Machine Learning

Lecture 16: Explainable AI (II)

Nevin L. Zhang

Department of Computer Science and Engineering
The Hong Kong University of Science and Technology

This set of notes is based on internet resources and references listed at the end.

Outline

1 Pixel-Level Explanations

2 Feature-Level Explanations
- LIME
- SHAP Values

3 Concept-Level Explanations
- TCAV
- ACE

4 Instance-Level Explanations
- Counterfactual Explanations

Outline

1 Pixel-Level Explanations

2 Feature-Level Explanations
- LIME
- SHAP Values

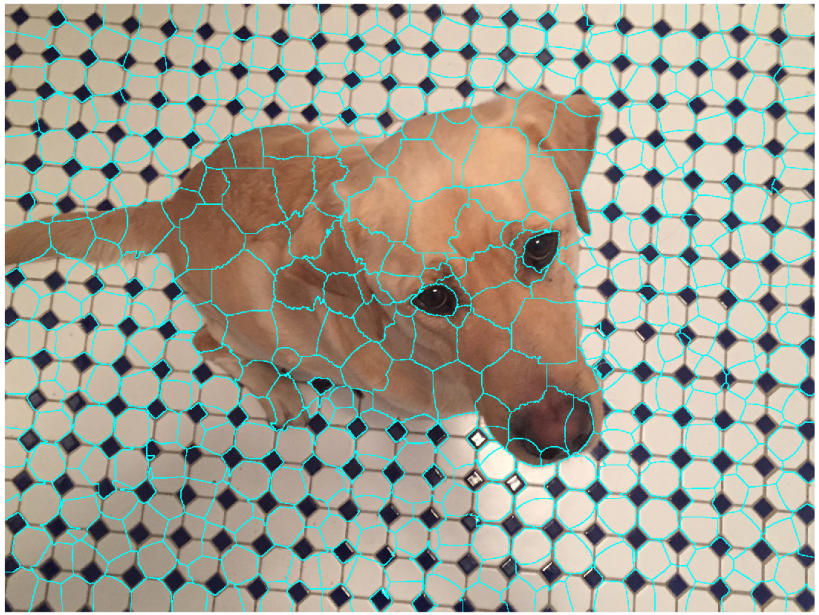
3 Concept-Level Explanations
- TCAV
- ACE

4 Instance-Level Explanations
- Counterfactual Explanations

Features

In this lecture, **features** refer to

- **Super-pixels** in images data obtained by standard image segmentation algorithms such as SLIC (Achanta *et al.* 2012).



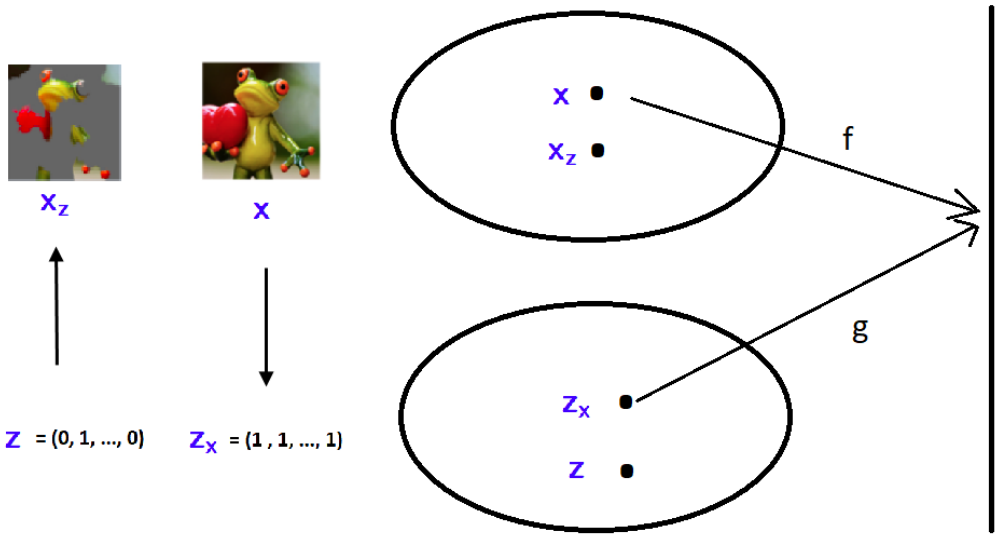
- **Presence or absence of words** in text data.
- **Input variables** in tabular data.

Interpretable Data Representations

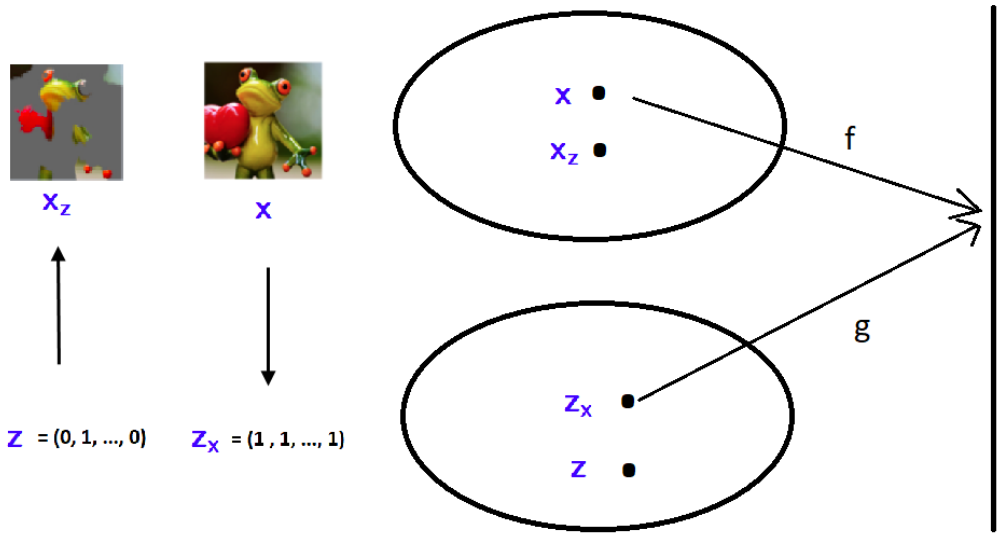
- The original representation of an image $\mathbf{x}$ is a tensor of pixels.
- A **simplified/interpretable representation** $\mathbf{z_x}$ is a binary vector over the M super-pixels, whose components are all 1's.
- For sparse binary vector $\mathbf{z} \in \{0,1\}^M$ correspond to an image that consists of a subset of the super-pixels. It is denoted as $\mathbf{x_z}$.
- The original and interpretable representations of text are related in a similar manner.

LIME (Ribeiro *et al.* 2016)

- **LIME** stands for Local Interpretable Model-Agnostic Explanations. It is for explaining a binary classifier.
  - $f(\mathbf{x})$: Probability of input $\mathbf{x}$ belonging to the class.
  - LIME explains $f(\mathbf{x})$ using an **surrogate model** $g(\mathbf{z})$, a linear model or a decision tree on the simplified features.
  - The surrogate model should be faithful to $f(.)$ in the neighborhood of $\mathbf{x}$. Ideally, $g(\mathbf{z}_\mathbf{x}) = f(\mathbf{x})$, and $g(\mathbf{z}) \approx f(\mathbf{x}_\mathbf{z})$ if $\mathbf{x}_\mathbf{z}$ is close to $\mathbf{z}$.



LIME (Ribeiro *et al.* 2016)



- The surrogate model g is determined by minimizing:

$$\xi(\mathbf{x}) = \arg\min_{g \in G} \mathcal{L}(f, g, \pi_{\mathbf{x}}) + \Omega(g)$$

G is a family of surrogate models. $\mathcal{L}(f, g, \pi_{\mathbf{x}})$ measures how unfaithful $g(\mathbf{z})$ is in approximating $f(\mathbf{x_z})$ in the neighborhood of $\mathbf{x}$. $\Omega(g)$ is a penalty for model complexity.

LIME with Sparse Linear Model

- Researchers often use sparse linear model for the surrogate model

$$g(\mathbf{z}) = \mathbf{w}^\top \mathbf{z}$$

- In this case, $\Omega(g)$ is the number of non-zero weights. And

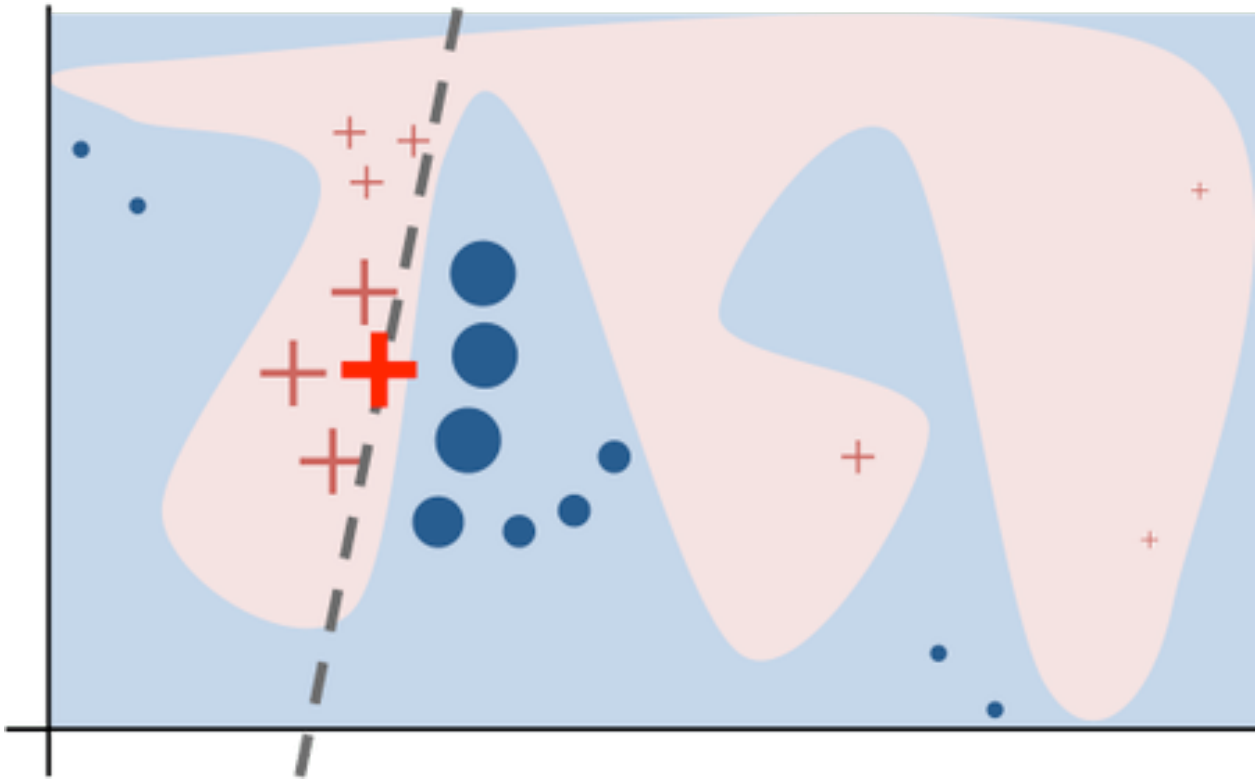
$$\mathcal{L}(f, g, \pi_{\mathbf{x}}) = \sum_{\mathbf{z}:\text{samples around } \mathbf{z}_{\mathbf{x}}} \pi_{\mathbf{x}}(\mathbf{z})(f(\mathbf{x}_{\mathbf{z}}) - g(\mathbf{z}))^2$$

  where $\pi_{\mathbf{x}}(\mathbf{z}) = \exp(-D(\mathbf{z}_{\mathbf{x}}, \mathbf{z})/\sigma)$ is a proximity measure between $\mathbf{x}$ and $\mathbf{x}_{\mathbf{z}}$, and D is L2 distance for images, and cosine distance for text.

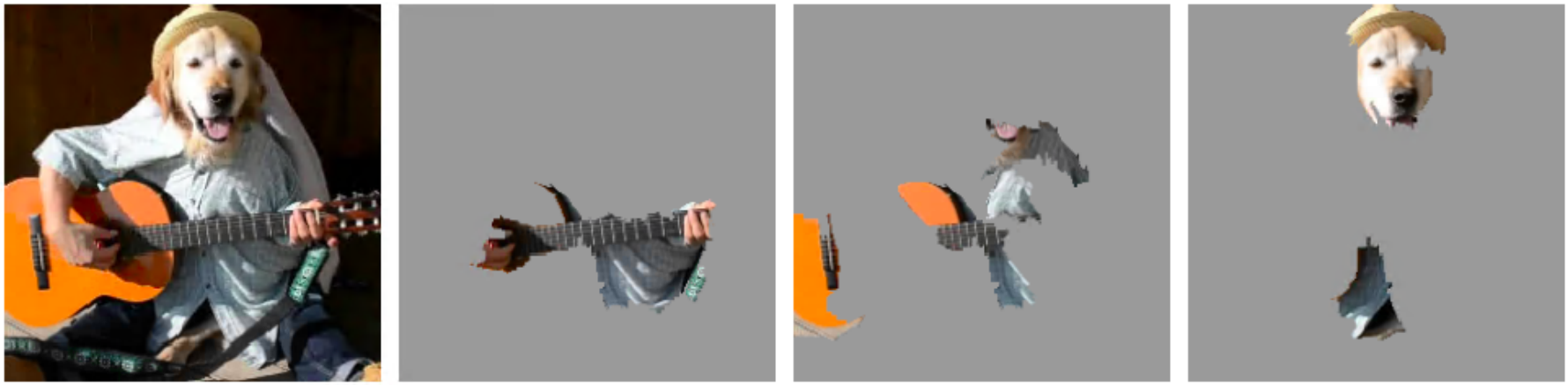
- $\mathcal{L}(f, g, \pi_{\mathbf{x}}) + \Omega(g)$ is minimized using K-LASSO to ensure that no more than K super-pixels are used in the explanation.

