Explainability Project

Sunday, November 21, 2021 6:14 PM

Project on Activation maximization and GradCAM

By
Oluwatosin Alabi, Abel Kahsay Gebreslassie, and Suin Choi

Introduction

Machine learning models are been deployed in the world regularly and in multiple fields. They are used for fraud detection, loan collection, in medicine, agriculture, self-driving cars and a lot of other industries. This machine learning models are used to give insight to various professions and play sa major role in decision making when used. Users of this machine learning models and people whose lives are affected by this machine learning models, generally question the reason why machine learning models make the decisions they make. For the general public to accept these decisions, the results from machine learning models need to be interpretable and trust-worthy.

Explainable AI is a field that produces methods that can interprete results and allow for trust in models. Explainable AI methods usually do one or more of the following.

1. interprete the data, this could be help us understand if there are biases in the data, what dimensions of the data are most important.
2. explain the model via the visualization of internal features. It answers questions like what kind of patterns is the model looking for, how do we make comparison across models, how nodes interact.
3. Explain the predictions and the results. Questions like why certain inputs led to a particular output, what parts of the inputs were used to produce an output

Some other ways in which explainability techniques may be grouped is via local vs global techniques or Model-sepcific vs post-hoc. Local techniques look to get explanations for only a specific instance of the data, while global interpretations look to give explanations over a set of inputs such as an entire class, or over time periods. Model specific models have interpretability already encoded in the model, such as decision trees are considered model specific models. Model post-hoc methods are methods that can be used on a model that has already been made, it offers interpretability on already trained models.

For this project, we choose to explore two popular methods for interpretability

1. Activation maximization, a way of finding out patterns that filters give maximum response to.
2. Gradient weighted class activation maps (Grad-CAM)

We motivate both of these methods, using examples on MNIST and with using pretrained models. We then use these methods to explain a custom task.

Custom Task: Potholes in roads are very common in Sub-Saharan African roads and can be very dangerous. We try to build an automatic classifier on a given annotated dataset to classify between good roads and bad roads. We perform activation maximization, and Grad-CAM to attempt to explain these tasks well. We use the Pothole detection dataset found at (https://www.kaggle.com/atulyakumar98/pothole-detection-dataset). It contains 352 normal roads and 329 potholes. Total 681 images.

Fig 1. Image of a good road vs Image of a potholes from dataset used.

Activation minimization

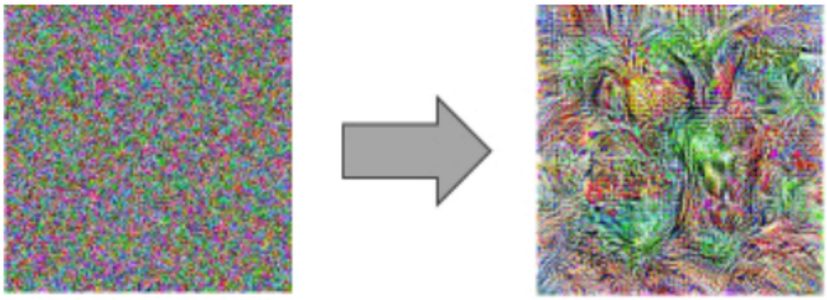
Activation maximization is a technique that originates from the adverserial methods field. It is described as a method for generating fake images that can fool a neural network by using the gradient ascent.

Gradient ascent is given as

$$x_{t+1} = x_t + \alpha \frac{\delta L_{a,f}(\theta, x_t)}{\delta(x_t)}$$

Where X is the image, x_t is the image at iteration t. $L\_a(\theta, x_t)$ is the activation of the of the filter f in layer a. and the differential is the gradient of this activation with respect to the pixels of the image.
This method produces the image that gives a high activation for this given image by freezing the neural network and making changes with respect to each pixel of the image.

An example from, Explainable AI: interpreting, explaining and visualizing deep learning by Samek et al



(a) Random initialization

(b) Synthesized rubbish example

Fig 2. An example of activation maximization.

We also have examples of performing the experiment with a layer of resnet50. This example was largely based of of the Keras example codes and we made some observations on what kind of images do early layers respond to, and what kind of images do later layers respond to

We first tested using an early layer , **the first conv in the block1**.
We visualize the images that the 64 filters in this layer

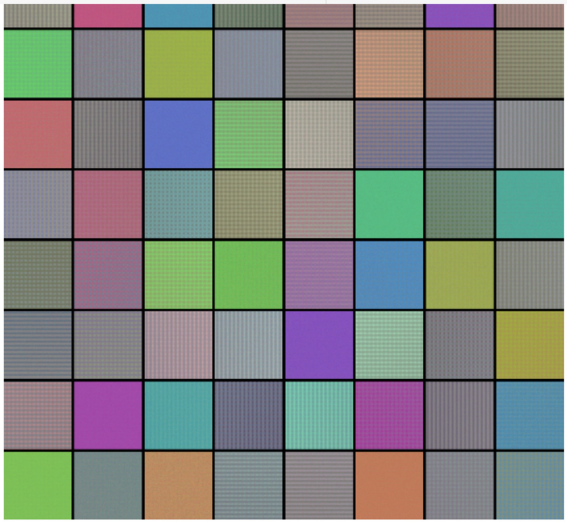


Fig 3. activation maximization of first conv in the first block.

As we can see, images that the filters in conv1 block 1 give the most activation are mostly of different colors, and not specific patters, or shapes.

We test for a middle layer, we took the **4th conv in the 3rd block**.

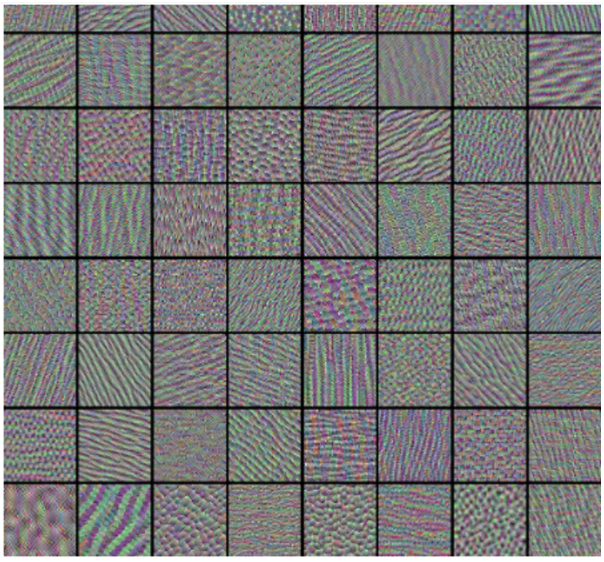


Fig 4. activation maximization of filters in fourth con in the 3rd block.
As we can see, there are a lot more complex patterns that produce the maximum activation for this layer.

