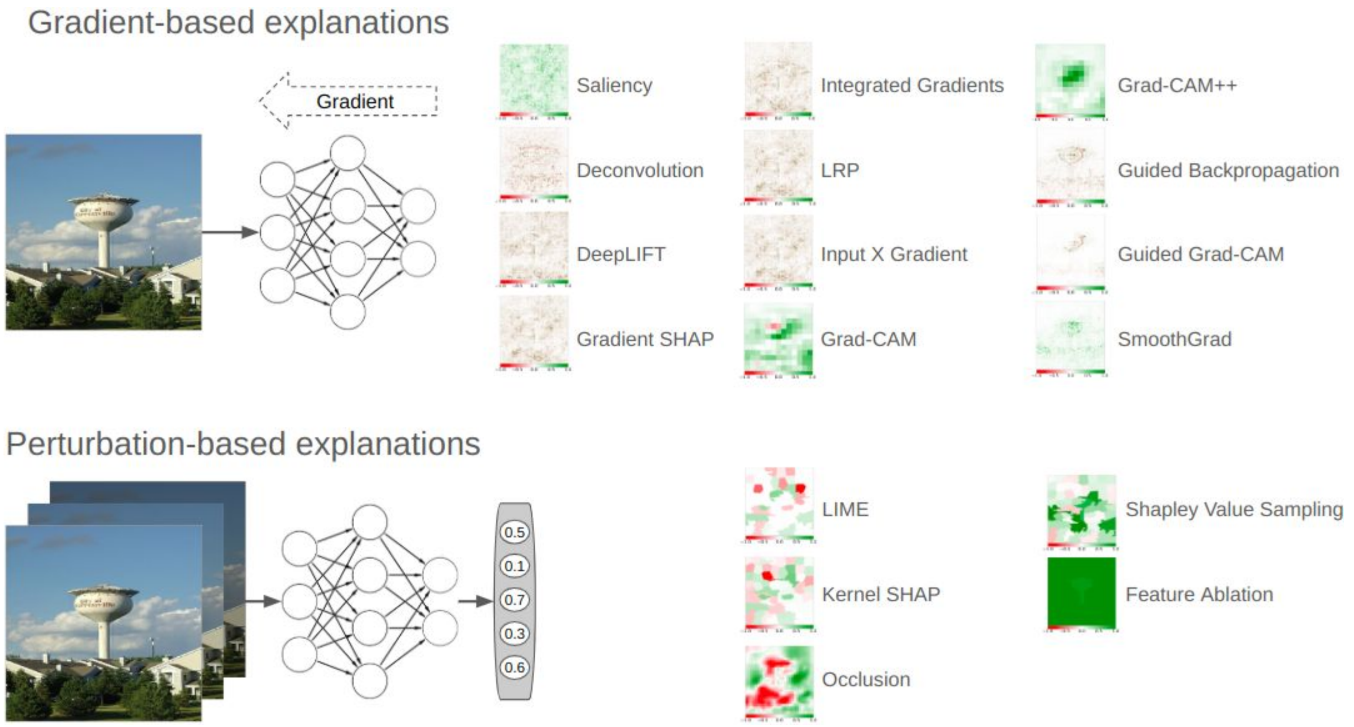
NormEnsembleXAI:
Unveiling the Strengths and Weaknesses of XAI Ensemble Techniques

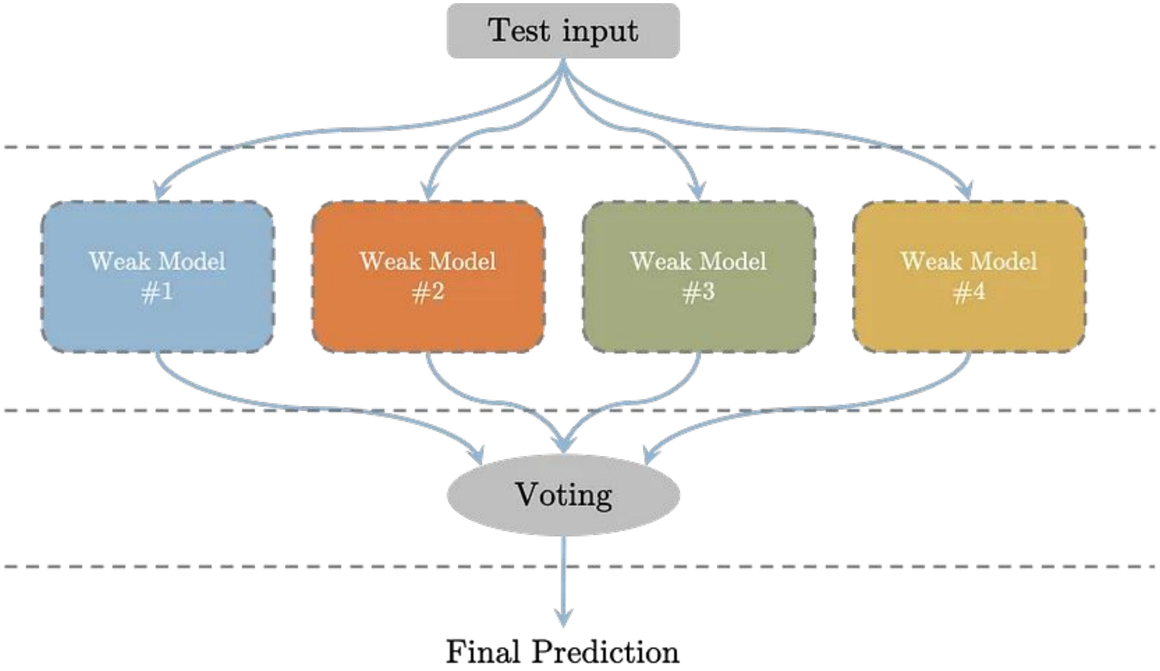
Weronika Hryniewska-Guzik
PhD candidate at Warsaw University of Technology

Post-hoc, local, attribution-based explanations



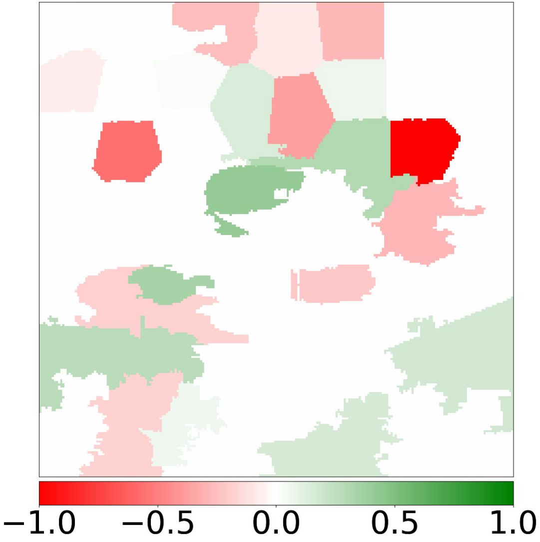
Source: Author's doctoral dissertation

Ensembling of explanations

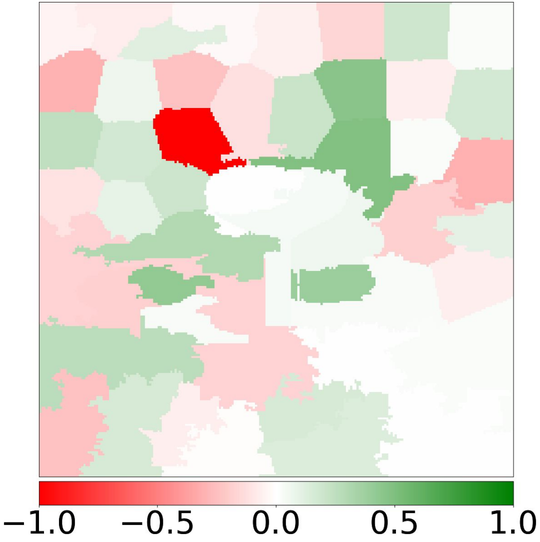


https://towardsdatascience.com/what-are-ensemble-methods-in-machine-learning-cac1d17ed349