LIME Intuition

- Although a model might be very complex globally, it can still be faithfully approximated using a linear model locally.



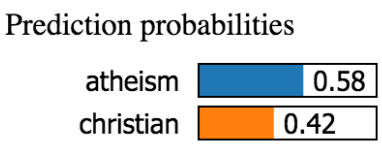
LIME Example

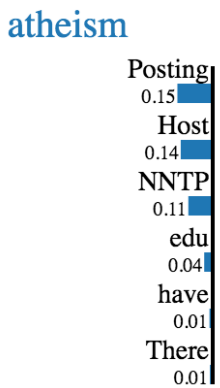


(a) Original Image (b) Explaining *Electric guitar* (c) Explaining *Acoustic guitar* (d) Explaining *Labrador*

Figure 4: Explaining an image classification prediction made by Google's Inception neural network. The top 3 classes predicted are "Electric Guitar" ($p = 0.32$), "Acoustic guitar" ($p = 0.24$) and "Labrador" ($p = 0.21$)

LIME Example





Text with highlighted words

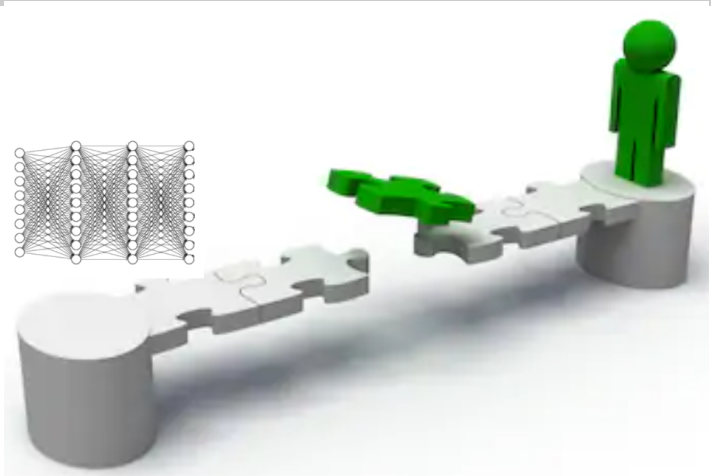
From: johnchad@triton.unm.edu (jchadwic)
Subject: Another request for Darwin Fish
Organization: University of New Mexico, Albuquerque
Lines: 11
NNTP-Posting-Host: triton.unm.edu

Hello Gang,

There have been some notes recently asking where to obtain the DARWIN fish.
This is the same question I have and I have not seen an answer on the
net. If anyone has a contact please post on the net or email me.

LIME reveals that the classification is based on the wrong reasons.

LIME Evaluation (Ribeiro *et al.* 2018)



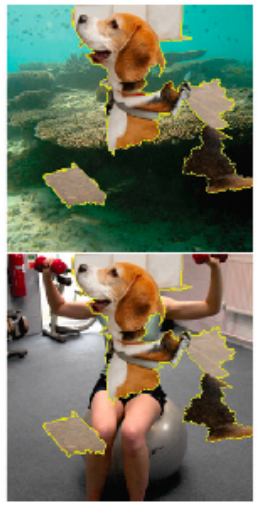
- Local Faithfulness:
  - Train interpretable models f with a small number of features, and check to see how many of them are considered important in g by LIME.
  - Remove some features and see how well prediction changes with g match those with f.
- Interpretability:
  - How much can LIME explanations help users choose a better model.
  - How much can LIME explanation help users improve a classifier by removing features that do not generalize.

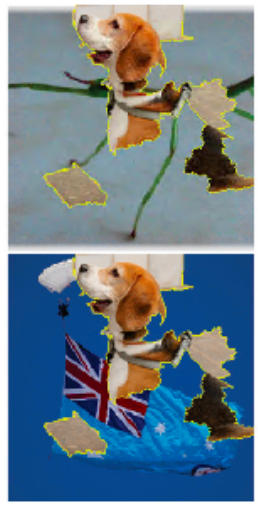
Anchors (Ribeiro *et al.* 2018)

- LIME does not allow human to predict model behaviour on unseen instances.
- **Anchors**: Another model-agnostic explanation method. It gives a rule that sufficiently anchors the prediction locally such that changes to the rest of the feature values of the instance do not matter. In other words, for instances on which the anchor holds, the prediction is (almost) always the same.

(a) Original image

(b) Anchor for "beagle"





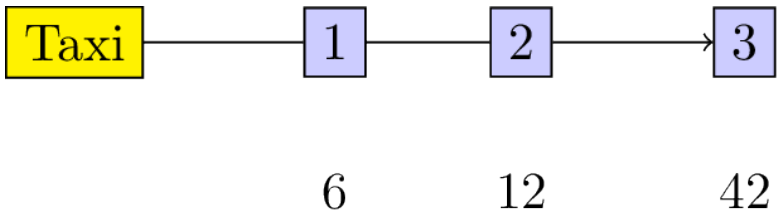
(c) Images where Inception predicts $P(\text{beagle}) > 90\%$

The Shapley values (Wikipedia)

- The Shapley value is a solution concept in cooperative game theory. It was named in honor of Lloyd Shapley, who introduced it in 1951 and won the Nobel Prize in Economics for it in 2012.
- To each cooperative game it assigns a unique distribution (among the players) of a total surplus generated by the coalition of all players.
- The Shapley value is characterized by a collection of desirable properties.

Cooperative Games (Knight 2020)

- A **characteristic function game**: $f : 2^{\{1,\ldots,M\}} \rightarrow \mathbb{R}$
  - For any subset S of players, $f(S)$ is their payoff if they act as a coalition.
- **Question**: What is the fair way to divide the total payoff $f(\{1, \ldots, M\})$

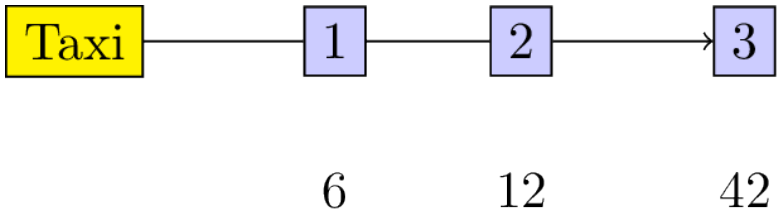


- **Example**: 3 persons share a taxi. Here are the costs for each individual journey:
  - Person 1: 6
  - Person 2: 12
  - Person 3: 42

  How much should each individual contribute?

Cooperative Games: Example

Define a set function



S	$f(S)$
{1}	6
{2}	12
{3}	42
{1, 2}	12
{1, 3}	42
{2, 3}	42
{1, 2, 3}	42

Question: How to divide the total cost 42 among the three persons?

The Shapley Values

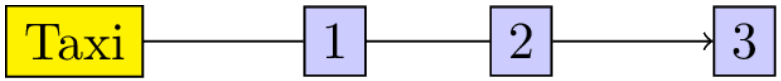
- A fair way to divide the total payoff $f(\{1, \ldots, M\})$ to the players is to use the Shapley values:
  - **The Shapley Value** for player i is:

$$\begin{aligned} \phi_i(f) &= \frac{1}{M!} \sum_{\pi:\text{permutation of}} \Delta_\pi^f(i) \\ &= \frac{1}{M!} \sum_{\pi:\text{permutation of } \{1,\ \ldots,\ M\}} [f(S_\pi^i \cup i) - f(S_\pi^i)] \end{aligned}$$

    - S_π is the set of predecessors of i in π,
    - $\Delta_\pi^f(i)$ is the **marginal contribution** of player i w.r.t π.
  - The Shapley values satisfy:

$$\sum_{i=1}^{M} \phi_i(f) = f(\{1, \ldots, M\})$$

The Shapley Values: Example



6 12 42

π	$\Delta^f_\pi(1)$	$\Delta^f_\pi(2)$	$\Delta^f_\pi(3)$
$\{1, 2, 3\}$	6	6	30
$\{1, 3, 2\}$	6	0	36
$\{2, 1, 3\}$	0	12	30
$\{2, 3, 1\}$	0	12	30
$\{3, 1, 2\}$	0	0	42
$\{3, 2, 1\}$	0	0	42
$\phi_i(f)$	2	5	35

Use of Shapley Values in XAI (Lundberg and Lee 2017)

- Consider **explaining the prediction $f(\mathbf{x})$ of a complex model f on an input $\mathbf{x}$.**
- We regard each feature i as a player. Their joint "payoff" is $f(\mathbf{x})$.
- How do we divide the "payoff" $f(\mathbf{x})$ among the features?
- Answer: Shapley values.

- To apply Shapley values, we need a set function $f_{\mathbf{x}} : 2^{\{1,\ldots,M\}} \rightarrow \mathbb{R}$.
  - Obviously, $f_{\mathbf{x}}(\{1,\ldots,M\}) = f(\mathbf{x})$.
  - How about $f_{\mathbf{x}}(S)$ for a proper subset S of $\{1,\ldots,M\}$?

Use of Shapley Values in XAI

- Given an input $\mathbf{x}$, $\mathbf{x}_S$ denotes the subset of feature values for features in S. Define $f_{\mathbf{x}}(S)$ using conditional expectation:

$$f_{\mathbf{x}}(S) = E_{\mathbf{x}'}[f(\mathbf{x}')|\mathbf{x}'_S = \mathbf{x}_S]$$

- To estimate $f_{\mathbf{x}}(S)$, sample a set of input examples $\mathbf{x}^1, \ldots, \mathbf{x}^N$ such that $\mathbf{x}^i_s = \mathbf{x}_s$, and set:

$$f_{\mathbf{x}}(S) \approx \frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}^i)$$

- Another way is to independently sample $\mathbf{x}^1_{\bar{S}}, \ldots, \mathbf{x}^N_{\bar{S}}$ for features not in S, and

$$f_{\mathbf{x}}(S) \approx \frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}_S, \mathbf{x}^i_{\bar{S}})$$

Use of Shapley Values in XAI

- Yet another (seemingly more common) way is to set

$$f_{\mathbf{x}}(S) \approx f(\mathbf{x}_S, \mathbf{x}^r_{\bar{S}})$$

where $\mathbf{x}^r_{\bar{S}}$ are reference values for features not in S:

  - For images, researchers usually use 0 (black) as the reference values.
  - For tabular data, the reference values can be data mean, data median, representative training example, etc. (Watzman 2020).

Use of Shapley Values in XAI

- Given a set function $f_{\mathbf{x}} : 2^{\{1,\ldots,M\}} \rightarrow \mathbb{R}$, the **Shapley value for feature** i is:

$$\phi_i(f, \mathbf{x}) = \frac{1}{M!} \sum_{\pi:\text{permutation of } \{1,\ldots,M\}} [f_{\mathbf{x}}(S^i_\pi \cup i) - f_{\mathbf{x}}(S^i_\pi)]$$

  where S_π is the set of predecessors of i in π.

- $\phi_0 = f(\mathbf{x}) - \sum_{i=1}^{M} \phi_i(f, \mathbf{x})$ is the **base value**, the value of f when no feature is present, i.e., $f_{\mathbf{x}}(\emptyset)$.

- **Additive Feature Attribution**:

$$f(\mathbf{x}) = \phi_0 + \sum_{i=1}^{M} \phi_i(f, \mathbf{x})$$

- $\phi_i(f, \mathbf{x})$ is the contribution of feature i to $f(\mathbf{x})$. It is called the **SHapley Additive exPlanation (SHAP) value** of i.

SHAP values are the only ones that satisfy ...

- **Property 1: Local accuracy/Additivity**:

$$f(\mathbf{x}) = \phi_0 + \sum_{i=1}^{M} \phi_i(f, \mathbf{x})$$

- **Property 2: Consistency/Monotonicity**: f and f' are two models. If, for any subset S of features,

$$f'_{\mathbf{x}}(S) - f'_{\mathbf{x}}(S \backslash i) \geq f_{\mathbf{x}}(S) - f_{\mathbf{x}}(S \backslash i), \text{ then } \phi_i(f', \mathbf{x}) \geq \phi_i(f, \mathbf{x}).$$

  If the marginal contributions of feature i are larger in f' than in f, then its contribution in f' should also be larger.

- **Property 3: Missingness**: If $f_{\mathbf{x}}(S \cup i) = f_{\mathbf{x}}(S)$ for any subset S of features, then $\phi_i(f, \mathbf{x}) = 0$.
  If the marginal contributions of feature i are always 0, then its attribution should be 0.

SHAP Value Computation

$$\phi_i(f, \mathbf{x}) = \frac{1}{M!} \sum_{\pi:\text{permutation of } \{1, \ldots, M\}} [f_{\mathbf{x}}(S^i_\pi \cup i) - f_{\mathbf{x}}(S^i_\pi)]$$

- Exact computation of SHAP values is NP hard.
- Solutions:
  - Sampling
  - Kernel SHAP
  - Fast algorithm for special models: TreeExplainer for tree-based modesl

Implementations can be found at `https://github.com/slundberg/shap`.

