Explainability Evaluation and Global Explainability

CPSC680: Trustworthy Deep Learning

Rex Ying

Readings

- Readings are updated on the website (syllabus page)
- **Lecture 6 readings**:
  - 1710.10547.pdf (arxiv.org) Explanations can also be vulnerable to adversarial attacks
  - 2005.00631.pdf (arxiv.org) Evaluating Explanations

Content

- Evaluating Explainability Methods
- Model-level Explainability
- Intrinsic Explainability / Interpretability

Content

- Evaluating Explainability Methods
- Model-level Explainability
- Intrinsic Explainability / Interpretability

Criteria of Good Explanation

- Fidelity

The explanation maximally supports model's prediction

- Sensitivity

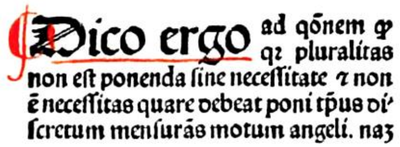
The explanation is stable for similar model input and output

- Conciseness

The explanation cannot be too large (Occam's razor)

- Interpretability

The explanation can be easily understood by human


Dico ergo ad q̃onem q̄ q2 pluralitas non est ponenda sine necessitate 7 non ē necessitas quare debeat poni tp̄us di' scretum mensurās motum angeli. naz

"pluarity is not to be posited without necessity"

Explanation Goal

- **Phenomenon** Explanation
  - Explain the underlying reasons for the ground truth phenomenon

- **Model** Explanation
  - Explain why model makes a particular prediction

- We will explain the **fidelity** metric in both cases:

Explanation Goal: Fidelity Metric

- Define 2 fidelity metrics: fid_+ and fid_- to capture different aspects of **explanation quality**
- The formula of fidelity depends on the goal:
  - **Goal 1**: explain **phenomenon** of the data
  - **Goal 2**: explain what has the **model** learned

Goal

Phenomenon

prediction

$$fid_+ = \frac{1}{N}\sum_{i=1}^{N}\left|\mathbb{1}(\hat{y}_i = y_i) - \mathbb{1}(\boxed{\hat{y}_i^{G_{C\setminus S}}} = \boxed{y_i})\right|$$

groundtruth

$$fid_- = \frac{1}{N}\sum_{i=1}^{N}\left|\mathbb{1}(\hat{y}_i = y_i) - \mathbb{1}(\hat{y}_i^{G_S} = y_i)\right|$$

Model

$$fid_+ = 1 - \frac{1}{N}\sum_{i=1}^{N}\mathbb{1}(\hat{y}_i^{G_{C\setminus S}} = \hat{y}_i)$$

$$fid_- = 1 - \frac{1}{N}\sum_{i=1}^{N}\mathbb{1}(\hat{y}_i^{G_S} = \hat{y}_i)$$

Fidelity Metric Details

- **Characteristics of a good explanation**
- fid_+: removal important features will result in large decrease of the model confidence
- fid_-: Using only the important features will result in similar confidence

Phenomenon

$$fid_+ = \frac{1}{N}\sum_{i=1}^{N}\left|\boxed{\mathbb{1}(\hat{y}_i = y_i)} - \boxed{\mathbb{1}(\hat{y}_i^{G_{C\setminus S}} = y_i)}\right|$$

Removal of important features

$$fid_- = \frac{1}{N}\sum_{i=1}^{N}\left|\boxed{\mathbb{1}(\hat{y}_i = y_i)} - \boxed{\mathbb{1}(\hat{y}_i^{G_S} = y_i)}\right|$$

Keeping only the important features

Original prediction probability / confidence

Explanation Evaluation Criteria

- Notably, the explanation evaluation criteria are **multi-dimensional**
- **Explanation quality**
  - High fidelity / characterization scores
  - Sufficiency and necessity aspects (see the previous slide)
- **Explanation stability**
  - Explanations are consistent across random optimization seeds (measure variance)
- **Explanation complexity**
  - The explanation should be concise and easy to understand by human (measure size)

Types of Explanations

- **Sufficiency**
  - An explanation is sufficient if it leads by its own to the initial prediction of the model explanation. ($fid_- \rightarrow 0$)
- **Necessity**
  - An explanation is necessary if the model prediction changes when removing it from the initial graph. ($fid_+ \rightarrow 1$)
- Use the **Characterization** score to summarize the explanation quality

$$charact = \frac{w_+ + w_-}{\frac{w_+}{fid_+} + \frac{w_-}{1 - fid_-}} = \frac{(w_+ + w_-) \times fid_+ \times (1 - fid_-)}{w_+ \cdot (1 - fid_-) + w_- \cdot fid_+}$$

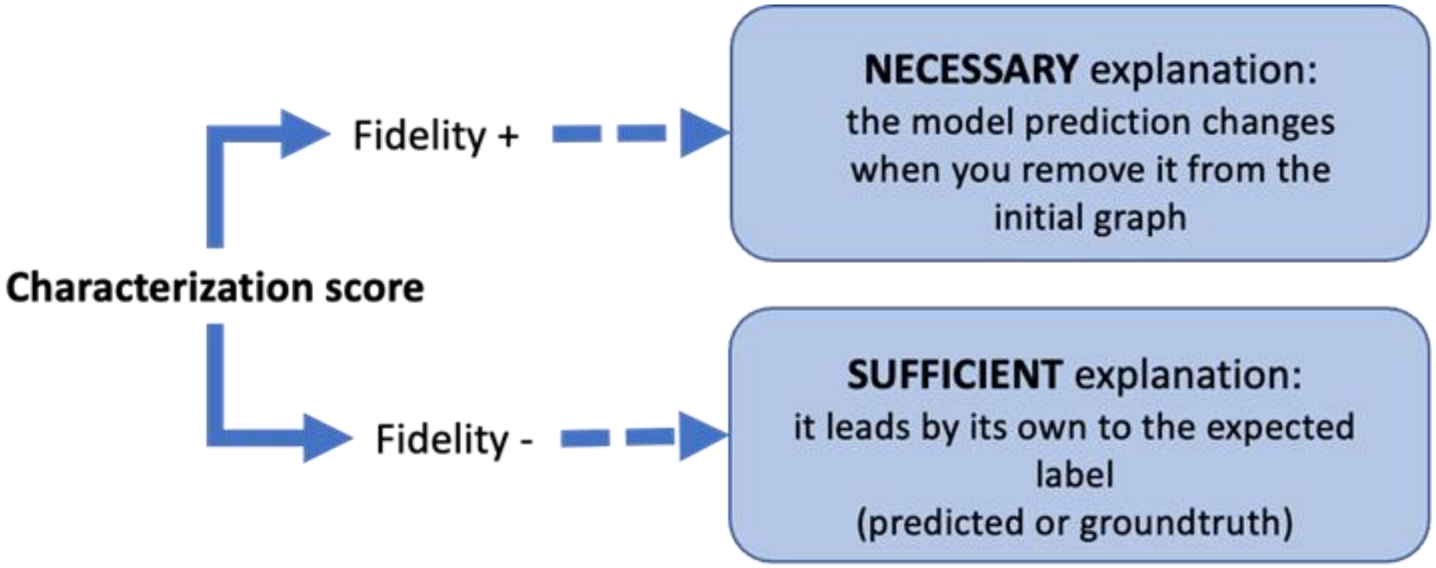
Where w_+ and w_- are the weights of both fidelity metrics (commonly set $w_+ = w_- = 1$)

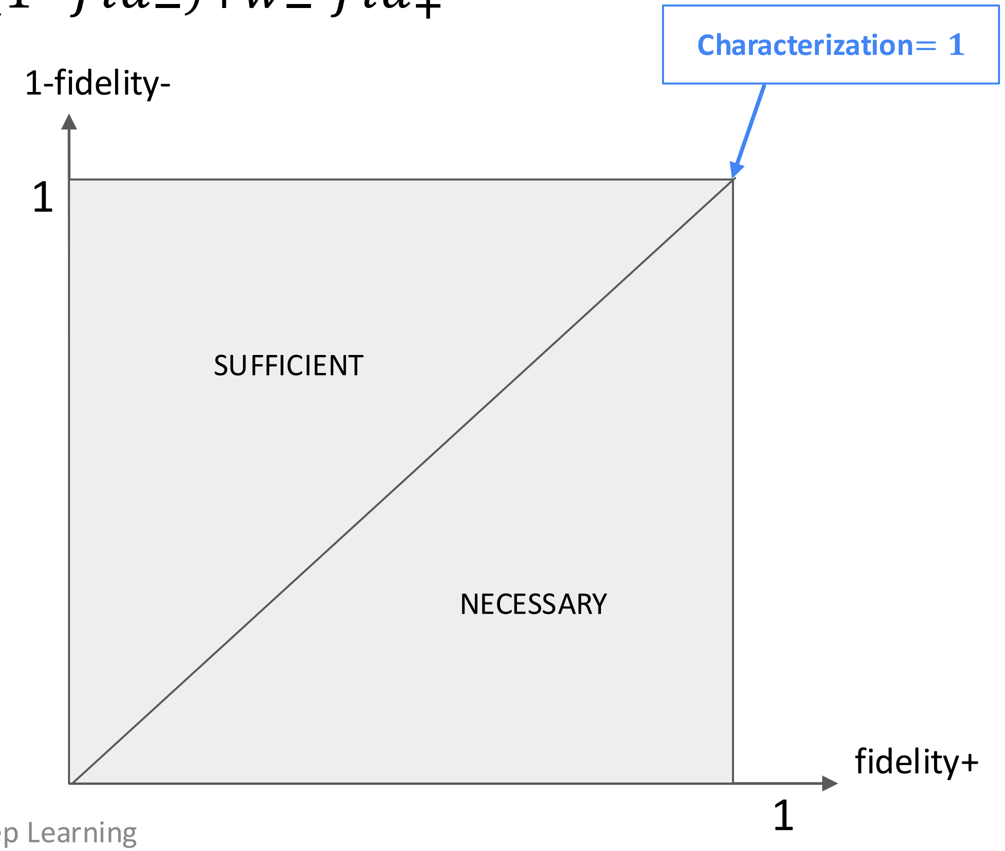
Characterization Score

- **Characterization** score to summarize the explanation quality

$$charact = \frac{w_+ + w_-}{\frac{w_+}{fid_+} + \frac{w_-}{1 - fid_-}} = \frac{(w_+ + w_-) \times fid_+ \times (1 - fid_-)}{w_+ \cdot (1 - fid_-) + w_- \cdot fid_+}$$

