WB-XIC, Lab7:

Wyjaśnienia konwolucyjnych sieci neuronowych c.d.

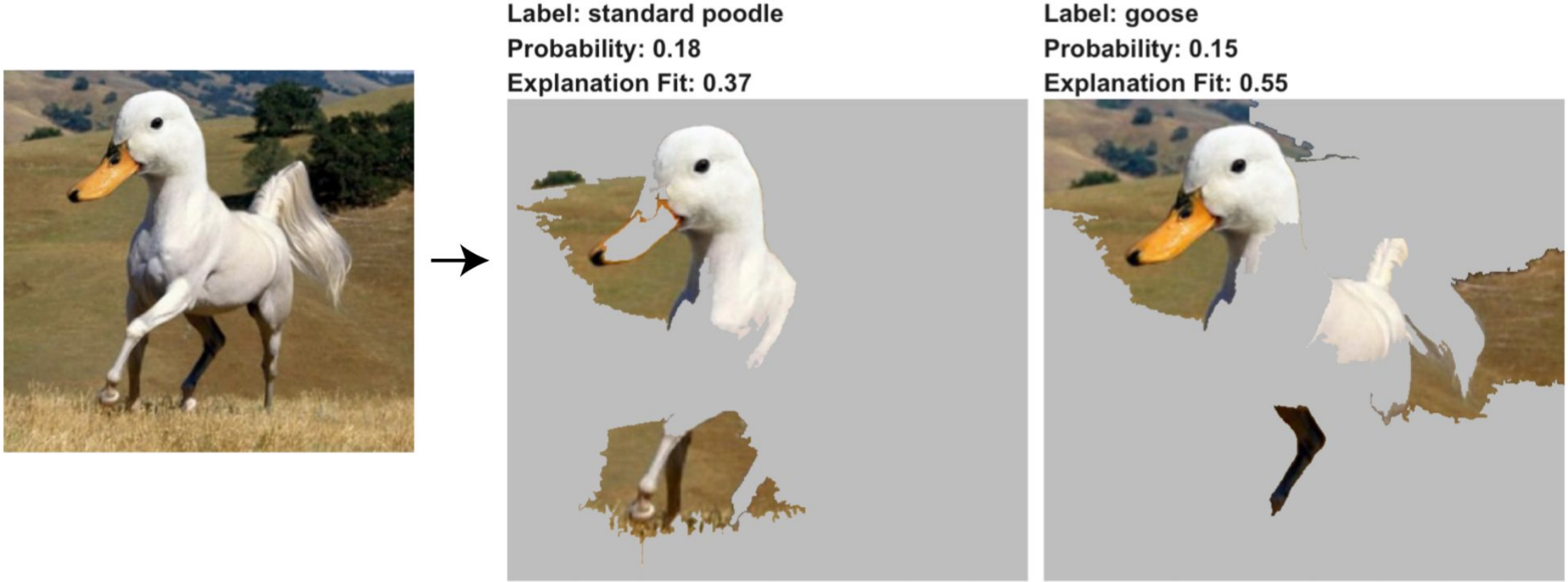
Hubert Baniecki

hbaniecki@gmail.com | http://hbaniecki.com

Outline

1. SHAP
2. Neuron & layer explanations
3. Check!
4. Concept-based explanations (they exist)
5. Global explanations (intuition)
6. Project

LIME for image: superpixels and image segmentation



https://ema.drwhy.ai/LIME

Saliency maps (*vanilla* gradients)

The recipe for this approach is:

1. Perform a forward pass of the image of interest.
2. Compute the gradient of class score of interest with respect to the input pixels:

$$E_{grad}(I_0) = \frac{\delta S_c}{\delta I}\Big|_{I=I_0}$$

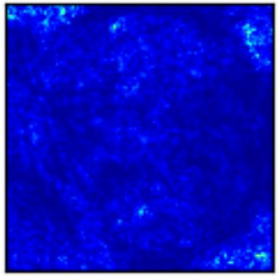
   Here we set all other classes to zero.

