

CP3106 Project Report

**An Evaluation of Explainability of  
Convolutional Neural Network Architectures  
Based on Heat Maps**

By  
Hu Weilin

Department of Computer Science  
School of Computing  
National University of Singapore  
AY2018/2019, Semester II

Advisor: Assoc Prof Leong Hon Wai  
Deliverables:

Report: 1 Volume

## **Abstract**

Convolutional Neural Network (CNN) have demonstrated excellent performance at computer vision task since AlexNet dominated ILSVRC Competition in 2012. Despite high accuracy, CNN is still a black box without much interpretability.

The most popular and direct way to visualise, interpret and understand CNN is by heat map, which quantify the importance of individual pixels with regards to the classification decisions. It has been suggested by [11] that heat maps obtained from a model depend on the training data and model parameters.

By literature review, we propose a hypothesis that the heat maps derived from a CNN model are affected by its architecture. By combining subjective observation and objective clustering techniques, we come to the conclusion that there do exists some relationships between CNN architecture and the heat maps derived from it. Since heat maps can suggest the explainability of a model in someway, then we can conclude that the explainability of a CNN model is indeed influenced by its architecture.

In the second round of experiment, we validate the patterns found in round one from which we gain a better understanding of some CNN architectures in terms of its explainability and gain some factors that contribute to explainability.

Then we argue that an objective and quantitative method to evaluate the quality of heat maps obtained by different explanation methods [15] cannot be transferred to evaluate and compare heat maps across different CNN architectures. At last, we propose a subjective standard to assess the explainability of a CNN model.

### **Subject Descriptors:**

- I.2.10 Vision and Scene Understanding
- I.4 Image Processing and Computer Vision
- I.5. Pattern Recognition

### **Keywords:**

Convolutional Neural Network, Saliency Map, Explainable AI,  
Explainability of Deep Networks, Clustering

### **Implementation Software and Hardware:**

MacOS 10.14.4, python 3.7.1, tensorflow 1.13.0, pytorch 1.0.1,  
keras 2.2.4

## **Acknowledgement**

My deepest gratitude goes first and foremost to Professor Leong Hon Wai, my supervisor, for his constant advice and guidance. He has walked me through many scientific research methodologies, presentation skills, and much background information in various CS fields from which I benefit a lot. Without his consistent and illuminating instruction, my project could not have reached its present form.

Second, I would like to express my heartfelt gratitude to my co-supervisor Dr. Yen Kaow Ng, who has suggested a very exciting and interesting research topic together with prof. Leong. Through frequent email contact with him, I have a clearer directions to undertake my experiments and deeper understanding of the problem I try to solve.

I would also like to thank all NUS SOC NGNE coordinators for giving us this opportunity to undertake such a wonderful project here in NUS and for taking care of our coursework and research.

Then my thanks would go to my fellows from our research group — Li Zhen, Jeremiah, Yan Hao, Terence — who delivered many wonderful speeches and gave me much constructive advice to my project. I also need to thank Su Xuan from my laboratory for providing so many vital suggestions in my code.

Last my thanks would go to my beloved family and friends for their loving considerations and great confidence in me all through these years.

# TABLE OF CONTENTS

<b>TITLE</b>	<b>i</b>
<b>ABSTRACT</b>	<b>ii</b>
<b>ACKNOWLEDGEMENT</b>	<b>iii</b>
<b>TABLE OF CONTENTS</b>	<b>iv</b>
<b>1 project Background</b>	<b>1</b>
1.1 motivation and background	1
1.2 Problem statement	1
1.3 Project Objective	2
1.4 Impact, Significance and Contribution	3
<b>2 Literature review</b>	<b>3</b>
2.1 Artificial Neural Network	3
2.2 Convolutional Neural Network	4
2.3 Architectures of CNN	5
2.3.1 LeNet	5
2.3.2 AlexNet	6
2.3.3 VGG	6
2.3.4 GoogLeNet	6
2.3.5 ResNet	7
2.3.6 DenseNet	8
2.3.7 SqueezeNet	9
2.4 XAL (Explainable AI)	9
2.4.1 Back Propagation Methods	10
2.4.2 Grad-CAM and Guided Grad-CAM	11
2.4.3 Layer-wise Relevance propagation ( LRP )	11
<b>3 Workflow of Research</b>	<b>12</b>
3.1 Derive Heat maps from different CNN architecture	12
3.1.1 CNN architectures	12
3.1.2 Explanation methods	12
3.1.3 Sample images	13
3.2 Compare between heat maps across CNN architectures	13
3.2.1 Calculate similarity matrix across CNN architectures	13
3.2.2 Perform clustering based on the similarity matrix	15