Lastly, we test the **last conv in the whole network**

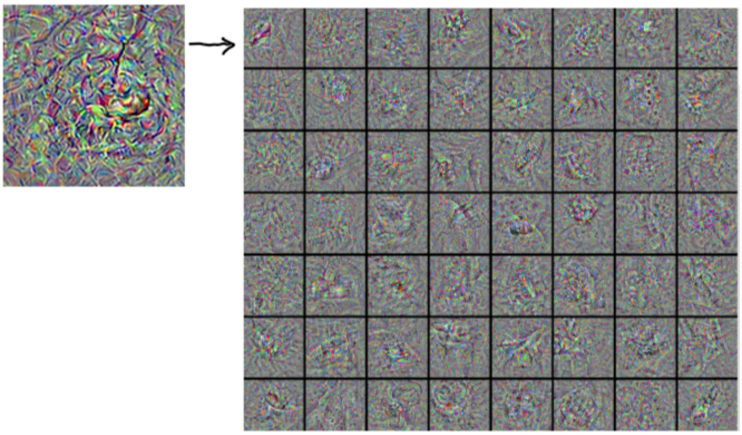


Fig 5. activation maximization of filters in last conv of resnet
We can see even more complex patterns an indication that the deeper layers learn more complex

Activation maximization on MNIST

We also run the same experiment on a simple model trained on 2 conv layers and 2 dense layers with the MNIST dataset. Accuracy of 99.24% on test dataset for the model trained.

We test for the first conv layer, we notice lines, diagonal, horizontal, vertical and some patterns,a nd superisingly, something that looks like a "1"

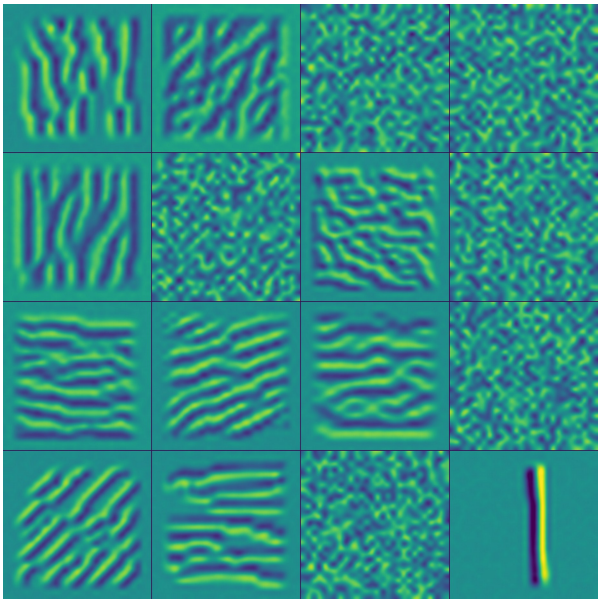


Fig 6. activation maximization of filters on first conv layer of MNIST model

Second Convolution

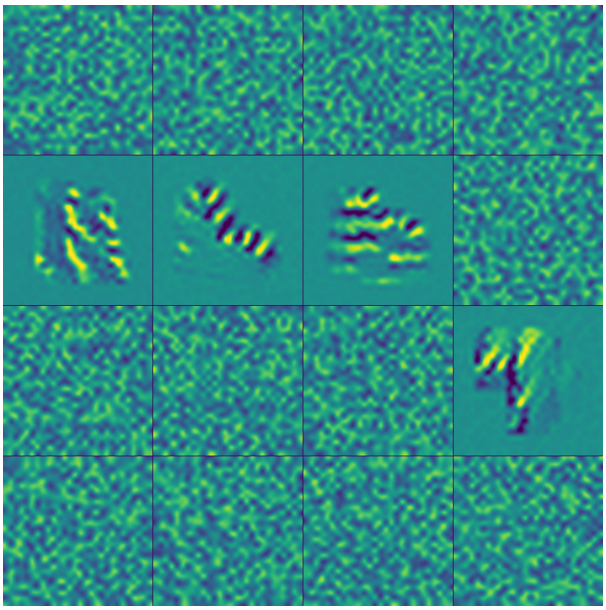


Fig 7. activation maximization of 16 filters on second conv layer of MNIST model

We noticed that a lot of the filters in the second layer respond to what seems to be noise, and not any particular patterns. We can see something that looks like a "7", maybe a "4", an "8" or "5"(not sure).

A side note: It is uncommon to perform Activation maximization for a dense nodes with the aim of using them to explain patterns in the image as dense layers do not usually encode spatial information and are used as classifiers, but we wanted to experiment and see if this was possible.