3. Visualize the gradients. You can either show the absolute values or highlight negative and positive contributions separately.

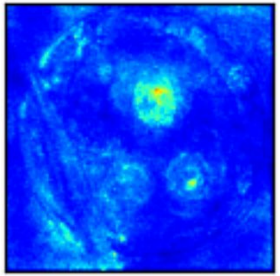
Smoothgrad: average multiple explanations for an image with added noise
Grad-Cam: gradient explanation tailored to CNN (ReLU, last Conv2d)

https://christophm.github.io/interpretable-ml-book/pixel-attribution

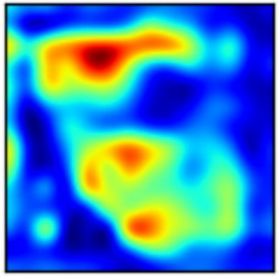
Soup Bowl (vanilla)



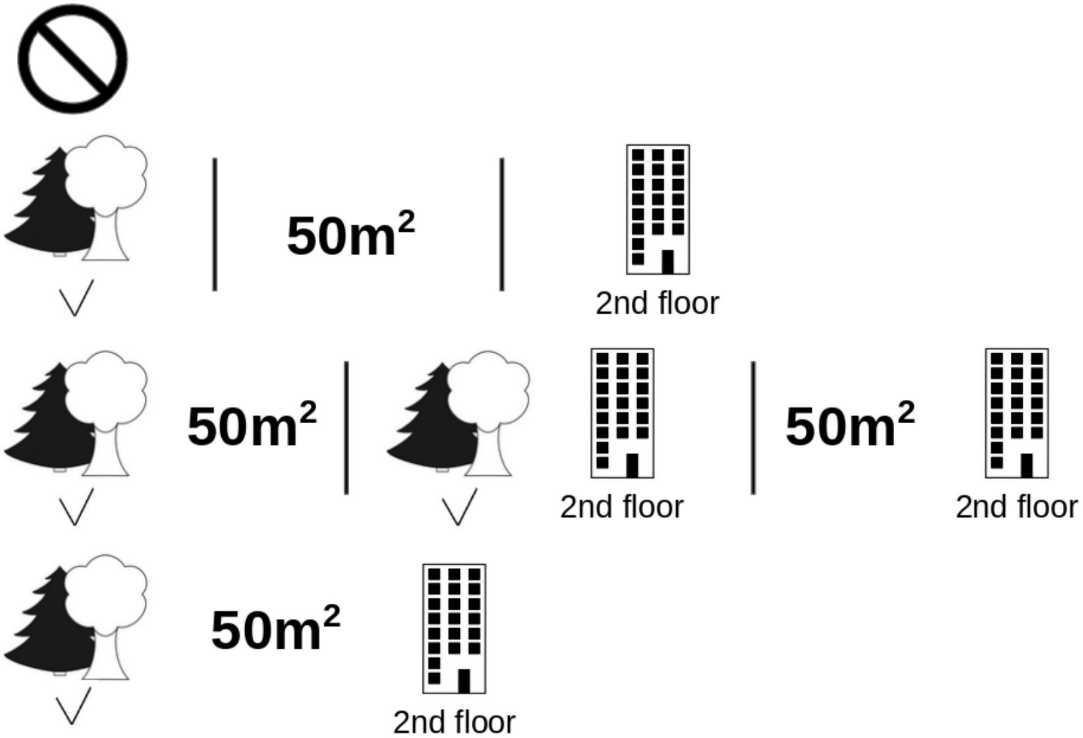
Soup Bowl (Smoothgrad)



Soup Bowl (Grad-Cam)



