TCAV: RELATIVE CONCEPT IMPORTANCE TESTING WITH LINEAR CONCEPT ACTIVATION VECTORS

Anonymous authors
Paper under double-blind review

ABSTRACT

Despite the high performance of deep neural networks, the lack of interpretability has been an obstacle for their use in practice. In domains with high stakes (e.g., medical diagnosis), the ability to gain insights into the network's predictions is critical for users to trust and adopt them. One way to improve the interpretability of a NN is to explain the importance of a particular concept (e.g., gender) in the prediction. This is useful for explaining the reasoning behind the networks predictions, and for revealing any biases the network may have. This work aims to provide quantitative answers to *the relative importance of concepts of interest* via concept activation vectors (CAV). In particular, this framework enables non-experts in machine learning to express concepts of interest and test hypotheses using examples (e.g., a set of pictures that illustrate the concept). We show that CAV can be learned given a relatively small set of examples. Hypothesis testing with CAV can answer whether a particular concept (e.g., gender) is more important in predicting a given class (e.g., doctor) than other sets of concepts. Interpreting networks with CAV does not require any retraining or modification of the network. We show that many levels of meaningful concepts are learned (e.g., color, texture, objects, a persons occupation). We also show that CAV's can combined with techniques such as deepdream in order to visualize a network's understanding of specific concepts of interest. We show how various insights can be gained from the relative importance testing with CAV.

1 INTRODUCTION

Neural networks often achieve high performance, yet can be too complex for humans understand why they make the predictions they do. When these powerful black boxes are used in domains with high stakes (e.g., medical diagnosis), being able to provide explanations of a NNs reasoning process is critical and can help gain trust from users.

One frequently-used type of explanation is to attach an importance weight to each input feature. For example, the learned coefficients in linear classifiers or logistic regressions can be interpreted as the importance of each feature for classification. A similar first order importance measure exists for NN's, such as saliency maps (Erhan et al., 2009; Selvaraju et al., 2016). These techniques which use first derivatives as a proxy for the importance of each input feature (pixels). While these approaches are useful, they do not allow users to test any arbitrary features or concepts. For example, if users want to learn the importance of the concept of gender, existing approaches require that gender be an input feature.

Our work offers quantitative measures of *the relative importance of user-provided concepts* using concept activation vectors (CAV). Testing with CAV (TCAV) is designed under the following desiderata:

1. **accessibility**: TCAV should require no user expertise in machine learning (ML)
2. **customization**: TCAV should be adaptable to any concept of interest (e.g., gender) on the fly without pre-listing a set of concepts before training
3. **plug-in readiness**: TCAV should require neither retraining nor modification of the model
4. **quantification**: TCAV should provide a quantitative and testable explanation

One of key ideas for TCAV is that we can test the network with a few concepts of interest at a time, and only aim to learn the *relative importance* of them, instead of ranking the importance of all possible features/concepts. For example, we can gain insights by learning whether gender concept was used *more* than the 'wearing scrubs' concept for the classification of doctor. Similar forms of sparsity (i.e., only considering a few concepts at a time) are used in many existing interpretable models (Kim et al., 2014; Doshi-Velez et al., 2015; Tibshirani, 1994; Zou et al., 2004; Ustun et al., 2013; Caruana et al., 2015). Note that interpretability does not mean understanding the entire network's behavior on every feature/concept of the input (Doshi-Velez, 2017). Such a goal may not be achievable, particularly for ML models with super-human performance (Silver et al., 2016).

TCAV satisfies all the desiderata — accessibility, customization, plug-in readiness and quantification — it enables quantitative relative importance testing for non-ML experts, for user-provided concepts without retraining or modifying the network. Users express their concepts of interest using examples — a set of data points exemplifying the concept. For example, if gender is the concept of interest, users can collect pictures of women. The use of examples has been shown to be powerful medium for communication between ML models and non-expert users (Koh & Liang, 2017; Kim et al., 2014; 2015). Cognitive studies on experts also support this approach (e.g., experts think in terms of examples (Klein, 1989)).

Section 2 relates this work to existing interpretability methods. Section 3 explains how this framework work. Section 4, we show that 1) how this framework learns meaningful directions and 2) the relative importance testing results that shed insights on how network is making predictions.

2 RELATED WORK

In this section, we provide a brief overview of existing related interpretability methods and their relations to our desiderata. We also include the need and the challenges of desiderata 3) plug-in readiness.

