Deep Learning for Computer Vision
Lab 04

Bolutife Atoki
bolutife-oluwabunmi.atoki@etu.u-bordeaux.fr
Université de Bordeaux

Abstract

This paper contains my solution and results of the fourth lab session, resources used are listed at the end of the paper

1 Introduction

The aim of this session was to implement two Blackbox explainer models; LIME and RISE. Blackbox explainer models are models that try to show the reasons behind a model's prediction or classification by highlighting the datapoints (in case of images; pixels) that are directly responsible for the model's output, without having access to the Model's Architecture or trained weights, such that the explainer model uses on the the input image as well as the classification score and the predicted class. The blackbox explainer methods considered in this lab are Feature Attribution methods that produce saliency maps to highlight interesting parts in the input image. They work by performing perturbations (changes) to the input images and then using these perturbed samples to compute the importance of each feature (pixel or group of pixels called regions or super-pixels) of the input sample, with the intuition really relevant feature combinations when perturbed, should have more perturbed predictions.

2 Methodology

2.1 Explainer models

Explainre models are models that provide insights in a form people are used to and understand, into the behavior and decision-making process of a trained model, with the aim of making the predictions or decisions of the trained model less complex, less interpretable models more transparent and understandable to humans The Isoline representation of a saliency map involves contouring the values in the saliency map using isolines or contour lines., such that these isolines connect points with the same saliency level. The areas with higher saliency values are represented by contour lines that are closer together, creating dense lines, while areas with lower saliency values have contour lines that are farther apart. Also, the colours of the contour lines can be Color / Heat mapped based on the corresponding value of that pixel position in the saliency map. Hence, with this represented density information, viewers can quickly identify which regions stand out the most based on the concentration of contour lines. The resulting Isolines can then further be blended / overlaid on the input image to highlight portions in the image having the most gaze points.

2.2 Blackbox models

Black-box models make predictions based such that the internal workings of these models are not readily interpretable or understandable by humans. I.e the logic, rules, and processes used by black-box models are not transparent and are complex and inscrutable.

2.2.1 LIME

LIME is a Feature attribution method which means that the method computes for each feature of an input sample its importance in the prediction. To do so, LIME uses perturbations of the considered sample and their corresponding (and perturbated) predictions to identify features of importance.

2.2.2 RISE

Randomized Input Sampling for Explanation (RISE) computes a saliency map emphasizing the features of interest for a given model prediction. It allows to identify the features that the model considers as important for the prediction of a given class (not necessarily the predicted one). The

main idea of RISE is to generate a large number of random (binary) masks in low resolution and then to upscale them to the dimension of the input image (the upscaled mask values are therefore in [0, 1]). For each of these masks, a prediction is done by the model which produces a score for the considered class. While many methods consider the masked features as responsible for the perturbed prediction, the authors of RISE make the opposite assumption. In RISE, unmasked (or partially masked) features are considered as responsible for the prediction, and, instead of considering how far the perturbed prediction is from the initial prediction, RISE considers that important features are kept if and only if the perturbed prediction score is high.

3 Dataset

The dataset used is composed of images of two classes; **African Elephant** and **Black bear** in various environments with each having 51 images.

4 Implementation

The code implementation is made up of multiple script files having a main script file (**main.py**) to be called. The other files include:

1. **representations.py**: which holds the various representation functions used in the project.
    - **represent_heatmap()**:
      This takes in as parameters the saliency map and the desired colormap to be used , and returns the input saliency map represented as a heatmapped image according to the specified color map.
    - **represent_heatmap_overlaid()**:
      This takes in as parameters the saliency map, the image and the desired colormap to be used , and then applies the specified colourmap to the saliency map to get a heatmap (by calling the **represent_heatmap()**) function. The heatmapped saliency map is then blended with the input image by an alpha parameter that controls the blending, and finally returns this blended image.
2. **utils.py:**
   which holds the helper function used in the project.
    - **normalise()**:
      It takes in a matrix and returns its normalized version (to range 0 - 1) by dividing each element in it by its max value.
    - **grid_layout()**:
      It takes a list of images and a list of string titles as inputs, and plots & shows these images in a grid and saves the grid image.
    - **generate_masks():**
      The function takes an input image and generates a set of random masks to perturb the image. These masks are created based on specified parameters, including the number of masks (n_masks), mask size (mask_size), and a threshold value (threshold). It is designed to create a set of random masks for perturbing an input image. This process involves resizing the original image to a square shape, ensuring consistency in dimensions. The user specifies the number of masks to generate (n_masks), the size of each mask (mask_size), and a threshold value (threshold) to determine the mask's pixel values. Random masks are generated with binary values, where the threshold decides whether a pixel in the mask should be set to 1 or 0. For each mask, the function randomly selects an origin point within the dimensions of an upsampled mask to ensure proper positioning during the perturbation process. These generated masks are then applied to the resized image, resulting in perturbed images where different regions of the image are selectively perturbed based on the mask patterns. The function returns a list containing both the perturbed images and the corresponding masks.
    - **make_prediction():**
      It takes in as parameters a pretrained model, the model's name, an array of perturbed_images (having dimensions batch_size, width, height, channels), and the class name for the object in the input image and makes predictions using the pre-trained deep learning model (either Xception or ResNet) on the set of perturbed images. It returns prediction scores and labels for the specified class.

