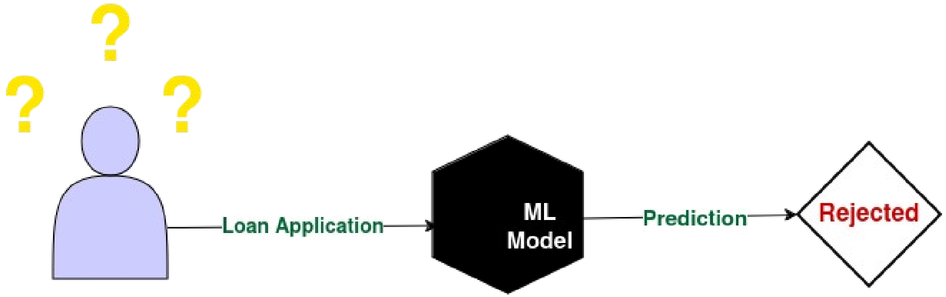
Explainable Artificial Intelligence (XAI)

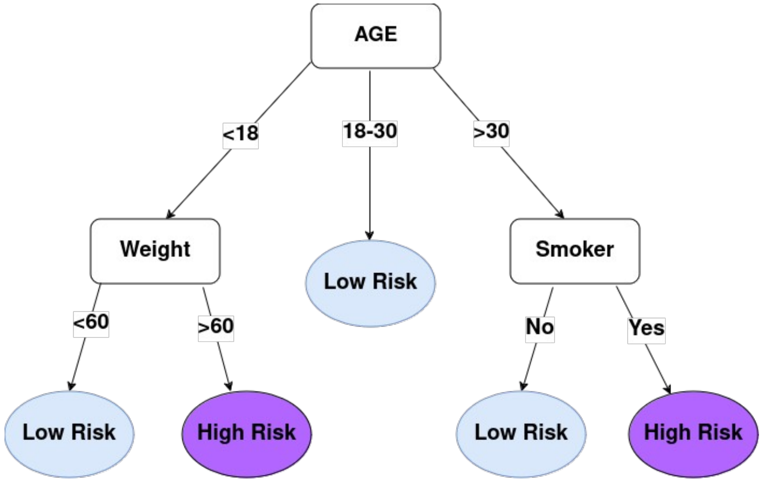
Introduction

- The need for explainability :
  - ML models achieved a significant results in data driven approaches.
  - ML has expanded into a broader array of industries.
  - In some fields failure is not an option.
  - Understand the decisions helps to increase the trust in ML models.
  - Improve performance of models.
  - Get a new knowledge(Hidden Laws) from ML Models



XAI Categories

- Types of Machine Learning Algorithms according interpretability:
  - Interpretable Models:
    - Regression Models
    - Decision Trees
    - KNN
  - Uninterpretable Models:
    - Neural Networks
    - SVM
    - Random Forests



XAI Categories

- According to the scope of the methods :
  - Local Explanations
  - Global Explanations
- According to the relationship to the model:
  - Model specific Explanations
  - Model-Agnostic Explanations
- According to the nature of the explanation:
  - Feature relevance explanations
  - Explanations by simplifications
  - Explanations by Visualizations
  - Explanations by Examples
  - Rule-based explanations

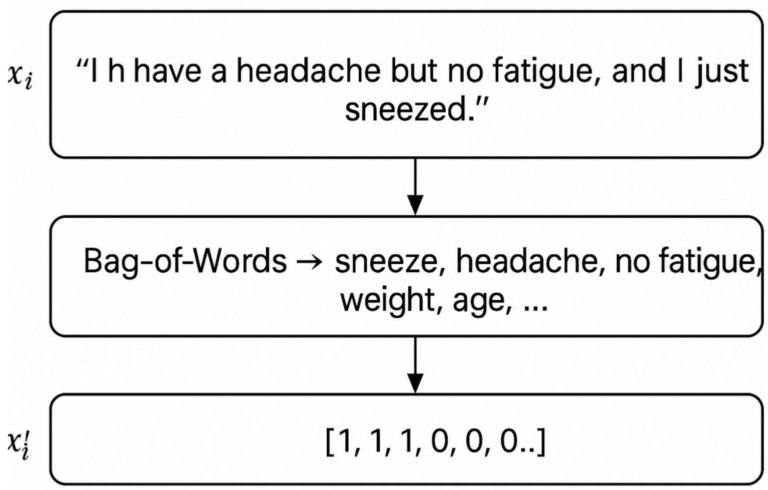
LIME (Why should I trust you?)