3.3 Evaluate the quality of heat maps given by different CNN architectures	15
<b>4 Results and Discussion</b>	<b>16</b>
4.1 Explainability of CNN Model is Influenced by its Architecture	16
4.1.1 Guided Backprop	16
4.1.2 Deconvnet	17
4.1.3 Grad-CAM	18
4.1.4 Guided Grad-CAM	19
4.1.5 Vanilla BP	19
4.1.6 Results and Summary of Findings	21
4.2 Explore the Relationships of Explainability across Different Architectures	22
4.2.1 Guided Backprop	22
4.2.2 Deconvnet	23
4.2.3 Grad-CAM	24
4.2.4 Guided Grad-CAM	26
4.2.5 Summary	28
4.2.6 Discussion of Patterns	28
4.3 Evaluate the Quality of Heat Maps Derived from Different Architectures	31
4.4 Further Discussion for the Perturbation Test	32
<b>5 Conclusions</b>	<b>33</b>
5.1 Summary of Findings	33
5.2 Open Questions	33
5.3 Possible Future Work	34
<b>Bibliography</b>	<b>35</b>

# 1 project Background

## 1.1 motivation and background

Since Alex Krizhevsky and Geoffrey E. Hinton introduced deep convolutional neural networks (CNN) to image classification task,[1] the era of deep learning has yet formally started. Take the famous image classification competition—ImageNet Large Scale Visual Recognition Challenge (ILSVRC) for example, since CNN outperformed the tradition computer vision techniques for the first time in 2012, the winner of ILSVRC competition has never been taken place by methods other than CNN. ResNet, a CNN architecture introduced by [2] in 2015 has swept 1st place in all five main tracks of ILSVRC and COCO 2015 competitions, achieving only 3.6% top 5 error in image classification task, which is even better than human performance.

Besides image classification, CNN and other deep neural networks have also enabled unprecedented breakthroughs in other Computer Vision tasks such as semantic segmentation (segment the detected category)[3], image captioning (describe the image in natural language)[4] and more recently, visual question answering[5] (answer questions in natural language about an image), human action recognition [6,7] and some non-CV tasks, such as Natural Language Processing[8,9], even in physics.[10]

Great classification accuracy although the CNN model have achieved in last few years, CNN still has a fatal drawback — lack of explainability. CNN is still a non-transparent “Black box” due to its high non-linear structure, i.e. we are still curious about how the CNN model arrive at their decision when fed with the input data to be considered. The lack of explainability can sometimes be a very serious drawback when CNN-based product comes to daily life. For example, In medical diagnosis, it would be irresponsible to trust the prediction results and accept treatment advice given by CNN by default. With the first fatality happening with Tesla autonomous car in 2016, understanding how CNN models work might be a legal requirement. Faced with all the tricky issues introduced by non-transparency of our model, we make a plea for a better understanding of the decisions made by CNN and call for a much more transparent deep convolutional neural network model.

Recently, a lot of methods has been proposed to open up the black-box to help people better understanding, visualising and interpreting deep learning models. However, the most direct and popular way is visualising CNN by heat map, which quantitatively show the importance of each individual pixel in the input image with regards to the classification decision made by this readily trained model. The basic logic of how heat map suggest the explainability of CNN model is shown in figure 1.1.