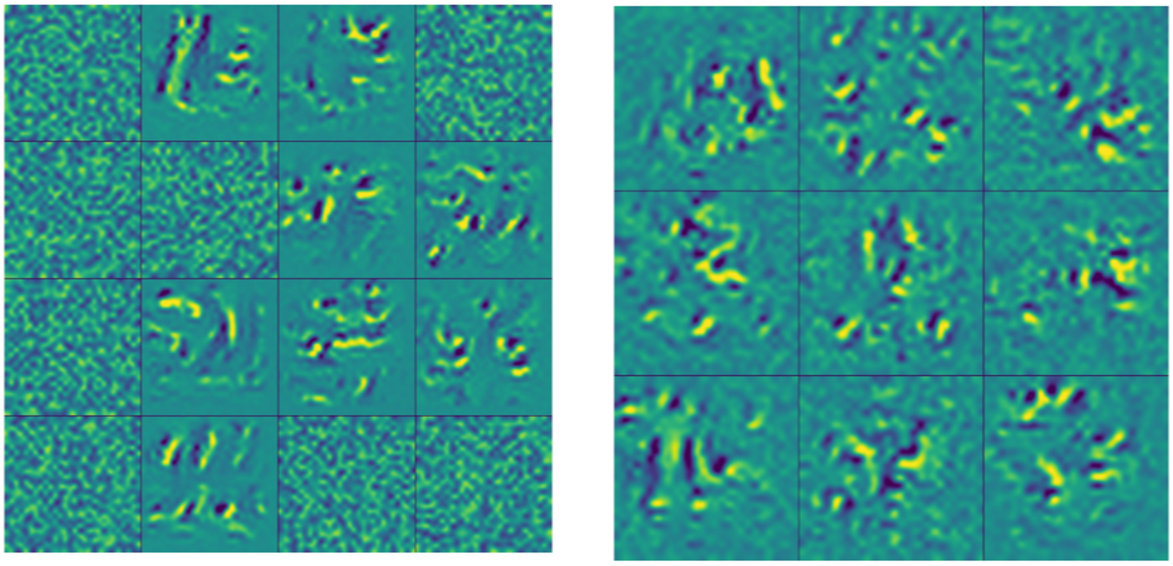


Fig 8. activation maximization on dense layers as a test, using MNIST model

GradCAM - Gradient weighted class activation maps.

Another popular method for interpretability that we explored in this project was Grad-CAM. It is meant to be an improvement over class activation maps.

Class activation maps are a method of generating heat maps that show which part of the image deep learning model the model checks for a particular class

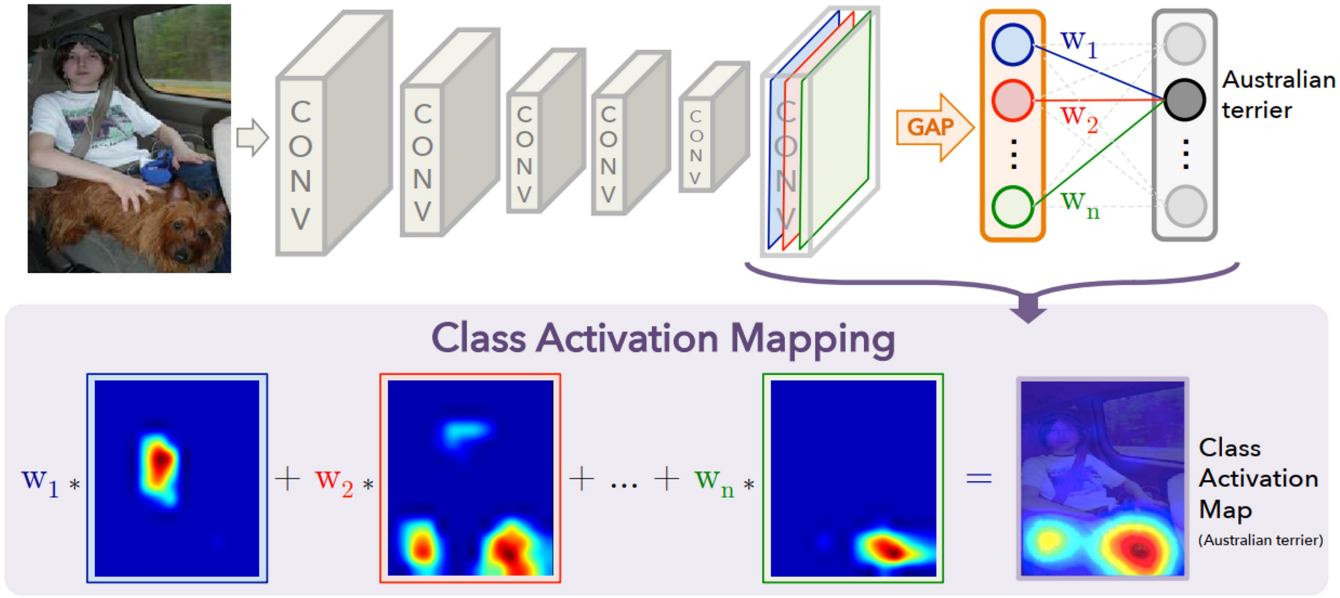


Fig 9. Learning Deep Features for Discriminative Localization by Zhou et al.

For example, the image from the paper, is for the class "Australian terrier". The method works by adding a global average pooling to the last layer and some dense layers without non-linearities to perform the classifications.

For a given image, let $f_k(x, y)$ represent the activation of unit k produced by filter k and normalized by the size n in the last convolutional layer at spatial location (x, y). Then, for unit k, the result of performing global average pooling without considering the normalization constant is given as,

$$F^k = \sum_{\{x,y\}} f_k(x, y)$$

Thus, for a given class c, the input to the softmax, S_c, is

$$S_c = \sum_k w_k^c \; F_k$$

where w_k^c is the weight corresponding to class c for unit k, as shown in Fig above. Essentially, w_k^c indicates the **importance** of F^k for class c. Finally the output of the softmax for class c, Pc is given by

$$P_c = \left(\frac{\exp(Sc)}{\sum_c(\exp(S_c))}\right)$$

All bias terms are set to zero for the softmax to have little effect on classification performance.
By plugging F^k the output of the global avergae pooling layer into the class score S_c, we obtain

$$S_c = \sum_k w_k^c \sum_{\{x,y\}} f_k(x,y)$$
$$= \sum_k \sum_{\{x,y\}} w_k^c f_k(x,y)$$

We define M_c the class activation map for class c, where each spatial element is given by

$$M_c(x,y) = \sum_k w_k^c f_k(x,y)$$

Thus,

$$S_c = \sum_{\{x,y\}} M_c(x,y)$$

Which shows that M_c indicates directly the importance of an activation at the spatial grid (x,y) leading to the classification of an image to class c.

As an example for the class australian terrier was shown in the paper, the weights w_1 to w_9 are used to perform a weighted sum with the activation maps of the last convolutional layer of the model. This gives the class activation map.

