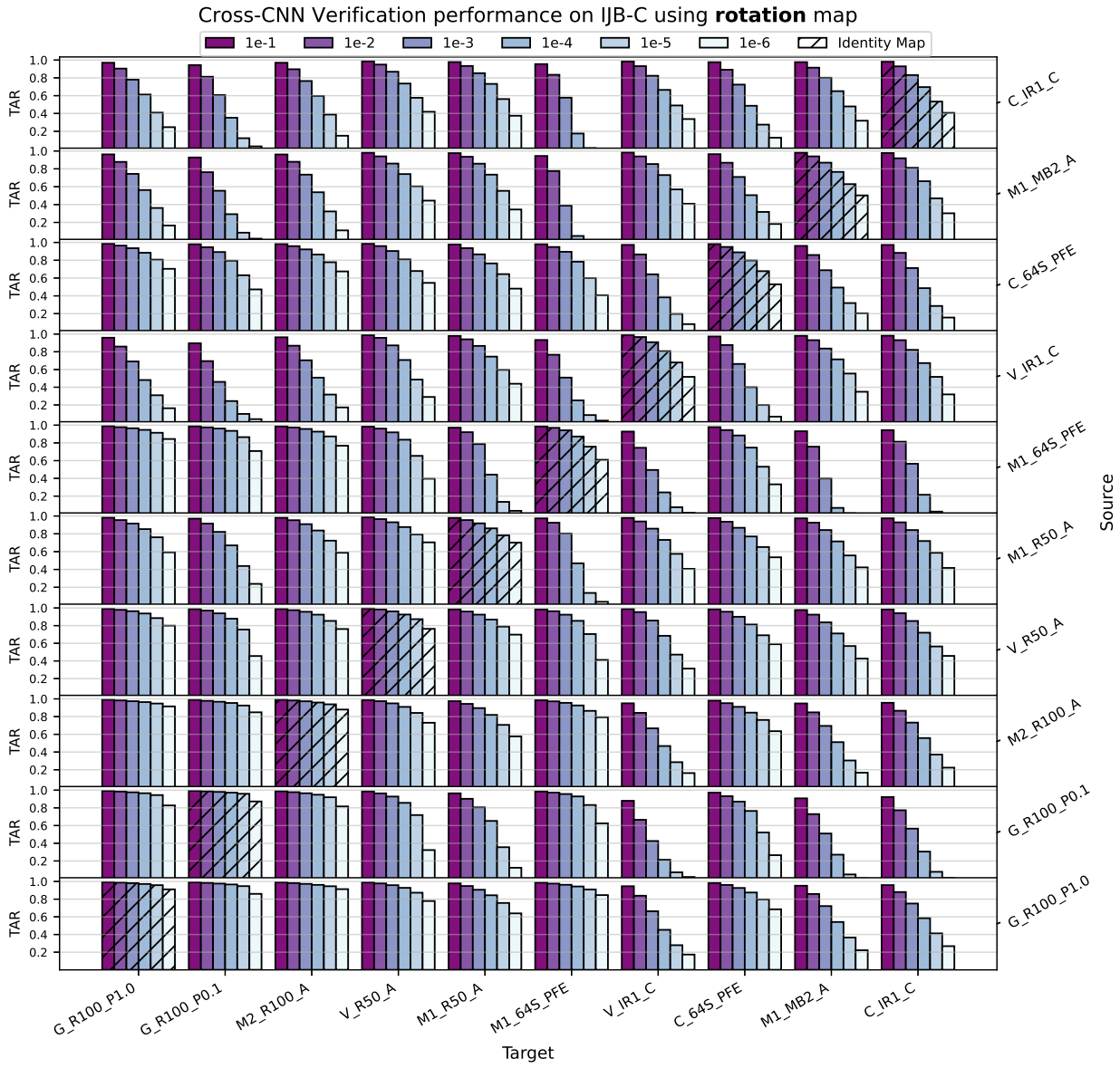


Figure B.2: Full **rotation** mapping performance results, in the same format as Figure 4.2

