Title: Class activation mapping

URL: https://en.wikipedia.org/wiki/Class_activation_mapping

PageID: 80384533

Categories: Category:Artificial intelligence, Category:Computer vision, Category:Image processing, Category:Neural network architectures

---

Class activation mapping methods are explainable AI (XAI) techniques used to visualize the regions of an input image that are the most relevant for a particular task, especially image classification , in convolutional neural networks (CNNs) . These methods generate heatmaps by weighting the feature maps from a convolutional layer according to their relevance to the target class. [1]

In the field of artificial intelligence , generically defined as "the effort to automate intellectual tasks normally performed by humans", [2] machine learning and deep learning were created. They both use statistical and computational methods to learn patterns from data, reducing the need for manually coded rules. [2] Machine learning models are trained on input data and the known respective answers, learning the underlying patterns or structures present in the data. Traditional Machine learning algorithms employ manually designed feature sets, posing a direct link between machine learning designers and employed features. [3] Deep learning is a subfield of machine learning, based on the concept of successive layers of representation, in which the data is progressively unfolded in different ways, to extract relevant and informative patterns in data analysis . Deep learning algorithms are defined as feature learning algorithms automatically learning hierarchical feature representations from raw data, extracting increasingly abstract features through multiple layers. [2] [3] CNNs are a specific architecture of deep learning models, designed to process spatially structured data, such as images, exploiting a series of convolution , non-linear activation and pooling operations to extract relevant features, contained in the so-called feature maps from input data. [3] CNNs have demonstrated to be highly effective in a variety of computer vision and image processing tasks. [1] CNNs (and deep learning models more broadly) are described as black boxes [4] due to their complex and non-transparent internal layers of representation. The need for clearer indications on its internal working and decision-making process gave birth to XAI techniques. [1] Among the proposed XAI techniques for computer vision tasks, Class activation mapping methods can show which pixels in an input image are important to the predicted logit for a class of interest, in a classification task. [1] Class activation mapping methods were originally developed for class-discriminative scenarios to visualize which parts of the input image influenced the classification decision. Namely, to visually highlight the regions of those feature maps which contribute most strongly to the prediction of a given class.

More advanced versions of these methods are not limited to image classification tasks, but have been extended also to several vision-related tasks, such as object detection, [5] image captioning, visual question answering and image segmentation. [1]

Background

The following methods laid the groundwork for the class activation maps approaches, forming the conceptual basis of using gradients to highlight class-discriminative regions.

Class model visualization and saliency maps for convolutional neural networks

The class model visualization and image-specific saliency maps approaches have been presented in the foundational work "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps" by Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman [6] and it generalizes the deconvnet method by Zeiler and Fergus. [7]

Class model visualization synthesizes an artificial input image that strongly activates the output neurons associated with a target class. Given a trained, fixed model, this method starts with a zero-initialized image, backpropagates the gradients from the class score to the image pixels,

updates the image pixels increasing the specific class scores and it repeats the pixel updating process, showing an encoded (idealized version) prototype of the class of interest. [6]

Image-specific class saliency visualization method provides a visual explanation by highlighting the most relevant pixels in an image for predicting a certain class C of interest. This is done by computing the gradient of the class score with respect to the input image, I 0 , {\displaystyle I_{0},} w = ∂ S C ∂ I | I 0 {\displaystyle w=\left.{\frac {\partial S_{C}}{\partial I}}\right|_{I_{0}}} approximating the model locally (around I 0 {\displaystyle I_{0}}) as linear, using a first-order Taylor expansion : S C (I) ≈ w C T I + b {\displaystyle S_{C}(I)\approx w_{C}^{T}I+b} . The magnitude of w C {\displaystyle w_{C}} , the gradient, indicates the importancy of the pixels: larger gradients suggest greater influence on the prediction. Once the gradient is known, the saliency map is defined as the maximum absolute gradient across the color channels: M i j = m a x C | ∂ S C ∂ I i j C | {\displaystyle M_{ij}=max_{C}\left|{\frac {\partial S_{C}}{\partial I_{ij}^{C}}}\right|} resulting in an saliency map (i.e. heatmap). [6] [8]

Guided backpropagation

The concept of guided backpropagation can be traced for the first time in the paper by Springenberg et al. "Striving For Simplicity: The All Convolutional Net" [9] and also this method builds upon the work by Zeiler and Fergus "Visualizing and Understanding Convolutional Networks". [7]

Guided backpropagation core is to understand what a CNN is learning, by visualizing the patterns that activate more strongly individual neurons (or filters), in architectures which do not rely on max-pooling layer.

