PROJECT REPORT: VISUALIZING CONVOLUTIONAL NEURAL NETWORKS

Amey Agrawal & Jaikumar Balani
{f2014148, f2014022}@pilani.bits-pilani.ac.in

1 ABSTRACT

Convolutional neural networks are becoming increasingly popular and have been the centre of the deep learning revolution but yet training a large CNN model has remained a mystery. We are attempting to build a framework to visualise a CNN model, using multiple methods. Including deconvolution nets proposed by Zeiler & Fergus (2014), activation based method proposed by Erhan et al. (2009) and t-SNE.

2 INTRODUCTION

Convolutional neural networks have become a popular technique in all kinds of computer vision task since they showed exceptional results on ImageNet challenge in 2011. Today CNNs are used for all kinds of applications including sentiment analysis of short sentences, generating voice from text and creating 3D models from 2D images.

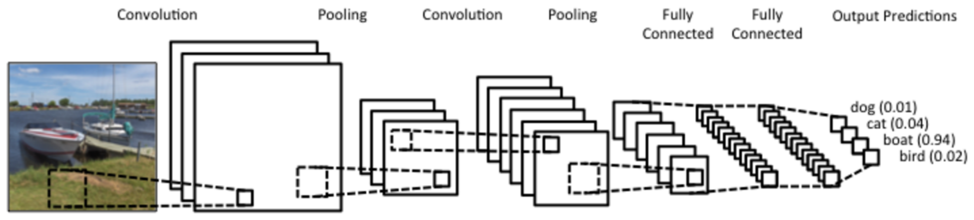


Figure 1: Genral feature extraction pipe of convolutional neural networks

Visualising the activations on each layer is the simplest and the most intuitive way of visualising convolutional neural networks but it has its obvious shortcomings. As the depth of filters is more than 3 it is generally hard to make sense beyond the first convolution layer.

3 PROBLEM STATMENT

To build a framework to visualise a CNN model, using deconvolution nets proposed by Zeiler & Fergus (2014), activation based method proposed by Erhan et al. (2009) and t-SNE.

4 RELEATED WORK

Multiple techniques were suggested by Erhan et al. (2009) for visualisation of CNNs based upon activations of individual units. Simonyan et al. (2013) established a connection between gradient based methods of CNN visualisation and deconvolution networks. Zeiler Fergus (2014) introduced a way to visualise CNNs using deconvolution nets.

Following is a brief review of the major works on which we have developed our toolkit.

4.1 Method 1: Deconvolutional Network

Deconvolution networks reveal the part of the input that excites individual feature maps at any layer in the model. Deconvolutional Network with switched pooling layers is used to project the activations back to the input pixel space. We use 2 different variations of this method which differ only in the fact that one stores the positions of the max values during pooling and reproduces similarly at the time of unpooling while the other only stores the maximum values and then reconstructs after unpooling.