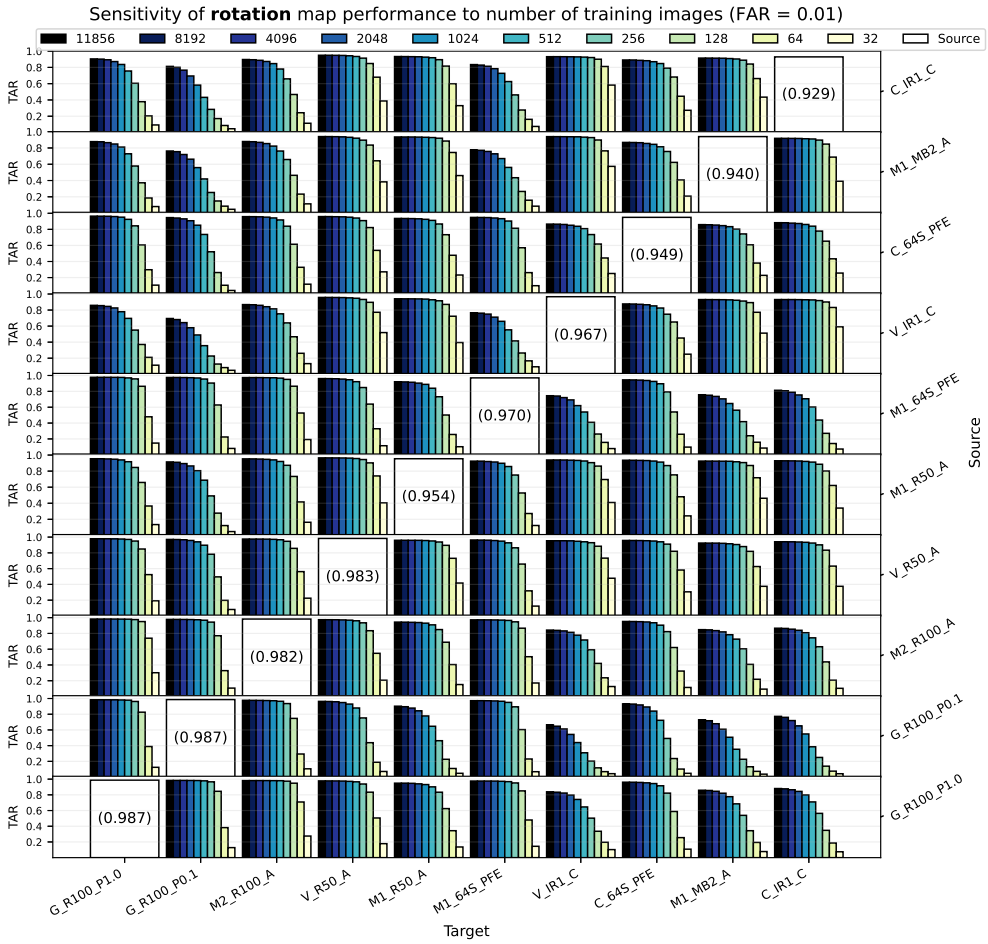


Figure B.3: Full mapping sensitivity results, including the same data presented in Figure 4.3 and following the same format. All performances represented here are generated from mapped features (i.e. no single-model performance is included).

Appendix C

Extended Face Subspace Angle Results

Table C.1: List of figures in Appendix C, including corresponding same-format figures from Section 5.2.

Figure type	**4-model version** (Section 5.2)	**6-model version**
Geodesic heatmaps	5.3	C.1
Pairwise heatmaps	5.4	C.2
Baseline-proportional geodesic heatmaps	5.5	C.3
Baseline-proportional pairwise heatmaps	5.6	C.4
Principal angles by covariate	5.7	C.5
Principal angles by variance proportion	5.8	C.6

As mentioned in Section 5.2, 4 of the 10 models studied were selected for initial discussion of results. The equivalent figures are each replicated here for the remaining 6 models, with similar patterns and trends. One edge case stands out, model G_R100_P0.1, which expresses variance across the greatest proportion of dimensions, and produces unique principal angle curves (Figures C.5 and C.6). Table C.1 is provided to facilitate easier navigation and comparison between these figures.