Shapley values: game theory

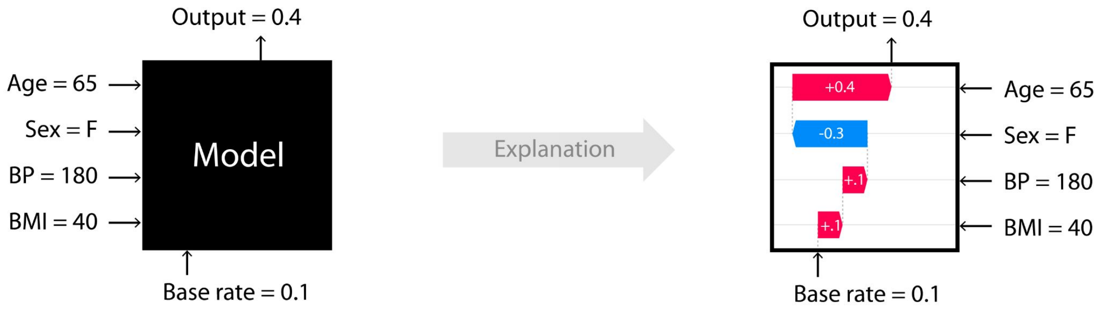


https://christophm.github.io/interpretable-ml-book/shapley

Shapley values: math

$$\phi_j(val) = \sum_{S \subseteq \{1,\ldots,p\} \setminus \{j\}} \frac{|S|!\,(p - |S| - 1)!}{p!} \left(val\left(S \cup \{j\}\right) - val(S)\right)$$

SHapley Additive exPlanations (SHAP)



Definition 1 **Additive feature attribution methods** *have an explanation model that is a linear function of binary variables:*

$$g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z'_i, \tag{1}$$

where $z' \in \{0,1\}^M$, *M is the number of simplified input features, and* $\phi_i \in \mathbb{R}$.

S. M. Lundberg & S. Lee. **A Unified Approach to Interpreting Model Predictions**. *NeurIPS*, 2017.
