

# Multi-aspect Matrix Factorization based Visualization of Convolutional Neural Networks

Uday Singh Saini

Dept. of Computer Science and Engineering  
University of California Riverside  
Email: usain001@ucr.edu

Evangelos E. Papalexakis

Dept. of Computer Science and Engineering  
University of California Riverside  
Email: epapalex@cs.ucr.edu

**Abstract**—What does the space learned by a Convolutional neural network look like? Can we automatically extract high-level concepts that concisely summarize this space in a human-understandable manner? Can we, then, use those concepts for neural network interpretability? In this work, we define a concept to be a co-cluster of data instances (e.g., images), raw features (e.g., pixels), and neuron activations per hidden layer. Such a co-clusters links human-understandable characteristics like data instances and raw features with the architectural elements like neurons of the neural network. In order to extract such multi dimensional concepts, we propose a framework based on regularized and constrained coupled matrix factorization, where the goal of regularization is to force the latent factors to correspond to the sought-after concepts. Our proposed framework is unsupervised since it only requires unlabeled data instances and their activations as an input. Through extensive qualitative and quantitative experimentation on a number of datasets and architectures we show that our proposed framework is able to extract coherent and human-understandable concepts. Finally, we demonstrate the flexibility and versatility of our proposed framework in its ability to be leveraged as an additional tool which complements the existing state-of-the-art neural network interpretability methods.

## I. INTRODUCTION

As the field of deep neural network emerged and rose to being the prominent machine learning paradigm [1] a common issue that has plagued their widespread adoption is the lack of interpretability of these models. As the spectrum of domains where deep learning replaces traditional and orthodox methods expands, and deep-learning percolate to areas of immediate applicability to daily life, understanding what networks do takes on a more central role. Future challenges that machine learning engineers will face, won't just be limited to improving model accuracy, but also debugging [2] and training networks in order to make them conform to ever evolving regulations concerning ethics [3] and privacy [4].

Most literature in the area of explainable AI focuses on providing explanations for pre-trained networks [5], [6], [7], [8], [9], [10], [11], [12], [13], [14]. While some methods focus on constructing models which are inherently interpretable [15], [16], [17], [18], [19], [20]. Our work belongs to the former category and focuses on providing explanation for already trained models, or what is colloquially called post-hoc explanation. Within the strata of post-hoc explanations, there exist multiple evolutionary branches, some focus on interpreting the features [21], and others like [22] interprets

the network by breaking down an input prediction into semantically interpretable components and works like [23] focus on interpreting neurons based on their behaviour when they activate for entities like different textures, colours and images. We focus on unsupervised discovery of concepts learned by the network by trying to cluster the neurons, input features and inputs themselves in the same latent space. The motivation for doing so comes from works like [24] where it has been conjectured that natural images usually lie on a manifold and that a neural network embeds this manifold as a subspace in its feature space. A Coupled Matrix Factorization Framework is the most natural choice to model such multi-modal data and provides the most straightforward framework to learn structures that depend on multi-faceted inputs, which in this case happen to be pixels, neurons and input data. The work that is most aligned with our goals is ACE[6], though it being a supervised framework, renders it tangential to our approach, thus justifying the need for existence of an unsupervised approach like ours. The goal in ACE[6] is to explain the prediction of neural networks not in terms of individual neurons, but rather, by focusing on learning the concepts utilized by the network that are most sensitive for a successful prediction, and learning of such concepts is a supervised process. ACE[6] utilizes existing algorithms or manual annotations to curate a set of concepts, feed it to the network and measure the sensitivity of the network to those concepts using TCAV[25], whereas we try to simultaneously learn meaningful and coherent concepts[6] already present in the activation space of the network. ACEs[6] solution, though elegant relies heavily on domain expert annotators or additional pre-trained tools. ACE[6] relies upon TCAVs[25] ability to back-propagate through the network for each input concept prototype, while we learn our concepts from analyzing forward propagated internal activations for each input and don't assume access to other aspects of the black-box network nor to other pre-existing tools, unlike ACE. Another line of work in TCAV[25] focuses on learning vectors which when measured for their effects on class prediction, align with high-sensitivity directions in the latent space of the network. We also utilize TCAV[25] as a means to validate our approach in section VII and make our case as a viable unsupervised alternative for network interpretation in domains which lack access to rich labeled data and in cases where the only

computation available on the network is a forward propagation through the layers, i.e. evaluation or inference phase.

Our approach aims to find a latent representation for neurons, input features and examples in a common subspace, where co-clustering them aims to elicit meaningful insights about the network. Using such a tri-factor clustering, we can analyze intersections between groups of neurons which fire for different classes, focus on which input features provide a basic structure upon which the model discerns its inputs and analyze an individual example based on their similarity and differences to other examples. We model our problem as a coupled matrix factorization, where the model is constrained to appropriate constraints like non negativity, which aid in interpretability [26] and the possibility of adding regularizations like group sparsity, orthogonality etc. to encode meaningful priors into the model. Given the uniqueness of our approach and a lack of any framework that can be compared head-to-head, We compose a demonstrative case-study in section VII where we analyze the behaviour of a network on a set of images previously unseen, for this, we train with CIFAR-10 and evaluate its characteristics by testing it on CIFAR-100. As there are no established metrics for this problem and a lack of an analogous work to facilitate a comparison, we use TCAV [25], LIME[5], GradCAM[7] and GradCAM++[10] to demonstrate the coherence and validity of our framework. Our *raison d’etre* is to approach the problem of concept discovery in an unsupervised manner, in order to bridge a gap unfulfilled by [6] and [23]. In doing so develop an unsupervised methodology which can seed or supplement other interpretability methods.

## II. RELATED WORK

In our work we aim to interpret a learned model using a set of images which may or may not have been a part of the set of training classes of the network. Our work comes in stark contrast with most existing literature, since the goal in our work is not to evaluate the network on just a feature or sample basis as in works like [27], [28], [29], [30], [7], [8], [9], [10]. Works such as [27] visualize a network based on images that maximize the activation of hidden units and works like [31] use back-propagation to generate salient features of an image. Works like [15], [16], [18] focus on explaining a network by proposing a new framework where the network is forced to learn concepts and demonstrate their relevance towards a prediction. This framework relies on prior constraints and encoding for what is thought to be a concept. In [22], [9] the focus is on explaining each prediction made by the network by decomposing the activations of a layer in the network into a basis of pre-defined concepts, where each explanation a weighted sum of these concepts, where the weights determine the impact each concept has towards prediction. Our work has similarities in philosophy with the previous works, but unlike SeNN[15] we don’t focus on learning an interpretable model, instead we focus on unsupervised explanation of an already trained network. And unlike [23] we do not have a pre-made notion of concepts, instead we let the model learn underlying

concepts based on the set of examples fed in the analysis. Our work is orthogonal to both in different aspects and this way our approach is application agnostic.

Recent work on Network Dissection [32] tries to provide a framework where they can tie up a neuron in the network to a particular concept for which the neuron activates. These concepts can be elements like colour, texture. They accomplish this through a range of curated and labeled semantic concepts whereas our work is unsupervised. Another work which relies on interpreting the network through the lens of abstract concepts is TCAV[33]. This work tries to provide an interpretation into network’s workings in terms of human interpretable concepts. Like our work, they too rely on the internal representation of the network to determine the network’s behaviour, but unlike us they utilize manual/pre-defined concepts and test the network’s sensitivity towards it. The work presented in SVCCA[34] uses a variant of canonical co-relational analysis and focuses on learning the complexity of the representations learned by the network to determine the dynamics of learning, our work differs as we use the structure of the learned representation as a guideline for our factorization framework and don’t comment on the inherent complexity.