- Paper:
  - Ribeiro, Marco Tulio, Sameer Singh, and Carlos Guestrin. "" Why should i trust you?" Explaining the predictions of any classifier." Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. 2016.
- Categorization:
  - Local
  - Model-agnostic
  - Feature relevance
- Github Repository:
  - https://github.com/marcotcr/lime-experiments

LIME

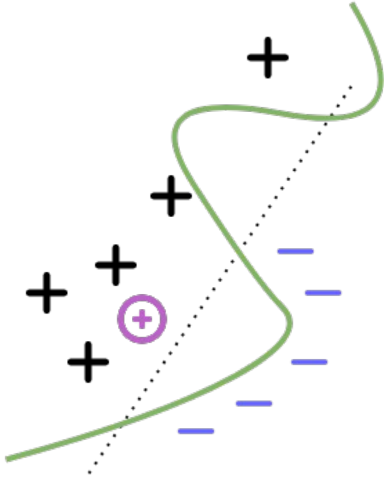
- Contribution:
  - LIME:Local Interpretable model agnostic.
  - SP-LIME: Method to select representative examples from dataset with their explanations. (global)
- Explanation:
  - A list of features with relative weight-features that either contribute to the prediction or are evidences against it.
- Interpretable Data Representations:
  - representation that is understandable to humans, regardless of the actual features used by the model

- Interpretable Data Representations(Cont):
  - Examples:
    - Text: binary vector indicating the presence or absence of a word
    - Images: may be a binary vector indicating the "presence" or "absence" of a contiguous patch of similar pixels (a super-pixel)
    - x ∈ Rd be the original representation
    - x' ∈ {0, 1}d' to denote a binary vector for its interpretable representation



LIME (Why should I trust you?)

- Idea:
  - Pick up a predection to explain
  - Generate new data points around the instance (Perturbation)
  - Feed these new data to the ML model to obtain outputs
  - Train a new (Interpretable) linear model to fit the inputs to the outputs
- Drawbacks:
  - If the model we want to explain is highly non-linear
  - The “Unclear coverage” problem

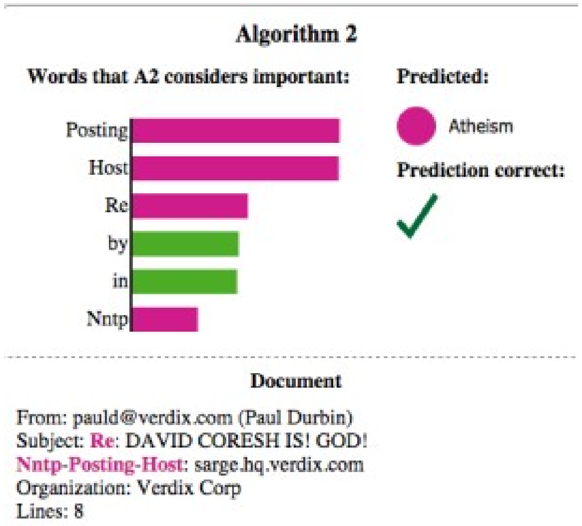


LIME: Technical Overview

- Explanation formula:
  - $\epsilon(x) = \underset{g \epsilon G}{argmin}\, L(f, g, \Pi_x) + \Omega(g)$
    - G family of interpretable models
    - F model to be explained
    - Πx proximity measure between x and z
    - Ω(g) complexity of g (depth for trees, #non-zero weights for linear,…)
    - General Framework
- Sampling:
  - Samples around x’ by drawing nonzero elements of x’ uniformly at random.
  - Samples are weighted by Πx (in vicinity of x heigh weight, far away low weight)
  - Sample z’ ∈{0,1}d’
  - Recover the sample to original representation z
  - Obtain f(z) as label

