TCAV

Jeong-Woon Kang

Master’s Course

School of Information Convergence Engineering

Pusan National University

dnsdudwk1027@naver.com

2022.01.20

부산대학교
PUSAN NATIONAL UNIVERSITY

Contents

- TCAV(Testing with Concept Activation Vectors)

TCAV(Testing with Concept Activation vector)

- Been Kim et al. (2018), ICML

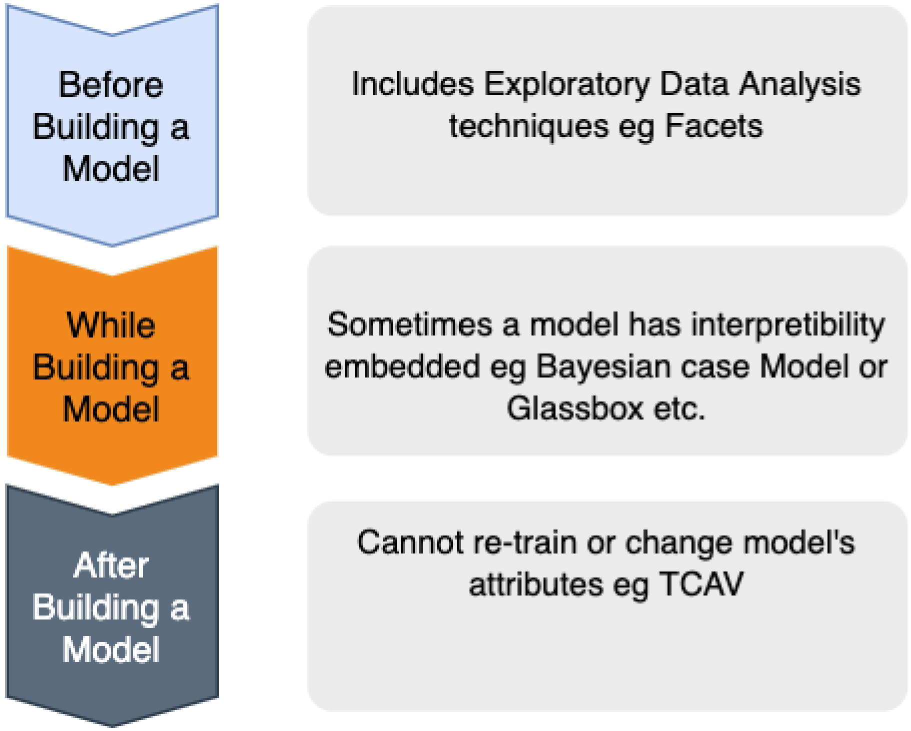
- TCAV_ICML (beenkim.github.io)
- https://beenkim.github.io/slides/TCAV_ICML_pdf.pdf
- TCAV를 받아들이기 위한 노력 – 라 코뮌 드 아트 (wordpress.com)
- MLSSToronto2018 (beenkim.github.io)
- TCAV: Interpretability Beyond Feature Attribution | by Parul Pandey | Towards Data Science
- XAI Review - 4. XAI in Computer Vision(Saliency map, TCAV, latest methodology) – YouTube
- [포테이토 논문 리뷰] (TCAV) Interpretability Beyond Feature Attribution:Quantitative Testing with Concept Activation Vectors (tistory.com)

XAI

- Saliency Map(2013) : pixelwise 1차 미분
- LRP(2015)
- CAM(2015) : 마지막 conv layer에 GAP
- LIME(2016)
- Grad-CAM(2016) : GAP제거, weight를 gradient 기준으로 변경
- SHAP(2017) : 게임이론
- TCAV(2018) : Concept
- Concept SHAP(2019)
- Concept Bottleneck Models(2020)

TCAV-introduction

- ML model’s interpretability



Before Building a Model

Includes Exploratory Data Analysis techniques eg Facets

While Building a Model

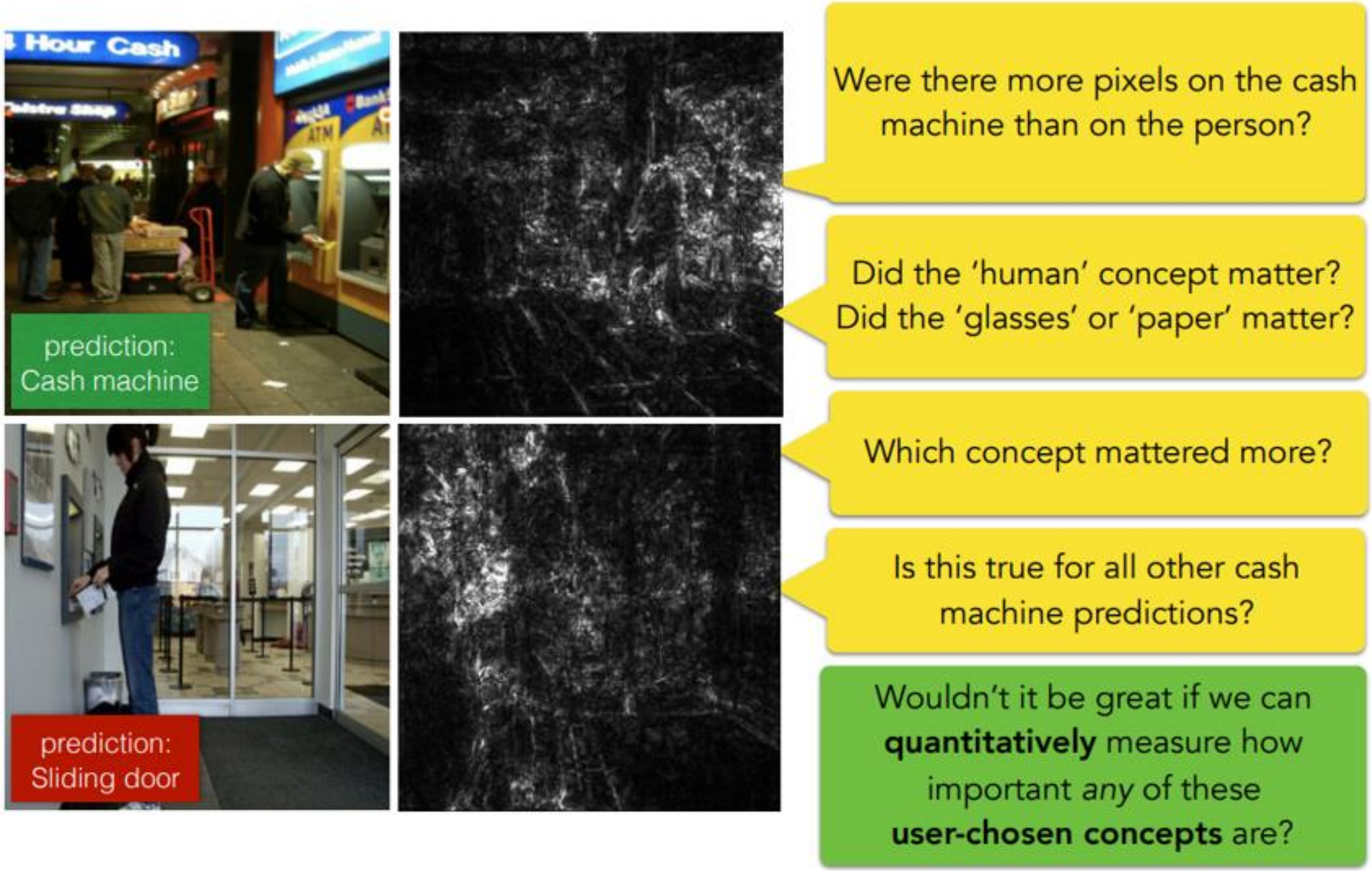
Sometimes a model has interpretibility embedded eg Bayesian case Model or Glassbox etc.

After Building a Model

Cannot re-train or change model's attributes eg TCAV

Types of Interpretability Techniques

TCAV-introduction



Were there more pixels on the cash machine than on the person?

Did the 'human' concept matter? Did the 'glasses' or 'paper' matter?

Which concept mattered more?

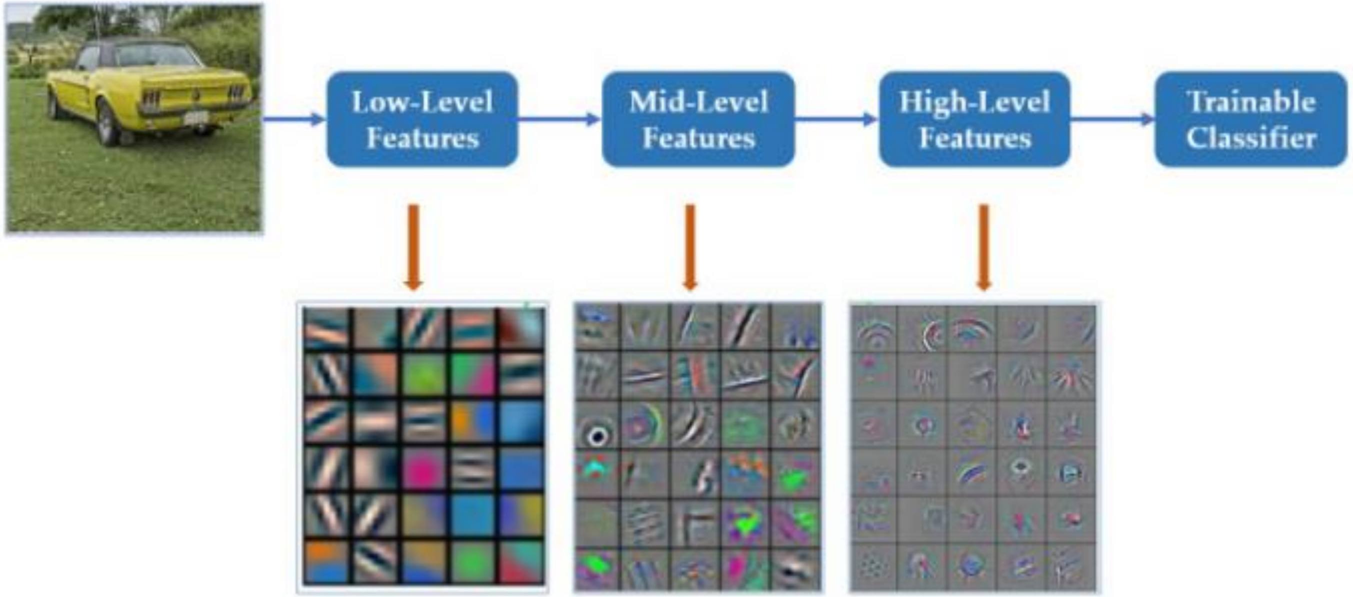
Is this true for all other cash machine predictions?

