Explainable Machine Learning Models

Review and Computational Analysis

Šimon Růžička, Utku Ipek

School of Computation, Information and Technology
Technical University of Munich

January 24th, 2024

Outline

Motivation

- Black-box ML models are used more and more
- Even today, we encounter the results of neural networks in our everyday life
  - Predictive AI – weather predictions, spam classification, self-driving cars, . . .
  - Generative AI – generating text (ChatGPT) or images (DALL·E, Midjourney)
- In many cases, black-box predictions are not reliable enough

Motivation

- Black-box ML models are used more and more
- Even today, we encounter the results of neural networks in our everyday life
  - Predictive AI – weather predictions, spam classification, self-driving cars, . . .
  - Generative AI – generating text (ChatGPT) or images (DALL·E, Midjourney)
- In many cases, black-box predictions are not reliable enough

Need for explanation:

- High-risk scenarios, such as law or medicine
- Need to convince humans currently making the predictions
- Working with sensitive user data (users deserve to know the explanation)

Prediction accuracy and explainability

Ideally, we want a model with the same prediction accuracy but transparent. However, ...

Definition – Accuracy-explainability trade-off

Models with high predicting accuracy on complex tasks are usually entangled and hard to explain. On the other hand, simpler transparent models may struggle with predicting accuracy, meaning a compromise between these aspects must be made.

Priorities need to be set when designing a model!

Interpretability, completeness

Explainability is commonly misinterpreted, partly due to disorganized terminology.

Explained decision needs to be both:

- Interpretable – understandable by humans
- Complete – describe the underlying model accurately

It is difficult to achieve both aspects simultaneously, especially for complex models.

Interpretability, completeness

Explainability is commonly misinterpreted, partly due to disorganized terminology.

Explained decision needs to be both:

- Interpretable – understandable by humans
- Complete – describe the underlying model accurately

It is difficult to achieve both aspects simultaneously, especially for complex models.

Watch out! – Incomplete explanations

Interpretable but incomplete explanations are very hard/impossible to recognize manually. We commonly need further knowledge about the model processing to compare that. This is a big problem for novel approaches such as using LLMs to explain neural networks.

Interpretability, completeness – example

We want to estimate an apartment price, given the following data:

Square meters	Distance from center	Distance from U-Bahn	Has garden	Price
105 m^2	2.3 km	0.3 km	False (=0)	???

We will be using a multiple-layer neural network for this task. How can we obtain:

1. Interpretable explanation?
2. Complete explanation?
3. Both interpretable and complete explanation?

Interpretability, completeness – example 1.

Square meters	Distance from center	Distance from U-Bahn	Garden	Price
x_1	x_2	x_3	x_4	y
105 m^2	2.3 km	0.3 km	0	582 463 €

Interpretability, completeness – example 1.

Square meters	Distance from center	Distance from U-Bahn	Garden	Price
x_1	x_2	x_3	x_4	y
105 m^2	2.3 km	0.3 km	0	582 463 €

An interpretable explanation might look like this:

The apartment is predicted to cost 582 463 €, mostly because of its proximity to U-Bahn. The size of the apartment is not relevant to the price estimate.

Is this explanation complete? How can we check?

Interpretability, completeness – example 2.

Square meters	Distance from center	Distance from U-Bahn	Garden	Price
x_1	x_2	x_3	x_4	y
105 m^2	2.3 km	0.3 km	0	582 463 €

A complete explanation might look like this:

$$\begin{aligned}\hat{y} = 3\,100 &\cdot \text{ReLU}(0.15x_1 + 2.3x_2 - 4.7x_3 + 1.6x_4)\\ + 11\,000 &\cdot \text{ReLU}(0.42x_1 - 1.6x_2 + 3.8x_3 + 9.4x_4)\\ + 5\,000 &\cdot \text{ReLU}(0.05x_1 + 3.1x_2 + 1.7x_3 - 5.1x_4)\end{aligned}$$

However, this explanation is not very interpretable.