LIME: Technical Overview

- Explanation formula according LIME method:
  - G class of linear models: $g(z') = w_g \cdot z'$
  - $\pi_x(z) = \exp(-D(x, z)^2/\sigma^2)$
    - D : cosine distance for text, L2 distance for images
- Example 1:
  - Task: classify texts as “Christianity” or “Atheism”
  - SVM model accuracy: 94%
  - Explanation shows that:
    - There are arbitrary reasons for classification
    - “Posting”, “Host”, and “Re” are not related to the classes
    - But “Posting” appears in 22% of data points, 99% of these have “Atheism” as label



LIME: SP-LIME

- It picks a small number of important, diverse, and non-redundant predictions to explain, so the user can get a good overall sense of how the model behaves.
  - Explain many predictions using LIME
  - For each prediction, record which features appeared and how strongly in a table (Matrix)
  - Score feature importance overall by show up most often and most strongly across all explanations.
  - Pick the best set of predictions to show:
    - Start with none.
    - Then, one by one, pick the predictions that help “cover” the most important features not already seen.
    - Keep going until you’ve picked enough (based on how many the user can look at).
  - Present those explanations to the user
  - The selected examples together give the user a clear, non-repetitive picture of the model’s behavior.

LIME: Experiments And Results

- Models
  - Inherently interpretable
    - Sparse Logistic Regression (LR)
    - Decision Tree (DT)
  - Not interpretable:
    - Nearest Neighbor (NN)
    - SVM
    - Random Forest (RF)
- Datasets:
  - Books and DVDs sentiment analysis datasets
  - (Each contains 2000 product reviews labeled as positive or negative.)
- Explanation Methods Compared:
  - Random – Picks K features at random.
  - Parzen – Approximates the model globally with Parzen windows and uses gradients.
  - Greedy – Iteratively removes features until the prediction changes.
  - LIME – Uses local perturbations and sparse linear approximation

LIME: Experiments And Results

- Experiment 1: Are Explanations Faithful to the Model?
  - Goal:Check if explanations recover the real features used by the model.
  - Setup:
    - Models: Sparse Logistic Regression and Decision Tree (max 10 features).
    - Datasets: Books and DVDs sentiment classification.
    - For each test example:
      - Compare features in explanation to features used by the interpretable model
    - Metric:
      - Recall — percent of true model features included in the explanation.

Method	Books (LR - DT)	DVD (LR- DT)
Random	~17–20%	~17–19%
Parzen	~73–79%	~61–81%
Greedy	~37–64%	~47–63%
LIME	92–97%	90–98%

LIME: Experiments And Results

- Experiment 2: Should I Trust This Prediction?
- Goal:
  - Simulate if users can detect untrustworthy predictions using explanations.
- Setup:
  - Randomly label 25% of features as “bad.”
  - Oracle: prediction is “untrustworthy” if removing bad features changes it.
  - Simulated users trust/mistrust based on explanations.
- Metric:
  - F1 score — measures agreement with the oracle’s trust labels.

Method	LR	NN	RF	SVM
Random	14.6	14.8	14.7	14.7
Parzen	84.0	87.6	94.3	92.3
Greedy	53.7	47.4	45.0	53.3
LIME	96.6	94.5	96.2	96.7

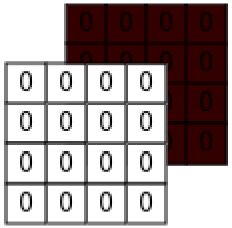
DeepLIFT

- Paper:
  - Shrikumar, Avanti, Peyton Greenside, and Anshul Kundaje. "Learning important features through propagating activation differences." International conference on machine learning. PMlR, 2017.
- Categorization:
  - Local
  - Model-specific (NN)
  - Feature relevance
- Github repository:
  - http://goo.gl/RM8jvH

DeepLIFT

- Idea:
  - Assign importance scores to the inputs for a given output
  - Backpropagation based method
  - Difference from reference instead of gradients
  - Reference is chosen according to the problem

- Main Drawbacks:
  - Choosing a reference
  - Implementations difficulties



Reference for MNIST dataset

Reference for CIFAR-10

DeepLIFT: Mechanism

- Propagate a signal from the output backward to the input
- Normally the gradient of the output will back-propagate
- Since gradients suffer from problems(saturation, vanishing)
- They introduced new signal to be back-propagated
- This signal depends on which called difference-from-reference
- t is the target output to explain, t_0 the reference output
- $\Delta t = t - t_0$ difference from reference
- Contribution $C_{\Delta x_i \Delta t}$:the amount of Δt that is 'blamed' on the Δ_{xi}
- Multipliers $m_{\Delta x \Delta t}$ = Difference from Reference/Contribution
- The multiplier is analogous to a gradient but calculated over finite differences rather than infinitesimally small ones, as in standard backpropagation.