2.1 SALIENCY MAP METHODS

Saliency methods seek to identify regions of the input responsible for the classification of the network (Smilkov et al., 2017; Selvaraju et al., 2016; Sundararajan et al., 2017; Erhan et al., 2009; Dabkowski & Gal, 2017). Qualitatively, these methods often successfully label regions of the input which are semantically relevant to the classification.

Unfortunately, these methods do not satisfy our desiderata of 2) customization and 4) quantification. The lack customization is clear, as the user has no control over what concepts of interest these maps pick up on. Regarding quantification, there is no way to meaningfully quantify and interpret the brightness of various regions in these maps. As a motivating example, suppose that given two saliency maps of two different cat pictures, one map highlighted the ear of the cat brighter than the other. It is unclear both how to quantify 'brightness' and second what kind of actionable insights this brightness level gives.

2.2 DEEPDREAM, NEURON LEVEL INVESTIGATION METHODS

There are techniques, such as DeepDream, which can be used to visualize the representation learned by each neuron of a neural network. The technique starts from an image of random noise and iteratively modifies the image in order to maximally activate a neuron or set of neurons of interest (Mordvintsev et al., 2015). This technique has offered some insights into what each neuron may be paying attention to. This technique also has opened up opportunities for AI-aided art (Mordvintsev et al., 2015).

However, the DeepDream method does not satisfy our desiderata 1) accessibility, 2) customization, and 4) quantification. It does not satisfy 1) because it does not treat the network as a black box: a user must understand the internal architecture in order to choose which neurons to visualize. It does not satisfy 2) because no current method exists to find which neurons correspond to semantically meaningful concepts such as gender, and it is unclear whether or not such a neuron even exists. It does not satisfy 4) because we do not understand these pictures and there is currently no method to

quantify how these pictures relate to the output of the network. This method again does not provide actionable insight.

Even if we could understand the pictures produced by DeepDream, often neural networks have too many neurons (e.g., Inception (Szegedy et al., 2015)) to investigate. Humans are typically not capable of digesting this amount of information.

2.3 WHY DO WE NEED DESIDERATA 3 - PLUG-IN READINESS

When interpretability is an objective, we have two options: (1) restrict ourselves to inherently interpretable models or (2) post-process our models to make them more interpretable.

Users may choose option (1) as there are a number of methods to build inherently interpretable models (Kim et al., 2014; Doshi-Velez et al., 2015; Tibshirani, 1994; Zou et al., 2004; Ustun et al., 2013; Caruana et al., 2015). If the users are willing and are able to adapt these models, then this is the gold standard. Although building inherently interpretable machine learning models may be possible for some domains, this often requires compromising on performance.

An interpretable method that can be applied without retraining or modifying the network could be instantly applied to existing models. The need to be able to explain an ML method without rebuilding the whole pipeline is urgent, and may also applied to models already in production. The EU Right to explanation requires ML methods with no human in the loop to provide explanation by year 2018 (Goodman & Flaxman, 2016). All highly engineered ML methods would have to comply to this rule.

2.4 A NOTE ON DESIDERATA 3 - PLUG-IN READINESS AND LOCAL EXPLANATIONS

One of many challenges of building a post-processing interpretable method is to ensure that the explanation is truthful to the model's behavior. There may exist inconsistencies between the explanations and the model's true reasoning behind its behavior.

For instance, one of post-processing methods for neural networks, saliency methods, have been shown to be vulnerable to constant shift in input that does not affect classification. Pieter-Jan Kindermans (2017) showed that simply applying a mean shift for the dataset can cause some saliency methods to result in significant changes in the explanation. This shows that despite the usefulness of these methods, they may not be truthful to how model is actually making decisions.

One way to ensure consistency between explanations and the model's reasoning is to use the generated explanation as an input, and check the network's output for validation. This is typically used in perturbation-based interpretable methods (Koh & Liang, 2017; Ribeiro et al., 2016). These methods perturb the input, and observe the network's response, and use that to generate explanations. They are truthful either locally or globally[1] by construction.

Even a truthful explanation may be misleading if it is only locally truthful (Ribeiro et al., 2016). For example, since the importance of features only needs to be truthful in the vicinity of the data point of interest, there is no guarantee that the method will not generate two completely conflicting explanations. These inconsistencies may result in decreased user trust at a minimum. On the other hand, making a globally truthful explanation may be difficult as the networks decision boundaries may be complex. TCAV produces globally truthful explanations, by focusing on a few concepts of interest at a time.