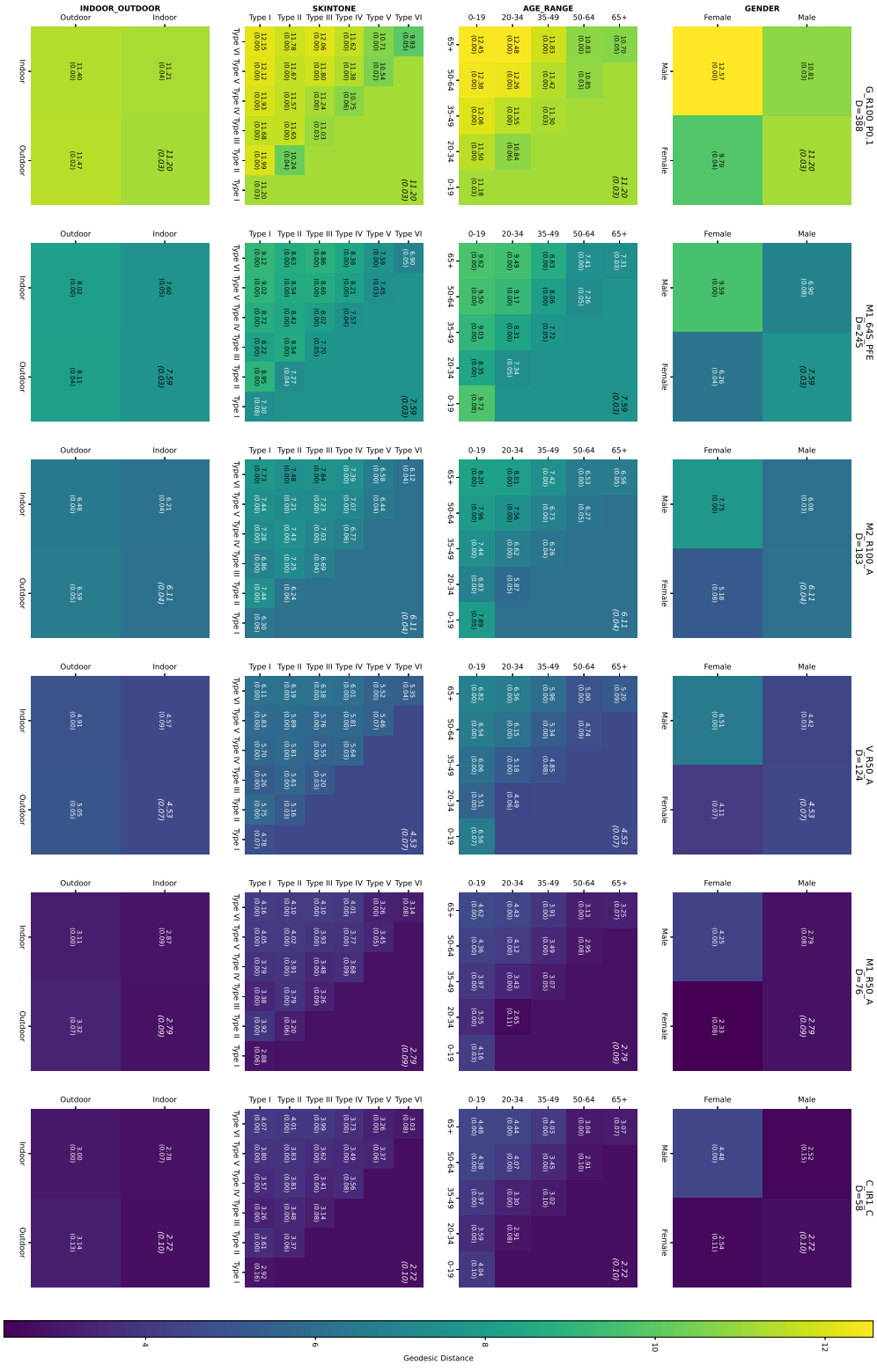


Figure C.1: Figure content and format is exactly the same as Figure 5.3, except in the models studied. (Nav table)