DeepLIFT: Experiments And Results

- 1. Digit Classification (MNIST)
  - Setup: A CNN was trained on MNIST using Keras and achieved 99.2% test-set accuracy.
  - Goal: Identify which pixels to erase to change the classification from class to another.
  - Metric: Change in log-odds score after erasing pixels most important..
  - Methods Compared:
    - DeepLIFT with RevealCancel
    - Integrated Gradients
    - Gradient × Input
    - Guided Backpropagation
  - Results:
    - DeepLIFT (RevealCancel) outperformed all other methods in altering the model's output by pixel masking.
    - Guided Backpropagation underperformed likely due to discarding negative gradients.

DeepLIFT: Experiments And Results

- 2. Genomic Sequence Classification (Simulated DNA)
  - Setup: Simulated DNA sequences with controlled motif insertions for proteins GATA1 and TAL1.
  - Tasks:
    - Task 1: Both motifs present
    - Task 2: GATA1 present
    - Task 3: TAL1 present
  - Model: Multi-task CNN with two convolutional layers and a fully-connected output.
  - Metric: Match between motif strength (log-odds) and importance score assigned by each method.
  - Methods Compared: Same as last Experiment
  - Results:
    - DeepLIFT had the best performance.
    - Reduced noise and highlighted real motif interactions.
    - Gradient × Input and Guided Backprop had:
      - False positives/negatives due to saturation and gradient discontinuities.

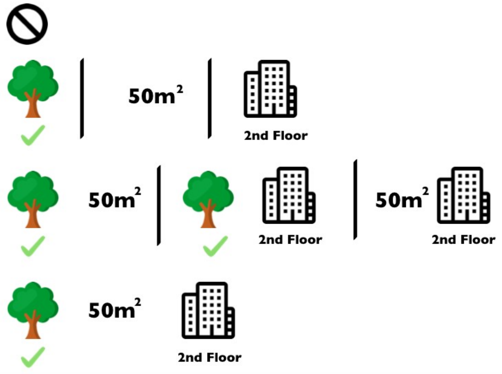
SHAP(SHapley Additive exPlanations)

- Paper
  - Lundberg, Scott M., and Su-In Lee. "A unified approach to interpreting model predictions." Advances in neural information processing systems 30 (2017).
- Contribution:
  - Identifies the class of additive feature importance methods
  - Define properties of this class:
    - Local accuracy
    - Missingness
    - Consistency
  - Shows that Shapely Values results guaranteeing a unique solution
  - Proposes new approximation methods for SHAP
- Github repository:
  - https://github.com/slundberg/shap

SHAP(SHapley Additive exPlanations)

- Idea:
  - Adapted from Shapley values concept in cooperative game theory.
  - Calculate the marginal contribution of each player (Feature) across all subsets
  - Average the marginal contributions

$$Shapely\ Values = \frac{1}{n!}\sum Marginal\ Contributions$$



- Drawbacks:
  - It can be very expensive in terms of computation time

Shapley Values: Example

Let:

v({1}) = 100, v({2}) =125, v({3}) = 50, v({1,2}) = 270, v({1,3}) = 375, v({2,3}) = 350 and v({1,2,3}) = 500

Then:

1's expected marginal contribution is: 1/6(100 + 100 + 145 + 150 + 325 + 150) =970/6

2's expected marginal contribution is: 1/6(170 + 125 + 125 + 125 + 125 + 300) = 970/6

3's expected marginal contribution is: 1/6(230 + 275 + 230 + 225 + 50 + 50) =1060/6

Sum = (970 +970 + 1060) / 6 = 500 = v({1,2,3})