https://dl.acm.org/doi/10.5555/3295222.3295230

SHAP

1. (model-agnostic) **KernelSHAP**: LIME + SHAP kernel
2. TreeSHAP: fast SHAP values for tree-ensemble models)
3. Gradient: SHAP based on IG and Smoothgrad
4. *SHAP based on DeepLIFT https://arxiv.org/abs/1704.02685

https://christophm.github.io/interpretable-ml-book/shap

Google Colab

Neuron & layer explanations

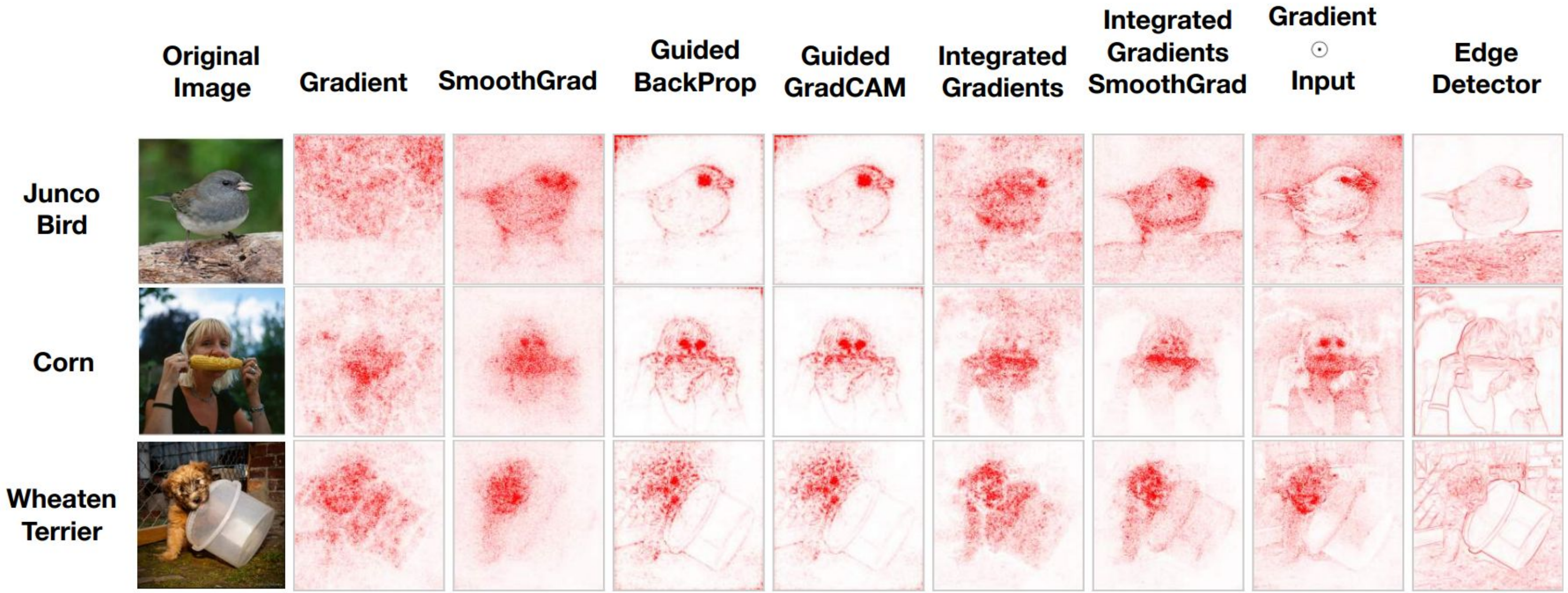
- Neuron Attribution
  - Neuron Gradient
  - Neuron Integrated Gradients
  - Neuron Conductance
  - Neuron DeepLift
  - Neuron DeepLiftShap
  - Neuron GradientShap
  - Neuron Guided Backprop
  - Neuron Deconvolution
  - Neuron Feature Ablation