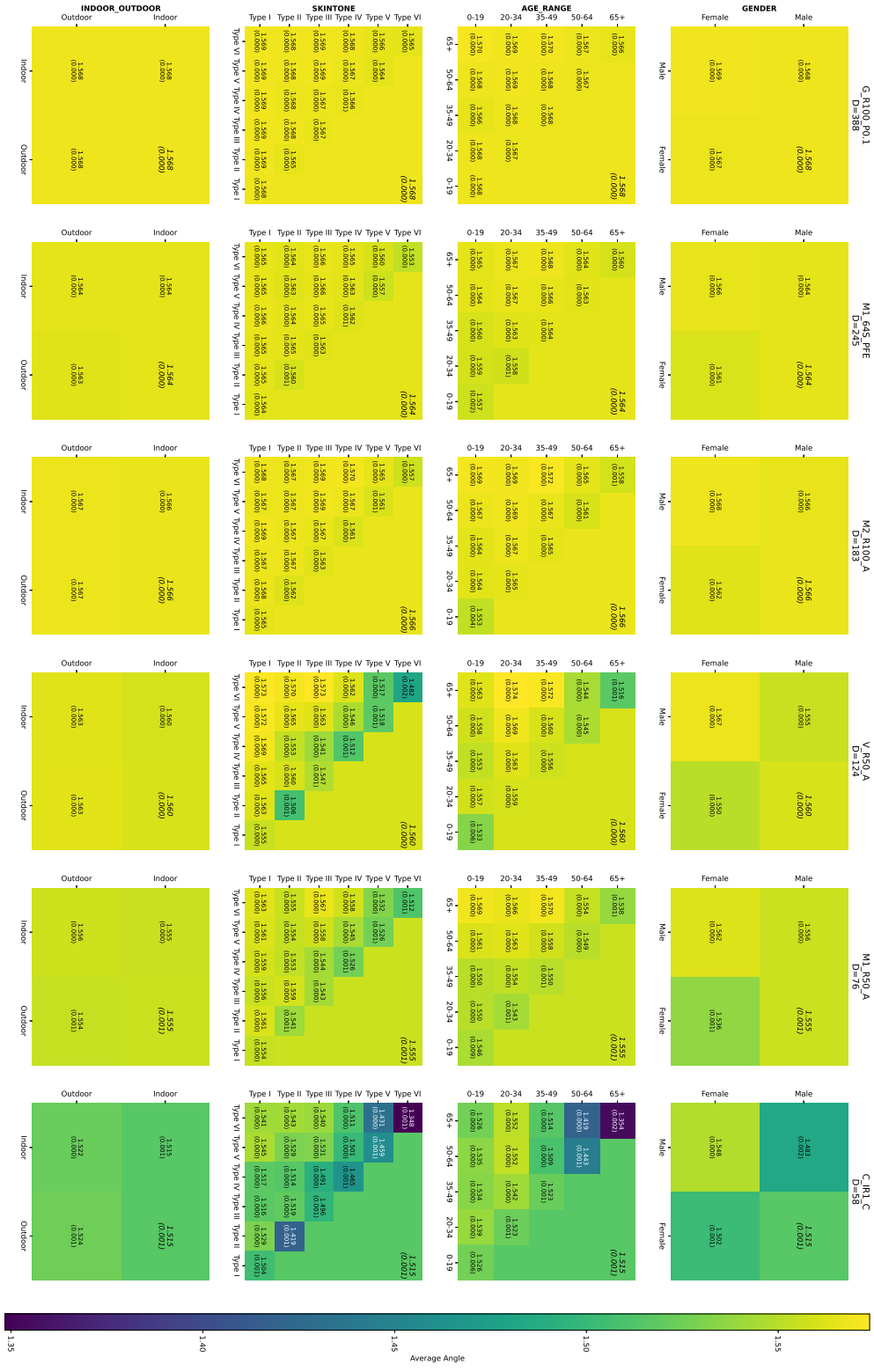


Figure C.2: Figure content and format is exactly the same as Figure 5.4, except in the models studied. (Nav table)

