

The SVD of Convolutional Weights: A CNN Interpretability Framework*

Brenda Praggastis[†] Davis Brown[†] Carlos Ortiz Marrero[‡] Emilie Purvine[†]
 Madelyn Shapiro[†] Bei Wang[§]

August 16, 2022

Abstract

Deep neural networks used for image classification often use convolutional filters to extract distinguishing features before passing them to a linear classifier. Most interpretability literature focuses on providing semantic meaning to convolutional filters to explain a model’s reasoning process and confirm its use of relevant information from the input domain. Fully connected layers can be studied by decomposing their weight matrices using a singular value decomposition, in effect studying the correlations between the rows in each matrix to discover the dynamics of the map. In this work we define a singular value decomposition for the weight tensor of a convolutional layer, which provides an analogous understanding of the correlations between filters, exposing the dynamics of the convolutional map. We validate our definition using recent results in random matrix theory. By applying the decomposition across the linear layers of an image classification network we suggest a framework against which interpretability methods might be applied using hypergraphs to model class separation. Rather than looking to the activations to explain the network, we use the singular vectors with the greatest corresponding singular values for each linear layer to identify those features most important to the network. We illustrate our approach with examples and introduce the DeepDataProfiler library, the analysis tool used for this study.

1 Introduction

Mathematical functions and equations provide elegant and concise expression of the relationships and dynamics of physical systems. While we might not understand the derivation or full significance of all the parameters in a given equation, we can still be persuaded to rely on its predictive value. We can be shown how to interpret the equation by linking its parameters to important values in the system and by expressing their relationships in terms of the dynamics of the system. Machine learning practitioners have long striven to obtain this same kind of interpretability for the trained neural networks they produce, but have had limited success due to their size and complexity [8, 10, 32].

*The research described in this paper is part of the MARS Initiative at Pacific Northwest National Laboratory. It was conducted under the Laboratory Directed Research and Development Program at PNNL, a Multiprogram National Laboratory operated by Battelle Memorial Institute for the U.S. Department of Energy under Contract DE-AC05-76RL01830.

[†]AI & Data Analytics Division, Pacific Northwest National Laboratory, Seattle, WA 98109.

[‡]AI & Data Analytics Division, Pacific Northwest National Laboratory, Richland, WA 99354; Department of Electrical & Computer Engineering, North Carolina State University, Raleigh, NC 27607.

[§]Scientific Computing and Imaging (SCI) Institute, University of Utah, Salt Lake City, UT 94112.

A trained deep learning model defines a system of multivariate continuous functions or tensor maps composed of linear and nonlinear maps referenced as layers. The success or failure of the model depends largely on the parameters or weights found in the linear layers, which are defined during a model’s training process. Consequentially, much interpretability literature offers methods for discovering semantic meaning from the feature maps or activations produced by the linear layers in order to tie them to the input domain [26, 3, 12].

Convolutional neural networks (CNNs) used for image classification provide the most accessible opportunities to derive semantic meaning from feature maps because the features of interest are visual objects. Image tensors can be used to probe network response by measuring their activation values [25]. Strong responses in convolutional layers indicate strong correlation (positive or negative) to the filters in the weight tensor, and in fully connected layers to the rows of the weight matrix. But, what does a strong correlation mean?

The singular value decomposition (SVD) holds a special place in the heart of data science and numerical linear algebra [36]. As a robust matrix factorization method it facilitates data compression [39, 2] and network pruning [29]. For our purpose the unitary nature of the singular vectors make the SVD an ideal factorization for understanding the dynamics, and hence the correlations measured by the weight tensors.

In the case of a multi-layer perceptron, domain concepts producing activations highly correlated to a singular vector are scaled by the corresponding singular value. Our belief is that these basic correlations drive the success of the model so that interpretability rests largely on learning what input features correlate most closely to the singular vectors. We define an SVD for convolutional layers in terms of the SVD for an unfolding of the weight tensor. This echos the handling of convolutional layers in Yoshida and Miyato [42] but differs from the approach used by Sedghi, Gupta, and Long [34]. We describe the differences in these approaches and demonstrate that our definition is consistent with notional ideas around correlation as well as recent empirical results on the spectral distribution of covariance matrices associated with fully connected layers.

Our contribution brings a novel perspective to interpretability research. Much of the work of feature attribution and visualization is focused on the channel activations generated by convolutional layers [26, 12, 25, 43, 23]. We consider the linear dynamics of the convolutional layers as transformations restricted to the subspace of receptive fields, the regions acted on by convolution. By shifting to the receptive field space it is possible to unfold the weight tensors into matrices to study their dynamics. This process gives mathematical meaning to the parameters and opens the door to discovery of unanticipated features in the input domain.

The paper is organized in two parts. Sections 2 to 4 give the mathematical justification behind our use of the SVD for convolutional layers. Sections 5 and 6 describe how the SVD might be used to guide interpretability research. We close with examples, and introduce the DeepDataProfiler library [28] used for this work.

2 Motivation and Background

The singular value decomposition of a real-valued matrix M is a factorization $M = USV^T$ into two orthonormal change of basis matrices U and V , whose columns are called singular vectors, and a diagonal matrix S of singular values. When viewed as a linear transformation, $M : \mathbb{R}^m \rightarrow \mathbb{R}^n$, this decomposition exposes the dynamical behavior of the transformation as a scaling of the subspaces spanned by the singular vectors [6]. The eigenvalues of the correlation matrix MM^T are the squares of the singular values.

Applying random matrix theory to deep learning, Martin and Mahoney [19, 20, 21] examine the

distribution of the eigenvalues of MM^T when M is the weight matrix of a fully connected layer of a trained neural network. They use these distributions to define a capacity metric for the model, which correlates with the quality of the training process. Their work points to the use of the SVD to understand training dynamics and the potential importance of the singular values in understanding the decision process used by the model. We discuss this more in section 4.

In their study of the learning dynamics of neural networks, Saxe, McClelland, and Ganguli [33] demonstrate that the training of a shallow multi-layer perceptron involves learning the semantic hierarchical structure of the domain data. Moreover, this knowledge is captured by the singular vectors and singular values learned by the network. This work is notable for its use of the projection onto the singular vectors as a measure of the importance of a feature for an individual classification.

While many different projections of hidden layer activations appear to be semantically coherent [37], Bau et al. [3] find evidence that the representations closer to the Euclidean (or “natural”) basis are more meaningful than random unitary transformations. This makes sense in part due to the element-wise nature of the activation maps following the linear transformations. But the dynamics of the linear transformations were learned under the constraints of the model’s architecture. For important domain features to persist as they are passed through the network, they must be scaled to persist through regularization. In particular, Dittmer, King, and Maass [9] demonstrate that only activations aligned to singular vectors with the largest corresponding singular values persist past ReLU.

From the above observations we infer that not only do the singular values indicate quality of training, but the singular vectors themselves may hold the key to understanding the latent features of the model. For this reason we use the singular vectors of an SVD to study the influence of convolutional and fully connected linear transformations on the network.

Both Saxe, McClelland, and Ganguli [33] and Martin and Mahoney [19, 20, 21] restrict their analysis to the fully connected linear layers.¹ To extend their work to all layers of a CNN, our first task is to extend the singular value decomposition to the convolutional layers and project the layer-wise activations onto the singular vectors to measure correlations.

A common approach for interpreting the activations produced by the weights of the hidden layers is to optimize an input to the CNN that reliably produces a large response in an activation of interest. This method has been most notably used to create *feature visualizations*, images optimizing a feature map, for image classification networks [37, 18, 38, 22]. We draw on these optimization techniques to gain an understanding of the concepts in the domain that correlate with the singular vectors for a set of examples.

Interpretability research often depends upon linking feature maps to predetermined human identifiable concepts [3, 31, 13]. For example, Concept Activation Vectors [13] and LIME [31] start with human-defined concepts and measure the model’s sensitivity to them. These are powerful tools for verifying the model’s ability to recognize important domain-centric concepts. But interpretability research starting from the premise that latent representations of trustworthy models must correspond to known domain-centric concepts assumes we know in advance everything in the domain that is meaningful. This could produce bias and eliminate the possibility that concepts used by the model to describe class distinctions could be very different than what we expect and yet still be domain-centric and legitimate for classification. As a consequence we will not try to incorporate these approaches as we are interested in first discovering what the model defines as important, and only then attaching them to something human interpretable. The subtle difference in discovering the semantics of the network versus the sensitivity of the network to predefined concepts is discussed extensively in Olah et al. [26, 24].

¹Martin and Mahoney [19, 20, 21] address convolutional layers but do not flesh out the details.

3 Preliminaries

Let \mathcal{D} be a domain of images partitioned into target classes $\mathcal{C} = \{C^j\}_{j=1}^N$. Let \mathcal{M} be an image classification CNN trained and tested to classify the images in \mathcal{D} into its N target classes. \mathcal{M} is composed of multiple *tensor maps* called *layers*. As discussed in the introductory remarks, our focus will be on a network’s linear layers (fully connected and convolutional)², which we denote as \mathcal{L} . Other layers in the network, such as ReLU, batch normalization, and pooling, are fixed by the architecture and learn any needed parameters indirectly from the layers in \mathcal{L} . While these other layers correspond to architectural considerations designed to constrain the learning process during training, improving accuracy and generalizability, the latent representations of the input data used for classification are ultimately determined by the linear layers.