In ACE[6] the authors seek to automatically discover concepts learned by the network which are of high predictive value, as measured by their TCAV score [33]. As described earlier the approach relies on pre-trained tools to process inputs and on being able to forward and back-propagate over inputs given to the network.

In Figure 1 we contextualize our work to other works in the area, most of which are tangential to our approach. While the axioms of interpretable machine learning are an ever evolving set of principles, we do so on a few features that help us highlight the differences between our work and its closest Neighbours in this space. Our work is the only unsupervised method in this space of model interpretability which helps us discover concepts learned by the network in terms of the examples clustered by the network. ACE[6], SeNN[15] learn concepts but either by utilizing explicit supervision or by employing pre-existing trained models, whereas works like [23] require detailed human labeling of neurons and image pixels and patches, thus making the process slow and sluggish for adaptation to a new domain. LIME[5] on the other hand tries to visualize a linear decision boundary across an input, which we also approximate by the input’s K-Nearest Neighbours, refer to Section IV-A3, but unlike our work LIME cannot discover abstract concepts learned by the network without significant modifications.

## III. PROPOSED METHOD

In this section we begin by outlining the motivation for our methodology, we then proceed to outline the implementation schema and optimization problem for our model. Subsequently we present the model details and lay down the groundwork for evaluation protocols suited for this method.

Fig. 1: Relevant Work Comparison

Model Features	ACE[6]	[7],[5],[10]	[15],[20]	[23],[25]	Our work
Post-Hoc Interpretability	✓	✓	✗	✓	✓
Unsupervised	✗	✗	Partial	✗	✓
Multi-Aspect Analysis	✗	✗	✗	✗	✓
Analysis of Representation Space	✗	Inputs Only	✗	Inputs Only	✓

### A. Motivation

Our goal is to visualize the latent representation space learned by a Neural Network by comparing and contrasting the behaviour of the network on different types of inputs. We want to accomplish this in a framework where we can explain the learned concepts in terms of the inputs that are used to probe the network. In doing so we can assess the generalization ability of the network, both to familiar and unseen datasets, thus providing insights to human evaluators about the health of the trained network and its suitability to a particular domain. This is possible because there are no restrictions on what qualifies as a legitimate dataset for evaluating network behaviour, thus in theory, we can evaluate a network on a dataset which is different from its training dataset and assess the suitability of the architecture to learn atomic concepts (which may be valid across domains) from the training data instead of learning its idiosyncrasies.

### B. Proposed Model

Given these goals in mind, we lay down the model principles behind our objectives. Our approach is a method that relies on a coupled factorization framework where we compute embeddings of features like pixels, test examples and individual neurons in a shared latent space. Our method relies on only having access to activations of internal layers of a network for a given input. Additionally, for ease of modeling, we assume that these activations are non-negative in nature. In case these assumptions don't hold for the network, we can relax the Non-Negativity Constraints on some Factor matrices of each factorization with the use of semi-NMF[35]. With these principles as its building block, our model does not introduce any external learning constraints while training the network, thus lending it universality. We probe various layers of a network with a set of test examples, and for each test example, we store the network's response across all observed layers. We do so with an aim to breakdown the process of interpretability into a process of finding common local structures across various evaluation examples, where each dimension in the latent representation is constrained to capture a latent semantic concept. Thus, through the lens of our model we can hopefully view individual concepts as a ranked over evaluation examples. In the following subsections we describe model construction and provide mathematical details of implementation.

1) *Model Construction*: For our analysis we need construct a set of matrices where each matrix  $A_j$  in the set is a matrix  $\in \mathbb{R}_+^{a_j \times N}$ , where  $a_j$  is the number of neurons in layer  $j$  of the network, and  $N$  is the number of examples on which our analysis is conducted. Each column  $k$  of matrix  $A_j$ , is a

vectorized activation of layer  $j$  of the network for a given test sample  $k$ , denoted using NumPy notation by  $A_j[:, k]$ . Along similar lines we construct another set of matrices where each Matrix  $D_i \in \mathbb{R}_+^{S_i \times N}$  where  $S_i$  is number of pixels in the  $i^{th}$  channel of input images. On the same lines as before, each column  $k$  of matrix  $D_i$ , is the  $k^{th}$  test sample's  $i^{th}$  channel vectorized.

2) *Model*: The objective function for our proposed method is as follows:

$$L = \sum_{i=0}^{C-1} \|D_i - P_i F\|_F^2 + \sum_{j=0}^{L-1} \|A_j - O_j F\|_F^2 + \sum_{i=0}^{C-1} \lambda_P \|P_i\|_p^2 + \sum_{j=0}^{L-1} \lambda_O \|O_j\|_p^2 + \lambda_F \|F\|_p^2 \text{ s.t. } P_i, O_j, F, \in \mathbb{R}_+^{S_i \times d}, \mathbb{R}_+^{a_j \times d}, \mathbb{R}_+^{d \times N}$$

*optional constraints* -  $\|P[:, i]\|_2 = 1, \|O[:, i]\|_2 = 1 \forall i, j$  (1)

In Equation 1, C is the number of channels in input data, L is the number of layers of the network being analyzed - as we can select the non-negative layers we want to analyze and are not obligated to include all the layers of any architecture.  $p$  is usually 2 for 2-Norm regularization although for the purposes of some experiments we instead normalize the column norms of the Pixel and Neural Factor matrices to unity. We do so by normalizing the respective matrices to unit column norm after obtaining the new iterates based on the update steps in Equation 2.

For each matrix  $D_i$  in Equation 1, its  $k^{th}$  column is the input data's channel  $i$  vectorized as input. Thus for instance, for a 3-channel image, with image number  $j$  of the test set,  $D_0[:, j]$  is the vectorized 0<sup>th</sup> channel of the  $j^{th}$  image and so on.

Each  $P_i$  in the first term of the summation in equation 1 is a latent representation matrix for each pixel in channel  $i$ . That is, Each row of  $P_i$ , for instance  $P_i[k, :]$  is the latent representation of channel  $i$ 's  $k^{th}$  pixel in the latent space.

For each matrix  $A_j$  in Equation 1, its  $k^{th}$  column is the activation of layer  $j$  of the network for  $k^{th}$  test input. Thus for instance, image number  $j$  of the test set,  $A_0[:, j]$  is the activation of layer 0 of the network for image  $j$ . Please note that the higher index of the layer, the deeper we are in the network, though index  $j$  doesn't necessarily correspond one-to-one with layers in the network.

As each matrix  $A_j$  encodes the activity of neurons of layer  $j$  for test inputs. Therefore, each  $O_j$  in the factorization encodes the latent representation of neurons of layer  $j$  in its rows. That is,  $O_j[k, :]$  is the latent representation of  $k^{th}$  neuron of layer  $j$ . Similarly, the matrix  $F$  encodes in its columns, the latent

representation of each test example fed to the network. That is,  $F[:, k]$  is a  $d$ -dimensional latent representation of test sample  $k$ . Each factor matrix in the objective function obeys non-negativity constraints, and we use multiplicative update rules as described in [36] to solve for the factor matrices.

Update Steps for solving the factor matrices in Equation 1 are as presented in the following Equation 2 :-

$$\begin{aligned}
 F &\leftarrow F * \frac{\sum_i P_i^T D_i + \sum_j O_j^T A_j}{\sum_i P_i^T P_i F + \sum_j O_j^T O_j F + \lambda_F F} \\
 P_i &\leftarrow P_i * \frac{D_i F^T}{P_i F F^T + \lambda_P P_i} \\
 O_j &\leftarrow O_j * \frac{A_j F^T}{O_j F F^T + \lambda_O O_j}
 \end{aligned} \tag{2}$$

After each iteration of multiplicative updates for  $P_i$  and  $O_j$ , if need be, we normalize their columns to unit squared norm.