## 1.2 Problem statement

It has been reported by [11] that the explainability of CNN is related to the model parameters and the labelling of the data that the model was trained on, i.e. transformations of the training data and model parameters will change the output heat map for some explainability methods.

However, it seems that the explainability of CNN model is not only related to the two variables mentioned above. [12][13] has shown the strength of randomly initialised models as classification priors. Moreover [14] suggests that randomly initialised networks trained on a single input can perform tasks like denoising and super-resolution without additional training data. All these results

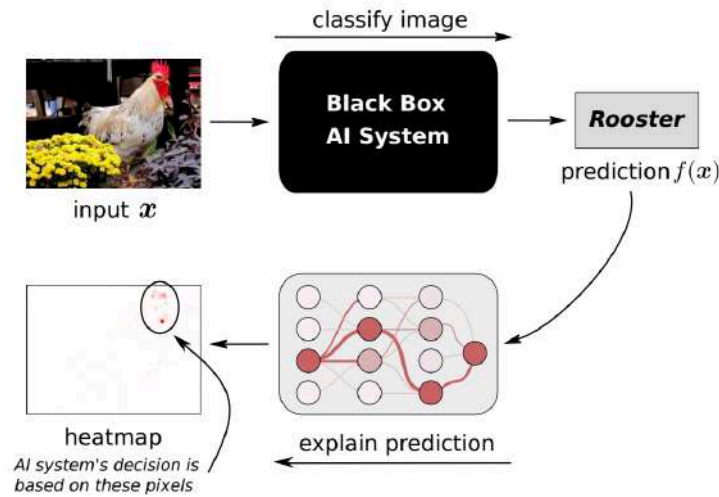


Figure 1.1 Heat map suggests Explainability of AI Model

point to one possible truth: architecture of a deep neural network also has an important effect on the representation derived from the network.

[39] has done some initial work to explore the connections between base model and the its explainability but fail to find the relationships of the CNN architectures and its explainability suggested by heat maps, which is ascribed to the naive clustering techniques by the author. So in our experiment, we use a much more sophisticated clustering algorithm — hierarchical clustering to explore the relationships between architectures and explainability of a CNN model.

### 1.3 Project Objective

Before stating the objective of our project, I have to do make the following declarations. First, the explainability of a model in our project is only suggested by the heat map derived from it. Second, the explainability of a model cannot be quantified, only can be compared with other models to find similarities and dissimilarities. Third, in our project, that two models or architectures have similar explainability is equals to that their heat maps given by the same explanation method are similar.

In the following paragraphs, we will elaborate the goal of our project.

The first part our work is to prove that CNN architectures do have some connections with its explainability. We want to combine subjective observation and objective clustering techniques to investigate the relationships between CNN models from various architectures and their heat maps in the context of several different explanation methods — Vanilla Backprop ( Gradient or Saliency Map) , Guided Backprop, Grad-CAM, Guided Grad-CAM and Deconvnet.

After ascertaining that there are indeed some differences between the heat maps of different CNN architectures and that there are some similarities within the same CNN architecture, we want to go further to discover how the heat maps—which suggest the explainability of CNN models—derived from different CNN architectures are different. By discovering the similarities and dissimilarities between heat maps derived from different architectures, we may have a better understanding of

some CNN architectures in terms of their explainability or gain some factors that contribute to explainability.

Then we want to go even further to evaluate and compare the quality of the heat map given by different CNN architectures by applying input perturbation analysis proposed by [15]. However, after serious consideration, we argue that the method proposed by [15] cannot be transferred to our case, then we discard the results gained in this section. Instead we propose a subjective standard to assess the explainability of a CNN model.

## 1.4 Impact, Significance and Contribution

As far as we know, most research papers working on the explainability of CNN model by heat maps either come up with new explanation methods or devise some test to compare them[11,15], no published publication has applied heat map to compare different CNN architectures, thus our work will be the first one to unveil the relationships between the CNN architectures and its explainability.

By the end of the project, we will have a better understanding of the relationships between CNN architectures and its explainability and we may find out some factors that contribute to explainability.