When propagating gradients back through a rectified linear unit (ReLU) , guided backpropagation passes the gradient if and only if the input to the ReLU was positive (forward pass) and the output gradient is positive (backward signal), tackling both inactive neurons, negative gradients and suppressing the noise. The result displays sharper, high-resolution visualizations of what each neuron is responding to. [9]

Guided backpropagation represents a simple and practical method for model interpretability, helping understand how and where neural networks detect semantic concepts across layers. Moreover, it can be applied to any network architecture, due to its working principle. [10]

Base versions

Class activation mapping and gradient-weighted class activation mapping are the original and most widely used methods for visual explanations in convolutional neural networks. These methods serve as the foundation for many later developments in explainable AI. [11]

Notation: In this article, the symbols i and j represent integer indices that disappear inside sums or averages, while x and y are the continuous (or up-sampled integer) coordinates of the final heat-map that is plotted.

Class activation mapping (CAM)

Class activation mapping (CAM) was the first, and the original, version of CAM methods, and it gave the name to the whole category. The approach was firstly introduced by Zhou et al. in their seminal work "Learning Deep Features for Discriminative Localization". [12] This approach achieves class-specific heatmaps by modifying image classification CNN architectures, replacing fully-connected layers with convolutional layers and a final global average pooling layer. Its main scope is to localize and highlight discriminative regions of an input image that a CNN uses to identify a particular class, without needing explicit bounding box annotations. [11] [12] [13] [14]

Global average pooling (GAP)

Global average pooling (GAP) represents the key element in the original CAM approach. It is a dimensionality reduction technique and, similarly to other pooling layers, it allows the downsampling of the feature maps, calculating representative values for a specific region of the feature map. The particularity of GAP is that it calculates a single value for an entire feature map, significantly reducing the model dimensions. [11] [12] [13] [14]

Mathematical description

The mathematical description considers as its key the combination of convolutional and GAP layers. In CAM, it is mandatory to have the GAP layer after the last convolutional layer and before the final linear classifier layer. This last element of the architecture connects the output logits (the network predictions) y C {\displaystyle y^{C}} , to the GAP values, with its respective fine-tuned weights, w k C {\displaystyle w_{k}^{C}} .

Considering A k {\displaystyle A^{k}} as the last feature maps of the last convolutional layer, GAP produces one value for each feature map, by averaging all the matrix elements (i, j) of the feature map:

F k = 1 m n ∑ i = 1 m ∑ j = 1 n A i j k {\displaystyle F^{k}={\frac {1}{mn}}\sum _{i=1}^{m}\sum _{j=1}^{n}A_{ij}^{k}}

with A k = [A 11 k A 12 k ■ A 1 n k A 21 k A 22 k ■ A 2 n k ■ ■ ■ ■ A m 1 k A m 2 k ■ A m n k] = { A i j k ■ 1 ≤ i ≤ m , 1 ≤ j ≤ n } {\displaystyle A^{k}={\begin{bmatrix}A_{11}^{k}&A;_{12}^{k}&\cdots &A;_{1n}^{k}\\A_{21}^{k}&A;_{22}^{k}&\cdots &A;_{2n}^{k}\\\vdots &\vdots &\ddots &\vdots \\A_{m1}^{k}&A;_{m2}^{k}&\cdots &A;_{mn}^{k}\end{bmatrix}}=\left\{A_{ij}^{k}\mid 1\leq i\leq m,\ 1\leq j\leq n\right\}}

Namely, in the GAP layer, each feature map is reduced to a single scalar via GAP, producing k values, hence reducing the dimensionality of the network. A k × {\displaystyle \times } m × {\displaystyle \times } n tensor is reduced to k scalars, shrinking the parameter count for the linear classifier head.