Wouldn't it be great if we can **quantitatively** measure how important *any* of these **user-chosen concepts** are?

Saliency Map

TCAV-introduction

- Low-level features : edges, line, color of single pixel
- High-level concepts(familiar to human) : stripes in a zebra



Low-Level Features

Mid-Level Features

High-Level Features

Trainable Classifier

- Model Interpretability를 위한 기존 접근 방법

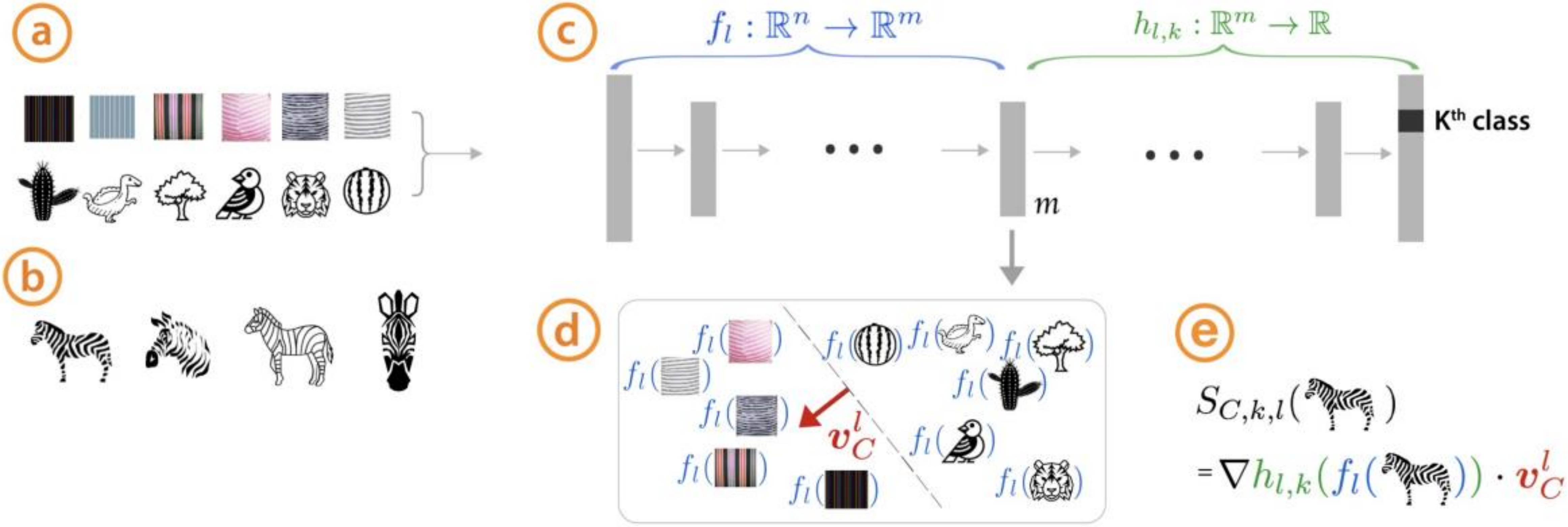
CAV

- TCAV : Testing with Concept Activation Vector
- CAV : low-level feature를 concept으로 변환하는 과정

- e_m : input features, neural activation과 같은 data vector
- E_m : basis vector e_m으로부터 span된 벡터공간
- e_h : human-interpretable concept에 관련된 vector
- E_h : basis vector e_h로부터 span된 벡터공간

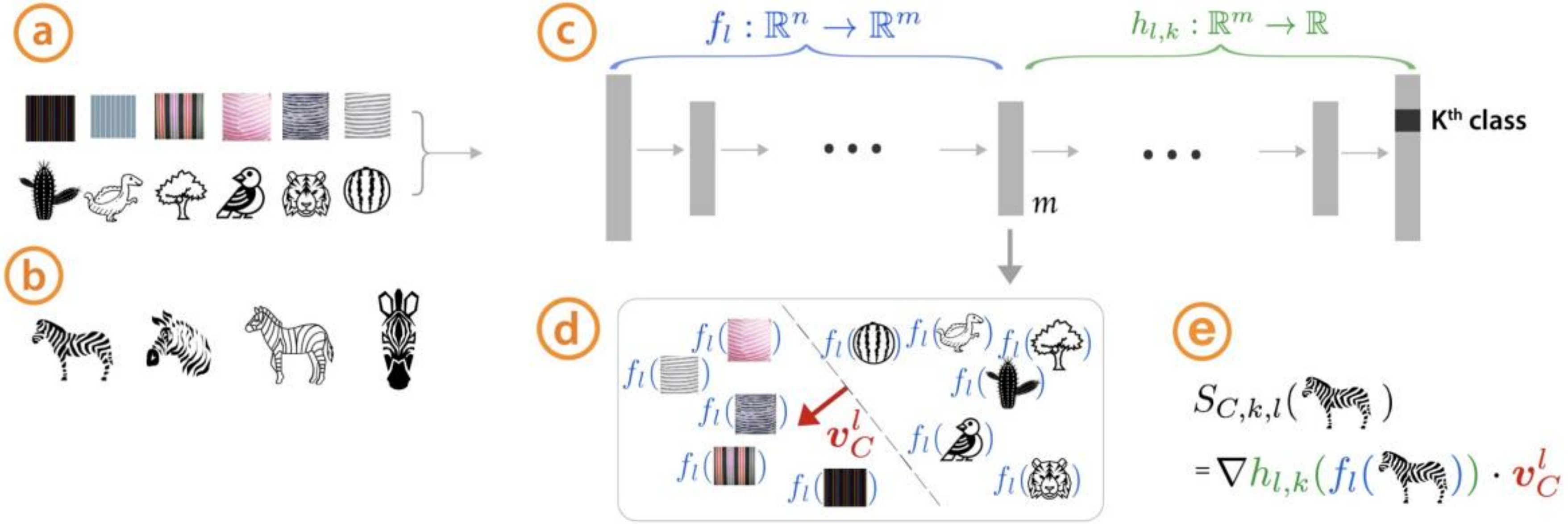
- “interpretation” of ML model $g : E_m \to E_h$
- If g is linear, linear interpretability
- CAV : E_m과 E_h사이를 변환하는 방법

TCAV


a
b
c
$f_l : \mathbb{R}^n \to \mathbb{R}^m$
$h_{l,k} : \mathbb{R}^m \to \mathbb{R}$
K^th class
m
d
f_l
v_C^l
e
$S_{C,k,l}(\ )$
$= \nabla h_{l,k}(f_l(\ )) \cdot v_C^l$

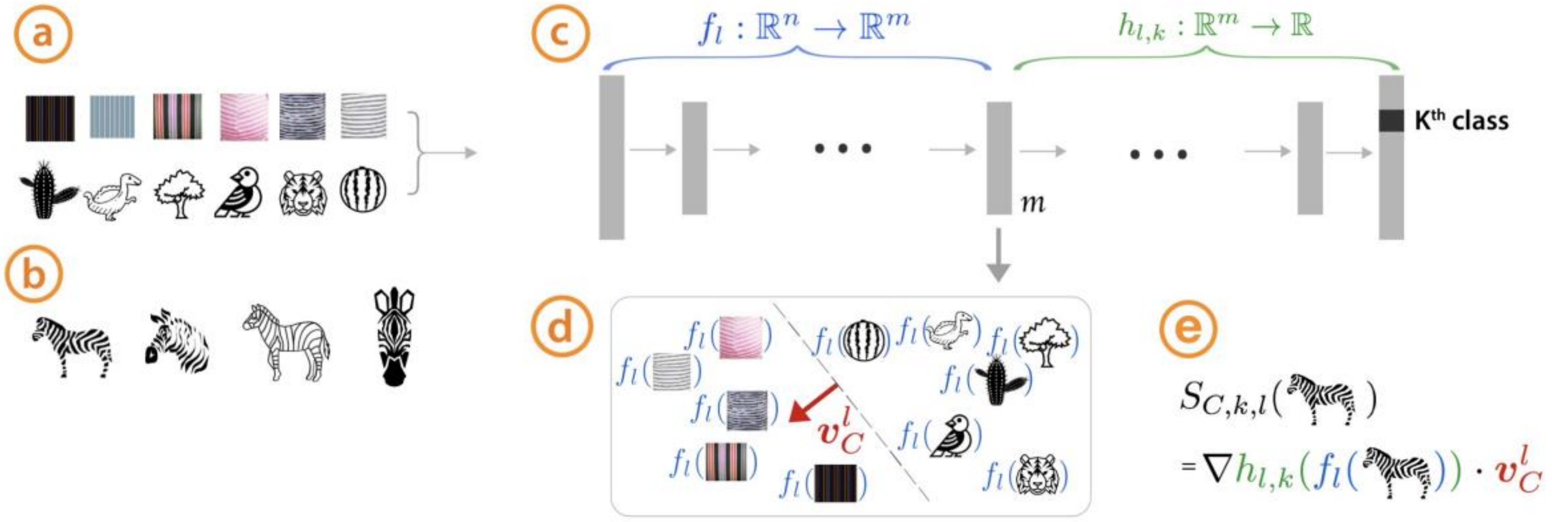
TCAV

- User defined concept : ‘striped’(줄무늬)
- a : 줄무늬에 해당하는 concept + (그에 반하는) random sample
- b : label이 지정된 학습 데이터(for the studied class(zebras))
- C : 훈련된 네트워크
- a, b, c로 인해 TCAV는 특정 클래스에 어떤 컨셉이 갖는 sensitivity를 quantify(정량화)할 수 있음.