## 2 Literature review

### 2.1 Artificial Neural Network

An artificial neural network is an interconnected group of nodes, inspired by a simplification of neurons in a brain. Basically, neural networks receive an input (a single vector) and transform it through a series of hidden layers, each of which is made up of a set of neurons. Each neuron in a specific layer is fully connected to all the neurons in the previous layer while neurons in the same layer function completely independently and do not share any connections. The last fully-connected layer is called the “output layer” and in classification settings it represents the class score.

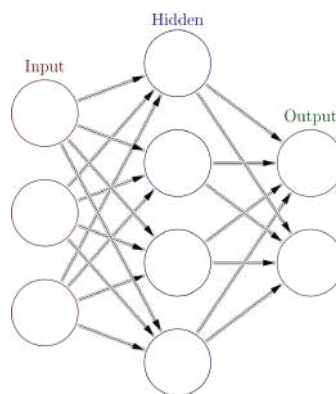


Figure 2.1 Artificial Neural Network

If we take close look at the inner mechanism of each neuron, it is easy to regard a single neuron as a linear function followed by a non-linear function. First it takes in whatever passed from the neuron from last layer to this neuron and sum it up with some predetermined weights, then use a non-linear



activation function ,e.g. sigmoid, ReLU to produce output which will be regarded as the input of the neurons from its next layer.

Thanks to the high non-linearity introduced by the activation function of each neuron, the artificial neural can theoretically approach any complex non-linear functions if the width, depth and activation function of the network arrives at a satisfactory harmony, however, highly complex non-linear functions always cannot escape the curse of overfitting.

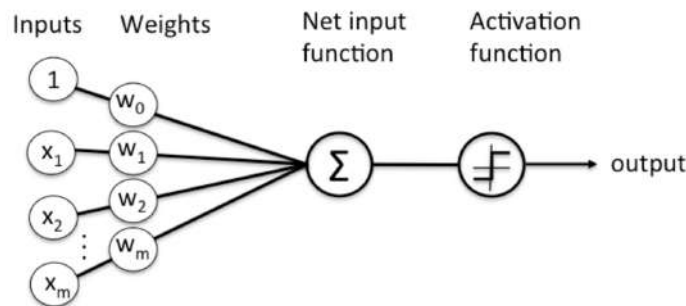


Figure 2.2 Structure of Neuron

## 2.2 Convolutional Neural Network

Convolutional Neural Networks are quite similar to ordinary Neural Network from the previous section, while the difference is convolutional architectures make explicit assumption that the inputs are images, which allows us to encode the spatial information of an image into the architecture. These then make the forward function more efficient to implement and vastly reduce the amount of parameters in the network.

Basically, a ConvNet is made up of layers. Every layer has a simple API: It transform an input 3D volume to an output 3D volume with some differentiable function that may or may not have parameters. There are three main types of layers in ConvNet architectures: Convolutional Layer, Pooling Layer and Fully-Connected Layer. Because fully connected layer has been introduced in last section, next two paragraphs will briefly introduce the convolutional layer and the pooling layer.

The CONV layer's parameters consist of a set of learnable filters. Every filter is small spatially (along width and height), but extends through the full depth of the input volume. During the forward pass, we slide (more precisely, convolve) each filter across the width and height of the input volume and compute dot products between the entries of the filter and the input at any position, after which we will get a 2-dimensional activation map that records the information at every spatial position in input image in the perspective of this specific filter. Intuitively, the network will learn filters that activate when they see some type of visual feature such as an edge of some orientation or a blotch of some colour on the first layer, or eventually entire honeycomb or wheel-like patterns on higher layers of the network. Now, we will have an entire set of filters in each CONV layer, and each of them will produce a separate 2-dimensional activation map. We will stack these activation maps along the depth dimension and produce the output volume.