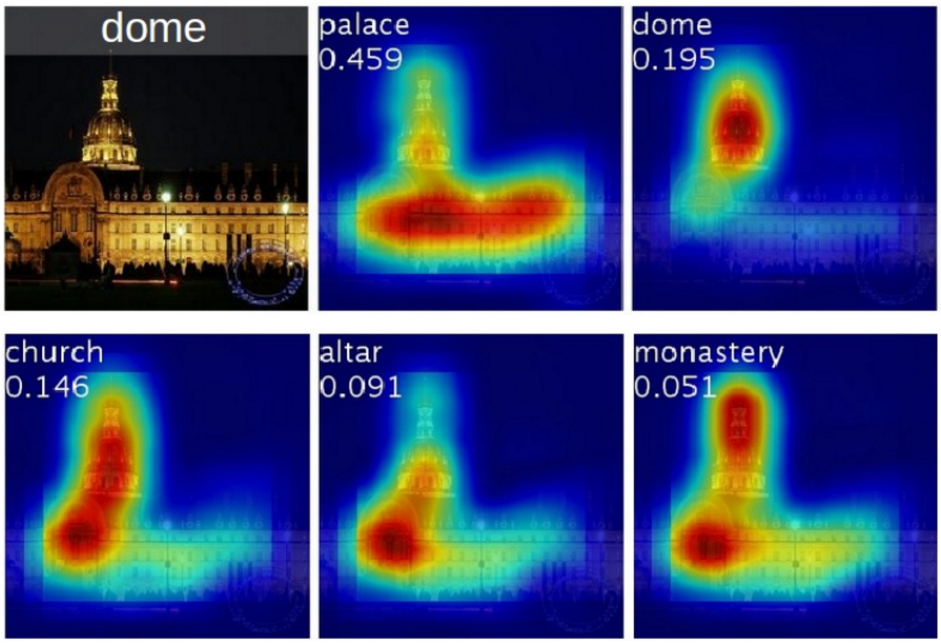


Fig 10. Examples from the paper by Zhou et al , Learning Deep Features for Discriminative Localization.

A drawback of this method is that it requires that the user to retrain the model to get the weights w_k^c . This retraining can affect the accuracy of the model. The method described above also puts a huge restriction on the architecture of the model and cant be used for problems like Visual Question and answering or Image Captioning.

Gradient based class activation maps (Grad-CAM) were designed to solve these problems.

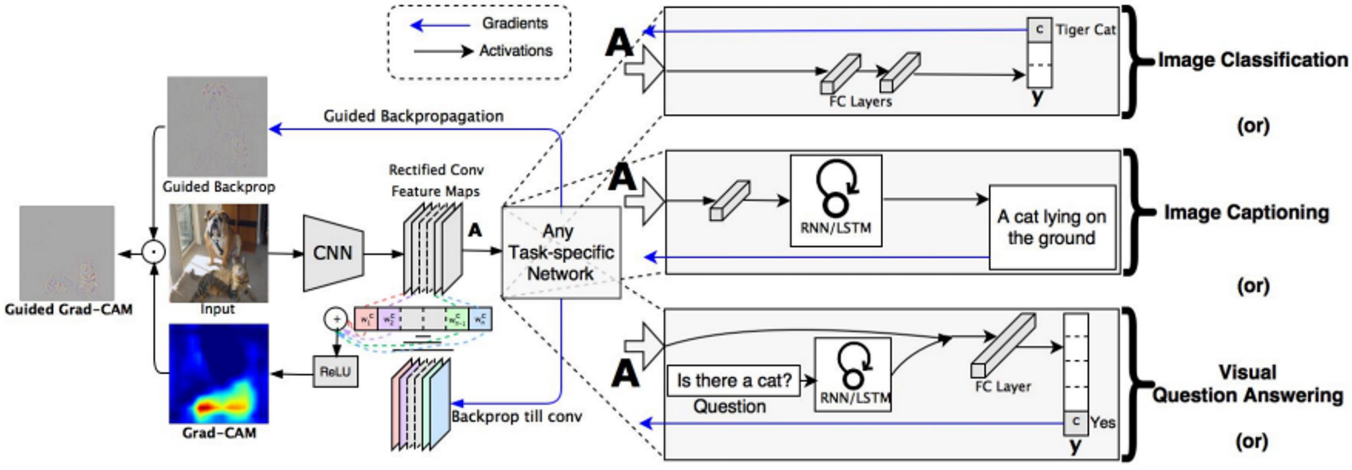


Fig 11. Methodology of Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization

It is characterized by any task specific layers, and not just a global average pooling followed by a dense layer. For this method, it is assumed that there is a last layer, a loss and a gradient, which is true for most neural networks. A quick explanation of the Grad-CAM methodology is that the gradients from the loss are used to calculate the gradients with respect to each feature map. The

gradients w.r.t. each feature map is then used to perform a weighted average ensuring that only positive correlations are used (Hence the ReLU layer). This eliminates the need to train to obtain w_k^c from the previous method, and also removes the restriction on the architecture.

In more details Grad-CAM is proved to be a generalization of CAM. The score for a single class for CAM was given by

$$Y^c = \sum_k w_k^c \frac{1}{Z} \sum_i \sum_j A_{i,j}^k$$

and

$$F_k = \frac{1}{Z} \sum_{\{x,y\}} A_{i,j}^k$$

Hence

$$Y^c = \sum_k w_k^c \; F_k$$

Taking the gradient with respect to the $A_{\{ij\}}$

$$\frac{\delta Y^c}{\delta F^k} = \frac{\frac{\delta Y^c}{\delta A_{i,j}^k}}{\frac{\delta F_k}{\delta A_{i,j}^k}} = \frac{\delta Y^c}{\delta A_{i,j}^k} . Z$$

We also know that $\frac{\delta Y^c}{\delta F^k} = w_k^c$

Hence,

$$w_k^c = \frac{\delta Y^c}{\delta A_{i,j}^k} . Z$$

Summing both sizes over pixels (I and j)

$$\sum_i \sum_j w_k^c = \sum_i \sum_j Z . \frac{\delta Y^c}{\delta A_{i,j}^k}$$

Since w_k^c and Z do not depend on I and j and simming over all I and j pixels then we can rewrite the previous equation as

$$Z w_k^c = Z \sum_i \sum_j \frac{\delta Y^c}{\delta A_{i,j}^k}$$

Hence we can write that

$$w_k^c = \sum_i \sum_j \frac{\delta Y^c}{\delta A_{i,j}^k}$$

Hence, the Grad-CAM method can be seen as a generalization for CAM.