3) *Model Intuition*: We now provide some intuition for our modeling choices. Our goal is to identify hidden patterns or concepts that the network learns. To achieve this our model clusters the test examples, neurons and pixels in the same inner product space. We achieve this clustering by incorporating a coupled non-negative matrix factorization framework. In our learned representation of these 3 types of objects, a high value along a latent dimension indicates that a particular latent concept participates in explaining the behaviour of the object. By constraining the model to adhere to a non negative framework, we encourage an interpretable sum-of-concepts based representation[26].

Further elaborating on the learned factor matrices, Each column  $j$  of Matrix  $P_i \in \mathbb{R}_+^{S_i \times d}$  is the activation of the pixels of channel  $i$  for the concept discovered in latent factor  $j$ . Collecting such information over all input channels  $i$  for a given  $j$  in the respective factor matrices we can uncover the average activation of pixels across channels for a given concept. This representation can be thought of as a channel-wise mask over features in the input, analogous to [5], [12], [37], [38], among others, but instead we discover a latent concept level mask as opposed to an input level mask. Matrix  $F \in \mathbb{R}_+^{d \times N}$  is the input representation matrix where each column  $k$  of  $F$  is a vector in  $\mathbb{R}_+^d$  representing the  $k^{th}$  example in the same latent space as Pixels and Neurons. For any input  $k$ , A high value along any component  $j$  of its  $d$ -dimensional representation indicates a high affinity of this input towards the latent concept encoded in the dimension  $j$ .  $P_0[:, j]$ - $P_2[:, j]$  together help us visualize the pixel activation mask for this latent concept  $j$  as discussed earlier. Collecting all the highest affinity inputs for each latent factor, we obtain a visual approximation of the concept learned in this latent dimension. Exploiting the ability of a Multi-Aspect Factorization framework to rank inputs to form concepts is how we propose to solve the problem of concept discovery. Given the unsupervised nature of this model, it extremely well suited for concept discovery for neural networks, akin to a similar role played by ACE [6] for TCAV [25]. Matrices  $O_j$ 's embed neurons of a layer

$j$  in the same latent space as inputs and features and help us visualize which neurons in a layer activate for which concept, we do this by demonstrating the similarity of latent concepts when measured w.r.t. neurons of a layer. We can also look at the behaviour of neurons across layers by observing the cohesiveness of latent space as the neurons go deeper in the network.

#### IV. EXPERIMENTAL EVALUATION

In the following subsections we will present the analysis of the latent space learned by a ResNet-18 [39] when trained on CIFAR-100 images [40], The Accuracy of the trained network is around 74% on top-1 classification. Our analysis touches all the modalities captured by our model, i.e. Analysis of Pixels, Analysis of Neurons and Analysis of Examples. We present this analysis in 3 subsections for a given network. We also released the code<sup>1</sup> for verification.

##### A. Analysis of A ResNet-18 on CIFAR-100 Dataset:

In the following subsections we analyze the behaviour of a ResNet-18<sup>2</sup> trained and analyzed on CIFAR-100. Each subsection represents a modality of analysis, namely, inputs, Neurons, and input features or pixels themselves.

1) *Analysis of Input representations*: In this section we present the analysis of representations learned in the input representation Matrix  $F$  in Table I. We begin by considering each latent dimension  $i$ , which we will represent as a row in Table I. We compute the total class-wise activation score of inputs in the row  $F[i, :]$  and present the top activated classes along that latent dimension in the first column of the corresponding row, top -20 images which had the highest affinity in this latent dimension and most activated super-classes in column 2 and 3 respectively. The Images in column 2 of a row are presented in descending order of their affinity for a particular latent dimensions. The motivation behind analyzing super class labels is to validate our assertion that each latent factor captures an abstract concept that is predominantly present in the member images. We reiterate that these super class labels were not used in training of the network but only used as a means to assign a pseudonym to each concept or latent factor, the validity of which can be verified by looking at the topmost activated images and the group of top most activate classes and super classes in Table I. The total number of Latent Dimensions or Latent concepts for this experiments was 20, but some latent factors are omitted from Table I for brevity.

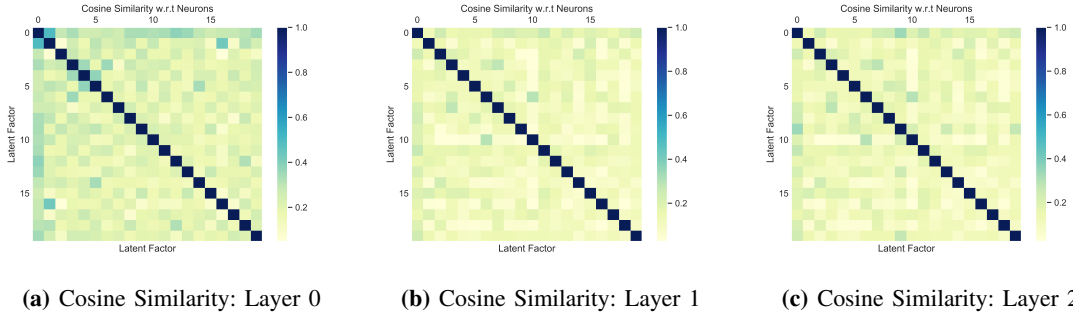
2) *Layer-wise Analysis of Neuron representations*: In this section we try to quantify the behaviour of neurons as a cluster and across layers. We utilize the neuron embedding matrix for a given layer  $j$ , as denoted by  $O_j \in \mathbb{R}_+^{N_j \times d}$ , where  $N_j$  is the number of neurons in layer  $j$ , whereas  $d$  is the number of latent factors in the factorization. Next we compute pairwise cosine similarity between the columns of a matrix  $O_j$  and we

<sup>1</sup>Code and Data: Link to Code and Data

<sup>2</sup> A Pytorch Code Repository for ResNets

**TABLE I: Matrix- $F$  Latent Factor Analysis For ResNet-18:** Each row of the table corresponds to a row in  $F$  and visualizes a latent dimension of the factorization. The first column shows a few of classes whose images have the highest alignment in this latent direction. The second column shows top 20 images that have the highest affinity in descending order. The third column shows the top most superclass membership of images activated along this direction. Such a superclass label helps us assign a pseudonym to the concept collectively represented by highest affinity images along each latent direction.

Factor: Top Classes	Top Images	Top 1-2 Super Class
1: kangaroo,beaver,bear		large omnivores and herbivores
2: mountain,castle,bridge		large man made outdoor things
3: willow, maple, pine, oak		trees
4: shark,dolphin,whale		fish and aquatic mammals
5: bee,beetle,spider		insects
6: tulip,rose,poppy		flowers
10: boy,woman,girl,baby		people
11: aquarium fish,trout		fish
14: sea,plain,cloud,mountain		large natural outdoor scenes
16: apple,orange,pear		fruit and vegetables

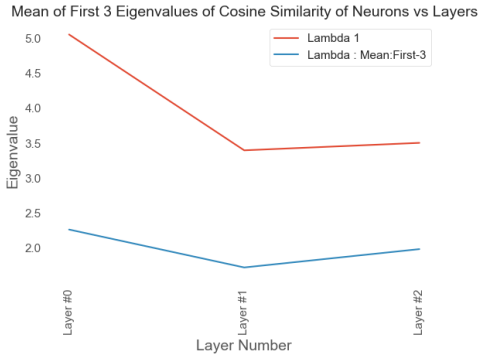


**Fig. 2:** Plots of Cosine Similarity of Latent Factors in Layers 0,1,2 of ResNet-18. This highlights the layerwise learning dynamics of the network and helps us visualize with concepts and classes occupy similar neural regions in a given layer of a network and how they evolve as we go deeper into the network. In fact, we observe that as we go deeper into the network, the similarity becomes diagonal, showing higher separation of the latent concepts

do this  $\forall j$  as shown in Figure 2a - Figure 2c. Here Layer 0,1,2 refer to 3 layers analyzed in the ResNet-18 in increasing order of depth and are not necessarily the first,second and third layers of the network. In these plots a high value at any entry  $(i, j)$  indicates a higher overlap between the number of neurons which fire for inputs belonging in the 2 super classes best approximated by latent factor  $i$  and latent factor  $j$ . As indicated in Figures 2a - 2c the activations tend to be more intra-superclass, a result similar in nature to one observed by SVCCA[34], i.e. more concentrated along the diagonal of the Similarity Matrix as we go deeper down the layers. This is also borne out by the eigen values of these Similarity Matrices, as the matrices tend to get closer to Identity, the lower the mean of first-K eigen-values as shown in 3a.