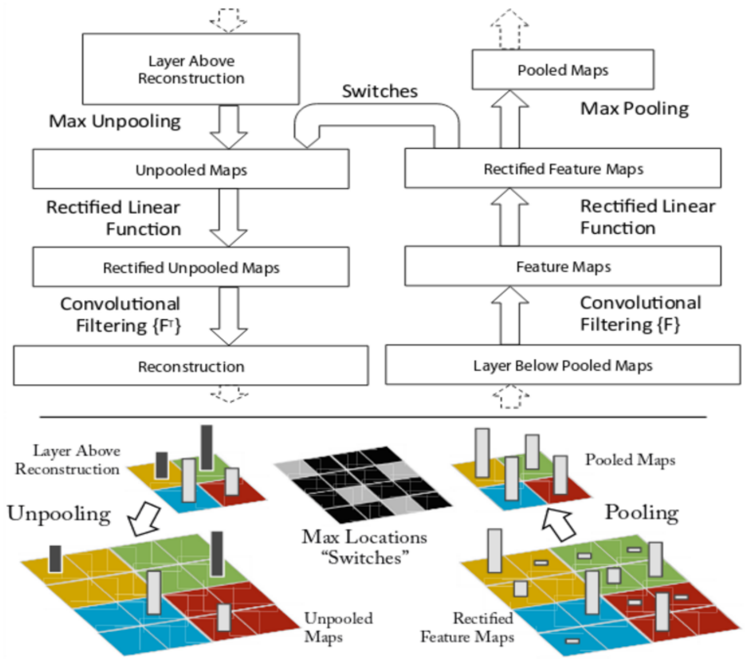


Figure 2: Deconvolution network with switched pooling

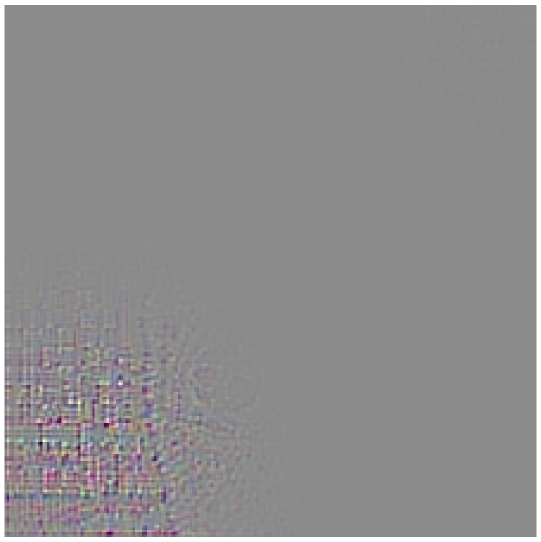


Figure 3: 16th Filter of layer 17th VGGnet visualized using deconvnet

4.2 Method 2: Gradient ascent on activation

In this method, we define the loss function as sum activations of all the neurones in the chosen filter. This loss is the maximised using gradient accent which enables us to visualise what excites a the given filter in general.

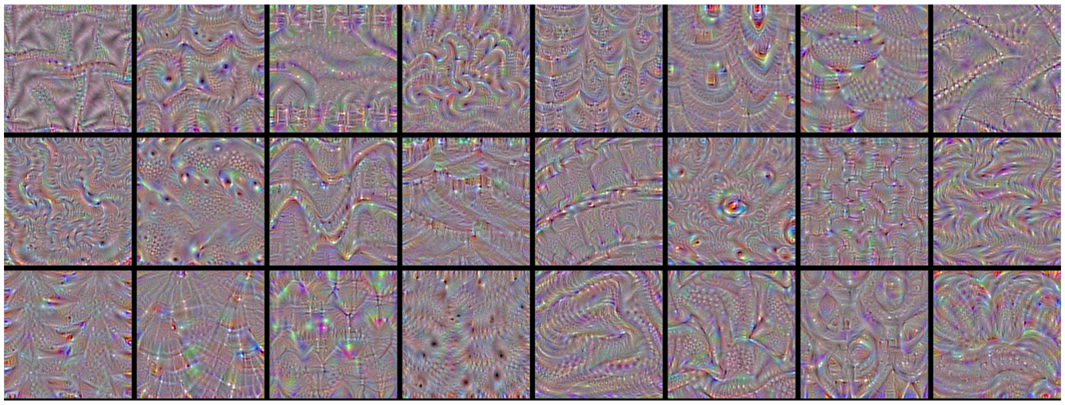


Figure 4: Visulalization of some filters of VGGnet using method 2.

4.3 Method 3: t-SNE visualisation of dense layers

Fully connected layers from CNNs have been used in multiple transfer learning application concluding image captioning and sentiment analysis. Hence visualising the last fully connected layer on a CNN can give us an understanding of what the network has learned overall. Maaten & Hinton (2008) proposed an optimisation based technique to visualise high dimensional vectors in lower dimensions called t-SNE. t-SNE tries preserve distance between two points between the two vector spaces.