The SVD expresses a linear map between vector spaces as a weighted sum of singular values times singular vectors. The SVD is commonly used for low rank approximations in data science [6]. We are interested in the geometric properties of a linear map exposed by the SVD, which describe the dynamics of the mapping.

The computation of the SVD of weight tensors in the fully connected layers is straightforward. However, while the linearity properties of convolution are well-understood, there is more than one way to express this operation using 2-tensors or matrices. The choice of representation affects the computational complexity of generating the SVD, and more importantly, informs the perspective from which we extract meaning from the matrix. Our goal is to choose a representation that accurately reflects the role the operation plays in a neural network and provides us with informative singular values within this context.

3.1 Cross-Correlation as Matrix Multiplication

First we recognize that the convolution used in a CNN is really cross-correlation, also known as a sliding dot product. Let \mathcal{L} be a convolutional layer in \mathcal{L} with weight tensor W . Let X be an input tensor to \mathcal{L} and $Y = W \star X$, where \star means cross-correlation. For ease of notation and without loss of generality, we make some basic assumptions about W and X . We assume that X is a 3-tensor of dimensions $c \times m \times m$, W is a 4-tensor of dimensions $d \times c \times k \times k$ where $k \leq m$, the stride is 1, and the operation is 2-dimensional.

Let $I = (i_0, i_1, \dots, i_{s-1}) \in \mathbb{N}^s$ denote a tuple. Let \mathbb{T}_I be the set of real-valued s -tensors (i.e., s -order tensors) where I indicates the size of each of the s dimensions. Each tensor $T \in \mathbb{T}_I$ is indexed by the set

$$\text{Index}(\mathbb{T}_I) = \{\alpha \in \mathbb{N}^s : \alpha_j < i_j, \forall j < s\},$$

and follows a row-major ordering. Following these conventions we say $W \in \mathbb{T}_{(d,c,k,k)}$ has d *filters*, each a 3-tensor with c *channels* and *spatial dimensions* of $k \times k$. Similarly, we say $X \in \mathbb{T}_{(c,m,m)}$ has c channels and spatial dimensions $m \times m$. In this setting, $Y \in \mathbb{T}_{(d,n,n)}$, for $n = m - k + 1$, has d channels and spatial dimensions $n \times n$. An element of Y is a valid³ sum of products:

$$Y_{h,i,j} = (W \star X)_{h,i,j} = \sum_{r,s,t} W_{h,r,s,t} \cdot X_{r,i+s,j+t}. \tag{1}$$

The *channel activations* of Y are its 2-dimensional slices indexed by the spatial dimensions and referenced as $Y_r = Y[r, :, :]$. The *spatial activations* are its 1-dimensional slices indexed by the

²A linear layer usually references a linear map followed by addition of a bias term and some non-linear map. For this work we will mean only the linear map when we reference a linear layer.

³By valid we mean that every term in the sum exists. Any padding required by the model to preserve tensor size is assumed to have already been applied to X .

channels and referenced as $Y_{i,j} = Y[:, i, j]$.

The set \mathbb{T}_I along with element-wise addition and scalar multiplication is a real-valued vector space isomorphic to $\mathbb{R}^{\Pi I}$, where $\Pi I = \prod_{k=0}^{s-1} i_k$ is the dimension of the vector space, i.e., the cardinality of its basis. As we will see, it is helpful to have access to this isomorphism. Let $\{\bar{\mathbf{e}}_{\bar{\alpha}}\}_{\bar{\alpha} \in [0.. \Pi I - 1]}$ be the standard Euclidean basis for $\mathbb{R}^{\Pi I}$. Define the corresponding Euclidean basis for \mathbb{T}_I using the subset of \mathbb{T}_I given by $\{\mathbf{e}_{\alpha}^I\}_{\alpha \in \text{Index}(\mathbb{T}_I)}$ such that $(\mathbf{e}_{\alpha}^I)_{\beta} = \delta_{\alpha\beta}$ ⁴ for $\beta \in \text{Index}(\mathbb{T}_I)$.

Define an isomorphism $\phi_I : \mathbb{T}_I \rightarrow \mathbb{R}^{\Pi I}$ by pairing the basis vectors using their indices: order the indices in $\text{Index}(\mathbb{T}_I)$ in ascending lexicographic order and pair each index with its position in the list. It is straightforward to check, if $\phi_I(\mathbf{e}_{\alpha}) = \bar{\mathbf{e}}_{\bar{\alpha}}$ then

$$\bar{\alpha} = \sum_{j=0}^{s-2} \left[\alpha_j \cdot \prod_{\beta=j+1}^{s-1} i_{\beta} \right] + \alpha_{s-1}. \quad (2)$$

For any tensor $T \in \mathbb{T}_I$, $\phi_I(T)$ is the *vectorized* form of T . We will often refer to this as the *flattening map*. If $\phi_I^{-1}(\bar{\mathbf{e}}_{\bar{\alpha}}) = \mathbf{e}_{\alpha}$, then α is defined recursively using the formulas:

$$\begin{aligned} \alpha_0 &= \left\lfloor \frac{\bar{\alpha}}{\prod_{\beta=1}^{s-1} i_{\beta}} \right\rfloor \\ \alpha_j &= \left\lfloor \frac{\bar{\alpha} - \sum_{\gamma=0}^{j-1} \alpha_{\gamma} \cdot \prod_{\beta=\gamma+1}^{s-1} i_{\beta}}{\prod_{\beta=j+1}^{s-1} i_{\beta}} \right\rfloor \text{ for } j < s-1 \\ \alpha_{s-1} &= \bar{\alpha} - \sum_{\gamma=0}^{s-2} \alpha_{\gamma} \cdot \prod_{\beta=\gamma+1}^{s-1} i_{\beta}. \end{aligned} \quad (3)$$

This isomorphism is useful because it provides a consistent way to reshape a tensor. For suppose $J \in \mathbb{N}^{s'}$ such that $\Pi J = \Pi I$, then $\phi_{IJ} = \phi_J^{-1} \circ \phi_I : \mathbb{T}_I \rightarrow \mathbb{T}_J$ is also an isomorphism. In general we call ϕ_I , ϕ_J and ϕ_{IJ} *reshaping maps* and reference them simply as ϕ when I and/or J are clear from context.

In order to rewrite eq. (1) as a simple dot product of two 1-dimensional tensors, define a projection map $\psi_{i,j}^{(m,k)} : \mathbb{T}_{(c,m,m)} \rightarrow \mathbb{T}_{(c,k,k)}$ as

$$\psi_{i,j}^{(m,k)}(X) = \sum_{r \leq c; s, t \leq k} X_{r, i+s, j+t} \cdot \mathbf{e}_{r, s, t}.$$

The tensor $\psi_{i,j}^{(m,k)}(X) \in \mathbb{T}_{(c,k,k)}$ is the *receptive field* associated with the spatial activation vector in Y with spatial index (i, j) . Now eq. (1) becomes:

$$Y_{h, i, j} = \phi_{(c,k,k)}(W_h) \cdot \phi_{(c,k,k)} \circ \psi_{i,j}^{(m,k)}(X). \quad (4)$$

The first term in the dot product is the flattened filter W_h . The second term in the dot product is a flattened receptive field in X .

The inverse of the projection provides a natural embedding $(\psi_{i,j}^{(m,k)})^{-1} : \mathbb{T}_{(c,k,k)} \rightarrow \mathbb{T}_{(c,m,m)}$, such that for all $T \in \mathbb{T}_{(c,k,k)}$:

$$(\psi_{i,j}^{(m,k)})^{-1}(T) = \sum_{r, s, t} T_{r, s, t} \cdot \mathbf{e}_{r, i+s, j+t}.$$

⁴The notation means that the β^{th} component of the α^{th} basis vector is equal to the Kronecker delta: $\delta_{\alpha\beta} = 1$ if $\alpha = \beta$ and 0 otherwise.

$$\begin{array}{ccc}
\mathbb{T}_{(d,c,k,k)} \times \mathbb{T}_{(c,m,m)} & \xrightarrow{\chi} & \mathbb{T}_{(d,n,n)} \\
\phi \times \Psi \downarrow & & \downarrow \phi \\
\mathbb{T}_{(d,ck^2)} \times \mathbb{T}_{(ck^2,n^2)} & \xrightarrow{\bar{\chi}_1} & \mathbb{T}_{(d,n^2)}
\end{array}
\qquad
\begin{array}{ccc}
\mathbb{T}_{(d,c,k,k)} \times \mathbb{T}_{(c,m,m)} & \xrightarrow{\chi} & \mathbb{T}_{(d,n,n)} \\
f \times \phi \downarrow & & \downarrow \phi \\
\mathbb{T}_{(dn^2,cm^2)} \times \mathbb{T}_{(cm^2)} & \xrightarrow{\bar{\chi}_2} & \mathbb{T}_{(dn^2)}
\end{array}$$

(a) Cross-correlation performed as matrix multiplication using \bar{W}_1 . (b) Cross-correlation performed as matrix multiplication using \bar{W}_2 .

Figure 1: Matrization of weight tensors.

Use the embedding map to rewrite eq. (1) as:

$$Y_{h,i,j} = \phi_{(c,m,m)} \circ (\psi_{i,j}^{(m,k)})^{-1}(W_h) \cdot \phi_{(c,m,m)}(X). \quad (5)$$

The first term in this dot product is the filter W_h embedded in $\mathbb{T}_{(c,m,m)}$ and then flattened into \mathbb{R}^{cm^2} . The second term in the dot product is the flattened input tensor X . The difference between eqs. (4) and (5) is the domain where the dot product is being performed. In eq. (4) the vectors belong to \mathbb{R}^{ck^2} , while in eq. (5) the vectors belong to \mathbb{R}^{cm^2} . Typically $k \ll m$ so that eq. (4) would seem preferable.

