ADVERSARIAL TCAV - ROBUST AND EFFECTIVE INTERPRETATION OF INTERMEDIATE LAYERS IN NEURAL NETWORKS

PREPRINT

Rahul Soni[1*,2], Naresh Shah[2], Chua Tat Seng[1], and Jimmy D. Moore[2]

[1]School of Computing, National University of Singapore
[2]Utangle AI, Singapore

February 11, 2020

ABSTRACT

Interpreting neural network decisions and the information learned in intermediate layers is still a challenge due to the opaque internal state and shared non-linear interactions. Although [10] proposed to interpret intermediate layers by quantifying its ability to distinguish a user-defined concept (from random examples), the questions of robustness (variation against the choice of random examples) and effectiveness (retrieval rate of concept images) remain. We investigate these two properties and propose improvements to make concept activations reliable for practical use.

Effectiveness: If the intermediate layer has effectively learned a user-defined concept, it should be able to recall — at the testing step — most of the images containing the proposed concept. For instance, we observed that the recall rate of Tiger shark and Great white shark from the ImageNet dataset with "Fins" as a user-defined concept was only 18.35% for VGG16. To increase the effectiveness of concept learning, we propose A-CAV — the Adversarial Concept Activation Vector — this results in larger margins between user concepts and (negative) random examples. **This approach improves the aforesaid recall to** 76.83% **for VGG16**.

For robustness, we define it as the ability of an intermediate layer to be consistent in its recall rate (the effectiveness) for different random seeds. We observed that [12] has a large variance in recalling a concept across different random seeds. For example the recall of cat images (from a layer learning the concept of `tail`) varies from 18% to 86% with 20.85% standard deviation on VGG16. We propose a simple and scalable modification that employs a Gram-Schmidt process to sample random noise from concepts and learn an average "concept classifier". **This approach improves the aforesaid standard deviation from** 20.85% **to** 6.4%.

1 Introduction

Interpretability in Deep Learning has been gaining traction given that the deep neural networks are being employed in critical applications such as medical diagnostics [26, 6] or autonomous vehicles [12, 11] among numerous other domains. Given the wide applications of neural networks, it is important to have model explanation systems that provide a means to debug the model and establish trust for production usage. For example, an explanation technique that tells which pixels are maximally activated for a particular prediction is one way to assess that the model is right for right reasons.

To provide explanations that are useful in probing and improving the model further, it is important that, (1) the explanation technique allows understanding of an object by interpreting its parts — similar to the way humans understand it – for example, a scuba diver wears different parts such as "oxygen regulator", "snorkel mask", "pressure guage" etc.,

*Untangle AI, 79 Ayer Rajah Crescent, #03-01, Singapore. Correspondence to: Rahul Soni <sn.rahul99@gmail.com>

and (2) the explanation technique outlines *where* in the model the concept was being learned — for example, in a face detection task [5], knowing that the k^{th}-layer of a CNN learns the concept of color will help in removing the color bias from the network.

To tackle the aforesaid challenges, Kim [12] proposes one such approach — Concept Activation Vector (CAV) — that provides layer-level understanding of a user-defined concept. CAV requires a small set of examples images of concepts that are easily understood to humans, and, for each layer in network, learns an activation vector representative of that concept. Specifically, given a small set of "concept examples", CAV generates random vectors, called "non-concept examples"; passes the two sets of examples through the network; collects layer activations, and builds a binary linear classifier. The coefficients of the linear model are then defined to be the representative of such a concept.

Although TCAV is a great approach to probe into intermediate layers of a neural network, its accuracy is low in practice, and requires many iterations to have right samples of "random examples" that represent most of the things about "non-concepts". We carry forward Kim's [12] work on testing with CAV to make it more effective (increase empirical performance) and robust to external variations such as the choice of random sampling of non-concept examples.

We pursue our study of CAVs to address the following objectives:

- **Effectiveness**: The effectiveness or strength of the CAV in learning a concept is determined by its strength in retrieving test set images (with known ground truth) containing the said concepts. We propose to improve the retrieval rate by separating the positive and negative examples farther away from the linear decision boundary. We call this method "Adversarial CAV" (A-CAV) and testing with this method — A-TCAV. Next, we transform the negative examples to learn a disjoint subspace of positive and negative data to prevent non-linearity. We call this method "Orthogonal Adversarial CAV" (OA-CAV) and testing with this method — "OA-TCAV".
- **Robustness**: TCAV performance significantly varies with changing the random seed since the random samples can take arbitrary shape including, but not limited to, overlapping with the concept activations. We propose to improve the robustness in CAV by learning coefficient vectors from multiple linear models and computing CAV in the direction orthogonal to the centroid of those coefficient vectors.
- **Prevention against adversarial attacks**: We empirically demonstrate the powerful side-effect of the proposed A-CAV in preventing adversarial attacks at *intermediate layers*. Since the A-CAV is learned from positive adversarial perturbations (that move data points towards the higher Softmax score), we observe that this step suppresses the effect of follow-up adversarial attacks at the testing phase.
- **Investigating Bias in the intermediate layers**: Through our experimental results, we analyze if the model's bottleneck layers are biased towards texture over shape or vice versa.