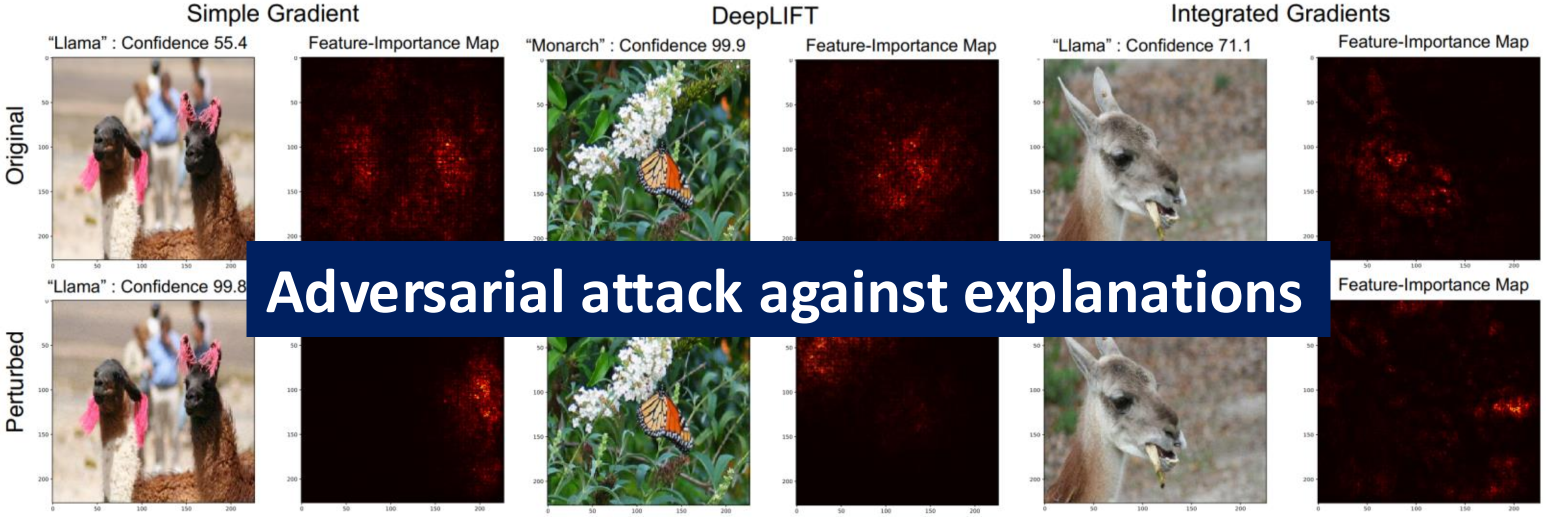
- Necessary AND sufficient





Sensitivity Desiderata

- Similar input & output → explanations should be similar
- Also called "stability"
- Local smoothness is not usually true for deep neural networks, but is a very common assumption in human cognition



Sensitivity Definition

- Define the neighborhood of a point of interest $\boldsymbol{x}$

$$\boldsymbol{N_r} = \{z \in D_{\boldsymbol{x}} | \rho(x, z) \le r, f(x) = f(z)\}$$

  - The local region around prediction of $\boldsymbol{x}$ that's stable

- Max Sensitivity μ_M

$$\mu_M(f, g, r; x) = \max_{z \in \boldsymbol{N_r}} D(g(f, x), g(f, z))$$

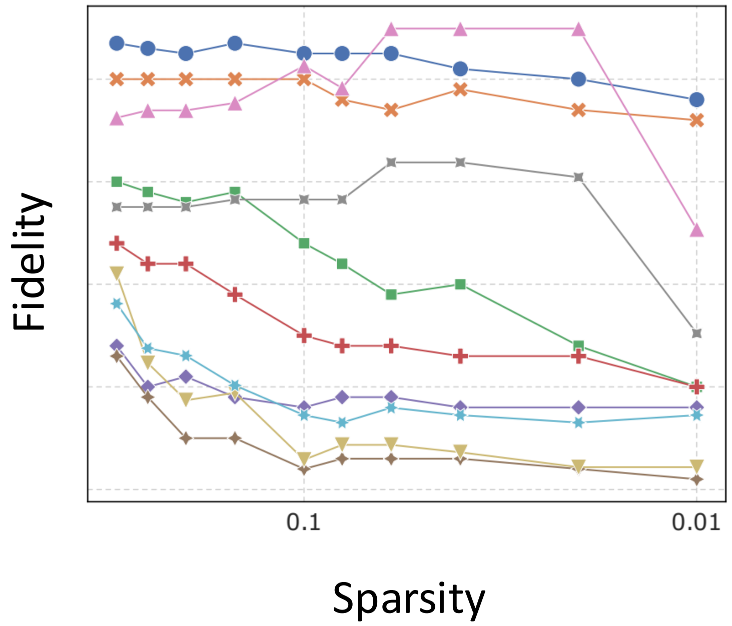
- Average Sensitivity

$$\mu_A(f, g, r; x) = \int_{z \in \boldsymbol{N_r}} D(g(f, x), g(f, z))\, dz$$

ρ: distances between input features
D: distance metric between explanation results
g: explanation method

Conciseness – Low Complexity

- **Sparsity** is important to ensure that the model explanation highlights the most relevant part of the input
- Sparsity can be measured by the **size** of the explanation
  - Often controlled in the experiments
- Sparsity can also be measured by **entropy**
  - For explanations with importance scores
  - Attribution methods, Mask-based methods etc.



Allows us to investigate the tradeoff

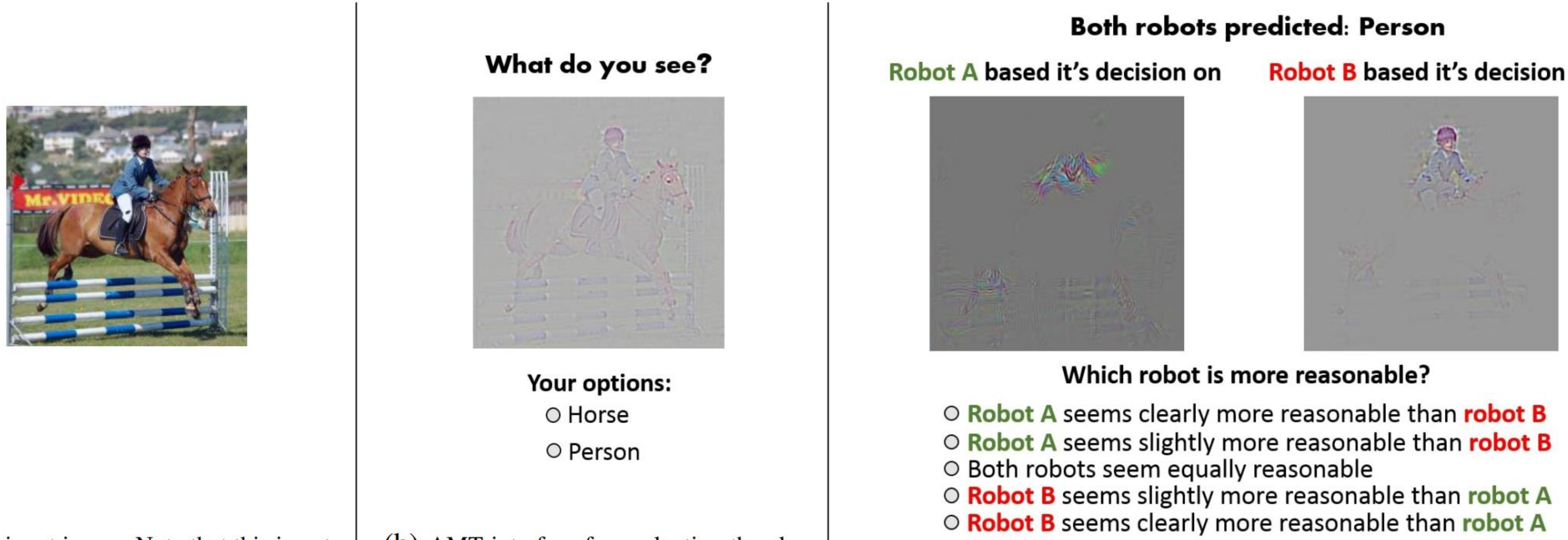
Global Explainability Evaluation

- Relative performance loss

$$RPF = \frac{(\log \mathcal{L}(M_{-F}) - \log \mathcal{L}(M))}{\log \mathcal{L}(M)}$$

- $\log \mathcal{L}(M)$: loss function value on all test data
- $\mathcal{L}(M_{-F})$: loss function value after feature pruning (on all test data)
- Analogous to instance-level fidelity

Human Evaluation



(a) Raw input image. Note that this is not a part of the tasks (b) and (c)

(b) AMT interface for evaluating the class-discriminative property

(c) AMT interface for evaluating if our visualizations instill trust in an end user

Utilizes human evaluation platform such as Amazon Mechanical Turk (AMT)

Content

- Evaluating Explainability Methods
- Global-level Explainability
- Intrinsic Explainability / Interpretability

Model-level Explanation

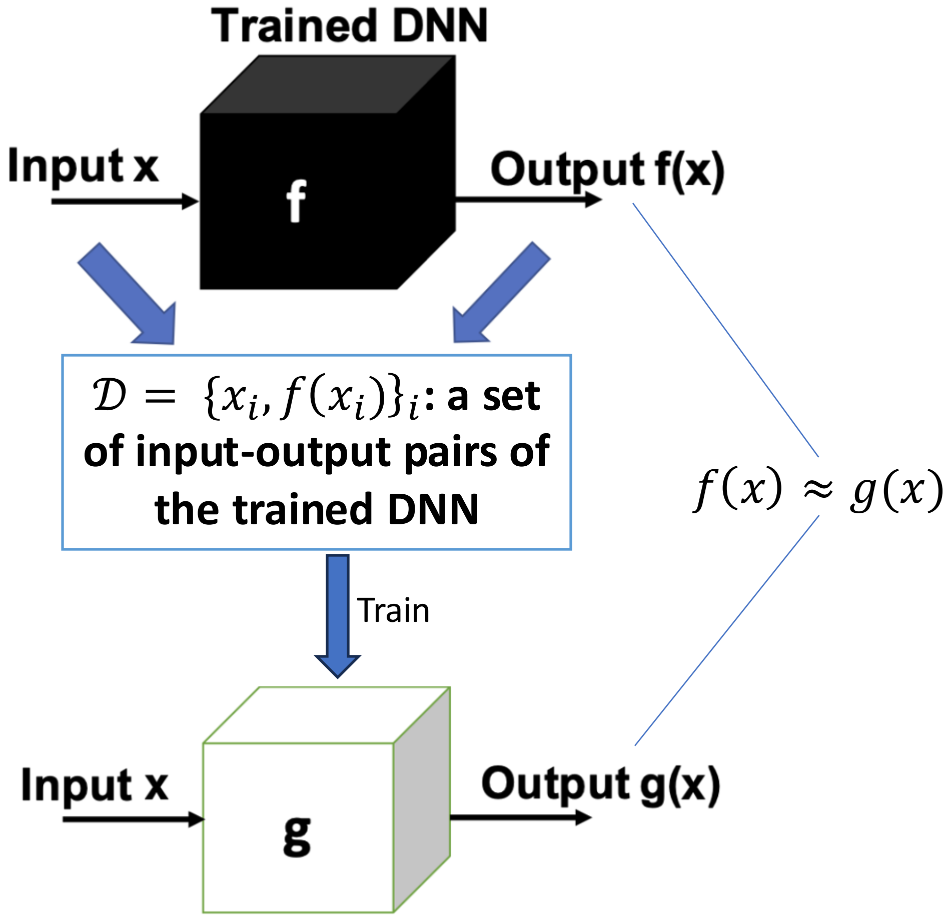
- **Model-level explanations** aim to shed light on a model's overall decision-making process on ***a set of inputs***, instead of a specific instance.
- provides a **bird-eye-view** of the model behavior, analyzing potential bias affecting a group/subgroup of instances.
- Examples:
  - Concept-based explanations: provide importance measurement for high-level concepts, instead of individual features or pixels.
  - Influence functions: measure the impact of each data point in the training set on the model’s predictions

Global Explanation via Model Distillation

- **Generalized Additive Model (GAM)**

$$g(x) = h_0 + \underbrace{\sum_i h_i(x_i)}_{\text{Functions of individual features}} + \underbrace{\sum_{i \neq j} h_{ij}(x_i, x_j) + \sum_{i \neq j} \sum_{i \neq k} h_{ijk}(x_i, x_j, x_k) + \ \ldots}_{\text{Higher-order feature interaction terms}}$$

What are the potential issues?