3.2 Two Matrix Representations of Cross-Correlation

Cross-correlation can be represented as the bilinear map:

$$\chi : \mathbb{T}_{(d,c,k,k)} \times \mathbb{T}_{(c,m,m)} \rightarrow \mathbb{T}_{(d,n,n)}. \quad (6)$$

The bilinear map in turn can be expressed as matrix multiplication using either of eq. (4) or eq. (5).

Equation (4) performs the dot product in \mathbb{R}^{ck^2} . We reshape $W \in \mathbb{T}_{(d,c,k,k)}$ into $\bar{W}_1 = \phi(W) \in \mathbb{T}_{(d,ck^2)}$, so that \bar{W}_1 is a matrix, where each row is a flattened filter from W . Note that \bar{W}_1 is an *unfolding* of the weight tensor similar to what is described in Kolda and Bader [15]. We also reshape $Y \in \mathbb{T}_{(d,n,n)}$ to $\bar{Y}_1 = \phi(Y) \in \mathbb{T}_{(d,n^2)}$. Define the receptive field matrix $\Psi(X) \in \mathbb{T}_{(ck^2,n^2)}$ so that each column is the flattened receptive field of X given by

$$\Psi(X)_{:,i-n+j} = (\phi_{(c,k,k)} \circ \psi_{i,j}^{(m,k)}(X))^T. \quad (7)$$

We now represent the cross-correlation $W \star X$ as the matrix multiplication:

$$\bar{W}_1 \cdot \Psi(X) = \bar{Y}_1. \quad (8)$$

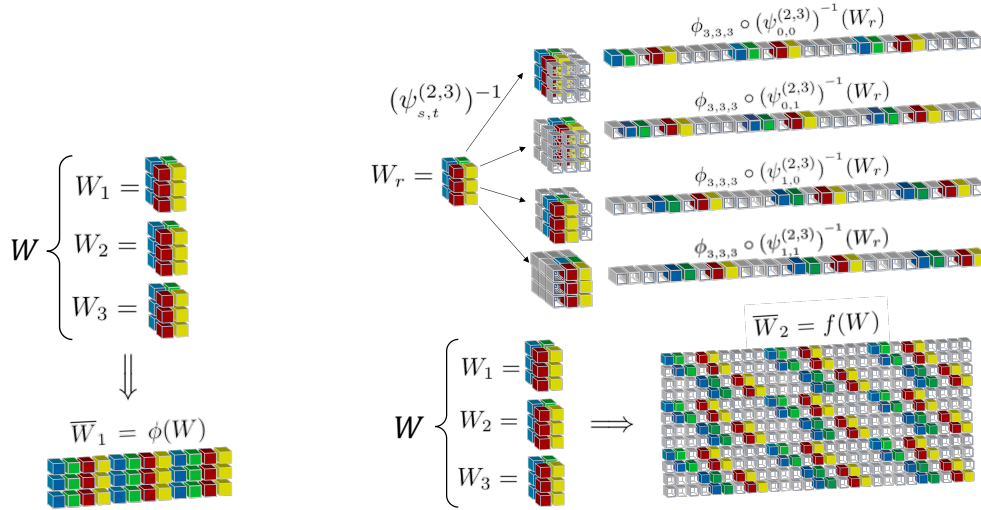
Let $\bar{\chi}_1$ be the bilinear map defined by eq. (8) required to complete the commutative diagram in fig. 1a.

Equation (5) performs the dot product in \mathbb{R}^{cm^2} . We flatten $X \in \mathbb{T}_{(c,m,m)}$ to $\phi(X) \in \mathbb{R}^{cm^2}$ and reshape $Y \in \mathbb{T}_{(d,n,n)}$ to $\bar{Y}_2 = \phi(Y) \in \mathbb{R}^{dn^2}$. Define an embedding

$$f : \mathbb{T}_{(d,c,k,k)} \rightarrow \mathbb{T}_{(dn^2,cm^2)} \quad (9)$$

and let $\bar{W}_2 = f(W)$ so that if $\phi_{(d,n,n)}(Y)_r = Y_{h,i,j}$ then the r^{th} row of \bar{W}_2 is a reshaped embedding of the h^{th} filter in W into \mathbb{R}^{cm^2} :

$$f(W)[r, :] = \bar{W}_2[r, :] = \phi_{(c,m,m)} \circ (\psi_{i,j}^{(m,k)})^{-1}(W_h). \quad (10)$$



(a) \bar{W}_1 is a reshaped weight tensor.

(b) \bar{W}_2 is a reshaped multiple embedding of the weight tensor into a larger tensor space.

Figure 2: **Illustration of the matrizations \bar{W}_1 and \bar{W}_2 of the weight tensor $W \in \mathbb{T}_{(3,3,2,2)}$.** Uncolored cubes correspond to zeros and cubes of the same color correspond to elements of the same spatial position in each filter

This gives us a second matrix representation for cross-correlation given by

$$\bar{W}_2 \cdot \phi(X) = \bar{Y}_2. \quad (11)$$

Let $\bar{\chi}_2$ be the bilinear map defined by eq. (11) required to complete the commutative diagram in fig. 1b. Note that \bar{W}_2 is a Toeplitz matrix representation of W . We illustrate the difference in the matrization of the tensors in fig. 2.

Equations (8) and (11) produce the same output tensors up to a simple reshaping isomorphism. Their difference lies in the coordinate systems in which their domains are represented, and in how we view their output. The matrix \bar{W}_2 can have up to cm^2 singular values. This is the matrization used in Sedghi, Gupta, and Long [34]. The matrix \bar{W}_1 can have up to ck^2 singular values. This is the map used by Yoshida and Miyato [42].⁵ Cross-correlation computes the spatial activations of Y independently, sharing the same dynamics with respect to the singular vectors. We claim that eq. (11) obfuscates these dynamics by requiring the high dimensional representation, while eq. (8) preserves the spirit of shared weights and the translation invariance of convolution by placing the redundancies in $\Psi(X)$.

4 The SVD in the Convolutional Layers

The unnormalized Gram matrix of the rows of \bar{W}_1 is defined by the $d \times d$ matrix $\bar{W}_1 \bar{W}_1^T$. The entries in the Gram matrix $\bar{W}_1 \bar{W}_1^T$ measure correlations between the linear maps defined by the

⁵An interesting discussion of the differences between the perspectives can be found in [34].

rows in \overline{W}_1 , i.e., the filters in W . This is analogous to the fully connected case.

In contrast, each filter is represented by n^2 rows in \overline{W}_2 so that its Gram matrix has dimensions $dn^2 \times dn^2$. Reordering the rows of \overline{W}_2 to group the filters by their embeddings shows the Gram matrix $\overline{W}_2 \overline{W}_2^T$ to have n^2 copies of $\overline{W}_1 \overline{W}_1^T$ down the diagonal while the off-diagonal elements correspond to correlations between incommensurate embeddings of the filters. It is unclear what meaning these off-diagonal elements convey about the action of the weight tensor on X .

To better understand the difference between the two matrix representations we consider the distribution of their singular values.

4.1 Validating the Matrix Representation of the Convolutional Layers

Martin and Mahoney [19, 20, 21] apply random matrix theory to analyze the distribution of the singular values of the weight matrices of a typical model \mathcal{M} . They demonstrate that well-trained generalizable models exhibit implicit self-regularization and these properties can be used to compute a metric for predicting test accuracy of CNNs without knowing the test data. They note that modern models learn the correlations in the data and these correlations are stored in the weight matrices. The empirical spectral distribution (ESD) of the Gram matrix of each linear layer \mathcal{L} tends to exhibit a heavy-tailed power law fit. The associated power law exponent $\alpha_{\mathcal{L}}$ provides a complexity metric for the layer \mathcal{L} ; the smaller the value of $\alpha_{\mathcal{L}}$ the greater the regularization [20]. Using a weighted average of these exponents, the authors define a capacity metric $\hat{\alpha}$ which is predictive of the test performance of the neural network [19]:

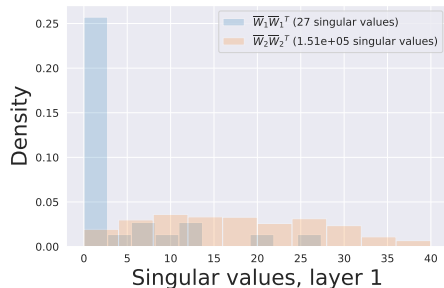
$$\hat{\alpha} = \sum_{\mathcal{L} \in \mathcal{L}} \alpha_{\mathcal{L}} \log \lambda_{\mathcal{L}}^{\max}, \quad (12)$$

where $\lambda_{\mathcal{L}}^{\max}$ is the maximum eigenvalue for the Gram matrix for layer \mathcal{L} . The smaller the value of $\hat{\alpha}$ the better the test performance.