- Layer Attribution
  - Layer Conductance
  - Layer Activation
  - Internal Influence
  - Layer Gradient X Activation
  - GradCAM
  - Layer DeepLift
  - Layer DeepLiftShap
  - Layer GradientShap
  - Layer Integrated Gradients
  - Layer Feature Ablation
  - Layer LRP

https://captum.ai/api

Google Colab (tasks)

Check!



J. Adebayo et al. **Sanity Checks for Saliency Maps**.
NeurIPS, 2018. https://arxiv.org/abs/1810.03292

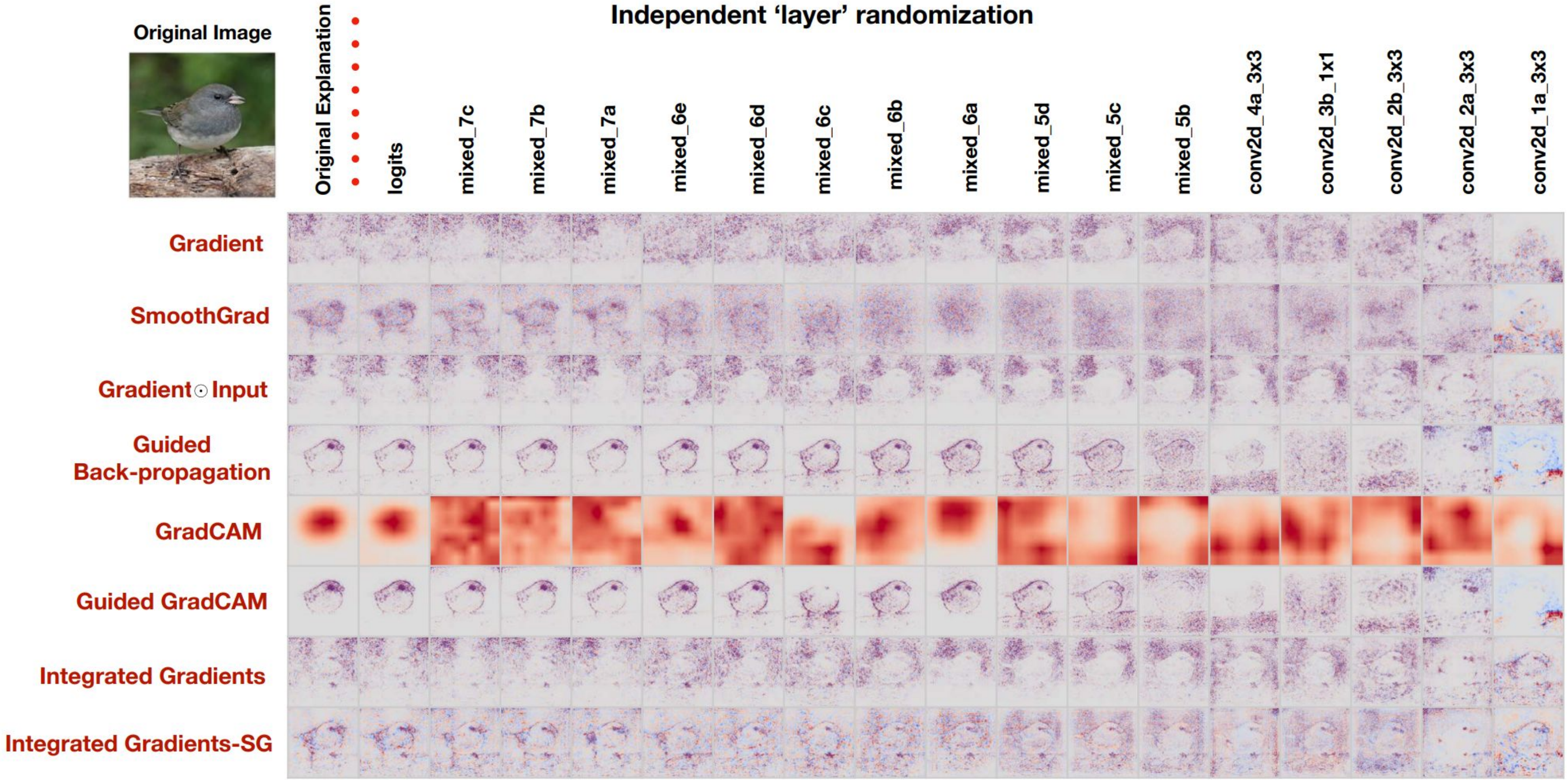


Figure 3: **Independent randomization on Inception v3 (ImageNet).** Similar to Figure 2, however

J. Adebayo et al. **Sanity Checks for Saliency Maps**. *NeurIPS*, 2018. https://arxiv.org/abs/1810.03292

More: the 2nd of June

- Concept-based Interpretability
  - TCAV
  - ConceptInterpreter
  - Concept https://captum.ai/api
  - Classifier