Probability	Order of arrival	1's marginal contribution	2's marginal contribution	3's marginal contribution
1/6	first 1 then 2 then 3: 123	v({1}) = 100	v({1,2}) - v({1}) = 270 - 100 = 170	v({1,2,3}) - v({1,2}) = 500 - 270 = 230
1/6	first 1 then 3 then 2:	v({1}) = 100	v({1,2,3}) - v({1,3}) = 500 - 375 = 125	v({1,3}) - v({1}) = 375 - 100 = 275
1/6	first 2 then 1 then 3:	V({1,2})- v({2}) = 270 - 125 = 145	v({2}) =125	v({1,2,3}) - v({1,2}) = 500 - 270 = 230
1/6	first 2 then 3 then 1:	v({1,2,3}) - v({2,3}) = 500 - 350 = 150	v({2}) =125	v({2,3}) - v({2}) = 350 - 125 = 225
1/6	first 3 then 1 then 2:	v({1,3}) - v({3}) = 375 - 50 = 325	v({1,2,3}) - v({1,3}) = 500 - 375 = 125	v({3}) = 50
1/6	first 3 then 2 then 1:	v({1,2,3}) - v({2,3}) = 500 - 350 = 150	v({2,3}) - v({3}) = 350 - 50 =300	v({3}) = 50

SHAP : Technical Overview

- Kernel SHAP
  - Lime + SHAP
  - General LIME formula
    - $\epsilon(x) = \underset{g \epsilon G}{argmin}\, L(f, g, \Pi_x) + \Omega(g)$
  - Solution to ensure shapely values:
    - $\pi_{x'}(z') = \frac{(M-1)}{(M\, choose\, |z'|)\, |z'|\, (M - |z'|)}$
    - $L(f, g, \Pi_{x'}) = \sum_{z' \in Z} [f(h_x(z')) - g(z')]^2 \Pi_{x'}(z')$
    - Where M = number of features, |z’| = number of non zero features

SHAP: Experiments And Results

- Experiment 1:
  - Goal:
    - Compare how well 3 methods estimate feature importance:
      - Kernel SHAP
      - Shapley Sampling
      - LIME
  - Models Tested:
    - Model A (Dense):
      - 10 input features
      - All features used by the model
    - Model B (Sparse):
      - 100 input features
      - Only 3 features used by the model

SHAP: Experiments And Results

- Method:
  - For each model, they explained one prediction by estimating the importance of one feature, repeated 200 times with increasing sample sizes.
- What They Measured:
  - They measured how accurately, how quickly, and how consistently each method estimated the true Shapley value.
- Results:
  - Kernel SHAP Most accurate
  - Needs fewer samples
  - Stable results

SHAP: Experiments And Results

- Experiment 2: Human Intuition Consistency
  - Goal:
    - Test which method best matches human reasoning in explaining model predictions.
  - Setup:
    - In the first task, participants judged which symptom (fever or cough) contributed more to a sickness score of 2.
    - In the second, they divided a $5 prize among three people based on who had the highest score.
  - Method:
    - They compared human answers to SHAP, LIME, and DeepLIFT explanations.
  - What they measured:
    - How often each method's attributions matched the majority of human responses.
  - Results:
    - SHAP aligned best with human intuition.
    - LIME and DeepLIFT gave inconsistent or wrong explanations

SHAP: Experiments And Results

- Experiment 3: Explaining Class Differences (Image Classification)
  - Goal:
    - Evaluate how well different methods identify which image pixels influence a model's prediction change between two classes.
  - Setup:
    - A neural network trained on MNIST digits was used to classify the digit “8”, and explanations were tested by masking pixels to flip the prediction to a “3”.
  - Method:
    - They used SHAP, LIME, and two versions of DeepLIFT to generate pixel importance maps, then measured how masking top-ranked pixels affected the model’s output.
  - What they measured:
    - How effectively each method's pixel attributions reduced the model's confidence in the original class.
  - Results:
    - SHAP and the updated DeepLIFT caused the biggest drop in confidence when masking, meaning better explanations.

