Transformer Interpretability Beyond Attention Visualization

Hila Chefer, Shir Gur, Lior Wolf

arXiv: https://arxiv.org/abs/2012.09838

Self-attention techniques, and specifically Transformers, are dominating the field of text processing and are becoming increasingly popular in computer vision classification tasks. In order to visualize the parts of the image that led to a certain classification, existing methods either rely on the obtained attention maps or employ heuristic propagation along the attention graph. This paper proposes a novel way to compute relevancy for Transformer networks, assigning local relevance based on the Deep Taylor Decomposition principle and then propagating these relevancy scores through the layers.

Relevance Propagation

Idea of the paper: Layerwise Relevance Propagation

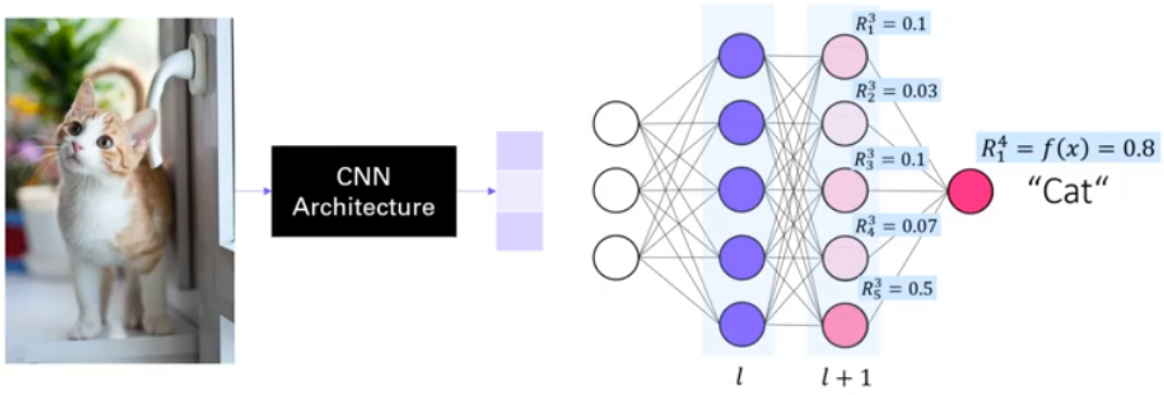
Given a prediction function $f(x)$ over an input sample $x \in \mathrm{R}^D$, compute a relevance score R_d for each input x at each dimension d

$$f(x) = \sum_{d=1}^{D} R_d(x)$$

- Constraints & objective:
  - We want a meaningful explanation for linear mappings: $f(x) = \sum_{d=1}^{D} w_d x_d$
  - Can be decomposed using the chain rule, f is a stack of functions $f(x) = g(h(x))$, in this case we can apply the decomposition to each function g and h separately, in this case the relevance can be redistributed top-down from outputs to neurons to inputs
  - relevance should be conversed when distributed from one layer to the next
  - a larger input get a larger share of the relevance
  - a bug input effects either a bit activation or a bin suppression (positive or negative relevance)
  - a zero input does no have any relevance

$$R_i^{(l)} = \sum_j \frac{z_{ij}}{\sum_{i'} z_{i'j}} R_j^{(l+1)} \quad \text{with} \quad \mathrm{z_{ij}} = \mathrm{x_i^{(1)}} \mathrm{w_{ij}^{(1,1+1)}}$$

→ So starting from the output, we choose a given class $t \in C$ and assign its neuron its output probability, then compute the relevance of the layer before it using the formula above.



Relevance Propagation for Transformer

Let C be the number of classes in the classification head, and $t \in 1...|C|$ the class to be visualized. We propagate relevance and gradients with respect to class t, which is not necessarily the predicted class. Following literature convention, we denote $x(n)$ as the input of layer $L(n)$, where $n \in [1...N]$ is the layer index in a network that consists of N layers, $x(N)$ is the input to the network, and $x(1)$ is the output of the network. We denote by $L^{(n)}(X, Y)$ the layer's operation on two tensors X and Y. Typically, the two tensors are the input feature map and weights for layer n. Relevance propagation follows the generic Deep Taylor Decomposition:

$$R_j^{(n)} = \sum_i \mathbf{X}j \frac{\partial Li^{(n)}(\mathbf{X},\mathbf{Y})}{\partial \mathbf{X}j} \frac{Ri^{(n-1)}}{L_i^{(n)}(\mathbf{X},\mathbf{Y})}$$

By considering a ReLU activation and $X = x$ and $Y = w$ as the layer's input and weights. The relevance propagation rule can be defined as follows;

$$R_j^{(n)} = \sum_i \frac{x_j^+ w_{ji}^+}{\sum_{j'} x_{j'}^+ w_{j'i}^+} R_i^{(n-1)}$$

However, to use such method for transformers, we need to adjust the following:

1- GELU activation

For transformer with GELU activation, the output have both positive and negative values, so the propagation above can only be computed for elements that have a positive weighted relevance, so let $q = \{(i,j) \mid x_j w_{ji} \geq 0\}$ be the subset of indices with positive relevance, then the propagation is done as follows:

$$R_j^{(n)} = \sum_{\{i|(i,j)\in q\}} \frac{x_j w_{ji}}{\sum_{\{j'|(j',i)\in q\}} x_{j'} w_{j'i}} R_i^{(n-1)}$$

2- Skip connections & attention modules

There are two operators in Transformer models that involve mixing of two feature map tensors (as opposed to a feature map with a learned tensor): skip connections and matrix multiplications (e.g. in attention modules). The two operators require the propagation of relevance through both input tensors. Note that the two tensors may be of different shapes in the case of matrix multiplication.