Interpretability, completeness – example 3.

Square meters	Distance from center	Distance from U-Bahn	Garden	Price
x_1	x_2	x_3	x_4	y
105 m^2	2.3 km	0.3 km	0	582 463 €

Are we able to make an explanation that is both interpretable and complete?

Interpretability, completeness – example 3.

Square meters	Distance from center	Distance from U-Bahn	Garden	Price
x_1	x_2	x_3	x_4	y
105 m^2	2.3 km	0.3 km	0	582 463 €

Are we able to make an explanation that is both interpretable and complete?

Yes, but the techniques used need to be more complex. And we still need to make some form of a trade-off between these features.

Outline

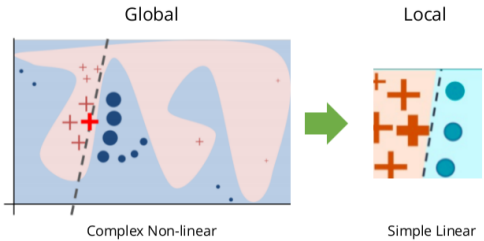
Overview

Methods presented here are:

- Model-agnostic.
- Able to handle various data types.
- Serve a dual purpose in scope: Local and Global

Examples: LIME, Learning Disentangled Representations, SHAP, Anchors, etc.

Local Interpretable Model-agnostic Explanations (LIME)



Source:https://c3.ai/

LIME – Characteristics

How do we generate an explanation-producing method?

[1] Ribeiro *et al.*, 2016

LIME – Characteristics

How do we generate an explanation-producing method?

First, three desired characteristics of explanation methods[1] should be examined:

- Being interpretable: It is subject to the audience.
- Being model-agnostic: The key idea is flexibility.
- Having local fidelity: It should function as expected, at least around a single instance.

[1] Ribeiro *et al.*, 2016

LIME – Producing An Explanation

To produce an explanation, the below components are needed:

- The model under explanation, denoted by f.
- A model $g \in G$ where G constitutes the group of models with potential interpretability.
- A proximity measure $\pi_x(z) = \exp\left(-\frac{D(x,z)^2}{\sigma^2}\right)$ where D is some distance function, $x \in \mathbb{R}^d$ is the original representation of an instance, and $z \in \mathbb{Z}$ is the perturbed instance.

LIME – The Produced Explanation

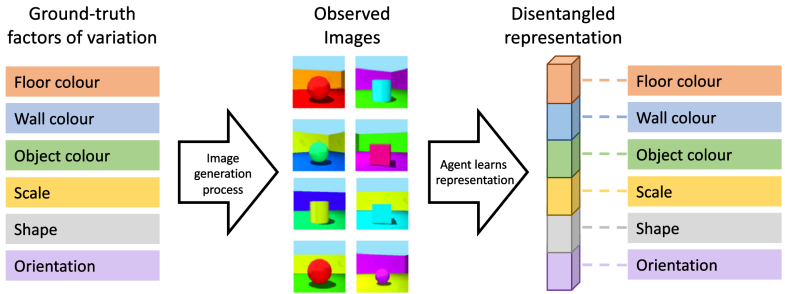
Using the components defined, the explanation is produced via:

$$\xi(x) = \underset{g \in G}{\operatorname{argmin}} \mathcal{L}(f, g, \pi_x) + \Omega(g)$$

Where

- $\mathcal{L}(f, g, \pi_x) = \sum_{z,z' \in \mathbb{Z}} \pi_x(z)(f(z) - g(z'))^2$ is the locality-aware loss function.
- $\Omega(g)$ is the complexity of the corresponding interpretable model.

Disentangled Representations



Source:Autonomous Agents Research Group Blog

Disentangled Representations – Motivation

How important is the way we represent the data for the performance of our AI algorithm?

Feature engineering is crucial to developing and running AI algorithms.

Should we rely on high-quality feature engineering every time?

Disentangled Representations – Motivation

How important is the way we represent the data for the performance of our AI algorithm?

