Deep learning Lab4

Explainable AI
LIME and RISE

Naween Dissanayake

Introduction

In the ever-evolving landscape of machine learning and artificial intelligence especially in neural networks, the performance of complex black box models has been remarkable. However, their very complexity sometimes results in a lack of interpretability, making it challenging to trust their predictions and understand why they make specific decisions. This opacity is a significant concern, particularly in high-stakes applications such as healthcare, finance, and autonomous systems. To address this challenge, interpretable AI techniques have emerged as a critical area of research and development. Two such techniques are LIME (Local Interpretable Model-agnostic Explanations) and RISE (Randomized Input Sampling for Explanation of Black-box Models). In this report, I delve into the implementation and application of LIME and RISE, with the aim of providing insight into the inner workings of black box models and enhancing the interpretability, transparency, and trustworthiness of AI systems.

Methodology

LIME

How lime works?

Select the Instance to Explain: First of all, start selecting a specific instance or data point for which you want an explanation of the model's prediction. This instance could be an input to the model, such as an image, text, or tabular data point in this case we are going to use images (African elephants' dataset)

Generate Perturbed Samples: LIME creates a dataset of perturbed samples based on the selected instance. These samples are generated by randomly perturbing the features of the original instance while keeping its label constant. The goal is to create a diverse set of data points that represent the local neighborhood of the instance.

Obtain Model Predictions: LIME obtains predictions for each perturbed sample from the blackbox model that we are trying to explain. It's important to note that LIME doesn't require knowledge of the internal workings of the model we only consider input and output basically the inputs are manipulated. It treats the model as a "black box" and only uses it to make predictions.

Fit an Interpretable Model: LIME fits an interpretable model, such as linear regression or decision trees, to the perturbed samples and their corresponding model predictions. This interpretable model is trained to approximate the behavior of the black-box model in the local neighborhood of the selected instance.

Interpret the Model: Once the interpretable model is trained, we can analyze it to gain insights into how the black-box model is making predictions for the selected images. For example, I can examine the coefficients in a linear regression model to understand the importance of different features.

Generate Feature Importance: LIME provides feature importance scores that indicate which features had the most influence on the model's prediction for the instance. This information helps users understand which aspects of the input data were most relevant in the decision-making process.

Visualize or Communicate the Explanation: Finally, visualizing or communicating the explanation to stakeholders. This may involve highlighting important features, showing feature importance scores, or other methods to make the model's behavior more understandable.

RISE

How lime works?

Input Sampling: RISE begins by selecting an image that we want to explain. This image could be an input for a deep learning model, such as an image classification task for example resnet50, vgg
Creating Random Masks: RISE generates a set of random binary masks. These masks have the same dimensions as the input image. Each mask is created by randomly sampling pixels, effectively covering parts of the image. The randomness is a crucial aspect of RISE, as it ensures diversity in the explanations.
Applying Masks to the Image: The random masks are applied to the selected image. This process masks out regions of the image and simulates the effect of occlusion or hiding certain areas. **Model Predictions**: For each masked image (the original image with certain regions covered), RISE obtains model predictions from the black-box model. The goal is to observe how different parts of the image affect the model's output.
Scoring Masks: RISE assigns a score to each mask based on the change in model predictions when that mask is applied. If the model's predictions change significantly when a particular area is masked, it suggests that the masked region is crucial for the model's decision.
Creating Heatmaps: To visualize the explanation, RISE generates a heatmap. This heatmap highlights the importance of different regions in the image by aggregating the scores assigned to the masks. Areas with higher scores are considered more influential in the model's prediction.
Visual Explanation: The final output of RISE is a heatmap that you can overlay on the original image. The heatmap shows which parts of the image had the most impact on the model's decision.

RISE is particularly valuable for understanding why a deep learning model made a specific classification decision for an image. By generating heatmaps that highlight important regions, it helps users gain insights into which visual features or patterns influenced the model's output.

Compared to LIME, RISE is specifically designed for image data and leverages random masking to explore the significance of different image regions. It doesn't require access to the model's internal parameters and can be applied to a wide range of image classification tasks.