Anchors

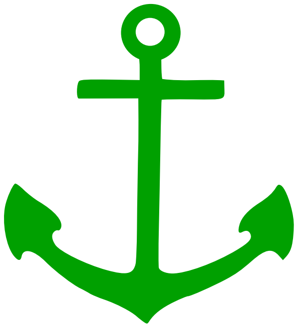
- Paper:
  - Ribeiro, Marco Tulio, Sameer Singh, and Carlos Guestrin. "Anchors: High-precision model-agnostic explanations." Proceedings of the AAAI conference on artificial intelligence. Vol. 32. No. 1. 2018
- Categorization:
  - Local
  - Model-agnostic
  - Rule-based
- Github repository:
  - https://github.com/marcotcr/anchor-experiments

Anchors

- Idea:
  - $X(x_1,x_2,\ldots,x_n)$ is an instance to explain it's prediction f(x)
  - A Rule (A) is a set of features
  - D(Z|A) is a set of perturbed samples of X when A applies
  - Example:
    - Instance (x) to explain: "This movie is **not bad" ,** f(x) = Positive
    - Rule A : (**Not,Bad**)
    - D(Z|A): "This actor is **not bad**"
"The idea is **not bad**"
"The audio was **not bad**"

Anchors

- A is an anchor to the model if:
  - If the proportion of Z when f(x) = f(z) >= τ then A is an Anchor for the model
- To find an Anchor:
  - Start with empty set of features
  - Add the best (new) candidate feature to A:
    - Multi Hands Brandit Problem
    - KL-LUCB to find best arm (Rule)
- Drawbacks:
  - Many hyper-parameters
  - Potentially conflicting anchors



Anchors:Experiments And Results

- Experiment 1: Simulated Users on Tabular Data
  - Goal:
    - Test whether anchors provide more accurate and consistent explanations than LIME for tabular models.
  - Setup:
    - Simulated users received explanations for predictions on three datasets (adult, recidivism, lending), using three models (logistic regression, gradient boosting, neural nets).
  - Method:
    - For each instance, anchors and LIME explanations were generated. Simulated users applied them to test data and predicted outcomes.
  - What they measured:
    - Average precision (correctness) and coverage (how many instances users could confidently predict).
  - Results:
    - Anchors consistently gave high precision (~95–99%), while LIME precision varied widely.
    - LIME sometimes had higher coverage, but only after fine-tuning thresholds.

Anchors:Experiments And Results

- Experiment 2: Real User Study
  - Goal:
    - See how well real users understand and apply anchors compared to LIME.
  - Setup:
    - 26 machine learning students predicted model outputs on adult income, recidivism, and visual question answering (VQA) tasks — before and after seeing explanations.
  - Method:
    - Each user saw predictions without explanation, then with LIME or anchor explanations (1 and 2 rounds), then made predictions on new test instances.
  - What they measured:
    - User precision, perceived coverage (how often they felt confident), and time taken per prediction.
  - Results:
    - With anchors, users achieved ~95–100% precision across all tasks.
    - LIME explanations led to lower precision and more errors.

Counterfactual Explanations

- Paper:
  - Wachter, Sandra, Brent Mittelstadt, and Chris Russell. "Counterfactual explanations without opening the black box: Automated decisions and the GDPR." Harv. JL & Tech. 31 (2017): 841.
- Categorization:
  - Local
  - Model-agnostic/specific
  - Explanations by example
- Datasets:
  - The Law School Admission Test (LSAT)
  - Pima Diabetes Database

Counterfactual Explanations

- Idea:
  - Select an instance (x) to explain
  - Select a desirable output y’
  - Find the minimal changes for features to get y’
  - By minimizing the loss between y and y’
- Example:
  - “You were denied a loan because your annual income was £30,000. If your income had been £45,000, you would have been offered a loan.”
- Drawbacks:
  - Sometimes the explanations are not realistic
  - Multiple explanations for same instance

Experiments And Results

- Experiment 1: LSAT Dataset
  - Goal:
    - Show that counterfactual explanations can reveal model bias and provide actionable changes without explaining the internal logic.
  - Setup:
    - A neural network was trained to predict law students’ first-year average grade using race, GPA, and LSAT score.
  - Method:
    - For each student, they generated counterfactuals answering: “What needs to change for the model to predict an average score (0)?”
    - They tested different distance functions to make the counterfactuals more realistic and sparse.
  - What they measured:
    - How often race had to change to “white” for black students to receive a better predicted score.
    - Also examined the realism and simplicity of the counterfactuals.
  - Results:
    - Counterfactuals showed that many black students would receive better predictions if they were white, revealing racial bias.
    - Using the right distance metric improved interpretability and prevented weird changes like negative race values.