a
b
c
$f_l : \mathbb{R}^n \to \mathbb{R}^m$
$h_{l,k} : \mathbb{R}^m \to \mathbb{R}$
K^{th} class
m
d
$f_l(\ )$
v_C^l
e
$S_{C,k,l}(\ )$
$= \nabla h_{l,k}(f_l(\ )) \cdot v_C^l$

TCAV

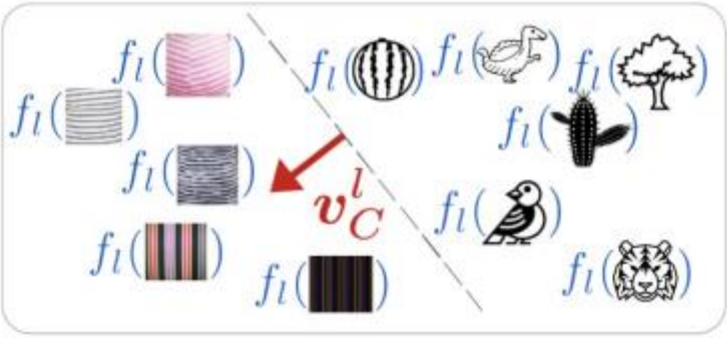
- d : CAVs는 concept's example의 activation값과 다른 레이어의 example의 activation을 구분하는 linear classifier(SVM, logistic regression)를 학습. CAV는 분류 경계선에 orthogonal한 벡터(red arrow).
- e : 어떤 클래스에 대해, TCAV는 directional derivative $S_{C,k,l}(x)$를 사용하여 conceptual sensitivity를 quantify(정량화)


a
b
c
$f_l : \mathbb{R}^n \rightarrow \mathbb{R}^m$
$h_{l,k} : \mathbb{R}^m \rightarrow \mathbb{R}$
K^th class
m
d
f_l
v_C^l
e
$S_{C,k,l}(\ )$
$= \nabla h_{l,k}(f_l(\ )) \cdot v_C^l$

Binary linear classifier

- Concept(P)와 concept이 아닌 것(N)에 의해 생성되는 두 activation space를 분리하는 classifier
- Positive, Negative
- l : neural activation layer (특정 layer)
- Layer l의 neuron의 개수 m
- conv-net layer must be flattened so width w, height h, channels c becomes a vector a of m = w x h x c activations.
- 코드에서는 SVM이나 logistic regression을 사용
- Ex) y= w0 + w1x1   /// CAV = weight vector

$\{f_l(\boldsymbol{x}) : \boldsymbol{x} \in N\}$



$\{f_l(\boldsymbol{x}) : \boldsymbol{x} \in P_C\}$

$S_{C,k,l}(x)$: Conceptual Sensitivity

- Saliency는 pixel (a,b)에서의 변화에 따른 output class k에의 sensitivity를 도출하기 위해 derivative를 사용
- $h_k(x)$: data point x의 class k에 대한 logit
- $x_{a,b}$: data x의 (a,b)위치의 픽셀

$$\frac{\partial h_k(\boldsymbol{x})}{\partial \boldsymbol{x}_{a,b}}$$

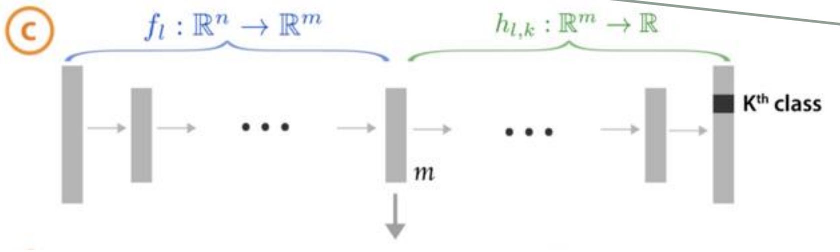
- C : concept
- 아무 layer나 적용 가능.
- Per-feature metric(per-pixel saliency map)이 아니라 per-concept scalar quantity이다.

$$\begin{aligned} S_{C,k,l}(\boldsymbol{x}) &= \lim_{\epsilon \to 0} \frac{h_{l,k}(f_l(\boldsymbol{x}) + \epsilon \boldsymbol{v}_C^l) - h_{l,k}(f_l(\boldsymbol{x}))}{\epsilon} \\ &= \underline{\nabla h_{l,k}(f_l(\boldsymbol{x}))} \cdot \boldsymbol{v}_C^l, \end{aligned} \qquad (1)$$

Activation 변화량

$$\boldsymbol{v}_C^l \in \mathbb{R}^m$$

$$h_{l,k} : \mathbb{R}^m \to \mathbb{R}.$$


c
$f_l : \mathbb{R}^n \to \mathbb{R}^m$
$h_{l,k} : \mathbb{R}^m \to \mathbb{R}$
K^th class
m

TCAV score

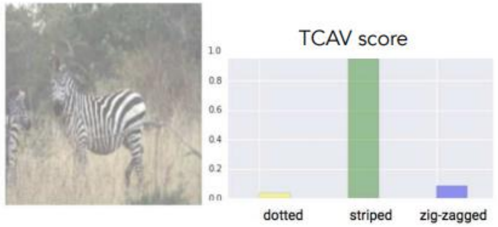
- CAV에 의해 학습된 high level concept에 대한 모델의 prediction의 민감도(sensitivity)를 측정.
- Ex. 얼룩말 분류 모델에서 “striped”라는 concept이 “zebra”라는 prediction에 얼마나 영향을 미치는가?

- TCAV 목표(장점)
  - -Accessibility : ML 전문가가 아니어도 분석 가능.
  - -Customization : 어떤 Concept이든 사용 가능.
  - -Plug-in readiness : 이미 학습된 ML 모델을 재학습하거나 수정할 필요가 없다.
  - -Global quantification : 한번의 정량적 측정으로 전체 class 해석 가능

- 의 부호에만 영향을 받는다. Magnitude를 고려한 다른 측정항목을 사용할 수도 있음.

$$\mathrm{TCAV}_{\mathrm{Q}_{C,k,l}} = \frac{|\{\boldsymbol{x} \in X_k : S_{C,k,l}(\boldsymbol{x}) > 0\}|}{|X_k|}$$