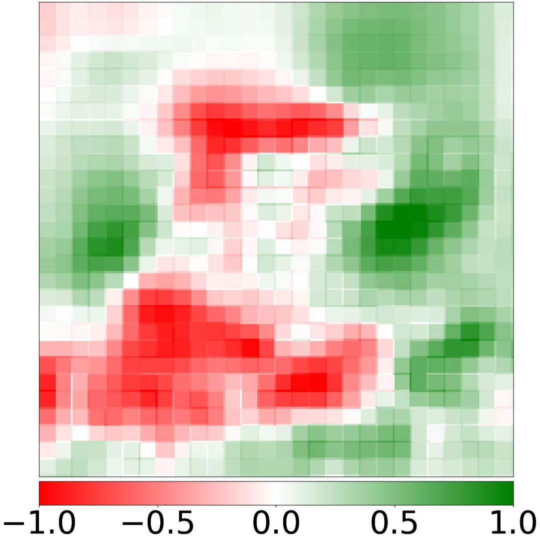
Motivation: mutually exclusive explanations



LIME



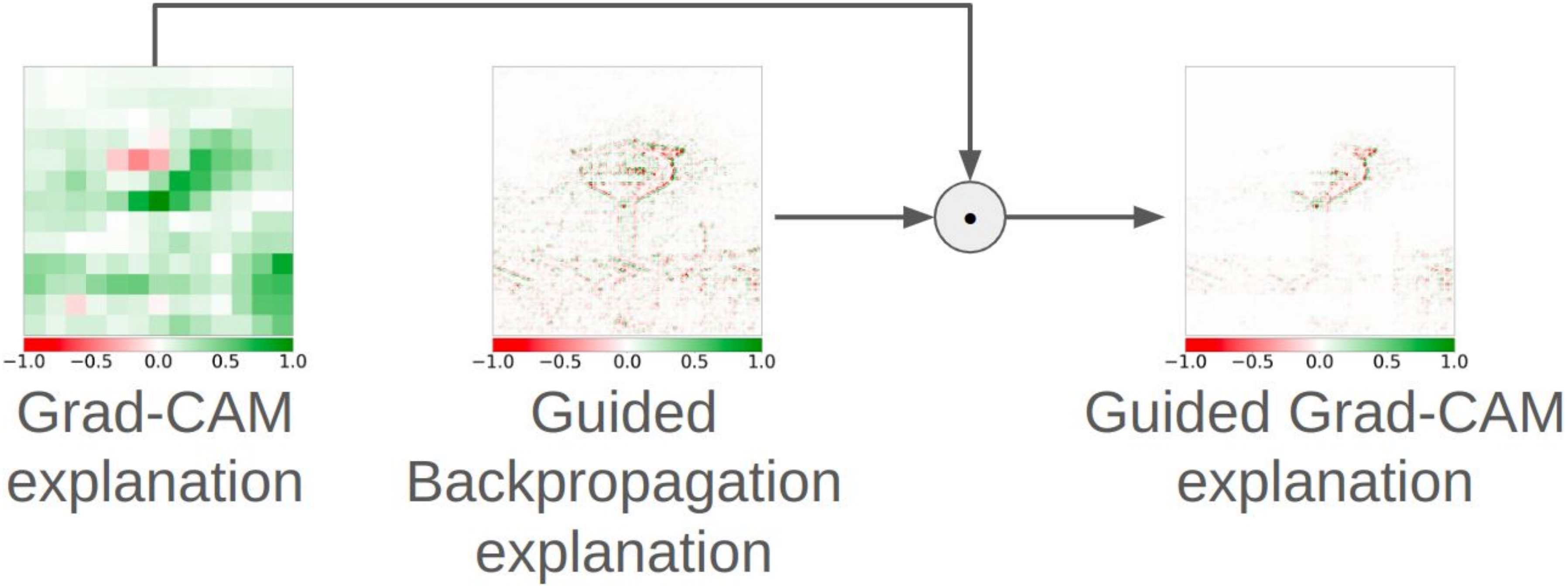
Kernel SHAP



Occlusion

Source: Author's doctoral dissertation

Examples of explanation ensembling



Source: Author's doctoral dissertation

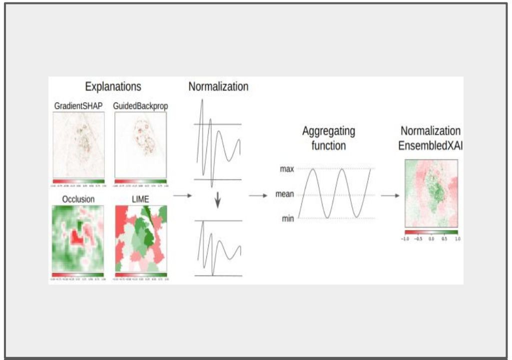
Scope of the presentation

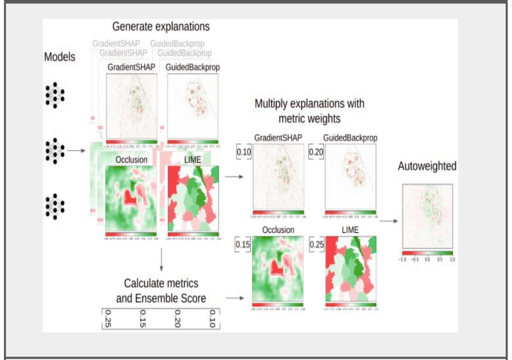
1. Introduction to three methods for ensembling explanations

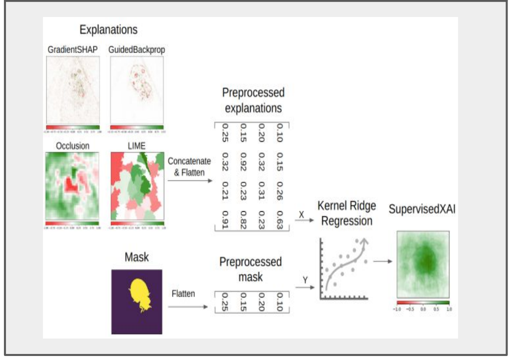
2. Evaluation of these methods

3. Practical applications using an open-source library

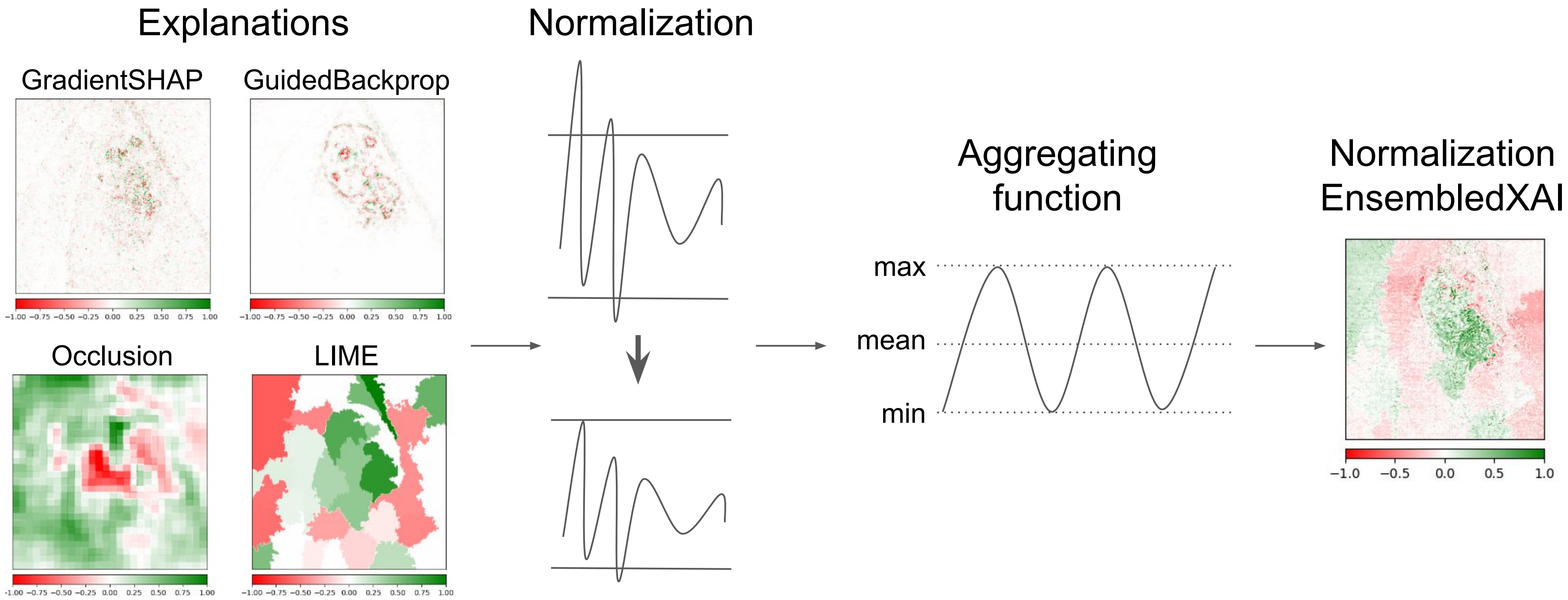
4. Summary







NormEnsembleXAI – our XAI ensemble method



Normalization methods

- Normal Standardization

$$\phi'^{e \to m}_{i,j} = \frac{\phi^{e \to m}_{i,j} - mean_{k,l}(\phi^{e \to m}k, l)}{std_{k,l}(\phi^{e \to m}_{k,l})}$$