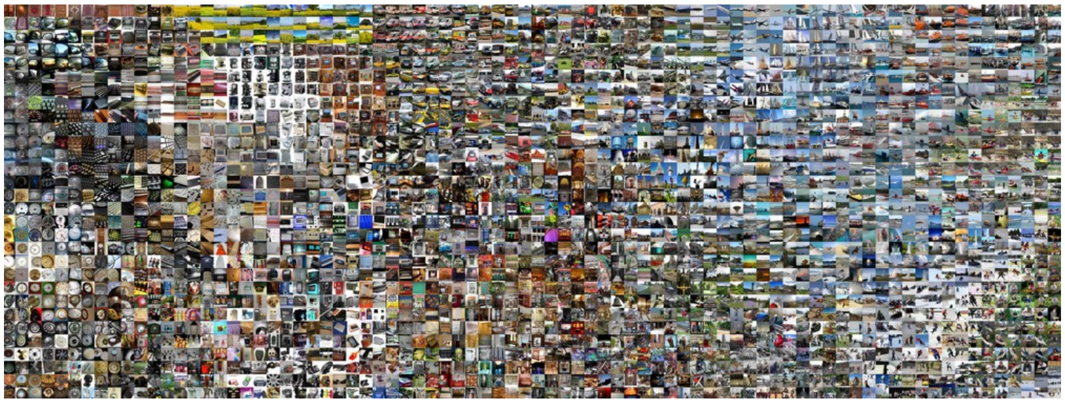


Figure 5: A visualization done using t-SNE from work of Stanford's CS 231n

t-SNE is a computationally extensive algorithm. Our implementation is a single threaded CPU based upon Scikit-learn. Hence while trying to visualise last layers of large models like VGGnet if we cannot use a large number of images.

5 Design of solution

The framework is built in python with Keras. Keras has been gaining popularity as deep learning framework due to its simplicity and flexibility. Keras can use both TensorFlow and Theano backends.

We have developed an extensible driver module with command interactive line interface. It can take any keras model stored in hdf5 format as input. If the user does not provide an input model VGG16 checkpoint is used by default.

Keras is used for automatic differentiation in method 1. While custom keras layers are made to create the deconvolution counterpart of given model. Scikit-learn is used to find the t-SNE embeddings of the last layer of the network. And the images are plotted on a 2D graph on their corresponding coordinates.

Matplotlib is used to plot and store images. Sometimes the output plotted by matplotlib looks very noisy though it is stored perfectly.

6 Screeshots and visuals

Following are few filters visualised with method 1.

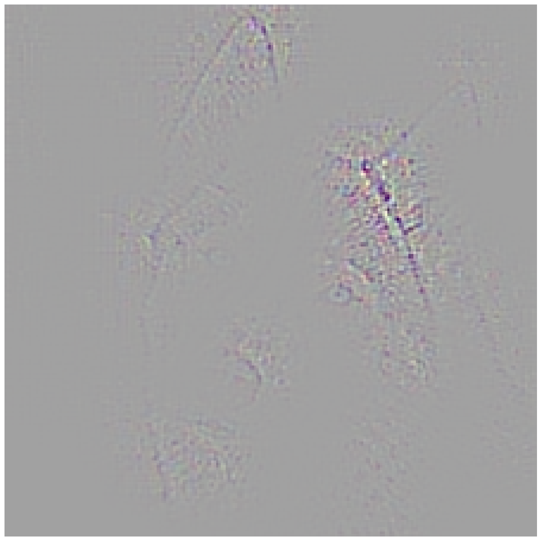


Figure 6: Deconvnet: Output 1

Following are some visualisation using method 2.