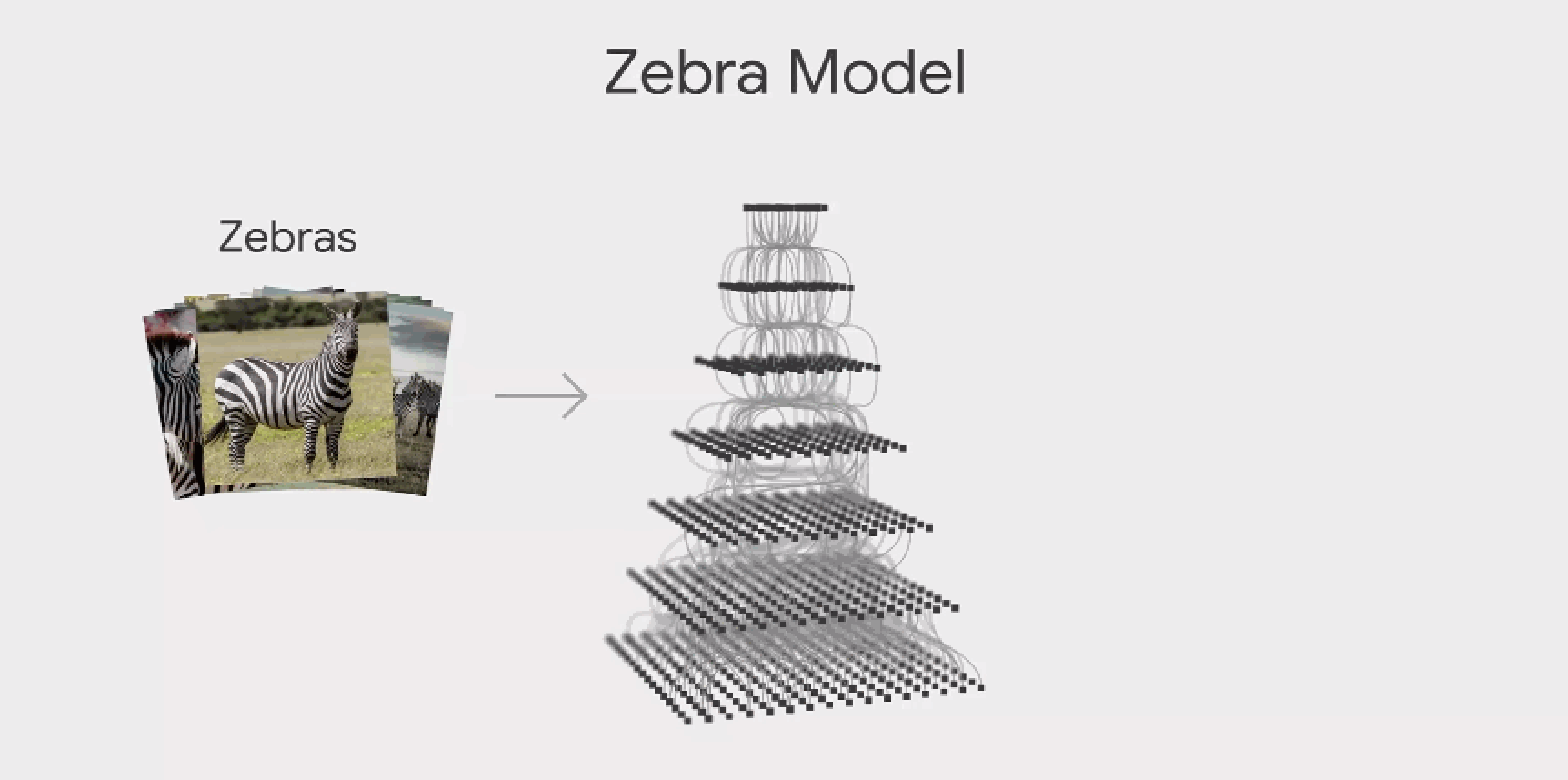
TCAV



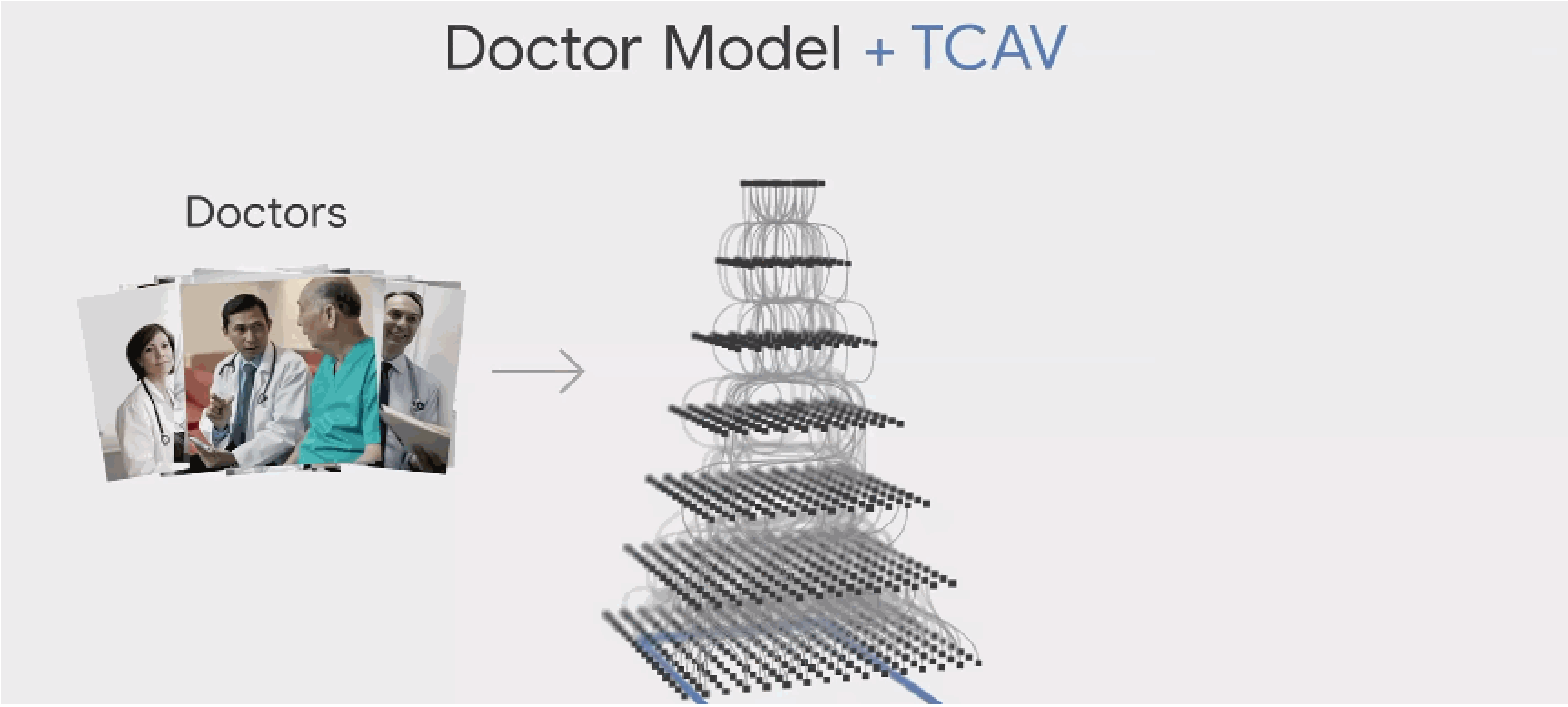
zebra-ness →

striped CAV →

$$\frac{\partial p(z)}{\partial \boldsymbol{v}_C^l} = S_{C,k,l}(\boldsymbol{x})$$



TCAV



통계적 유의성(statistically significance test)

- TCAV의 함정 : 의미 없는 CAV를 학습할 수 있다.
- 무작위로 선택한 이미지 세트를 사용하면 여전히 CAV가 생성된다.
- CAV를 한번 훈련하는 것이 아니라, random examples N의 단일 배치, 여러 번 훈련(일반적으로 500)
- 의미 있는 concept은 훈련 실행 전반에 걸쳐 TCAV score가 일관성을 보여야 한다.
- 0.5의 TCAV점수 = 귀무가설 기각(=통계적으로 유의)

부산대학교 PUSAN NATIONAL UNIVERSITY

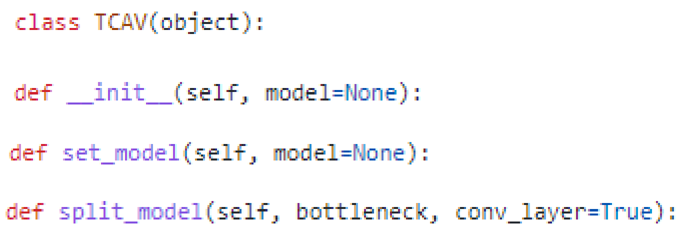
TCAV extensions: Relative TCAV

- 의미적으로 연관되어 있지만 다른 컨셉(갈색머리 vs 검은 머리) : 직교와는 거리가 먼 CAV를 생성. 이를 이용하여 해석에 도움

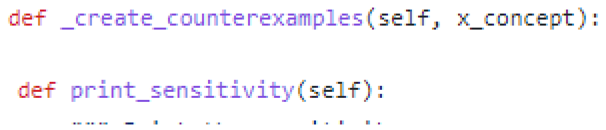
- Relative CAVs를 이용하여 세분화된 비교 수행

- CAVs : weight matrix (concept 개수 x layer 뉴런 개수)