In the rest of the paper, we empirically demonstrate the effect of simple modifications proposed in this work by defining several human level concepts, and performing experiments with sequential and non-sequential network architectures.

2 Related work

Previous methods have shown interpretability in neural networks, either by (1) updating models to yield interpretability [23, 2] or (2) explaining models in-lieu of simpler / self-explaining examples ([18, 17, 14]), or (3) generating explanations from a trained neural network ([21, 25, 9, 17, 14, 21, 19, 22, 20, 13]). Given that models have become complex, large, and take a long time to train, the later approach to interpretability has become prominent in recent years.

In category (1), [18], for instance, attempts to explain model predictions by generating alternate, "explainable" input vectors which are binary masks that depict either a presence or absence of a feature. [14] attempts to identify which training examples maximally influence the prediction of a given test point. One of the positive side effects of Koh's work [14] was to use influence functions to sort the most relevant training examples, or, remove bad training examples from the training set as a post augmentation process.

To generate explanations from a trained neural network, a typical approach is to observe the effect of model on its prediction when the input pixels are perturbed [21]. One of the enhancements to such perturbations was Selvaraju's work [19] — GradCAM — where the last convolution layer is perturbed instead of the input layer followed by linearly pooling the gradients from all channels. This gradient is then reshaped to the input dimension to generate explanations. [22, 20] proposed a meta attribution algorithm that can sit on top of existing perturbation based algorithms and helps to suppress noise. [13] proposed a method to learn the noise (called distraction) and filter it out at the time of generating explanations. [13] is the most recent work on suppressing distractions in the input signal.

3 Adversarial TCAV

In this section, we investigate the mechanics of Concept Activation Vectors (CAV) [10] and propose improvements to make it more effective and robust. The overall architecture is outlined in Fig 1.

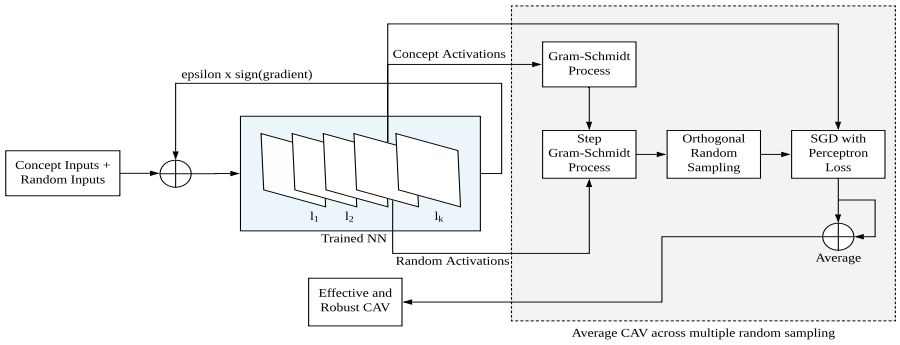


Figure 1: Proposed improvements to CAV. Both, concept inputs and random inputs are updated with adversarial perturbation first, before collecting their activations at a given layer. A repeated Gram-Schmidt process and linear modelling re-computes the random activations and CAV, respectively, to obtained adversarial and orthogonal CAVs

Standard, gradient based, explanation techniques [21, 19, 3, 22, 25] measure the sensitivity of the model prediction w.r.t the input tensor. Kim's work [10] proposes conceptual sensitivity - the directional input sensitivity projected along the direction of the Concept Activation Vector (CAV).

To compute such a directional derivative, the CAV is computed first, as follows: Given a small set of example images of a concept, C (for instance, "fins" as a concept to retrieve shark images), a binary dataset is constructed containing activations of concepts as positive examples and activations of non-concepts (random vectors) as negative examples. A concept activation vector, ${\boldsymbol{v}^l}_C$ at layer l is then defined as the normal vector to the linear decision boundary separating the binary dataset.

For an input test point $\boldsymbol{x} \in \mathbb{R}^n$, let $f_l(\boldsymbol{x}) \in \mathbb{R}^m$ be the activation at layer l and let $h_{l,k}$ be the mapping ($\mathbb{R}^m \to \mathbb{R}$) of activation $f_l(\boldsymbol{x})$ to logit value of k^{th} prediction class. The conceptual sensitivity, $S_{C,k,l}$ w.r.t concept C is then defined as the projection of the gradient of input at that layer onto the concept activation vector[2]:

$$S_{C,k,l} = \nabla_{h_{l,k}}(f_l(\boldsymbol{x})) \cdot {\boldsymbol{v}^l}_C \tag{1}$$

To make the CAVs of great practical use, for instance to retrieve all images containing a given concept from a large unlabelled corpora, we need to assess the effectiveness (higher retrieval rate) and robustness (consistent retrieval rate) of the method.