Following is few screenshots from the application:

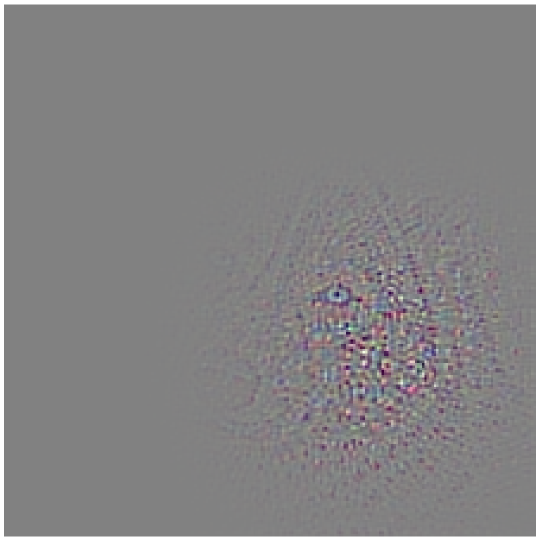


Figure 7: Deconvnet: Output 2

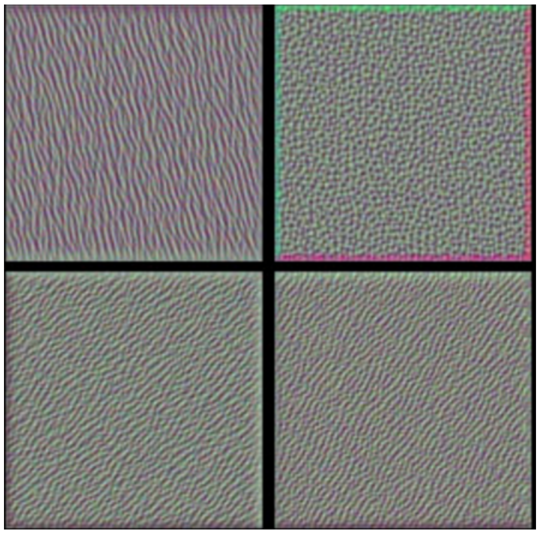