- Robust Standardization

$$\phi'^{e \to m}_{i,j} = \frac{\phi^{e \to m}_{i,j} - median_{k,l}(\phi^{e \to m}_{k,l})}{IQR_{k,l}(\phi^{e \to m}_{k,l})}$$

- Second Moment Scaling

$$\phi'^{e \to m}_{i,j} = \frac{\phi^{e \to m}_{i,j}}{avg_{channels}(std_{k,l}(\phi^{e \to m}_{k,l}))}$$

Where:

- $\phi^{e \to m}_{i,j}$ represents the importance of feature j of instance i by explanation method e of machine learning model m.
- Mean and standard deviation are calculated across all instances k and features l.

Aggregation methods

Once normalization is applied, the normalized values $\phi'^{e \to m}_{i,j}$ are aggregated using one of the functions (e.g., maximum, minimum, or mean):

$$\phi^{ens \to m}_{i,j} = f_{e \in E_i}(norm(\phi^{e \to m}_{i,j}))$$

Where:

- f is the aggregation function (e.g., max, min, or average).
- norm refers to one of the normalization methods described above.

Aggregation methods

- Average
- Maximum
- Maximum absolute
- Median
- Entropy-based
- Exponential

Visualizations of NormEnsembleXAI

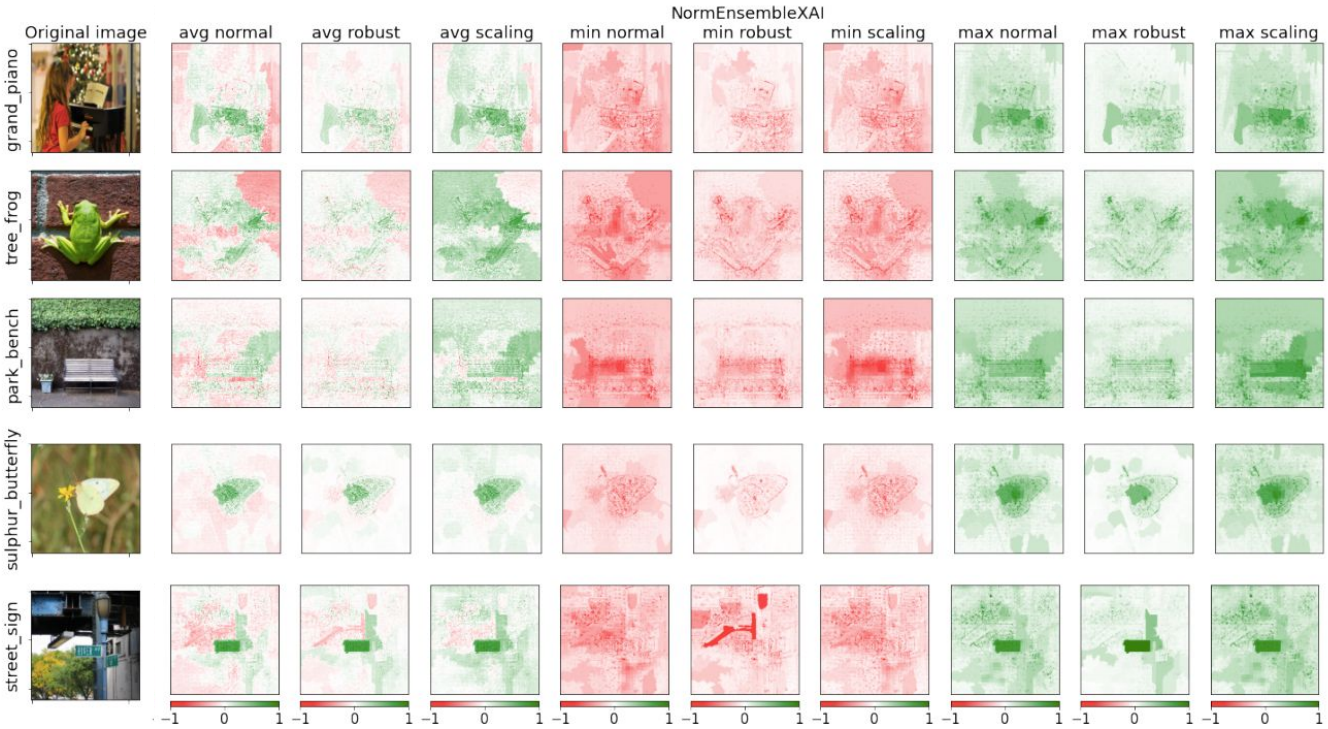
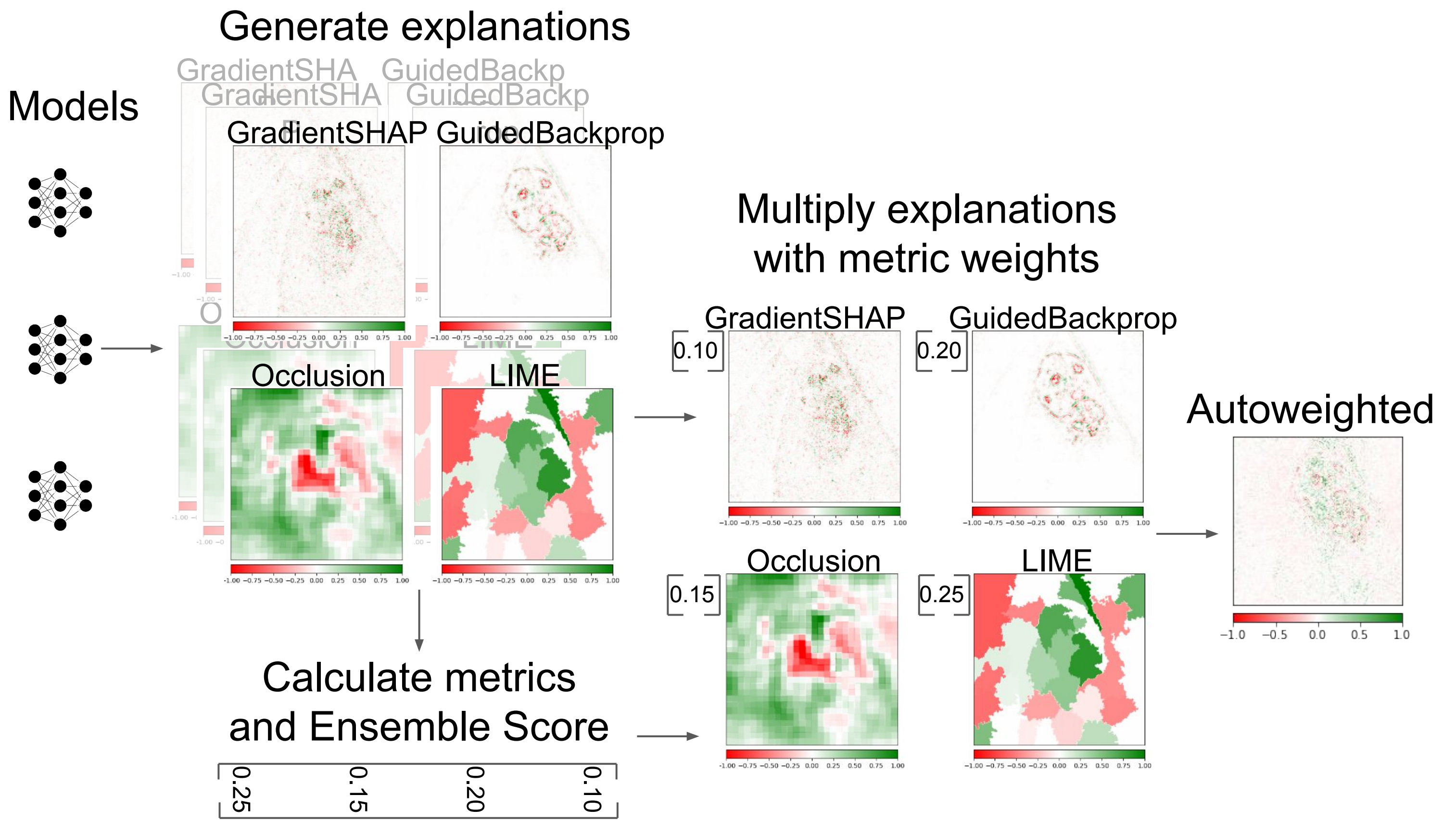