3) *Co-Analysis of Pixels and Inputs:* In this section we analyze the pixel space along with inputs. The Matrices  $P_i$ 's  $\in \mathbb{R}_+^{S_i \times d}$  hold the input representation of pixels in the input channel  $i$  where  $S_i$  is the number of Pixels in Input Channel- $i$ , or the vectorized size of the channel. Each column of a matrix  $P_i$  represents a feature activation score of all the pixels in channel  $i$  for the given latent factor. Therefore for e.g.,

by collecting information from column 2 of  $P_0, P_1$  and  $P_2$  and resizing them appropriately we get an average pattern of activation across the pixel space for all the images that belong to Latent Factor 2, as shown in Figure 4a, and for Latent-Factor-6 in Figure 5a. This functionality is very similar to LIME [5], GradCAM[7] and GradCAM++[10] but instead of individual images we operate on pixel representations which represent learned concepts. We then take these Latent-Images, and create a mask where we assign a value of 1 at a pixel location if it's activation value is above the median activation value for the Latent Image and 0 otherwise and overlay it with the topmost images of the Latent Factor as found in our analysis of Matrix- $F$  in Table I. To continue the analysis further we also take around 30 Nearest Neighbours of the input image as determined by the Latent Space of Matrix- $F$  and give a distribution over the Latent Concepts of those Neighbouring images, thereby helping us achieve interpretability on an input-by-input basis by being able to say that a given image is close to another, in terms of their concept distribution. Next, via 2 examples we present a per example case study of interpretability analysis possible by the use of this model.

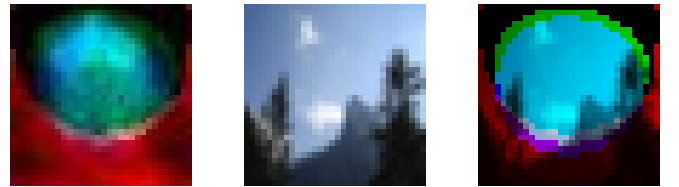


(a) Eigenvalues of Similarity Matrices of ResNet-18

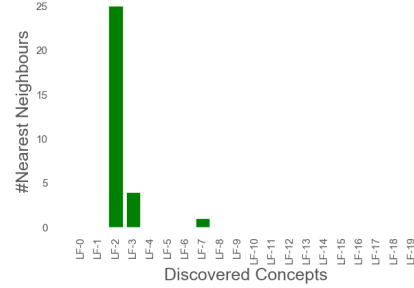
**Fig. 3:** Plots of Eigenvalues of Similarity Matrix for ResNet-18 . This plot shows increasing independence of learned latent concepts w.r.t. neurons as we go deeper in the non-classification layers of the network. The closer the a matrix is to Identity the closer the average of it's eigen values is to 1 and vice versa. The last layer in each of the 2 figures is the output of pre log softmax of the network, which is usually a much lower dimensional space than the previous layers

In Figures 4a - 4d For Latent Factor-2 we present the Latent Representations of Pixels, The topmost Image in Latent Factor-2, The top 50% activated pixels super imposed on the original image, and the Latent Concept Distribution of top-30 Nearest Neighbours of the image 4b, respectively. As noted previously in Table I, Latent Factor 2 Represents classes like mountain, bridge, castles, skyscrapers etc, leading to its topmost super class being "large man-made outdoor things". On average, the most activated pixels for images belonging to this superclass tend to be blue pixels towards the top, green towards the middle and red towards the bottom. And the set of top-30 nearest-neighbours for this particular image of a Mountain also has members belonging to Latent Factor 3 and 7, 2 concepts which have a high affinity for inputs belonging to super class of trees. In Figures 5a - 5d we do the same for Latent Factor-6. Latent Factor-6 represents entities like tulips, rose, poppy - all belonging to super class "flowers". As we observe, the most activated pixels tend to be around the body of the tulip, encapsulating the petals, an appropriate evaluation for entities belonging to the super class "flowers". The set of top-30 nearest-neighbours for this particular image of a Tulip also has members belonging to Latent Factor-16, concepts which have a high affinity for inputs belonging to super class of fruits and vegetables. Which is semantically a plausible conclusion.

A note on hyper-parameters : Hyper-parameters values for  $\lambda_P$ ,  $\lambda_O$ ,  $\lambda_F$  for the aforementioned experiment were 0.0001, 0.0001, 1 respectively. We also show the robustness of output to variations in  $\lambda_F$  by varying the value of the hyper-parameter and comparing the similarity of different outputs in the resulting matrix  $F$  obtained via different hyper-parameter values. For 2 different values of  $\lambda_F$ , say  $\lambda_1$  and  $\lambda_2$ , we obtain the respective factor matrices  $F_1$  and  $F_2$ . Given that each  $F$  is a matrix that is a column wise collection of latent

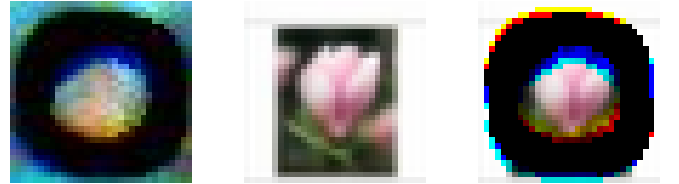


(a) Latent-Factor:2 (b) Image:Mountain (c) Filtered Image



(d) Top-30 Nearest Neighbours of this Image

**Fig. 4:** Analysis of Topmost Image from Latent-Factor:2



(a) Latent-Factor:6 (b) Image: Tulip (c) Filtered Image



(d) Top-30 Nearest Neighbours of this Image

**Fig. 5:** Analysis of Topmost Image from Latent-Factor:6

representation of input samples, we compute the dot product kernels  $F_1^T F_1$ ,  $F_2^T F_2$  and compute the similarity of these kernels using Centered-Kernel-Alignment [41]. We present the results for all pairwise combinations of various values of  $\lambda_F$  ranging from 0.001 to 1 in Table II. As we can observe in Table II, the relationship between input examples is fairly stable regardless of the value of hyper parameters as shown by the high similarity of dot-product kernels over a wide range of  $\lambda_F$ . A note on running time: The running time of the experimental setup once given a trained neural network, where we analyze 3 separate layers over 3000 examples is less than 15 minutes.