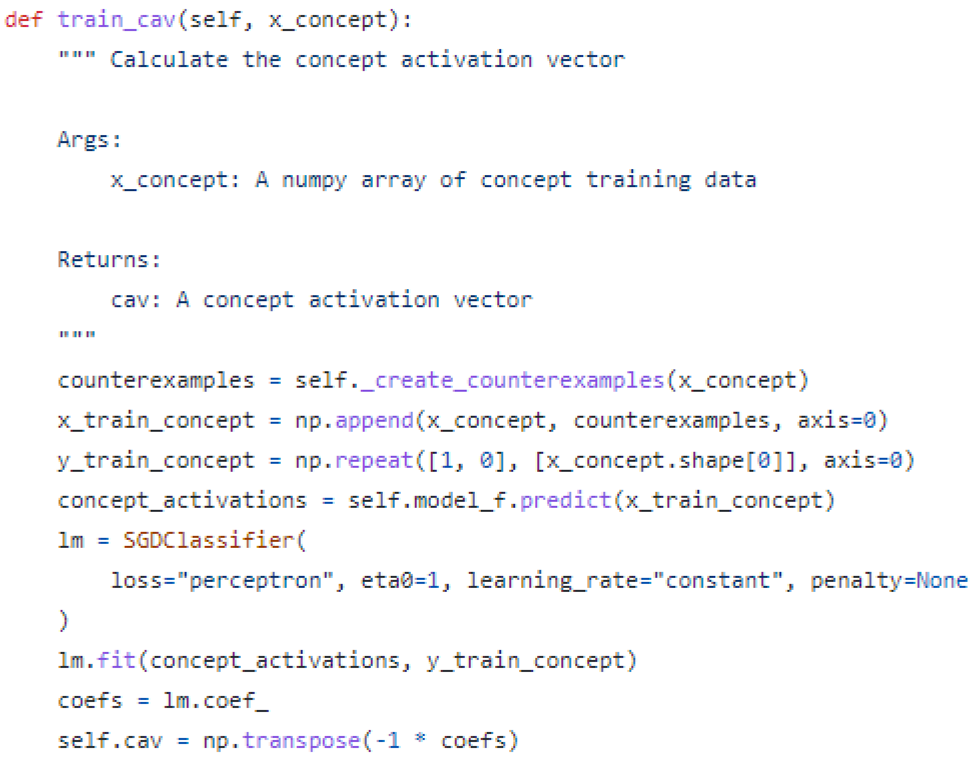
TCAV code for understanding



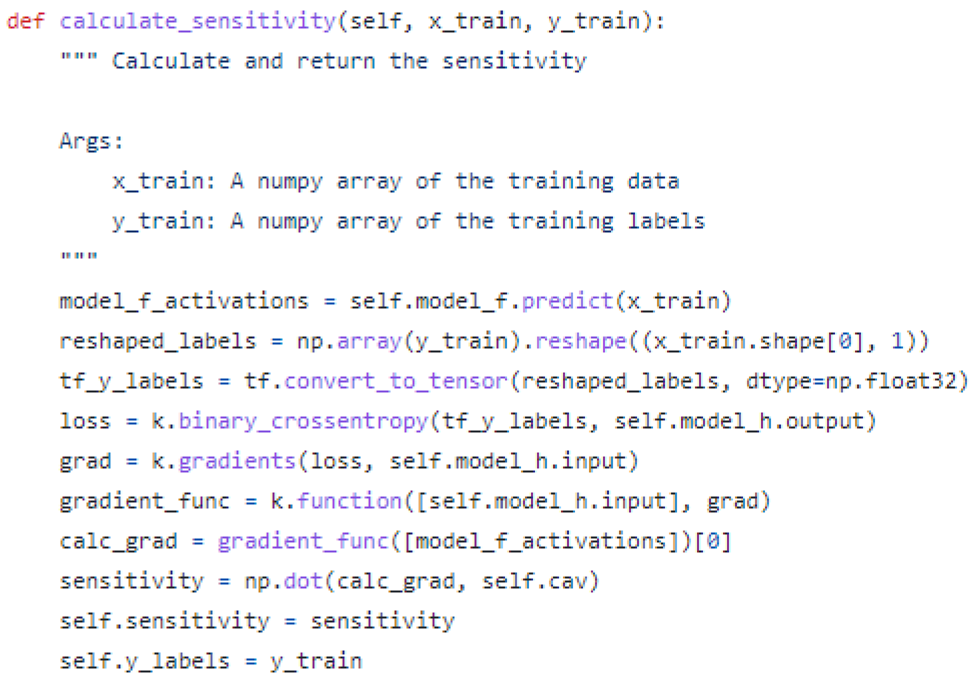
```
class TCAV(object):

 def __init__(self, model=None):

def set_model(self, model=None):

def split_model(self, bottleneck, conv_layer=True):
```



```
def _create_counterexamples(self, x_concept):

 def print_sensitivity(self):
```



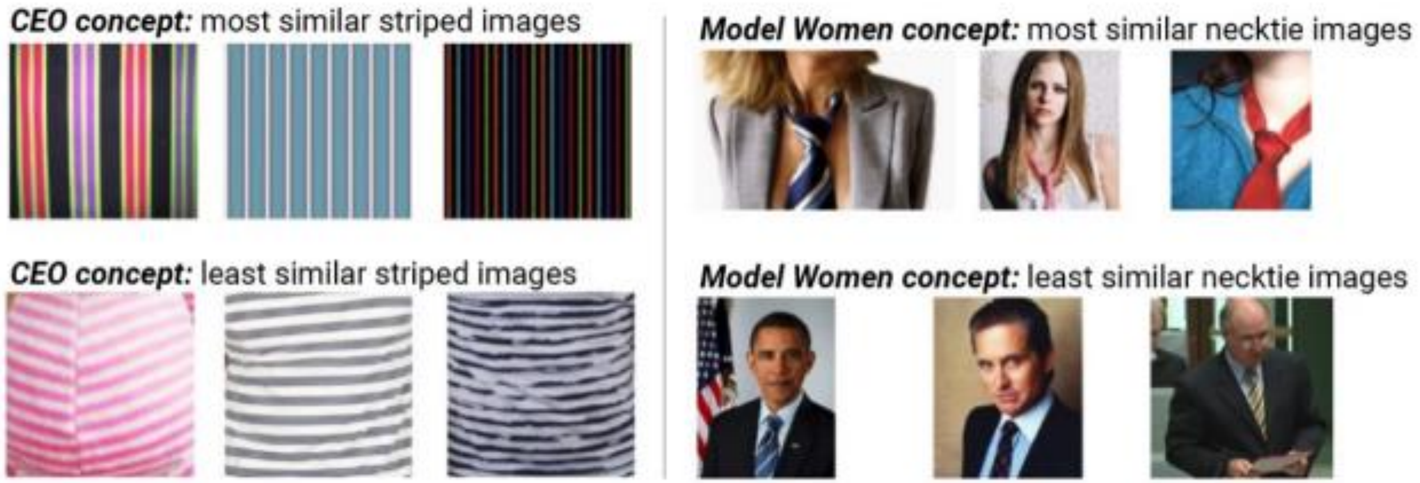
```
def train_cav(self, x_concept):
    """ Calculate the concept activation vector

    Args:
        x_concept: A numpy array of concept training data

    Returns:
        cav: A concept activation vector
    """
    counterexamples = self._create_counterexamples(x_concept)
    x_train_concept = np.append(x_concept, counterexamples, axis=0)
    y_train_concept = np.repeat([1, 0], [x_concept.shape[0]], axis=0)
    concept_activations = self.model_f.predict(x_train_concept)
    lm = SGDClassifier(
        loss="perceptron", eta0=1, learning_rate="constant", penalty=None
    )
    lm.fit(concept_activations, y_train_concept)
    coefs = lm.coef_
    self.cav = np.transpose(-1 * coefs)
```



```
def calculate_sensitivity(self, x_train, y_train):
    """ Calculate and return the sensitivity

    Args:
        x_train: A numpy array of the training data
        y_train: A numpy array of the training labels
    """
    model_f_activations = self.model_f.predict(x_train)
    reshaped_labels = np.array(y_train).reshape((x_train.shape[0], 1))
    tf_y_labels = tf.convert_to_tensor(reshaped_labels, dtype=np.float32)
    loss = k.binary_crossentropy(tf_y_labels, self.model_h.output)
    grad = k.gradients(loss, self.model_h.input)
    gradient_func = k.function([self.model_h.input], grad)
    calc_grad = gradient_func([model_f_activations])[0]
    sensitivity = np.dot(calc_grad, self.cav)
    self.sensitivity = sensitivity
    self.y_labels = y_train
```

TCAV Result

- Sorting images with CAVs
- Concept과의 relation을 기준으로 이미지를 나열
- CAV가 concept의 direction을 벡터로 인코딩하기 때문에, cosine similarity 계산 가능
- $v_c^l \in \mathbb{R}^m$, $f_l(x_i)$ cosine similarity 계산하여 나열


CEO concept: most similar striped images
Model Women concept: most similar necktie images
CEO concept: least similar striped images
Model Women concept: least similar necktie images

TCAV Result

- Sorting images with CAVs

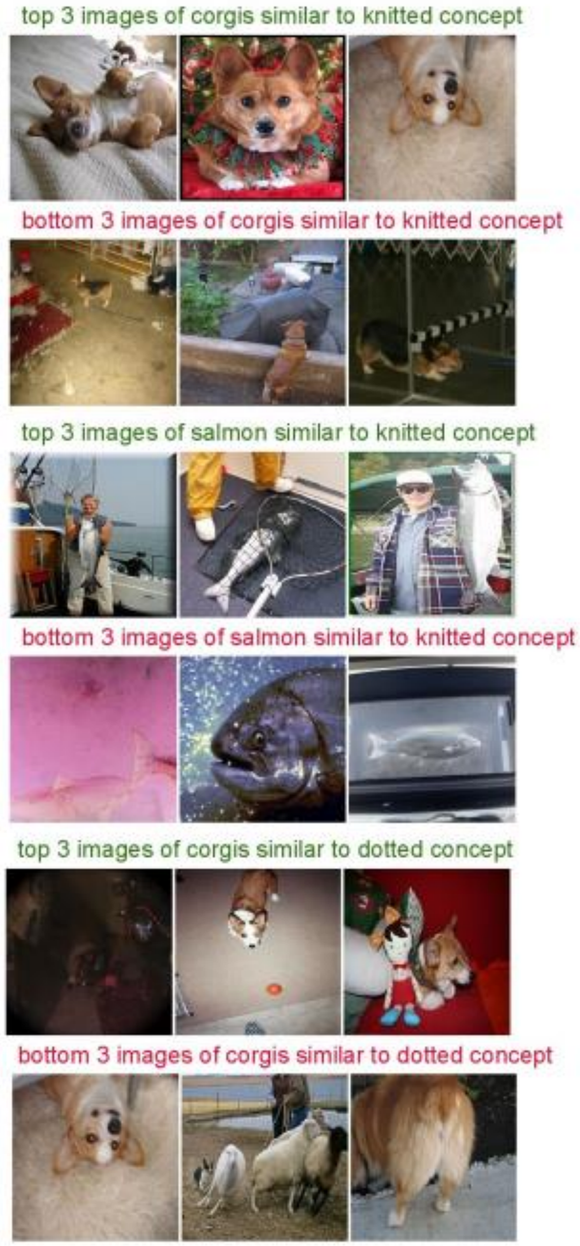

top 3 images of corgis similar to knitted concept
bottom 3 images of corgis similar to knitted concept
top 3 images of salmon similar to knitted concept
bottom 3 images of salmon similar to knitted concept
top 3 images of corgis similar to dotted concept
bottom 3 images of corgis similar to dotted concept