The final output logits are calculated as the linear sum of the GAP values, weights and bias:

y C = ∑ k w k C F k {\displaystyle y^{C}=\sum _{k}w_{k}^{C}F^{k}}

The localization map is computed as follows:

L C A M C (x , y) = R e L U (∑ k w k C A k (x , y)) {\displaystyle L_{CAM}^{C}(x,y)=ReLU(\sum _{k}w_{k}^{C}A_{k}(x,y))}

namely, A k (x , y) {\displaystyle A_{k}(x,y)} is the activation of node k in the target layer of the model, and w k C {\displaystyle w_{k}^{C}} is the class-specific weight, for the channel k, in the linear classifier layer. [11] [12] [13] [14]

Advantages and drawbacks

The use of the GAP layer represents an example of an interpretability by design (IBD) approach. IBD refers to a technique which uses the model's own architecture to help explain its predictions.

The main drawback of CAM is that it is highly model-specific, being applicable to CNN architectures whose layer before the softmax one is a GAP.

Since the approach relies on the post-GAP weights for the overall evaluation, the method can't be applied to intermediate layers.

The choice of dealing with an IBD approach restricts the possibility to generalize the model architecture. Moreover, IBD methods often require re-training of the model. [11] [12] [13] [14]

Gradient-weighted class activation mapping (Grad-CAM)

Gradient-weighted class activation mapping (Grad-CAM) is a generalized version of CAM and it tackles its architectural limitations. Grad-CAM computes the gradient of a target class score, the pre-softmax logit, with respect to the feature maps of a convolutional neural network. The gradients are global-average-pooled to obtain importance weights, which are used to compute a class-specific localization map by linearly weighting the feature maps. The result is a heatmap that highlights the regions in the input image that are the most influential for predicting the target class.

The main advantage of Grad-CAM, with respect to the standard CAM, is that it is model agnostic (provided that the network still needs to be differentiable), meaning that it generates visual explanation for any CNN-based network without architectural changes or re-training, making it broadly applicable to pre-trained models. [11] [13] [15] [16]

Mathematical description

Considering:

y C {\displaystyle ^{C}} the logits (i.e. the pre-softmax activated neurons responsible for a certain class prediction) of interest;

A k {\displaystyle ^{k}} the feature activated map for a specific convolutional layer;

L Grad-CAM C {\displaystyle _{\text{Grad-CAM}}^{C}} ∈ R u × v {\displaystyle \mathbb {R} ^{u\times v}} the class-discriminative localization map, of width u and height v for any class c;

Grad-CAM, employing backpropagation, computes the logit gradient with respect to the feature map A k {\displaystyle ^{k}} as

∂ y C ∂ A k (i , j) {\displaystyle {\frac {\partial {y^{C}}}{\partial {A^{k}}(i,j)}}}

highlighting the importance of a certain class discrimination decision process of the logit. These gradients are global-average-pooled over each element of the feature map (hence, highlighting the "importance" of the elements of a feature map k for a target class C):

α k C = 1 u v ∑ i ∑ j ∂ y C ∂ A k (i , j) {\displaystyle \alpha _{k}^{C}={\frac {1}{uv}}\sum _{i}\sum _{j}{\frac {\partial {y^{C}}}{\partial {A^{k}}(i,j)}}}

So, to account for the total number of feature maps, each of them is multiplied by its weight (via dot-product) and element-wise summation is done:

∑ k α k C A k {\displaystyle \sum _{k}\alpha _{k}^{C}A^{k}}

It can be observed that, due to the intrinsic nature of the gradient operation, some elements of the weighted feature map will have negative value, so, since only elements that have increased the logit of the predicted class are of interest, a ReLU activation function is applied:

L G r a d – C A M C (x , y) = R e L U (∑ k α k C A k (x , y)) {\displaystyle L_{Grad-CAM}^{C}(x,y)=ReLU(\sum _{k}\alpha _{k}^{C}A^{k}(x,y))}

Lastly, the output heatmap image dimensions are upsampled to the original image size to match the input dimensions. [11] [13] [15] [16]

Advantages and drawbacks

Grad-CAM addresses the most important CAM limitations. It makes CAM free from the GAP layer need, generalizing its behavior and enabling visual explanation at intermediate layers.