Feature engineering is crucial to developing and running AI algorithms.

Should we rely on high-quality feature engineering every time?

- The load on feature engineering can be lightened.
- A model can learn the data representations.

Disentangled Representations – Formal Definition

An approach[2] exists to formally define them that uses natural transformations from the real world, and the group theory.

Definition – Disentangled vector space

Given the group G, the world state W, and the vector space Z, Z is said to be disentangled relating to the decomposition $G = G_1 \times \ldots \times G_n$ if an action $\cdot : G \times Z \to Z$ exists, a mapping $f : W \to Z$ exhibits equivariance between the transformations applied to W and Z, and there exists a decomposition $Z = Z_1 \times \ldots \times Z_n$ where each Z_i solely influenced by G_i, and remain fixed otherwise.

[2] Higgins *et al.*, 2018

Disentangled Representations – A Quantitative Evaluation Framework

An approach[3] to assessing the learned representations.

We need ground-truth latent spaces to quantitatively evaluate them.

The courses of action:

- Train model M on a synthetic dataset featuring generative factors $\boldsymbol{z}$.
- Extract a disentangled code $\boldsymbol{z}$ for every data point x in the dataset ($\boldsymbol{c} = M(\boldsymbol{x})$).
- Train a regressor to estimate z ($\hat{\boldsymbol{z}} = f(\boldsymbol{c})$).
- Assess the regressor's divergence by comparing it with the ideal mapping and the error in prediction.

[3] Eastwood *et al.*, 2018

Disentangled Representations – A Quantitative Evaluation Framework

(a) **Lasso**				(b) **Random forest**			
Code	**Disent.**	**Compl.**	**Inform.**	**Code**	**Disent.**	**Compl.**	**Inform.**
PCA	0.29	0.32	0.44	PCA	0.50	0.52	0.27
VAE ($\beta = 1$)	0.67	0.62	0.37	VAE ($\beta = 1$)	0.86	0.75	0.09
β-VAE ($\beta = 6$)	0.66	0.59	0.35	β-VAE ($\beta = 6$)	0.90	0.76	0.10
InfoGAN	0.75	0.72	0.23	InfoGAN	0.91	0.87	0.13

Source: Eastwood *et al.*, 2018

Outline

Motivation

When do we care about transparent models?

- When the explanation quality is of highest importance.
- Unexplained prediction is considered worthless.
- We can afford to sacrifice some predicting accuracy for that.

Common use of these models – **statistics**.

Opposite of transparent models – **black-box** models.

Linear model

The simplest (and probably oldest[4]) transparent model.

Prediction using a linear model

For input variables $\mathbf{x} = (x_1, \ldots, x_n); \mathbf{x} \in \mathbb{R}^n$ and learn weights $\mathbf{w} = (w_1, \ldots, w_n), w_0$, prediction of the output variable $\hat{y}$ is computed as

$$\hat{y} = \mathbf{x}^T \mathbf{w} + w_0$$

Learnt weights directly indicate us which inputs are the most relevant! But for this purpose, they are commonly divided by the variance of given input variable.

[4] Dates back to Legendre (1805) and Gauss (1809).

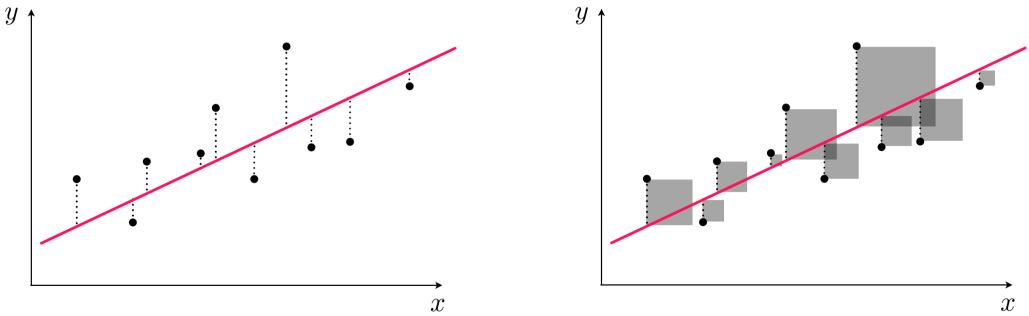
Optimal Least Squares

How do we find optimal weights for the best linear model? We need m input samples, stored in matrix $\mathbf{X} \in \mathbb{R}^{m,n}$ and vector $\mathbf{y} \in \mathbb{R}^m$.