3.1 Effectiveness in TCAVs

From conceptual sensitivity 1, it is evident that the effectiveness of TCAV is dependent on the ability of a linear classifier to separate concepts from random activations. As the choice of sampling random tensors is not restricted to a particular subpspace, (for example, sampling from outside the subspace of concept activations), the random activations can make the resulting dataset non-linear, thus resulting in poorer retrieval rate (as observed empirically).

3.1.1 Adversarial Separation

To prevent this uncontrolled overlap of activation vectors, we propose to push the activation vectors in their respective category, away from the decision boundary. This can be achieved by adversarial separation - adding a small perturbation in the input vector along the direction of its gradient.

[2]As noted in [10], the Conv layer activation tensors are 4-dimensional (H, W, C_{in}, C_{out}). In this case, we flatten the output to obtain an activation vector of size $(H * W * C_{in} * C_{out})$

$$\boldsymbol{x}^{new} = \boldsymbol{x} + \epsilon sign\left(\nabla_{h_{0,k}}(f_0(\boldsymbol{x}))\right) \tag{2}$$

where $h_{0,k}$ is the mapping ($\mathbb{R}^n \rightarrow \mathbb{R}$) from input $\boldsymbol{x}$ to output prediction k.

Adding such a perturbation increases the Softmax score for the predicted class. Since the positive and negative activations predict different classes (naturally), this increases the separation between the two activations resulting in large margin decision boundary. Such adversarial approach is not new and has been shown to have a large positive effect in separating data points [8, 16, 15].

3.1.2 Disjoint Concept Subspaces

Lee and Seung [**?**] in their `Nature` article propose Non-Negative Matrix factorization (NMF) (and, later, [4] proposes geometrical interpretations) as a way to learn parts of an object (called the basis vectors).

We employ a procedure similar to NFM to further enhance the separability of concept activations from non-concept activations. Specifically, we run a two-step Gram-Schmidt Orthogonalization process ([1]), where, in the first step, we compute an orthogonal basis of the subspace spanned by concept activation vectors — called the `concept basis`. In the second step, we compute another orthogonal basis — called the `non-concept orthogonal basis` (or, without loss of generality, `non-concept basis`) - which is disjoint to the `concept basis`. With concept activations as positive examples, we can now generate negative examples for the downstream binary classifier by sampling activations from the `non-concept basis`. Our empirical observations suggests that sampling from more than one instances of `non-concept basis` provides good generalization. An example of such a sampling is shown in Fig 2. Algorithm 1 and Algorithm 2 demonstrate the two modifications to CAV respectively.

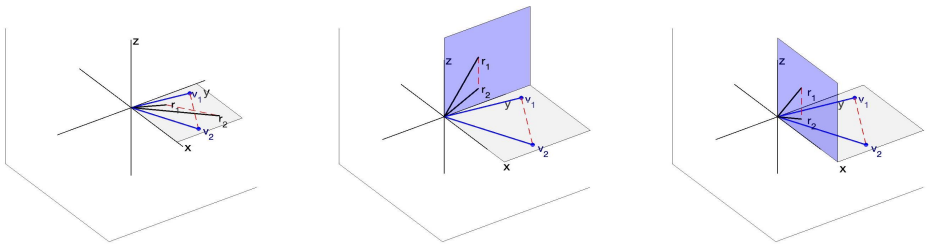


Figure 2: Examples of random sampling. **left**: no constraints on random sampling can lead to overlapping points or distributions with smaller decision margin, **middle, right**: two instances of the proposed random sampling using the Gram-Schmidt process. $\mathbf{r}_1$, $\mathbf{r}_2$ represent random activations and $\mathbf{v}_1$, $\mathbf{v}_2$ concept activations in a $2D$ subspace.

3.2 Robustness in TCAVs

We performed several validations of CAV (across multiple random seeds and multiple concepts) to sort images in relation to a given concept. We observed that TCAV was not effective consistently, i.e., it had a high standard deviation in recall rate. For instance, to sort Tiger shark and Great white shark images from the Imagenet validation categories in relation to `fins` concept, the recall rate varied from 19% to 78% with standard deviation of 19.03%.