So given two tensors u and v, the idea is to compute two relevance score, each time we consider one tensor as an input and the other tensor as a weight using the same formula as above, resulting in $R_j^{u^{(n)}}$ with u as input and v as weight, and $R_k^{v^{(n)}}$ with v as input and u as weight.

However, we have two problems:

→ The conservation rule $\sum_j R_j^{u^{(n)}} + \sum_k R_k^{v^{(n)}} = \sum_i R_i^{(n-1)}$ is not preserved for attention modules where u and v are multiplied

→ For skip connection where u and v are added, we have numerical instability

To address the lack of conservation in the attention mechanism due to matrix multiplication, and the numerical issues of the skip connections, the method applied a normalization to $R_j^{u^{(n)}}$ and $R_k^{v^{(n)}}$

$$\bar{R}_j^{u^{(n)}} = R_j^{u^{(n)}} \frac{\left|\sum_j R_j^{u^{(n)}}\right|}{\left|\sum_j R_j^{u^{(n)}}\right| + \left|\sum_k R_k^{v^{(n)}}\right|} \cdot \frac{\sum_i R_i^{(n-1)}}{\sum_j R_j^{u^{(n)}}}$$

$$\bar{R}_k^{v^{(n)}} = R_k^{v^{(n)}} \frac{\left|\sum_k R_k^{v^{(n)}}\right|}{\left|\sum_j R_j^{u^{(n)}}\right| + \left|\sum_k R_k^{v^{(n)}}\right|} \cdot \frac{\sum_i R_i^{(n-1)}}{\sum_k R_k^{v^{(n)}}}$$

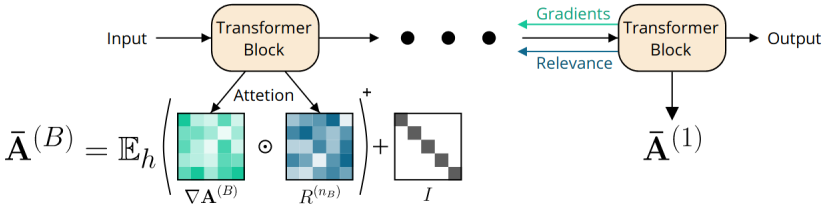
Relevance and gradient diffusion

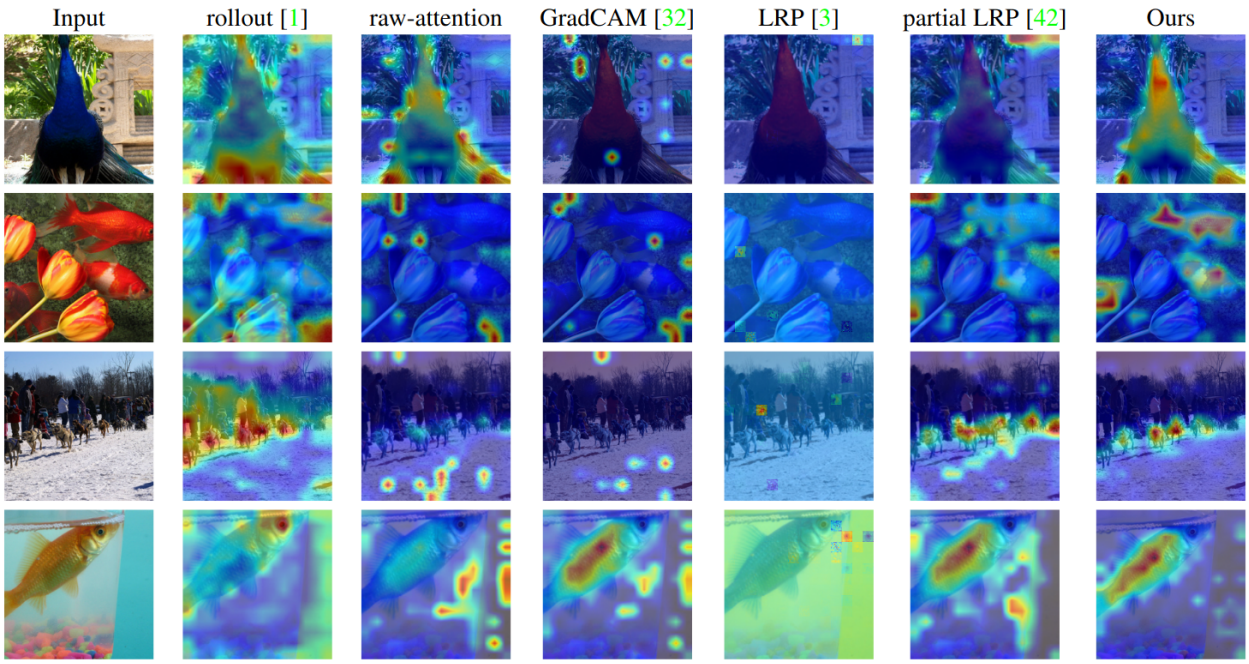
Let M be a Transformer model consisting of B blocks,where each block b is composed of self-attention, skip connections, and additional linear and normalization layers in a certain assembly. The model takes as an input a sequence of s tokens, each of dimension d, with a special token for classification, commonly identified as the token [CLS]. M outputs a classification probability vector y of length C, computed using the classification token. The self-attention module operates on a small sub-spaced d_h of the embedding dimension d, where h is the number of "heads", such that $h \times d_h = d$. The self-attention module is defined as follows:

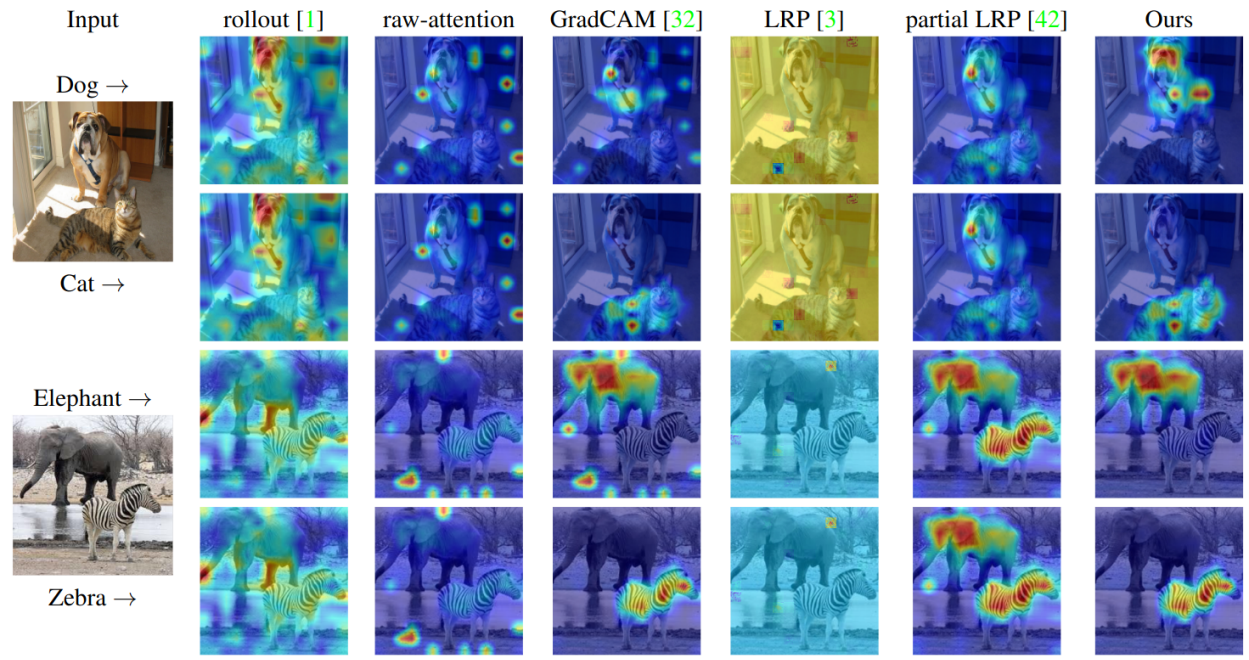
The final output C is given by a combination of both the gradients of the attention $\nabla \mathbf{A}$ and the Relevance R:

$$\overline{\mathbf{A}}^{(b)} = I + \mathbb{E}h\left(\nabla \mathbf{A}^{(b)} \odot R^{(nb)}\right)^{+}$$
$$\mathbf{C} = \overline{\mathbf{A}}^{(1)} \cdot \overline{\mathbf{A}}^{(2)} \cdot \ldots \cdot \overline{\mathbf{A}}^{(B)}$$

$$\mathbf{C} = \bar{\mathbf{A}}^{(1)} \cdot \bar{\mathbf{A}}^{(2)} \cdot \ldots \cdot \bar{\mathbf{A}}^{(B)}$$




Input
rollout [1]
raw-attention
GradCAM [32]
LRP [3]
partial LRP [42]
Ours


Input
rollout [1]
raw-attention
GradCAM [32]
LRP [3]
partial LRP [42]
Ours
Dog →
Cat →
Elephant →
Zebra →