Figure 8: Gradient acent: Output 1

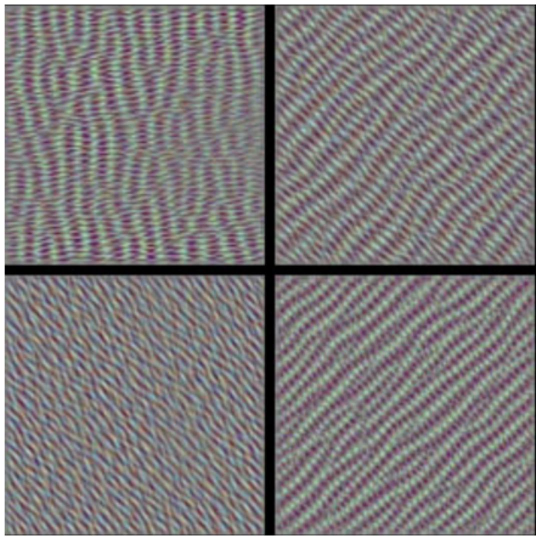


Figure 9: Gradient acent: Output 2

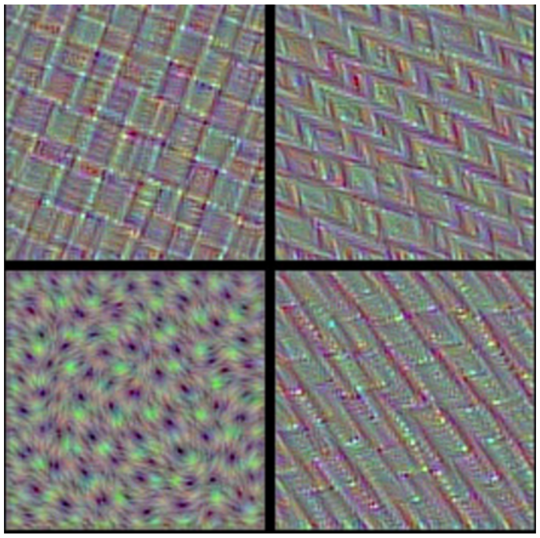
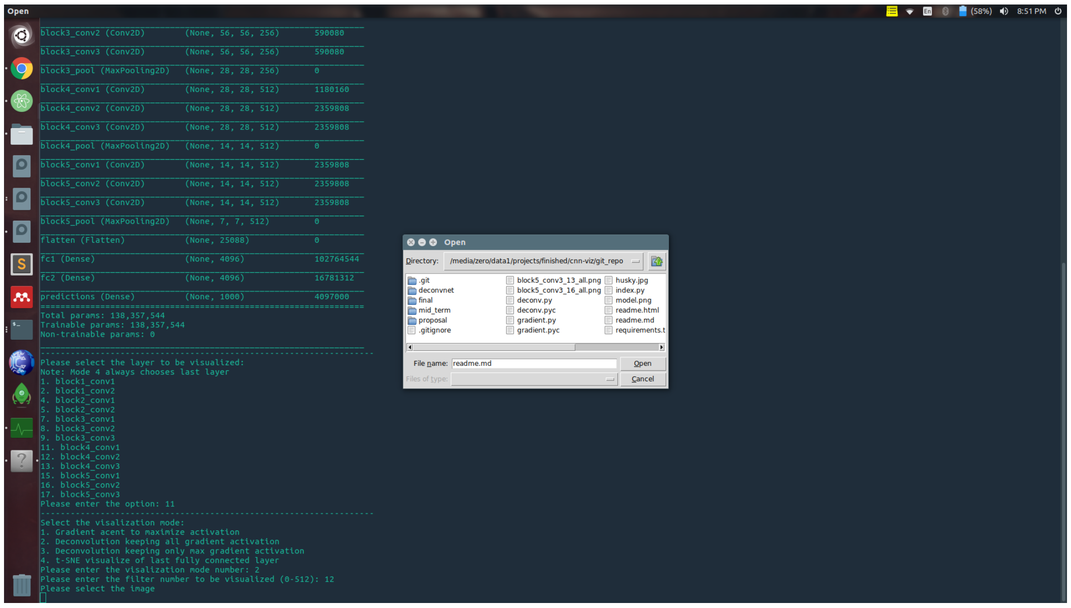