To make CAVs robust to random seed selection, we propose to run multiple instances of linear model on different random examples sampled from the `non-concept basis`. CAV, then points in the centroid direction of the learned

Algorithm 1 Adversarial Separation

Input: Concept examples $\boldsymbol{x}_i \in \mathbb{R}^n$, trained Neural Network estimator, $f(\cdot)$
Initialize $\boldsymbol{r}_i \in \mathbb{R}^n \longleftarrow$ `random tensors`
Initialize $K \longleftarrow$ `# concept examples`
Initialize $L \longleftarrow$ `# non-concept examples`
Initialize $l \longleftarrow$ `neural network layer index`
Initialize `concept activations, cact` $\longleftarrow$ `{}`
Initialize `non-concept activations, ncact` $\longleftarrow$ `{}`
Initialize $g(\cdot) \longleftarrow$ `SGD Classifier with perceptron loss`
for $i = 1$ **to** K **do**
  $\boldsymbol{x}_i \longleftarrow \boldsymbol{x}_i + \epsilon sign\left(\nabla_{h_{0,k}}(f_0(\boldsymbol{x}_i))\right)$
  `cact`$.insert(f_l(\boldsymbol{x}_i))$
end for
for $i = 1$ **to** L **do**
  $\boldsymbol{r}_i \longleftarrow \boldsymbol{r}_i + \epsilon sign\left(\nabla_{h_{0,k}}(f_0(\boldsymbol{r}_i))\right)$
  `ncact`$.insert(f_l(\boldsymbol{r}_i))$
end for
`TRAIN` $g($`cact`$,$ `ncact`$)$
$\boldsymbol{u}^l{}_C \longleftarrow$ `coefficients of g(cact, ncact)`
$\boldsymbol{v}^l{}_C \longleftarrow -1 * \boldsymbol{u}^l{}_C$

Algorithm 2 Orthogonal Adversarial Separation. Here, `GRAM-SCHMIDT` denotes the Gram-Schmidt Orthogonalization (GSO) process and `STEP GRAM-SCHMIDT` is a single step of GSO that returns a component of input vector orthogonal to the given basis set.

Input: Concept examples $\boldsymbol{x}_i \in \mathbb{R}^n$, trained Neural Network estimator, $f(\cdot)$
Initialize $\boldsymbol{r}_i \in \mathbb{R}^n \longleftarrow$ `random tensors`
Initialize $K \longleftarrow$ `# concept examples`
Initialize $L \longleftarrow$ `# non-concept examples`
Initialize $l \longleftarrow$ `neural network layer index`
Initialize `concept activations, cact` $\longleftarrow$ `{}`
Initialize `non-concept activations, ncact` $\longleftarrow$ `{}`
Initialize $g(\cdot) \longleftarrow$ `SGD Classifier with perceptron loss`
for $i = 1$ **to** K **do**
  $\boldsymbol{x}_i \longleftarrow \boldsymbol{x}_i + \epsilon sign\left(\nabla_{h_{0,k}}(f_0(\boldsymbol{x}_i))\right)$
  `cact`$.insert(f_l(\boldsymbol{x}_i))$
end for
for $i = 1$ **to** L **do**
  $\boldsymbol{r}_i \longleftarrow \boldsymbol{r}_i + \epsilon sign\left(\nabla_{h_{0,k}}(f_0(\boldsymbol{r}_i))\right)$
  `ncact`$.insert(f_l(\boldsymbol{r}_i))$
end for
`concept_basis` $\longleftarrow$ `GRAM-SCHMIDT(cact)`
for $i = 1$ **to** L **do**
  $\boldsymbol{q}_i \longleftarrow$ `ncact.get(i)`
  $\boldsymbol{q}_i{}^{\perp} \longleftarrow$ `STEP GRAM-SCHMIDT(cact,` $\boldsymbol{q}_i$`)`
end for
`non_concept_basis` $\longleftarrow$ `GRAM-SCHMIDT(`$\{\boldsymbol{q}_1{}^{\perp}, \boldsymbol{q}_2{}^{\perp} \cdots \boldsymbol{q}_L{}^{\perp}\}$`)`
`ncact` $\longleftarrow$ `sample(non_concept_basis)`
`TRAIN` $g($`cact`$,$ `ncact`$)$
$\boldsymbol{u}^l{}_C \longleftarrow$ `coefficients of g(cact, ncact)`
$\boldsymbol{v}^l{}_C \longleftarrow -1 * \boldsymbol{u}^l{}_C$

linear coefficient vectors. The number of such draws is a fixed, configurable hyper-parameter, N_d. This modification is surprisingly effective - especially when combined with the adversarial and orthogonal separation. We observe that the approach is not only robust against changing seeds, it is also robust against changing the hyper-parameter, N_d itself. For example, changing the hyper-parameter from $N_d = 10$ to $N_d = 100$ does not yield additional performance gains (reduction in standard-deviation) — thus promising a stable result consistently.