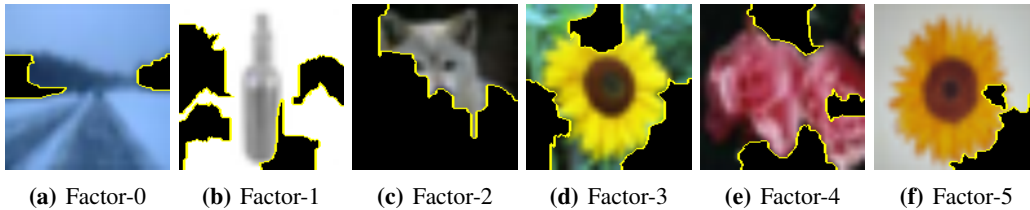


Fig. 6: Explanations given by LIME for topmost images of Latent Factors 0,1,2,3,4,5 of a ResNet-18 trained on 128x128 CIFAR-100 Images.

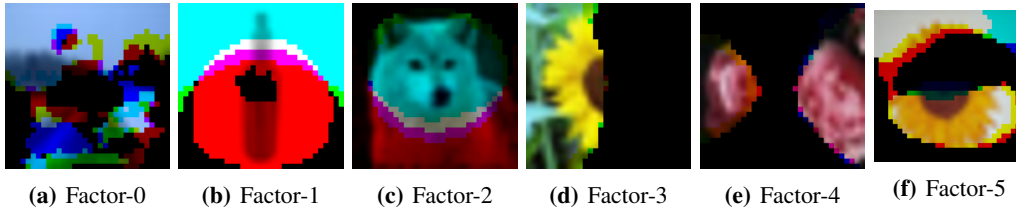


Fig. 7: Superimposition of latent pixel representation obtained by our method on the topmost images of Latent Factors 0,1,2,3,4,5 of the same ResNet-18 trained on 128x128 CIFAR-100 Images.

$\lambda_F$	1	0.1	0.01	0.001
1	1	0.96	0.96	0.95
0.1	0.96	1	0.96	0.95
0.01	0.96	0.96	1	0.96
0.001	0.95	0.95	0.96	1

TABLE II: Hyperparameter Similarity : CKA Similarity Scores for different configurations of  $\lambda_F$ .  $\lambda_P, \lambda_O$  are set to 0.0001

## V. EXPERIMENTAL COMPARISONS WITH LIME

In this section we present comparisons of pixel space representations obtained by our method with explanations obtained by LIME [5]. We couldn't get LIME to work with 32x32 images used for analysis in section IV, so we train a ResNet-18 on 128x128 CIFAR-100 images and run our analysis to obtain top images for every latent factors following the same methodology as described in section IV. We then analyze the topmost image of a latent factor with LIME and obtain its explanation for a positive prediction and compare it with the its masked counter part, where the latent factor masked image is obtained using the same protocols as described in section IV. The resulting explanations obtained via LIME for 6 images, each being the topmost image of its latent factor is shown in Figure 6a-Figure 6f. Corresponding to each explanation by LIME, From Figure 7a-Figure 7f, we show the output of our latent pixel analysis on the same corresponding topmost images of each latent factor. In contrast to the approach taken by LIME where the the algorithm receives super-pixels using pre-existing algorithm, the latent pixel representations yielded by our method are learned from the data in an unsupervised manner and the learned pixel representations are over each channel of the image.

## VI. EXPERIMENTAL COMPARISONS WITH GRADCAM AND GRADCAM++

Continuing our qualitative comparisons further, we next juxtapose our work with a family of Class Activation Mapping based methods, namely GradCAM[7] and its extension

GradCAM++[10]<sup>3</sup>. We perform this analysis on a ResNet-18 trained on CIFAR-10, we then analyze the network using our approach and take the topmost images from each latent factor , 10 in case of CIFAR-10, and overlay it top 50% of the most activated pixel locations as determined by analyzing the matrices  $P_i$ 's. For the purposes of analyzing GradCAM and GradCAM++ we take the output of the final convolution layer of the network and seek explanations w.r.t. to true label of the image. Figure 8a-Figure 8j are the images which will be analyzed by all the 3 methods. From Figure 9a-Figure 9j we present the analysis of our model along the lines of SectionIV-A3, where we mask the most active image along a latent factor with the top 50% most active pixels for that latent factor. In our analysis we notice that for same factors, the most activated pixels indeed capture the object of the image, but for some latent factors like in Figure 9e, more emphasis might be on the background.

We would like to re-iterate that our method's explanation is on a concept level, an aggregate of the most commonly activated pixels for this concept. Unlike CAM based models, we don't have access to the gradients for each image w.r.t. different class labels and thereby don't proclaim to solve the same problem but still try to offer an insight into the flexibility of our multi-modal factorization based approach to offer similar insights to more supervised and invasive methods like GradCAM[7] and GradCAM++[10]. In Figure 10 and Figure 11 we show the results of GradCAM and GradCAM++ on the same images. As can be noted by observation, the performance of these methods are very similar, partly because they rely on first and second order gradients for each input w.r.t. an output class. Compared to GradCAM and GradCAM++ we observe that our method does mostly highlight objects of the said class label e.g. Figure 9c, Figure 9d, however for some factors like Figure 9e and Figure 9f, the highlighted features of the image

<sup>3</sup>Code Here : Pytorch Repo for GradCAM and other methods

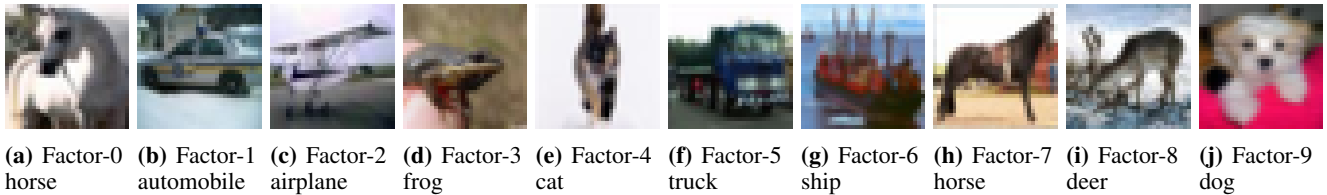


Fig. 8: Topmost images of Latent Factors 0-9, where each image’s label is under the Factor Number. Net : ResNet-18 on CIFAR-10.



Fig. 9: Explanations given by our method for topmost images of Latent Factors 0-9 of a ResNet-18 trained on CIFAR-10 Images.

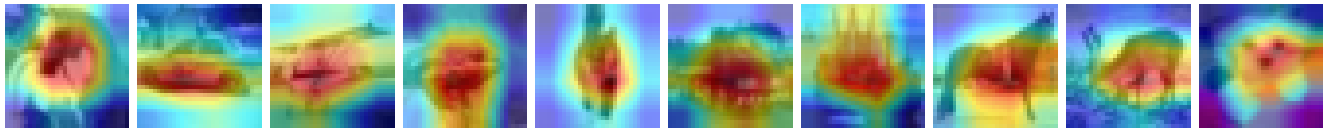


Fig. 10: Explanations given by GradCAM for topmost images of Latent Factors 0-9 of a ResNet-18 trained on CIFAR-10 Images.