1. Define loss function $L(\mathbf{w})$ as the sum of squares of errors.
2. Calculate gradient of this function and use it for minimization.
3. Gradient of a quadratic function can be found analytically $\rightarrow$ exact solution!

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

Weights are found without any iterative process (unlike more advanced models).



L2 Loss function visualization. Source: ML Refined blog

Generalized Linear Model (GLM)

We want a more advanced model, but one that is still interpretable.

- We still use the linear model, but transform it nonlinearly afterwards.
- We denote $\eta = \mathbf{x}^T\mathbf{w} + w_0$.
- However, we simulate the output with a random variable Y.
- Mean of Y denoted as $\mu = g^{-1}(\eta)$.
- Variance of Y, if needed, as var $Y = V(\mu)$.

Link function g (and inverse mean function g^{-1}) with the variance function V are connected to the distribution used to model the output.

GLM – distribution overview

Brief overview of distributions and their functions. Source: Wikipedia

Distribution	Support	Link name	$g(\mu)$	$g^{-1}(\eta)$
Normal	$(-\infty, \infty)$	Identity	μ	η
Exponential	$(0, \infty)$	Negative inverse	$-\mu^{-1}$	$-\eta^{-1}$
Poisson	$0, 1, 2, \ldots$	Logarithm	$\ln(\mu)$	e^{η}
Bernoulli	$[0, 1]$	Logit	$\ln\left(\frac{\mu}{1-\mu}\right)$	$\frac{1}{1+e^{-\eta}}$

Special cases: Normal (Linear regression/model) and Bernoulli (Logistic regression).

GLM – steps to use

1. Use your domain knowledge to select adequate distribution for Y.
2. Find the link and variance functions for that distribution.
3. Run the iterative training process – Iterative Weighted Least Squares.
4. Use your trained model for prediction.

Interpretation of weights depends on the distribution! For example in Bernoulli (modelling probability), increasing x_i by 1 means that predicted **odds** $\frac{p}{1-p} \in [0, \infty)$ are multiplied by e^{w_i}.

Generalized Linear Rule Models (GLRMs)

Can be thought of as the extension of GLMs.

Instead of using the linear combination of raw features like GLMs, use a set of rules derived from the data.

Two perks of GLRMs with the rules formulated:

- Being naturally interpretable.
- Able to represent complex, non-linear associations.

GLRMs – Rule Based Modeling

What is a rule?

A logical statement that is derived from the features of the dataset.

GLRMs – Rule Based Modeling

What is a rule?

A logical statement that is derived from the features of the dataset.

Defining a Rule Space

Let $\mathcal{K}$ represent the group of conjunctions associated with features $\mathbf{X} = (X_1, \ldots, X_d)$. Note that each feature in $\mathbf{X}$ should be converted to binary form. For $k \in \mathcal{K}$, A_k takes binary values for every instance i in the dataset ($a_{ik} \in \{0, 1\}$ for sample i), indicating if a data point adheres to the conditions set by the corresponding rule.

GLRMs – Fundamentals

Formulations to build GLRMs:

- Canonical parameter $\eta = \sum_{k \in \mathcal{K}} \beta_k A_k$, where β_k denotes the learned parameters.
- Using $\Phi(\eta)$, the probability distribution function can be defined as $p_{Y|\mathbf{X}}(y|\mathbf{x}) = h(y) \exp(\eta y - \Phi(\eta))$, where $y \in \mathcal{Y}$ denotes the target variable.