Figure 15. Additional Results: Sorting Images with CAVs

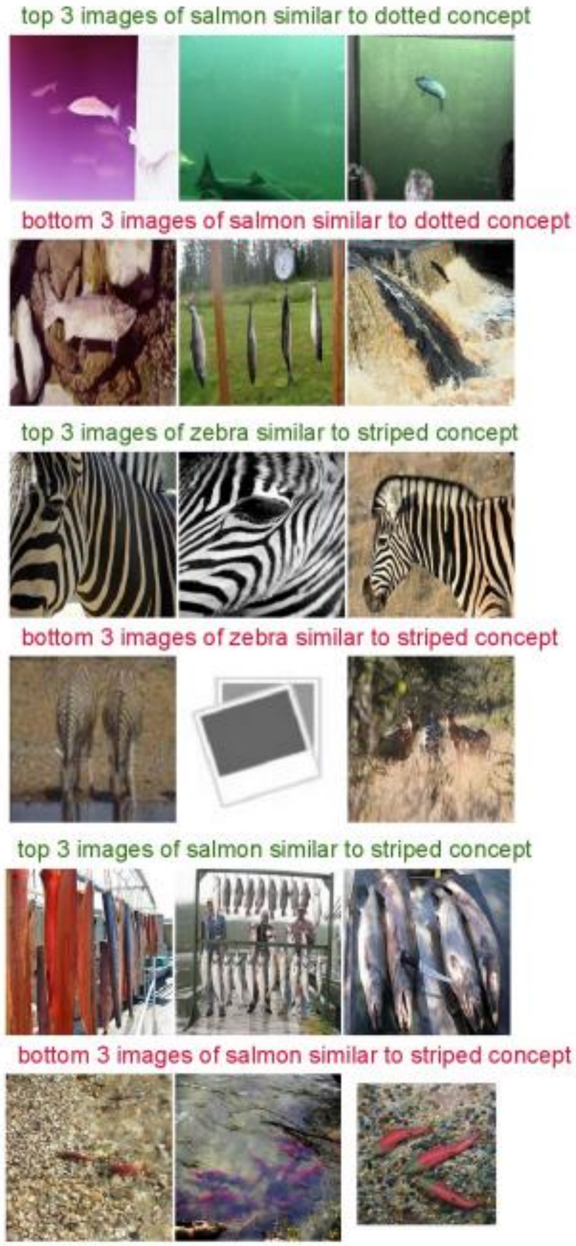

top 3 images of salmon similar to dotted concept
bottom 3 images of salmon similar to dotted concept
top 3 images of zebra similar to striped concept
bottom 3 images of zebra similar to striped concept
top 3 images of salmon similar to striped concept
bottom 3 images of salmon similar to striped concept

Figure 16. Additional Results: Sorting Images with CAVs

Empirical Deep Dream

- CAV + empirical deep dream(2015, LUCID(2017))
- Layer 내에서 identify and visualize interesting directions를 반영

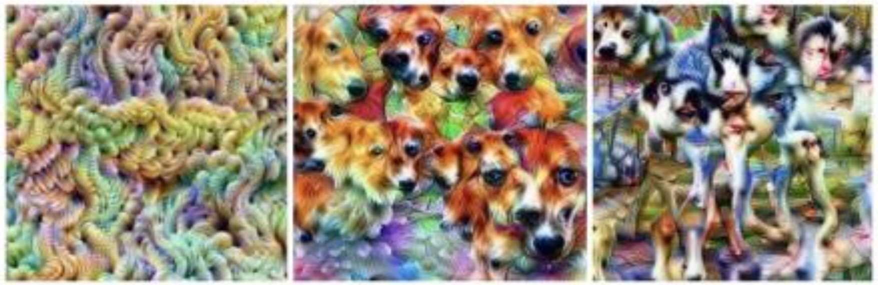


Figure 3. Empirical Deepdream using knitted texture, corgis and Siberian huskey concept vectors (zoomed-in)

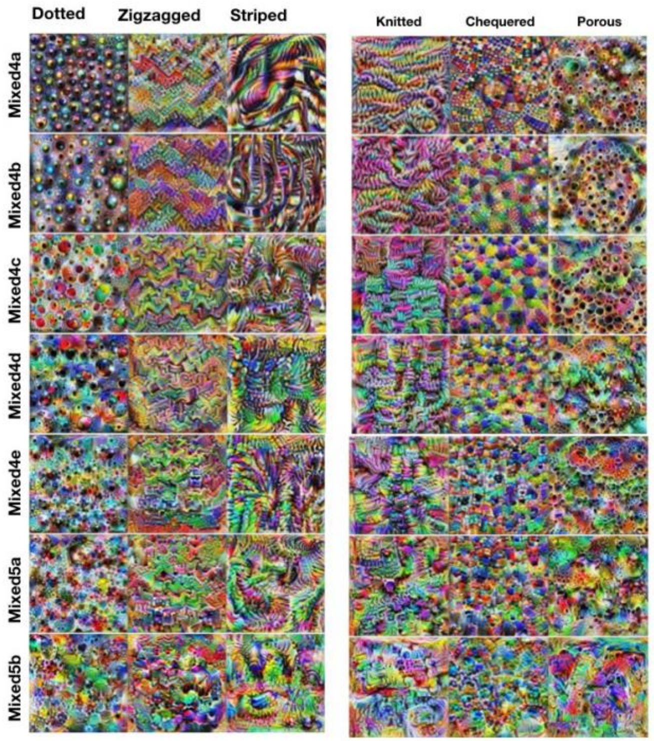


Figure 14. Empirical deepdream using CAVs for each layer in Googlenet.

TCAV Result

- TCAV for where concepts are learned
- 각 concept에 해당하는 linear classifier의 accuracy를 각 layer마다 측정.
- 단순한 concept(색, texture)는 lower-layer에서 학습
- 추상적/복잡한 concept(사람, 물체)는 higher-layer에서 학습

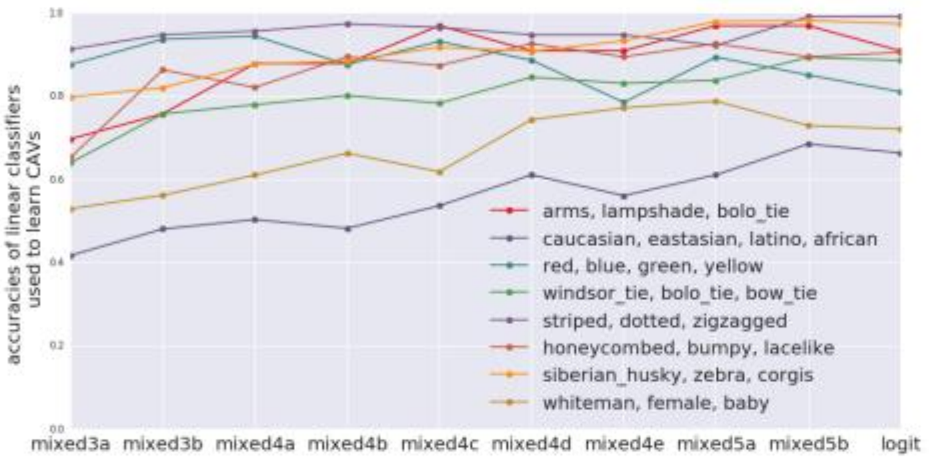


Figure 5. The accuracies of CAVs at each layer. Simple concepts (e.g., colors) achieve higher performance in lower-layers than more abstract or complex concepts (e.g. people, objects)

TCAV Result

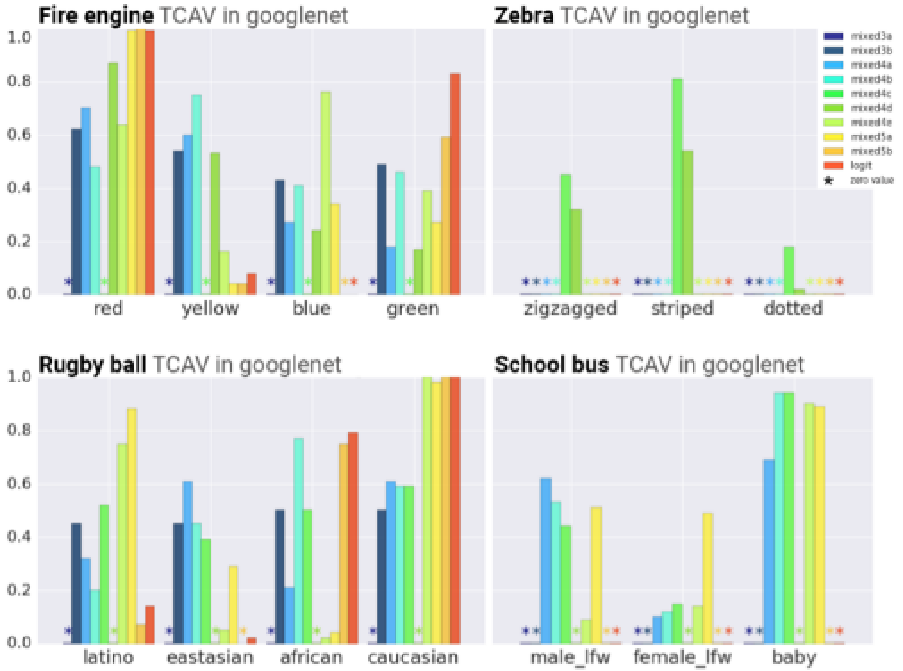
- Relative TCAV
- GoogleNet의 모든 layer, Inception V3 마지막 3 layer
- Concept의 중요도를 표시
- Gender나 race로 따로 학습을 하지 않아도 해당 concept에 대해서 sensitive하다.