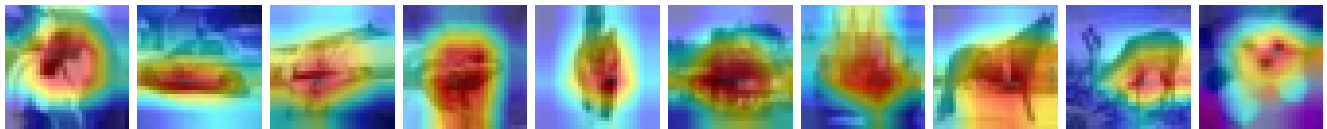


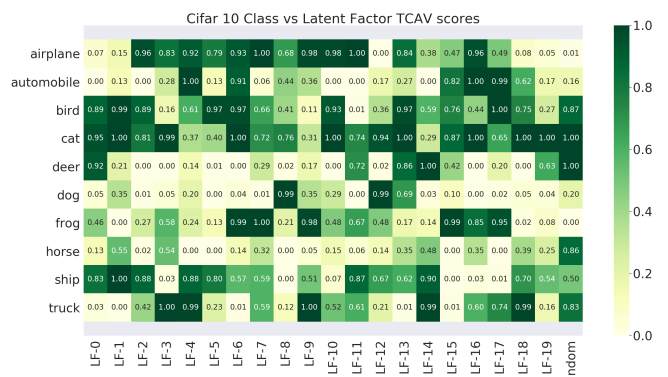
Fig. 11: Explanations given by GradCAM++ for topmost images of Latent Factors 0-9 of a ResNet-18 trained on CIFAR-10 Images.

tend to belong to the background. An observation of a similar nature has also been made in [42].

## VII. CASE STUDY WITH TCAV AND MODEL EXTENSIONS AND EXPERIMENTS

In this section we present some possible extensions to the model in order to demonstrate its flexibility to work as a framework that can embed 3-Modes, namely, Neurons, Features and Data points, all in the same space, while incorporating constraints. We then present an experiment to demonstrate flexibility of this framework to analyze neural networks. We take a network trained on CIFAR-10, and analyze its behaviour on CIFAR-100, the goal in doing so is to demonstrate the ability of this framework to be adept at discovering hidden structures in the latent representations of the neural network. The extensions we propose allow the model to imbibe priors in the form of structured regularizations like Group Sparsity, based on [43] and Total Variation Norms, based on [44], [45], [46]. The motivation behind both sets of extensions is to encourage disjoint usage of the latent factors by examples that are fundamentally different in nature. This helps avoid any diffusion of concept across an excessive number of latent factors. Group Sparsity Constraint is useful when the number of different groups are known and Total Variation (TV) norm

penalty coupled with a soft orthogonality regularization is useful in case no such information is available.



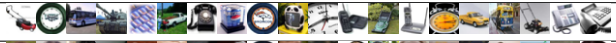






(a) TCAV scores for Group Sparsity based model - Inputs Included  
Fig. 12: TCAV scores for Group Sparsity based Models

In the following experiment, we use a ResNet-18 that has been Trained on CIFAR-10 but we try to analyze the concept space learned by the network by probing it with CIFAR-100 inputs. This setup aims to find how a network trained on CIFAR-10, provides a meaningful clustering over classes, if



TABLE III: Latent Factor Analysis Group Sparsity - Abridged

Factor: Classes	Top Images	Latent Image
2: dolphin,whale,plain		
4: pickup truck,lawn mower, telephone		
8: girl,lion,boy		
11: plain,mountain,sea		

any, in CIFAR-100, thereby demonstrating its ability to truly generalize. We do this with both regularization schemes, and in-order to validate and attest the results for generalization, we take the latent concepts learned by the model over CIFAR-100 and evaluate their TCAV[25] scores<sup>4</sup> for each combination of latent concept and input class of CIFAR-10. This experiment also serves to accomplish in principle the ability of our method to synergize with existing interpretability methods like TCAV by providing them with seed concepts and help provide a solution to the issue of cold starts and generalization to domains with limited labeled data. We present the results of this experiment, first for the coupled factorization model with Group Sparsity Regularization in Table III, and its corresponding TCAV scores in Figure 12a. Just as in IV-A1, We display some latent factors out of 20 for brevity. We provide additional implementation details and data on Group Sparsity analysis in Table IV, Total Variation Analysis in Table V and their corresponding TCAV analysis in Figure 13a and Figure 13b respectively. All included in section A to ensure verifiability and encourage reproducibility.

In Table III, Upon gleaning the latent-factors for CIFAR-100 classes deemed similar by the pre-trained ResNet-18, we observe that the network clusters CIFAR-100 classes like dolphin, whale and plain - as seen in latent factor 2. Looking further into the CIFAR-10 classes that had the highest TCAV scores for images in latent factor 2, we observe that CIFAR-10 classes like ship, airplane and bird had a high influence in the network’s output for images in latent factor 2. Such a correspondence does fall in line with intuition as images of ships, airplanes and birds tend to be against a blue backdrop. Such a feature is common in images of dolphins and whales as they live in the ocean and also to images of outdoor empty plains which are normally set against the backdrop of the sky. Pursuing this investigation further we come across latent factor 4 in Table III where the network clusters CIFAR-100 classes like Pickup truck, Lawn mower as similar. The highest activated Classes in CIFAR-10 for latent factor 4 happen to be Automobile and Truck, which further lends credence to the belief that the network tries to activate its pathways that learn features present in vehicles. Next we perform an analogous but brief analysis on the concepts learned via the factorization model involving TV Regularization with orthogonality, the Results of which are in Table V and the corresponding TCAV scores in Figure 13b. Having a look at the first row in Table V, i.e. Latent Factor 2, we learn that the network deems CIFAR-

100 classes Train, Bridge and Castle to be similar in nature and the most prominent CIFAR-10 classes that have the highest TCAV for Latent Factor 2 happen to be Truck, Airplane and Ship. All Vehicular classes in CIFAR-10. Latent Factor 9 displays similar behaviour. Next we look at Latent Factor 5, where the networks clusters CIFAR-100 classes of large animals like Cattle, Elephant and Camel together. A look at the maximally scored TCAV classes from CIFAR-10 demonstrates that network pathways related to CIFAR-10 animal classes like Cat, Deer and Horse have their highest Activations for Latent Factor 5.

## VIII. CONCLUSIONS

In this paper, we introduced an unsupervised framework based on coupled matrix factorization for exploration of the representations learned by a CNN. Our proposed method is the first such framework to allow for *joint* exploration of the representations that a CNN has learned across features (pixels), activations, and data instances. This is in stark contrast to existing state-of-the-art works, which are typically restricted to one of those three modalities, as shown in Fig. 1. As a result, our proposed framework offers maximum flexibility and bridges the gap between existing works. Case in point, in this paper, we demonstrate a number of applications of our framework drawing parallels to what existing work can offer compared to our results, including the extraction of instance-based interpretable concepts (Sec. IV-A1), and based on those concepts we provide insights on the the behavior of neurons in different layers (Sec. IV-A2), and instance-level pixel-based insights (Sec. IV-A3). In future work, we will investigate the adaptation to our framework to different architectures.

## IX. ACKNOWLEDGEMENTS

Research was supported by the National Science Foundation under CAREER grant no. IIS 2046086 and grant no. 1901379, and a CISCO Faculty Research Award. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding parties. The authors would like to thank NVIDIA for a GPU grant which facilitated computations in this work. Finally, the authors would like to thank Madhav Ram Nimishakavi and Tushar Nagarajan for their valuable inputs and anonymous reviewers for their feedback.