Experiments on Xception using Grad-CAM

We performed experiments for Grad-Cam using code based on Keras Example by francois chollet.Testing the idea of Grad-CAM on an image classification model for a pretrained Xception Image classifcation network. We produce class activation maps using Grad-CAM for a picture containing African elephants, one of the classes in imageNet.

The class activation produced is given by the Image below.

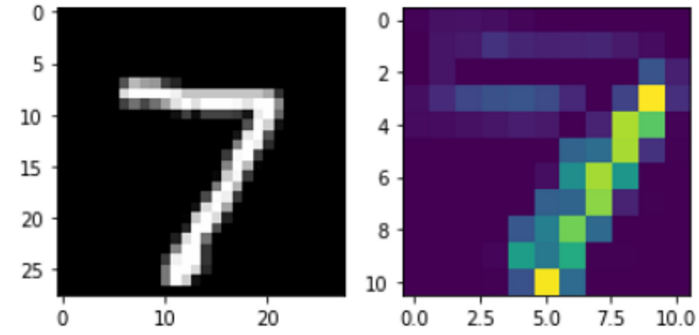
Fig 12. class activation map produced using grad-CAM on images with African elephants

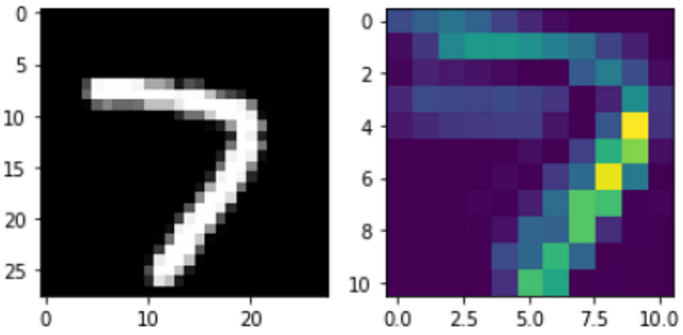
We also perform Grad-CAM on an image containing a cat and and dog and produce the class activation maps for the class dog and the class cat.

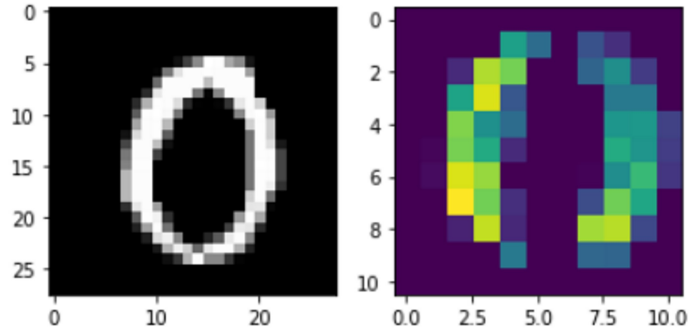
Fig 13. class activation map produced using grad-CAM on images with a dog and a cat, class activation map for class dog(top), Class activation map for class cat(bottom)

Experiments on MNIST using Grad-CAM

Using a simple model with 2 conv's and 2 dense layer we perform Grad-CAM and produce feature visualization.







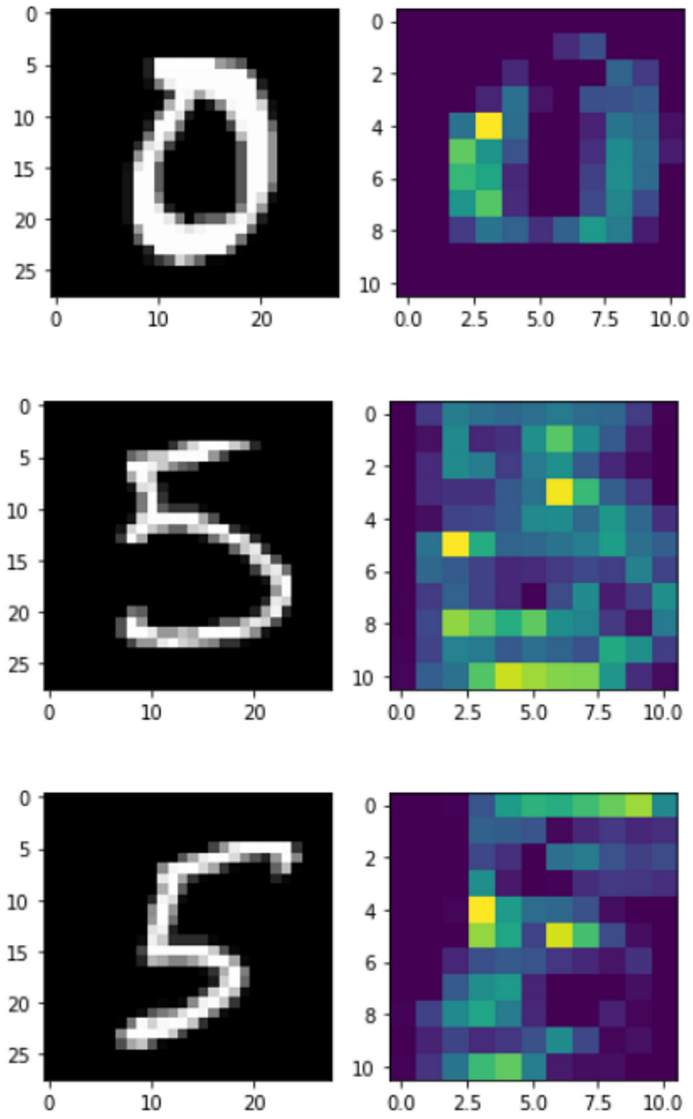


Fig 14. Class activation maps using Grad-CAM on MNIST

MNIST is not a complicated example, but we can see where the model is looking at to make decisions for each example given. For example for 7, it looks more towards the vertical line than the vertical, "0" looks almost like two moons that are connected.

Activation Maximization and Grad-CAM on good road vs pothole dataset.

We build a model with 3 convolution layers with 64, 32 and 16 filters respectively. We were able to get around 0.8984 accuracy.