4 Experimental Setup

Without loss of generality, we validate our method on a subset of 20 Imagenet categories — which we call the "Imagenet20 Dataset" — to simplify the CAV generation process. We test our method on modified "VGG16", "Resnet18", and "Alexnet" — where the last layer is modified to output 20 class probabilities.

4.1 Models

We fine-tune the modified models to Imagenet20 dataset using the $80 - 20$ train-validation split, with the Adam optimizer, for 50 epochs, and learning rate decay when no validation loss improvement over a patience of 10 epochs. The initial learning rate is set to 0.001 and reduced by a factor of 0.25 after $patience = 10$ *epochs* is reached. Table 1 summarizes the training statistics of the three models.

Table 1: Training summary of modified VGG16 (**VGG16***), modified Resnet18 (**Resnet18***), and modified Alexnet (**Alexnet***) after the fine-tuning step. Each model is first loaded with pretrained Imagenet weights and then finetuned (for 50 epochs) to output 20 class probabilities.

	VGG16*		**RESNET18***		**ALEXNET***	
	LOSS	ACC	LOSS	ACC	LOSS	ACC
TRAIN	0.08	96.95	0.001	99.98	0.098	97.13
VALID	0.121	95.68	0.56	85.55	0.33	88.65
TEST	0.28	91.2	0.74	80.70	0.456	84.70

4.2 Dataset

To construct Imagenet20, we selected categories such that more than one category shares a common user defined concept, for example: the `fins` concept is common to `Great white shark` and `Tiger shark`. The concept of `tail` was common to `Egyptian cat and Tiger cat`. This was done to ensure that CAV learns a concept and does not simply do pattern matching with the context. For example, the `Tiger cat` validation dataset of Imagenet contains 24% actual tiger images (12 out of 50 images). Naturally, the context (background) of these images depicts "outdoor" setting, which is in contrast to the `Egyptian cat` category where the context is mostly indoors. Following list of concepts are constructed and validated for our method:

- **Fins** - To recall images of `Great white shark` and `Tiger shark` from the Imagenet20 validation set.
- **Cat's fur** - To recall images of `Egyptian cat` and `Tiger cat` from the Imagenet20 validation set.
- **Cat's tail** - To recall images of `Egyptian cat` and `Tiger cat` from the Imagenet20 validation set.

4.3 Parameter Setting

To compute CAV, we used the SGD linear model with perceptron loss and constant learning rate (to avoid overfitting the small concept dataset). We computed both the baseline CAV and our method (A-CAV, OA-CAV) for 50 random seeds and compared recall results. For the adversarial separation in our method, the ϵ term is set to a tune-able hyper-parameter taking values in $\{0.1, 0.01, 0.005, 0.001, 0.0001\}$. We computed 3 instances of the Grahm-Schmidt Orthogonalization (GSO) process to obtain the "non-concept activations". The linear model takes the concept activations and the output of GSO to learn a CAV. We performed 10 such iterations of linear model to obtain an "Adversarial CAV" (A-CAV).

5 Results and Insights

We summarize our experimental results in this section and shed light on the following insights:

- Is the bottleneck Convolutional layer biased towards shape or textures?
- Effect of adversarial attacks on shapes versus textures

5.1 Image Retrieval (Recall rate)

We use the learned A-CAV and OA-CAv and test its strength by sorting images of the Imagenet20 test dataset. For each image, we compute the test activation vector at the same layer on which the CAV was built and compute the inner product according to Eq 1. We summarize the recall results of different models tested for different concepts in Table 2, 3, 4. In the tables, we refer to **TCAV** as the baseline method [12] test, **A-TCAV** — the proposed adversarial CAV test, and **OA-TCAV** — the proposed A-TCAV test with orthogonal random sampling of "non-concept" activations. We observed that the proposed methods are consistent across 50 random seeds with significantly lower standard deviation in recall compared to baseline TCAV.

We observe that A-CAV performs consistently well across all formats with occasional higher performance from OA-TCAV (testing `fins` on VGG16, `fur` on Alexnet). This is due to the reason that OA-CAV learns orthogonal disjoint subspaces which relaxes the degree of freedom of the linear decision boundary. Nevertheless, we believe that OA-CAV based sampling provides meaningful insights and opens possibilities for future investigations.

Fig 3 shows a trend in the recall rate for a sample of seeds that have shown better performance for A-CAV. The occasional jumps in recall rate of baseline CAV is attributed to a *good* sampling of random vectors that inherently tilts the dataset towards linear separability.