Surrogate model g is inherently interpretable

Concept Definition

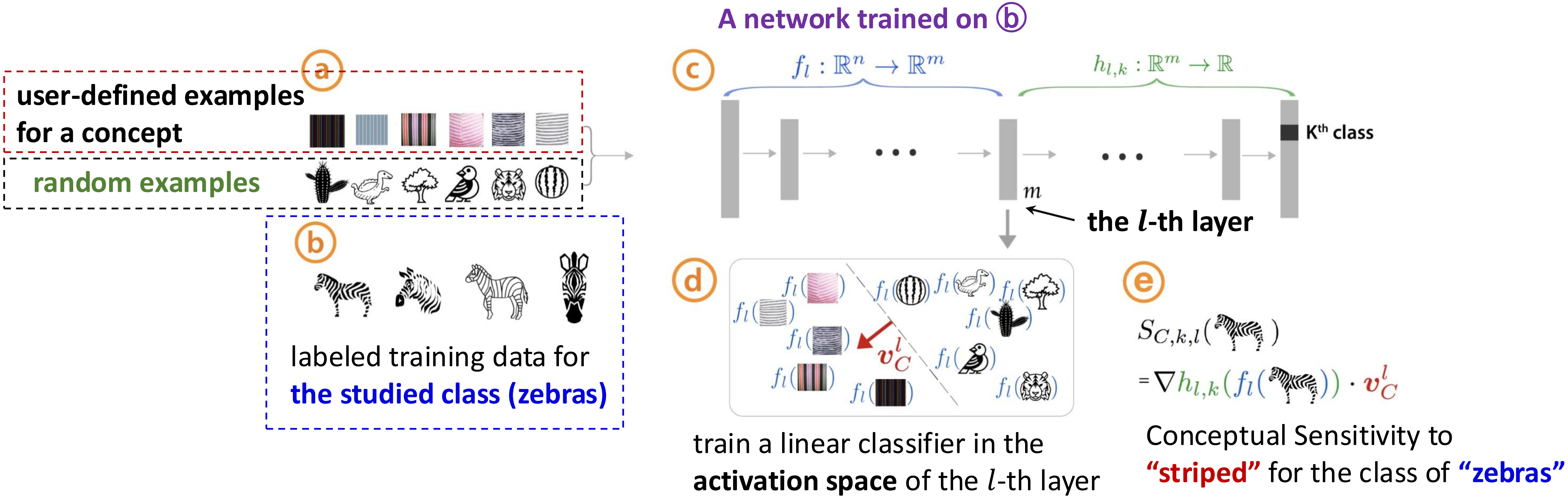
- **Concept**: high-level units that are more understandable to human than individual features, pixels, etc.
- For example, **the wheel** and the **police logo** are important concepts for police vans.

concept 1: **wheel**

concept 2: **police logo**

TCAV Pipeline

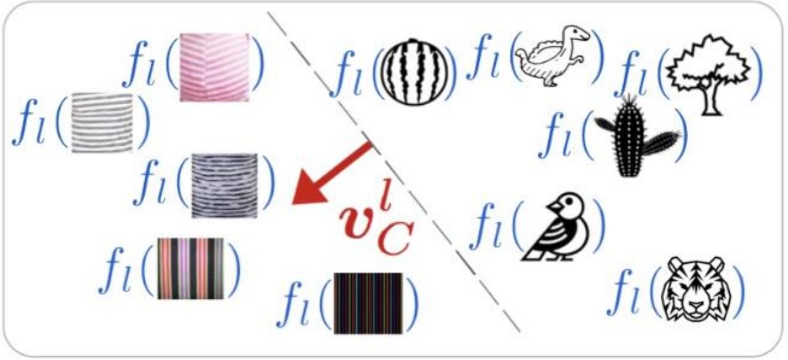
- Testing with Concept Activation Vectors (CAV) [paper]



CAV Definition

- For a user-defined concept, we seek a **vector in the embedding space** of the l-th layer that represents this concept
- Concept Activation Vector (CAV): a unit vector orthogonal to the classification boundary

Activation produced by **random examples**



v_C^l: CAV for a concept C in layer l

Activation produced by inputs with the studied concept

Train **a linear classifier** to separate examples *without* a concept and examples *with* a concept

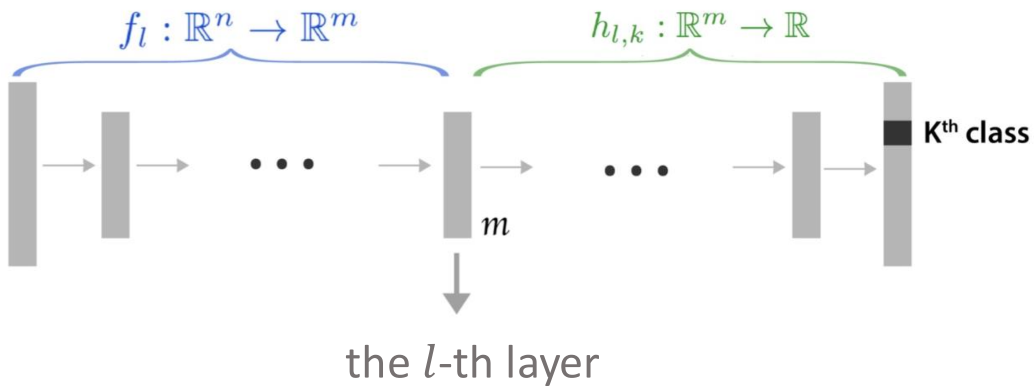
Conceptual Sensitivity

- $f_l(\boldsymbol{x})$: the activations for input $\boldsymbol{x}$ at layer l ; $h_{l,k}(f_l(\boldsymbol{x}))$: the logit for class k
- **Sensitivity of class k to concept $C \in \mathbb{R}$:**

$$S_{C,k,l}(\boldsymbol{x}) = \lim_{\epsilon \to 0} \frac{h_{l,k}\left(f_l(\boldsymbol{x}) + \epsilon \boldsymbol{v}_C^l\right) - h_{l,k}\left(f_l(\boldsymbol{x})\right)}{\epsilon}$$

$$= \nabla h_{l,k}\left(f_l(\boldsymbol{x})\right) \cdot \boldsymbol{v}_C^l,$$

$\boldsymbol{v}_C^l \in \mathbb{R}^m$: CAV for a concept C in layer l



Testing with CAVs

- TCAV score is defined as:

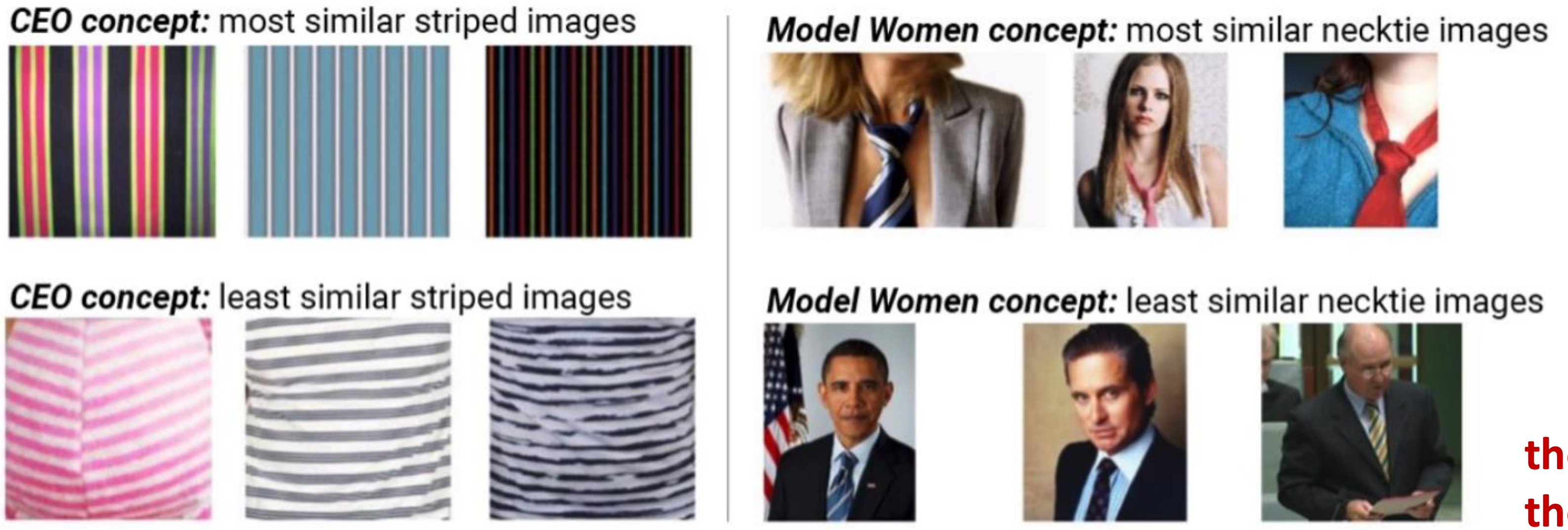
$$\mathrm{TCAVQ}_{C,k,l} = \frac{\left|\left\{\boldsymbol{x} \in X_k : S_{C,k,l}(\boldsymbol{x}) > 0\right\}\right|}{|X_k|} \in [0,1]$$

- X_k: all inputs with the class k
- TCAV measures the fraction of inputs with the class k whose l-th layer activation vector was **positively sensitive** to concept C (i.e., $S_{C,k,l}(\boldsymbol{x}) > 0$)
- Note: TCAV only depends on the **sign** of $S_{C,k,l}$ (**sensitivity**)
  - could be further improved to consider the magnitude

Example: Sorting Images with CAVs

- CAV essentially encodes the direction of a concept.
- The **cosine similarity** between the picture of interest to the CAV reflects the relation between the picture and the concept.
  - First learn a CAV from CEO / Model Women class (collected from ImageNet)
  - Sort similar/dissimilar images with respect to the learned CAVs