3 TCAV METHOD

The interpretability methods discussed in the previous section illustrate two common themes. First, linearity is a powerful technique for interpretability, but usually requires significant trade-offs in order to apply. For example, Ribeiro et al. (2016) creates only local interpretations, while generalized

[1]The global explanation means explanations that are globally true, while local explanations mean explanations that are only locally (i.e., a data point and its neighbors) true, following definitions from (Doshi-Velez, 2017).

additive models represent a highly restricted class of functions. A second theme in interpretability research is that directly tying interpretations to training data is a powerful tool.

Motivated by these two ideas, we introduce a method that seems to allow global linear interpretability of a highly flexible class of models, namely deep feedforward neural networks trained on classification tasks. Furthermore, this method is intrinsically tied to real-world data that represents concepts of interest to a human analyst, and can be used to test the relative importance of concepts.

Informally, the key idea is this: although feedforward networks learn highly nonlinear functions there is evidence that they work by gradually disentangling concepts of interest, layer by layer (Alain & Bengio, 2016; Bau et al., 2017). It has also been shown that representations are not necessarily contained in individual neurons but more generally in linear combinations of neurons (Raghu et al., 2017). Thus the space of activation vectors for later layers may have a meaningful global linear structure. Furthermore, if such a structure exists, we can uncover it by training a linear classifier on activation vectors corresponding to a human-selected subset of input examples.

We now formalize this intuition. First, we formally define a *concept activation vector*. Let us imagine that an analyst is interested in a given concept C (e.g., zebra) and has gathered two sets of data points, P_C and N, that represent positive and negative examples of this concept (say, photos of zebras, versus a set of random photos). Typically the set N can simply be all training examples except for P_C. Here we represent the input to the network as vector in $\mathbb{R}^n$, and $P_C, N \subset \mathbb{R}^n$. Suppose that our layer l of the feedforward network consists of m neurons. Then running inference on an input example and looking at the activations at layer l yields a function $f_l : \mathbb{R}^n \to \mathbb{R}^m$.

Our two sets P_C and N naturally give rise to two sets of activation vectors in layer l, namely $f_l(P_C)$ and $f_l(N)$. We can then train a linear classifier on the binary classification task of distinguishing between these two sets. The weights of this classifier are an element $v^l_C \in \mathbb{R}^m$. We call this the concept activation vector for the concept C.

A variation on this idea is a *relative* concept activation vector. Here the analyst selects two sets of inputs that represent two different concepts, C and D. Training a classifier on $f_l(P_C)$ and $f_l(P_D)$ yields a vector $v^l_{C,D} \in \mathbb{R}^m$ which intuitively represents the distinction between the two concepts.

A key benefit of this technique is the flexibility allowed in choosing the set P_C. Rather than being tied to the class labels in the training data, an analyst can–after training–select sets that correspond to any concept of interest. We now describe two ways that this technique be used to interpret a feedforward classifier network. In the next section we discuss the results of experiments with these methods.

3.1 Application: Where are concepts learned?

As a simple demonstration of the value of concept activation vectors, we describe a technique for localizing where a concept is disentangled in a network. A common view of feedforward networks is that different concepts are represented at different layers. For example, many image classification networks are said to detect textures at low levels, with increasingly complex concepts recognized at later layers. Typically evidence for this view comes from various methods of reverse-engineering (Bau et al., 2017; Zeiler & Fergus, 2014; Mordvintsev et al., 2015) the behavior of individual neurons in the network.

Using concept activation vectors, however, we can approach this question from the opposite direction. We can pick out a particular concept C in advance, and consider the concept vectors $v^1_C, v^2_C, \ldots, v^k_C$ for each layer in the network. The changes in accuracies of the classifiers, $a^1_C, a^2_C, \ldots, a^3_C$ then provide a measure of how well disentangled the concept is in each layer. This may be viewed as a generalization of the ideas in (Alain & Bengio, 2016; Bau et al., 2017). In the next section we describe the results of an experiment based on this idea.

3.2 Application: Testing the relative importance of concepts

The real value of concept activation vectors comes from the fact that they may be used to test the relative importance of concepts. For example, an analyst might ask about a photo of a zebra, "was the presence of stripes in the image relevant to the model's classification of the image as a zebra?" In

some cases, with tedious effort in a photo retouching program, it might be possible answer questions like this directly. However, concept activation vectors provide a faster and more general technique.