However, Grad-CAM focuses on the most discriminative region when contributing to classification. If multiple similar objects are present, Grad-CAM often highlights only one of them, or part of one, providing also coarser maps and lower localization accuracy.

Moreover, Grad-CAM retrieves backwards information (the gradients), without taking into consideration how the activation flowed forward during prediction (unless combined with the guided backpropagation technique), resulting in a certain probability of missing patterns highlighted in the forward signal. On top of that, Grad-CAM heat maps are low-resolution when choosing a very deep layer.

Lastly, false emphasis in the heatmap may be present when large gradients are computed for low activation values. Grad-CAM assumes that gradient implies importance, ignoring the activation features value. [11] [13] [15] [16]

Grad-CAM and CAM comparison

Fine-tuned versions

Several methods have refined Grad-CAM to improve clarity and flexibility. Guided Grad-CAM, Grad-CAM++, Score-CAM, and Layer-CAM enhance aspects such as localization accuracy, gradient independence, and multi-layer visualization. These techniques build directly on the principles of CAM and Grad-CAM.

Guided Grad-CAM

Guided Grad-CAM fuses the coarse, class■discriminative localization of Grad-CAM with the high■resolution details of guided backpropagation. Grad-CAM heatmap is first computed for the target class, and it is upsampled to the input size. Then, a Guided Backpropagation saliency map for the same class is computed. A final element■wise product of the two results in the Guided Grad-CAM visualization map.

The result is a high-resolution, class-specific saliency map that highlights exactly which pixels contribute most to the network's decision. [15]

Grad-CAM++

Grad-CAM++ introduces a more refined way of computing the weights for each feature map, bypassing the global average of the gradients approach provided by Grad-CAM. This approach aims to improve the visual effect when multiple target instances are present in a single image.

Specifically, Grad-CAM++ employs pixel-wise gradients (via higher-order gradients), to compute the importance of a specific pixel for a prediction, lighting up multiple object instances in the same image. These improvements allow for a more sensible and detailed output heatmap.

The associated mathematical framework is defined by the following localization map:

L G r a d – C A M + + C (x , y) = ∑ k w k C A k (x , y) {\displaystyle L_{Grad-CAM++}^{C}(x,y)=\sum _{k}w_{k}^{C}A_{k}(x,y)}

in which the coefficient w k C {\displaystyle w_{k}^{C}} is defined as:

w k C = ∑ i , j α k C (i , j) × R e L U (∂ y C ∂ A k (i , j)) {\displaystyle w_{k}^{C}=\sum _{i,j}\alpha _{k}^{C}(i,j)\times ReLU({\frac {\partial y^{C}}{\partial A_{k}(i,j)}})}

with A k (x , y) {\displaystyle A^{k}(x,y)} , the activation of node k in the target layer of the model at position (x,y); y C {\displaystyle y^{C}} the logit score for class C, and α k C (i , j) {\displaystyle \alpha _{k}^{C}(i,j)} being:

α k C (i , j) = ∂ 2 y C ∂ (A k (i , j)) 2 2 × ∂ 2 y C ∂ (A k (i , j)) 2 + ∑ a , b A k (a , b) × ∂ 3 y C ∂ (A k (i , j)) 3 {\displaystyle \alpha _{k}^{C}(i,j)={\frac {\frac {\partial ^{2}y^{C}}{\partial (A_{k}(i,j))^{2}}}{2\times {\frac {\partial ^{2}y^{C}}{\partial (A_{k}(i,j))^{2}}}+\sum _{a,b}A_{k}(a,b)\times {\frac {\partial ^{3}y^{C}}{\partial (A_{k}(i,j))^{3}}}}}}

While addressing some Grad-CAM problems, Grad-CAM++ method still relies on gradients, and it only improves the underlying math. It is, however, still based on the idea of assigning a direct and valid relationship between gradient and importance. [13] [17]

Notation: (a,b) indexes all pixel positions in the feature■map, exactly like (i,j) does, but for the summation in the denominator.

Score-CAM

Score-CAM is a gradient-free CAM technique, thus redefining the original Grad-CAM and Grad-CAM++ working principles. It uses the model confidence scores instead of gradients.