A dataset of
Strip Images



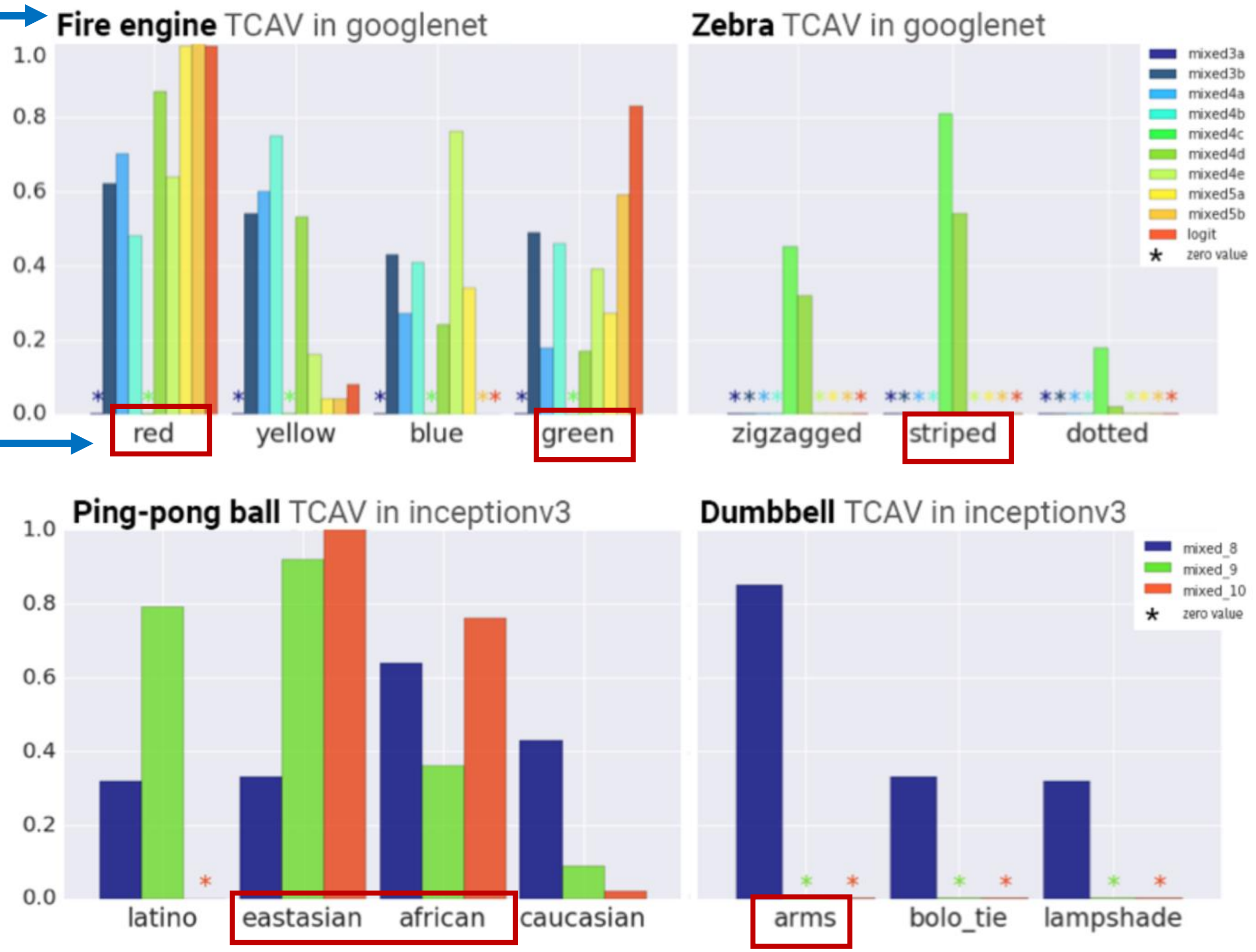
A dataset of
Tie Images

the CAVs correctly reflect
the concept of interest

TCAV Results

Class name

X-axis: concepts



different layers in Googlenet

Last 3 layers in Inception v3

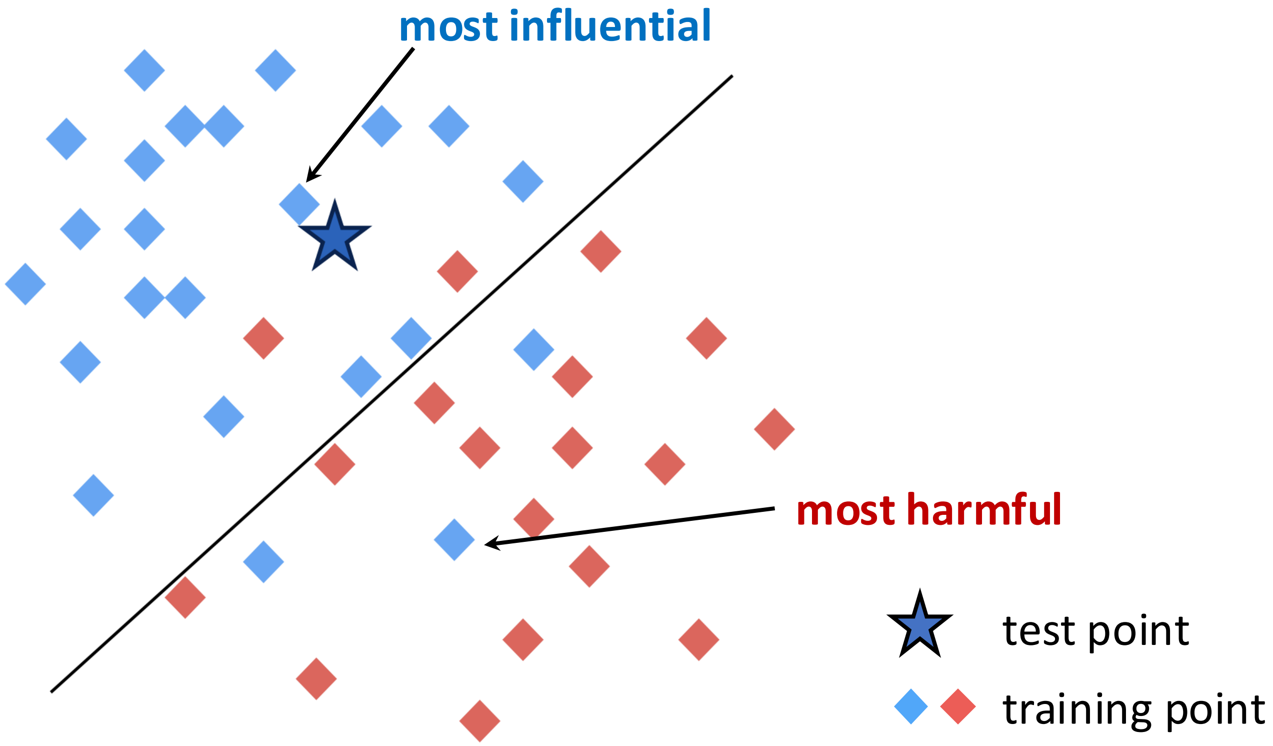
Concept with high TCAV

TCAVQs in layers close to the logit layer (red) represent more direct influence on the prediction than lower layers in general.

Influence Functions: Motivation

Given a well-trained deep learning model, we are interested in

- Which training points were most **influential** for this prediction?
- Which training points were most **harmful** for the prediction?



Influence Functions: Setting (1)

- **Question:** How to measure the impact of a training point on a prediction?
- We are given training points $z_1, \dots, z_n$. How to measure the impact of a training point z_{train} on the prediction of z_{test}
- Instead of retraining the model on $\hat{Z} = \{z_i\}_{i=1}^{n} \cup z_{train}$, we use influence functions to measure the model changes as we upweight z_{train} by **an infinitesimal amount**

Influence Functions: Setting (2)

- Let $L(z_i, \theta)$ be the loss, where $\theta \in \Theta$ represents model parameters.
- $\hat{\theta} \coloneqq argmin_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta)$ is the original optimal parameters
- Given $\hat{Z} = \{z_i\}_{i=1}^{n} \cup z_{train}$, the optimal parameters become:

$$\hat{\theta}_{\varepsilon, z_{train}} \coloneqq argmin_{\theta \in \Theta} \boxed{\frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta)} + \varepsilon L(z_{train}, \theta)$$

Assumption: the empirical risk is twice-differentiable and strictly convex in θ.

- Goal: approximate the change in $L(z_{test}, \hat{\theta}_{\varepsilon, z_{train}})$ as we increase ε

In order to measure the influence of the example z_{train}

Influence Functions: Definition

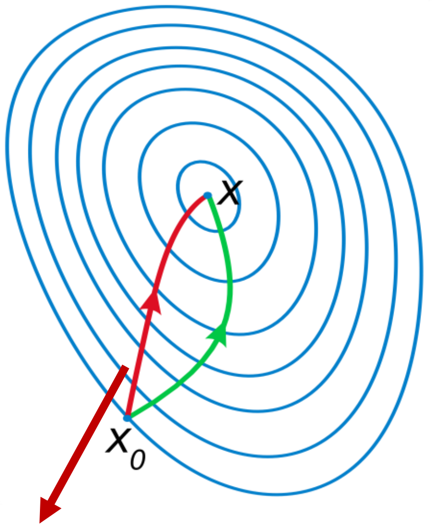
- Under smoothness assumptions:

$$\mathcal{J}_{up,loss}(z_{train}, z_{test}) \stackrel{\text{def}}{=} \left. \frac{dL(z_{test}, \hat{\theta}_{\varepsilon, z_{train}})}{d\epsilon} \right|_{\epsilon=0}$$

$$= \nabla_\theta L(z_{test}, \hat{\theta})^\top \left. \frac{d\hat{\theta}_{\varepsilon, z_{train}}}{d\epsilon} \right|_{\epsilon=0}$$

$$= -\nabla_\theta L(z_{test}, \hat{\theta})^\top H_{\hat{\theta}}^{-1} \nabla_\theta L(z_{train}, \hat{\theta})$$

  - where $H_{\hat{\theta}} = \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta^2 L(z_i, \hat{\theta})$



Newton's method uses 2nd order derivatives

- In essence, influence functions form a **quadratic approximation (via Hessian)** to the empirical risk around $\hat{\theta}$ and take a single Newton step.

Cook, R. D. and Weisberg, S. Residuals and influence in regression. New York: Chapman and Hall, 1982

Use case: Understand Model Behavior (1)

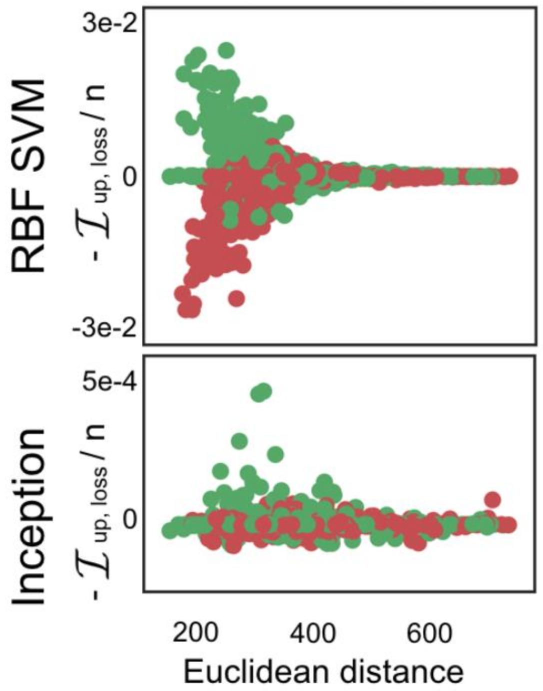
Influence functions reveal insights about how models rely on and extrapolate from the training data.

- Dataset: Dog & Fish image classification from ImageNet dataset
- Two well trained models: (1) Inception v3 network and (2) an SVM with an RBF kernel
- Investigate the impact of training points on a test image (**fish**) that both models got correct prediction

Test image

Use case: Understand Model Behavior (2)

$\mathcal{I}_{up,loss}(z_{train}, z_{test})$ V.S. Euclidean distance $||z_{train} - z_{test}||$



In RBF-SVM: training images far from the test image in pixel space having almost no influence;
(emphasizing nearby samples)
Fish images are mostly helpful, while **dog** images are mostly harmful

In Inception network, fish and **dogs** both could be helpful or harmful for correctly classifying the test image. The influence is not related to the distance.