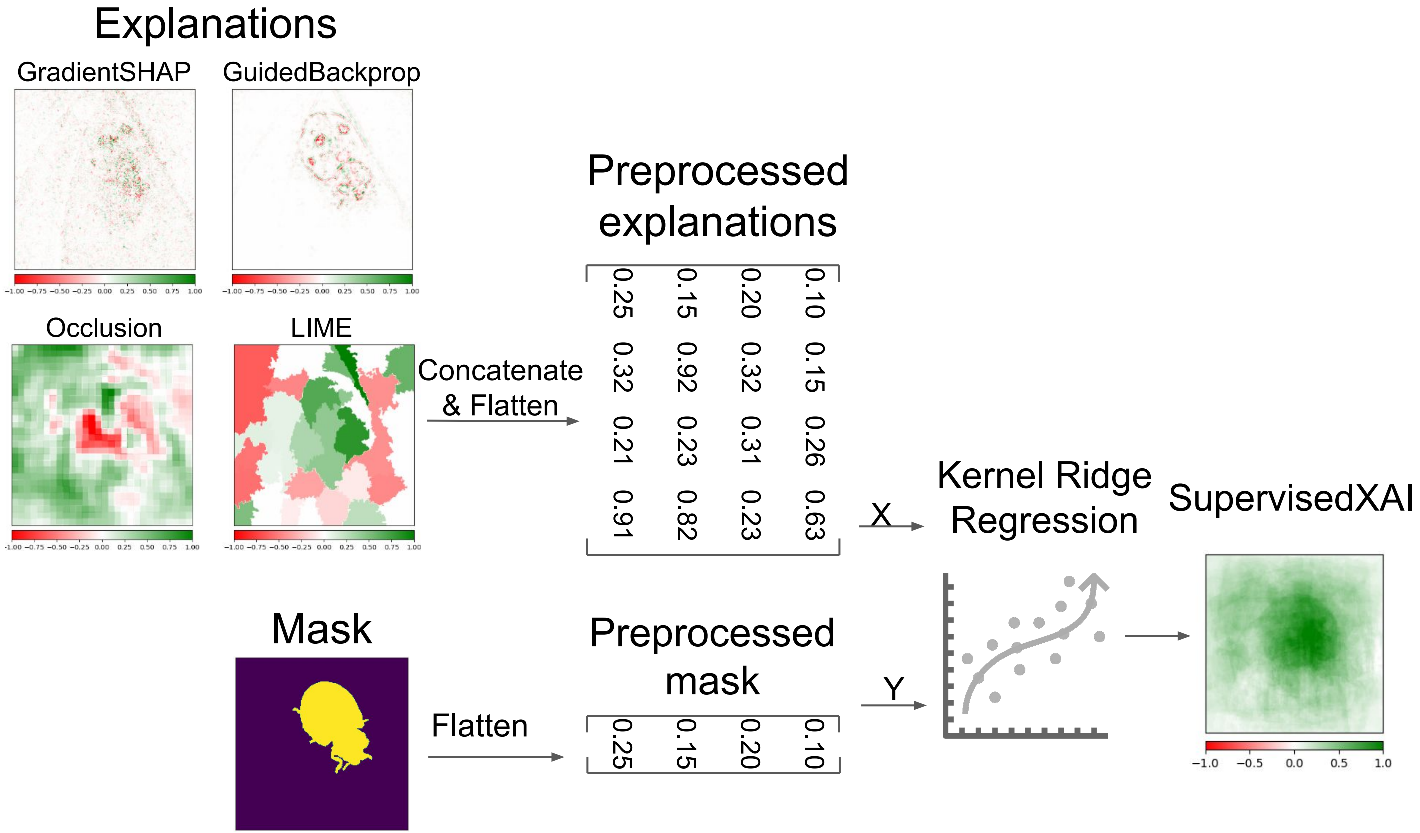
Figure 7. Examples of XAI ensembling results.

Autoweighted – XAI ensemble method



Bobek, S., Bałaga, P., & Nalepa, G. J. (2021). Towards Model-Agnostic Ensemble Explanations. *Lecture Notes in Computer Science*, *12745 LNCS*, 39–51. https://doi.org/10.1007/978-3-030-77970-2_4

SupervisedXAI – XAI ensemble method



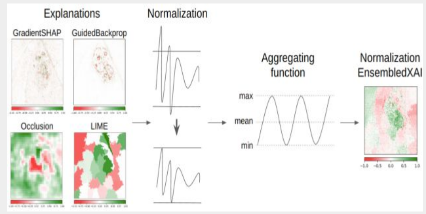
Zou, L., Goh, H. L., Liew, C. J. Y., Quah, J. L., Gu, G. T., Chew, J. J., Prem Kumar, M., Ang, C. G. L., & Ta, A. (2022). Ensemble image explainable AI (XAI) algorithm for severe community-acquired pneumonia and COVID-19 respiratory infections. *IEEE Transactions on Artificial Intelligence*, 1–1. https://doi.org/10.1109/TAI.2022.3153754

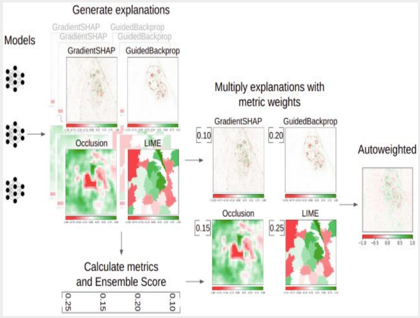
Limitations of XAI ensembling methods

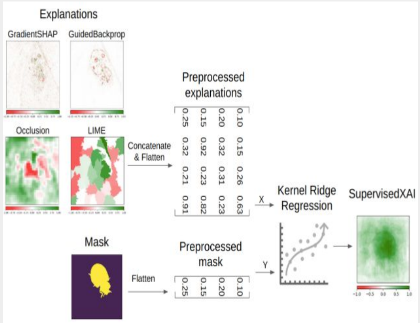
Time consumption

Method	Average time (s)
Autoweighted	48.699 ± 3.297
SupervisedXAI(500 samples)	41.858 ± 0.028
SupervisedXAI(20 samples)	0.972 ± 0.046
NormEnsembleXAI normal avg	0.114 ± 0.078
NormEnsembleXAI scaling avg	0.088 ± 0.059

Table 3. Average time (in seconds) of ensembling explanations without generating component explanations.