Figure 10: Gradient acent: Output 3



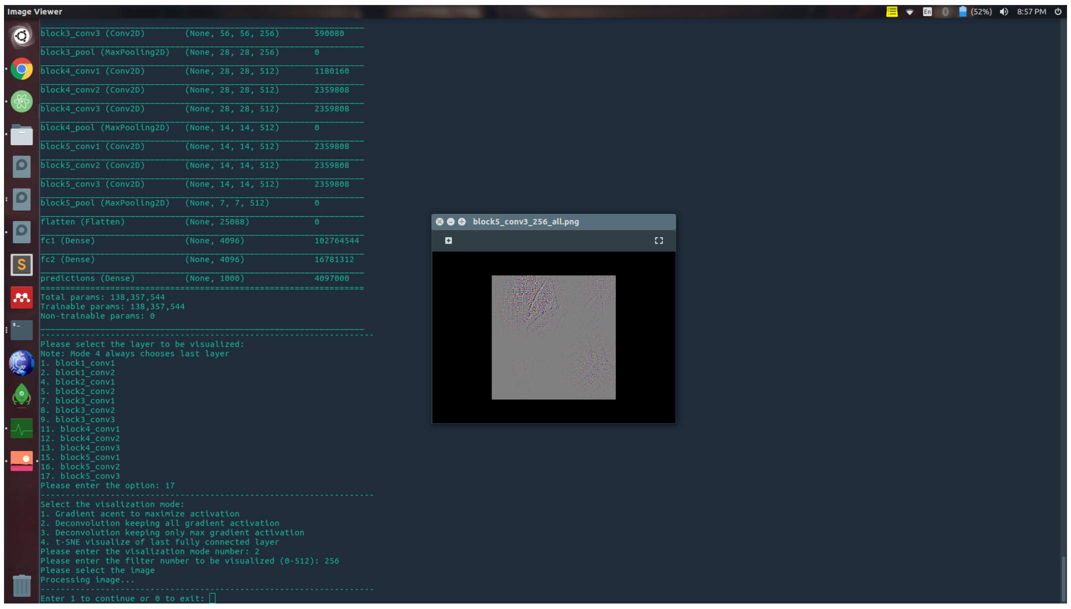
```
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 25088)             0
_________________________________________________________________
fc1 (Dense)                  (None, 4096)              102764544
_________________________________________________________________
fc2 (Dense)                  (None, 4096)              16781312
_________________________________________________________________
predictions (Dense)          (None, 1000)              4097000
=================================================================
Total params: 138,357,544
Trainable params: 138,357,544
Non-trainable params: 0

_________________________________________________________________
------------------------------------------------------------------
Please select the layer to be visualized:
Note: Mode 4 always chooses last layer
1. block1_conv1
2. block1_conv2
4. block2_conv1
5. block2_conv2
7. block3_conv1
8. block3_conv2
9. block3_conv3
11. block4_conv1
12. block4_conv2
13. block4_conv3
15. block5_conv1
16. block5_conv2
17. block5_conv3
Please enter the option: 11
------------------------------------------------------------------
Select the visalization mode:
1. Gradient acent to maximize activation
2. Deconvolution keeping all gradient activation
3. Deconvolution keeping only max gradient activation
4. t-SNE visualize of last fully connected layer
Please enter the visalization mode number: 2
Please enter the filter number to be visualized (0-512): 12
Please select the image
```

Directory: /media/zero/data1/projects/finished/cnn-viz/git_repo

.git	block5_conv3_13_all.png	husky.jpg
deconvnet	block5_conv3_16_all.png	index.py
final	deconv.py	model.png
mid_term	deconv.pyc	readme.html
proposal	gradient.py	readme.md
.gitignore	gradient.pyc	requirements.t