Green dot: fish
Red dot: dog
Note: the test image is a ***fish*** *image*

Most helpful training images for **RBF-SVM**

Most helpful training images for **Inception**

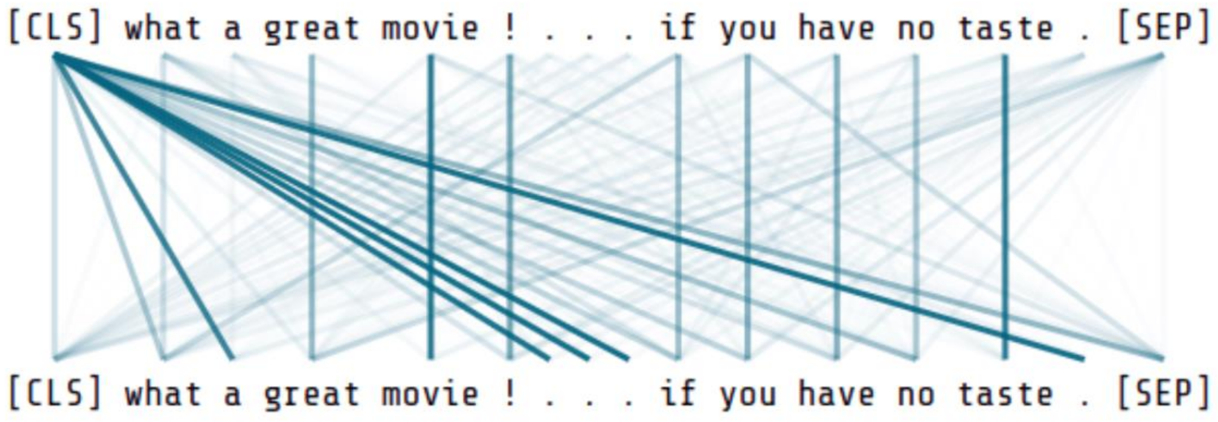
the 5th most helpful training image for **Inception** is a dog image

Content

- Evaluating Explainability Methods
- Model-level Explainability
- Intrinsic Explainability / Interpretability

Explainability via Attention

- **Attention Mechanisms**
  - DNNs can be endowed with attention mechanisms that simultaneously
    - preserve or even **improve their performance**
    - obtain **explainable outputs**
  - Visualize attention weights in an attention model:



Color represents the value of attention weight
darker blue ⟺ larger attention weight

Attention in RNN

RNN model for Natural Language Translation

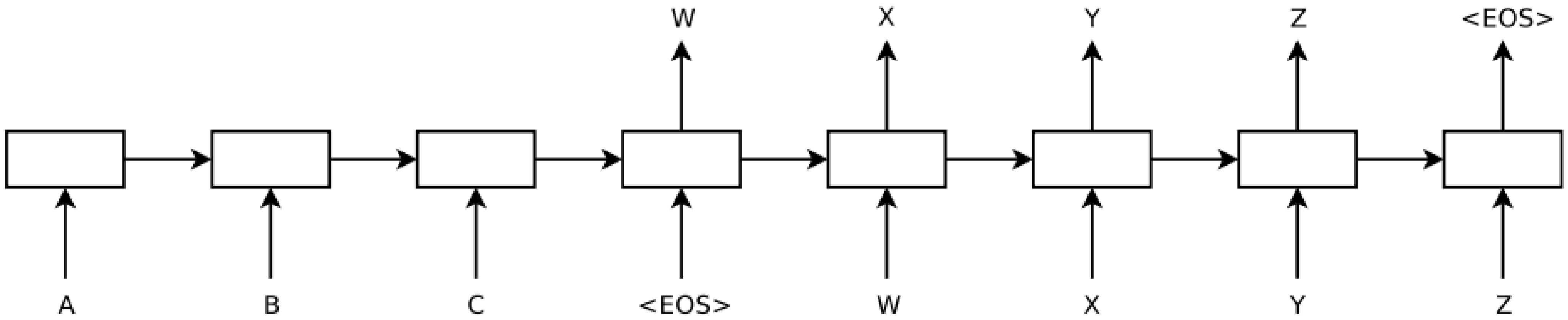


Figure 1: Our model reads an input sentence "ABC" and produces "WXYZ" as the output sentence.

How do we know which word(s) correspond to which word(s)?

Attention in RNN

Attention-based RNN model for Natural Language Translation

Attention score: $\boldsymbol{\alpha_{ij}} = \frac{\exp e_{ij}}{\sum_k \exp e_{ik}}$

where $e_{ij} = g(s_{i-1}, h_j)$

$\boldsymbol{c_i} = \sum_j \alpha_{ij} h_j$

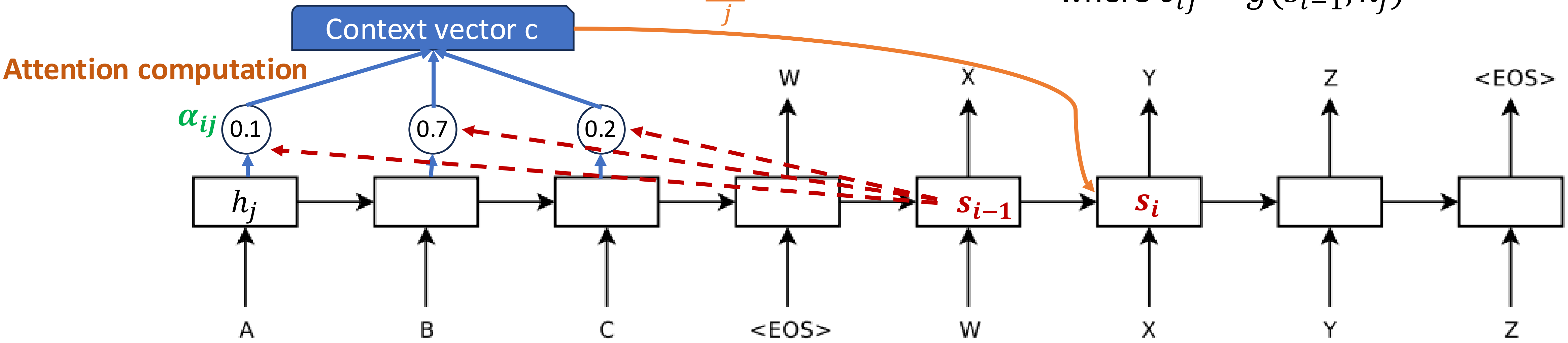


Figure 1: Our model reads an input sentence "ABC" and produces "WXYZ" as the output sentence.

Bahdanau et al. Neural Machine Translation by Jointly Learning to Align and Translate

Signed Attention (1)

- **Attention weight is always positive**
- Ideal explanation should discriminate between **positive** and **negative** contributions towards a prediction
- Solution: **signed attention**

$$A_i = -\boxed{\frac{\partial \mathcal{L}}{\partial \alpha_i}} \times \alpha_i$$

Indicates the **positive or negative contribution**

Recap: for hidden state $\mathbf{h_i}$

attention weight: $\alpha_i = \frac{\exp e_i}{\sum_k \exp e_k} \geq 0$

where $e_i = \mathbf{v}^\top \tanh(W_h \mathbf{h}_i + W_q \mathbf{q})$

$\mathbf{v}, \mathbf{q}, W_h, W_q$: learnable parameters

  - $\mathcal{L}$: loss function
  - α_i: original attention weight
  - value of α_i measures the strength of the contribution

Signed Attention (2)

Explanation for sentiment analysis

Input: *"Though the price may be tooexpensive, I love its surprisingly high quality."*
Output: y ="Positive"

original attention visualization

Though the price may be too expensive, I love its surprisingly high quality.

unimportant important

with signed attention

signed attention visualization

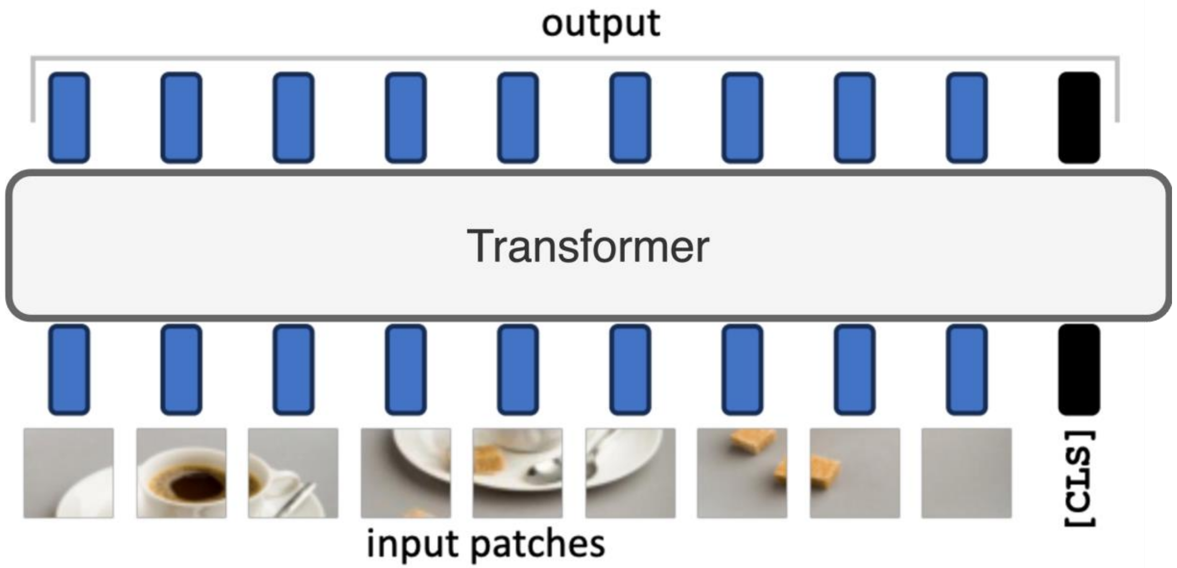
Though the price may be too expensive, I love its surprisingly high quality.

negative neutral positive

Liu, Shengzhong, et al. "On exploring attention-based explanation for transformer models in text classification."

Explainability in Vision Transformer (ViT)

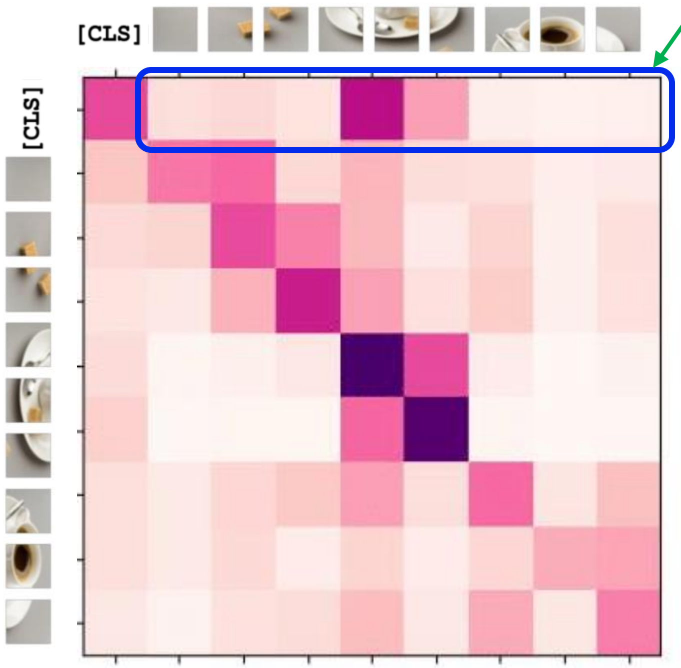
- **Vision Transformer**
  - Split an input image into MxM patches
  - Add a [CLS] token as a global embedding of the input



Architecture

resize it to (3x3) grid and run interpolation to smoothen the importance score to visualize

Importance score for patches



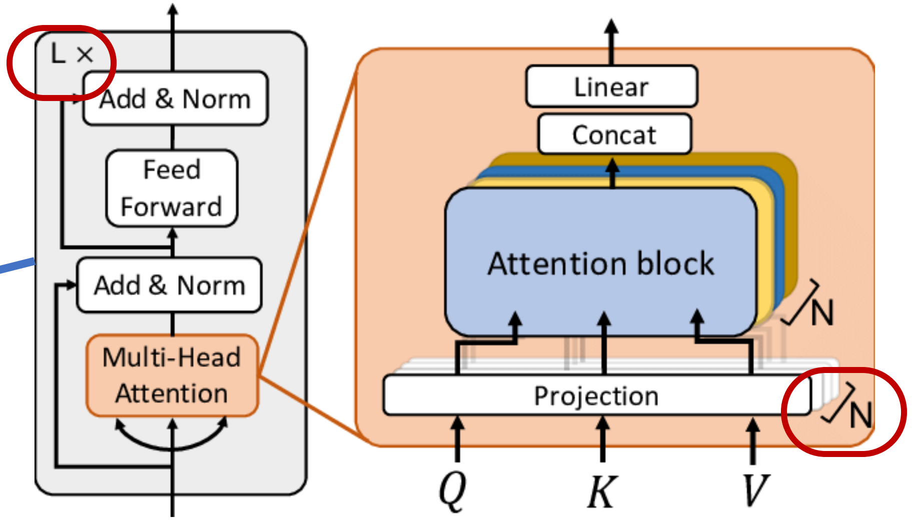
Attention Matrix: $A = softmax(\frac{QK^T}{\sqrt{d_k}})$

Explainability in Vision Transformer (ViT)

- **Vision Transformer**
  - Split an input image into MxM patches
  - Add a [CLS] token as a global embedding of the input

- **Challenge**:
  - Transformer has multiple heads and layers, thus having multiple attention matrices
    → **Which attention matrix to visualize?**

$N \times L$ attention matrices



Explainability in Vision Transformer (ViT)

- **Naïve approach (Rollout)**
  - Aggregate attention matrices across multiple heads: **Mean averaging**

$$\bar{A}^{(l)} = \boxed{I +} \sum_i A^{(l,i)}$$

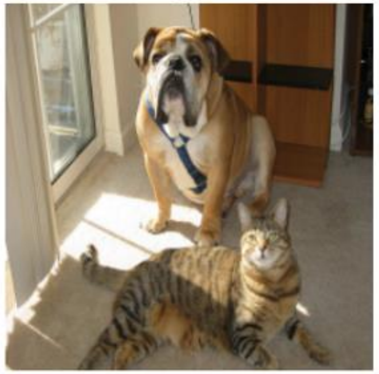
Why do we need to add I?