Experiments And Results

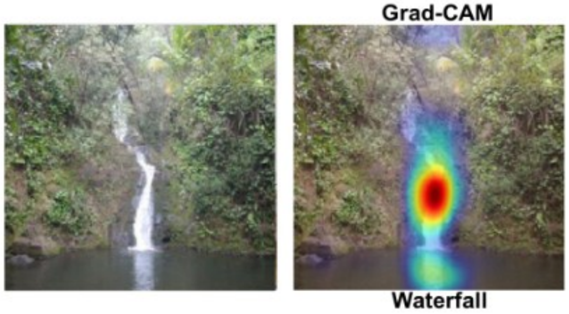
- Experiment 2: Pima Diabetes Dataset
  - Goal:
    - Test if counterfactuals can generate realistic, personalized risk explanations for complex health predictions.
  - Setup:
    - A neural network was trained to predict diabetes risk for Pima Indian women using 8 health-related features (e.g. BMI, age, insulin).
  - Method:
    - They created counterfactuals for women with high risk, asking: “What small changes would reduce the risk to 0.5?”
  - What they measured:
    - How few variables needed to change to lower risk, and whether the results were easy to interpret.
  - Results:
    - Most counterfactuals changed only 1 or 2 features, like insulin level or glucose concentration, and matched how doctors explain risks.

Grad-CAM

- Paper:
    - Selvaraju, Ramprasaath R., et al. "Grad-cam: Visual explanations from deep networks via gradient-based localization." Proceedings of the IEEE international conference on computer vision. 2017.
- Categorization:
    - Local
    - Model-specific
    - Explanations by visualization
- Github repository:
    - https://github.com/ramprs/grad-cam/

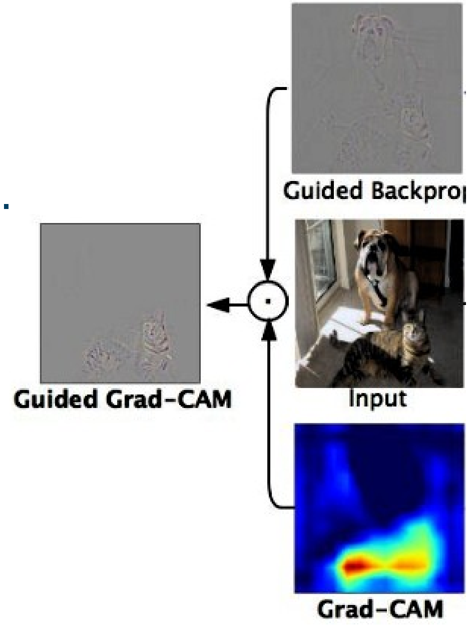
Grad-CAM

- Idea:
  - visualize which regions of an image are most important for a (CNN)
  - Since Deeper convolutional Layers capture High-level constructs
  - Obtain the importance weights α_k for each feature map A_k in last CL:
    - Calculate the gradients of the predicted class w.r.t to the feature map activations
    - Average these gradients
  - Compute localizations-maps L_{cam} as the weighted sum of feature maps



Guided Grad-CAM

- Idea:
  - A visualization method that combines Grad-CAM (class-discriminative localization) with Guided Backpropagation (high-resolution detail).
  - Grad-CAM highlights where the model looks to make a decision.
  - Guided Backpropagation shows what features the model detects.
  - Guided Grad-CAM multiplies both, giving sharp, class-specific visualizations.
  - Why it's useful:
  - Localizes specific object features (e.g. cat stripes, car wheels).
  - Helps users understand why a model made a prediction.



Grad-CAM: Experiments And Results

- Experiment 1: Weakly-Supervised Localization (ImageNet)
  - Goal:
    - Test how well Grad-CAM localizes objects in image classification tasks.
  - Setup:
    - Used VGG-16, AlexNet, and GoogleNet pretrained on ImageNet. No bounding boxes were provided during training.
  - Method:
    - Predicted class → generated Grad-CAM heatmap → thresholded → drew bounding box around top region.
  - What they measured:
    - Top-1 and top-5 localization error on the ILSVRC-15 validation set.
  - Results:
    - Grad-CAM had lower localization error than other methods (Backprop, c-MWP, CAM) while maintaining classification accuracy.