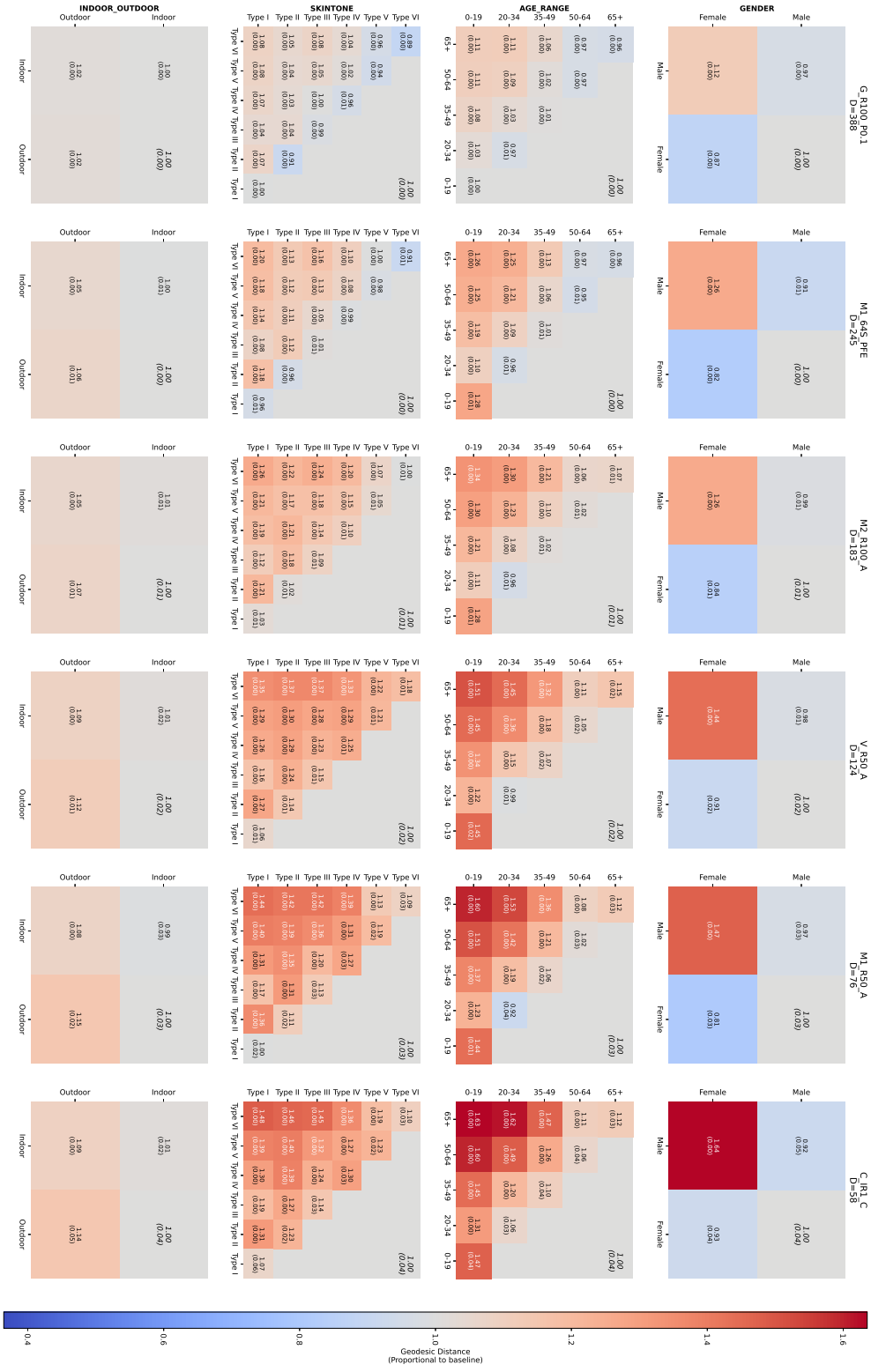


Figure C.3: Figure content and format is exactly the same as Figure 5.5, except in the models studied. (Nav table)