Consider an input x and concept C of interest. At inference time, this will give rise to activations $f_l(x)$ at layer l. We can then modify this vector in various ways using the concept vector v_C^l. For example, we might create a new vector $w^- = f_l(x) - v_C^l$, which might be viewed as a counterfactual of "x with less of concept C". Performing inference starting at layer $l+1$ based on w^- will give a new classification result y_w. We can also create a new vector $w^+ = f_l(x) + v_C^l$, which might be viewed as a counterfactual of "x with more of concept C".

Thus to test the relevance of stripes to the classification of a zebra, we would either add or subtract a "stripe" concept activation vector from the zebra image activation vector, run forward propagation the results, and examine the final layer of the network. Large changes in the probability of zebra would indicate that stripes played a key role in the classification.

Quantitatively, the metric of the influence of the concept C to class k can be measured by

$$I_w^{up} = \frac{1}{N}\sum_i^N 1(p_k(y) < p_k(y_w)) \qquad I_w^{down} = \frac{1}{N}\sum_i^N 1(p_k(y) > p_k(y_w)) \tag{1}$$

where $p_k(y)$ represents the probability of class k of prediction y. Intuitively speaking, $I_w^{up/down}$ measures the ratio of data points that become 'more/less like concept C' after the modification with concept vector v_C^l.

4 Results

In this section, we first show evidence that the learned concept activation vector indeed reflect the concepts of interest. Then we show the results for hypothesis testing using these concept activation vectors, and the insights one can gain from it.

All our experiments are based on the model from Szegedy et al. (2015) using the publicly available model parameters from Inc (2017). The pictures used to learn the concept activation vectors are collected from the latest ImageNet Fall 2011 release Russakovsky et al. (2015). Each concept vector is learned using between 100-500 pictures as input. We intentionally chose not to use all the available pictures in order to mirror realistic scenarios where users may not be able to collect a large amount of data. The class 'arms' only had 30 pictures, since ImageNet does not include this as a part of the dataset. The pictures used to learn texture concepts are taken from the Bau et al. (2017) data set. In order to learn concept vectors for colors, we generated 500 pictures synthetically by generating the color channels randomly.

4.1 Evidence of the validity of linearity assumption

In this section we provide evidence that the concept activation vectors accurately reflect the given concepts. We show that the linear maps used to construct them are accurate in predicting the concepts. Where these concepts in the networks are learned is consistent with general knowledge about the representations learned by neural networks. Low level features such as edges and textures are detected in the early layers and higher level concepts are only reliably detected in the upper layers. Next, we use activation maximization techniques (Mordvintsev et al., 2015) to visualize each of the concept activation vectors and observe that the patterns are consistent with the provided concept. Finally, we show the top k images that are most similar in terms of cosine similarity to each concept activation vector for further qualitative confirmation.

4.1.1 Where each concept are learned.

Figure 1 shows the accuracy of the linear classifiers at each layer in Inception for each type of concept activation vector. Overall, we observe high accuracy. This is evidence that the given concepts are linearly separable in many layers of the network.

Note that the accuracy of more abstract concept activation vector (e.g., objects) increases in higher layers of the network. The accuracy of a simpler concept activation vector, such as color, is high

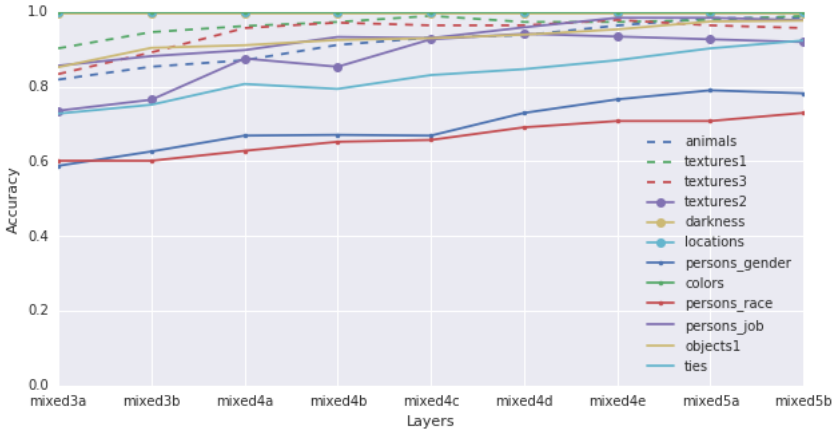


Figure 1: Where each of the concept activation vectors are learned. Accuracy at each layer for concept activation vector. Simple concepts (e.g., colors) achieve high performance in all layers than more abstract concepts (e.g. persons, objects)

throughout the entire network. This agrees with prior work on visualizing the learned representations at different layers of neural networks (Zeiler & Fergus, 2014).