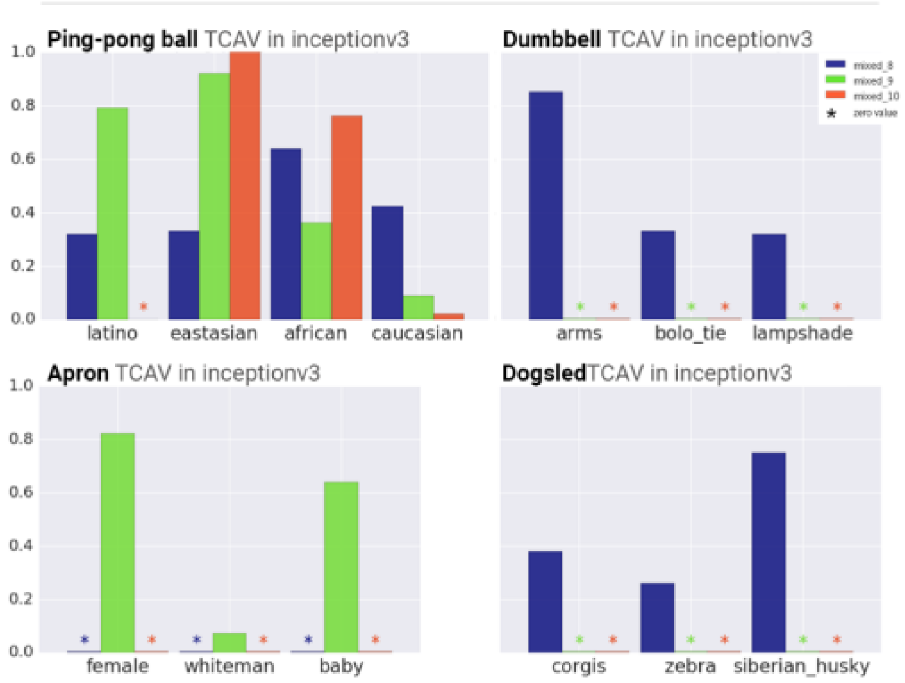
(race: final layer에 가까울수록 강한 중요도, texture: earlier layer일수록 강한 중요도)

Fire engine:소방차

Ping-pong ball:탁구공

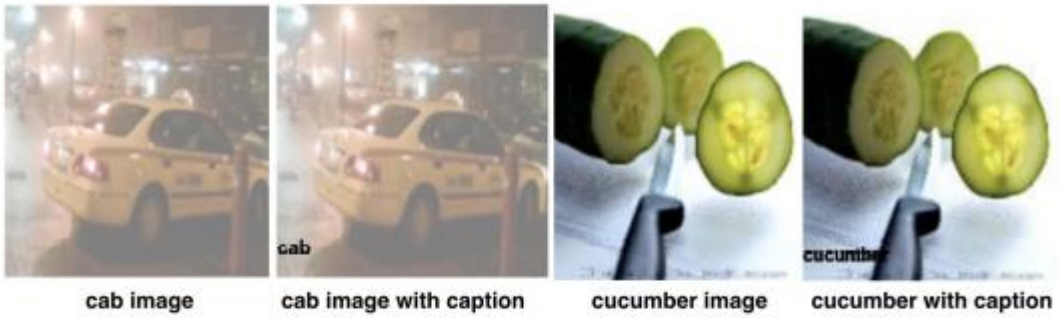
Dogsled:개썰매





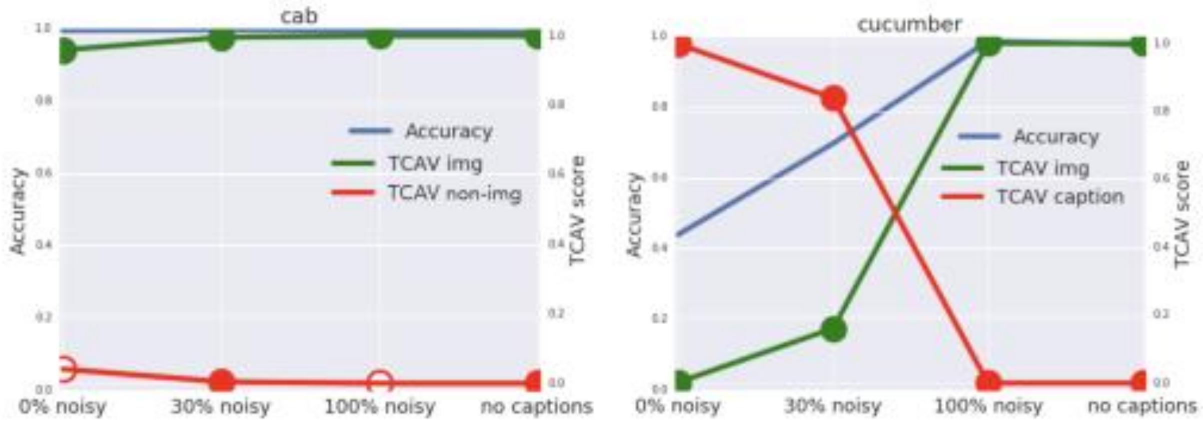
TCAV Result

- Controlled experiment with ground truth
- Noisy caption이 있는 이미지를 생성하여 TCAV의 classification accuracy 추출
- Cab 분류: caption concept 보다 이미지 concept 사용.
- Cucumber 분류 : caption concept 참고. 이미지 concept의 분류 성능은 noisy한 정도에 반비례하여 향상



cab image cab image with caption cucumber image cucumber with caption

Figure 6. A controlled training set: Regular images and images with captions for the cab and cucumber class.



TCAV in medical image

- DR(당뇨병성 망막병증)
- 미세동맥류(MA)
- 범망막 레이저 흉터(PRP)

- 모델이 level1을 level2로 예측하는 오류가 있음.
- HMA를 가지고 있는 경우 level2가 많다.
- HMA의 영향을 줄이는 방향 고려

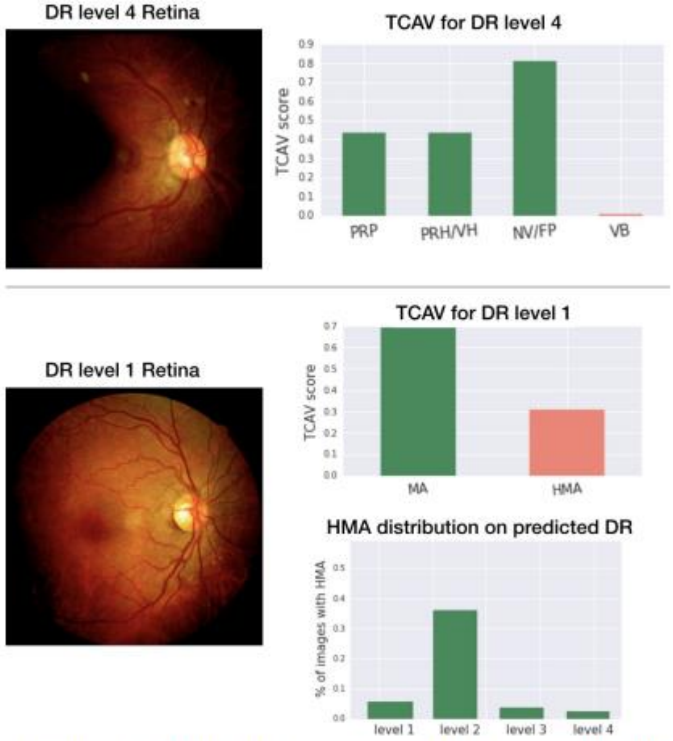


Figure 10. Top: A DR level 4 image and TCAV results. TCAVQ is high for features relevant for this level (green), and low for an irrelevant concept (red). Middle: DR level 1 (mild) TCAV results. The model often incorrectly predicts level 1 as level 2, a model error that could be made more interpretable using TCAV: TCAVQs on concepts typically related to level 1 (green, MA) are high in addition to level 2-related concepts (red, HMA). Bottom: the HMA feature appears more frequently in DR level 2 than DR level 1.

TCAV in medical image(another paper)

- 폐섬유증
- “honeycomb(벌집)”