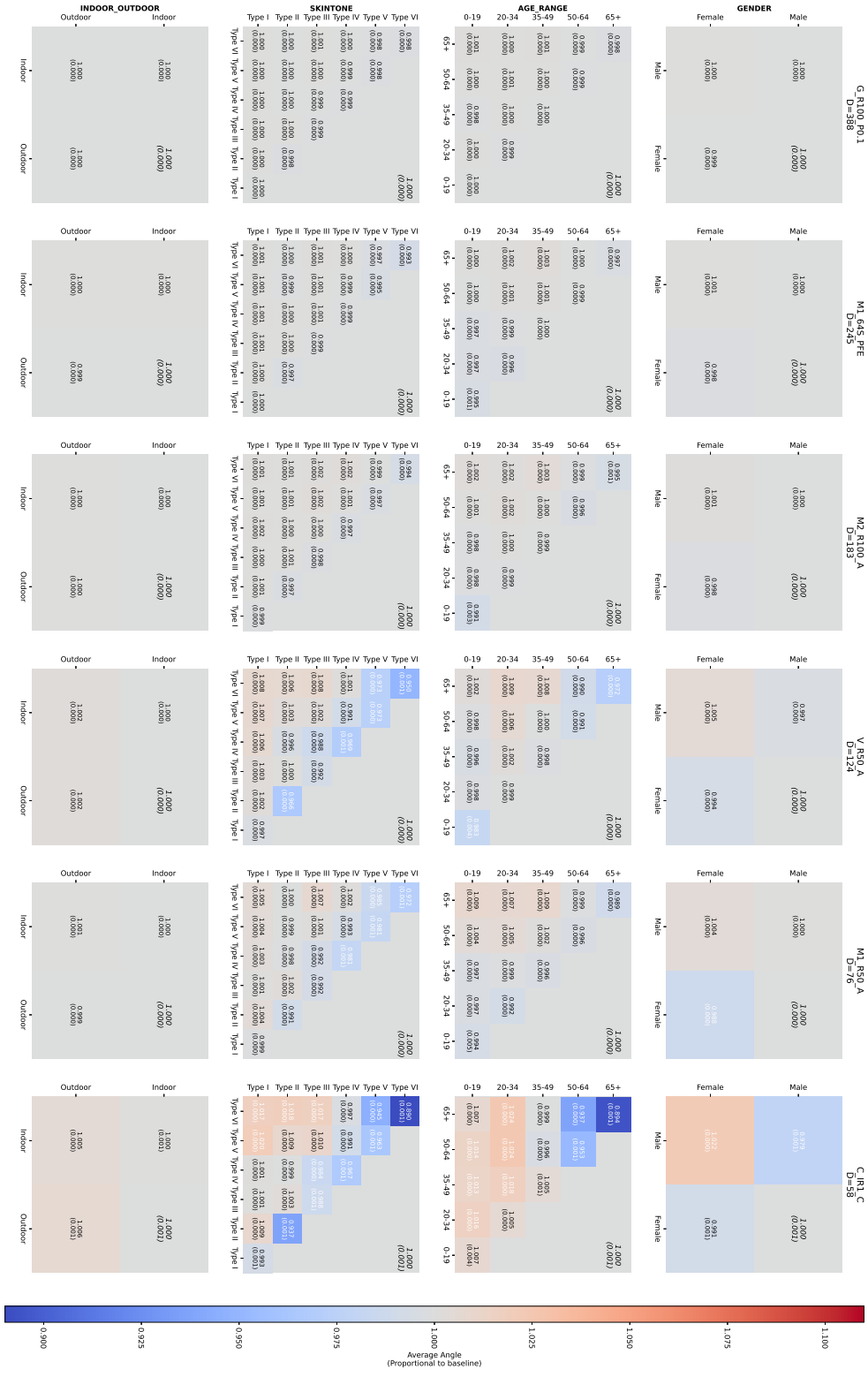


Figure C.4: Figure content and format is exactly the same as Figure 5.6, except in the models studied. (Nav table)