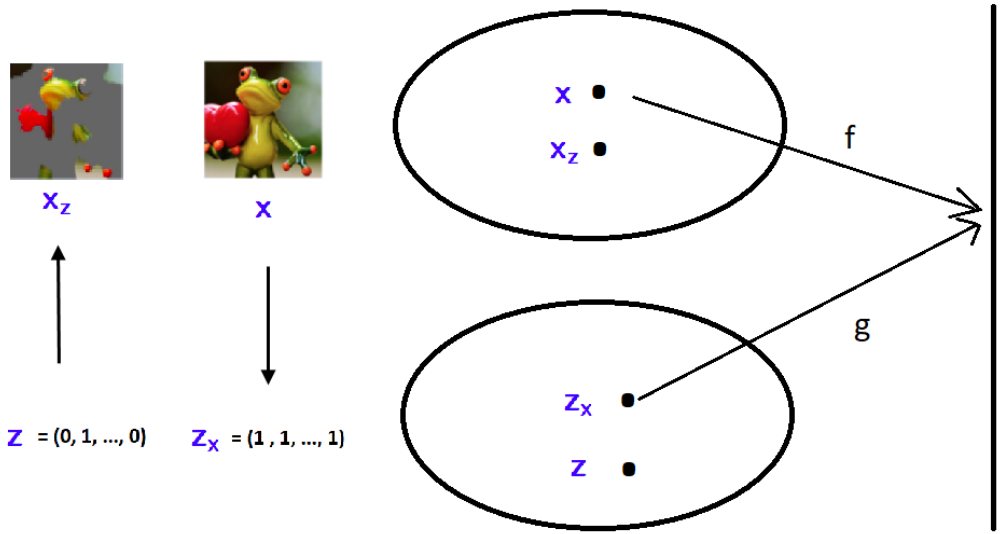
Kernel SHAP (Lundberg *et al* 2019)

- For each $\mathbf{z} = (z_1, \ldots, z_M) \in \{0,1\}^M$, define $g(\mathbf{z}) = \phi_0 + \sum_{i=1}^{M} \phi_i(f, \mathbf{x}) z_i$,
- LIME: $\xi(\mathbf{x}) = \arg\min_{g \in G} \mathcal{L}(f, g, \pi_{\mathbf{x}}) + \Omega(g)$
- **Theorem**: The SHAP values $\phi_i(f, \mathbf{x})$ can be found by solving LIME if

$$\begin{aligned} \Omega(g) &= 0 \\ \pi_{\mathbf{x}}(\mathbf{z}) &= \frac{M-1}{\binom{M}{|\mathbf{z}|}|\mathbf{z}|(M-|\mathbf{z}|)} \\ \mathcal{L}(f, g, \pi_{\mathbf{x}}) &= \sum_{\mathbf{z} \in \{0,1\}^M} [f(\mathbf{x}_{\mathbf{z}}) - g(\mathbf{z}))]^2 \pi_{\mathbf{x}}(\mathbf{z}) \end{aligned}$$

where $|\mathbf{z}|$ is the number of non-zero elements in $\mathbf{z}$.

Kernel SHAP



- **Algorithm** (weighted linear regression): Sample $\mathbf{z}_j \in \{0,1\}^M$ $(j = 1, \ldots, N)$, and determine $\phi_i(f, \mathbf{x})$ $(i = 1, \ldots, M)$ by minimizing:

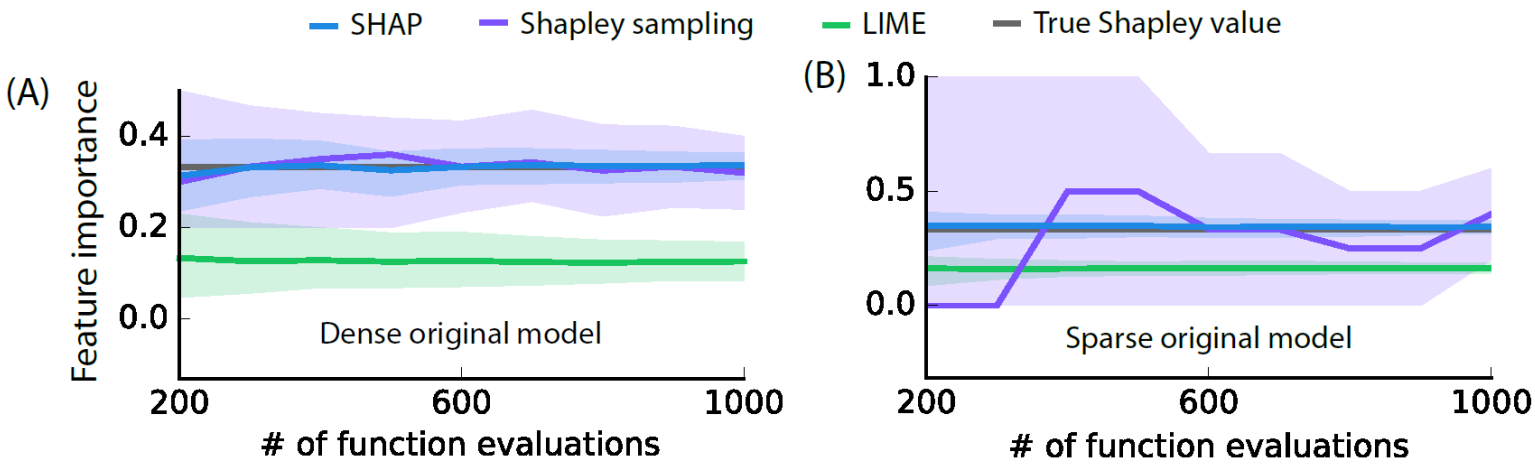
$$\frac{1}{N}\sum_{j=1}^{N}(f(\mathbf{x}_{\mathbf{z}_j}) - g(\mathbf{z}_j))^2 \pi_{\mathbf{x}}(\mathbf{z}_j)$$

Kernel SHAP

$$
\begin{aligned}
\Omega(g) &= 0 \\
\pi_{\mathbf{x}}(\mathbf{z}) &= \frac{M-1}{\binom{M}{|\mathbf{z}|}|\mathbf{z}|(M-|\mathbf{z}|)} \\
\mathcal{L}(f,g,\pi_{\mathbf{x}}) &= \sum_{\mathbf{z}\in\{0,1\}^M} [f(\mathbf{x}_{\mathbf{z}}) - g(\mathbf{z}))]^2 \pi_{\mathbf{x}}(\mathbf{z})
\end{aligned}
$$

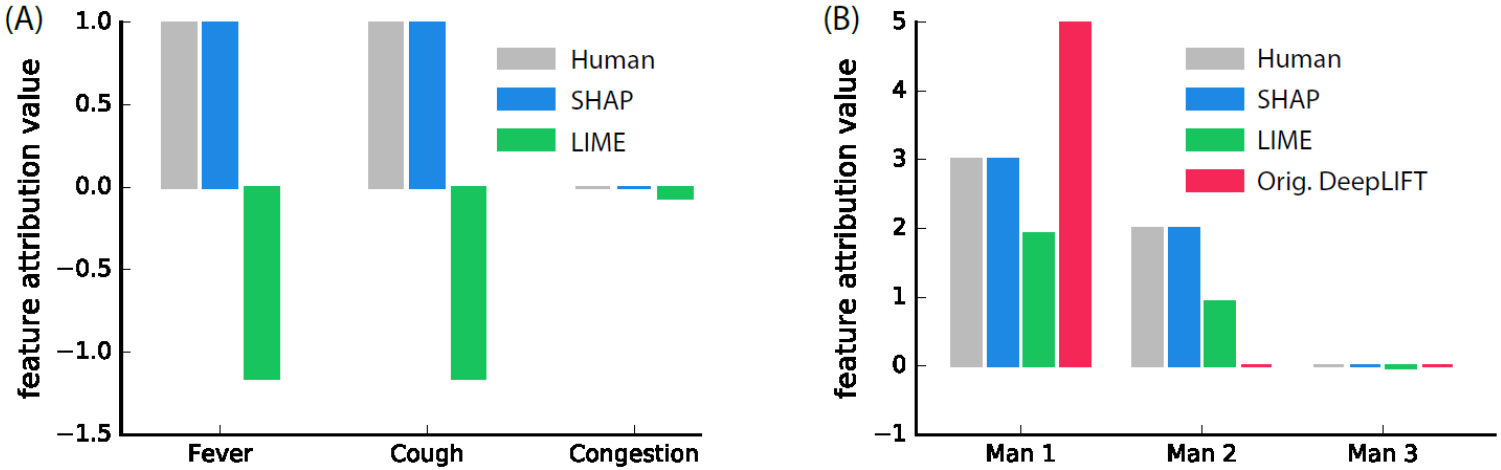
- Two special cases that are handled separately:
  - When $|\mathbf{z}| = 0$, $\pi_{\mathbf{x}}(\mathbf{z}) = \infty$. This enforces $\phi_0 = f_{\mathbf{x}}(\emptyset)$
  - When $|\mathbf{z}| = M$, $\pi_{\mathbf{x}}(\mathbf{z}) = \infty$. This enforces local accuracy

Comparison of Algorithms



- (A) A decision tree model using all 10 input features is explained for a single input. (B) A decision tree using only 3 of 100 input features is explained for a single input.
- Kernel SHAP yields more accurate estimates with fewer evaluations of the original model than previous sampling-based estimates.
- LIME can differ significantly from SHAP values that satisfy local accuracy and consistency.

Consistence with Human Intuition

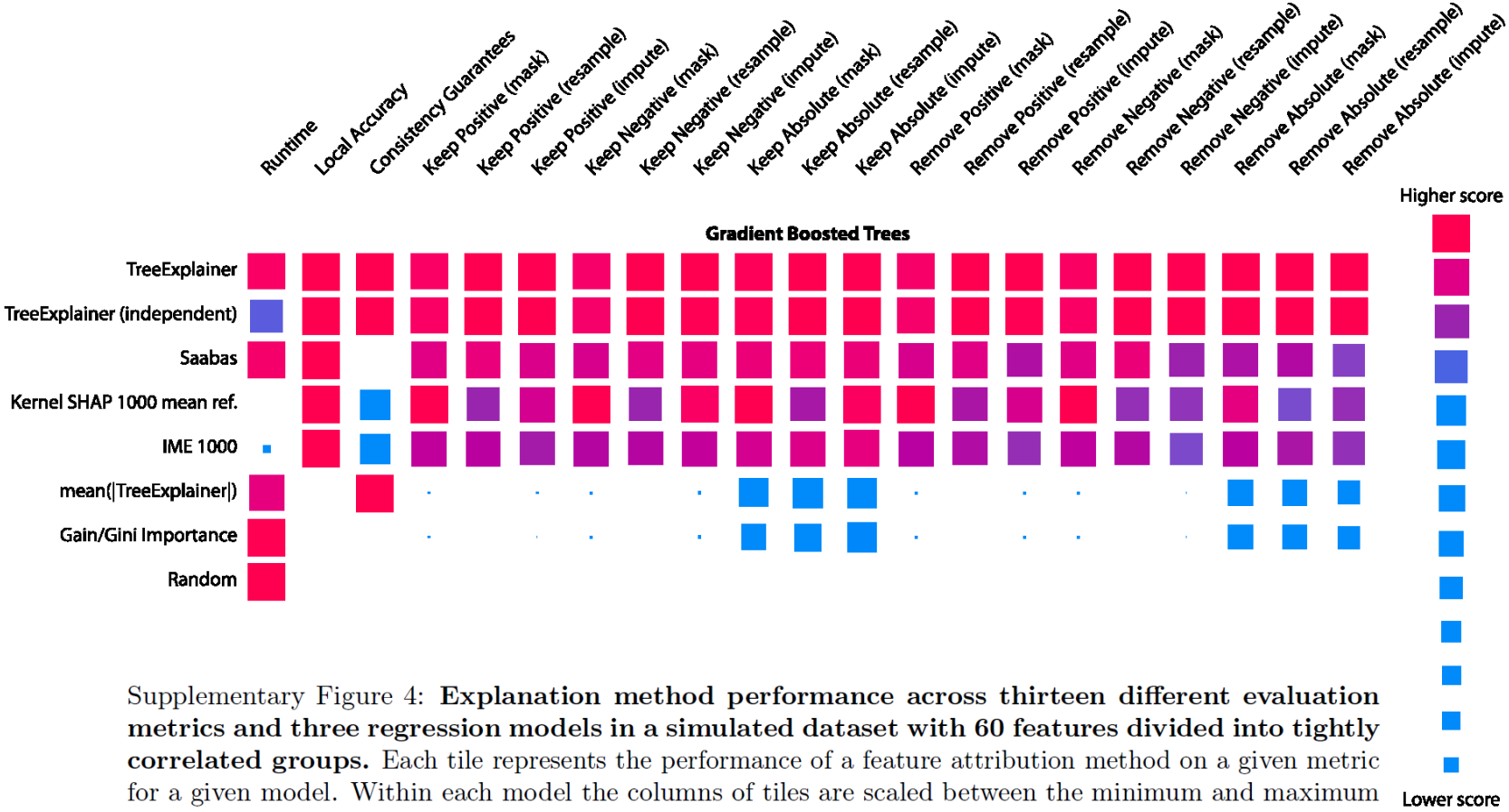


Feature attributions for two models by Human, SHAP, and LIME

(A) Model $f(cough, fever)$: $f(0,0,-) = 0$, $f(0,1,-) = 5$, $f(1,0,-) = 5$, $f(1,1,-) = 2$.
Task: Feature attributions for the case $(1,1)$.

(B) Model: $f(x,y,x) = \max\{x,y,z\}$.
Task: Feature attributions for the case $(5,4,0)$.

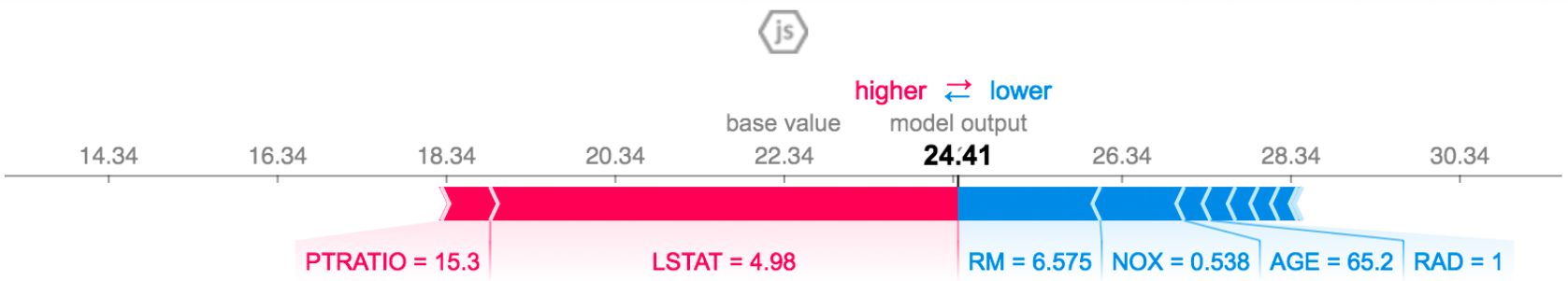
Faithfulness



Supplementary Figure 4: **Explanation method performance across thirteen different evaluation metrics and three regression models in a simulated dataset with 60 features divided into tightly correlated groups.** Each tile represents the performance of a feature attribution method on a given metric for a given model. Within each model the columns of tiles are scaled between the minimum and maximum value, and methods are sorted by their overall performance. Some of these metrics have been proposed before and others are new quantitative measures of explanation performance that we are introducing (see Methods).