Table 2: **VGG16 results**. Recall rate of testset images from the Imagenet20 dataset containing a given concept when CAVs were trained on the bottleneck layer Numbers marked in bold represent the best performance and CAVs were learned on the bottleneck layer

	FINS RECALL (IN %)			TAIL RECALL (IN %)			FUR RECALL (IN %)		
	TCAV	A-TCAV	OA-TCAV	TCAV	A-TCAV	OA-TCAV	TCAV	A-TCAV	OA-TCAV
MEAN	18.35	76.83	**79.66**	51.92	**80.60**	67.08	42.92	**56.36**	50.96
STD	11.08	**3.44**	3.49	20.85	**6.4**	12.88	10.55	**3.98**	7.42
MIN	1	66	70	18.0	62.0	27.0	20.0	50.0	32.0
MAX	41	83	85	86.0	90.0	88.0	62.0	66.0	63.0

Table 3: **Resnet18 results**. Recall rate of testset images containing a given concept from the Imagenet20 dataset when CAVs were trained on the bottleneck layer. Numbers marked in bold represent the best performance.

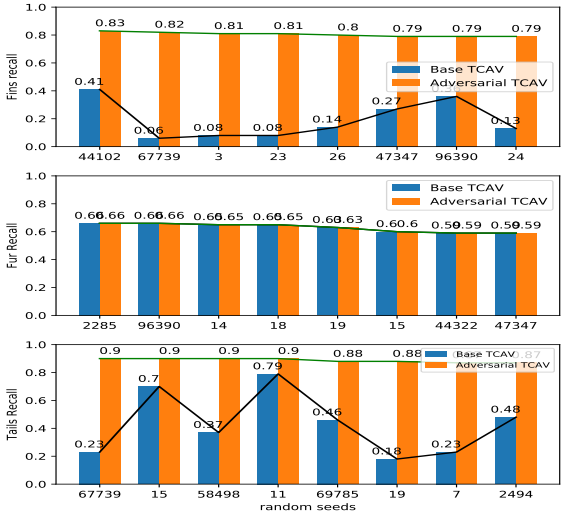
	FINS RECALL (IN %)			TAIL RECALL (IN %)			FUR RECALL (IN %)		
	TCAV	A-TCAV	OA-TCAV	TCAV	A-TCAV	OA-TCAV	TCAV	A-TCAV	OA-TCAV
MEAN	43.89	62.34	**66.51**	67.26	**77.11**	72.87	56.11	71.83	**76.92**
STD	19.28	**6.04**	12.29	14.79	**4.17**	6.67	17.87	**4.62**	8.11
MIN	3.0	38.0	35.0	3.0	69.0	41.0	12.0	62.0	49.0
MAX	77.0	76.0	84.0	85.0	84.0	84.0	86.0	81.0	89.0

Table 4: **Alexnet results**. Recall rate of testset images containing a given concept from the Imagenet20 dataset when CAVs were trained on the bottleneck layer. Numbers marked in bold represent the best performance.

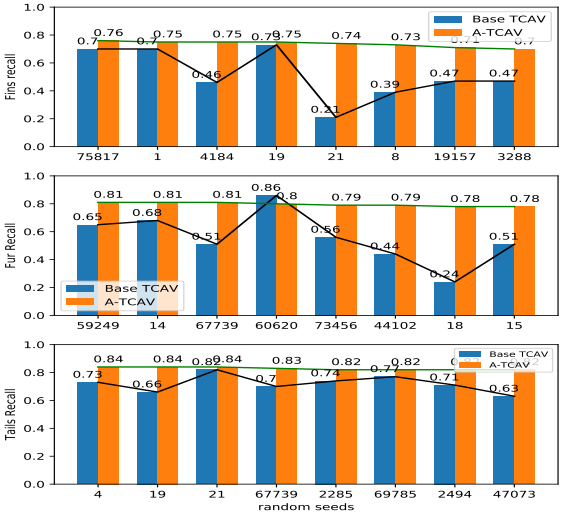
	FINS RECALL (IN %)			TAIL RECALL (IN %)			FUR RECALL (IN %)		
	TCAV	A-TCAV	OA-TCAV	TCAV	A-TCAV	OA-TCAV	TCAV	A-TCAV	OA-TCAV
MEAN	36.87	64.72	**65.56**	41.79	**60.06**	47.31	42.92	56.36	**50.96**
STD	22.16	11.02	**7.58**	14.56	**5.23**	11.09	10.55	**3.98**	7.42
MIN	6.0	24.0	35.0	17.0	47.0	24.0	20.0	50.0	32.0
MAX	71.0	78.0	75.0	63.0	69.0	70.0	62.0	66.0	63.0

