Home Assignment 2 – Explainable AI

June 5, 2023

1 Loading a pretrained vision model

We chose to use a pretrained variant of the **ResNet-50** model, which consists of 50 convolutional layers. The model was trained on the famous 1000 classes *ImageNet* dataset with a total of $2,557,032$ parameters. A quick demostration of the pretrained network is demonstrated in figure 1, showing the original image, the preprocessed image given to the model and top 4 un-softmaxed output scores, corresponding to their predicted classes.

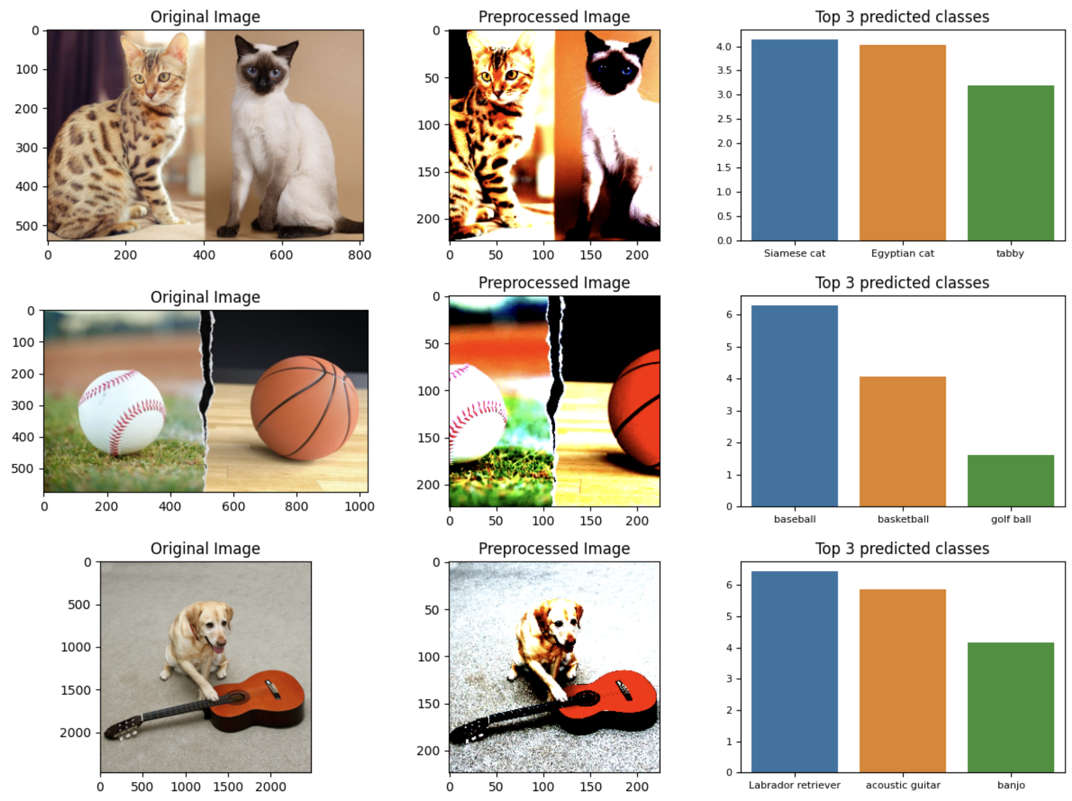

Figure 1: Pretrained model performance

2 Running LIME

In this part we implemented an explainable-AI model called LIME in order to explain which pixels have "convinced" the model the most to yield a certain class score given an image. We will do this process for the top 3 output scores. Given an image x, Our algorithm consists of the following pipeline:

1. Preprocess the image according to the pretrained model requirements.
2. Get classes a, b, c corresponding to the top 3 output scores from the model $f_1(x), f_2(x), f_3(x)$. e.g baseball, basketball and golf ball in the second image from figure 1
3. Segment x into 20 super pixels using *slic* algorithm.
4. sample n random permutations of superpixels (out of 2^{20}). Each permutation is defined as taking some superpixels and leaving out the rest. In our experiment, $n = 500$.

5. Generate n new perturbed images $\{x'\}_{i=1}^{n}$, considering only the corresponding superpixels from the previous item. Other pixels are set to black.
6. feed the model with all perturbed images to get $f_a(x'_i), f_b(x'_i), f_c(x'_i), \forall i \in \{1...n\}$ where a, b, c are the classes from the second step
7. For each label $\ell \in \{a, b, c\}$, fit a lasso regression model with the superpixels permutations used to create x'_i as features and $f_\ell(x'_i)$ as the label. The features were weighted according to the similarity of the *cosine distance* between the perturbed instance and the original image. A weight decay parameter $\alpha \approx 0.1$ was used in the process to zero out weak superpixels weights.
8. Return only positive weights, which ideally contributed the most to the model output score.

3 Results

Running the above pipeline on the 3 images from the first section resulted in pretty good results, as implied from figure 2. It is worth noting that only 500 random permutations of 20 superpixels were used to obtain these results, thus using a bigger n value and/or different number of superpixels should probably lead to improvement.

Figure 2: LIME - Segmentation Results