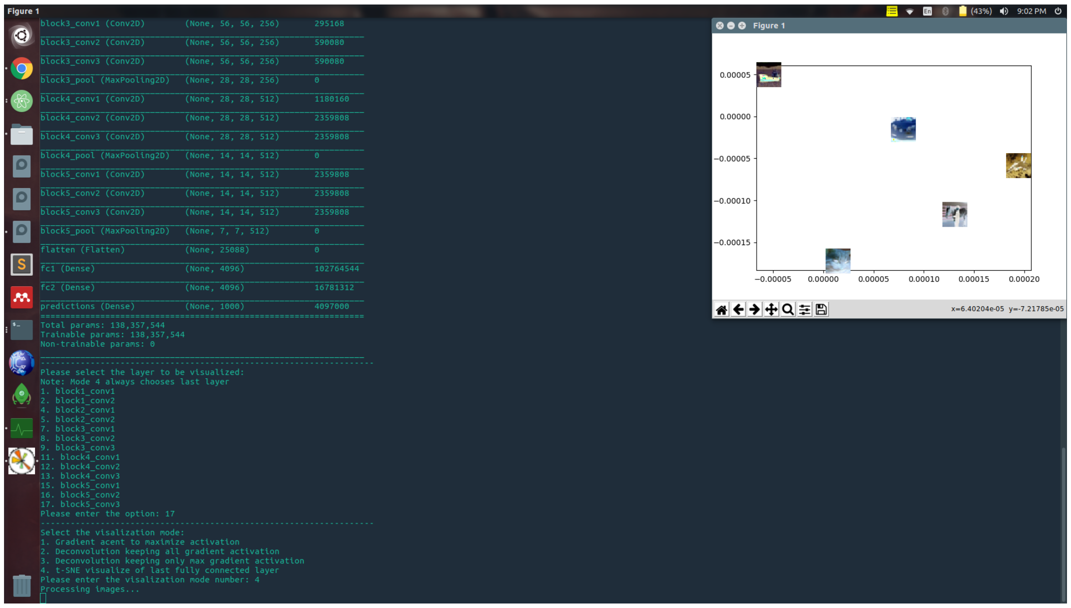
File name: readme.md

Figure 11: Screenshot 1



```
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 25088)             0
_________________________________________________________________
fc1 (Dense)                  (None, 4096)              102764544
_________________________________________________________________
fc2 (Dense)                  (None, 4096)              16781312
_________________________________________________________________
predictions (Dense)          (None, 1000)              4097000
=================================================================
Total params: 138,357,544
Trainable params: 138,357,544
Non-trainable params: 0

_________________________________________________________________
------------------------------------------------------------------
Please select the layer to be visualized:
Note: Mode 4 always chooses last layer
1. block1_conv1
2. block1_conv2
4. block2_conv1
5. block2_conv2
7. block3_conv1
8. block3_conv2
9. block3_conv3
11. block4_conv1
12. block4_conv2
13. block4_conv3
15. block5_conv1
16. block5_conv2
17. block5_conv3
Please enter the option: 17
------------------------------------------------------------------
Select the visalization mode:
1. Gradient acent to maximize activation
2. Deconvolution keeping all gradient activation
3. Deconvolution keeping only max gradient activation
4. t-SNE visualize of last fully connected layer
Please enter the visalization mode number: 2
Please enter the filter number to be visualized (0-512): 256
Please select the image
Processing image...
------------------------------------------------------------------
Enter 1 to continue or 0 to exit:
```

Figure 12: Screenshot 2



```
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 25088)             0
_________________________________________________________________
fc1 (Dense)                  (None, 4096)              102764544
_________________________________________________________________
fc2 (Dense)                  (None, 4096)              16781312
_________________________________________________________________
predictions (Dense)          (None, 1000)              4097000
=================================================================
Total params: 138,357,544
Trainable params: 138,357,544
Non-trainable params: 0

_________________________________________________________________
-------------------------------------------------------------------
Please select the layer to be visualized:
Note: Mode 4 always chooses last layer
1. block1_conv1
2. block1_conv2
4. block2_conv1
5. block2_conv2
7. block3_conv1
8. block3_conv2
9. block3_conv3
11. block4_conv1
12. block4_conv2
13. block4_conv3
15. block5_conv1
16. block5_conv2
17. block5_conv3
Please enter the option: 17
-------------------------------------------------------------------
Select the visalization mode:
1. Gradient acent to maximize activation
2. Deconvolution keeping all gradient activation
3. Deconvolution keeping only max gradient activation
4. t-SNE visualize of last fully connected layer
Please enter the visalization mode number: 4
Processing images...
```

Figure 13: Screenshot 3

7 Conclusion and Future Remaining

The toolkit provides a simple interface for machine learning researchers to visualise any CNN model using the method suited best for their application.

Though we currently have deconvolution network counterpart for most of the commonly used layers, implementations of many more layers could be added to visualise more complicated models. Also, the t-SNE visualisation can be scaled up using a preprocessing the data with PCA or developing a multiprocessing algorithm.

8 Contribution

Amey Agrawal: Development of custom keras layers for method 1 and implementation of method 2 in keras along with generation of t-SNE embedding in method 3.

Jaikumar Balani: Plotting and storing generated images using matplotlib.

References

Erhan, Dumitru, Bengio, Yoshua, Courville, Aaron, and Vincent, Pascal. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341:3, 2009.

Maaten, Laurens van der and Hinton, Geoffrey. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

Simonyan, Karen, Vedaldi, Andrea, and Zisserman, Andrew. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

Zeiler, Matthew D and Fergus, Rob. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pp. 818–833. Springer, 2014.