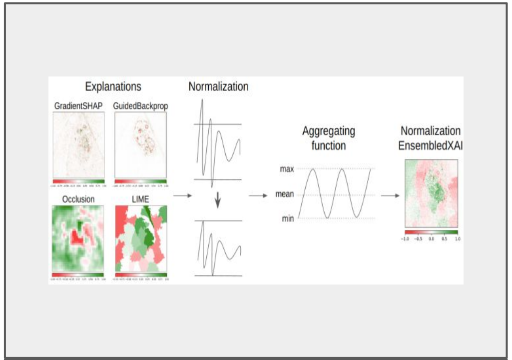


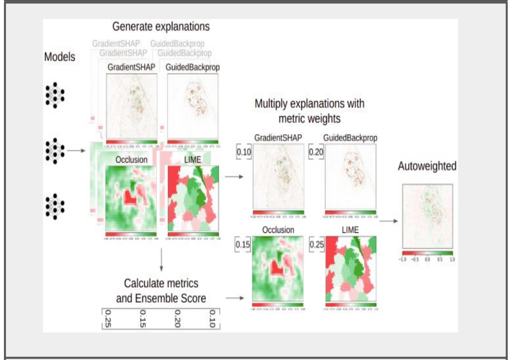


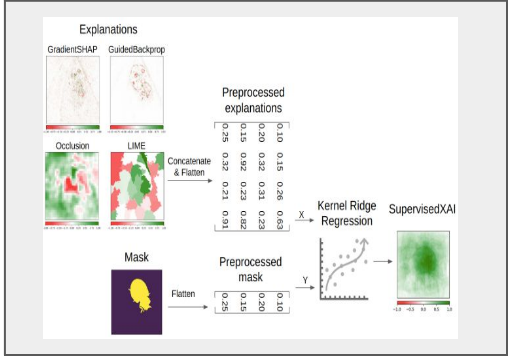
Limitations of XAI ensembling methods

Possibility of bias:

- NormEnsembleXAI - choice of aggregation function and normalization
- Autoweighted - the metric selection may introduce bias towards selected metric
- SupervisedXAI - the highest attribution was in the center of the image, and the attribution was close to 0 near the edges



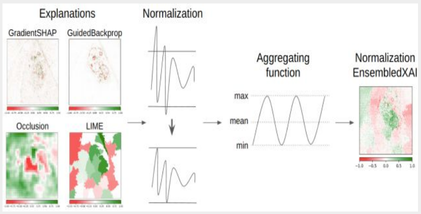


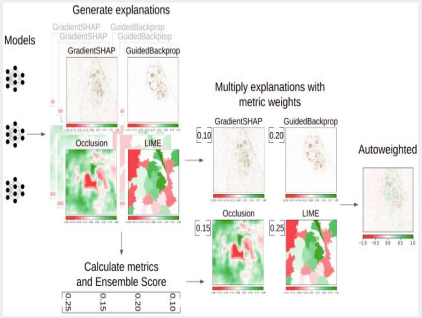


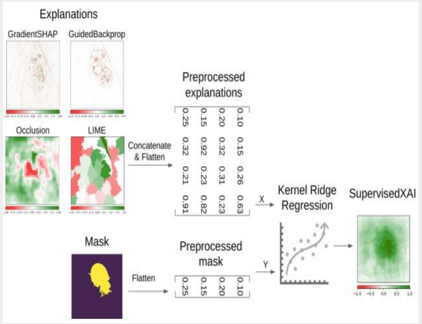
Limitations of XAI ensembling methods

Requirement of additional resources:

- NormEnsembleXAI - nothing
- Autoweighted - multiple models
- SupervisedXAI - pixel-wise annotations



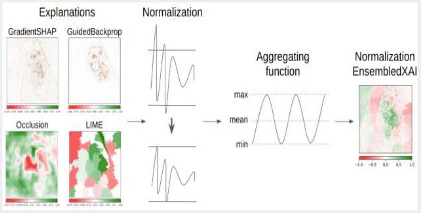


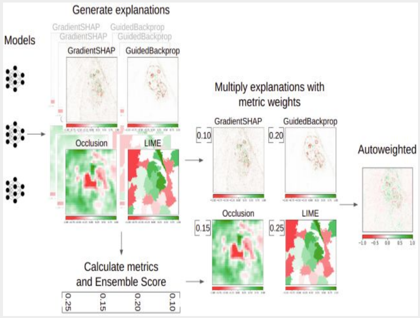


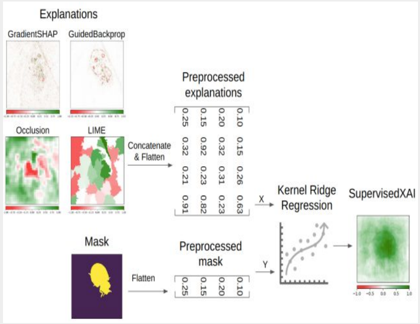
Limitations of XAI ensembling methods

Only positive feature attributions:

- SupervisedXAI - binary basks [0, 1]







Visualizations of all EnsembleXAI methods

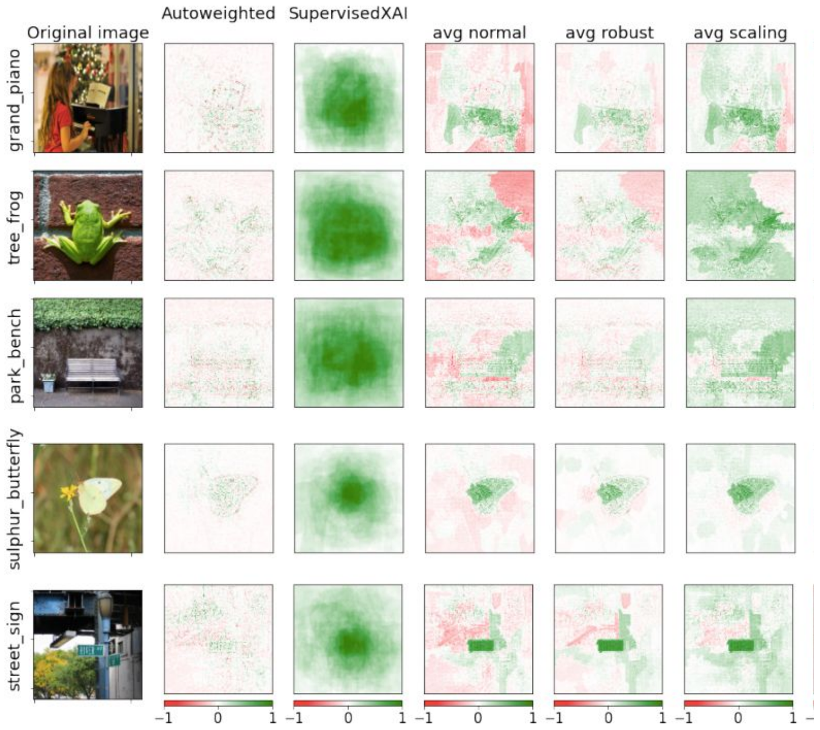


Figure 7. Examples of XAI ensembling results.

Metrics for measuring the quality of explanations

- Faithfulness (Fa) is assessed using Pixel-Flipping,
- Randomization (Ra) through Random Logit,
- Robustness (Ro) via Local Lipschitz Estimation,
- Complexity (Co) using Sparseness,
- Localization (Lo) determined with the Pointing-Game.

Complexity

Uses Sparseness to evaluate simplicity of explanations.

ImageNet

Metric	Aggr.	normal	robust	scaling
Complexity	avg	0.462 ± 0.028	0.728 ± 0.108	0.469 ± 0.045
	entr	0.465 ± 0.051	0.530 ± 0.014	0.451 ± 0.054
	exp	0.516 ± 0.075	**0.829 ± 0.095**	0.522 ± 0.084
	max	0.359 ± 0.044	**0.778 ± 0.172**	0.311 ± 0.046
	med	0.465 ± 0.027	0.519 ± 0.035	0.552 ± 0.045
	min	0.304 ± 0.024	**0.823 ± 0.180**	0.364 ± 0.035

CIFAR

Metric	Aggr.	normal	robust	scaling
Complexity	avg	0.443 ± 0.021	0.754 ± 0.095	0.405 ± 0.053
	entr	0.455 ± 0.042	0.547 ± 0.026	0.374 ± 0.050
	exp	0.477 ± 0.047	**0.849 ± 0.078**	0.458 ± 0.064
	max	0.346 ± 0.033	**0.833 ± 0.122**	0.237 ± 0.048
	med	0.451 ± 0.033	0.519 ± 0.029	0.527 ± 0.041
	min	0.276 ± 0.035	**0.859 ± 0.137**	0.370 ± 0.031