Result and Implementation

LIME

Exercise_ According to the Exception and Resnet black box

Question 1

1. As you can see in the previous cell, many parameters have to set manually according to your model and data. Try to identify the right combination of parameters to explain the prediction of the given image (here an African elephant).

top_labels: This parameter determines how many top labels (classes) to consider when generating explanations. In this model for classification top_label parameter has to be 1 since it has one class to classify.

hide_color: This parameter sets the color used to hide parts of the image when generating neighboring samples. It can be an RGB color or "None." Using an RGB color can be helpful when you have a specific idea of what to hide. Using "None" will result in the average color of superpixels being used. The choice depends on whether you have prior knowledge of what regions might be important. According to my experiment trial and error is the solution to selecting this parameter. **num_lime_features:** This parameter determines the size of the explanation. It controls the number of groups of features considered during the explanation generation. Smaller values may produce more concise explanations, while larger values may capture more fine-grained details. The choice depends on your preference for explanation granularity.

num_samples: This parameter defines the number of perturbed samples used to generate explanations. Increasing the number of samples can lead to more stable and reliable explanations. However, it also increases computation time.

Since I have limited time and computational resources rendering parameters won't be considered in this case and it will not affect to the result only for better visualization.

Question 2

1. Now consider another image of an African elephant Is your parameter setting still appropriate?

Fig:01

Fig:01

Fig:01: top_labels =1, hide color = [0,0,0], num_lime_features = 100000 unsampled = 5000

I tried to experiment with two images from the African elephant's class using given parameters by understanding some information about XAI and I realized since we have binary classification we used 1 as a top_label parameter because we do not have multiclass in the input image. We can incorporate the number of lime features but the given default parameter is well enough and it gives a good explanation about the model. 5000 perturbation number is also well enough to give a reliable explanation.

Fig:02: top_labels =1, hide color = [255,0,0],num_lime_features= 105000 unsampled=10000

I have changed the color parameter to red, increased the lime features a bit, and the sample size as well but the results weren't nice which means not explainable like Fig 1 I assume because the red color wasn't in the input image and there is no such region to hide then I tired blue and green as well to make sure and results are following in the figures.

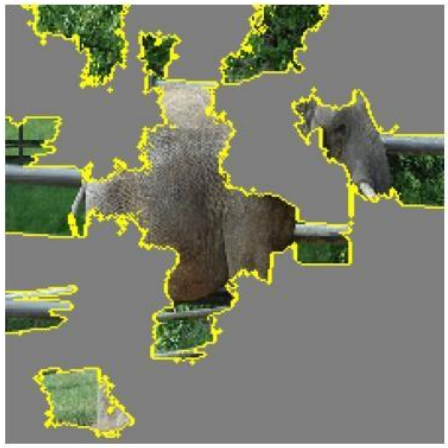
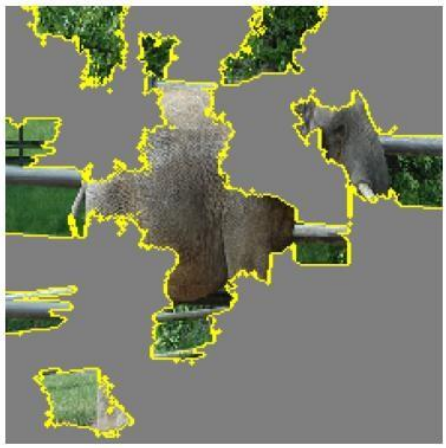
blue and green(left to right)

Fig:03: top_labels =1, hide color = [0,0,255],num_lime_features= 105000 unsampled=10000
Fig:03: top_labels =1, hide color = [0,255,0],num_lime_features= 105000 unsampled=10000

No difference in blue and green but more reasonably explained results can be seen once the color into blue and green.
According to my trial and error, the best combination would be the default parameters: top_labels =1, hide color = [0,0,0], num_lime_features = 100000 unsampled = 5000

Question 3

We now consider images from another class to assess whether the identified setting is appropriate for another class. You can find a black bear image here.
What can you conclude?