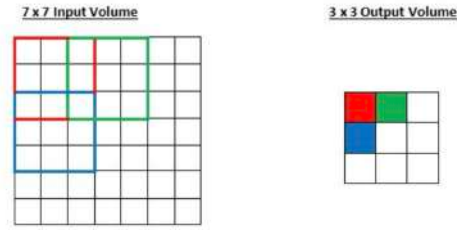


Figure 2.3 Convolve Operation

The function of pooling layer is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network, and hence to also control overfitting. The Pooling Layer operates independently on every depth slice of the input and resizes it spatially, using the MAX operation.

## 2.3 Architectures of CNN

The main purpose of our work is to investigate the relationships between the architectures and the explainability of CNN model, so in this section we will briefly review some of most important and famous CNN architectures. A very brief overview of CNN architectures can be seen in Figure 2.4.

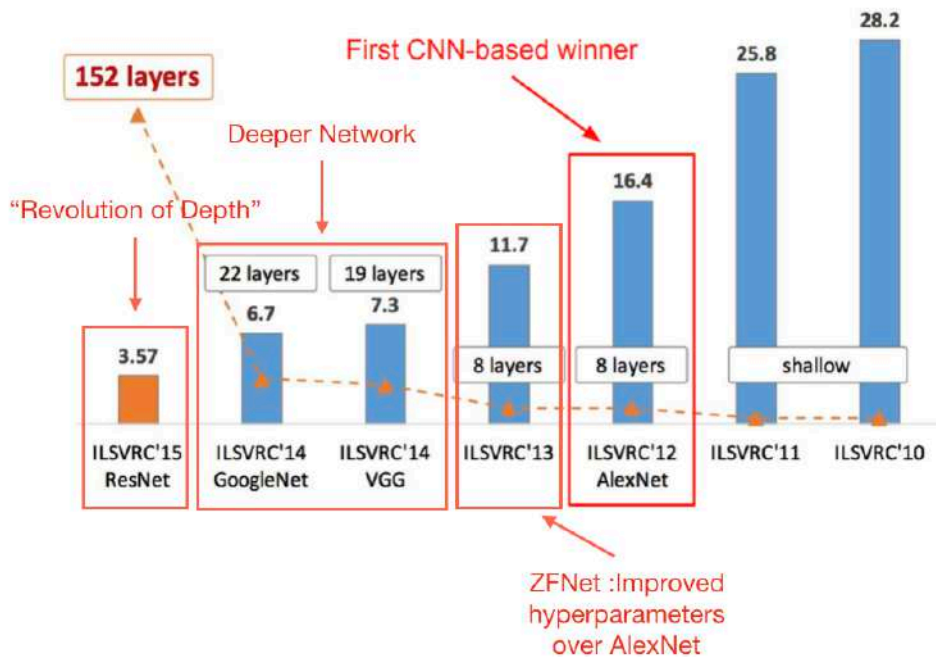


Figure 2.4 ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners

### 2.3.1 LeNet

LeNet [16] was proposed by Yann LeCun in 1998 to recognise hand digits, which was the first successful application of convolutional network. It only consists of 3 convolution layers, 2 pooling layers, 2 fully connected layers, but performed well on classifying digits without being affected by small distortions, rotation, and variation of position and scale.