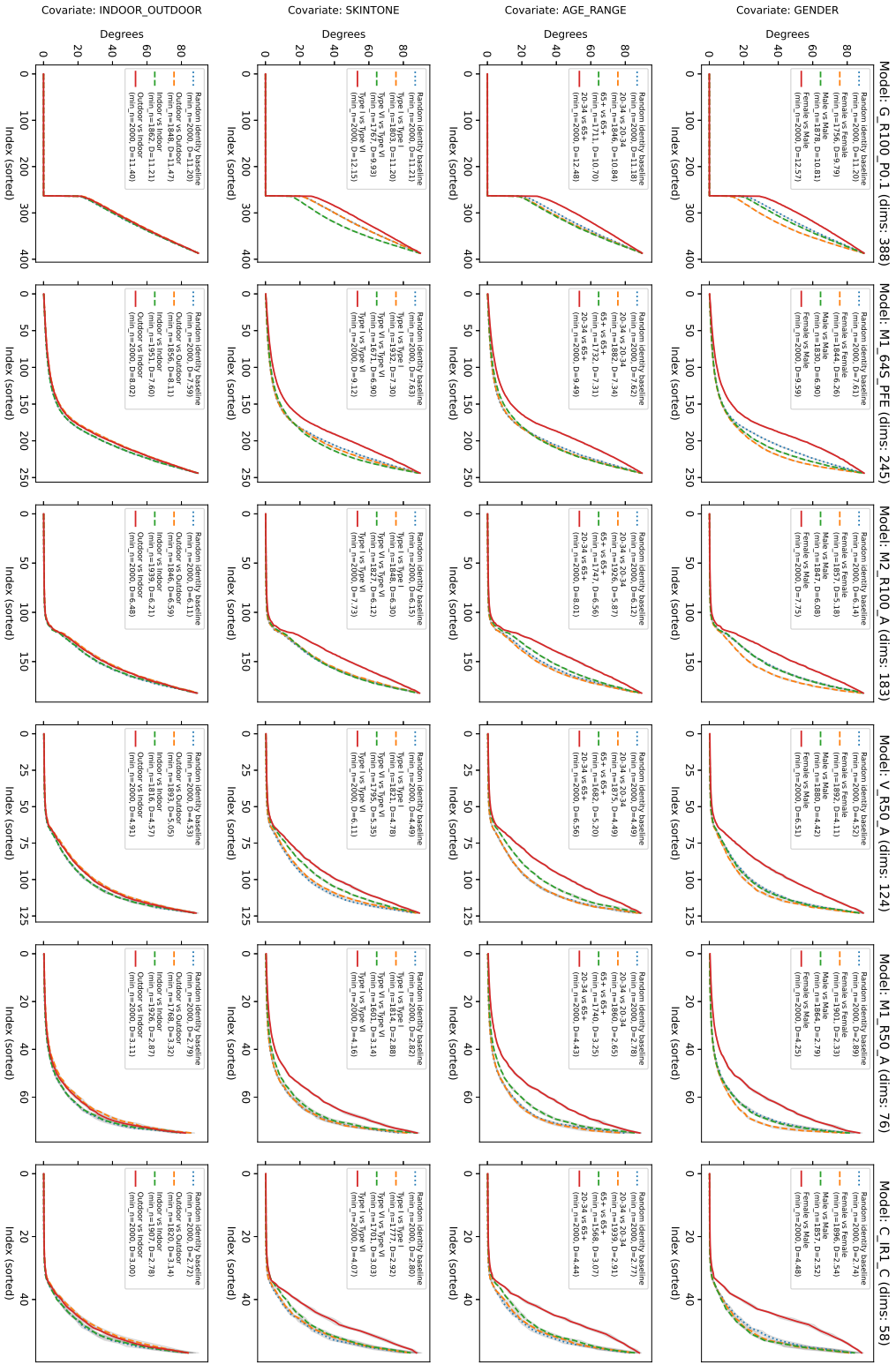


Figure C.5: Figure content and format is exactly the same as Figure 5.7, except in the models studied. (Nav table)