Black bear

```
Top-3 predicted classes :
        sloth_bear: 0.51714736
        American_black_bear: 0.1375108
        howler_monkey: 0.021555936
```


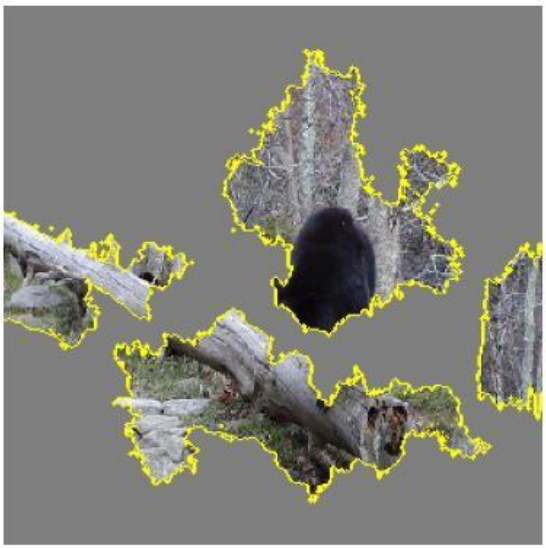

Fig:03: top_labels =1, hide color = [0,0,0], num_lime_features = 100000 unsampled = 5000 I believe the results are less accurate, and not the same as African class. Need to play with the parameters to find the best set of parameters.

Fig:04: top_labels =3, hide color = [255,0,0], num_lime_features = 105000 unsampled = 10000

Now I change by expecting accurate results but the top label is 3 it is obvious to get a less accurate result and it explains well which means the model does not give any inappropriate results.

Question 4

Here, we want to answer the following question: If we change the model, would the parameter setting still be appropriate? In other words, is the parameter setting more related to the data and tasks than it is to the model architecture?

Results from resent

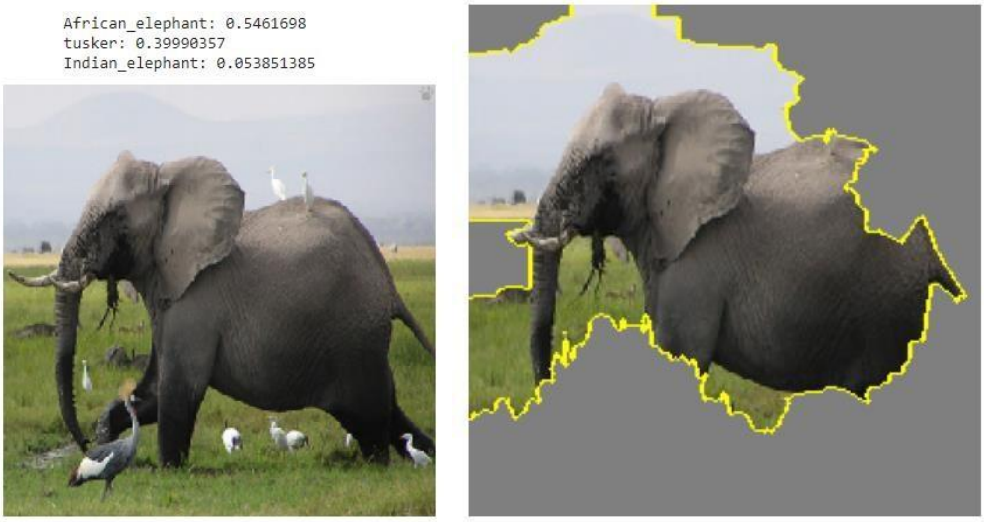

Fig:05: top_labels =1, hide color = [0,0,0], num_lime_features = 100000 unsampled = 5000