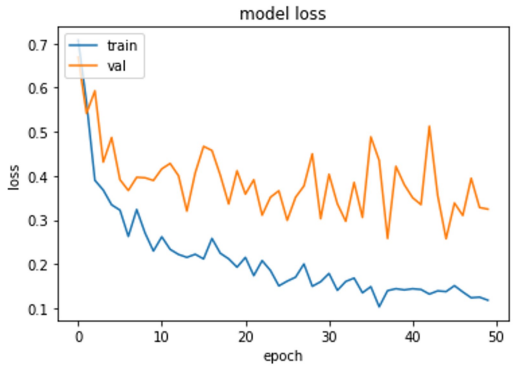


Fig 15. Loss graph for model trained on good road vs pothole dataset

For the first conv layer, We notice that they are majorly color filters.

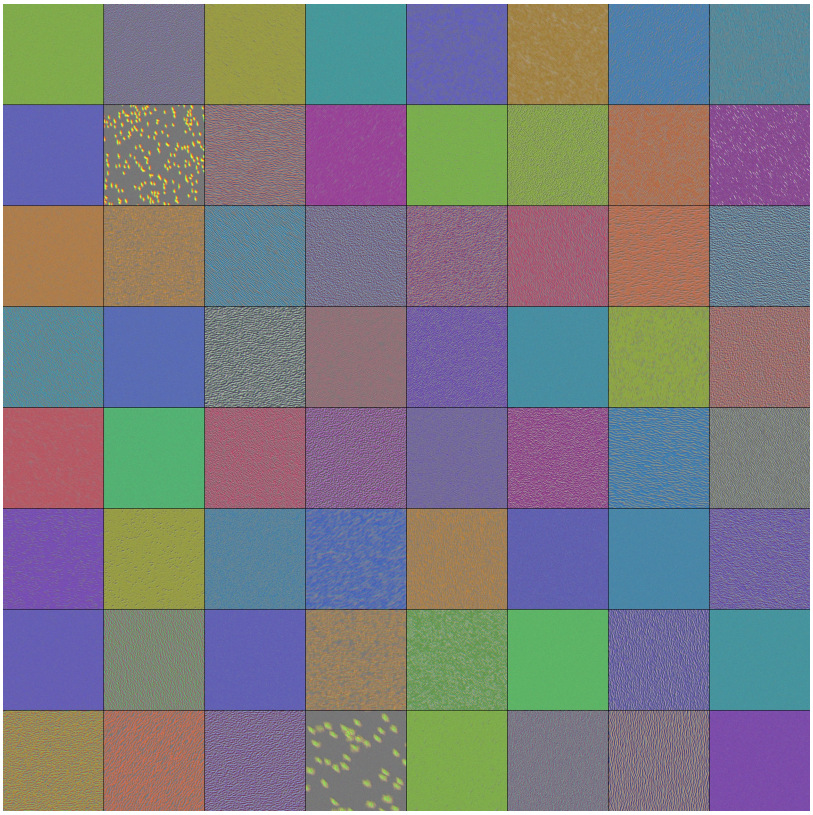


Fig 16. Activation maximization on Pothole model for first conv filters

For the middle conv layer, we have some patterned images, which looks like cracks and spots. We also have some filters that do not seem to respond to any specific patterns, and some filters that respond to green (maybe vegetation)

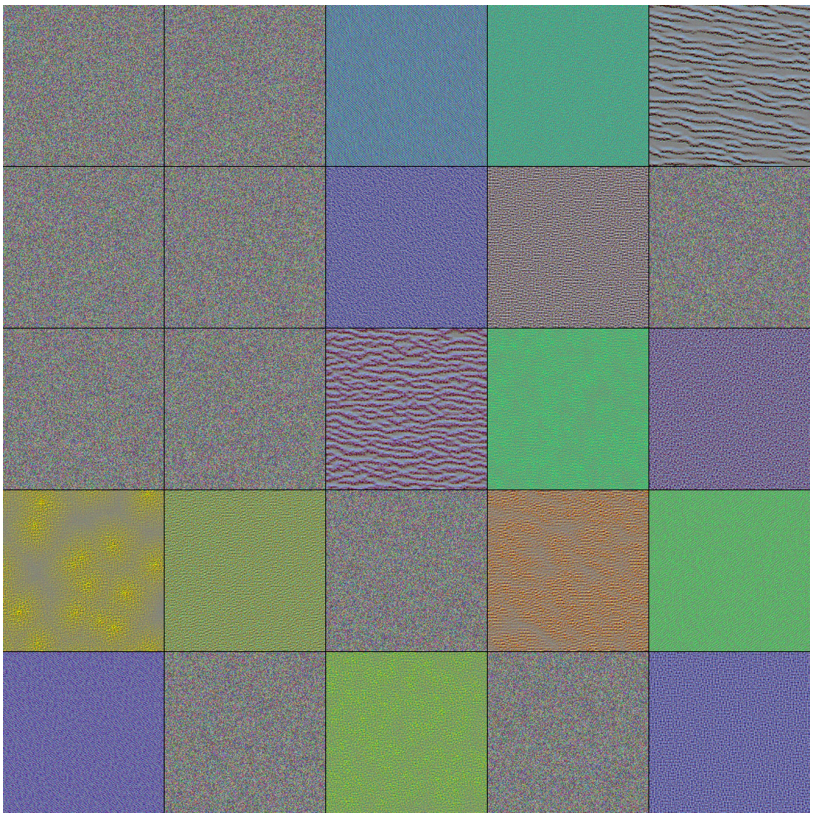


Fig 17. Activation maximization on Pothole model for second conv filters

For the final layer, we get even more complicated and textured patterns, that look like road patterns, and some green too.