It is notable that the experimental results performed in [20] are representative of the capacity metric restricted to fully connected layers. This is in part because the choice of matrix representation for the convolutional layers was in question [21]. While the authors suggest the theory could be extended to convolutional layers and use it to compute $\hat{\alpha}$ for a large number of architectures, they do not fully extend their theoretical results to these layers. In particular, the matrices used to compute the capacity metric $\hat{\alpha}$ in [21] were the channels of the weight kernels in the layer. Given the small size of the individual weight channels, it is unlikely that accurate power-law fits were obtained. Moreover, there is no a priori reason to believe that the correlations within a channel of the weight kernel is related to network capacity.

To validate our choice of matrix representation, we calculate $\hat{\alpha}$ for three different matrixizations and across three classes of architectures (ResNeXt [40], FD-MobileNet [30], and MeNet [41] models of varying depths) where the $\hat{\alpha}$ computed in [21] yielded mixed results.

Figures 3 and 4 show examples of the empirical spectral distributions and capacity metrics for models trained on ImageNet [7]. Figure 3 shows the spectral distribution for the Gram matrices associated with \overline{W}_1 and \overline{W}_2 for the convolutional layers of VGG-16 [35]. For the first layer of VGG-16, the Gram matrix for \overline{W}_1 has a spectral distribution consistent with the distribution found in the fully connected layers for the models tested, while \overline{W}_2 does not. In fig. 4, we find that \overline{W}_1 better aligns with theory and is considerably more predictive of test accuracy than \overline{W}_2 . If we view the heavy tail as corresponding to extracted correlations from the data, then the features themselves should correlate to the corresponding singular vectors. It is with this intuition that we use $\overline{W} = \overline{W}_1$ to represent W as a matrix. Similarly, let $\overline{Y} = \overline{Y}_1$.



(a) Singular values for the first layer of VGG-16 for $\overline{W}_1\overline{W}_1^T$ and $\overline{W}_2\overline{W}_2^T$. Note that the Toeplitz matricization \overline{W}_2 does not have a heavy tail.

layer	number of s-val		$\alpha_{\mathcal{L}}$ metric	
	\overline{W}_1	\overline{W}_2	\overline{W}_1	\overline{W}_2
2	64	3.21e6	1.66	4.63
3	128	3.21e6	1.8	3.77
4	128	6.42e6	1.87	4.48
5	256	1.61e6	3.9	5.05
6	256	3.21e6	2.07	3.42
7	256	3.21e6	4.42	2.92
8	512	3.21e6	5.58	3.27
9	512	6.42e6	4.14	2.57
10	512	1.61e6	3.03	2.28
11	512	1.61e6	4.81	2.58
12	512	1.61e6	4.09	2.48
13	512	1.61e6	4.02	1.75

(b) The number of singular values and the layer $\alpha_{\mathcal{L}}$ metric for $\overline{W}_1\overline{W}_1^T$ and $\overline{W}_2\overline{W}_2^T$. The Toeplitz matricization \overline{W}_2 has on the order of a million singular values for each layer.

Figure 3: **Validating our choice of matrix representation via the SVD for VGG-16 [35] trained on ImageNet [7] – Comparison of ESDs.** We compare the ESD of the Gram matrices for the two matricizations \overline{W}_1 and \overline{W}_2 across the convolutional layers. Note the large number of singular values for the centered Toeplitz matricization \overline{W}_2 , which has an unclear interpretation.

4.2 Decomposition of a Simple CNN

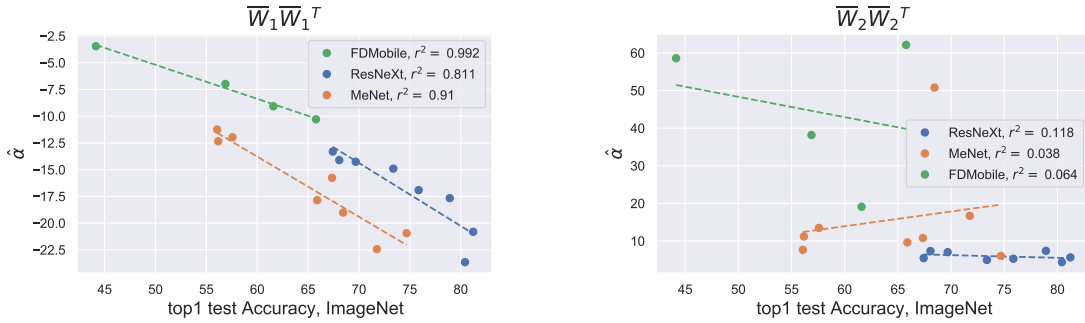
CNN interpretability literature tends to look to the channel and spatial activations of hidden layer representations to explain the feature maps learned by the model [26, 12, 13]. Since the channel and spatial activations of Y are projections onto a subset of the Euclidean basis for $\mathbb{T}_{(d,n,n)}$, this is consistent with the assertion in [3] that the projection of activations onto a basis close to the Euclidean basis provides more meaningful representations of stored features than projections onto a random orthonormal basis. But we have just observed there is a basis for each subspace of spatial activations derived from the singular vectors that holds the extracted features in terms of their singular vectors. The relationship between the channel activations and the singular vectors for a layer \mathcal{L} is described by the equation:

$$\overline{Y} = \overline{W} \cdot \Psi(X) = USV^T\Psi(X). \quad (13)$$

The factor $SV^T\psi(X)$ in the term on the right is a $d \times n^2$ matrix such that the i^{th} row is the vector of correlations gotten from the receptive fields of X with the i^{th} right singular vector. Each channel in Y is a reshaped row of \overline{Y} , which is a linear combination of these vectors of correlations.

To illustrate this relationship and other concepts in this work we use a simple CNN \mathcal{N} for classifying the MNIST dataset of handwritten digits [17]. The model \mathcal{N} consists of four linear layers: two convolutional layers, *conv1* and *conv2*, and two fully connected layers, *fc1* and *fc2*.⁶

⁶The full architecture is given by the sequence: *conv1*, ReLU, *conv2*, ReLU, Maxpool, *fc1*, ReLU, *fc2*, Softmax.



(a) The $\hat{\alpha}$ computed with \overline{W}_1 performs better than the original matrix choices used in [19].

(b) The $\hat{\alpha}$ computed with the Toeplitz matricization \overline{W}_2 are considerably less predictive of test accuracy.

Figure 4: **Validating our choice of matrix representation via the SVD for various models [40, 30, 41] trained on ImageNet [7] – Comparison of $\hat{\alpha}$.** We compute the $\hat{\alpha}$ metric eq. (12) vs test accuracy for \overline{W}_1 and \overline{W}_2 , and perform linear fits for each of the architecture classes and matricizations.

In fig. 5 we visualize each step of the decomposition for layer *conv1*. Let $X \in \mathbb{T}_{(1,28,28)}$ represent a sample image from MNIST. Layer *conv1* has weight tensor $W_{conv1} \in \mathbb{T}_{32,1,3,3}$ and $\overline{W}_{conv1} \in \mathbb{T}_{(32,9)}$. Let $U_{conv1} S_{conv1} V_{conv1}^T$ be the SVD of \overline{W}_{conv1} . Each array of images represents a 2-tensor or matrix reshaped for visualization purposes. For example the receptive field matrix $\Psi(X_{conv1}) \in \mathbb{T}_{(9,26^2)}$ has 9 rows, each of length 676, which are reshaped as 26×26 2-tensors and visualized as heatmaps rescaled for consistent display. The matrix $V_{conv1}^T \Psi(X)$ shows the projection of the receptive fields on each of the nine singular vectors. Matrix multiplication by $S_{conv1} \in \mathbb{T}_{(32,9)}$ rescales each row in $V_{conv1}^T \Psi(X)$ and embeds the 2-tensor into $\mathbb{T}_{32,26^2}$.⁷ Each heatmap is an indication of the strength of the signal in the direction of the corresponding singular vectors. The matrix $\overline{Y}_{conv1} = U_{conv1} S_{conv1} V_{conv1}^T \Psi(X)$ shows the projection of $S_{conv1} V_{conv1}^T \Psi(X)$ onto the Euclidean basis. Each small image of \overline{Y}_{conv1} is the heatmap of a channel of Y_{conv1} . The rows of \overline{Y}_{conv1} are linear combinations of the rows in $S_{conv1} V_{conv1}^T \Psi(X)$.

In [9], Dittmer, King, and Maass describe the relationship between the singular values pre- and post-ReLU in the linear layers of a multi-layer perceptron. In particular they show that ReLU has the effect of dampening signal and that only inputs strongly correlated to the right singular vectors with the largest singular values will persist through ReLU. We can extend their observation to the convolutional layers because the matrix multiplication of the fully connected layers is analogous to the matrix multiplication mapping receptive fields to spatial activations. In fig. 5, $ReLU(Y_{conv1} + B_{conv1})$ is the tensor that passes out of the layer after ReLU is applied; it includes a small translation by a bias term B_{conv1} . To see the dampening effect ReLU had on the original signals we include $U^T(ReLU(Y_{conv1} + B_{conv1}))$, which is the projection of the layer’s output onto the left singular vectors corresponding to nonzero singular values and is best compared with $S_{conv1} V_{conv1}^T \Psi(X)$.

4.3 The SVD of Weight Tensors

With the above observations, we put forth a simple definition for the SVD for the weight tensor used in convolutional layers. Let USV^T be an SVD of \overline{W} . We use the reshaping isomorphisms to

⁷We only show the projections onto singular vectors with nonzero singular values.