## REFERENCES

- [1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems* 25, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: <http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf>

<sup>4</sup>PyTorch Implementation of TCAV

- [2] S. Tan, R. Caruana, G. Hooker, and Y. Lou, “Distill-and-compare,” *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, Feb 2018. [Online]. Available: <http://dx.doi.org/10.1145/3278721.3278725>
- [3] D. K. Mulligan and K. A. Bamberger, “Saving governance-by-design,” *Calif. L. Rev. California Law Review*, vol. 106, no. IR, p. 697. [Online]. Available: <http://lawcat.berkeley.edu/record/1128572>
- [4] D. K. Mulligan, C. Koopman, and N. Doty, “Privacy is an essentially contested concept: a multi-dimensional analytic for mapping privacy,” *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 374, no. 2083, p. 20160118, 2016. [Online]. Available: <https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2016.0118>
- [5] M. T. Ribeiro, S. Singh, and C. Guestrin, ““why should I trust you?”: Explaining the predictions of any classifier,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, 2016, pp. 1135–1144.
- [6] A. Ghorbani, J. Wexler, J. Zou, and B. Kim, “Towards automatic concept-based explanations,” 2019.
- [7] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-CAM: Visual explanations from deep networks via gradient-based localization,” *International Journal of Computer Vision*, vol. 128, no. 2, pp. 336–359, oct 2019. [Online]. Available: <https://doi.org/10.1007%2Fs11263-019-01228-7>
- [8] H. Wang, Z. Wang, M. Du, F. Yang, Z. Zhang, S. Ding, P. Mardziel, and X. Hu, “Score-cam: Score-weighted visual explanations for convolutional neural networks,” 2019. [Online]. Available: <https://arxiv.org/abs/1910.01279>
- [9] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Learning deep features for discriminative localization,” 2015. [Online]. Available: <https://arxiv.org/abs/1512.04150>
- [10] A. Chattopadhyay, A. Sarkar, P. Howlader, and V. N. Balasubramanian, “Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks,” in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018, pp. 839–847.
- [11] B. Zhou, Y. Sun, D. Bau, and A. Torralba, “Interpretable basis decomposition for visual explanation,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.
- [12] D. Omeiza, S. Speakman, C. Cintas, and K. Weldermariam, “Smooth grad-cam++: An enhanced inference level visualization technique for deep convolutional neural network models,” 2019. [Online]. Available: <https://arxiv.org/abs/1908.01224>
- [13] S. Desai and H. G. Ramaswamy, “Ablation-cam: Visual explanations for deep convolutional network via gradient-free localization,” in *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020, pp. 972–980.
- [14] P. C.-H. Lam, L. Chu, M. Torgonskiy, J. Pei, Y. Zhang, and L. Wang, “Finding representative interpretations on convolutional neural networks,” 2021. [Online]. Available: <https://arxiv.org/abs/2108.06384>
- [15] D. Alvarez-Melis and T. S. Jaakkola, “Towards robust interpretability with self-explaining neural networks,” 2018.
- [16] P. W. Koh, T. Nguyen, Y. S. Tang, S. Mussmann, E. Pierson, B. Kim, and P. Liang, “Concept bottleneck models,” 2020. [Online]. Available: <https://arxiv.org/abs/2007.04612>
- [17] C.-K. Yeh, B. Kim, S. O. Arik, C.-L. Li, T. Pfister, and P. Ravikummar, “On completeness-aware concept-based explanations in deep neural networks,” 2019. [Online]. Available: <https://arxiv.org/abs/1910.07969>
- [18] Z. Chen, Y. Bei, and C. Rudin, “Concept whitening for interpretable image recognition,” *Nature Machine Intelligence*, vol. 2, no. 12, pp. 772–782, dec 2020. [Online]. Available: <https://doi.org/10.1038%2Fs42256-020-00265-z>
- [19] A. B. J. S. C. R. Chaofan Chen, Oscar Li, “This looks like that: Deep learning for interpretable image recognition,” in *Proceedings of Neural Information Processing Systems (NeurIPS)*, 2019.
- [20] O. Li, H. Liu, C. Chen, and C. Rudin, “Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions,” in *Proceedings of AAAI*, 2018.
- [21] I. G. M. H. B. K. Julius Adebayo, Justin Gilmer, “Sanity checks for saliency maps,” in *advances in neural information processing systems*, 2018.
- [22] B. Zhou, Y. Sun, D. Bau, and A. Torralba, “Interpretable basis decomposition for visual explanation,” in *ECCV*, 2018.
- [23] B. Zhou, D. Bau, A. Oliva, and A. Torralba, “Interpreting deep visual representations via network dissection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 9, pp. 2131–2145, Sep. 2019.
- [24] J. R. Gardner, M. J. Kusner, Y. Li, P. Upchurch, K. Q. Weinberger, and J. E. Hopcroft, “Deep manifold traversal: Changing labels with convolutional features,” *CoRR*, vol. abs/1511.06421, 2015. [Online]. Available: <http://arxiv.org/abs/1511.06421>
- [25] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas, and R. Sayres, “Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav),” 2017.
- [26] D. Lee and H. Seung, “Learning the parts of objects by non-negative matrix factorization,” *Nature*, vol. 401, no. 6755, pp. 788–791, Oct. 1999.
- [27] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” 2013.
- [28] P. W. Koh and P. Liang, “Understanding black-box predictions via influence functions,” 2017.
- [29] D. Smilkov, N. Thorat, B. Kim, F. Vigas, and M. Wattenberg, “Smoothgrad: removing noise by adding noise,” 2017.
- [30] K. Simonyan, A. Vedaldi, and A. Zisserman, “Deep inside convolutional networks: Visualising image classification models and saliency maps,” *arXiv preprint arXiv:1312.6034*, 2013.
- [31] A. Mahendran and A. Vedaldi, “Understanding deep image representations by inverting them,” 2014.
- [32] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba, “Network dissection: Quantifying interpretability of deep visual representations,” *CoRR*, vol. abs/1704.05796, 2017. [Online]. Available: <http://arxiv.org/abs/1704.05796>
- [33] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas *et al.*, “Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav),” in *International Conference on Machine Learning*, 2018, pp. 2673–2682.
- [34] M. Raghu, J. Gilmer, J. Yosinski, and J. Sohl-Dickstein, “Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability,” in *Advances in Neural Information Processing Systems*, 2017, pp. 6078–6087.
- [35] C. H. Ding, T. Li, and M. I. Jordan, “Convex and semi-nonnegative matrix factorizations,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 1, pp. 45–55, 2010.
- [36] D. D. Lee and H. S. Seung, “Algorithms for non-negative matrix factorization,” in *Proceedings of the 13th International Conference on Neural Information Processing Systems*, ser. NIPS’00. Cambridge, MA, USA: MIT Press, 2000, pp. 535–541. [Online]. Available: <http://dl.acm.org/citation.cfm?id=3008751.3008829>
- [37] Q. Zhang, R. Cao, F. Shi, Y. N. Wu, and S.-C. Zhu, “Interpreting cnn knowledge via an explanatory graph,” 2017. [Online]. Available: <https://arxiv.org/abs/1708.01785>
- [38] M. D. Zeiler, G. W. Taylor, and R. Fergus, “Adaptive deconvolutional networks for mid and high level feature learning,” in *2011 International Conference on Computer Vision*, 2011, pp. 2018–2025.
- [39] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: <http://arxiv.org/abs/1512.03385>
- [40] A. Krizhevsky, “Learning multiple layers of features from tiny images,” Tech. Rep., 2009.
- [41] C. Cortes, M. Mohri, and A. Rostamizadeh, “Algorithms for learning kernels based on centered alignment,” *CoRR*, vol. abs/1203.0550, 2012. [Online]. Available: <http://arxiv.org/abs/1203.0550>
- [42] T. Nguyen, M. Raghu, and S. Kornblith, “On the origins of the block structure phenomenon in neural network representations,” 2022. [Online]. Available: <https://arxiv.org/abs/2202.07184>
- [43] J. Kim, R. D. C. Monteiro, and H. Park, *Group Sparsity in Nonnegative Matrix Factorization*, pp. 851–862. [Online]. Available: <https://epubs.siam.org/doi/abs/10.1137/1.9781611972825.73>
- [44] H. Yin and H. Liu, “Nonnegative matrix factorization with bounded total variational regularization for face recognition,” *Pattern Recogn. Lett.*, vol. 31, no. 16, pp. 2468–2473, Dec. 2010. [Online]. Available: <http://dx.doi.org/10.1016/j.patrec.2010.08.001>
- [45] C. Leng, G. Cai, D. Yu, and Z. Wang, “Adaptive total-variation for non-negative matrix factorization on manifold,” *Pattern Recogn. Lett.*, vol. 98, no. C, pp. 68–74, Oct. 2017. [Online]. Available: <https://doi.org/10.1016/j.patrec.2017.08.027>
- [46] B. Li, G. Zhou, and A. Cichocki, “Two efficient algorithms for approximately orthogonal nonnegative matrix factorization,” *IEEE Signal Processing Letters*, vol. 22, no. 7, pp. 843–846, July 2015.