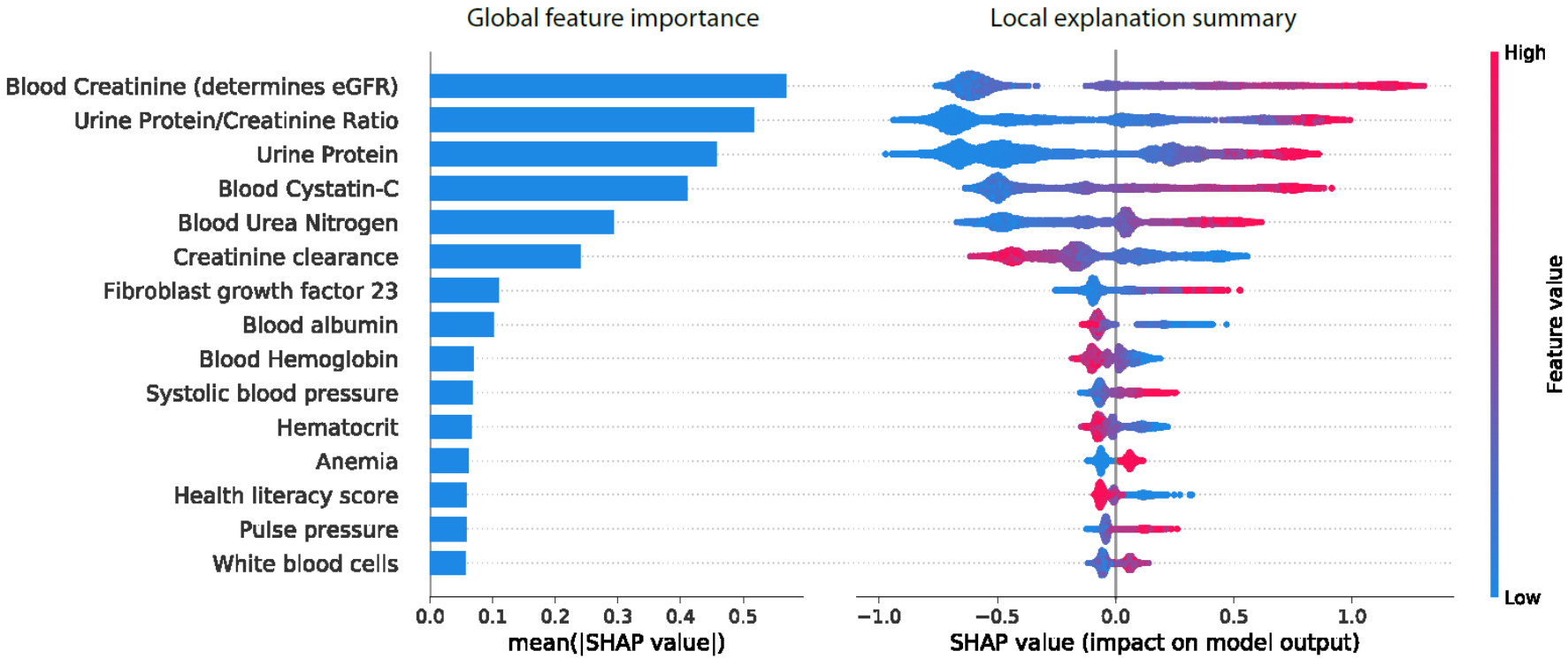
Keep Positive (mask): Given input the most positive input features are kept at their original values, while all the other input features are masked with their mean value. Check how the how the output changes with the number of features kept.

SHAP Values for Individual Input Shown as Force Plots



The above explanation shows features each contributing to push the model output from the base value (the average model output over the training dataset we passed) to the model output. Features pushing the prediction higher are shown in red, those pushing the prediction lower are in blue.

SHAP Values for Global Explanation



Supplementary Figure 12: **Bar chart (left) and summary plot (right) for a gradient boosted decision tree model trained on the chronic kidney disease data.** For the summary plot red dots indicate a high value of that feature for that individual, while blue dots represent a low feature value. The x-axis is the SHAP value of a feature for each individual's prediction, representing the change in the log-hazard ratio caused by observing that feature. High blood creatinine increases the risk of end-stage kidney disease. Conversely, low creatinine clearance increases the risk of end-stage kidney disease.

Outline

1 Pixel-Level Explanations

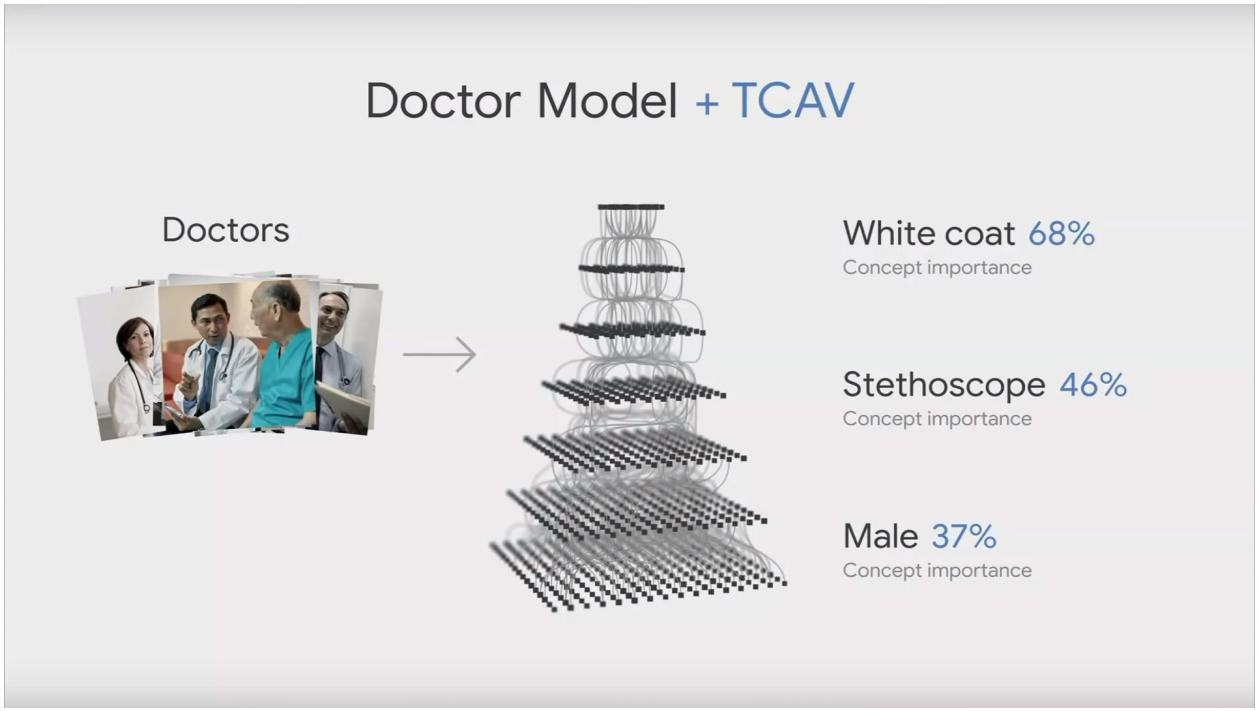
2 Feature-Level Explanations
- LIME
- SHAP Values

3 Concept-Level Explanations
- TCAV
- ACE

4 Instance-Level Explanations
- Counterfactual Explanations

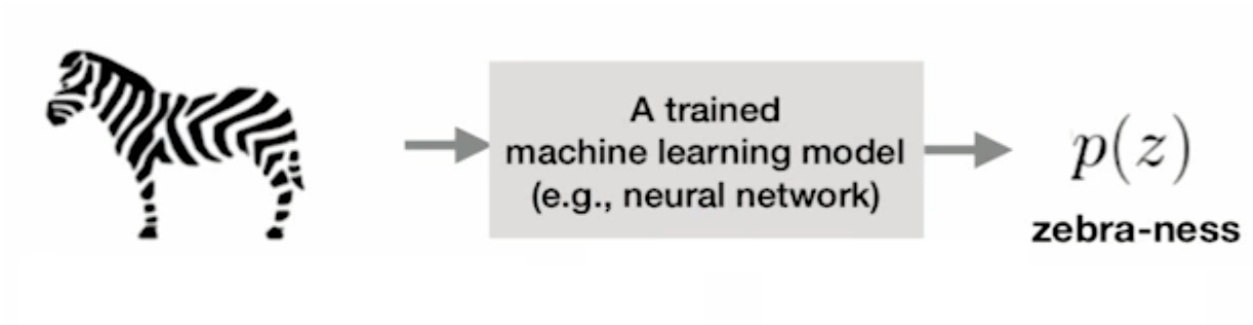
Concept Importance

- **Pixel Importance**: Which input pixels are important when model classifies one example? (Local Explanation)
- **Concept Importance**: Which external concepts are important when model classifies a class of examples? (Global Explanation)



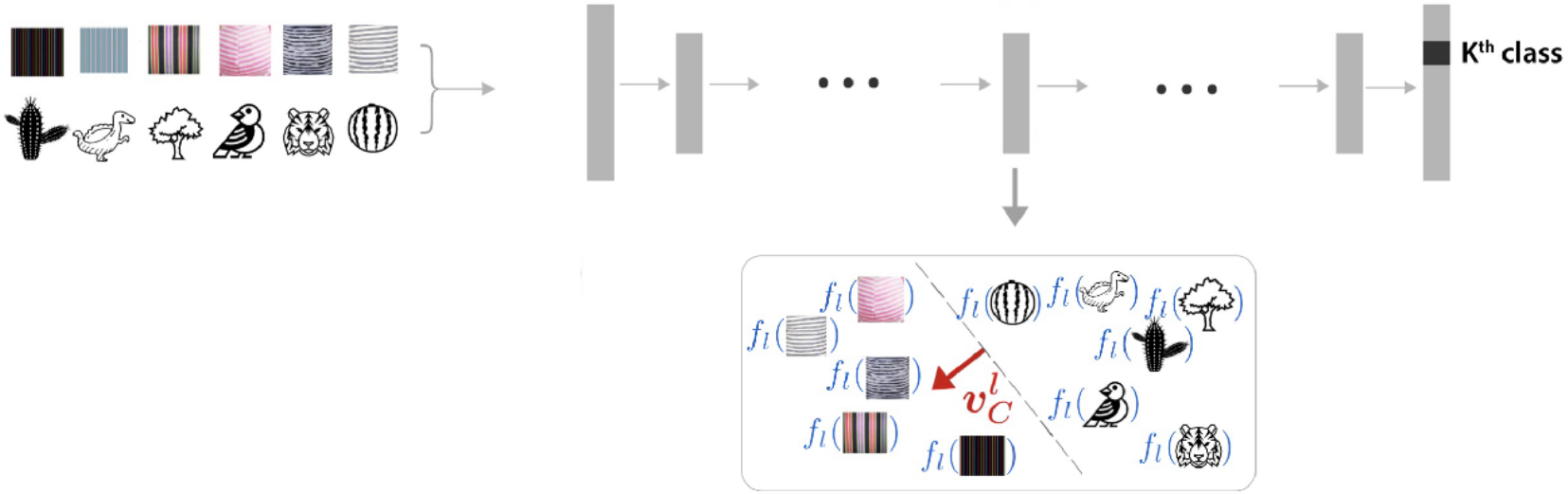
External concepts such as "White coat" are NOT class labels in training data.

TCAV (Kim *et al.* 2018)



- **Question**: How important is the concept "stripedness" to a zebra image classifier?
- The question is answered in three steps.

Step 1: Representing the Concept as a Vector



- Collect a positive set P_C of examples of the concept C, and a negative set N. Get the **activations** $f_l(\mathbf{x})$ of each example $\mathbf{x}$ at layer l.
- Train a linear classifier to separate the two sets:

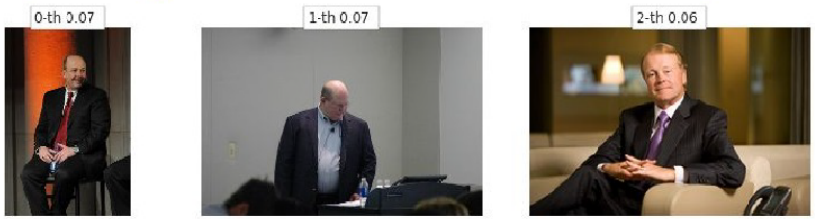
$$\{f_l(\mathbf{x}) : \mathbf{x} \in P_C\}, \{f_l(\mathbf{x}) : \mathbf{x} \in N\}$$

- The gradient vector $\mathbf{v}_C^l$ of the classifier the **concept activation vector** of C. It is the direction along which the probability of the class increases the fastest.