This experiment does not yet show that these concept activation vectors align with the concepts that makes sense semantically to humans. We demonstrate this with the next set of experiments.

4.1.2 Empirical DeepDream: Activation maximization in the direction of the concept activation vectors

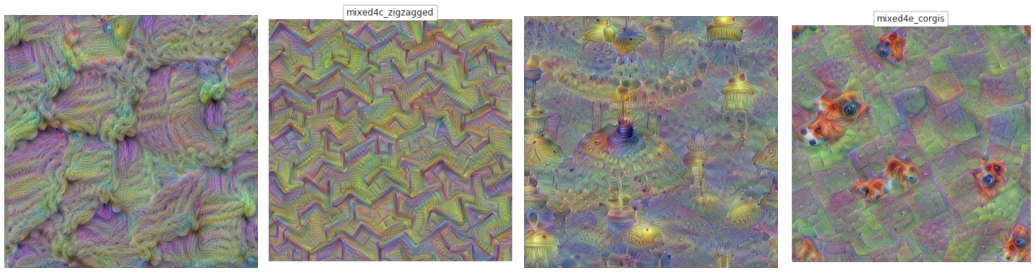


Figure 2: Deepdreamed knitted texture, zigzag texture, lampshades and corgis concept activation vectors.

In this section, we use the activation maximization technique Mordvintsev et al. (2015) to visualize the learned representations in the direction of the concept activation vector. We use an implementation from Mordvintsev. Figure 2 shows highly recognizable features, such as knitted textures, and corgi faces.

The top images in Fig. 3 show the results for set of colors — red, blue, green and yellow (each column) and for the set of layers (lower to higher layers from top to bottom). There seems to be less variance in each layer as the depth increase. The bottom image shows textures. Each column represents lace-like, honeycombed and bumpy. It appears that lower layers learn a uniform scale of texture, and introduce different scales as the depth increases.

These pictures provide some qualitative confirmation that the learned directions align with the given concepts. However, they do not rule out the possibility that the concept activation vectors are 'noisy' in some way. Since the process which produces the concept activation vector only cares about accurately distinguishing between the small set of provided concepts, the concept activation vectors may have also contain other concepts. For example, striped-ness vector may also be correlated t-shirt-

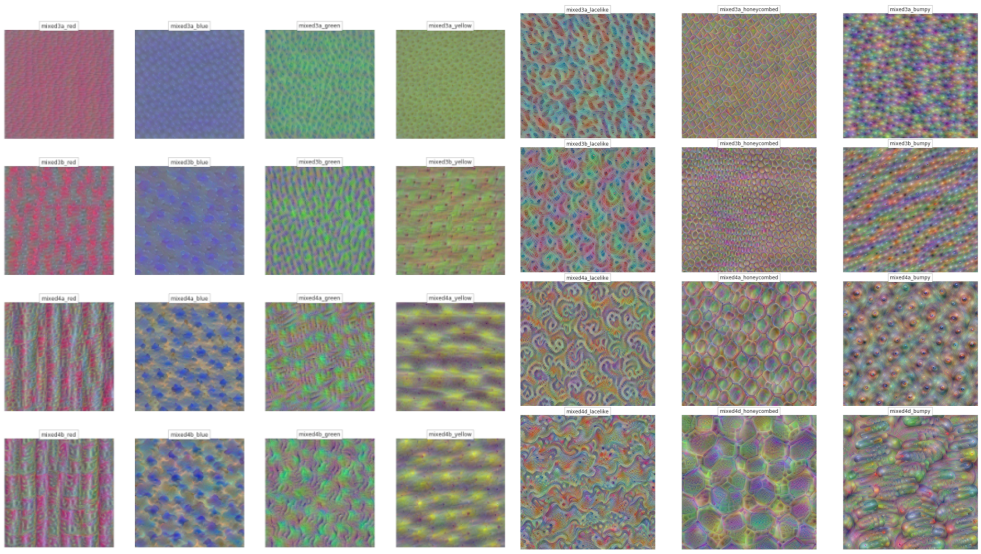


Figure 3: Deepdreamed concept activation vector colors (red, blue, green and yellow in each column) and textures (lace-like, honeycombed and bumpy) for each layer (each row) in inception.

Figure 4: Top and bottom 3 pictures of each concept

ness. The next section provides experiments that the concept activation vector s are indeed aligned with the concept of interest.