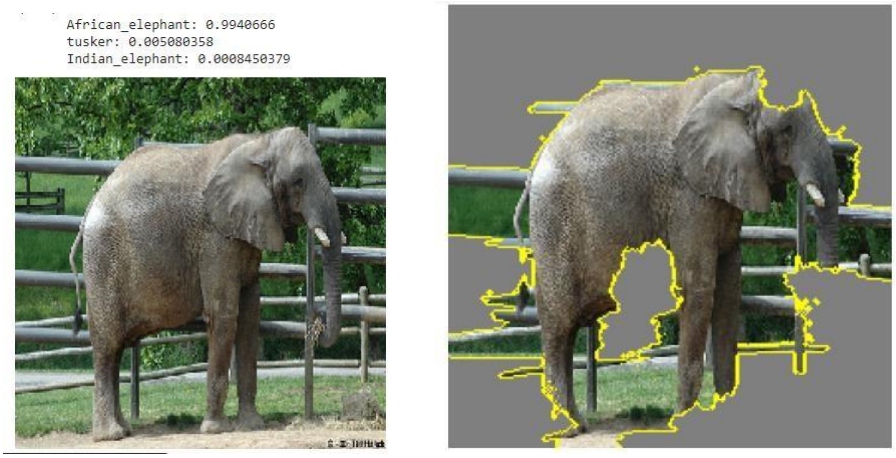

Fig:06: top_labels =1, hide color = [0,0,0], num_lime_features = 100000 unsampled = 5000

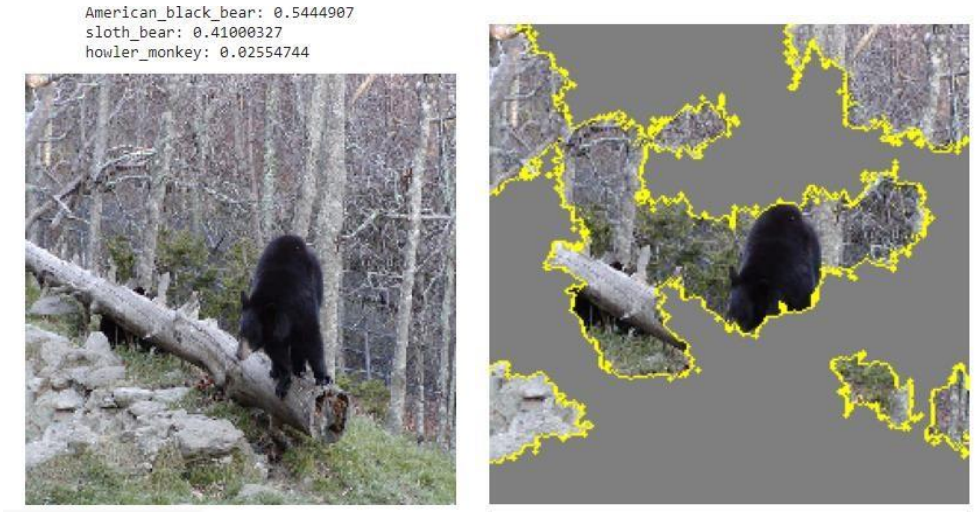

Fig:07: top_labels =1, hide color = [255,0,0], num_lime_features = 105000 unsampled = 10000

After carefully seeing the results images from the exception black box model can be more explainable than the Resnet in every situation.

Model Differences: ResNet and Xception have distinct architectures, and they capture different features and patterns from the input data. These architectural differences alone can lead to variations in the explanations generated by LIME the Fig 1 to 7.

Learned Features: The two models were trained on different datasets and learned different sets of features. As a result, they might prioritize different aspects of the input image when making predictions. LIME's explanations will reflect these differences.

Complexity: The complexity of the models also plays a role. Xception, for example, employs depthwise separable convolutions, which can capture fine-grained details differently compared to ResNet's residual blocks.

Prediction Mechanisms: ResNet and Xception may have different decision boundaries and prediction mechanisms. They might assign varying importance to different features or use different combinations of learned features in making predictions.

RISE implementation

- N specifies the number of masks to generate.
- s determines the size of the grid (s x s) used for generating masks.
- p1 is the probability that a cell in the grid is set to 1.