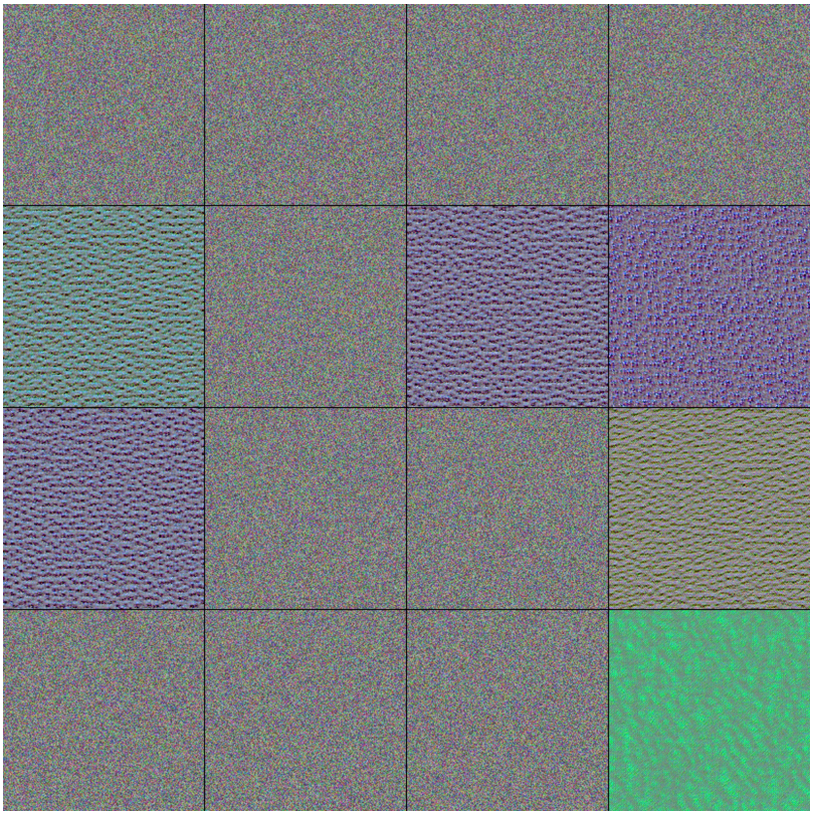


Fig 18. Activation maximization on Pothole model for third conv filters

Just like other examples(MNIST and Pretrained network), the earlier layers respond to lines and textures. But the later layers respond to more complex patterns.

Although activation maximization is great, we don't get so much information from activation maximization. We would like to see the behaviour of the model as it responds to a particular input. there are some things that would be interesting to interprete, such as the part of a particular image that contributed the most to the decision, and maybe that would give us more knowledge that would improve our knowledge gotten from activation maximization results.

Grad-CAM on the model for the Pot-holes gives us some of this examples

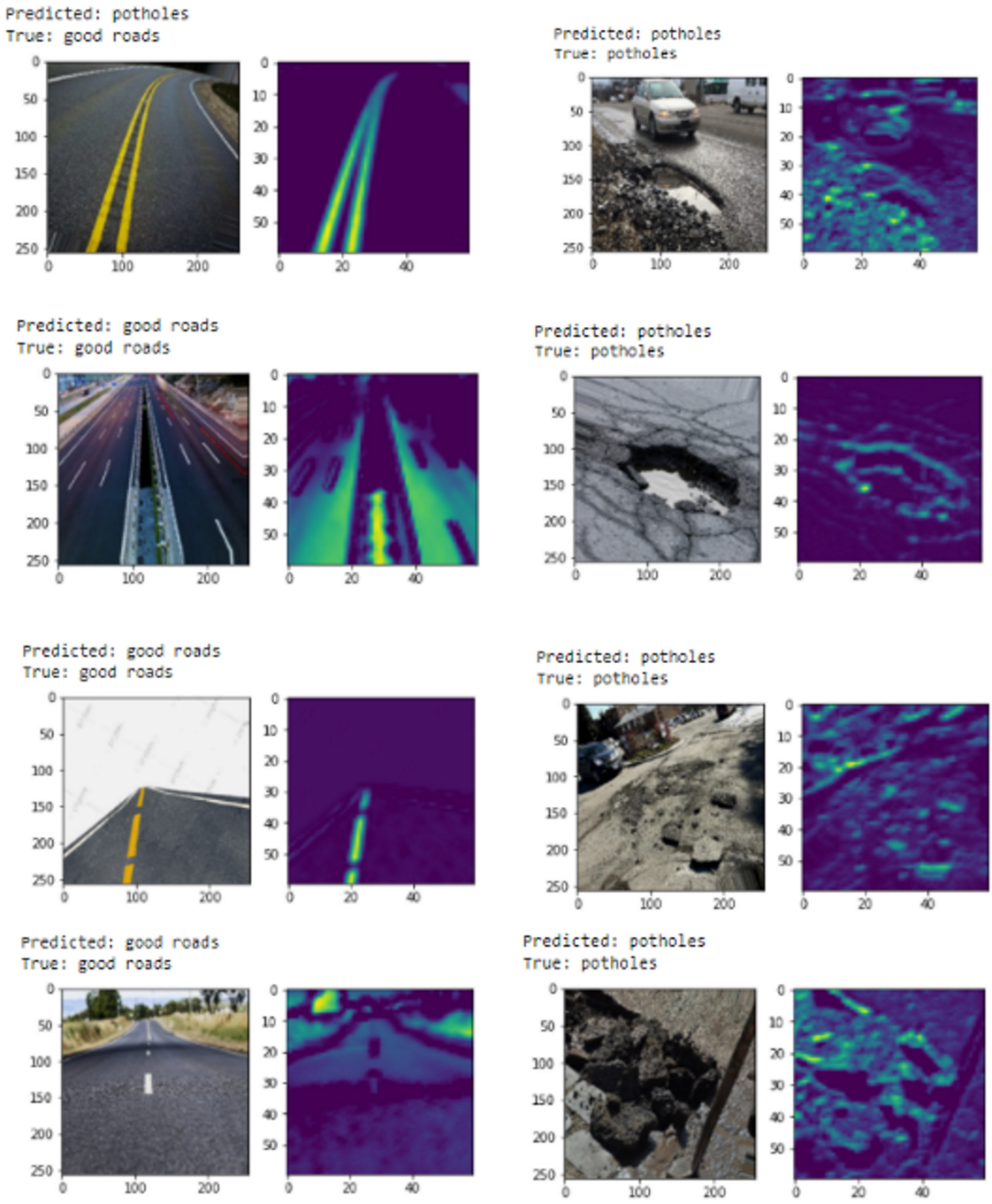


Fig 19. Class activation maps using Grad-CAM on Pothole dataset (good results)

As can be seen it interprets straight roads very well over this examples by taking into account the road, and markers on the road, and it also recognizes cracks, by using the parts on the cracks of the road for its predictions. The model clearly understands the meaning of a pothole, when the pothole is a major part of the image.