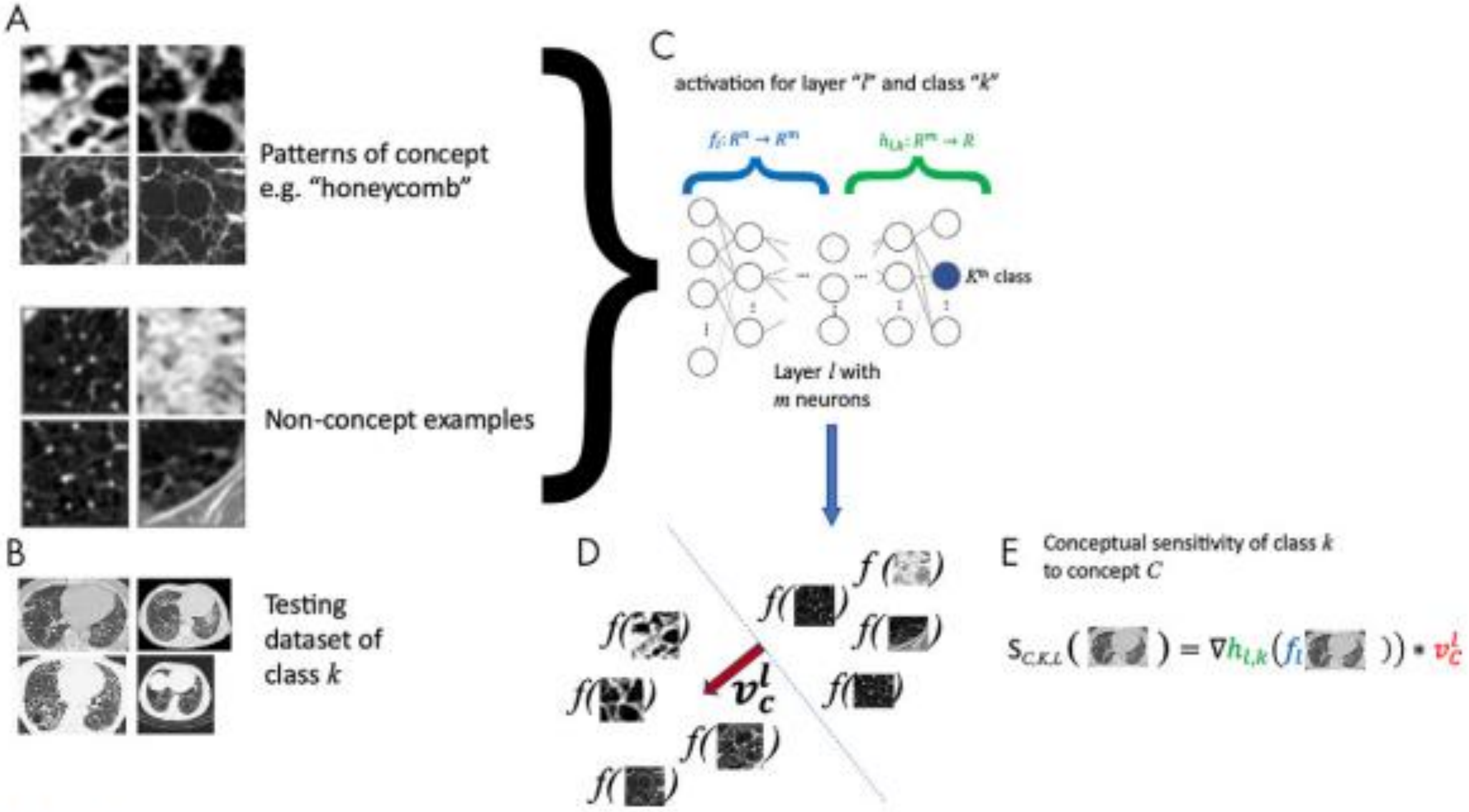


Figure 4: *A*, Testing with concept activation vectors (TCAVs) requires a set of samples characterizing the concept (eg, “honeycomb pattern,” a set of “nonconcept” examples, which are not related to the concept being studied), *B*, a testing dataset of the class *k* of interest (eg, idiopathic pulmonary fibrosis), and, *C*, a complex model *f* (eg, neural network) that one desires to interpret, and which has been trained to perform classification of these classes. *D*, A linear model is built from the concept and nonconcept samples using model *f*, by employing model *f* to generate classification labels for the concept and nonconcept samples. *E*, From the resulting linear model, separating concept from nonconcept examples (dotted line in *D*), its main perpendicular direction v_c^l (red arrow in *D*) can be obtained to assess the sensitivity of model *f* to concept C at layer *l* by quantifying changes to the activations of model *f* in the v_c^l direction.

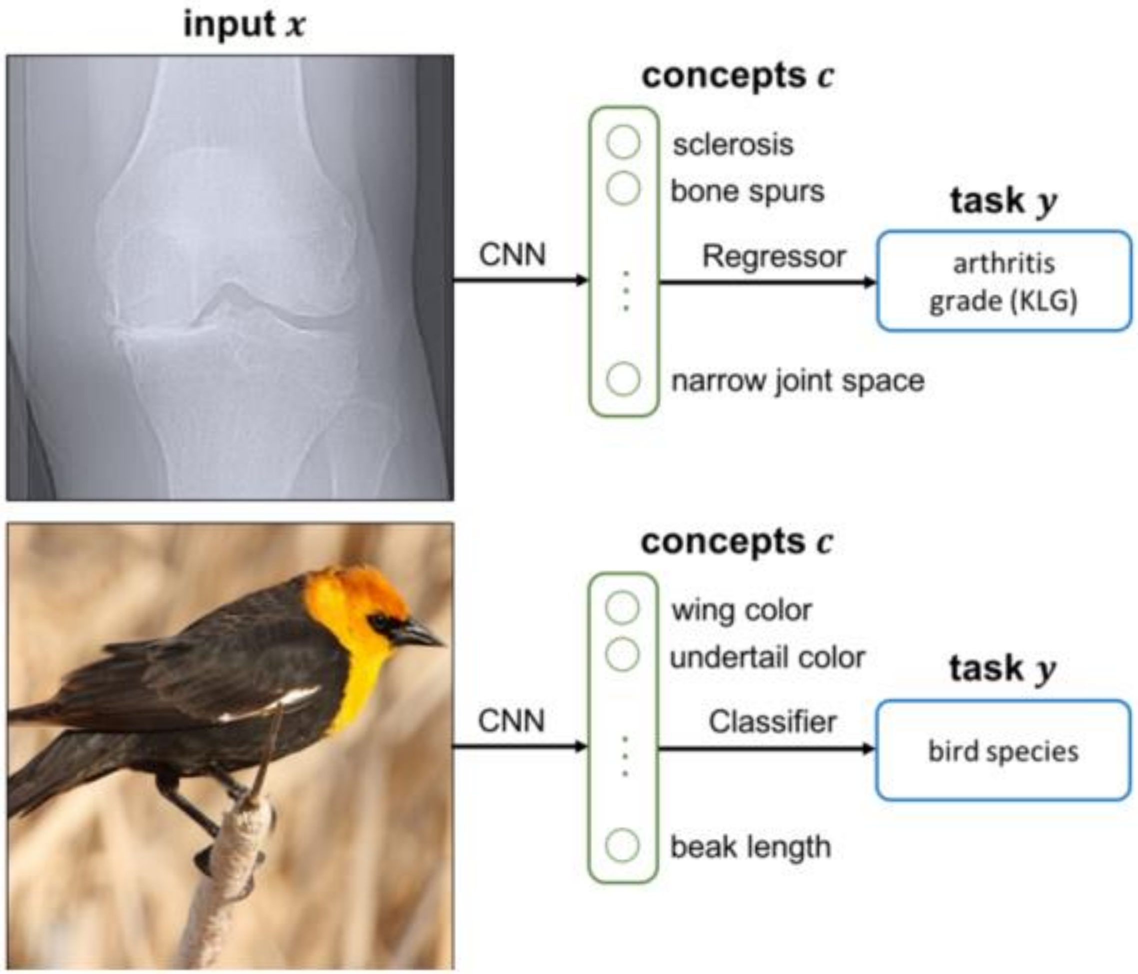
Conclusion

- Human-friendly concept
- 사용자 정의 concept 이 분류결과에 중요한 정도를 정량화 (CAV)
- 적절한 concept 을 선정하여 인공지능에게도 도움이 되고, 인간의 이해도 돕는 보조장치

XAI

- Saliency Map(2013) : pixelwise 1차 미분
- LRP(2015)
- CAM(2015) : 마지막 conv layer에 GAP
- LIME(2016)
- Grad-CAM(2016) : GAP제거, weight를 gradient 기준으로 변경
- SHAP(2017) : 게임이론
- TCAV(2018) : Concept
- Concept SHAP(2019)
- Concept Bottleneck Models(2020)

Concept Bottleneck Models(2020)



Concept Bottleneck Models

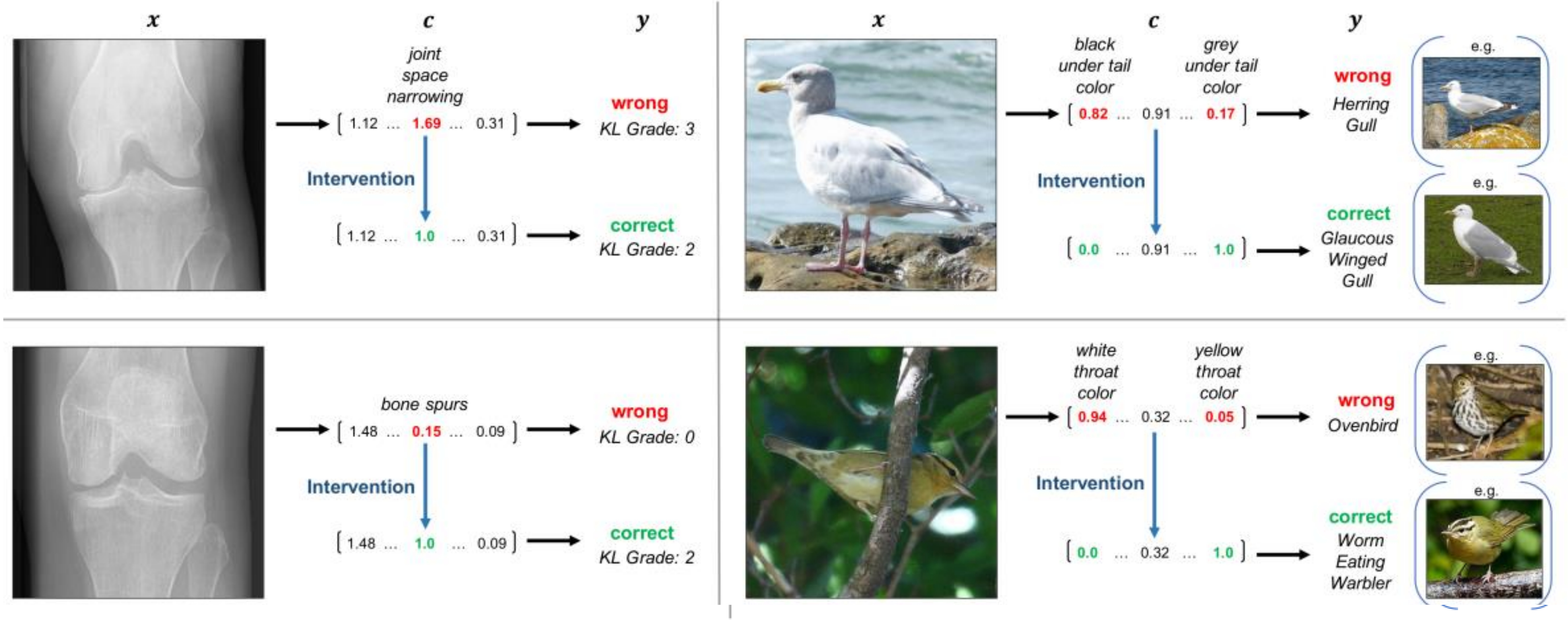


Figure 3. Successful examples of test-time intervention, where intervening on a single concept corrects the model prediction. Here, we show examples from independent bottleneck models. **Right**: For CUB, we intervene on concept groups instead of individual binary concepts. The sample birds on the right illustrate how the intervened concept distinguishes between the original and new predictions.