Quality of Concept Activation Vectors

- To assess whether the vector $\mathbf{v}_C^l$ captures the intended concept, use it to sort images (not in training) $\mathbf{x}$ using its inner product with $f_l(\mathbf{x})$.

top 3 images of CEO similar to suit concept



top 3 images of boss similar to labcoat concept

bottom 3 images of CEO similar to suit concept

bottom 3 images of boss similar to labcoat concept

Step 2: Partial Derivative of Class Score w.r.t Concept

- New, let $\mathbf{x}$ be an example classified into class k. $z_k(\mathbf{x})$ be the score for class k. It is also a function of $\mathbf{h}_l = f_l(\mathbf{x})$ and can be written $z_k(\mathbf{h}_l)$.
- The gradient $\frac{\partial z_k(\mathbf{x})}{\partial x_i}$ measures how sensitive the class score z_k is to small perturbations to the pixel x_i. It quantifies how important the pixel x_i is to class k.
- The **directional derivative** (https://en.wikipedia.org/wiki/Directional_derivative):

$$\lim_{\epsilon \to 0} \frac{z_k(\mathbf{h}_l + \epsilon \mathbf{v}_c^l) - z_k(\mathbf{h}_l)}{\epsilon} = \nabla z_k(\mathbf{h}_l) \cdot \mathbf{v}_c^l$$

  measures how sensitive the class z_k is to small changes in the direction of $\mathbf{v}_c^l$
- It quantifies how important the concept C is to the classification of the input $\mathbf{x}$ into class k. It is denoted as $S_{C,k,l}(\mathbf{x})$.

Step 3: Testing with CAVs (TCAV)

- Let $\mathbf{X}_k$ be the set of inputs in class k. Define

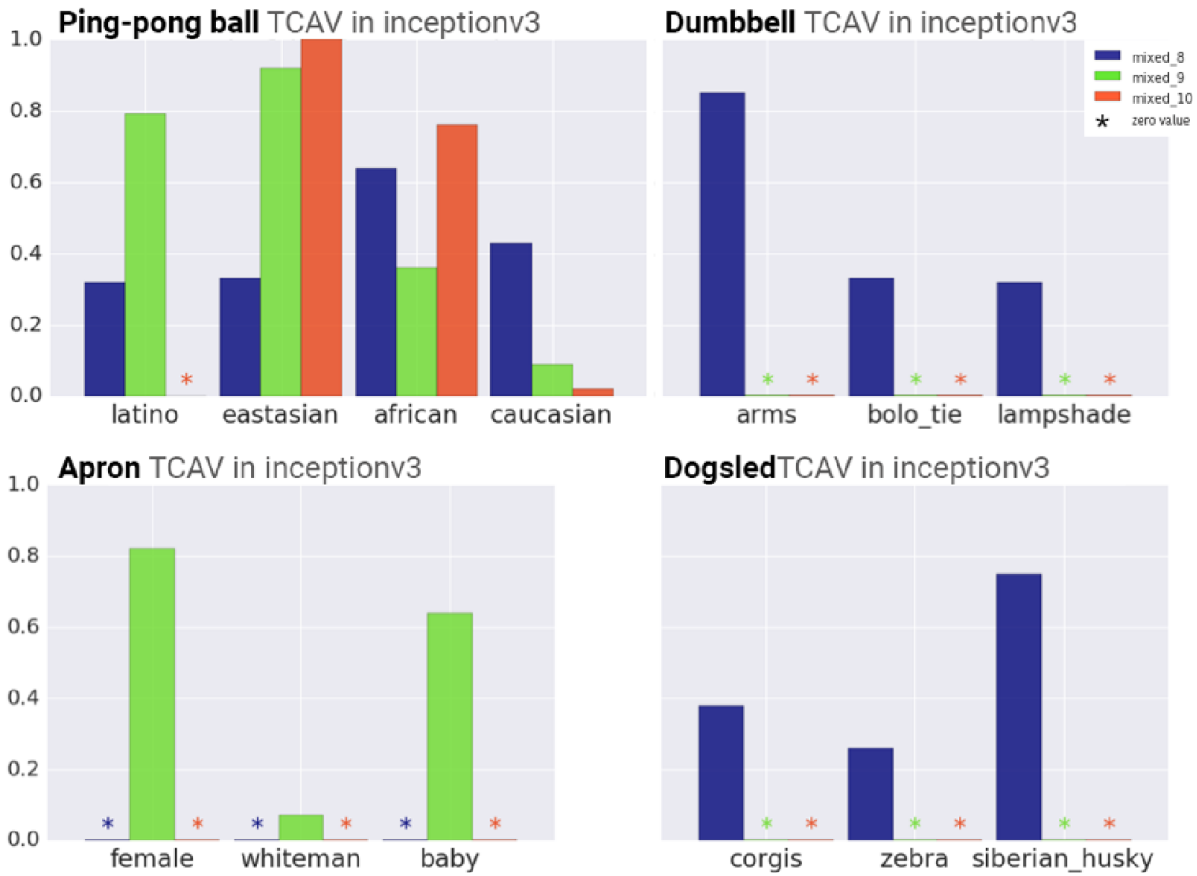
$$TCAV_Q(C, k, l) = \frac{|\{\mathbf{x} \in \mathbf{X}_k : S_{C,k,l}(\mathbf{x}) > 0\}|}{|\mathbf{X}_k|}$$

  i.e. the fraction of k-class inputs whose l-layer activation vector was positively influenced by concept C.

- It measures how important the concept C is when the model classify inputs to class k.

- To guard against spurious results, calculate TCAV scores many times and do hypothesis testing. The null hypothesis: TCAV score is 0.5. If P_C and N are chosen randomly, expected TCAV score is 0.5.

Testing with CAVs (TCAV)



Note: Multiple studies suggest that lower layers of CNN operate as lower level feature detectors (e.g., edges), while higher layers use these combinations of lower-level features to infer higher-level features (e.g., classes.

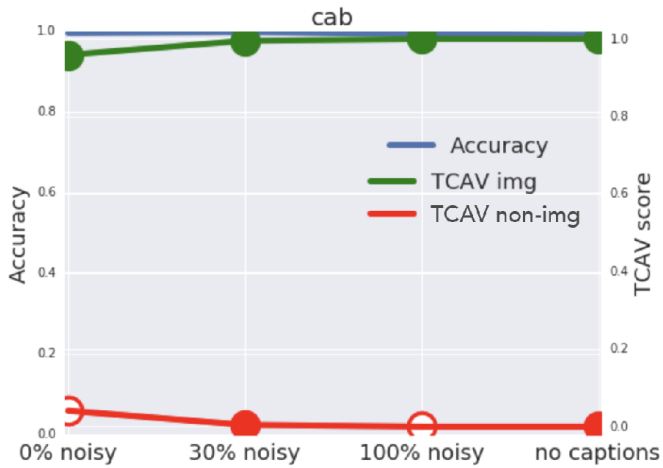
Evaluation of TCAV

cab image | cab image with caption | cucumber image | cucumber with caption

- Training set has three classes: zebra, cab, and cucumber.
- Some training images have captions, and the caption is noise with probability p. ($p = 0$, 0.3, or 1)
- 4 networks are learned, one for each value of p, and the fourth one for training examples with no captions.

Image Features or Caption Features

- Does a network rely more on image features or caption features?
- Two ways to find out:
  (a) Test on images with no captions. If high accuracy, then the network relies primarily on image features. (Ground truth)
  (b) Use TCAV to determine whether network relies on image concept:
    - P_C: Set of cab images with no captions; N: Set of random images.
    - Calculate $S_{C,k,l}(\mathbf{x})$ and $TCAV_Q(C,k,l)$.
    - If TCAV high, then the network relies primarily on image features. (according to image TCAV explanation)
- If image TCAV matches the ground truth, then it is of high quality.



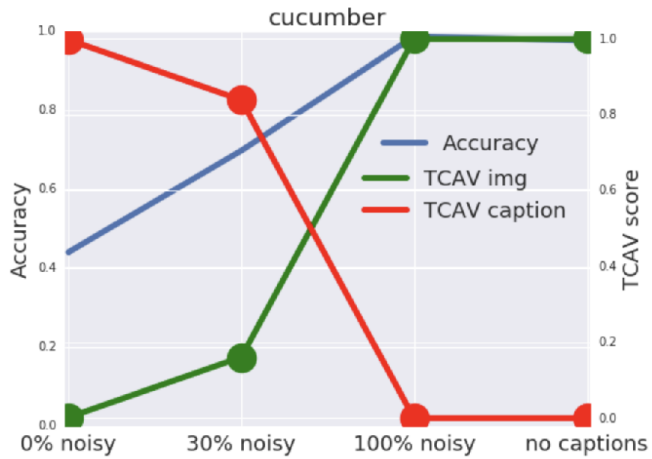
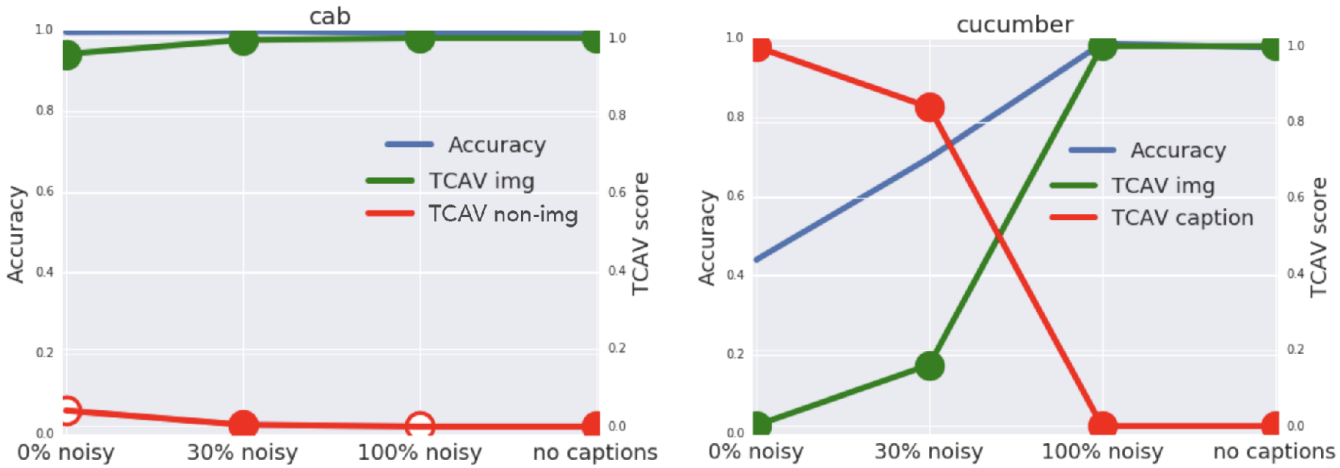


Image Features or Caption Features

- Caption CAV: P_C: Set of images with "Cab" caption and other pixels randomly shuffled. N: Set of images with random captions and other pixels randomly shuffled.
- The caption TCAV is low if the network relies on image features, not caption features. That is, if ground truth accuracy is low.
- It turns out to be the case, demonstrating high quality of TCAV.



For Cab, image features are used all cases. For cucumber, caption features are used in the first two cases, and images features are used in the last two cases.

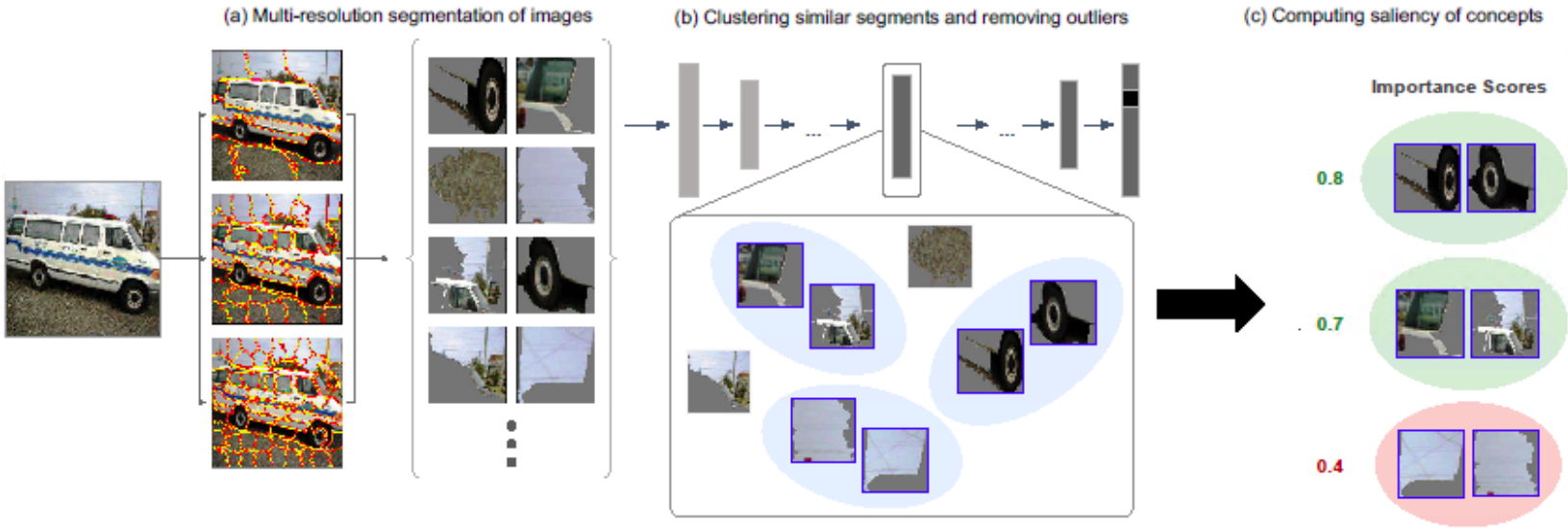
Summary of TCAV

- Inputs:
  - A pre-trained model.
  - A set of images $\mathbf{X}_k$ of class k.
  - A set of positive examples P_C of concept C, and a set of negative examples N.
- Output: A score $TCAV_Q(C, k, l)$ of importance of concept C w.r.t class k.

- Drawback: It is time consuming to construct the positive set P_C manually.

Automatic Concept-based Explanations (ACE) (Ghorbani *et al.* 2019)

- ACE automatically extract visual concepts.