4.1.3 QUALITATIVE CONFIRMATION: PICTURES THAT ARE SIMILAR TO CONCEPT ACTIVATION VECTOR

In order to qualitatively confirm that the learned concept activation vector aligns with meaningful concepts, we compute cosine similarity between a set of pictures (all from one class) to the concept activation vector. Fig. 4 shows that [point out a couple things].

We also observe that if there is not much relation between pictures and the concept, the cosine similarity is low. This experiment qualitatively supports our argument that the alignment between concept activation vectors and the meaningful concepts. We recommend doing this before

When using TCAV, in order to ensure that the provided examples were sufficient to learn CAV, we recommend users to check cosine similarity with each concepts to the class.

Now we need quantitative measures. See one-sided hypothesis testing in the next section.

4.2 TESTING RELATIVE IMPORTANCE OF CONCEPTS

To recap, $I^{up/down}_{w^+}$ is the ratio of increased/decreased probability of class for $w^+ = f_l(x) + v^l_C$. $I^{up/down}_{w^-}$ is the same metric for $w^- = f_l(x) - v^l_C$.

When comparing $I^{up}_{w^+}$ and $I^{up}_{w^-}$ (first two columns) in Fig 5, we see that they are often almost inverse of each other. This confirms that adding or subtracting v^l_C clearly shows impact on the classification task, confirming that the v^l_C captures an important aspect of the prediction.

Also note that $I^{up}_{w^+}$ is often similar to $I^{down}_{w^-}$, and $I^{up}_{w^-}$ is to $I^{down}_{w^+}$. Intuitively, this means that there were rarely the case that adding/subtracting v^l_C maintained the probability of class k prediction, $p_k(y_w)$ — adding and subtracting v^l_C always caused predictions to swing.

In our view, this testing captures intrinsic aspects of the class (e.g., fire engine is to red, cucumber is to bumpy, zebra is to striped) and associations (e.g., suit is to CEO). For example, in Fig 5, the yellow color was more influential to cab class than any other colors. This shows that the collected dataset is mostly from countries or cities where color of the cab is yellow (in some cities, cabs are red, green, etc.). From the result that our *women* class was highly relevant to *bikini* class, we infer that the bikini pictures we collected have humans wearing the bikini rather than pictures of the clothing.

The graph for dumbbell class in Fig 5 shows that arms concept was more important to predict dumbbell class than other concepts. This finding is consistent with findings from Mordvintsev et al. (2015), where it discovered that deepdream picture of a dumbbell also showed an arm holding it. Unlike all other concepts in this figure, we only collected 30 pictures of each concept (ImageNet did not have arms as a label). Despite the small number of examples, we are still able to perform TCAV.

5 CONCLUSION

We have introduced the TCAV method, which enables quantitative relative importance testing for non-ML experts. TCAV uses user-provided concepts and does not depend on retraining or modifying the network. TCAV satisfies the desiderata — accessibility, customization, plug-in readiness and quantification.

TCAV can be also used to contrast two trained networks. For example, one can compare the relative importance of concepts to determine how the different choices of training process or architecture influences learning of each concept. Based on this, users can perform model selection based on the concepts that are more or less important for the task.

We hope TCAV can be also used to remove or amplify concepts in the network. This is useful for de-biasing network.