5.2 Bias in the Bottleneck Conv Layer

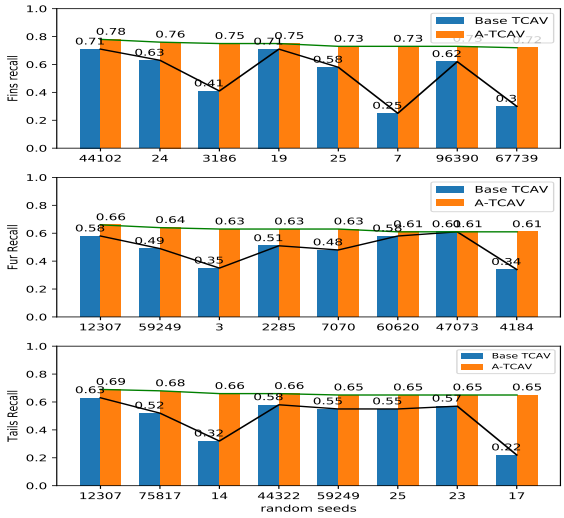
[7] argues that neural networks are more reliant on texture than shape to recognize an object. Without having access to the internal working mechanisms of a model, it is hard to comment if the model is globally more reliant on texture or only its localized neuron groups. To resolve this issue, our data construction and experiments provide great insights.



(a) **VGG16**



(b) **Resnet18**



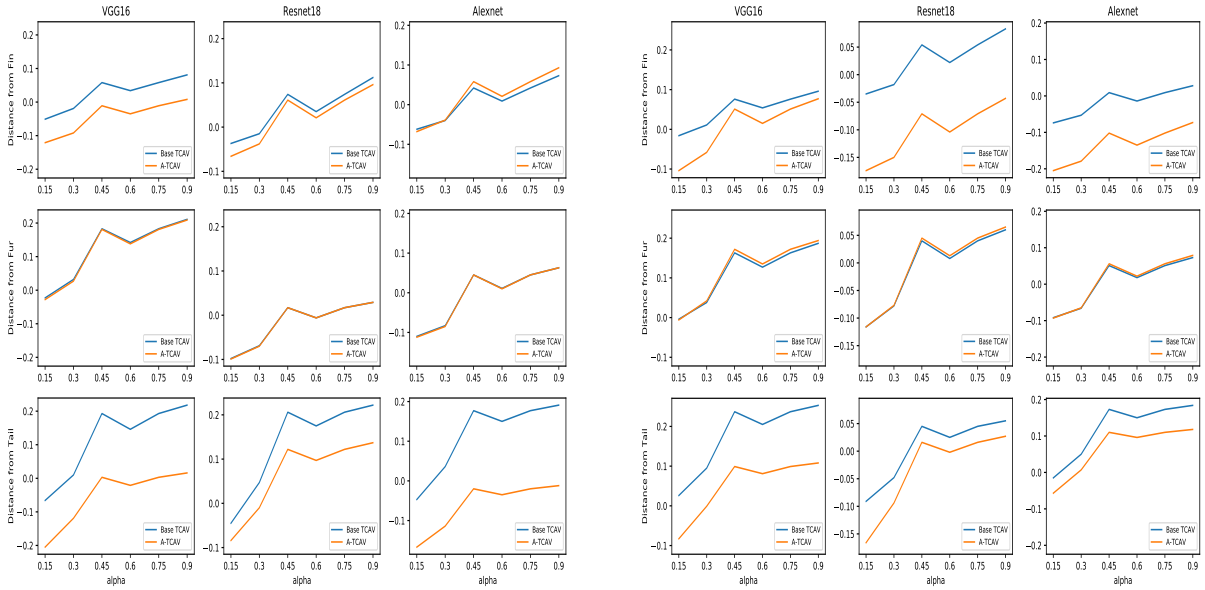
(c) **Alexnet**

Figure 3: Comparison of recall rate of TCAV, A-TCAV, and OA-TCAV; and variation across different seeds (seeds that have resulted in higher accuracy for A-CAV). We observe that the A-TCAV is both effective (higher recall) and robust (consistent to random sampling) compared to the baseline

We note that the model's testset recall (in relation to a *(*conceptual-layer)) is achieved by sorting the sensitivity scores which are achieved by inner product of CAV and a given test example. This linear operation makes the recall rate proportional to the concept sensitivity.

From Table 2, 3, 4 which summarizes retrieval results from the *bottleneck layer* of VGG16, Resnet18, and Alexnet, we observe that the recall rate of "Egyptian cat" and "Tiger cat" in relation to the concept `fur` is lower than recall rate in relation to the concept of `tail`.

We therefore hypothesize that: *the bottleneck layers are less sensitive to learning texture based concepts* (in this case `cat's fur`) *and more sensitive to shape based concepts* (in this case `fins`, `tail`).



(a) seed=14

(b) seed=21

Figure 4: Average distance (defined as inverse of sensitivity) between CAVs and the adversarial dataset with increase level of perturbation in the dataset. Lower numbers mean *closeness* to the CAVs.