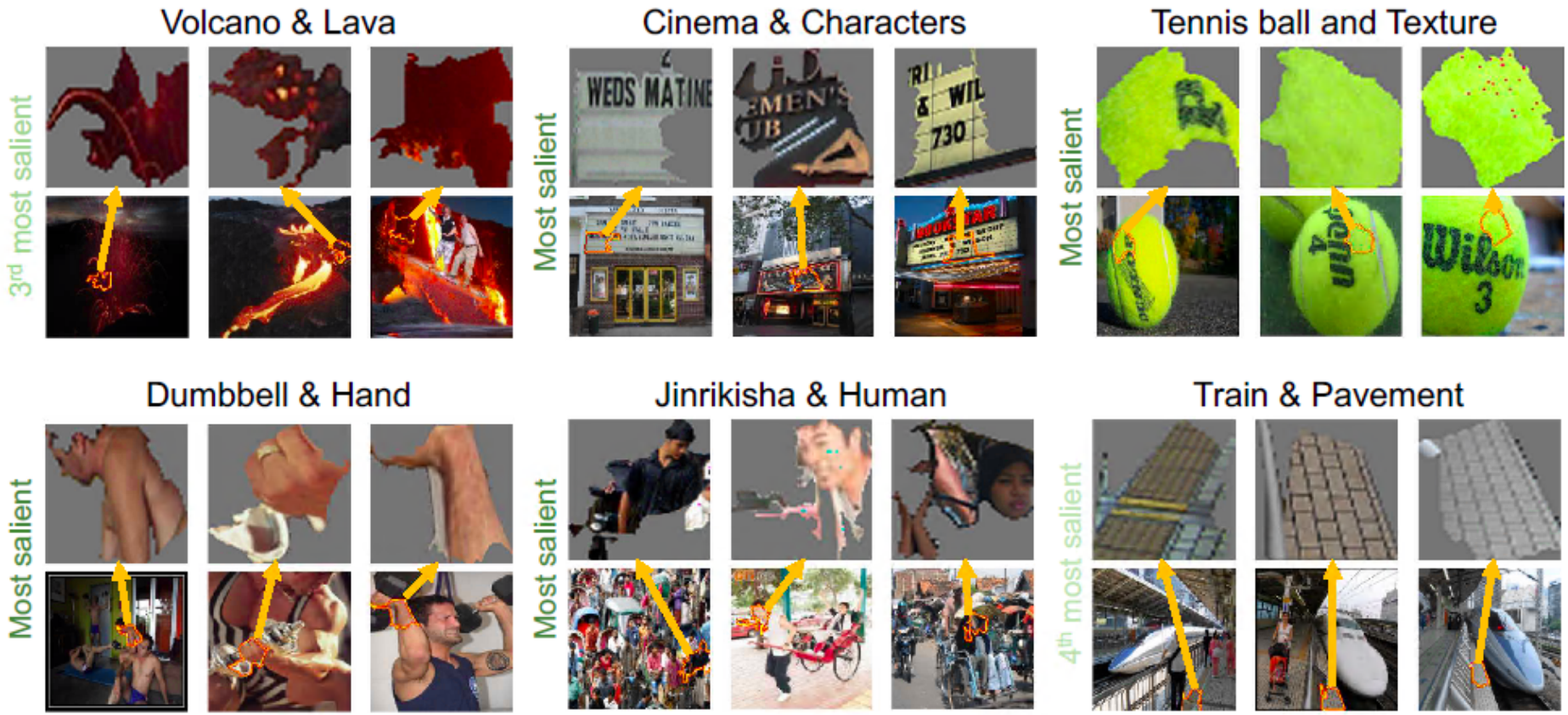
(a) Images from the same class are segmented with multiple resolutions resulting in a pool of segments using SLIC (Achanta *et al.* 2012).

(b) The activation space of one bottleneck layer of a CNN is used as a similarity space. After resizing each segment to the standard input size of the model, similar segments are clustered in the activation space into concept clusters.

(c) For each resulting concept, its TCAV importance score is computed.

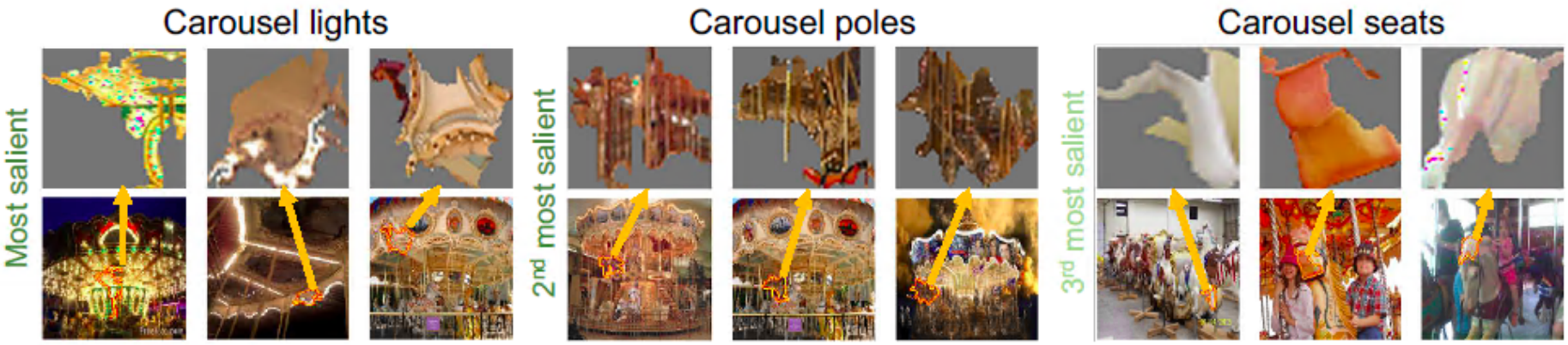
- The Police characters are important for detecting a police van while the asphalt on the ground is not.
- The most important concept for predicting basketball images is the players jerseys rather than the ball itself.

ACE (Ghorbani *et al.* 2019)



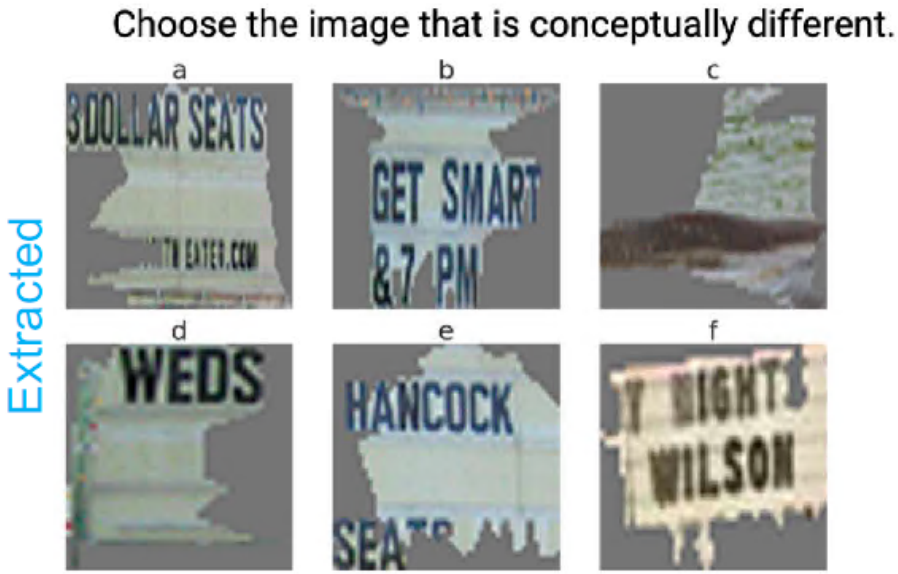
ACE reveals intuitive correlations (first row) and unintuitive correlations (second row).

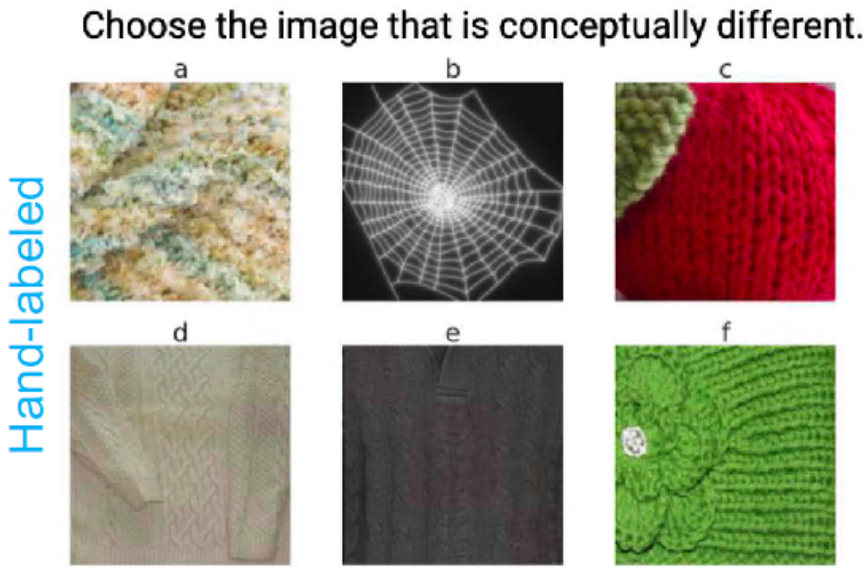
ACE (Ghorbani *et al.* 2019)



- In some cases, when the objects structure is complex, parts of the object as separate concepts have their own importance and some parts are more important than others.
- The example of carousel is shown: lights, poles, and seats. It is interesting to learn that the lights are more important than seats.

ACE Evaluation: Coherency of Concepts

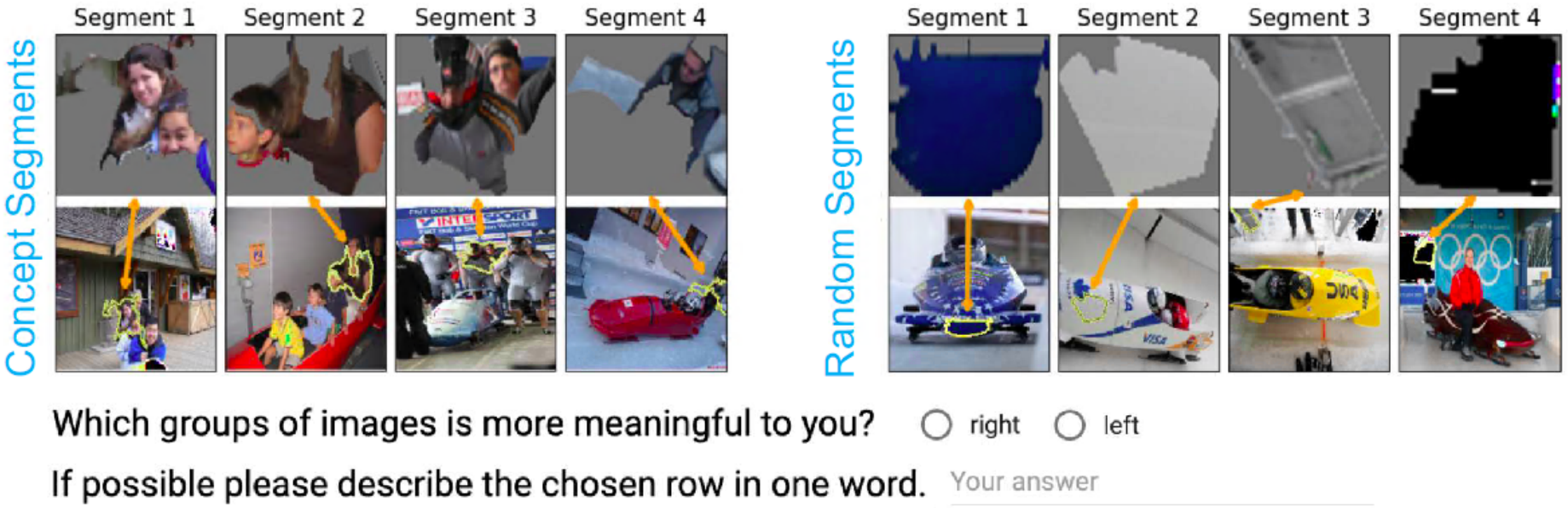




Intruder detection experiment:

- 30 participants answered the hand-labeled dataset 97% correctly,
- while discovered concepts were answered 99% correctly.

ACE Evaluation: Meaningfulness of Concepts

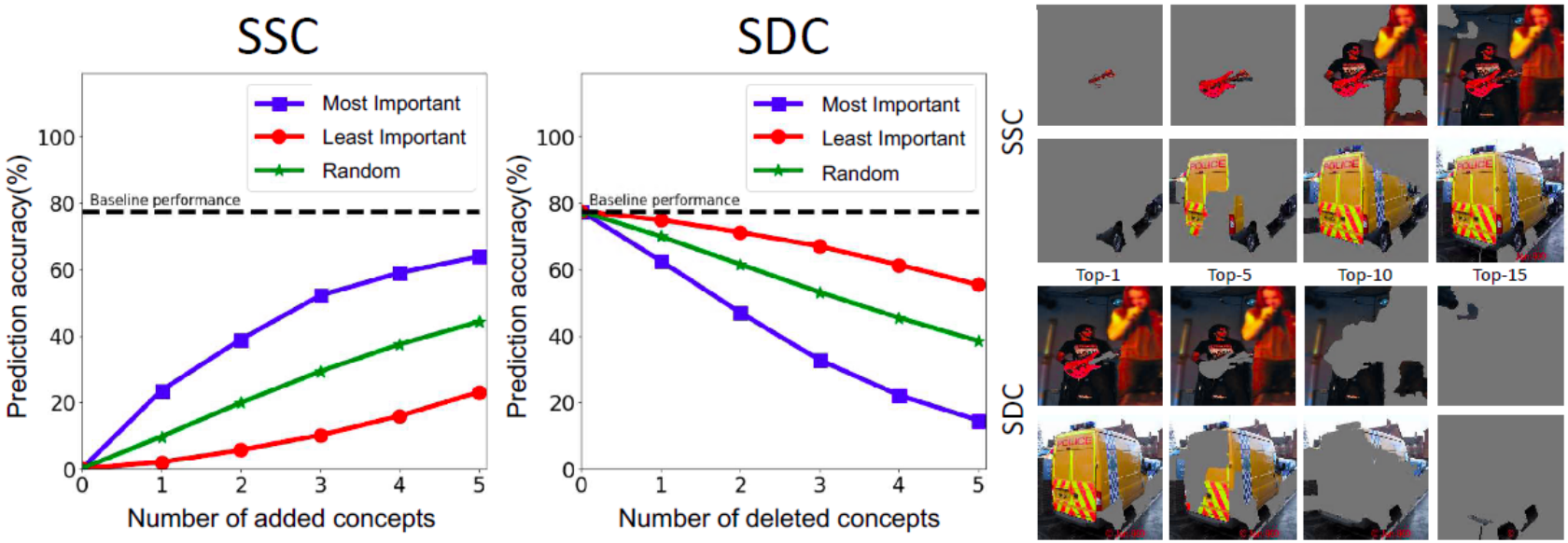


(a) Four segments of the same concept (along with the image they were segmented from,

(b) Four random segments of images in the same class.