FashionMNIST

Metric	Aggr.	normal	robust	scaling
Complexity	avg	0.505 ± 0.030	0.709 ± 0.109	0.509 ± 0.041
	entr	0.514 ± 0.034	0.527 ± 0.040	0.448 ± 0.066
	exp	0.553 ± 0.055	**0.834 ± 0.094**	0.522 ± 0.075
	max	0.387 ± 0.039	**0.732 ± 0.121**	0.277 ± 0.064
	med	0.654 ± 0.058	0.656 ± 0.081	0.728 ± 0.079
	min	0.325 ± 0.026	**0.746 ± 0.121**	0.466 ± 0.052

Metric	Method	ImageNet results
Complexity	autoweighted	0.526 ± 0.030
	supervisedXAI_auto	0.395 ± 0.045
	supervisedXAI_no	0.393 ± 0.042

Metric	Method	CIFAR results
Complexity	autoweighted	0.470 ± 0.025
	supervisedXAI_auto	0.249 ± 0.098
	supervisedXAI_no	0.247 ± 0.098

Metric	Method	FashionMNIST results
Complexity	autoweighted	0.408 ± 0.021
	supervisedXAI_auto	0.375 ± 0.093
	supervisedXAI_no	0.376 ± 0.093

Localization

Uses Pointing-Game to assess if explanations focus on relevant areas.

ImageNet

Metric	Aggr.	normal	robust	scaling
Localisation	avg	**0.790 ± 0.409**	0.700 ± 0.461	**0.790 ± 0.409**
	entr	0.560 ± 0.499	**0.910 ± 0.288**	0.620 ± 0.488
	exp	0.700 ± 0.461	0.690 ± 0.465	0.740 ± 0.441
	max	0.670 ± 0.473	0.750 ± 0.435	0.660 ± 0.476
	med	0.780 ± 0.416	0.780 ± 0.416	0.720 ± 0.451
	min	0.638 ± 0.484	0.635 ± 0.484	0.570 ± 0.498

CIFAR

Metric	Aggr.	normal	robust	scaling
Localisation	avg	0.614 ± 0.489	**0.743 ± 0.439**	0.594 ± 0.494
	entr	0.554 ± 0.500	**0.941 ± 0.238**	0.564 ± 0.498
	exp	0.584 ± 0.495	0.703 ± 0.459	0.614 ± 0.489
	max	0.564 ± 0.498	0.733 ± 0.445	0.554 ± 0.500
	med	0.535 ± 0.501	0.584 ± 0.495	0.574 ± 0.497
	min	**1.000**	0.500 ± 0.522	0.580 ± 0.499

FashionMNIST

Metric	Aggr.	normal	robust	scaling
Localisation	avg	0.940 ± 0.239	0.910 ± 0.288	0.950 ± 0.219
	entr	0.980 ± 0.141	**0.990 ± 0.100**	**1.000 ± 0.000**
	exp	0.950 ± 0.219	0.930 ± 0.256	0.950 ± 0.219
	max	0.940 ± 0.239	0.930 ± 0.256	0.950 ± 0.219
	med	0.960 ± 0.197	**0.990 ± 0.100**	0.950 ± 0.219
	min	0.966 ± 0.183	0.960 ± 0.198	0.980 ± 0.141

Metric	Method	ImageNet results
Localisation	autoweighted	0.760 ± 0.429
	supervisedXAI_auto	**0.810 ± 0.394**
	supervisedXAI_no	0.760 ± 0.429

Metric	Method	CIFAR results
Localisation	autoweighted	0.614 ± 0.489
	supervisedXAI_auto	0.683 ± 0.468
	supervisedXAI_no	0.653 ± 0.478

Metric	Method	FashionMNIST results
Localisation	autoweighted	**1.000 ± 0.000**
	supervisedXAI_auto	**0.990 ± 0.100**
	supervisedXAI_no	**0.990 ± 0.100**

Faithfulness

Uses Pixel-Flipping to test how important features impact prediction

ImageNet

Metric	Aggr.	normal	robust	scaling
Faithfulness	avg	0.023 ± 0.044	0.061 ± 0.066	0.023 ± 0.040
	entr	0.024 ± 0.038	0.052 ± 0.066	**0.022 ± 0.041**
	exp	0.026 ± 0.046	0.077 ± 0.082	0.024 ± 0.042
	max	**0.022 ± 0.041**	0.064 ± 0.069	0.022 ± 0.042
	med	**0.021 ± 0.032**	**0.021 ± 0.031**	0.025 ± 0.040
	min	0.058 ± 0.104	0.072 ± 0.115	0.078 ± 0.131

CIFAR

Metric	Aggr.	normal	robust	scaling
Faithfulness	avg	0.059 ± 0.110	0.151 ± 0.124	0.056 ± 0.107
	entr	0.072 ± 0.107	0.136 ± 0.137	0.083 ± 0.114
	exp	0.076 ± 0.122	0.162 ± 0.150	0.069 ± 0.110
	max	0.087 ± 0.115	0.168 ± 0.125	0.097 ± 0.127
	med	**0.042 ± 0.074**	**0.040 ± 0.063**	0.049 ± 0.084
	min	**0.027 ± 0.025**	0.088 ± 0.103	0.079 ± 0.120

FashionMNIST

Metric	Aggr.	normal	robust	scaling
Faithfulness	avg	**0.019 ± 0.061**	0.028 ± 0.118	**0.020 ± 0.068**
	entr	0.451 ± 0.283	0.132 ± 0.197	0.597 ± 0.280
	exp	0.320 ± 0.273	0.239 ± 0.203	0.274 ± 0.260
	max	0.059 ± 0.127	0.054 ± 0.108	0.122 ± 0.172
	med	**0.012 ± 0.035**	0.025 ± 0.102	0.021 ± 0.095
	min	0.055 ± 0.146	0.052 ± 0.148	0.041 ± 0.118

Metric	Method	ImageNet results
Faithfulness	autoweighted	0.024 ± 0.045
	supervisedXAI_auto	0.102 ± 0.114
	supervisedXAI_no	0.104 ± 0.115

Metric	Method	CIFAR results
Faithfulness	autoweighted	0.052 ± 0.110
	supervisedXAI_auto	0.087 ± 0.105
	supervisedXAI_no	0.089 ± 0.105

Metric	Method	FashionMNIST results
Faithfulness	autoweighted	0.035 ± 0.100
	supervisedXAI_auto	0.437 ± 0.366
	supervisedXAI_no	0.436 ± 0.365

Randomization

Uses Random Logit to check if explanations hold under model randomness.

ImageNet

Metric	Aggr.	normal	robust	scaling
Randomisation	avg	**0.107 ± 0.091**	0.344 ± 0.108	0.167 ± 0.110
	entr	0.397 ± 0.044	0.317 ± 0.040	0.335 ± 0.054
	exp	0.789 ± 0.098	0.546 ± 0.086	0.772 ± 0.099
	max	0.590 ± 0.084	0.480 ± 0.227	0.582 ± 0.095
	med	**0.139 ± 0.077**	0.151 ± 0.101	**0.142 ± 0.091**
	min	0.631 ± 0.091	0.503 ± 0.256	0.599 ± 0.092

CIFAR

Metric	Aggr.	normal	robust	scaling
Randomisation	avg	**0.029 ± 0.046**	0.147 ± 0.070	0.070 ± 0.080
	entr	0.201 ± 0.047	0.111 ± 0.073	0.151 ± 0.073
	exp	0.381 ± 0.112	0.284 ± 0.085	0.377 ± 0.110
	max	0.176 ± 0.065	0.304 ± 0.170	0.206 ± 0.079
	med	**0.047 ± 0.043**	0.058 ± 0.049	**0.056 ± 0.056**
	min	0.140	0.373 ± 0.159	0.155 ± 0.072

FashionMNIST

Metric	Aggr.	normal	robust	scaling
Randomisation	avg	0.127 ± 0.088	0.347 ± 0.219	**0.120 ± 0.109**
	entr	0.244 ± 0.101	0.166 ± 0.090	0.134 ± 0.144
	exp	0.610 ± 0.176	0.626 ± 0.280	0.601 ± 0.193
	max	0.150 ± 0.088	0.245 ± 0.198	0.263 ± 0.122
	med	0.220 ± 0.165	0.263 ± 0.166	0.469 ± 0.166
	min	**0.000 ± 0.133**	0.223 ± 0.201	**0.006 ± 0.109**

Metric	Method	ImageNet results
Randomisation	autoweighted	0.344 ± 0.121
	supervisedXAI_auto	0.735 ± 0.084
	supervisedXAI_no	0.757 ± 0.088

Metric	Method	CIFAR results
Randomisation	autoweighted	**0.050 ± 0.052**
	supervisedXAI_auto	0.181 ± 0.193
	supervisedXAI_no	0.180 ± 0.195

Metric	Method	FashionMNIST results
Randomisation	autoweighted	**0.008 ± 0.154**
	supervisedXAI_auto	0.204 ± 0.243
	supervisedXAI_no	0.201 ± 0.244

Robustness

Uses Local Lipschitz Estimation to measure stability to small input changes.