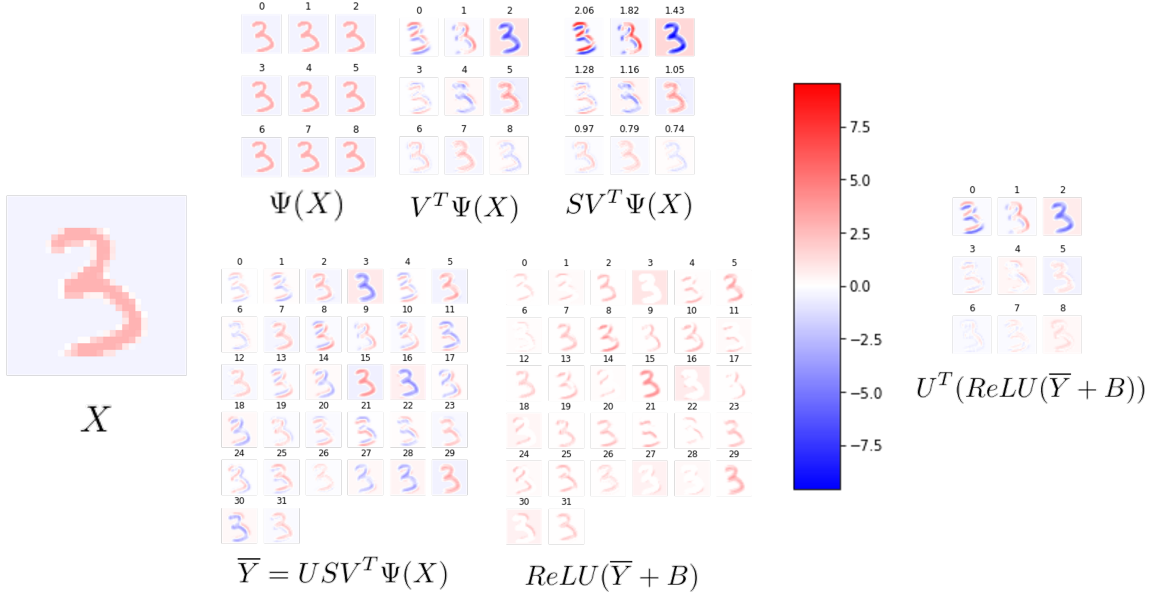


Figure 5: **Steps of computation for conv1:** Each array of images represents a 2-tensor or matrix. Each small image is a heatmap of a reshaped row of the matrix, scaled for consistent display with a blue-red color scheme to best discern contrast. The input to the layer is the tensor X and the output after ReLU is $ReLU(\bar{Y} + B)$. The singular values are shown with the corresponding projection above each image in $SV^T\Psi(X)$. To emphasize the dampening effect of ReLU on signal we also show the projection back onto the left singular vectors corresponding to nonzero singular values as $U^T(ReLU(\bar{Y} + B))$. The layer subscript *conv1* is not shown on any of the variables.

transform each matrix in the decomposition into a 4-tensor and replace matrix multiplication with tensor cross-correlation. Define the reshaping maps as $\phi_{V^T} : \mathbb{T}_{(ck^2, ck^2)} \rightarrow \mathbb{T}_{(ck^2, c, k, k)}$, $\phi_S : \mathbb{T}_{(d, ck^2)} \rightarrow \mathbb{T}_{(d, ck^2, 1, 1)}$, and $\phi_U : \mathbb{T}_{(d, d)} \rightarrow \mathbb{T}_{(d, d, 1, 1)}$. Using a stride of 1 for the \star operation, it is easily checked that for each $X \in \mathcal{D}$

$$W \star X = \phi_U(U) \star (\phi_S(S) \star (\phi_{V^T}(V^T) \star X)). \quad (14)$$

While we could use this *tensor SVD* for the rest of the discussion, we will stick with the matrix representation as it is more intuitive and reference the SVD for \bar{W} using:

$$\bar{W} \cdot \Psi(X) = USV^T\Psi(X) = \bar{Y}. \quad (15)$$

The column vectors $\{v_i\}$ of V form an orthonormal basis for \mathbb{R}^{ck^2} and the column vectors $\{u_i\}$ of U form an orthonormal basis for \mathbb{R}^d . The diagonal matrix $S \in \mathbb{T}_{(d, ck^2)}$ is non-negative and its diagonal entries are ordered in descending order, $s_0 \geq s_1 \geq \dots \geq s_r \geq 0$, such that $\bar{W}(v_i) = s_i u_i$ and $r = \min\{d, ck^2\}$.

The matrices U and V need not be unique as there are potentially multiple bases that could be chosen, in particular when the $\{s_i\}$ contains duplicates, but the subspaces corresponding to each distinct singular value are unique. Without loss of generality we assume there are no duplicate nonzero singular values so that each of the subspaces are 1-dimensional copies of \mathbb{R} .⁸

⁸This assumption eases the notation without changing the arguments and has been observed to be true in practice.

For each $v \in \mathbb{R}^{ck^2}$, there is a unique representation $v = \sum_{i=0}^{r-1} \langle v, v_i \rangle v_i$, where $\langle \cdot \rangle$ denotes the Euclidean inner product on \mathbb{R}^{ck^2} , and

$$\overline{W}v = \sum_{i=0}^{r-1} \overline{W} \langle v, v_i \rangle v_i = \sum_{i=0}^{r-1} s_i \langle v, v_i \rangle u_i. \quad (16)$$

The inner product $\langle v, v_i \rangle$ is a measure of correlation between the two vectors. When v is a receptive field (i.e. a column of $\Psi(X)$) we will call $s_i \langle v, v_i \rangle$ the *signal* of v in the direction of the singular vector v_i .

The sign of the signal indicates if the receptive field is positively or negatively correlated with v_i . The value of s_i indicates whether the correlation is increased or suppressed by the model before it is passed to a nonlinear activation and on to the next layer. Since \overline{W} operates independently on each receptive field, the latent representations generated by the model are essentially defined by these signals. From this we infer the discriminative power of a CNN lies in its singular vectors and interpretability might be achieved by determining the features of the domain which have strong positive correlation with the singular vectors of each layer.⁹

We overload our notation a bit and let X be the tensor representation for an image in \mathcal{D} . Let $\mathcal{L} \in \mathcal{L}$ and $\overline{W}_{\mathcal{L}}$ be the weight tensor for the layer. Let $X_{\mathcal{L}}$ be the latent representation for X used as input to the layer \mathcal{L} when passing X through the model. Let $r_{\mathcal{L}}$ be the number of nonzero singular values for $\overline{W}_{\mathcal{L}}$. Define the signal vector $\sigma_{\mathcal{L}}(X) \in \mathbb{R}^{r_{\mathcal{L}}}$ so that the i^{th} element of the vector is the average signal of all of the receptive fields in $\Psi(X_{\mathcal{L}})$ in the direction of v_i . The signal vectors provide a summary of signal strength in the direction of each singular vector. We acknowledge this is a coarse summary as it loses density information of how much signal is concentrated in one part of the image; nevertheless, strong average signals can still provide differentiation as we will see.

5 Hypergraphs and the Model’s Semantic Hierarchy

For each $X \in \mathcal{D}$ a singular vector v_i from layer \mathcal{L} is *significant* for X with respect to a threshold q if the i^{th} element of $\sigma_{\mathcal{L}}(X)$ is greater than q . A singular vector v_i from layer \mathcal{L} is *significant for a class C^j* with respect to a threshold q and a percentile p if v_i is significant for at least $p\%$ of the elements in C^j with respect to q . To be representative the signal should be significant for a majority of the class, so $p > 50$. We used $p = 75$ for most of our examples. Our thresholds were chosen for each layer by taking a high percentage quantile from the full distribution of signals for a representative sampling of the latent representations for images in \mathcal{D} . The greater the threshold, the fewer singular vectors will be significant for each image. Singular vectors significant for multiple classes are highly correlated to features common to those classes. Singular vectors significant for a single class are highly correlated to some feature that is more prevalent in that class than in other classes. By studying the many-to-many relationships between classes and the significant vectors in each layer we begin to define the discriminatory features used by the model.

5.1 Hypergraphs

Hypergraphs are generalizations of graphs which model the many-to-many relationships within data. Hypergraphs preserve the important mutual relationships that can be lost in ordinary graphs [4]. A hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{E})$ consists of a set of nodes \mathcal{V} and a set of hyperedges \mathcal{E} such that each $e \in \mathcal{E}$

⁹We considered both positive and negative correlation but found positive correlations were the most informative using the framework we outline here. However, further exploration of negative correlations could prove valuable.