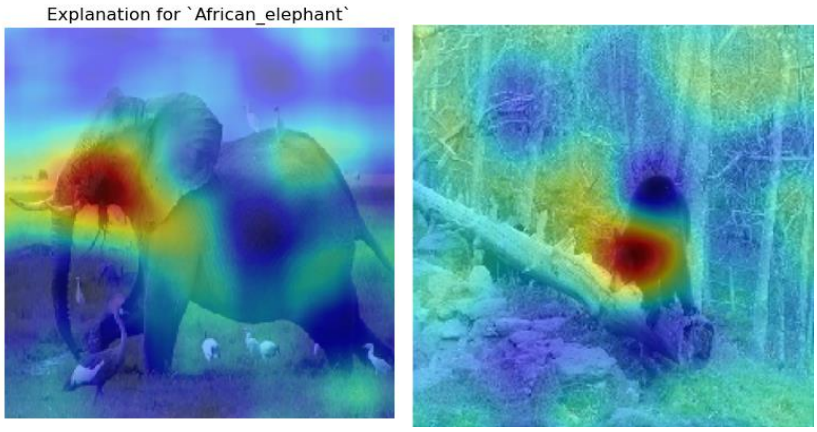

Fig 08: RISE results N = 2000, s = 8, p1 = 0.5 / overlay

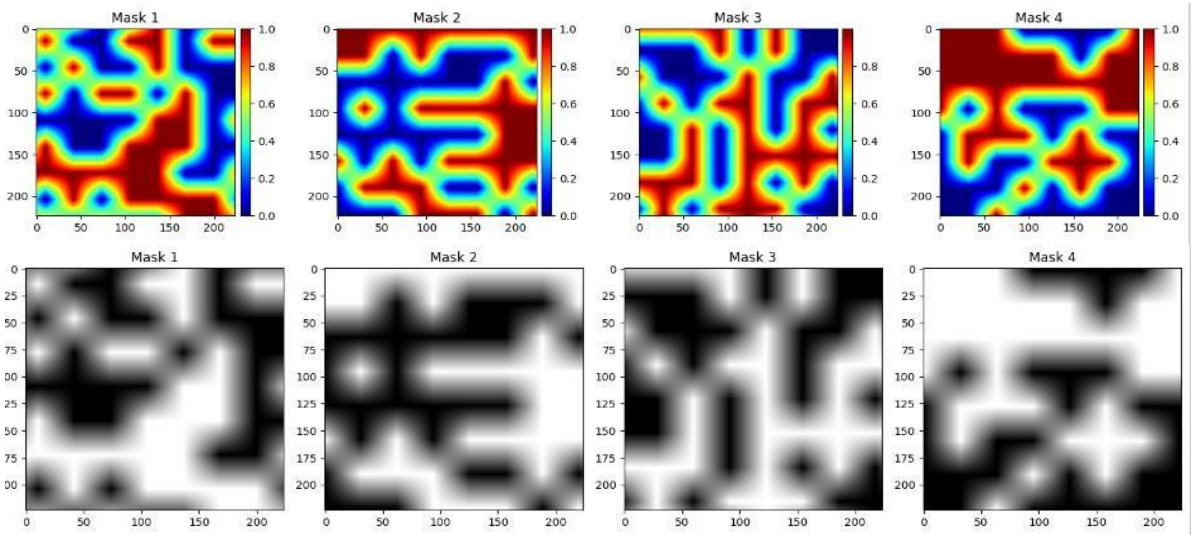

Fig09: Some examples for mask / binary [0,1]