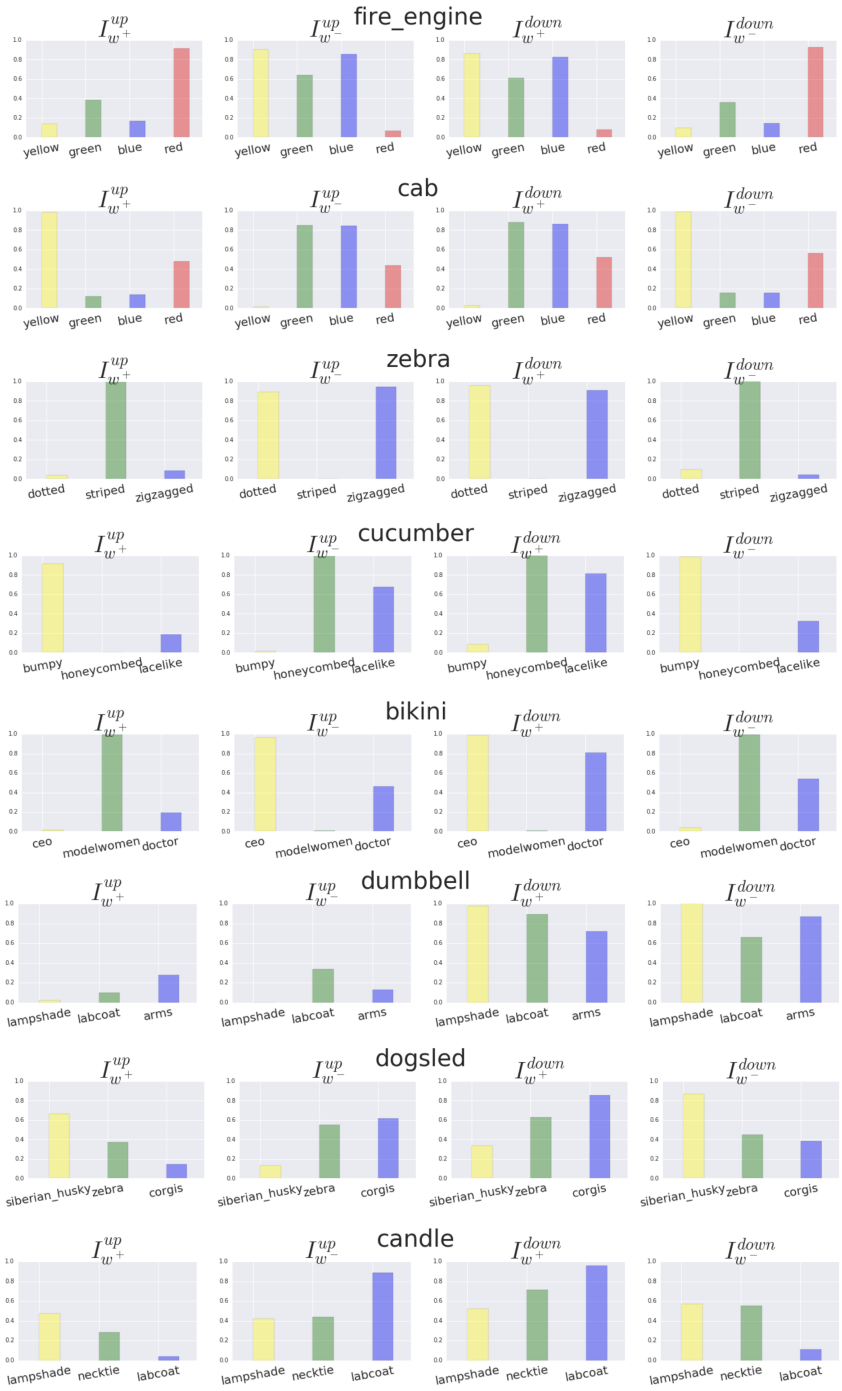


Figure 5: Testing Relative Importance of Concepts

REFERENCES

"http://storage.googleapis.com/download.tensorflow.org/models/inception5h.zip", 2017.

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

David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Computer Vision and Pattern Recognition*, 2017.

R. Caruana, Y. Lou, J. Gehrke, P. Koch, M. Sturm, and N. Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *KDD*, 2015.

Piotr Dabkowski and Yarin Gal. Real time image saliency for black box classifiers. *arXiv preprint arXiv:1705.07857*, 2017.

Been Doshi-Velez, Finale; Kim. Towards a rigorous science of interpretable machine learning. In *eprint arXiv:1702.08608*, 2017.

Finale Doshi-Velez, Byron C Wallace, and Ryan Adams. Graph-sparse lda: A topic model with structured sparsity. In *Aaai*, pp. 2575–2581, 2015.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341:3, 2009.

Bryce Goodman and Seth Flaxman. European union regulations on algorithmic decision-making and a" right to explanation". *arXiv preprint arXiv:1606.08813*, 2016.

B. Kim, C. Rudin, and J.A. Shah. The Bayesian Case Model: A generative approach for case-based reasoning and prototype classification. In *NIPS*, 2014.

Been Kim, Julie Shah, and Finale Doshi-Velez. Mind the gap: A generative approach to interpretable feature selection and extraction. In *Advances in Neural Information Processing Systems*, 2015.

G.A. Klein. Do decision biases explain too much. *HFES*, 1989.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. *arXiv preprint arXiv:1703.04730*, 2017.

Alexander Mordvintsev. Deepdream opensourced code. Assessed: 2017-07.

Alexander Mordvintsev, Christopher Olah, and Mike Tyka. Inceptionism: Going deeper into neural networks. *Google Research Blog. Retrieved June*, 20:14, 2015.