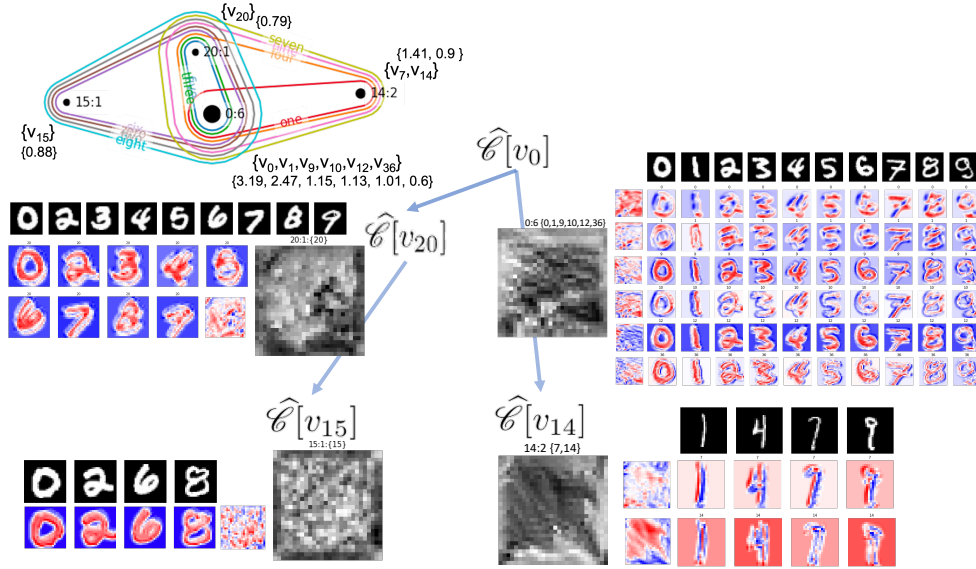


Figure 6: **Hypergraph and semantic hierarchy induced by $\mathcal{H}^{\text{conv2}}$.** The hypergraph uses a 85%-quantile threshold and 75% majority. Along with each optimized and exemplary image are its projections onto the subspaces spanned by the singular vectors in the equivalence class. Beside each equivalence class in the diagram are exemplary images from the corresponding target classes and a single image optimized using eqs. (19) and (20).

is a subset of \mathcal{V} . While graph edges correspond to exactly two nodes, hyperedges correspond to any number of nodes so that hypergraphs are often thought of as set systems but with more structure [1]. We model the relationships between the target classes and the singular vectors significant for each class using hypergraphs.

Fix a threshold q and a percentage $p\%$. For each layer $\mathcal{L} \in \mathcal{L}$ we construct a hypergraph $\mathcal{H}^{\mathcal{L}}$ with hyperedges indexed by the target classes $\{C^j\}$ and nodes indexed by the layer’s singular vectors $\{v_i\}$. A node v_i belongs to a hyperedge C^j if the singular vector v_i is defined as significant for $p\%$ of the class C^j with respect to the threshold q . We discard the index for any empty hyperedges or singular vectors not significant for any class. The resulting hypergraph describes differences between classes in terms of the singular vectors most highly correlated to images in these classes.

The size of the hypergraph depends on the parameters q and p . The higher the threshold and the percentage, the fewer nodes and hyperedges and the more specific certain singular vectors will be to fewer classes. By varying both parameters we are able to generate a collection of hypergraphs of differing sizes.

5.2 Semantic Hierarchy of a CNN

Let $\mathcal{H}^{\mathcal{L}}$ be the hypergraph of a linear layer \mathcal{L} of model \mathcal{M} defined by the parameters q and p . For each node v_i in $\mathcal{H}^{\mathcal{L}}$, let $\hat{\mathcal{C}}[v_i]$ be the set of hyperedges to which v_i belongs, i.e., the subset of target classes for which v_i is significant. Define an equivalence relation \sim on the set of nodes in $\mathcal{H}^{\mathcal{L}}$ so that any two nodes, v_i, v_j , are equivalent if and only if they belong to the same set of hyperedges. Let

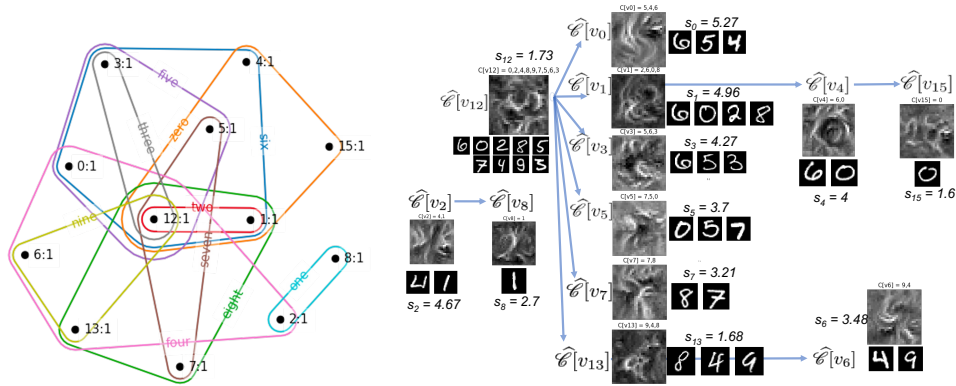


Figure 7: **Hypergraph and semantic hierarchy induced by \mathcal{H}^{fc1} .** The hypergraph uses a 95%-quantile threshold and 75% majority. Beside each equivalence class in the diagram are exemplary images from the corresponding target classes and a single image optimized using eqs. (19) and (20).

$[v_i]$ reference the equivalence class containing v_i . Then

$$v_i \sim v_j \iff \hat{\mathcal{C}}[v_i] = \hat{\mathcal{C}}[v_j] \iff [v_i] = [v_j]. \quad (17)$$

The hypergraph $\mathcal{H}^{\mathcal{L}}$ defines a partial ordering on the equivalence classes given by:

$$[v_i] > [v_j] \iff \hat{\mathcal{C}}[v_i] \supset \hat{\mathcal{C}}[v_j]. \quad (18)$$

This partial ordering induces a *semantic hierarchy* based on the features positively correlating to the singular vectors in each equivalence class. Let $[v_i] \neq [v_j]$ be two equivalence classes. The concepts in the domain described by features positively correlated with the singular vectors in $[v_i]$ will be considered more general than concepts from features positively correlated with the singular vectors in $[v_j]$ if $[v_i] > [v_j]$. Diagrammatically we represent this as $\hat{\mathcal{C}}[v_i] \longrightarrow \hat{\mathcal{C}}[v_j]$. We use this notation to diagram the semantic hierarchy induced by the hypergraph as in figs. 6, 7, and 9.

To illustrate, we construct two hypergraphs \mathcal{H}^{conv2} and \mathcal{H}^{fc1} for the model \mathcal{N} trained on MNIST of section 4.2. We use a test set with 50 images in each class using thresholds corresponding to the top 85%- and 95%-quantiles respectively for at least 75% of the classes. We visualize the hypergraphs in figs. 6 and 7.¹⁰ Rather than displaying all of the nodes in the hypergraph, the nodes are grouped by equivalence class and labeled by the index of one singular vector in the class followed by the number of nodes in the class. Next to each hypergraph we show the corresponding diagram depicting the semantic hierarchy induced by the hypergraph along with exemplary images from each class. We use these diagrams to direct our interpretability analysis to the singular vectors most significant for each class and hence most influential for discriminating the class.

6 A Suggested Framework for Interpretability

One goal of model interpretability is to identify the domain-centric features the model uses for classification. We have seen that any enhancement or suppression of a feature from the data depends

¹⁰Hypergraph figures were generated by the HyperNetX library [27].

on the correlation of its latent representations with the singular vectors. From the point of view of the model, the features used for classification are the ones with latent representations most highly correlated to the singular vectors with the greatest singular values.

In this section we suggest a methodology for interpreting image classification CNNs using the framework defined by the hypergraphs and semantic hierarchies of the model described in section 5 and figs. 6, 7, and 9. The diagram generated by the semantic hierarchy indicates how the model responds to images of different classes. Features corresponding to singular vectors particularly significant for one class but not for another characterize how the model differentiates classes.

Using existing visualization techniques, we look for features highly correlated with the singular vectors and their equivalence classes. By identifying the domain-centric features correlated to each equivalence class we identify the features that provide linear separability within the model.

This is very much in the spirit of Olah et al. [25, 26] where features are discovered by optimizing images with latent representations positively correlated to the channels of the weight tensors. The difference between projecting on the Euclidean directions and the singular vectors is illustrated in fig. 8. Projection onto the singular vectors preserves the dynamical changes imposed by the layer on its input. Much of this information can be lost when projecting onto the Euclidean basis vectors. By optimizing for a specific singular vector we look for features in the domain that the model either deems important, hence increases their value in the latent representation, or unimportant, hence decreases their value. This provides meaning to the learned parameters and explains the decision process in terms of the dynamics of the linear layers.

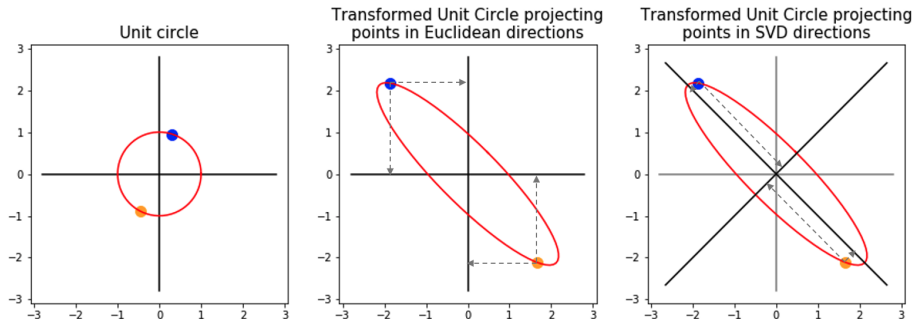


Figure 8: Projections of linearly transformed circle show that the dynamics (shrinking and stretching) of the map are better captured when projecting in the direction of the singular vectors.

We generate an optimal image X with a latent representation $X_{\mathcal{L}}$, which highly correlates with a particular singular vector v_i in layer $\mathcal{L} \in \mathcal{L}$ by computing:

$$\begin{aligned}
 X_{opt} &= \arg \max_{T \in \mathbb{T}_{(c,m,m)}} \sum_j (s_i v_i^T \Psi(T_{\mathcal{L}}))[j] \\
 &= \arg \max_{T \in \mathbb{T}_{(c,m,m)}} \sum_j (u_i^T \overline{W} \Psi(T_{\mathcal{L}}))[j]
 \end{aligned} \tag{19}$$

subject to the transformation robustness constraints from [25].