Optimization Problem

$\min_\beta \frac{1}{n} \sum_{i=1}^{n} \left[\Phi\left(\sum_{k \in \mathcal{K}} \beta_k a_{ik}\right) - y_i \sum_{k \in \mathcal{K}} \beta_k a_{ik}\right] + \sum_{k \in \mathcal{K}} \lambda_k |\beta_k|$

We seek to minimize the negative log-likelihood to conclude the coefficients β_k.

GLRMs – Generating the Rule Space

Building rule-based models is difficult because of the exponentially large number of conjunctions generating the rule space.

Exploring all possible rules for datasets with many features is practically infeasible. But, there are two approaches[5]:

- First Degree Rules: Single feature per rule.
- Column Generation: Based on solving the reduced version of the problem iteratively. Starts with a subset of rules.

[5] Wei *et al.*, 2019

Generalized Additive Models with Pairwise Interactions

Builds upon the foundation of generalized additive models (GAMs).

Designed to grasp not only the singular impacts of features in the data but also their combined interactions in a predictive setting.

Each feature's impact on the prediction can be distinctly determined. That makes GA^{2}Ms inherently interpretable.

Generalized Additive Models with Pairwise Interactions

Before diving into GA^{2}Ms, GAMs should be introduced briefly:

- The equation $g(E[y]) = \sum_i f_i(x_i)$ describes GAMs, g being the link function, y being the outcome, x_i being the i-th feature and f_i being the i-th smooth function.
- Cannot capture the complex interactions in the dataset.

Generalized Additive Models with Pairwise Interactions

Before diving into GA^{2}Ms, GAMs should be introduced briefly:

- The equation $g(E[y]) = \sum_i f_i(x_i)$ describes GAMs, g being the link function, y being the outcome, x_i being the i-th feature and f_i being the i-th smooth function.
- Cannot capture the complex interactions in the dataset.

The Equation Introducing Pairwise Interactions

$g(E[y]) = \sum_i f_i(x_i) + \sum_{i,j} f_{ij}(x_i, x_j)$

Now, we have an extensive array of feature combinations to examine.

GA2M – Solving The Main Challenge

Notation:

- $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ to describe the problem's dataset, where $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,n})$ stands for the input features, y_i for the target variable, n for the number of features, and N for the number of samples in the dataset.
- To represent the indices for all features in the dataset, they used $\mathcal{U} = \mathcal{U}^1 \cup \mathcal{U}^2$, where $\mathcal{U}^1 = \{\{i\} | 1 \leq i \leq n\}$ and $\mathcal{U}^2 = \{\{i, j\} | 1 \leq i < j \leq n\}$

The objective function $\min_{F \in \mathcal{H}} E[L(y, F(\mathbf{x}))]$:

- By employing the Hilbert space $\mathcal{H} = \sum_{u \in \mathcal{U}} \mathcal{H}_u$ and its additive group of functions $F(\mathbf{x}) = \sum_{u \in \mathcal{U}} f_u(x_u)$

GA2M – Solving The Main Challenge

The iterative algorithm[6]:

- Begins with two interaction sets and the best model from the Hilbert space.
- Creates a model for each feature pair that is not chosen.
- Selects the best pair and adds it to the set containing chosen pairs.
- Iterates until an improvement in accuracy is no longer observed.

FAST Approach

This approach significantly reduced the computational complexity of the prediction process from $O(n^2)$ to $O(n)$.

[6] Lou *et al.*, 2013

Outline

Specifics of image classification

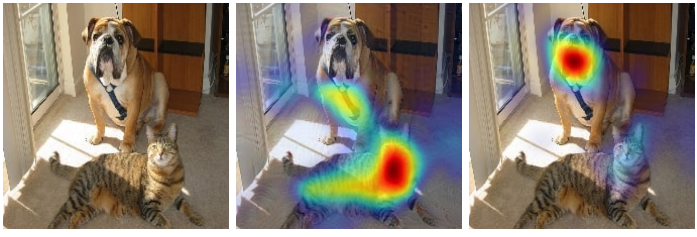
- Very high input dimension $\Rightarrow$ complex models (NNs)
- Possibility to provide a visual explanation

These factors mean that other explainability techniques are not ideal. Alternative approach – focus mapping.