## APPENDIX

In this supplementary section we present details of the extensions to the framework proposed in section VII. Equation 3 specifies the model with group sparsity constraints, and Equation 4 is the model with Soft-Orthogonality and Total Variation Norm regularization.

In Equation 3 and Equation 4,  $C$  is the number of channels in input data,  $L$  is the number of layers of the network that are part of analysis - as we can select the non-negative layers we want to analyze and are not obligated to include all the layers of any architecture.

$$L = \sum_{i=0}^{C-1} \|D_i - P_i F\|_F^2 + \sum_{j=0}^{L-1} \|A_j - O_j F\|_F^2 + \lambda_F \|F\|_{1,2} \quad (3)$$

$$s.t. P_i, O_j, F, \in \mathbb{R}_+^{A_i \times d}, \mathbb{R}_+^{A_j \times d}, \mathbb{R}_+^{d \times N}, \|F[i, :]\|_1 = 1 \forall i, j$$

The update steps for group sparsity in Equation 3 is provided in the code released alongside this paper.

$$L = \sum_{i=0}^{C-1} \|D_i - P_i F\|_F^2 + \sum_{j=0}^{L-1} \|A_j - O_j F\|_F^2 + \lambda_{F_1} \|F\|_{TV} + \lambda_{F_2} tr(F^T S F) \quad s.t. P_i, O_j, F, \in \mathbb{R}_+^{A_i \times d}, \mathbb{R}_+^{A_j \times d}, \mathbb{R}_+^{d \times N} \forall i, j, \\ S = \mathbf{11}^T - I \in \mathbb{R}^{N \times N}, \|F[i, :]\|_1 = 1 \forall i \quad (4)$$

The following are the update steps for Equation 4. In the update for matrix  $F$ ,  $\text{lap}(F)$  refers to the laplacian of  $F$  and  $\text{sgn}(F)$  refers to the spatial gradient norm of  $F$ .

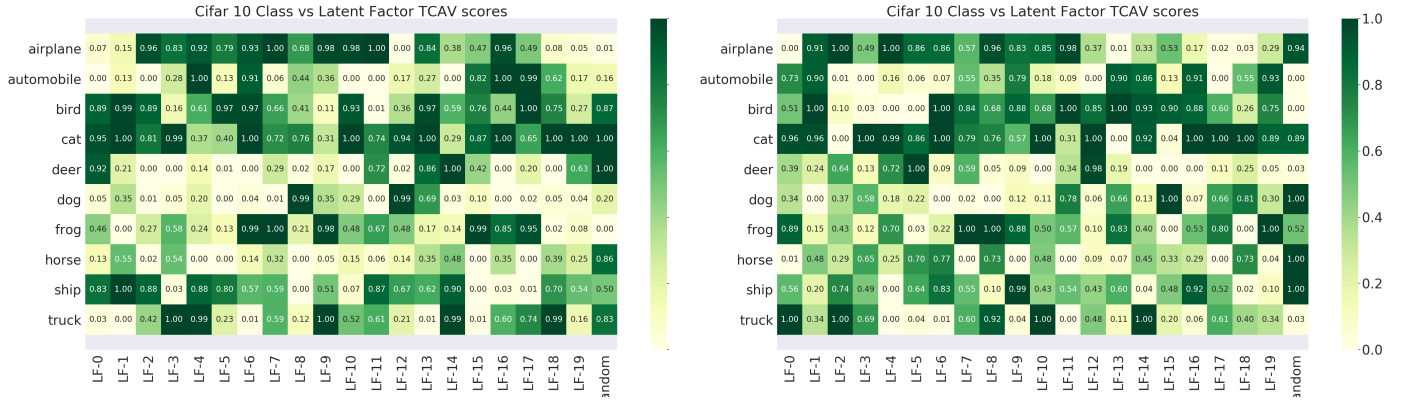
$$F = F * \frac{\sum_i P_i^T D_i + \sum_j O_j^T A_j + \lambda_{TV} \text{lap}(F) / \text{sgn}(F)}{\sum_i P_i^T P_i F + \sum_j O_j^T O_j F + \lambda_{ortho}(S^T F + S F)}$$

$$P_i = P_i * \frac{D_i F^T}{P_i F F^T}$$

$$O_j = O_j * \frac{A_j F^T}{O_j F F^T}$$

Note : The time complexity for the update of  $F$  in Equation 1 is  $O(Nd + \sum_i (dS_i N) + \sum_j (da_j N) + \sum_i (d^2 S_i) + \sum_i (d^2 N) + \sum_j (d^2 a_j) + \sum_j (d^2 N))$ , for  $P_i$  is  $O(S_i d + S_i d^2 + S_i d^3)$ , for  $O_j$  is  $O(a_j d + a_j d^2 + a_j d^3)$ .

Continuing our discussion from the case study of section VII, Next we present a more complete set of the same results from Group Sparsity framework of Equation 3 in Table IV and the results from TV Norm framework of Equation 4 in Table V. The corresponding TCAV scores for the 2 frameworks are presented in Figure 13a and Figure 13b respectively.



(a) TCAV scores for Group Sparsity based model - Inputs Included

(b) TCAV scores for TV based model - Inputs Included

Fig. 13: TCAV scores for Group Sparsity and Total variation based Models

**TABLE IV: Latent Factor Analysis Group Sparsity**

Factor: Classes	Top Images	Latent Image
0: leopard,kangaroo,shrew,tiger		
1: woman,man,camel,girl		
2: dolphin,whale,plain		
3: chair,wardrobe,streetcar		
4: pickup truck,lawn mower, telephone		
5: lamp,telephone,apple		
6: apple,cockroach,orange		
7:mountain,oak,pine		
8: girl,lion,boy		
9: oak,tank,tractor		
10: orange,flatfish,shark		
11: plain,mountain,sea		
12: lion,plate,hamster		
13: dolphin,cockroach,otter		
14: elephant,cattle,palm tree		
15: snail,leopard,porcupine		
16: chair,telephone,pickup truck		
17: sunflower,tulip,poppy,rose		
18: wardrobe,streetcar,bus		
19: squirrel,shrew,rabbit		

**TABLE V: Latent Factor Analysis Total Variation Norm - Inputs Included**

Factor: Classes	Top Images	Latent Image
0: streetcar, bus,wardrobe		
1: apple,chair,lamp		
2: train,bridge,castle		
3: hamster,wolf,baby		
4: cloud,shark,trout		
5: cattle,elephant,camel		
6: cockroach,chair,plain		
7: caterpillar,lizard,shrew		
8: oak,maple,beetle,pine		
9: sea,mountain,cloud,rocket		
10: chair,telephone,wardrobe		
11: cockroach,whale,dolphin		
12: kangaroo,fox,rabbit		
13: rose,tulip,sweet pepper		
14: bus,house, pickup truck		
15: chimpanzee,girl,lion		
16: apple,pear,bottle,orange		
17: hamster,beaver,lion,leopard		
18: wolf,lion,girl		
19: crab,beaver,porcupine		