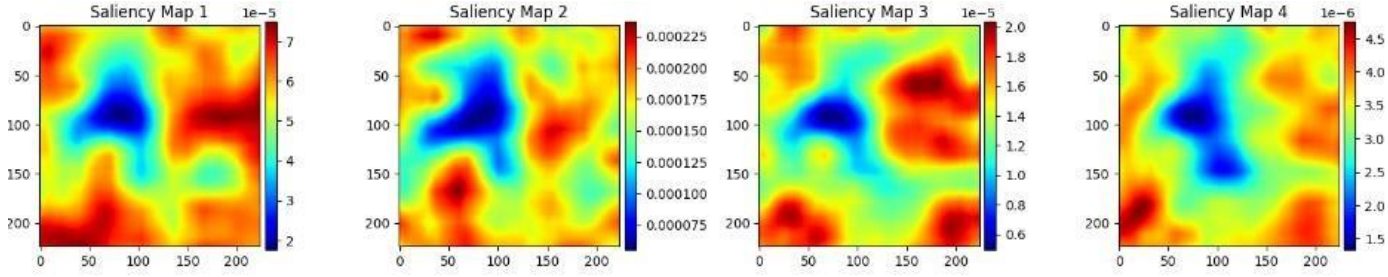

Fig10: Some examples for sal/saliency maps

Discussion

The Evolving landscape of machine learning, artificial intelligence and Deep learning, especially in neural networks, complex black box models have delivered remarkable performance. However, their inherent complexity often leads to a lack of interpretability, which poses challenges in trusting their predictions and comprehending the reasoning behind their decisions. This opacity is a significant concern, particularly in high-stakes applications like healthcare, finance, and autonomous systems. To address this challenge, interpretable AI techniques have emerged as a critical area of research and development. In this task, we learned two techniques LIME (Local Interpretable Model-agnostic Explanations) and RISE (Randomized Input Sampling for Explanation of Black-box Models), which have become valuable tools in enhancing the interpretability, transparency, and trustworthiness of AI systems.

LIME begins by selecting a specific instance or data point for which you want an explanation of the model's prediction. Perturbed samples are generated by randomly altering the features of the original instance while keeping its label constant. Model predictions are obtained for each perturbed sample from the black-box model, and an interpretable model is fitted to the perturbed samples and their corresponding model predictions. This interpretable model helps analyze and understand the black-box model's behavior, providing feature importance scores to highlight relevant input data aspects. Finally, the explanation is visualized or communicated to stakeholders.

RISE works specifically with image data and follows a different approach. It begins by selecting an image for explanation and then generates a set of random binary masks. These masks have the same dimensions as the input image and are created by randomly sampling pixels to simulate the effect of occlusion. Model predictions are obtained for each masked image, and the masks are scored based on their impact on the model's predictions. A heatmap is generated to overlay on the original image, highlighting the regions that most influenced the model's decision.

The choice of parameters for LIME is critical for obtaining meaningful explanations. The number of top labels, hide color, the number of LIME features, and the number of samples are all parameters that should be fine-tuned. For different classes and models, these parameters may need adjustment. The appropriateness of parameter settings may also vary with different classes and models. As demonstrated in the report, the parameter settings should be considered in conjunction with the specific class and model being analyzed.

When evaluating different classes, it becomes evident that parameter settings need to be adjusted accordingly. Each class may have unique characteristics and features that impact the parameter settings. For instance, the top label parameter should match the number of classes in the data, and the hide color parameter may need to be adjusted to better highlight influential regions.
Furthermore, the choice of model can also impact the parameter settings. Different models have varying architectures, learned features, complexities, and decision mechanisms. These differences can influence the effectiveness of LIME's explanations. As shown in the report, the ResNet and Xception models delivered distinct results with the same parameter settings, indicating that the parameter setting may be more related to the data and tasks than the model architecture. The RISE implementation provides insights into the importance of the number of masks (N), the grid size (s), and the probability (p1). These parameters affect the quality of explanations and may need to be adjusted based on the specific context and requirements. The overlay of heatmaps onto images highlights regions of interest and assists in understanding the model's decision.

Reference

[LIME] Ribeiro, M. T., Singh, S., & Guestrin, C. (2016, August). " Why should i trust you?" Explaining the predictions of any classifier. In Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining (pp. 1135-1144)

[RISE] Petsiuk V., Das A. and Saenko K., Rise: Randomized input sampling for explanation of black-box models.arXiv preprint arXiv:1806.07421, 2018