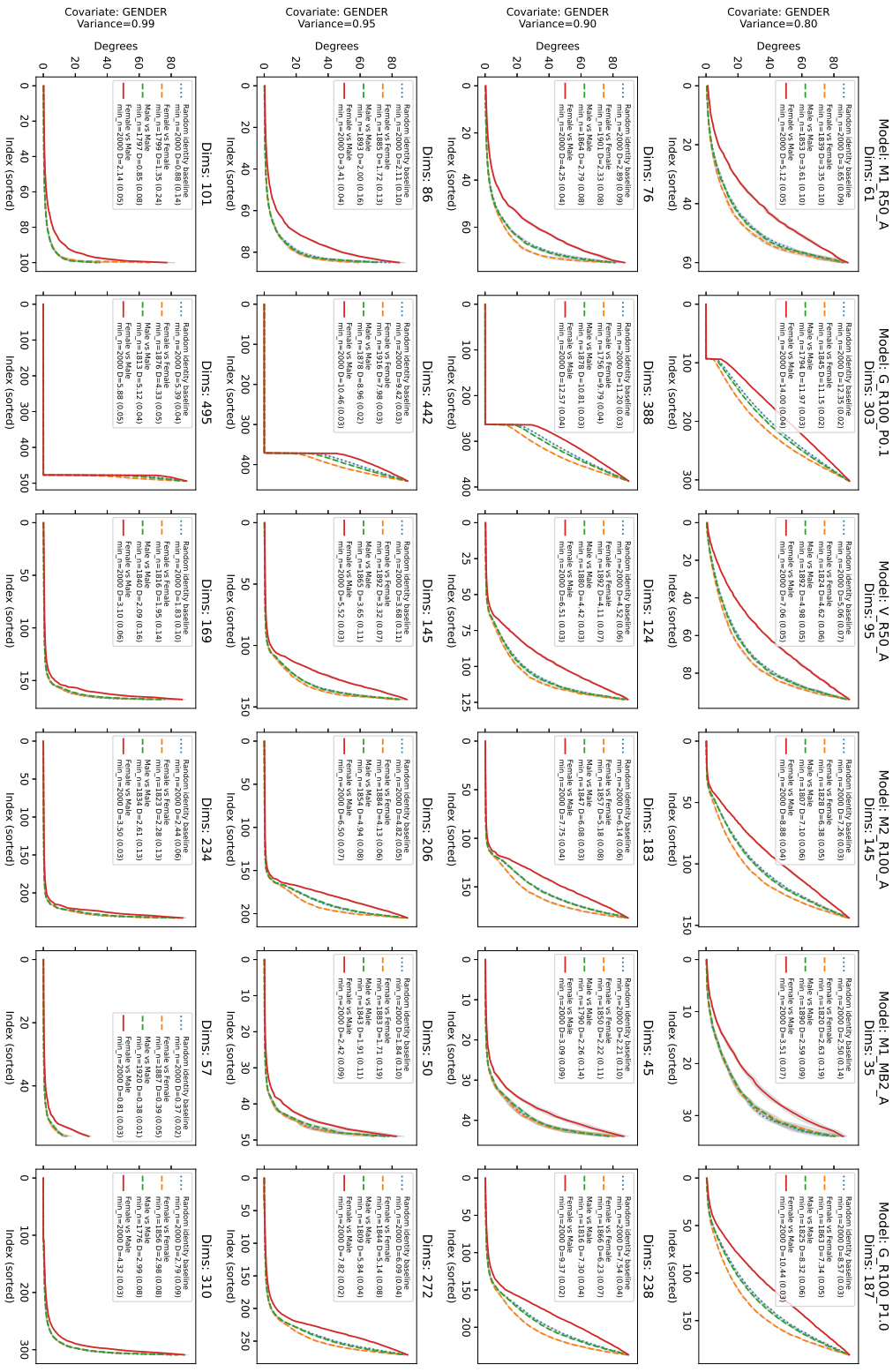


Figure C.6: Figure content and format is exactly the same as Figure 5.8, except in the models studied. (Nav table)