The right option was chosen 95.6%.

ACE Evaluation: Importance of Concepts



(a) **Smallest sufficient concepts (SSC)**: The smallest set of concepts that are enough for predicting the target class.

(b) **Smallest destroying concepts (SDC)**: The smallest set of concepts removing which will cause incorrect prediction. Note

The top-5 concepts is enough to reach within 80% of the original accuracy and removing the top-5 concepts results in misclassification of more than 80% of samples that are classified correctly.

Outline

1 Pixel-Level Explanations

2 Feature-Level Explanations
- LIME
- SHAP Values

3 Concept-Level Explanations
- TCAV
- ACE

4 Instance-Level Explanations
- Counterfactual Explanations

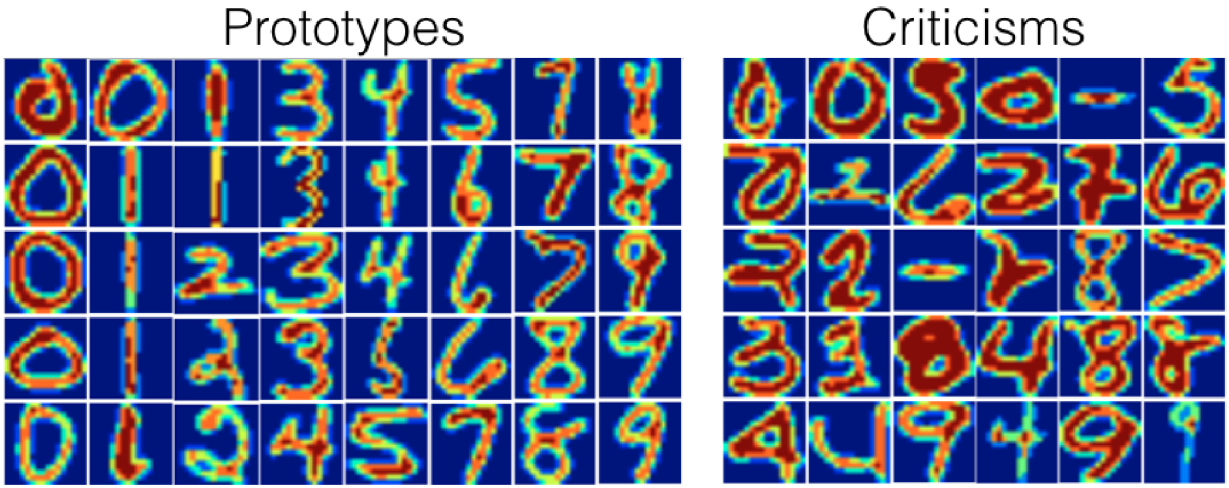
Instance-Level Explanations

- Instance-based explanation methods use examples to explain behavior of a model.
- The examples can be
  - From a dataset (prototypes or criticisms)
  - Generated by a generative model (counterfactuals).

Prototypes and Criticisms (Kim *et al.* 2016)

A global explanation method. It explains a class, as defined by a model, using:

- **Prototypes**: Representative examples of the class
- **Criticisms**: Examples of the class that do not quite fit the model.

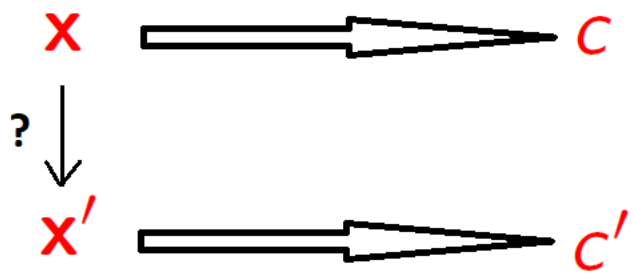


The prototypes capture many of the common ways of writing digits, while the criticism clearly capture outliers.

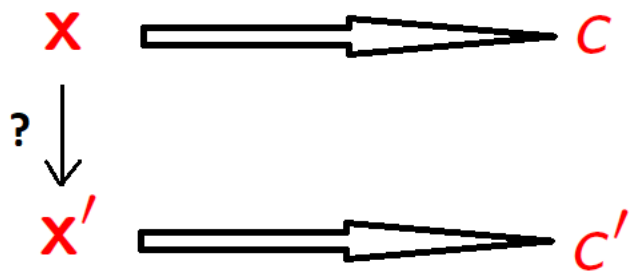
Counterfactual Explanations (Wachter *et al* 2017)

- **Counterfactual**: Answer to "what would have happened if" question.

- John's feature vector is $\mathbf{x}$. His loan application is denied (class c) by a model.
- **Counterfactual explanation**:
  - The loan would have been granted if income was \$45,000 instead of \$30,000.
  - Abstractly speaking, the classification would have been c' instead of c if John's feature vector was $\mathbf{x}'$ instead of $\mathbf{x}$.



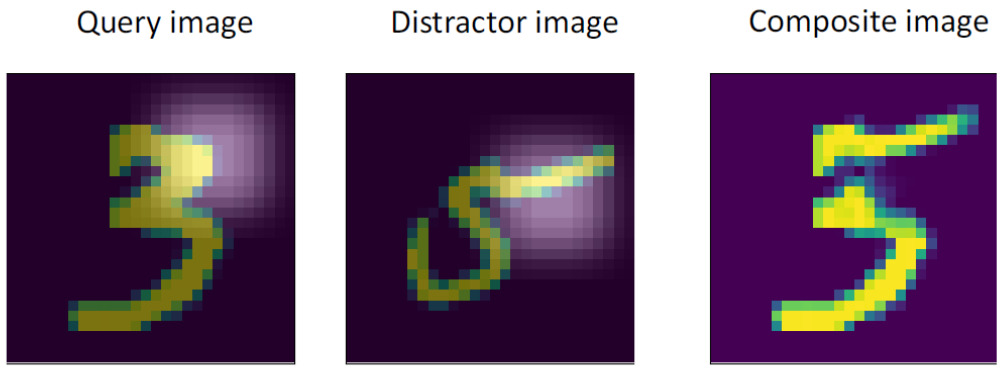
Counterfactual Explanations: Problem Statement (Wachter *et al* 2017)



- Suppose input **x** is classified as c by model.
- Find a **counterfactual example** $\mathbf{x}'$ s.t.
  - $\mathbf{x}'$ is classified as another class c' by model.
  - The difference between $\mathbf{x}$ and $\mathbf{x}'$ should be small.
  - The change from $\mathbf{x}$ to $\mathbf{x}'$ should be feasible in the real-world (actionability).
    - Actionable: Loan would be granted if income was \$45,000 instead of \$30,000.
    - Not actionable: Loan would be granted if age was 30 instead of 50.

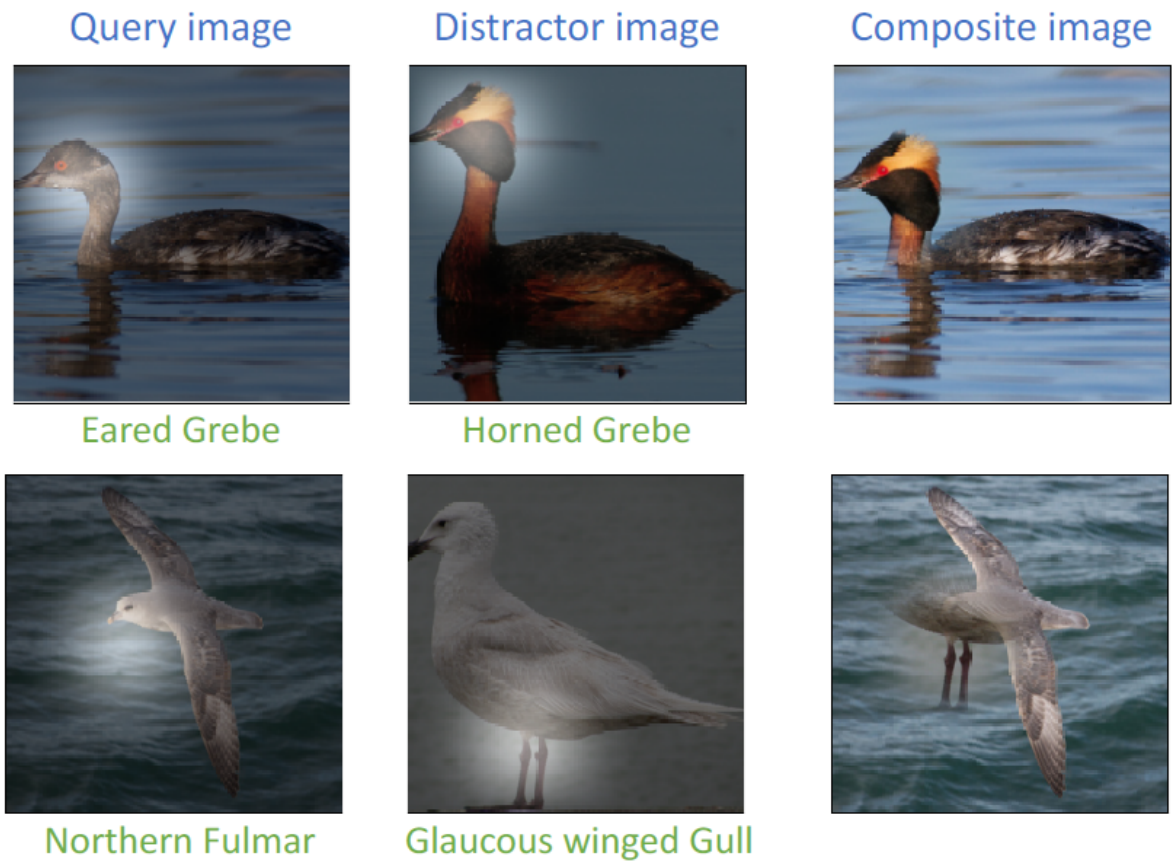
It is an instance-level local explanation method.

Counterfactual visual explanations (Goyal *et al* 2019)



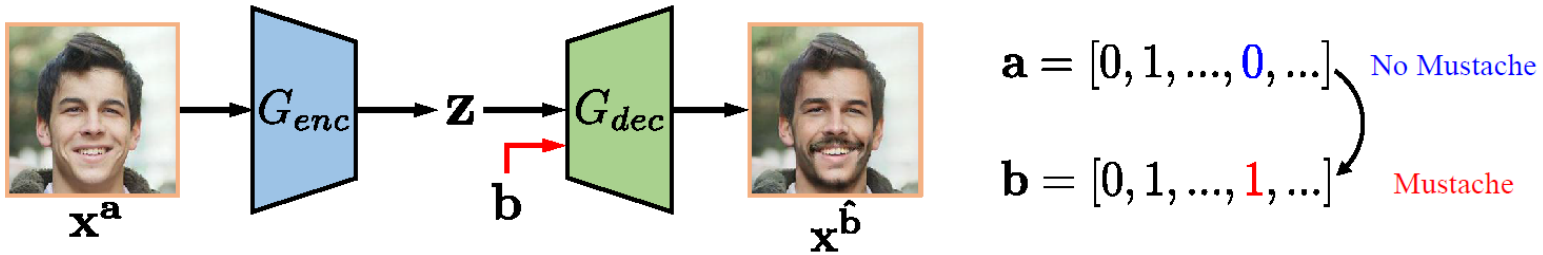
- Left image classified as 3 by model.
- **Question**: What changes should be made to image if it is to be classified as 5 (as represented by the distractor)?
- **Answer**: The counterfactual image on the right.

Counterfactual visual explanations (Goyal *et al* 2019)



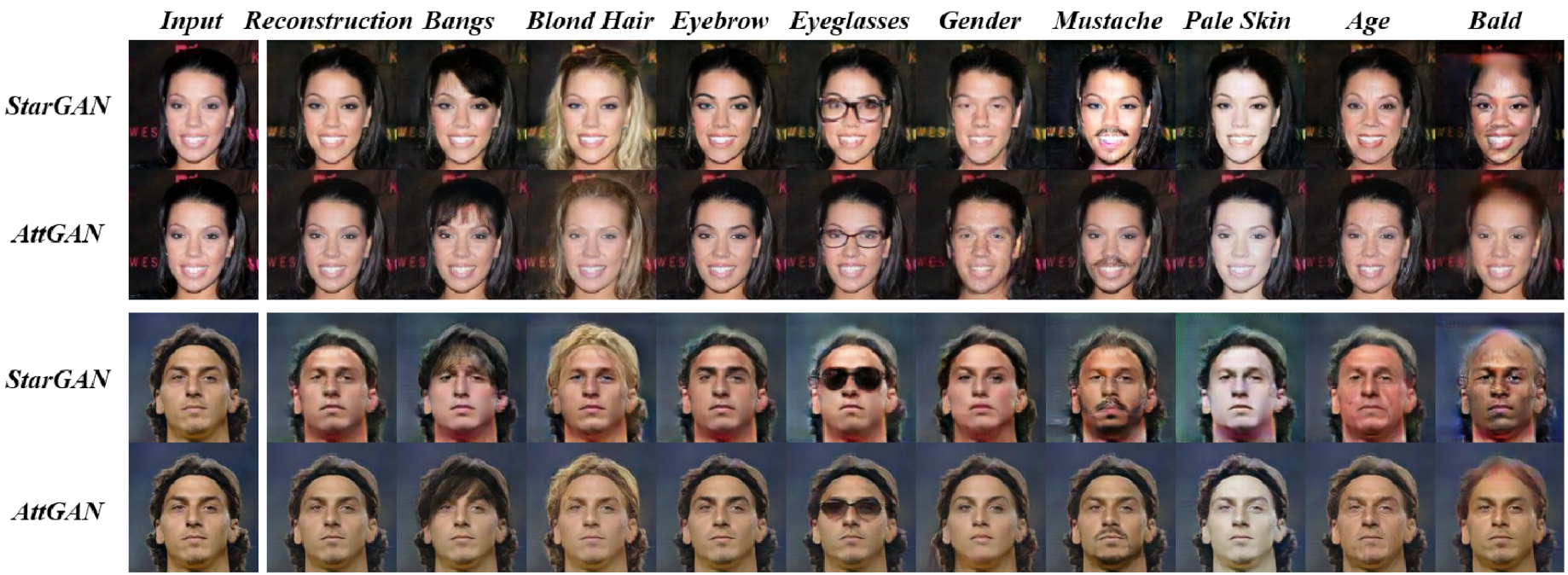
Low actionality. Counterfactual image in the second case does not look realistic.