- **make_classifier():**
  It takes in a string having the name of the blackbox model, creates and returns a pre-trained deep learning classifier model based on the specified model (Xception or ResNet).
- **calculate_saliency_map():**
  The function takes a list of prediction scores for perturbed images and an array of masks that were used to perturb the images, and calculates a saliency map which is derived by combining the prediction scores obtained from a model's analysis of perturbed images with the corresponding perturbing masks. By fusing the scores with their respective masks, the function assigns higher weights to regions considered more significant by the model during the prediction process. It then normalizes the mask to ensure that its values fall within the range of 0 to 1.

3. **main.py**
   It is the main python script for this project and it can be called directly from the command line, it has a main function which accepts the following arguments when called using argument parser;

   - path to test image
   - Index of the object in the image
   - The name of the model to be used
   - the display type; **grid** which shows a grid of all results or **singles** which shows the results individually.

   An example format of calling the function is shown below

```
python main.py --test_image_path='ILSVRC2012_val_00001177.JPEG' ...
    --test_image_index=0 --model_name='ResNet' --display_type='grid'
```

   The main function does all the necessary imports of the utils, representations and objects files and then calls the functions for making the masks and pertubed images, loading the model, making predictions to obtain the scores, obtaining the saliency map, representing it with a heatmap, overlaying the heatmap on the input image and finally displaying the obtained images either as a grid or individually, depending on the type specified with the grid display set as default.

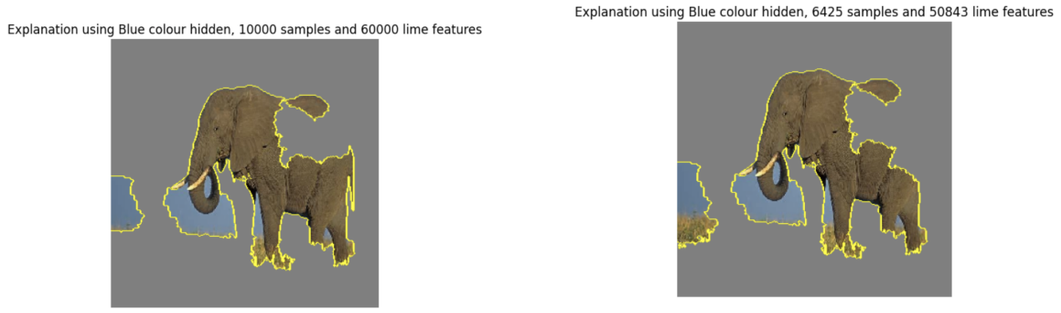
5 Results and Discussion

The results and subsequent discussions of the implementations are split into two; LIME and RISE

5.1 LIME Results

The Lime notebook was run and the provided questions are answered below:

1. **Question 1:**



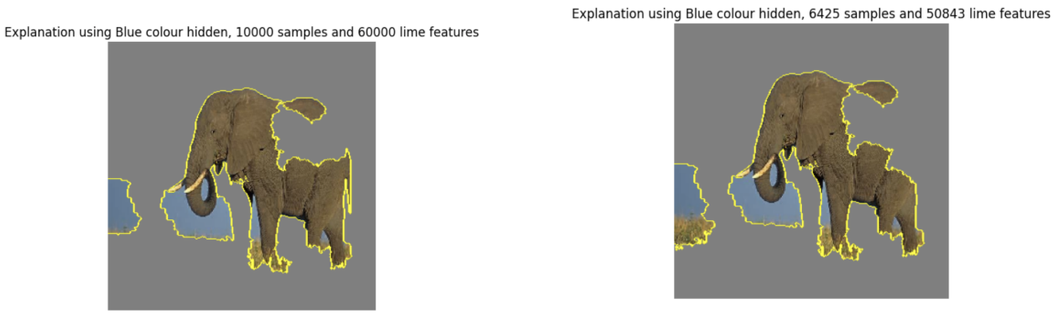
(a) Explanation result using chosen parameter values (b) Explanation result using random parameter values