Concept-based explanations

https://christophm.github.io/interpretable-ml-book/detecting-concepts

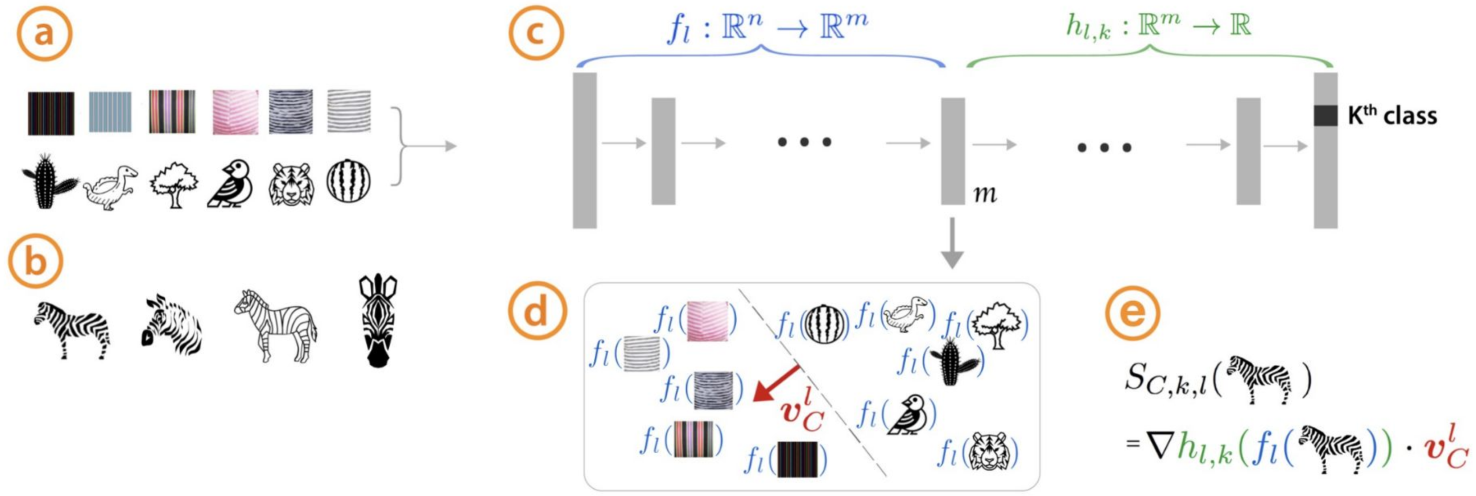


Figure 1. **Testing with Concept Activation Vectors:** Given a user-defined set of examples for a concept (e.g., 'striped'), and random

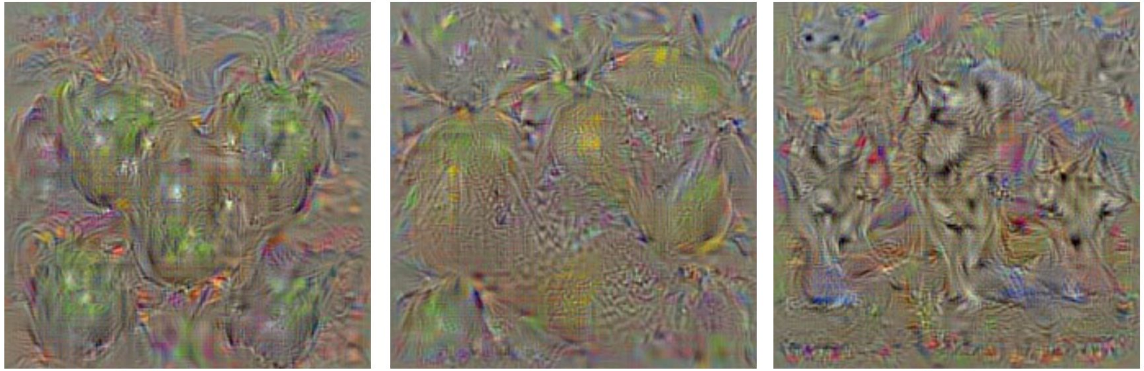
B. Kim et al. **Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV)**. *ICML*, 2018. https://arxiv.org/abs/1711.11279

Global explanations?

More formally, let $S_c(I)$ be the score of the class c, computed by the classification layer of the ConvNet for an image I. We would like to find an L_2-regularised image, such that the score S_c is high:

$$\arg\max_I S_c(I) - \lambda \|I\|_2^2, \quad (1)$$

where λ is the regularisation parameter. A locally-optimal I can be found by the back-propagation



bell pepper **lemon** **husky**

M. Ibrahim et al. **Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps**. *ICLR*, 2014. https://arxiv.org/abs/1312.6034

Global explanations

Formally, we may pose the activation maximization problem for a unit with index j on a layer l of a network Φ as finding an image $\mathbf{x}^*$ where:

$$\mathbf{x}^* = \arg\max_{\mathbf{x}} (\Phi_{l,j}(\mathbf{x}) - R_\theta(\mathbf{x}))$$

Here, $R_\theta(\mathbf{x})$ is a parameterized regularization could include multiple regularizers (i.e. pric which penalizes the search in a different wa