Diversion: Image Attribute Editing



- An **image attribute editing/style transfer** model $G(\mathbf{x}, \mathbf{b})$
  - Inputs: An image $\mathbf{x}$ and a target style $\mathbf{b}$, represented as binary vector.
  - Output: A new image of the target style $\mathbf{b}$.

Diversion: Image Attribute Editing



Diversion: Image Attribute Editing

Generative Counterfactual Explanation (Liu *et al.* 2019)

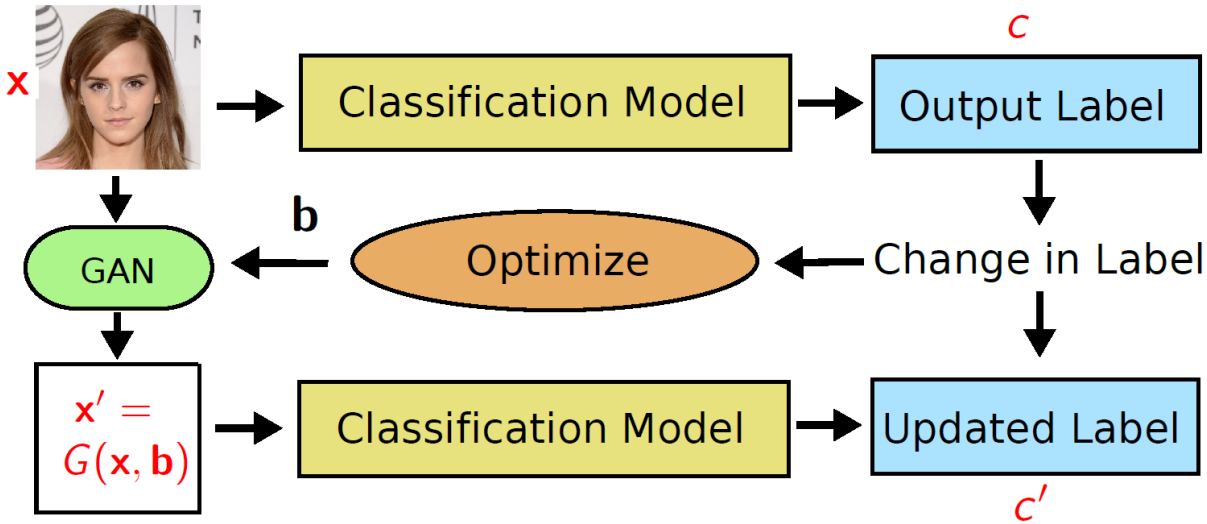


Figure 1: The illustration of the generative counterfactual introspection concept.

Generative Counterfactual Explanation (Liu *et al.* 2019)

- C — class assignment function of model;
- $\mathbf{x}$ — input image; $c = C(\mathbf{x})$;
- c' — another class; $\mathbf{b}$ — target attribute vector (real values, not binary).
- Problem statement: Identify attribute values $\mathbf{b}$ that would take $\mathbf{x}$ cross the decision boundary between c and c' to become $\mathbf{x}' = G(\mathbf{x}, \mathbf{b})$:

$$\begin{aligned} &\min_{\mathbf{b}} ||\mathbf{x} - \mathbf{x}'||_1 \\ &s.t.\ \ C(\mathbf{x}') = c' \end{aligned}$$

  If $\mathbf{x}$ were changed to $\mathbf{x}'$, then it would have been assigned to class c'. The changes were to be made by fixing the attribute values to $\mathbf{b}$ so that human can see clearly that features are important for class c.

- Similar to the formulation of adversarial examples, except there the changes (perturbations) are applied to pixels, and the objective is to make the changes imperceptible to human.

Generative Counterfactual Explanation (Liu *et al.* 2019)

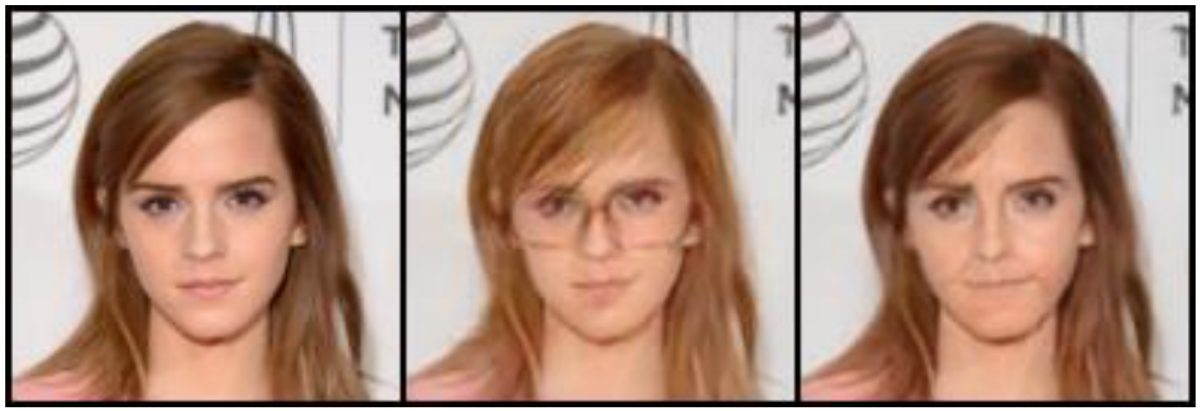
$\min_{\mathbf{b}} ||\mathbf{x} - \mathbf{x}'||_1$
$s.t.\ C(\mathbf{x}') = c'$

- The above optimization problem is difficult to solve.
- So, relax it to

  $\min_{\mathbf{b}} \lambda \mathcal{L}(\mathbf{x}', c') + ||\mathbf{x} - \mathbf{x}'||_1$
  $s.t.\ C(\mathbf{x}') = c'$

  where $\mathcal{L}(\mathbf{x}', c') = -logP(y = c'|\mathbf{x}')$.
- The new problem can be solved using gradient descent.
- Recall similar ideas from the lecture on adversarial examples.

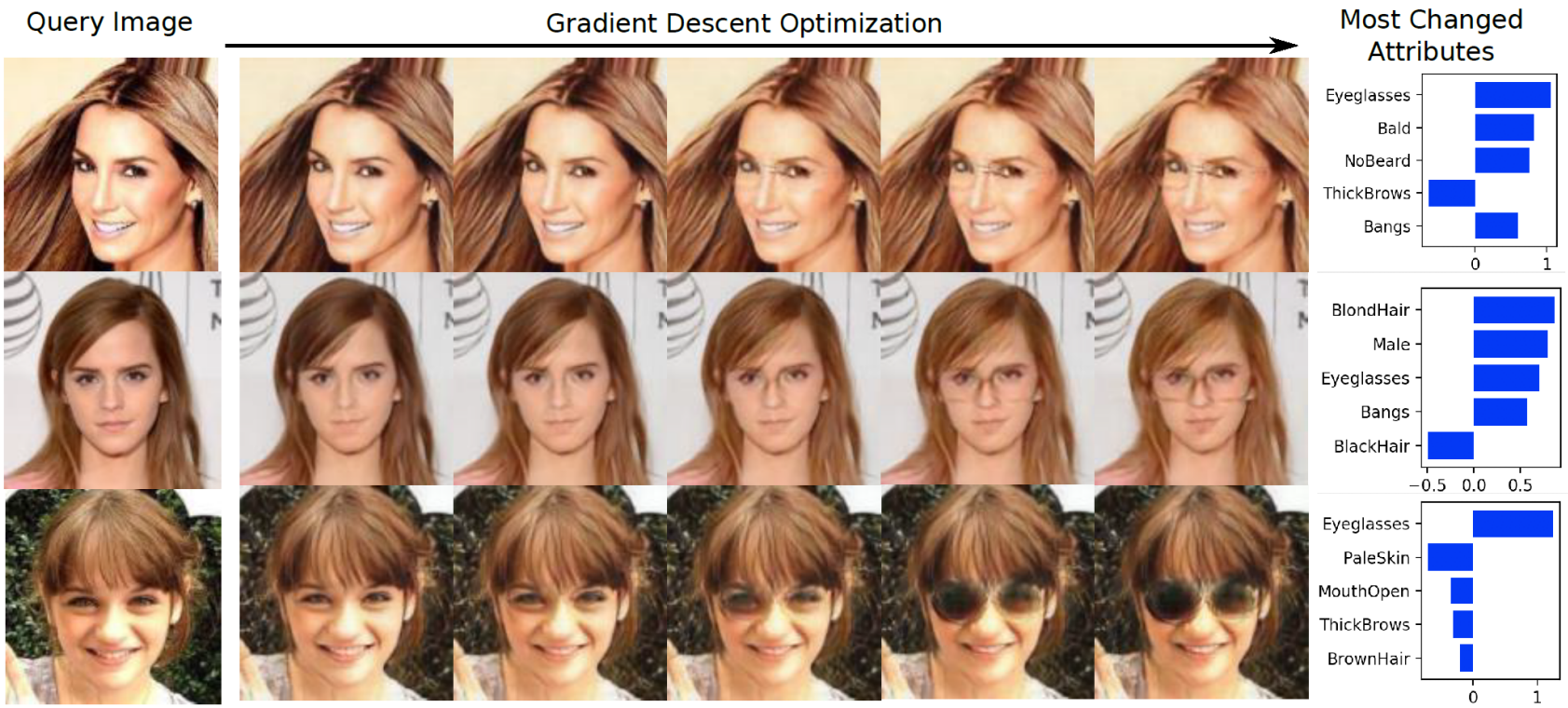
Generative Counterfactual Explanation (Liu *et al.* 2019)



(a) Original (b) Optimized (c) "Old" Target

- A model classify (a) as young.
- (c) is a counterfactual example obtained by altering the "old/young" attributes. A visual comparison of (a) and (c) gives human a understanding of what the model considers young, and what it considers old.

Generative Counterfactual Explanation (Liu *et al.* 2019)



- Counterfactual examples obtained by altering attributes other than the "old/young" attributes. The results shows how those other attributes affects the class "young" in the model.
- So far, counterfactual explanation methods have been proposed only for specialized datasets such as MNIST, Caltech-UCSD Birds (CUB), CelebA.

References: Feature and Concept-Level Explanations

- Achanta, Radhakrishna, et al. "SLIC superpixels compared to state-of-the-art superpixel methods." IEEE transactions on pattern analysis and machine intelligence 34.11 (2012): 2274-2282.
- Ghorbani, Amirata, et al. "Towards automatic concept-based explanations." Advances in Neural Information Processing Systems. 2019.
- Jackson, Matthew O., https://www.youtube.com/watch?v=qcLZMYPdpH4. 2014.
- Knight, Vincent, https://vknight.org/Year_ 3_ game_ theory_ course/Content/Chapter_ 16 _ Cooperative _ games/. 2020
- Lundberg, Scott M., and Su-In Lee. "A unified approach to interpreting model predictions." Advances in neural information processing systems. 2017.
- Lundberg, Scott M., et al. "Explainable AI for trees: From local explanations to global understanding." arXiv preprint arXiv:1905.04610 (2019).
- Kim, Been. https://www.youtube.com/watch?v=Ff-Dx79QEEY. (2018)
- Kim, Been, et al. "Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav)." International conference on machine learning. 2018.
- Ribeiro, Marco Tulio, et al. "" Why should I trust you?" Explaining the predictions of any classifier." Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. 2016.
- Ribeiro, Marco Tulio, et al. "Anchors: High-precision model-agnostic explanations." Thirty-Second AAAI Conference on Artificial Intelligence. 2018.
- Watzman, Adi, SHAP Values for ML Explainability, https://www.youtube.com/watch?v=0yXtdkIL3Xk & t=851s, 2020.

References: Instance-Level Explanations

- Goyal, Yash, et al. "Counterfactual visual explanations." arXiv preprint arXiv:1904.07451 (2019).
- He, Zhenliang, et al. "Attgan: Facial attribute editing by only changing what you want." IEEE Transactions on Image Processing 28.11 (2019): 5464-5478.
- Liu, Shusen, et al. "Generative counterfactual introspection for explainable deep learning." arXiv preprint arXiv:1907.03077 (2019).
- Kim, Been, Rajiv Khanna, and Oluwasanmi O. Koyejo. "Examples are not enough, learn to criticize criticism for interpretability." Advances in neural information processing systems. 2016.
- Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." arXiv preprint arXiv:1511.06434 (2016).
- Wachter, Sandra, Brent Mittelstadt, and Chris Russell. "Counterfactual explanations without opening the black box: Automated decisions and the GDPR." Harv. JL & Tech. 31 (2017): 841.
- Zhang, Quanshi, Ying Nian Wu, and Song-Chun Zhu. "Interpretable convolutional neural networks." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.