Score-CAM performs the following operations:

Extracts the feature maps A k {\displaystyle A_{k}} of the final convolutional layers, as in the original Grad-CAM;

Upsamples each activation map A k {\displaystyle A_{k}} to the same input image dimensions, defining a mask M k {\displaystyle M_{k}} and each mask is normalized;

Multiplies the original input image by the mask, defining a masked image X k ′ = M k ■ X {\displaystyle X'_{k}=M_{k}\odot X} (■ {\displaystyle \odot } is the element-wise multiplication);

Gets a confidence score (a softmax probability is the output value after the softmax operation; the logit is the value before the softmax) for the masked images X k ′ {\displaystyle X'_{k}} , by feeding it into the CNN (either the soft-max probability or the raw logit can be used; both yield similar results in practice);

Considers that confidence score as the weight w k c {\displaystyle w_{k}^{c}} for the feature map A k {\displaystyle A_{k}} .

These operations allow to replace the gradient calculations with the actual model outputs, building more accurate heatmaps.

Mathematically, the localization map is defined as:

L S c o r e – C A M C (x , y) = R e L U (∑ k w k c A k (x , y)) {\displaystyle L_{Score-CAM}^{C}(x,y)=ReLU(\sum _{k}w_{k}^{c}A_{k}(x,y))}

and the coefficient w k C {\displaystyle w_{k}^{C}} s:

w k C = s o f t m a x k (y C (X k ′)) = e x p (y C (X k ′)) ∑ m e x p (y C (X m ′)) {\displaystyle w_{k}^{C}=softmax_{k}(y^{C}(X'_{k}))={\frac {exp(y^{C}(X'_{k}))}{\sum _{m}exp(y^{C}(X'_{m}))}}}

where A k (x , y) {\displaystyle A_{k}(x,y)} is the activation of channel k at location (x,y), y C (X) {\displaystyle y^{C}(X)} is the logit for class C for an input X and M k {\displaystyle M_{k}} is the mask, defined as:

M k (x , y) = U (A k) (x , y) – m i n x , y U (A m) m a x x , y U (A k) – m i n x , y U (A k) {\displaystyle M_{k}(x,y)={\frac {U(A_{k})(x,y)-min_{x,y}U(A_{m})}{max_{x,y}U(A_{k})-min_{x,y}U(A_{k})}}}

with U {\displaystyle U} as the upsampling operation.

Since the process of score calculation is repeated for every channel, Score-CAM is slow with respect to gradient-based methods. Moreover, it focuses on regions highlighted by individual feature maps, ignoring the context of the full image, reducing interpretability in complex scenes. [13] [18]

LayerCAM

LayerCAM enhances backwards class-specific gradients using both intermediate and final convolutional layers. Combining information across layers allows to achieve higher resolution and more fine-grained detail, improving localization.

Specifically, for each position in the feature map, LayerCAM evaluates the gradient. The positive gradients are employed as the weights, w k C {\displaystyle w_{k}^{C}} . Namely:

w k C (x , y) = R e L U (∂ y C ∂ A k (i , j) (x , y)) {\displaystyle w_{k}^{C}(x,y)=ReLU({\frac {\partial y^{C}}{\partial A_{k}(i,j)}}(x,y))}

The activations are then weighted, and the final class activation map is retrieved by summing over the channels.

L L a y e r – C A M C (x , y) = R e L U (∑ k w k C (x , y) A k (x , y)) {\displaystyle L_{Layer-CAM}^{C}(x,y)=ReLU(\sum _{k}w_{k}^{C}(x,y)A_{k}(x,y))}

This technique offers high-resolution heatmaps, flexible localization and per-location precision, employing positive gradients. [13] [19]

References

External links

"the World Conference on eXplainable Artificial Intelligence" .

"ACM Conference on Fairness, Accountability, and Transparency (FAccT)" .

"PyTorch image-specific saliency maps" . GitHub .

"Guided Backpropagation with PyTorch and TensorFlow" . 21 December 2021.

"PyTorch implementation of "Learning Deep Features for Discriminative Localization" " . GitHub .