5.3 Prevention Against Adversarial Attacks

Adversarial attacks have grown to be prominent in deceiving models prediction[8]. As a natural extension, we pose the following question. *What is the effect of adversarial attacks on the concept understanding of intermediate layers?*

Consider a concept C learned at layer l with a given sensitivity score, $S_{C,l}$ for a set of test images. If we replace the testing set with its adversarial equivalent, the distance of testing images from the concept activation would increase and would continue to increase with increasing levels of adversarial perturbation. A method immune to adversarial attack must separate slowly from the CAVs.

To investigate empirically, we take the testset images of Imagenet20 dataset and construct adversarial examples using iterative perturbation until a mis-classification has occured [8]. Next, we compare the separation of the adversarial dataset for baseline CAV and A-CAV at the bottleneck layer. As evident from Fig 4, we observe that the proposed method consistently remains *closer* to the concept learned in the intermediate layers — thus improving the robustness of intermediate layers against adversarial attacks.

6 Conclusion & Future Scope

TCAV is an excellent approach to probe intermediate layers of the neural network and gain human-level understanding of what concept a layer has learned. TCAVs, however, have shortcomings such as effectiveness and robustness that we attempted to overcome.

In the future, we aim to investigate the effects of the following proposals (which are currently outside of the scope of current work): (1) Effect of iterative adversarial updates on the CAV sensitivity, for example the Fast Gradient Sign Attack [8]. Other sophisticated methods could be explored additionally, (2) Consistency measures of Shapley TCAV: [24], (3) Imposing strictly disjoint sets of positive and negative examples has led to larger standard deviation that A-CAV as we see in Table 2, 3, 4. Methods to relax this criteria and learn a better representation of "non-concept" examples would be an interesting topic to explore.

Currently, CAV helps to learn concepts at layer level which is too broad a definition, since layers contain shared information about concepts from different target categories. Methods to explore and isolate a group of neurons, learning a particular concept in a layer, would be a great future contribution to further probe into the working mechanics of black-box neural networks.

References

[1] Åke Björck. Numerics of gram-schmidt orthogonalization. *Linear Algebra and Its Applications*, 197:297–316, 1994.

[2] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1721–1730, 2015.

[3] Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 839–847. IEEE, 2018.

[4] David Donoho and Victoria Stodden. When does non-negative matrix factorization give a correct decomposition into parts? In *Advances in neural information processing systems*, pages 1141–1148, 2004.

[5] Sachin Sudhakar Farfade, Mohammad J Saberian, and Li-Jia Li. Multi-view face detection using deep convolutional neural networks. In *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, pages 643–650, 2015.

[6] William Gale, Luke Oakden-Rayner, Gustavo Carneiro, Andrew P Bradley, and Lyle J Palmer. Producing radiologist-quality reports for interpretable artificial intelligence. *arXiv preprint arXiv:1806.00340*, 2018.

[7] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.

[8] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[9] Been Kim, Rajiv Khanna, and Oluwasanmi O Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. In *Advances in Neural Information Processing Systems*, pages 2280–2288, 2016.

[10] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, and Rory Sayres. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). *arXiv preprint arXiv:1711.11279*, 2017.

[11] Jiman Kim and Chanjong Park. End-to-end ego lane estimation based on sequential transfer learning for self-driving cars. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 30–38, 2017.

[12] Jinkyu Kim and John Canny. Interpretable learning for self-driving cars by visualizing causal attention. In *Proceedings of the IEEE international conference on computer vision*, pages 2942–2950, 2017.

[13] Pieter-Jan Kindermans, Kristof T Schütt, Maximilian Alber, Klaus-Robert Müller, Dumitru Erhan, Been Kim, and Sven Dähne. Learning how to explain neural networks: Patternnet and patternattribution. *arXiv preprint arXiv:1705.05598*, 2017.

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

[15] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Advances in Neural Information Processing Systems*, pages 7167–7177, 2018.

[16] Shiyu Liang, Yixuan Li, and Rayadurgam Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv preprint arXiv:1706.02690*, 2017.

[17] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pages 4768–4777, 2017.

[18] Andrew Slavin Ross, Michael C Hughes, and Finale Doshi-Velez. Right for the right reasons: Training differentiable models by constraining their explanations. *arXiv preprint arXiv:1703.03717*, 2017.

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

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

[21] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

[22] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3319–3328. JMLR. org, 2017.

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

[24] Chih-Kuan Yeh, Been Kim, Sercan O Arik, Chun-Liang Li, Pradeep Ravikumar, and Tomas Pfister. On concept-based explanations in deep neural networks. *arXiv preprint arXiv:1910.07969*, 2019.

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

[26] Kai Zhang, Xiyang Liu, Fan Liu, Lin He, Lei Zhang, Yahan Yang, Wangting Li, Shuai Wang, Lin Liu, Zhenzhen Liu, et al. An interpretable and expandable deep learning diagnostic system for multiple ocular diseases: qualitative study. *Journal of medical Internet research*, 20(11):e11144, 2018.