Julius Adebayo Sven Dhne Maximilian Alber Kristof T. Schtt Been Kim Dumitru Erhan Pieter-Jan Kindermans, Sara Hooker. The lack of accountability of saliency methods. 2017.

Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. Svcca: Singular vector canonical correlation analysis for deep understanding and improvement. *arXiv preprint arXiv:1706.05806*, 2017.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier. *arXiv preprint arXiv:1602.04938*, 2016.

O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A.C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 2015.

Ramprasaath R Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh, and Dhruv Batra. Grad-cam: Why did you say that? *arXiv preprint arXiv:1611.07450*, 2016.

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. 2016.

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. *arXiv preprint arXiv:1703.01365*, 2017.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Computer Vision and Pattern Recognition (CVPR)*, 2015. URL `http://arxiv.org/abs/1409.4842`.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.

Berk Ustun, Stefano Tracà, and Cynthia Rudin. Supersparse linear integer models for interpretable classification. *arXiv preprint arXiv:1306.6677*, 2013.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pp. 818–833. Springer, 2014.

Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15:2006, 2004.

--- top 3 knitted imgs of corgis ---

--- bottom 3 knitted imgs of corgis ---

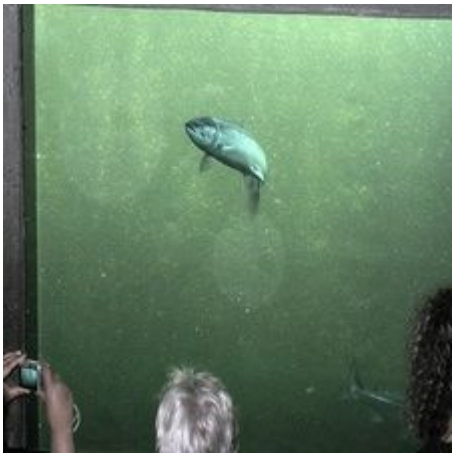
--- top 3 knitted imgs of salmon ---

--- bottom 3 knitted imgs of salmon ---

--- top 3 dotted imgs of corgis ---

--- bottom 3 dotted imgs of corgis ---

--- top 3 dotted imgs of salmon ---



--- bottom 3 dotted imgs of salmon ---

--- top 3 striped imgs of corgis ---

--- bottom 3 striped imgs of corgis ---

--- top 3 striped imgs of zebra ---

--- bottom 3 striped imgs of zebra ---

--- top 3 striped imgs of salmon ---

--- bottom 3 striped imgs of salmon ---

--- top 3 porous imgs of corgis ---

--- bottom 3 porous imgs of corgis ---

--- top 3 porous imgs of salmon ---

--- bottom 3 porous imgs of salmon ---

--- top 3 labcoat imgs of ceo ---

0-th 0.07

1-th 0.03

2-th 0.03

--- bottom 3 labcoat imgs of ceo ---

0-th -0.12

1-th -0.12

2-th -0.13

--- top 3 labcoat imgs of doctor ---

0-th 0.09

1-th 0.09

2-th 0.08

--- bottom 3 labcoat imgs of doctor ---

0-th -0.04

1-th -0.05

2-th -0.05

--- top 3 labcoat imgs of boss ---

0-th 0.06

1-th 0.04

2-th 0.04

--- bottom 3 labcoat imgs of boss ---

0-th -0.11

1-th -0.11

2-th -0.12

--- top 3 labcoat imgs of modelwomen ---

0-th 0.02

1-th 0.01

2-th 0.01

--- bottom 3 labcoat imgs of modelwomen ---

0-th -0.09

1-th -0.10

2-th -0.10

--- top 3 suit imgs of ceo ---

0-th 0.07

1-th 0.07

2-th 0.06

--- bottom 3 suit imgs of ceo ---

0-th -0.05

1-th -0.05

2-th -0.05

--- top 3 suit imgs of headnurse ---

0-th 0.02

1-th 0.01

2-th 0.01

--- bottom 3 suit imgs of headnurse ---

0-th -0.09

1-th -0.09

2-th -0.09

--- top 3 suit imgs of boss ---

0-th 0.10

1-th 0.07

2-th 0.06

--- bottom 3 suit imgs of boss ---

0-th -0.06

1-th -0.07

2-th -0.08

--- top 3 suit imgs of modelwomen ---

--- bottom 3 suit imgs of modelwomen ---

--- top 3 striped imgs of headnurse ---

--- bottom 3 striped imgs of headnurse ---

0-th -0.11

1-th -0.12

2-th -0.14