Figure 1: Comparison of explanation using different parameter values

Observing both images, it is seen that by increasing the values to a certain amount, there were only slight differences (in the back side of the elephant). This could be because of:

- The model being explained is stable or robust with respect to different settings of the Lime explainer, such that even with different parameter configurations, the model's predictions for the same image remain consistent.
- It is also possible that the original explanation was quite comprehensive and accurate
- The fact that only a few additional pixels in the back side of the elephant were highlighted suggests that this part of the image has a higher influence on the model's prediction. The remaining pixels are not as sensitive to perturbations in the context of the model's decision.

2. **Question 2:**



(a) Explanation result using chosen parameter values (b) Explanation result using random parameter values

Figure 2: Comparison of explanation using different parameter values

The explanation result using the parameters on the new image is still okay (could be better) as it still captures some useful features of the object (the elephant) in the image like its tusks, trunk, and hind legs which would be useful for classification. However, from the second image where different parameter values were used, it is seen that a bit more information is added like the right ear of the elephant, and more of its frontal leg, which also would be information that the model would used for its classification, hence why I said the explanation result using the parameters are okay (as some information is still captured) by could be better and capture more information using some other parameters like the ones used in the right image.

3. **Question 3:**

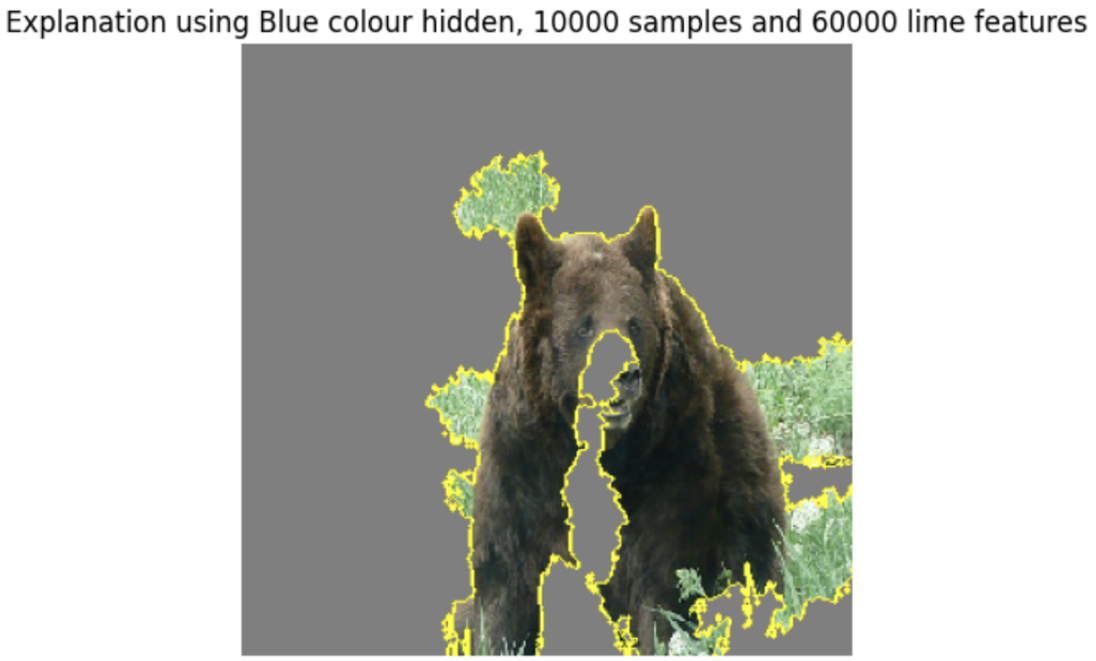
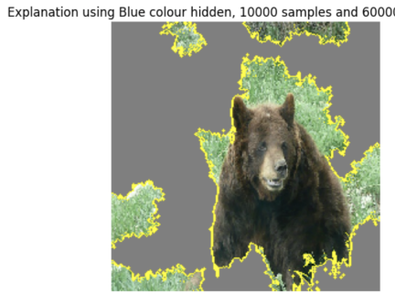