  - Aggregate attention matrices across multiple layers: **Matrix multiplication**

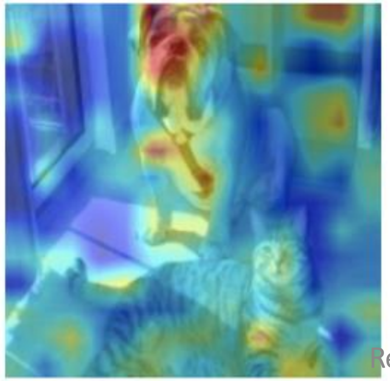
$$C = \bar{A}^1 \bar{A}^2 \boxed{\ldots \bar{A}^L}$$

Why should it be matrix multiplication?

Dog →

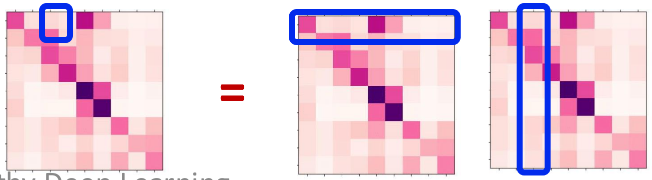


Attention visualization



Importance of patch 3

Attention weights of patch 3 to others



Importance of each patch

Explainability in Vision Transformer (ViT)

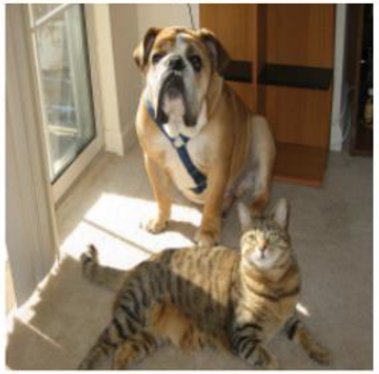
- **Naïve approach (Rollout)**
  - Aggregate attention matrices across multiple heads: **Mean averaging**

$$\bar{A}^{(l)} = I + \sum_{i} A^{(l,i)}$$

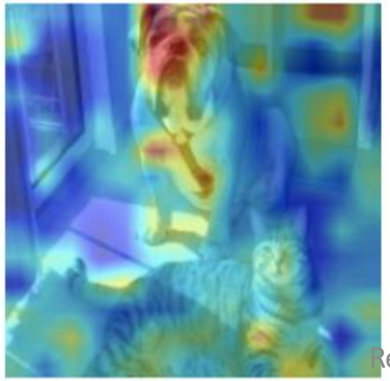
  - Aggregate attention matrices across multiple layers: **Matrix multiplication**

$$C = \bar{A}^{1} \bar{A}^{2} \ldots \bar{A}^{L}$$

Dog →

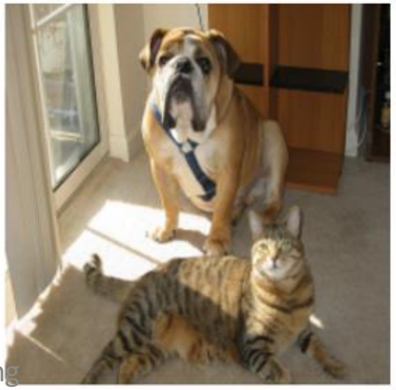


Attention visualization

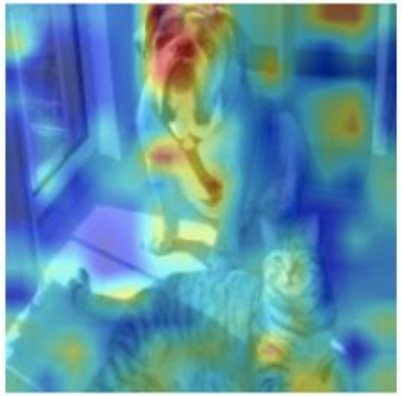


However, the explanation for cat prediction is the same as for dog

Cat →



Attention visualization



Explainability in Vision Transformer (ViT)

- **Targeted Explanation**
  - Aggregate attention matrices across multiple heads: **Relevance and gradient diffusion**

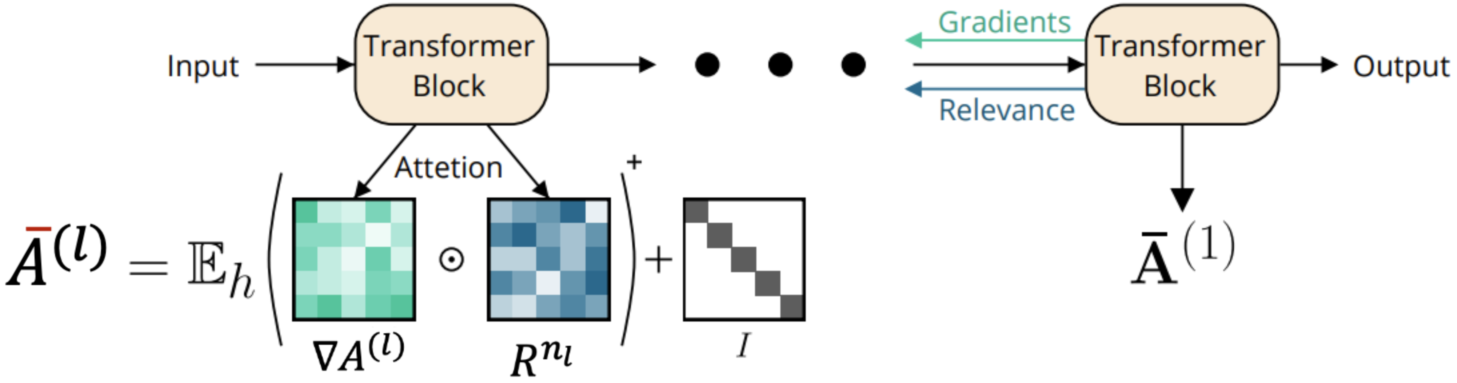
$$\bar{A}^{(l)} = I + \sum_h \boxed{\nabla A^{(l)}} \odot \boxed{R^{n_l}}$$

The gradient of the targeted output to the attention matrix

Layer-wise Relevance Propagation matrix (to be defined)

  - Aggregate attention matrices across multiple layers: **Matrix multiplication**

$$C = \bar{A}^1 \bar{A}^2 \ldots \bar{A}^L$$



Layer-wise Relevance Propagation (LRP) (1)

- **Layer-wise Relevance Propagation** (LRP)
  - LRP is a method to compute the relevance of input to the target output using the weights and the neural activations to propagate the output back through the network up until the input layer

$$R_j^{(l-1)} = \sum_k \frac{\boxed{z_{jk}}}{\sum_i z_{ik}} R_k^l$$

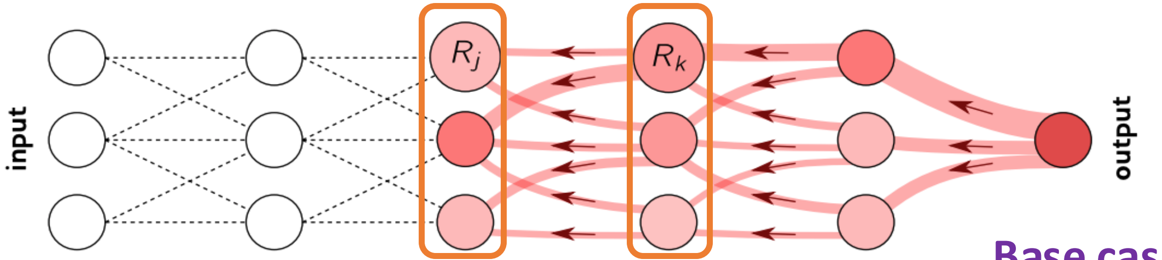
A number quantifying the contribution of a neuron j at layer $(l-1)$ to the relevance score of neuron k at layer (l)

E.g., for a linear layer with ReLU activation: $\boldsymbol{y}^{(l)} = ReLU\left(W^\top \boldsymbol{x}^{(l-1)}\right)$, the relevance propagation can be computed as follows

$$R_j^{(l-1)} = \sum_k \boxed{\frac{x_j^{(l-1)} \cdot W_{jk}}{\boxed{\sum_{j'} x_{j'}^{(l-1)} \cdot W_{j'k}}}} R_k^{(l)}$$

$y_k^{(l)}$

Contribution of the input neuron j to output neuron $y_k^{(l)}$



$$\sum_j R_j^{(l-1)} = \sum_k R_k^{(l)} = output$$

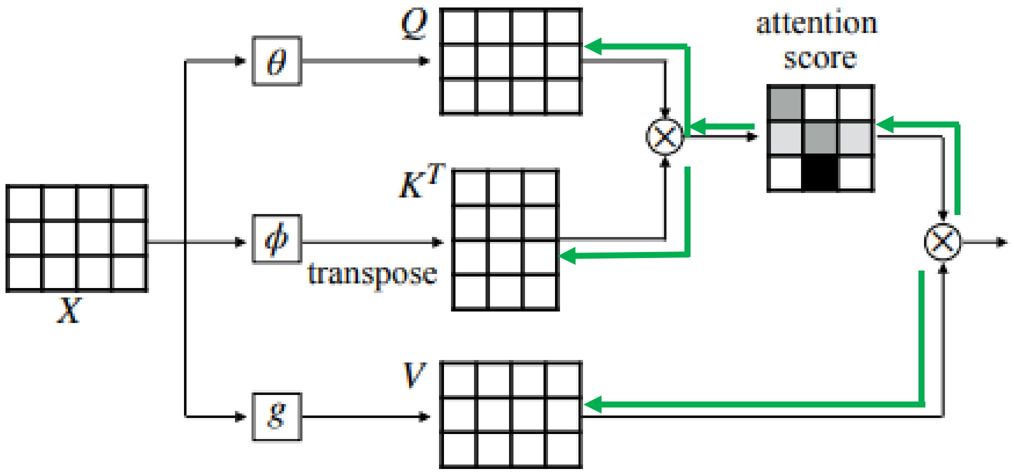
Base case for R

Layer-wise Relevance Propagation (LRP) (2)

- **Layer-wise Relevance Propagation for Transformer**
  - In Transformer, for some binary operations (e.g. $\mathrm{O} = AV$, or $A = softmax(QK^{\top})$), or skip connection $\mathrm{O} = \mathrm{X} + \mathrm{Y}$ where $Y = f(x)$), we need to propagate the relevance score through both input tensors.
  - For an operator with two tensors $O = f(XY)$, the relevance score is ensured to be fully distributed to both tensors

$$\sum_i R_i^O = \sum_j R_j^X + \sum_{\boxed{k}} R_k^Y$$

i, j, k will iterate over all elements in the output matrix O, X, Y



Layer-wise Relevance Propagation (LRP) (2)