We generate an image maximally triggering a collection of singular vectors $\{v_i\}$ using a sum of

weighted terms:

$$\begin{aligned}
 X_{opt} &= \arg \max_{T \in \mathbb{T}_{(c,m,m)}} \sum_i w_i \left(\sum_j (s_i v_i^T \Psi(T_{\mathcal{L}}))[j] \right) \\
 &= \arg \max_{T \in \mathbb{T}_{(c,m,m)}} \sum_i w_i \left(\sum_j (u_i^T \overline{W} \Psi(T_{\mathcal{L}}))[j] \right)
 \end{aligned} \tag{20}$$

for some set of weights $\{w_i\}$.

As tantalizing as these images can sometimes be, we must be careful. If the objective function simply adds the signals across the spatial activations as we did in eqs. (19) and (20) it could smear signal out across an input image and lose the density of the signal actually found in the images. Recognizing the limitations of feature visualization, we supplement our analysis by identifying exemplary images, which highly activate the signal or group of signals in an equivalence class. This is done by restricting eqs. (19) and (20) to tensors in $\mathbb{T}_{(c,m,m)}$ representing images in the model’s test or training set. In early layers where latent representations retain much of the spatial information of the original image, we examine the scaled projects of spatial activations from the latent representations onto the singular vectors and study the heatmap of signals produced. The heatmaps define spatial regions with features highly correlated with the singular vectors similar to a saliency map. By comparing optimized images and heatmaps of projections with exemplary images we hope to gain intuition around the domain-centric features deemed important by the model. Where a dataset has large enough image resolution, similarity overlays from [11] can also be used to relate the feature visualizations and exemplary images.

6.1 Interpreting \mathcal{N} with MNIST

In fig. 6, hypergraph \mathcal{H}^{conv2} has 10 nodes divided into 4 equivalence classes. Equivalence class $[v_0]$ is at the top of the semantic hierarchy, because its elements are significant for every target class in MNIST [17] with respect to the 85%-quantile. This means the features, which are increased or suppressed, are common to images in all of the classes. We also show optimized and exemplary images for each of the equivalence classes in \mathcal{H}^{conv2} . We found feature visualization to be more informative in the later layers of larger networks, where the receptive fields correspond to greater regions in the original image. For early layers and in shallow networks, much can be learned by visualizing the scaled projections of the latent representations of exemplary images onto the significant singular vectors in each equivalence class. For each exemplary image with tensor representation X and each significant vector v_i we compute the dot product of $s_i v_i$ with each of the columns of $\Psi(X)$. The resulting vectors are reshaped into 26×26 2-tensors and visualized in red, white, and blue. The color scale is set to show strong positive correlation with dark red and strong negative correlation in dark blue as is seen in the diagram.

Of particular interest are the features exhibited for $[v_{14}] = \{v_7, v_{14}\}$ and $[v_{15}] = \{v_{15}\}$. Both their exemplary images and their projections are quite different. The scaled projections onto v_{14} positively correlate with large black regions in the image and are slightly suppressed by the singular value 0.9. The scaled projections onto v_7 also positively correlate with large black regions but the signal is stronger where the black is just to the left of the white of the image. These regions are darker red and were slightly enhanced by the singular value 1.41. Images with high positive correlation to v_{15} have more white in their receptive fields and narrow bands of black around the edges. Their signals were slightly suppressed by the singular value 0.88. This branch in the semantic hierarchy of the first layer appears to separate images in part based on what portion of the image is background.

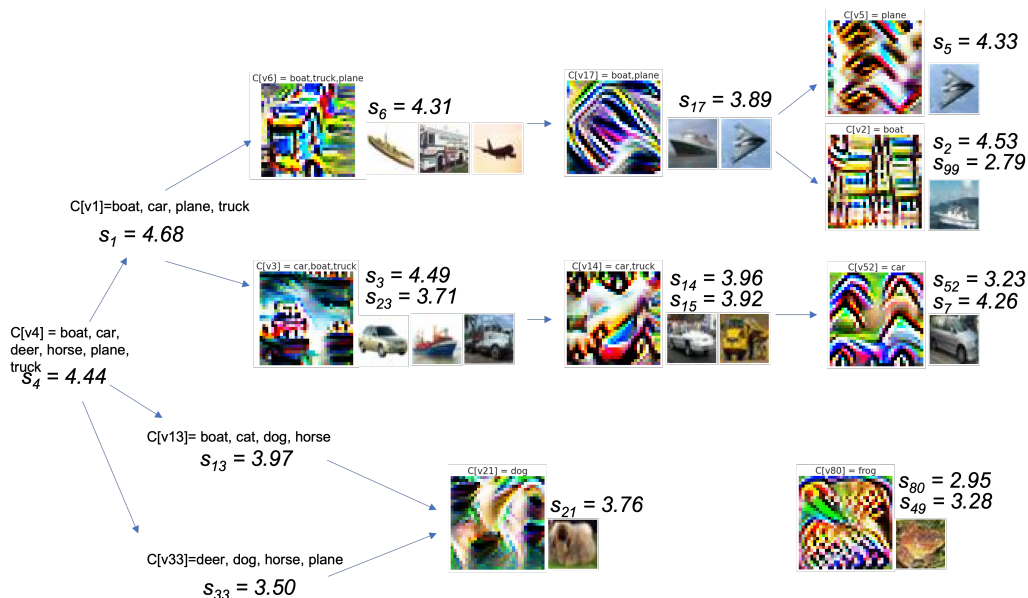


Figure 9: **Exploration of VGG-16 [35] on CIFAR-10 [17] – Diagram for part of the semantic hierarchy induced by a hypergraph for the 7th convolutional layer.** The hypergraph was created using a significance threshold equal to the 90%-quantile of all signals and percentage majority of greater than 50% for each class.

With deeper layers more complex shapes arise in the feature visualizations. Even in such a simple model as \mathcal{N} we see more discernible shapes in the optimized images for *fc1* than we did in *conv2*. In fig. 7 we see the greatest singular values, s_0 and s_1 are associated with recognizable forms including an *S*-curve and figure eight.

6.2 DeepDataProfiler

We use this framework to guide our exploration of VGG-16 [35] trained on CIFAR-10 [16] and ImageNet [7] in our DeepDataProfiler (DDP) library available on GitHub [28]. DDP is a Python library for generating attribution graphs called *profiles* for convolutional neural networks. The library includes a module for studying the SVD of convolutional layers along with Jupyter notebooks [14] for generating hypergraphs and feature visualizations.

Our goal is to understand the features used by the network to differentiate images in each of the datasets. The semantic hierarchy guides our inquiry by recognizing the features most strong for each image and class. For example, fig. 9 shows part of the semantic hierarchy for the 7th convolutional layer of VGG-16 when trained on CIFAR-10. Each level of the diagram displays an optimized image for the corresponding equivalence class, exemplary images from CIFAR-10, and the values of the corresponding singular vectors. One observable difference gleaned from the figure is the softer edges in the feature visualizations connected with living things versus the objects for transportation.

Feature visualizations for singular vectors associated with VGG-16 trained on ImageNet have superior resolution and tend to be more interpretable. Figure 10 shows a feature visualization for a singular vector significant to two images from the dog class in ImageNet. The similarity overlay

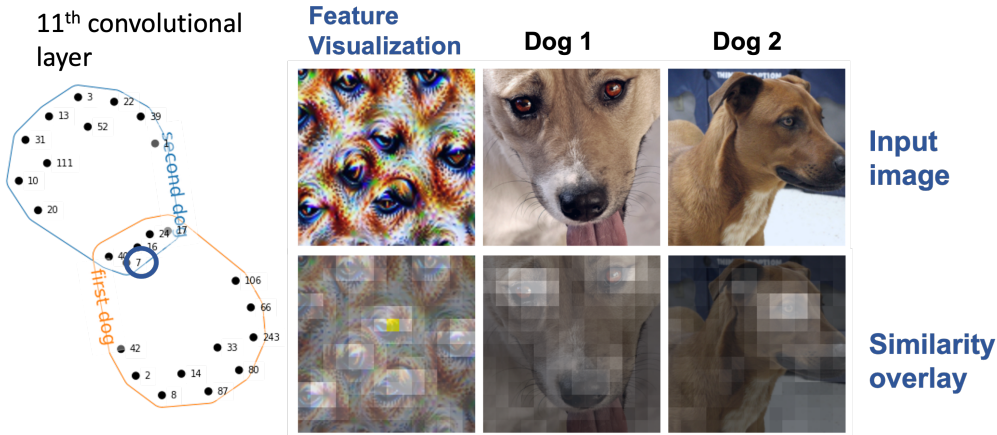


Figure 10: **Exploration of VGG-16 [35] on ImageNet [7] – Significant features shared by two images.** The hypergraph models the significant singular vectors for two dog images (using 50%-quantile) from the 11th convolutional layer. Exemplary images and feature visualization are for the singular vector v_7 . Similarity overlays highlight cosine similarity between the latent representations of the three images.

highlights the cosine similarity of the spatial activations in the 11th convolutional layer between the optimized image and the two dog images. Exemplary images and feature visualizations for each layer of VGG-16 trained on ImageNet are available from our Streamlit application [5], which supports the DDP library.