Here are some other examples that shows that the model trained is using shortcuts.

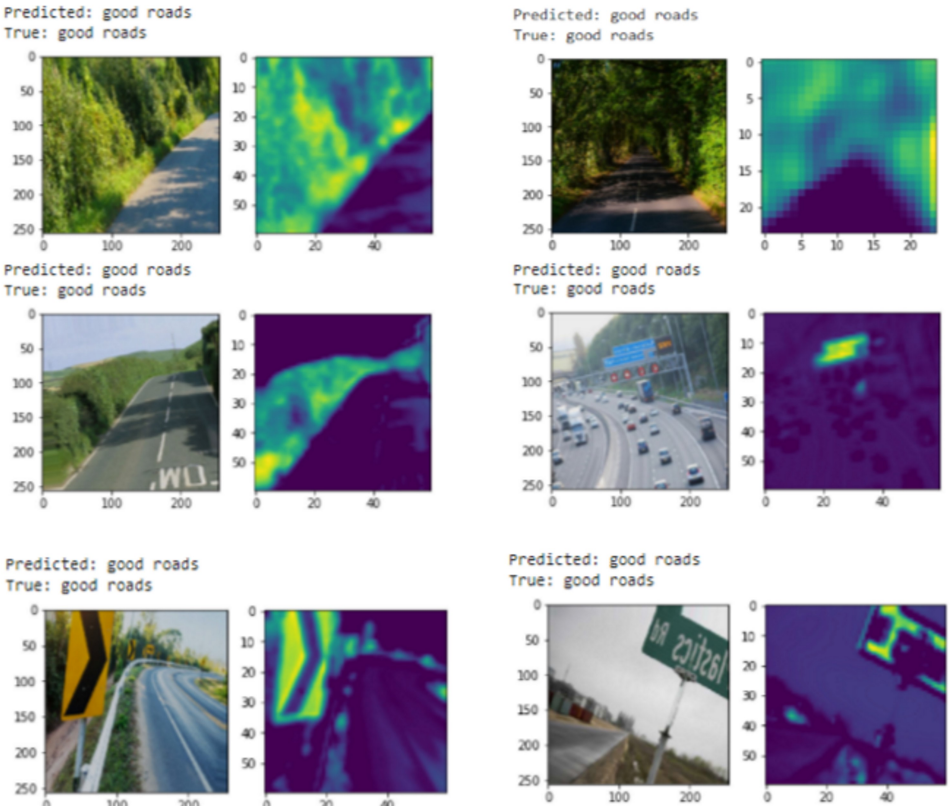


Fig 20. Class activation maps using Grad-CAM on Pothole dataset (good results)

We notice that the model uses green vegetation as a sign that there is a good road. It also looks at signboards to give an indication that there are good roads.

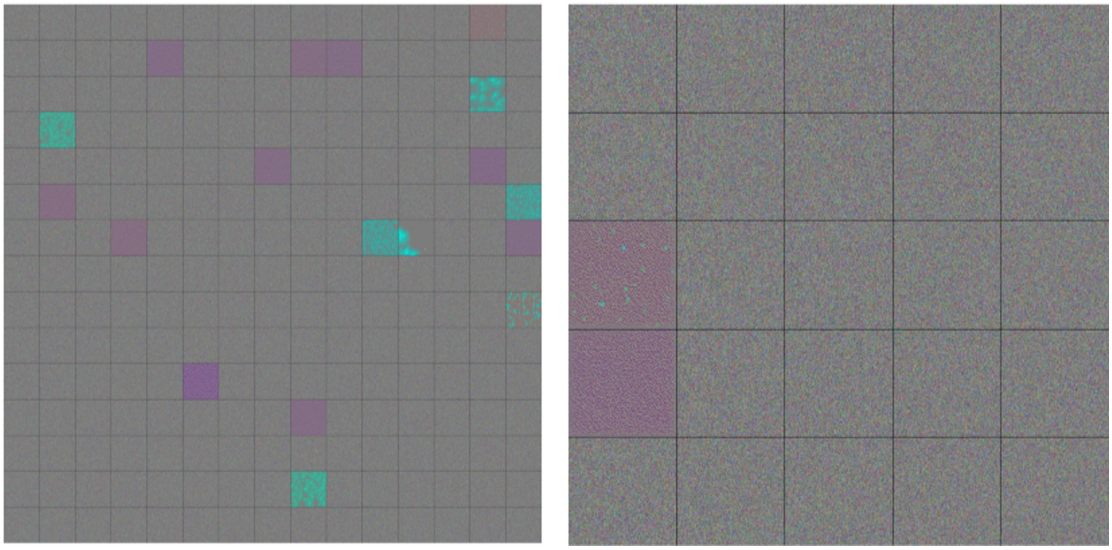
We looked at the dataset and we noticed that most of the good roads have vegetation around, and it is common sense that traffic lights, and signboards are generally in the middle of the city and are used where the roads are very good.

Conclusions and notes

We note that if we used a better dataset that focuses mainly on roads, we probably would get even better results than we have currently as there would be no distractions such as vegetation, signboards, the sky, and the model would able to focus majorly on the road.

We also learnt about transfer learning. We tried to perform transfer learning with ImageNet weights on ResNet to get better accuracy, but we did not get it to work very well. We got training accuracy to be 1.000, but the validation accuracy was around 0.5. This was a reminder that transfer learning works well if the data the network is trained on is similar to the domain you want to build a model for.

We also learnt about filters. And networks. We used much larger networks with more filters, in a bid to increase the accuracy. We got the images below. Which shows that most of the filters do not respond to any particular patter at all, but majorly noise.



We also note that using multiple explainability methods to explain a task is a good practice. We noticed that there were a lot of "green" in the activation maximization of the conv layers. We were not sure what the greens were, but after using grad-cam, we were certain that vegetation were playing a huge role in the predictions.