- **Layer-wise Relevance Propagation for Transformer**
  - E.g., Let us compute LRP for $O = AV$. Consider an example

$$A = \begin{bmatrix}0.2 & 0.8\end{bmatrix}, \quad V = \begin{bmatrix}1 \\ 3\end{bmatrix}, \quad O = AV = \begin{bmatrix}2.6\end{bmatrix}, \quad R^O = \begin{bmatrix}1\end{bmatrix}$$

  - The contribution of each triplet (i, j, k) to the output position (i, k) is

$$m_{ijk} = \frac{A_{ij} V_{jk}}{\underbrace{\boxed{\sum_{j'} A_{ij'} V_{j'k}}}_{O_{ik}} + \varepsilon \, \text{sign}\left(\sum_{j'} A_{ij'} V_{j'k}\right)} R^O_{ik}$$

In case $O_{ik} \approx 0$

  - Split the contribution to input components

$$R^A_{ij} = \lambda \sum_k m_{ijk}, \quad R^V_{jk} = (1 - \lambda) \sum_i m_{ijk}$$

Typically, we can choose $\lambda = \frac{1}{2}$

Layer-wise Relevance Propagation (LRP) (2)

- **Layer-wise Relevance Propagation for Transformer**
  - E.g., Let us compute LRP for $O = AV$. Consider an example

$$A = \begin{bmatrix} 0.2 & 0.8 \end{bmatrix}, \quad V = \begin{bmatrix} 1 \\ 3 \end{bmatrix}, \quad O = AV = \begin{bmatrix} 2.6 \end{bmatrix}, \quad R^O = \begin{bmatrix} 1 \end{bmatrix}$$

  - The contribution of each triplet (i, j, k) to the output position (i, k) is (no need ε)

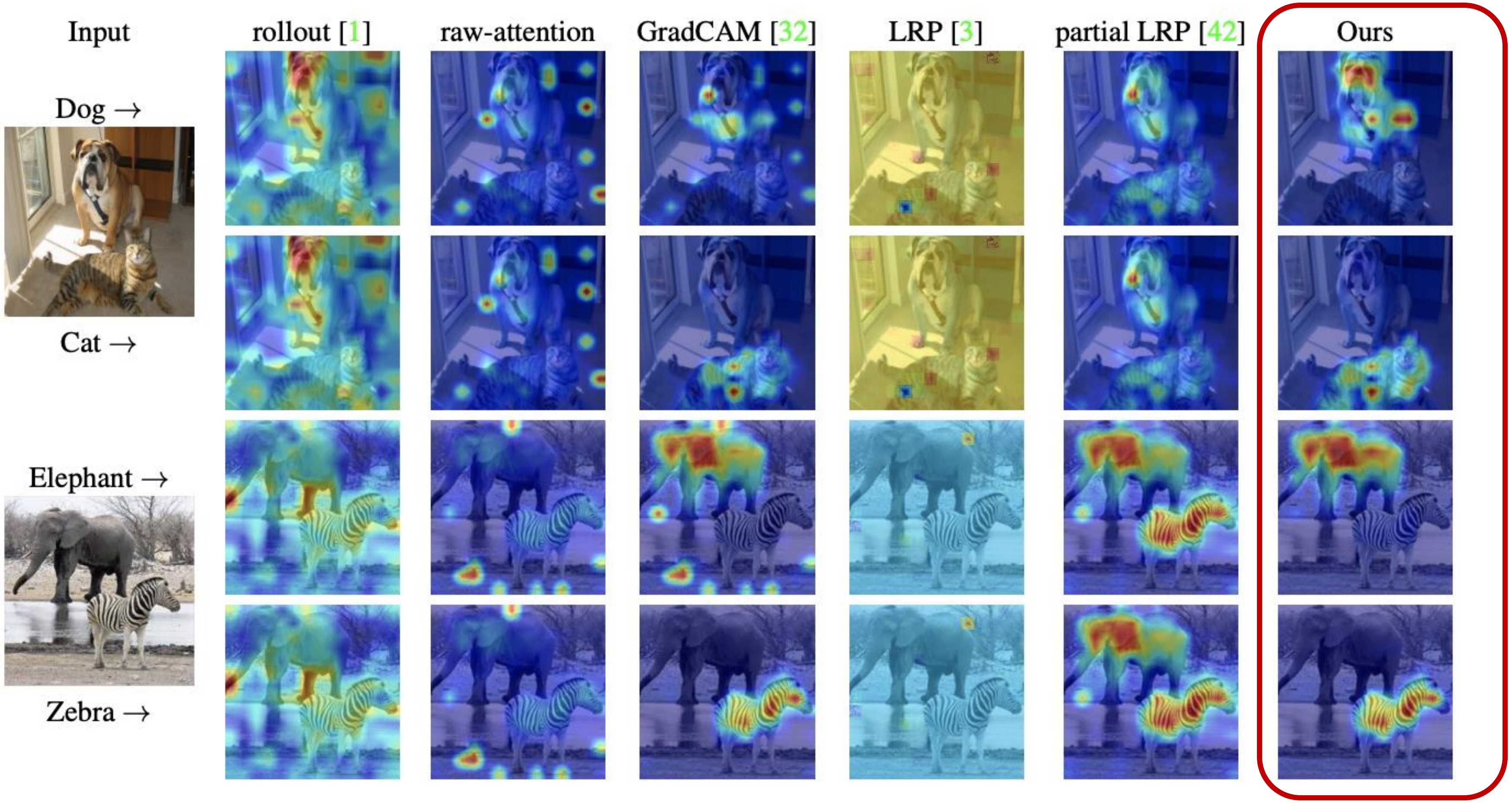
$$m_{111} = \frac{A_{11}V_{11}}{O_{11}} R^O_{11} = \frac{0.2 \cdot 1}{2.6} \cdot 1 = 0.076923 \qquad m_{121} = \frac{A_{12}V_{21}}{O_{11}} R^O_{11} = \frac{0.8 \cdot 3}{2.6} \cdot 1 = \frac{2.4}{2.6} = 0.923077$$

  - Split the contribution to input components (for $\lambda = 1/2$)

$$R^A = \begin{bmatrix} 0.03846 & 0.46154 \end{bmatrix}, \quad R^V = \begin{bmatrix} 0.03846 \\ 0.46154 \end{bmatrix}$$

  - Check: $\sum R^A + \sum R^V = 0.5 + 0.5 = 1 = \sum R^O$

Transformer Interpretability



Chefer et al., Transformer Interpretability Beyond Attention Visualization, CVPR2021

Attention Artifacts (1)

- **Artifacts in Vision Transformer**
  - Most of the existing transformer-based models exhibit artifacts on their attention matrix.

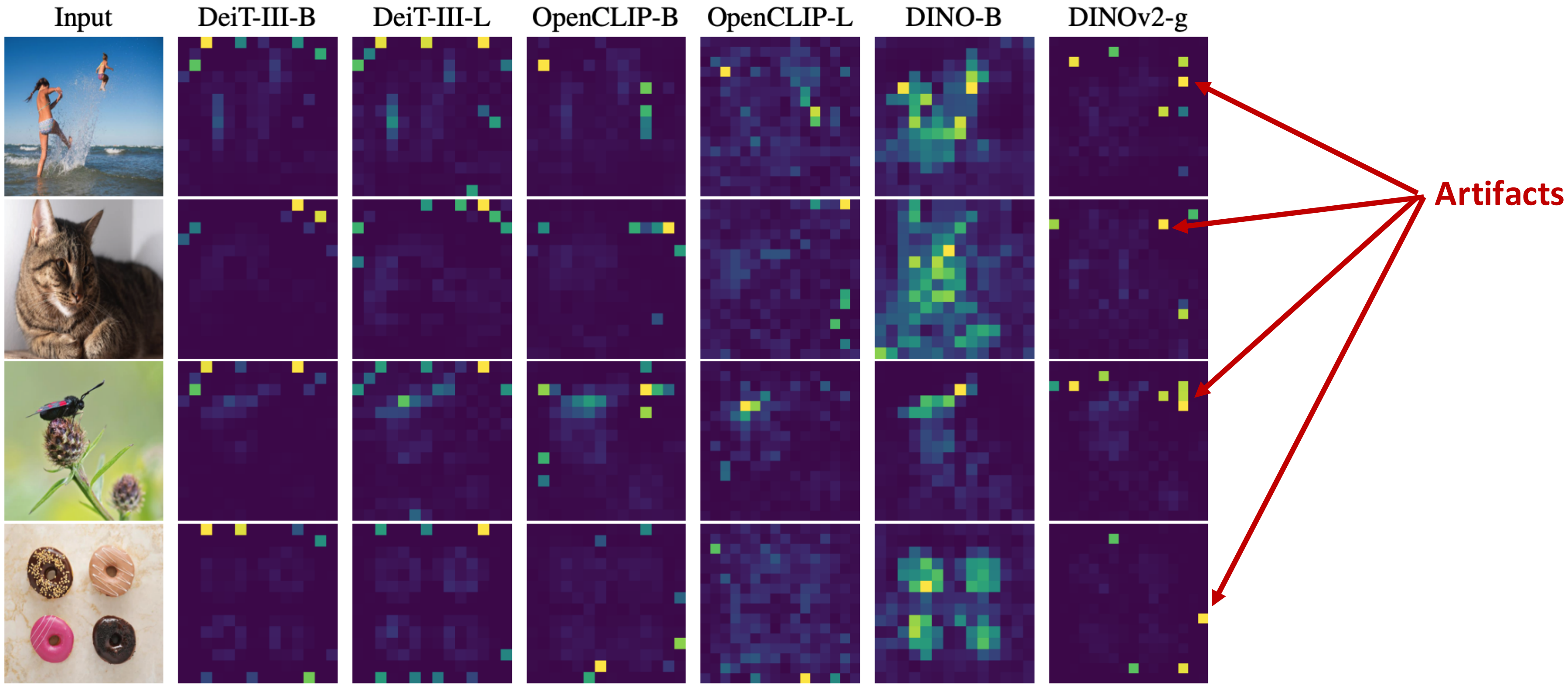
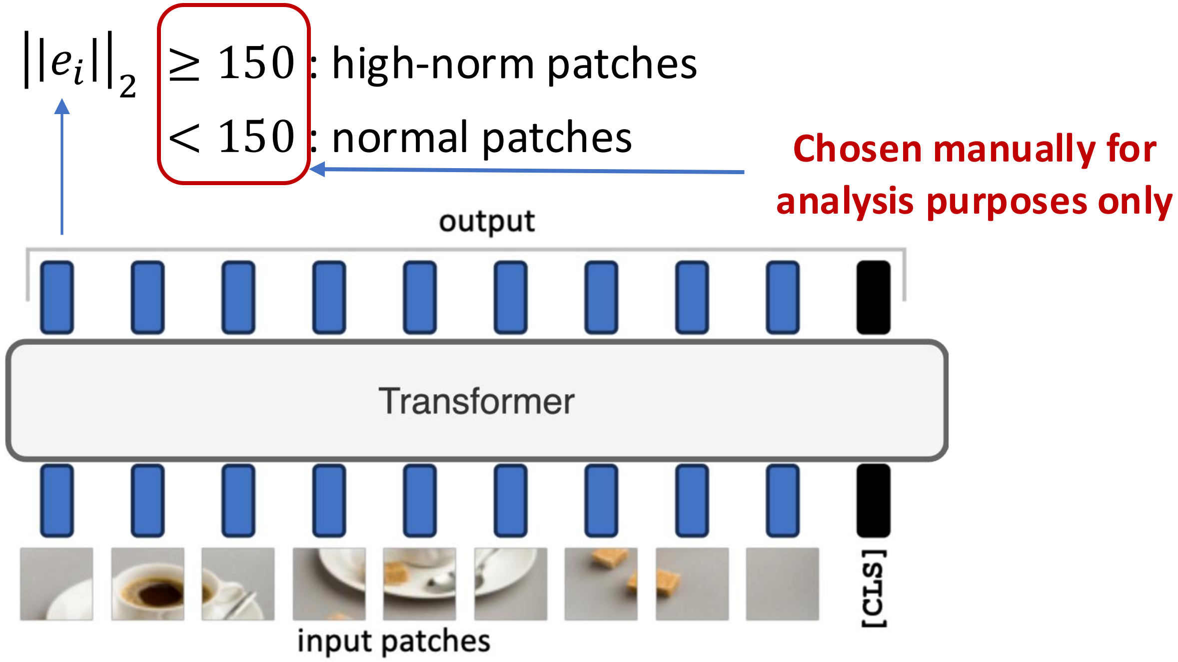