7 Conclusion

In this work we examine the linear transformations of the convolutional layers of a CNN. We describe a collection of isomorphisms and embeddings on the vector space of tensors over a fixed index set and define convolution as a bilinear map on these tensor spaces that can be represented as matrix multiplication. Our approach, which maps a 4-dimensional weight tensor of the convolutional layers isomorphically to a 2-dimensional matrix, provides an alternative to the commonly used Toeplitz representation for treating convolution as matrix multiplication. This simple unfolding of the weight tensor preserves the spirit of shared weights and exposes the dynamics of the linear map as it acts on the space of receptive fields.

Our choice of matricization is validated both notionally by looking at the corresponding Gram matrix and empirically by applying results from random matrix theory. The singular vectors of the matrix representation for the weight tensors identify the features of the input domain that the model has learned to either increase or suppress depending on their associated singular values.

We define the significance of the singular vectors to each of the target classes using two parameters and model this relationship using hypergraphs. By varying the parameters we generate a family of hypergraphs, each inducing a semantic hierarchy. We describe how these hierarchies might be used to explore the model and to discover the discriminative features it uses for classification. We introduce

the DeepDataProfiler library, which uses these matricizations to gain interpretability of CNNs.

This work brings fresh eyes to interpretability research by applying the same mathematical tools to convolutional layers that have been used to explain linear transformations for more than a century, giving us an opportunity to remove the black box stigma that haunts CNN models. By identifying the features the model deems important we can study their inter-relationships and determine their discriminatory value and relevance to the data domain. The greatest challenge moving forward will be to refine our optimization techniques, which translate the model’s representation of these features into something we can recognize. Future work should also explore the range of singular vectors to better understand how the model uses negative correlation and identifies noise.

As a final note, the implication of this new perspective extends beyond the scope of this paper. The SVD of convolutional layers offers us an opportunity to prune convolutional models and identify misclassifications analogous to the work of Dittmer, King, and Maass [9] on multi-layer perceptrons.

References

- [1] Sinan G. Aksoy, Cliff A. Joslyn, Carlos Ortiz Marrero, Brenda Praggastis, and Emilie Purvine. Hypernetwork science via high-order hypergraph walks. *EPJ Data Sci.*, 9(1):16, 2020.
- [2] Harry C. Andrews and Claude L. Patterson III. Singular value decomposition (SVD) image coding. *IEEE Trans. Commun.*, 24(4):425–432, 1976.
- [3] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 3319–3327. IEEE Computer Society, 2017.
- [4] Claude Berge. *Hypergraphs - combinatorics of finite sets*, volume 45 of *North-Holland mathematical library*. North-Holland, 1989.
- [5] Davis Brown, Madelyn Shapiro, and Brenda Praggastis. PNNL DeepDataProfiler visualization tool. Streamlit web app, 2022.
- [6] Steven L. Brunton and J. Nathan Kutz. *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge University Press, 2 edition, 2022.
- [7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 248–255. IEEE Computer Society, 2009.
- [8] Sanjoy Dey, Prithwish Chakraborty, Bum Chul Kwon, Amit Dhurandhar, Mohamed F. Ghalwash, Fernando J. Suarez Saiz, Kenney Ng, Daby Sow, Kush R. Varshney, and Pablo Meyer. Human-centered explainability for life sciences, healthcare, and medical informatics. *Patterns*, 3(5):100493, 2022.
- [9] Sören Dittmer, Emily J. King, and Peter Maass. Singular values for ReLU layers. *IEEE Trans. Neural Networks Learn. Syst.*, 31(9):3594–3605, 2020.
- [10] Fenglei Fan, Jinjun Xiong, Mengzhou Li, and Ge Wang. On interpretability of artificial neural networks, 2020.

- [11] Ruth Fong, Alexander Mordvintsev, Andrea Vedaldi, and Chris Olah. Interactive similarity overlays. In *VISzAI*, 2021.
- [12] Fred Hohman, Haekyu Park, Caleb Robinson, and Duen Horng (Polo) Chau. Summit: Scaling deep learning interpretability by visualizing activation and attribution summarizations. *IEEE Trans. Vis. Comput. Graph.*, 26(1):1096–1106, 2020.
- [13] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie J. Cai, James Wexler, Fernanda B. Viégas, and Rory Sayres. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2673–2682. PMLR, 2018.
- [14] Thomas Kluyver, Benjamin Ragan-Kelley, Fernando Pérez, Brian E. Granger, Matthias Bussonnier, Jonathan Frederic, Kyle Kelley, Jessica B. Hamrick, Jason Grout, Sylvain Corlay, Paul Ivanov, Damián Avila, Safia Abdalla, Carol Willing, and Jupyter Development Team. Jupyter notebooks - a publishing format for reproducible computational workflows. In Fernando Loizides and Birgit Schmidt, editors, *Positioning and Power in Academic Publishing: Players, Agents and Agendas, 20th International Conference on Electronic Publishing, Göttingen, Germany, June 7-9, 2016*, pages 87–90. IOS Press, 2016.
- [15] Tamara G. Kolda and Brett W. Bader. Tensor decompositions and applications. *SIAM Rev.*, 51(3):455–500, 2009.
- [16] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.
- [17] Yann LeCun, Corinna Cortes, and CJ Burges. MNIST handwritten digit database. *ATT Labs [Online]*, 2, 2010.
- [18] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 5188–5196. IEEE Computer Society, 2015.
- [19] Charles H. Martin and Michael W. Mahoney. Heavy-tailed universality predicts trends in test accuracies for very large pre-trained deep neural networks. In Carlotta Demeniconi and Nitesh V. Chawla, editors, *Proceedings of the 2020 SIAM International Conference on Data Mining, SDM 2020, Cincinnati, Ohio, USA, May 7-9, 2020*, pages 505–513. SIAM, 2020.
- [20] Charles H. Martin and Michael W. Mahoney. Implicit self-regularization in deep neural networks: Evidence from random matrix theory and implications for learning. *Journal of Machine Learning Research*, 22(165):1–73, 2021.
- [21] Charles H. Martin, Tongsu (Serena) Peng, and Michael W. Mahoney. Predicting trends in the quality of state-of-the-art neural networks without access to training or testing data. *Nature Communications*, 12(4122), 2021.
- [22] Anh Mai Nguyen, Jason Yosinski, and Jeff Clune. Multifaceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks, 2016.
- [23] Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. An overview of early vision in InceptionV1. *Distill*, 2020.

- [24] Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 2020.
- [25] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2017.
- [26] Chris Olah, Arvind Satyanarayan, Ian Johnson, Shan Carter, Ludwig Schubert, Katherine Ye, and Alexander Mordvintsev. The building blocks of interpretability. *Distill*, 2018.
- [27] Brenda Praggastis, Dustin Arendt, Cliff Joslyn, Emilie Purvine, Sinan Aksoy, and K Monson. HyperNetX, 2022. Version 1.2.4.
- [28] Brenda Praggastis, Davis Brown, and Madelyn Shapiro. PNNL DeepDataProfiler, 2022. Version 2.0.0.
- [29] Dimitris C. Psychogios and Lyle H. Ungar. SVD-NET: an algorithm that automatically selects network structure. *IEEE Trans. Neural Networks*, 5(3):513–515, 1994.
- [30] Zheng Qin, Zhaoning Zhang, Xiaotao Chen, Changjian Wang, and Yuxing Peng. Fd-mobilenet: Improved mobilenet with a fast downsampling strategy. In *2018 IEEE International Conference on Image Processing, ICIP 2018, Athens, Greece, October 7-10, 2018*, pages 1363–1367. IEEE, 2018.
- [31] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi, editors, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144. ACM, 2016.
- [32] Cynthia Rudin, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong. Interpretable machine learning: Fundamental principles and 10 grand challenges, 2021.
- [33] Andrew M. Saxe, James L. McClelland, and Surya Ganguli. A mathematical theory of semantic development in deep neural networks. *Proceedings of the National Academy of Sciences*, 116(23):11537–11546, 2019.
- [34] Hanie Sedghi, Vineet Gupta, and Philip M. Long. The singular values of convolutional layers. In *International Conference on Learning Representations*, 2019.
- [35] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.
- [36] G. W. Stewart. On the early history of the singular value decomposition. *SIAM Rev.*, 35(4):551–566, 1993.
- [37] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.
- [38] Donglai Wei, Bolei Zhou, Antonio Torralba, and William T. Freeman. Understanding intra-class knowledge inside CNN, 2015.

- [39] Jyh-Jong Wei, Chuang-Jan Chang, Nai-Kuan Chou, and Gwo-Jen Jan. ECG data compression using truncated singular value decomposition. *IEEE Trans. Inf. Technol. Biomed.*, 5(4):290–299, 2001.
- [40] Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5987–5995. IEEE Computer Society, 2017.
- [41] Yunyang Xiong, Hyunwoo J. Kim, and Vikas Singh. Mixed effects neural networks (menets) with applications to gaze estimation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 7743–7752. Computer Vision Foundation / IEEE, 2019.
- [42] Yuichi Yoshida and Takeru Miyato. Spectral norm regularization for improving the generalizability of deep learning, 2017.
- [43] Quanshi Zhang, Ruiming Cao, Feng Shi, Ying Nian Wu, and Song-Chun Zhu. Interpreting CNN knowledge via an explanatory graph. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 4454–4463. AAAI Press, 2018.