Grad-CAM: Experiments And Results

- Experiment 2: Class Discrimination (Human Study)
  - Goal:
    - Check if Grad-CAM explanations help humans identify the class being visualized.
  - Setup:
    - Amazon Mechanical Turk users saw images with two objects and class-specific visualizations from four methods(Guided Grad-CAM, Guided Backpropagation, Deconvolution,Deconv Grad-CAM).
  - Method:
    - Participants guessed which object was being explained in the visualization.
  - What they measured:
  - Human classification accuracy based on visual explanations.
- Results:
  - Guided Grad-CAM outperformed all other methods, improving accuracy by 17% over Guided Backprop.

Grad-CAM: Experiments And Results

- Experiment 3: Trust Evaluation (Human Study)
  - Goal:
    - Test if Grad-CAM helps users trust better-performing models.
  - Setup:
    - VGG-16 and AlexNet made the same prediction on an image. Users compared their explanations.
  - Method:
    - Workers rated which model looked more reliable using the visualization alone.
  - What they measured:
    - Relative trust score assigned by users to each model.
  - Results:
    - With Guided Grad-CAM, users consistently preferred VGG-16, the more accurate model.

Class Model Visualisation & Saliency Maps

- Paper:
  - Simonyan, Karen, Andrea Vedaldi, and Andrew Zisserman. "Deep inside convolutional networks: Visualising image classification models and saliency maps." arXiv preprint arXiv:1312.6034 (2013).
- Categorization:
  - Local/Global
  - Model-Specific
  - Explanations by visualizations
- Dataset:
  - ILSVRC-2013

Class Model Visualisation & Saliency Maps

- Two methods proposed:
  - Global:
    - Class model visualization:
      - Generating an Image for a class of interest
      - Find an Image (Input X) that has high class-score

goose

  - Local:
    - Backpropagation as well but to the input layer

Experiments And Results

- Experiment 1: Class Appearance Visualisation
  - Goal:
    - Visualize what a ConvNet has learned for each class.
  - Setup:
    - Used a deep ConvNet trained on ILSVRC-2013 (ImageNet) with 1.2 million labeled images.
  - Method:
    - Generated synthetic images that maximize the score of a specific class neuron using gradient ascent with L2 regularization.
  - What they measured:
    - Visual clarity of generated class images — how well they reflect class features.
  - Results:
    - The optimized images revealed meaningful visual patterns (e.g. dog fur, bird shapes) for each class.

Experiments And Results

- Experiment 2: Image-Specific Saliency Maps
  - Goal:
    - Identify which pixels in a given image contribute most to a specific class prediction.
  - Setup:
    - Used the same ConvNet from Experiment 1, applied to real ILSVRC-2013 test images.
  - Method:
    - Computed the gradient of the class score w.r.t. the input image to produce a saliency map.
    - Then averaged saliency maps from 10 cropped sub-images (to match test-time augmentation).
  - What they measured:
    - Pixel-level relevance to the top predicted class, visually inspected on test images.
  - Results:
    - Saliency maps highlighted object regions clearly, without needing bounding box labels.
    - Used these maps to initialize object segmentation via GraphCut and submitted to ILSVRC-2013.

Other References

- Holzinger, Andreas, et al. "Explainable AI methods-a brief overview." International workshop on extending explainable AI beyond deep models and classifiers. Springer, Cham, 2022.
- Adadi, Amina, and Mohammed Berrada. "Peeking inside the black-box: a survey on explainable artificial intelligence (XAI)." IEEE access 6 (2018): 52138-52160.
- Nadeem, Azqa, et al. "Sok: Explainable machine learning for computer security applications." 2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P). IEEE, 2023.
- Doshi-Velez, Finale, and Been Kim. "Towards a rigorous science of interpretable machine learning." arXiv preprint arXiv:1702.08608 (2017).
- Brdnik, Saša, and Boštjan Šumak. "Current Trends, Challenges and Techniques in XAI Field; A Tertiary Study of XAI Research." 2024 47th MIPRO ICT and Electronics Convention (MIPRO). IEEE, 2024.
- Belle, Vaishak, and Ioannis Papantonis. "Principles and practice of explainable machine learning." Frontiers in big Data 4 (2021): 688969.

Thanks!