Definition – Focus map

A focus map is a matrix of the same size as the input image $I \in [0, 255]^{m,n}$. Values in this map represent the importance of the given image pixel for the made prediction. Changes in highlighted regions would **positively** affect the prediction of an explained model the most.

Focus map – example



(a) Original Image **(b)** 'Cat' Focus map **(c)** 'Dog' Focus map

Created using the Grad-CAM[7] algorithm mentioned further, images from the same source.

[7] Selvaraju *et al.*, 2019

Gradient approach (1)

"Changes in highlighted regions would affect the prediction of an explained model the most." This can be easily determined by computing the gradient[8].

From input image I, we compute:

- Forward pass... class prediction $c = f(I)$
- Backward training pass... weight gradients $\frac{\partial f}{\partial \mathbf{w}}(I) = \nabla_{\mathbf{w}} f(I)$
- Backward explaining pass... input gradient $\frac{\partial f}{\partial I}(I) = \nabla_I f(I)$

[8] Simonyan *et al.*, 2014

Gradient approach (2)

Interpretation of input gradient $\frac{\partial f}{\partial I}(I) = \nabla f(I)$:

- It can be used for locally linear prediction around input I
- For any I' close to I, we have $f(I') \approx \nabla f(I) \cdot I' + f(I)$, where $\cdot$ performs dot product on flattened images
- Greater absolute value in $\nabla f(I) \Rightarrow$ greater change in prediction $\Rightarrow$ important region

Signs in gradient are irrelevant for interpretation since we care about any change in the image region (not whether it gets lighter/darker).

Problems of gradient approach

This method is very simple to describe, but it comes with multiple drawbacks:

- Computationally expensive, especially for deep networks
- Negatively affecting regions

What if there are regions that affect the prediction negatively?

Problems of gradient approach

This method is very simple to describe, but it comes with multiple drawbacks:

- Computationally expensive, especially for deep networks
- Negatively affecting regions

What if there are regions that affect the prediction negatively? See the previous example with cat/dog:

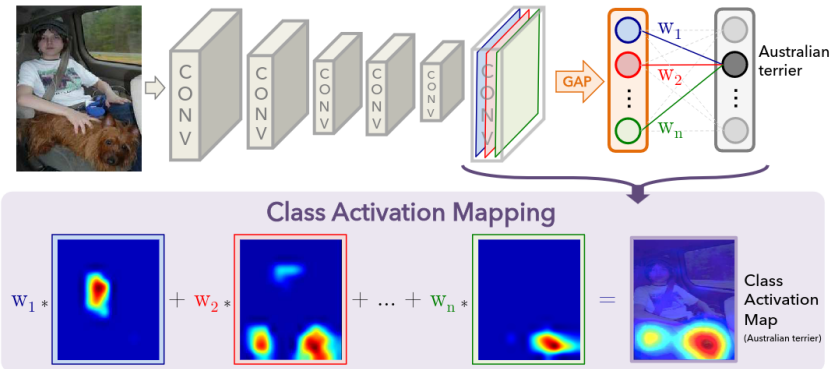
- Assume that a classifier has predicted $c =$ dog.
- When explaining that decision, we find out that "dog pixels" affect it positively.
- However, the cat pixels affect the prediction as well, only negatively.
- Gradient-based focus map would not be able to differentiate these regions.

Class Activation Mapping

Alternative approach was proposed[9], which solves the problem of gradient methods. However, it only works on a specific CNN architecture. It follows these principles:

- All the explaining process occurs only after the last convolutional layer. This results in quicker but less precise explanations.
- In the predicting process, Global Average Pooling (GAP) is used after the convolutions.
- Afterwards, exactly one fully connected layer is used to predict the class.
- In the explaining process, GAP is ommited. We get vector of images instead of numbers.
- We use the weights from FC layer connected to the predicted class.