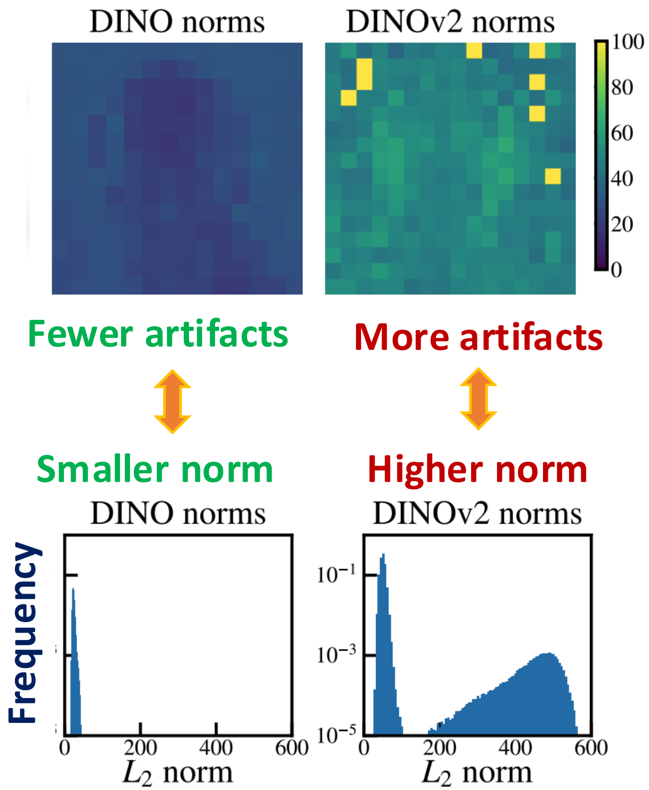
Figure 2: Illustration of artifacts observed in the attention maps of modern vision transformers.

Attention Artifacts (2)

- **Why does it happen?**
  - The artifacts have a connection to the norm of token embedding.

  → Let's analyze tokens with high-norm





Role of High-norm Patches

- **Settings**
  - For each patch/token embedding, add simple linear layers to
    - Predict the position of the patch
    - Reconstruct pixel values on the patch
    - Predict image class from the patch embedding
- **Observation**
  - Position prediction & reconstruction: **normal patches give better results.**
    → normal patches can maintain local information about patches
  - Image class classification:
    high-norm patches perform better
    → high-norm patches discard local information, having more global information (image class)

	position prediction		reconstruction
	top-1 acc	avg. distance ↓	L2 error ↓
normal	**41.7**	**0.79**	**18.38**
outlier	22.8	5.09	25.23

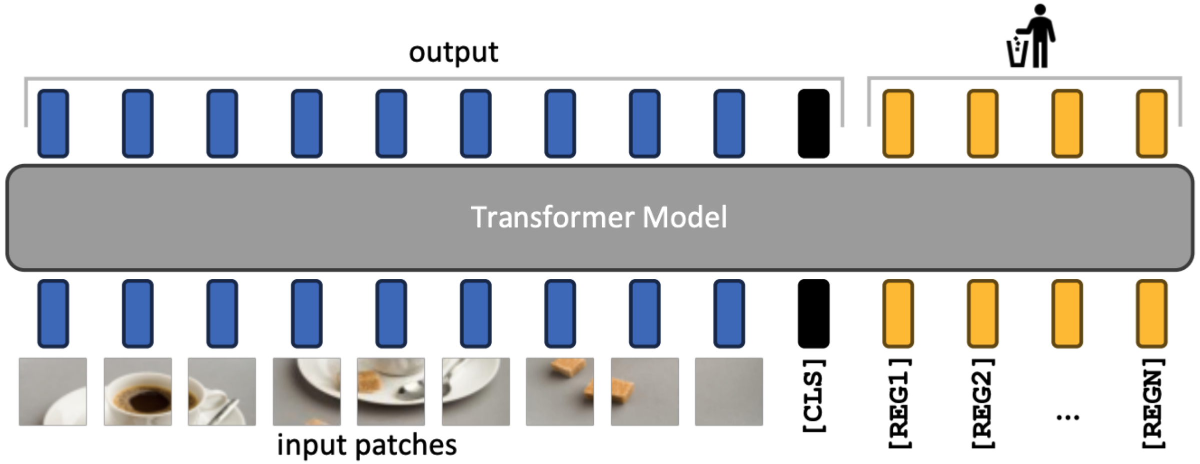
(b) Linear probing for local information.

	IN1k	P205	Airc.	CF10	CF100	CUB
[CLS]	**86.0**	**66.4**	**87.3**	**99.4**	**94.5**	**91.3**
normal	65.8	53.1	17.1	97.1	81.3	18.6
outlier	69.0	55.1	79.1	99.3	93.7	84.9

Image classification

Register Tokens

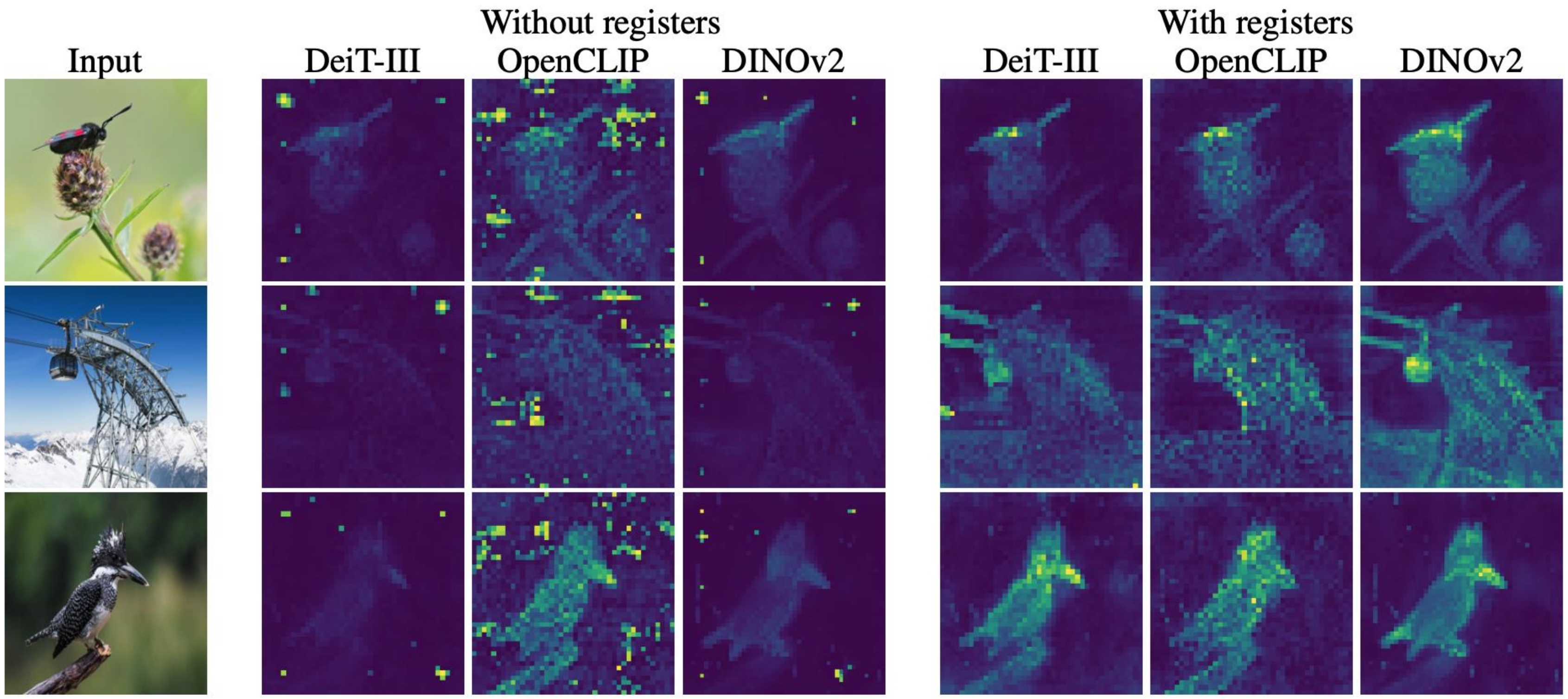
- **Hypothesis**
  - Large Transformer models can ***recognize redundant patches*** *(do not have much information)* and leverage them to ***store, process, and retrieve*** global information.
  - It may be undesirable as it may discard information from some patches.
- **Solution**
  - **Use some additional tokens as registers (along with [CLS]) for saving global information purposes**



In the paper, the authors use 16 registers

Register Improves Interpretability

- Adding registers provides much better interpretability and reduces artifacts for the attention matrix.



Register Improves Performance

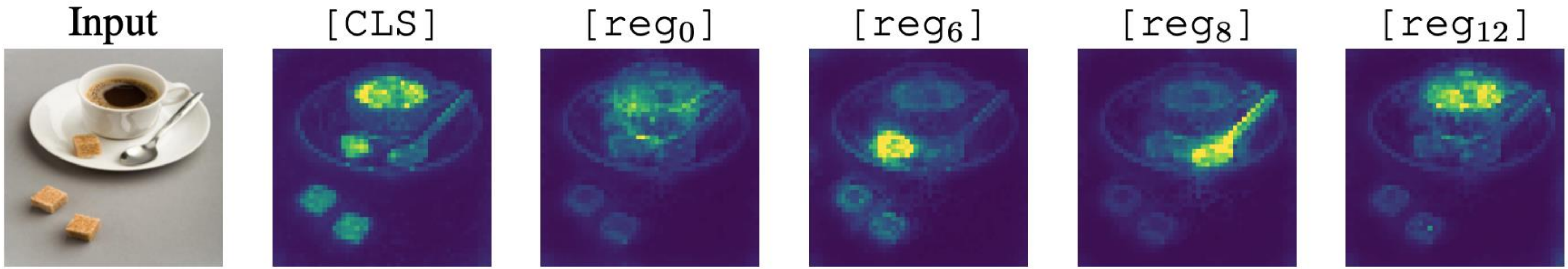
- Adding registers provides much better interpretability and reduces artifacts for the attention matrix.
- Performance slightly improved

	ImageNet Top-1	ADE20k mIoU	NYUd rmse ↓
DeiT-III	84.7	38.9	0.511
DeiT-III+reg	84.7	39.1	0.512
OpenCLIP	78.2	26.6	0.702
OpenCLIP+reg	78.1	26.7	0.661
DINOv2	84.3	46.6	0.378
DINOv2+reg	84.8	47.9	0.366

(a) Linear evaluation with frozen features.

Vision Transformers Need Registers

- Adding registers provides much better interpretability and reduces artifacts for the attention matrix.
- Performance slightly improved.
- Each register pays attention to different regions (naturally emerged from training).



Oral ICLR24 → A simple idea but good analyses/observations would also be appreciated