ImageNet

Metric	Aggr.	normal	robust	scaling
Robustness	avg	0.325 ± 0.082	1.064 ± 0.152	0.314 ± 0.077
	entr	0.468 ± 0.039	1.443 ± 0.102	0.497 ± 0.052
	exp	**0.096 ± 0.033**	0.778 ± 0.115	**0.103 ± 0.036**
	max	0.207 ± 0.055	0.862 ± 0.119	0.218 ± 0.062
	med	0.403 ± 0.094	0.370 ± 0.091	0.379 ± 0.095
	min	**0.184 ± 0.054**	0.816 ± 0.112	0.189 ± 0.050

CIFAR

Metric	Aggr.	normal	robust	scaling
Robustness	avg	0.825 ± 0.147	1.163 ± 0.107	0.816 ± 0.241
	entr	0.697 ± 0.052	1.575 ± 0.096	0.779 ± 0.144
	exp	**0.435 ± 0.075**	0.843 ± 0.077	**0.457 ± 0.093**
	max	0.627 ± 0.075	0.942 ± 0.086	0.654 ± 0.177
	med	0.836 ± 0.120	0.803 ± 0.124	0.800 ± 0.151
	min	**0.512**	0.904 ± 0.014	0.687 ± 0.119

FashionMNIST

Metric	Aggr.	normal	robust	scaling
Robustness	avg	0.198 ± 0.053	0.494 ± 0.405	0.187 ± 0.048
	entr	0.455 ± 0.072	1.106 ± 0.233	0.390 ± 0.092
	exp	**0.116 ± 0.051**	0.316 ± 0.291	**0.115 ± 0.046**
	max	0.379 ± 0.081	0.480 ± 0.306	0.309 ± 0.095
	med	0.139 ± 0.039	0.257 ± 0.088	**0.121 ± 0.038**
	min	0.450 ± 0.101	0.466 ± 0.319	0.489 ± 0.117

Metric	Method	ImageNet results
Robustness	autoweighted	0.249 ± 0.058
	supervisedXAI_auto	0.227 ± 0.077
	supervisedXAI_no	0.217 ± 0.077

Metric	Method	CIFAR results
Robustness	autoweighted	0.690 ± 0.103
	supervisedXAI_auto	0.772 ± 0.335
	supervisedXAI_no	0.764 ± 0.337

Metric	Method	FashionMNIST results
Robustness	autoweighted	0.270 ± 0.101
	supervisedXAI_auto	0.203 ± 0.115
	supervisedXAI_no	0.207 ± 0.121

Method ranking

Method	Faithfullness	Complexity	Localization	Randomisation	Robustness	Score
NormEnsembleXAI entr normal	1.000	0.900	1.000	0.385	0.706	3.990
NormEnsembleXAI entr scaling	1.000	0.800	0.000	0.000	0.706	2.506
NormEnsembleXAI max abs normal	0.538	0.900	0.000	0.615	0.118	2.171
NormEnsembleXAI exp normal	0.538	0.500	0.000	1.000	0.000	2.038
NormEnsembleXAI avg robust	0.000	0.200	0.333	0.462	1.000	1.995
NormEnsembleXAI max abs scaling	0.462	0.900	0.000	0.615	0.000	1.977
NormEnsembleXAI exp scaling	0.538	0.500	0.000	0.923	0.000	1.962
NormEnsembleXAI exp robust	0.538	0.000	0.000	0.769	0.647	1.955
NormEnsembleXAI entr robust	0.231	0.500	0.000	0.000	0.941	1.672
NormEnsembleXAI max abs robust	0.462	0.100	0.000	0.308	0.588	1.457
NormEnsembleXAI avg normal	0.000	1.000	0.000	0.000	0.118	1.118
NormEnsembleXAI avg scaling	0.000	0.900	0.000	0.000	0.118	1.018

Table. Scaled ANOVA test results for ensemble explanation methods.

Method ranking

Method	Faithfullness	Complexity	Localization	Randomisation	Robustness	Score
Deconvolution	0.000	1.000	0.300	1.000	0.600	2.900
NormEnsembleXAI entr normal	0.933	0.500	0.400	0.200	0.520	2.553
KernelShap	0.800	0.722	0.000	0.000	1.000	2.522
NormEnsembleXAI entr scaling	1.000	0.500	0.100	0.000	0.520	2.120
Lime	0.800	0.056	0.000	0.080	0.920	1.856
GuidedBackprop	0.000	0.500	0.200	0.920	0.120	1.740
NormEnsembleXAI max abs normal	0.533	0.611	0.100	0.360	0.080	1.684
FeatureAblation	0.000	0.556	0.000	0.280	0.800	1.636
NormEnsembleXAI exp robust	0.533	0.000	0.100	0.480	0.480	1.593
NoiseTunnel	0.000	0.167	1.000	0.320	0.080	1.567
NormEnsembleXAI exp normal	0.533	0.333	0.100	0.600	0.000	1.567
NormEnsembleXAI exp scaling	0.533	0.333	0.100	0.560	0.000	1.527
NormEnsembleXAI max abs scaling	0.467	0.556	0.100	0.320	0.000	1.442
NormEnsembleXAI avg robust	0.000	0.111	0.200	0.280	0.760	1.351
NormEnsembleXAI max abs robust	0.467	0.056	0.100	0.200	0.440	1.262
NormEnsembleXAI entr robust	0.200	0.333	0.000	0.000	0.720	1.253

Table. Scaled ANOVA test results for all explanation methods.

XAI ensembling Python library

Library available under a link:

https://github.com/Hryniewska/EnsembleXAI

Example of Workflow

1. Choose Model to Explain
2. Read an Image to Explain
3. Generate Explanations
4. Stack Explanations
5. (Optional) Normalize Explanations
6. Ensemble Your Explanations
7. Visualize Your Results
8. (Optional) Calculate Quality Metric

1. Choose Model to Explain

First, select a pre-trained model to be explained. This example uses the `resnet18` model from the `torchvision` library:

```
from torchvision.models import resnet18

# Load a pretrained model
model = resnet18(weights='IMAGENET1K_V1')
model = model.eval()
```

2. Read an Image to Explain

Next, read in an image and preprocess it to match the model's input requirements. The image will be resized to 224x224 pixels, which is the expected input size for the `resnet18` model.

```
import cv2

# Load and preprocess the image
img = cv2.imread("path_to_image.jpg")  # Replace with the path to your image
img = cv2.resize(img, (224, 224))      # Resize the image to fit the model input size
```

3. Generate Explanations

Now, use the Captum library to generate explanations for the image. You can also use any other library, but the output type and shape should be the same as in the Captum library. This example demonstrates the usage of three different methods: Integrated Gradients, GradientShap, and Saliency.

```
import torch
from captum.attr import IntegratedGradients, GradientShap, Saliency

# Prepare the input for the model
inputs = torch.tensor(img).unsqueeze(0).float()  # Add batch dimension and convert to tensor

# Generate explanations using different methods
ig = IntegratedGradients(model).attribute(inputs, target=3)
gs = GradientShap(model).attribute(inputs, target=3)
sal = Saliency(model).attribute(inputs, target=3)
```

4. Stack Explanations

Stack the generated explanations to create a consolidated tensor for further processing:

```
# Stack the explanations
concatenated_explanations = torch.stack([ig, gs, sal], dim=1)
```