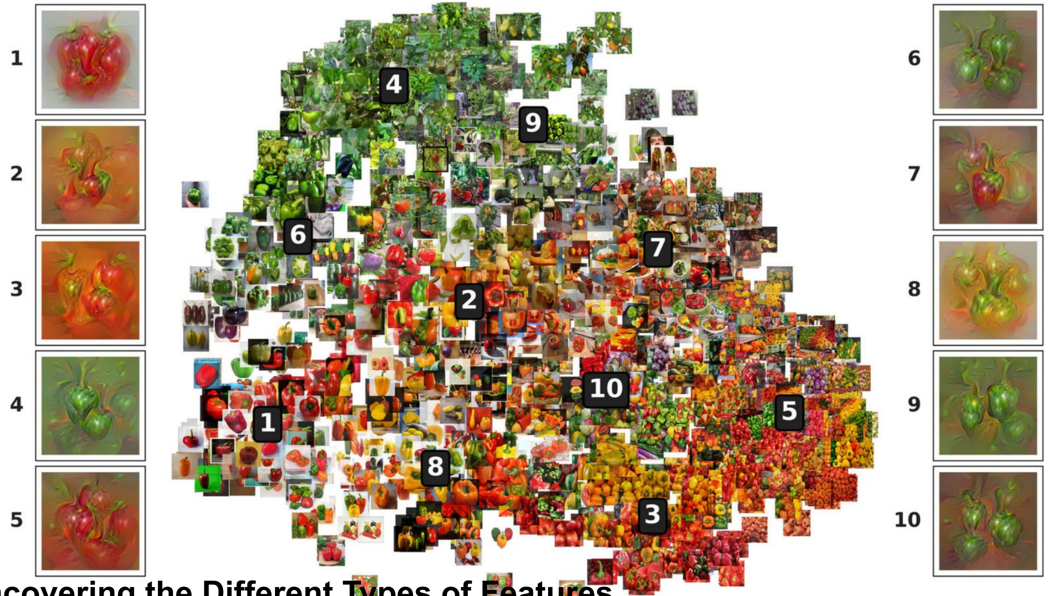


Figure 4. Visualizing the different facets of a neuron that detects bell peppers. Diverse facets include a single, red bell pepper on a white

A. Nguyen et al. **Multifaceted Feature Visualization: Uncovering the Different Types of Features Learned By Each Neuron in Deep Neural Networks**. *ICML*, 2016. https://arxiv.org/abs/1602.03616

Figure 5. Visualizing the different facets of a neuron that detects images in the "fishing reel" class. Diverse facets include reels on

A. Nguyen et al. **Multifaceted Feature Visualization: Uncovering the Different Types of Features Learned By Each Neuron in Deep Neural Networks**. *ICML*, 2016. https://arxiv.org/abs/1602.03616

Global explanations: from neuron to layers

A. Nguyen et al. **Multifaceted Feature Visualization: Uncovering the Different Types of Features Learned By Each Neuron in Deep Neural Networks**. *ICML*, 2016. https://arxiv.org/abs/1602.03616

More recently: another aggregate/cluster approach

M. Ibrahim et al. **Global Explanations of Neural Networks: Mapping the Landscape of Predictions**. *AIES*, 2019. https://arxiv.org/abs/1902.02384

Algorithm 1: Generating Global Attributions (GAM)

Input: local attributions
Output: medoids and corresponding members

```
/* 1. Normalize the set of local
   attributions                              */
foreach local attribution do
    normalized = abs(attribution) /
     sum(abs(attribution))
end

/* 2. Compute pair-wise rank
   distance matrix                           */
distances = []

foreach attribution1 in normalizedAttributions do
    foreach attribution2 in normalizedAttributions
     do
        distances += rankDistance(attribution1,
         attribution2)
    end
end

/* 3. Cluster Attributions                   */
initialMedoids = random.choice(attributions)

for x iterations do
    foreach cluster do
        foreach attribution in cluster do
            tempMedoid = attribution;
            cost = sum(distance(attribution,
             tempMedoid));
            reassign medoid to attribution
             minimizing cost;
        end
        update cluster membership by assigning to
         closest medoid
    end
end
```

Projekt