Figure 3: Comparison of explanation using same parameter values on image of different class

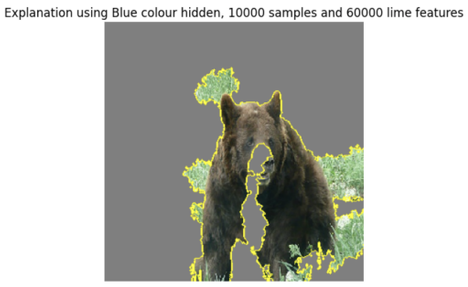
The Classfication model's prediction was Brown Bear (which is correct) and by visually observing the highlighted pixels from the LIME model, it is seen that the pixels contain mostly just the bear, meaning that the Classfication model used the features from the highlighted

region (e.g the colour, the silhoutte of the bear, certain body parts e.t.c.). Considering this information, I think this parameter also works for the LIME model for this image as its explanation (highlighted areas) contain features relevant to brown bears

4. **Question 4:**
   Using Image from Question 3



(a) Explanation result using ResNet

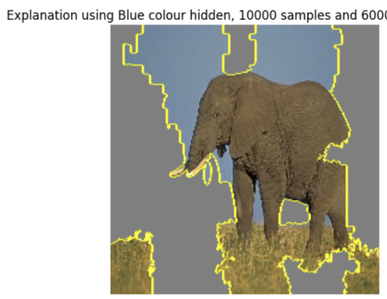


(b) Explanation result using Xception

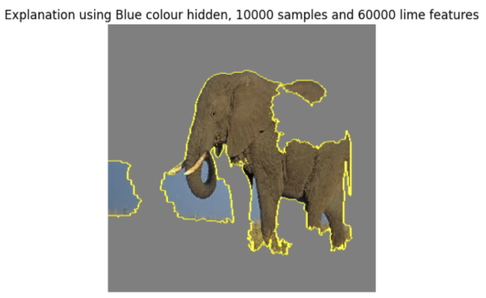
Figure 4: Comparison of explanation using same parameter values but different models

Observing the explanation result, it is observed the ResNet model also predicted Brown Bear (with a confidence of 98.5% which is correct), and also that the highlighted pixels contain the bear hence features the ResNet model must have used for prediction, however upon comparing the visual results of the LIME + Xception and LIME + ResNet, it is seen that the ResNet explanation contains a bit more distracting background information like the grass unlike the former which mostly contained juat the silhoutte of the Bear i.e more precise and useful features and infromation.

Using Image from Question 1



(a) Explanation result using ResNet



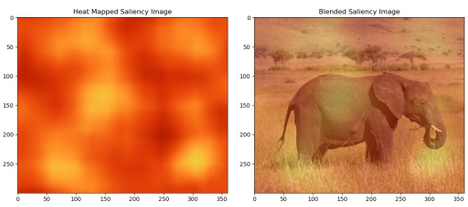
(b) Explanation result using Xception

Figure 5: Comparison of explanation using same parameter values but different models

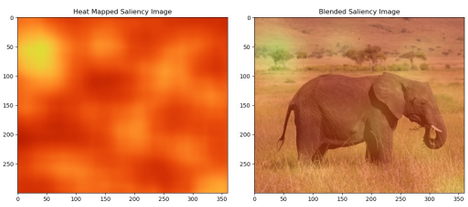
Again, Observing the explanation result, it is observed the ResNet model predicted African Elephant (with a confidence of 95% which is correct, while the Xception model predicted Tusker Elepahant with a confidence of 39% which is wrong), and also that the highlighted pixels contain the elephant hence features the ResNet model must have used for prediction, upon comparing the visual results of the LIME + Xception and LIME + ResNet, it is seen that although the former's prediction was wrong, it however still highlighted mostly the elephant's silhoutte which indicates that the Xception model identified the elephant in the image, but just wrongly matched the features extracted from the African Elephant with that of a Tusker Elephant. However, the ResNet model which correctly classified the elephant had an explanation that included more distracting information like the clouds and the grass than the Xception model.