5. (Optional) Normalize Explanations

Normalization of explanations is recommended before using the NormEnsembleXAI method. Normalize the stacked explanations using, for example, the second_moment_normalize function to ensure they are on the same scale before ensembling:

```
from EnsembleXAI.Normalization import second_moment_normalize

# Normalize explanations using Second Moment Scaling
normalized_explanations = second_moment_normalize(concatenated_explanations)
```

6. Ensemble Your Explanations

Use the NormEnsembleXAI method with an averaging function to ensemble the explanations into a final output:

```
from EnsembleXAI.Ensemble import normEnsembleXAI

# Use NormEnsembleXAI with 'avg' aggregation function
output_normEnsembleXAI = normEnsembleXAI(normalized_explanations, aggregating_func='avg')
```

7. Visualize Your Results

Visualize the original image alongside the ensembled explanations:

```
import matplotlib.pyplot as plt
from captum.attr import visualization as viz

# Create a figure to visualize the original image and ensembled explanation
fig, ax = plt.subplots(1, 2, figsize=(12, 3))

# Display the original image
ax[0].imshow(img)
ax[0].set_axis_off()
ax[0].set_title("Original Image")

# Display the NormEnsembleXAI explanation
viz.visualize_image_attr(
    output_normEnsembleXAI.numpy().transpose(1, 2, 0), plt_fig_axis=(fig, ax[1]), use_pyplot=False, t
)
```

7. Visualize Your Results

Visualize the original image alongside the ensembled explanations:

```
import matplotlib.pyplot as plt
from captum.attr import visualization as viz

# Create a figure to visualize the original image and ensembled explanation
fig, ax = plt.subplots(1, 2, figsize=(12, 3))

# Display the original image
ax[0].imshow(img)
ax[0].set_axis_off()
ax[0].set_title("Original Image")

# Display the NormEnsembleXAI explanation
viz.visualize_image_attr(
    output_normEnsembleXAI.numpy().transpose(1, 2, 0), plt_fig_axis=(fig, ax[1]), use_pyplot=False, t
)
```

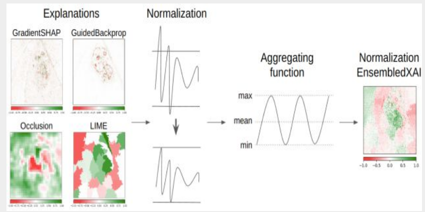
8. (Optional) Calculate Quality Metric

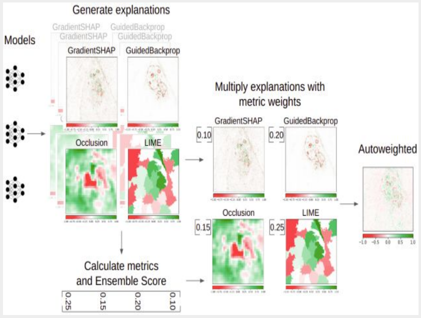
Finally, it is possible to assess the quality of the explanations in dirrent ways. Here, calculate the consistency between two explanation methods (e.g., Integrated Gradients and NormEnsembleXAI explanation) using the consistency metric from EnsembleXAI:

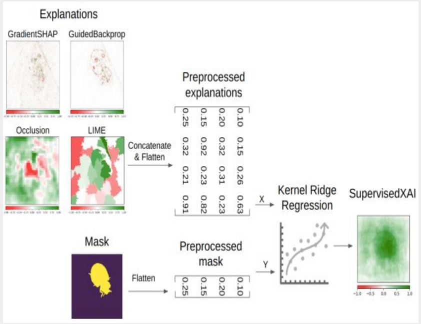
```python
from EnsembleXAI.Metrics import consistency

# Calculate the consistency between Integrated Gradients and NormEnsembleXAI explanation
consistency_score = consistency(ig, output_normEnsembleXAI)
print(f"Consistency Score between IG and NormEnsembleXAI: {consistency_score}")
```

Summary

- Significant benefits of using Ensemble XAI methods
- Aggregation method is a key factor, but normalization is important as well
- NormEnsembleXAI min might be a great method to show very salient regions,
- EnsembleXAI library is open-source and ready to use :)

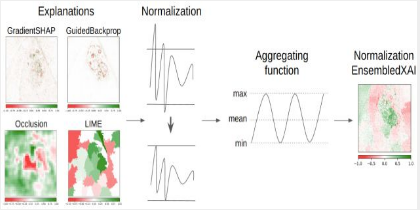




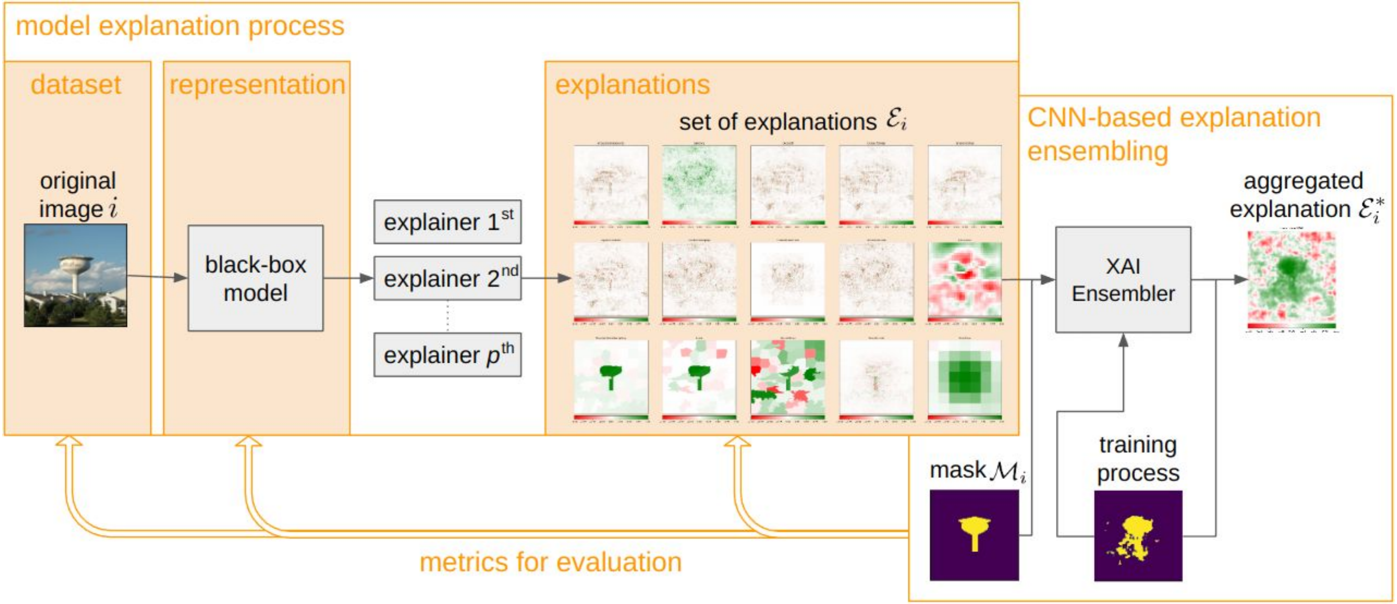


Future research

- Expanding to other types of data
- Addressing limitations of these algorithms
- Improving SupervisedXAI method



“CNN-based explanation ensembling for dataset, representation and explanations evaluation”



NORMENSEMBLEXAI: UNVEILING THE STRENGTHS AND WEAKNESSES OF XAI ENSEMBLE TECHNIQUES

Weronika Hryniewska-Guzik, Bartosz Sawicki, Przemysław Biecek
Faculty of Mathematics and Information Science
Warsaw University of Technology
Koszykowa 75, 00-662 Warsaw (Poland)
weronika.hryniewska.dokt@pw.edu.pl

ABSTRACT

This paper presents a comprehensive comparative analysis of explainable artificial intelligence (XAI) ensembling methods. Our research brings three significant contributions. Firstly, we introduce a novel ensembling method, NormEnsembleXAI, that leverages minimum, maximum, and average functions in conjunction with normalization techniques to enhance interpretability. Secondly, we offer insights into the strengths and weaknesses of XAI ensemble methods. Lastly, we provide a library, facilitating the practical implementation of XAI ensembling, thus promoting the adoption of transparent and interpretable DL models.

Thank you for your attention!