[9] Zhou *et al.*, 2016



CAM Visualization. Source: Zhou *et al.*, 2016

CAM – comparison with gradient methods

CAM:

- Faster to compute (operates on the last layer)
- Only the positively affecting regions count

Gradient methods:

- More precise (gradient for each input pixel individually).
- Information is not lost with convolution entangling.

Ideally, we would want benefits of both these algorithms. Solution – (Guided) Grad-CAM.

Grad-CAM

- Similar to CAM in principle, but looser structure.
- Split network into two parts – convolutional and task-specific.
- Convolutional part – same explainer as in CAM (plus ReLU).
- Task-specific part – compute gradient of this part to get the "weights".

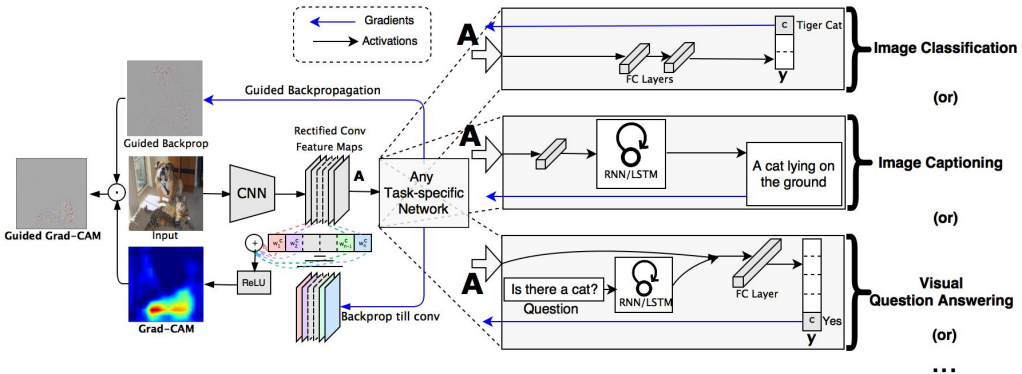
Grad-CAM

- Similar to CAM in principle, but looser structure.
- Split network into two parts – convolutional and task-specific.
- Convolutional part – same explainer as in CAM (plus ReLU).
- Task-specific part – compute gradient of this part to get the "weights".

Guided Grad-CAM:

- Motivated by higher precision of the resulting focus map.
- Calculate *guided backpropagation* through the convolutional layers.
- Multiply element-wise with Grad-CAM.

Grad-CAM



Grad-CAM Visualization. Source: Selvaraju *et al.*, 2019

Outline

Summary

By now, you should know. . .

- that explainability is (sometimes) important, and achievable with adequate tradeoffs.
- how to select an explainer with regards to the accuracy-explainability trade-off.
- the difference between interpretability and completeness.

Summary

By now, you should know...

- that explainability is (sometimes) important, and achievable with adequate tradeoffs.
- how to select an explainer with regards to the accuracy-explainability trade-off.
- the difference between interpretability and completeness.
- the advantages of model-agnostic explainers and when to use them.

Summary

By now, you should know...

- that explainability is (sometimes) important, and achievable with adequate tradeoffs.
- how to select an explainer with regards to the accuracy-explainability trade-off.
- the difference between interpretability and completeness.
- the advantages of model-agnostic explainers and when to use them.
- the specifics of transparent models and when to replace ML by statistics.

Summary

By now, you should know...

- that explainability is (sometimes) important, and achievable with adequate tradeoffs.
- how to select an explainer with regards to the accuracy-explainability trade-off.
- the difference between interpretability and completeness.
- the advantages of model-agnostic explainers and when to use them.
- the specifics of transparent models and when to replace ML by statistics.
- how image classification differs from other tasks in terms of explainers.
- that creating complete explanations for image classifier is hard (but we can try).