"Advanced AI explainability for PyTorch" . GitHub .

v

t

e

History timeline

timeline

Companies

Projects

Parameter Hyperparameter

Hyperparameter

Loss functions

Regression Bias–variance tradeoff Double descent Overfitting

Bias–variance tradeoff

Double descent

Overfitting

Clustering

Gradient descent SGD Quasi-Newton method Conjugate gradient method

SGD

Quasi-Newton method

Conjugate gradient method

Backpropagation

Attention

Convolution

Normalization Batchnorm

Batchnorm

Activation Softmax Sigmoid Rectifier

Softmax

Sigmoid

Rectifier

Gating

Weight initialization

Regularization

Datasets Augmentation

Augmentation

Prompt engineering

Reinforcement learning Q-learning SARSA Imitation Policy gradient

Q-learning

SARSA

Imitation

Policy gradient

Diffusion

Latent diffusion model

Autoregression

Adversary

RAG

Uncanny valley

RLHF

Self-supervised learning

Reflection

Recursive self-improvement

Hallucination

Word embedding

Vibe coding

Machine learning In-context learning

In-context learning

Artificial neural network Deep learning

Deep learning

Language model Large language model NMT

Large language model

NMT

Reasoning language model

Model Context Protocol

Intelligent agent

Artificial human companion

Humanity's Last Exam

Artificial general intelligence (AGI)

AlexNet

WaveNet

Human image synthesis

HWR

OCR

Computer vision

Speech synthesis 15.ai ElevenLabs

15.ai

ElevenLabs

Speech recognition Whisper

Whisper

Facial recognition

AlphaFold

Text-to-image models Aurora DALL-E Firefly Flux Ideogram Imagen Midjourney Recraft Stable Diffusion

Aurora

DALL-E

Firefly

Flux

Ideogram

Imagen

Midjourney

Recraft

Stable Diffusion

Text-to-video models Dream Machine Runway Gen Hailuo AI Kling Sora Veo

Dream Machine

Runway Gen

Hailuo AI

Kling

Sora

Veo

Music generation Riffusion Suno AI Udio

Riffusion

Suno AI

Udio

Word2vec

Seq2seq

GloVe

BERT

T5

Llama

Chinchilla AI

PaLM

GPT 1 2 3 J ChatGPT 4 4o o1 o3 4.5 4.1 o4-mini 5

1

2

3

J

ChatGPT

4

4o

o1

o3

4.5

4.1

o4-mini

5

Claude

Gemini Gemini (language model) Gemma

Gemini (language model)

Gemma

Grok

LaMDA

BLOOM

DBRX

Project Debater

IBM Watson

IBM Watsonx

Granite

PanGu-Σ

DeepSeek

Qwen

AlphaGo

AlphaZero

OpenAI Five

Self-driving car

MuZero

Action selection AutoGPT

AutoGPT

Robot control

Alan Turing

Warren Sturgis McCulloch

Walter Pitts

John von Neumann

Claude Shannon

Shun'ichi Amari

Kunihiko Fukushima

Takeo Kanade

Marvin Minsky
John McCarthy
Nathaniel Rochester
Allen Newell
Cliff Shaw
Herbert A. Simon
Oliver Selfridge
Frank Rosenblatt
Bernard Widrow
Joseph Weizenbaum
Seymour Papert
Seppo Linnainmaa
Paul Werbos
Geoffrey Hinton
John Hopfield
Jürgen Schmidhuber
Yann LeCun
Yoshua Bengio
Lotfi A. Zadeh
Stephen Grossberg
Alex Graves
James Goodnight
Andrew Ng
Fei-Fei Li
Alex Krizhevsky
Ilya Sutskever
Oriol Vinyals
Quoc V. Le
Ian Goodfellow
Demis Hassabis
David Silver
Andrej Karpathy
Ashish Vaswani
Noam Shazeer
Aidan Gomez
John Schulman
Mustafa Suleyman
Jan Leike

Daniel Kokotajlo

François Chollet

Neural Turing machine

Differentiable neural computer

Transformer Vision transformer (ViT)

Vision transformer (ViT)

Recurrent neural network (RNN)

Long short-term memory (LSTM)

Gated recurrent unit (GRU)

Echo state network

Multilayer perceptron (MLP)

Convolutional neural network (CNN)

Residual neural network (RNN)

Highway network

Mamba

Autoencoder

Variational autoencoder (VAE)

Generative adversarial network (GAN)

Graph neural network (GNN)

Category