5.2 RISE Results

5.2.1 Comparing results on same Image of African Elephant using ResNet and Xception models



(a) Explanation result using ResNet

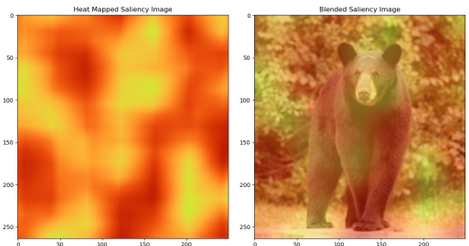


(b) Explanation result using Xception

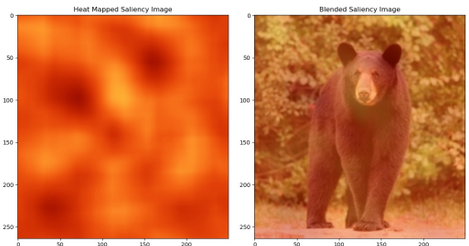
Figure 6: Comparison of explanation on African elephant image using different models

From the above figures, it is observed that the Explanation using the Resnet model highlights more areas on the elephant which contain information and features useful for prediction than with the Xception model which highlights more background portions, indicating that the explanation with the ResNet model was better.

5.2.2 Comparing results on same Image of Black bear using ResNet and Xception models



(a) Explanation result using ResNet

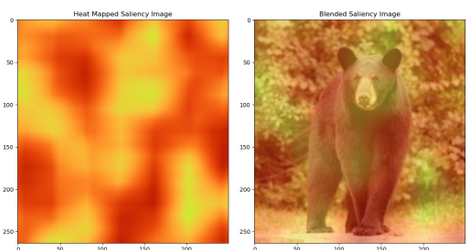


(b) Explanation result using Xception

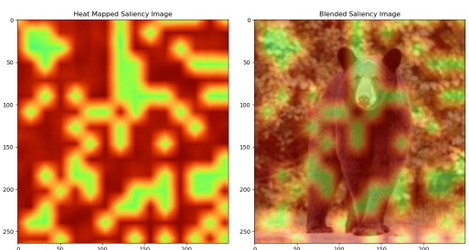
Figure 7: Comparison of explanation on Black bear image using different models

Here also it is observed that the Explanation using the Resnet model highlights more areas on the bear (like the face, ears, limbs) which contain information and features useful for prediction than with the Xception model which highlights more background portions, indicating that the explanation with the ResNet model was better.

5.2.3 Comparing results of same Image of Black bear using different mask sizes on ResNet Model



(a) Explanation result using mask size of 7

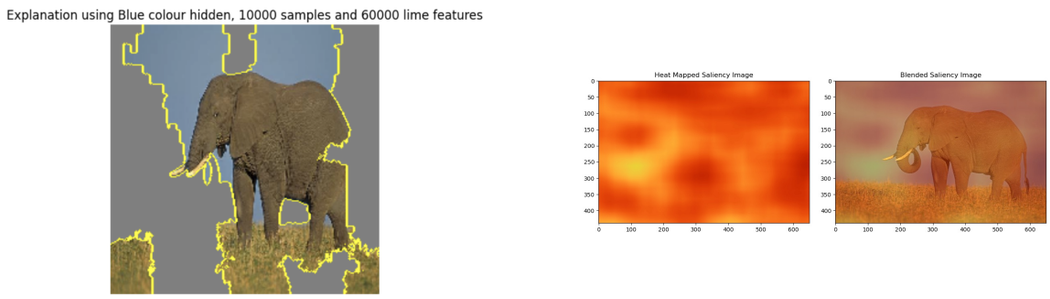


(b) Explanation result using mask size of 15

Figure 8: Comparison of explanation on Black bear image using different mask sizes

Here, it is observed that when the low res mask size is increased, the smoothing effect decreases substantially and there are more hot spots in the explanation which increases the likelihood for explainanbility errors.

5.2.4 Comparing results of same Image of African Elephant for LIME and RISE Model (using ResNet for both)



(a) Explanation result using LIME

(b) Explanation result using RISE

Figure 9: Comparison of explanation on African Elephant for LIME and RISE Model

Here, it is observed that although the RISE model highlights the trunk, back and hind legs of the elephant (which contain features), the LIME model tries to highlight the silhoutte of the elephant although this explanation is still noisy as it contains a lot of the background information as well.

6 Conclusion

From the two explainer methods implemented (LIME and RISE) and the various explanations gotten from them, it is observed that the explanation results rely heavily on the stability, generalization of blackbox model used, the number of samples and lime feature parameters used for the LIME explainer, the size of the low res mask (for the RISE Explainer) as the smaller the mask, the better (up to a certain point), as well as the input image, meaning that the two explainer models will not always generalize well for all test images.