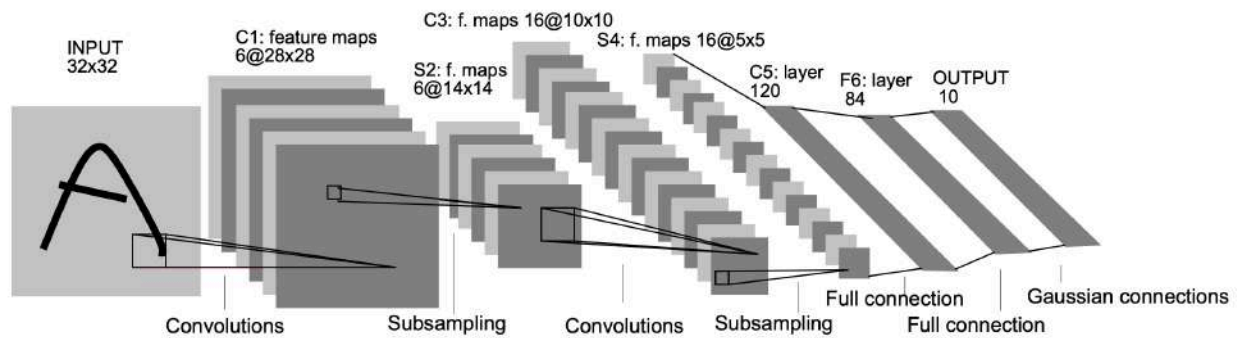


Figure 2.5 Architecture of LeNet

### 2.3.2 AlexNet

The emergence of AlexNet [1] completely popularised convolutional networks in Computer Vision. In ImageNet ILSVRC challenge 2012 AlexNet significantly outperformed the second runner-up (top 5 error of 16% compared to runner-up with 26% error). The architecture of AlexNet is quite similar to that of LeNet, but was deeper, bigger, including 5 convolution layers and 3 fully connected layers and have three convolution layers continuously stacked on top of each other, compared of which every CONV layer of LeNet was always followed by a pooling layer.

AlexNet exploited ReLU non-saturating activation function to improve convergence rate by alleviating the problem of vanishing gradient which is introduced by Sigmoid activation function when the network is deep. Dropout layer was also been added to avoid overfitting. LRN normalization layer was also used to create a competitive mechanism for local neurons: neurons with less feedback were suppressed, and neurons with larger feedback were amplified, which enhances the generalization ability of the model. The network was spread across 2 GPUs, half the neurons ( feature map ) on each GPU due to the lack of GPU memory at that time.

### 2.3.3 VGG

The main difference between VGG [17] and AlexNet is that VGG has smaller filters but deeper networks, and one of the biggest contributions of this paper is showing that the depth of the network is a crucial component for good performance. (Figure 2.6) Their final best network contains 16 CONV/FC layers and, appealingly, features an extremely homogeneous architecture that only performs 3x3 convolutions and 2x2 pooling from the beginning to the end.

Though have achieved good accuracy, VGG needs huge memory and 140M parameters which were mainly introduced by the first fully connected layer. Since then, fully connected layers have been found to be able to removed with no performance downgrade, thus significantly reducing the number of unnecessary parameters in later CNN architectures.

### 2.3.4 GoogLeNet

GoogLeNet [18] has deeper architecture with total 22 layers though none of which is fully connected layers. “Inception” module was creatively introduced to increase the width of the model, enabling parallel filters operation on input from previous layer. Thanks to the exploitation of



Figure 2.6 Comparison between Architectures of AlexNet and VGG

“Inception” module and replacing fully connected layers with Average Pooling Layer at the top of the ConvNet, GoogLeNet has eliminated a large amount of parameters that do not seem to matter much, thus increase computational efficiency a lot.

The basic structure of the Inception Module has four components—1\*1 convolution, 3\*3 convolution, 5\*5 convolution, 3\*3 max pooling. Finally, the results of the operations of these four components are combined on the channel. By extracting and merging the information of different scales of the image through multiple convolution kernels, the image can be better characterised.

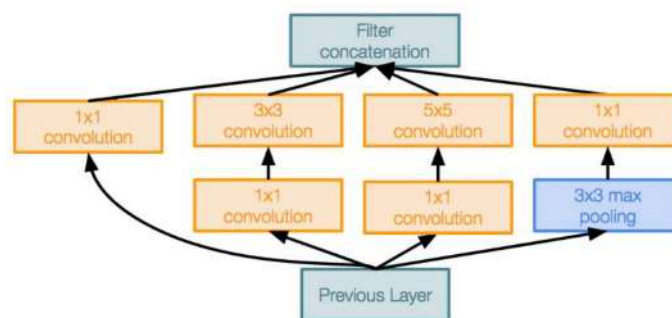


Figure 2.7 Inception module

### 2.3.5 ResNet

ResNet [19], ILSVRC’15 classification winner ( 3.57% ) top 5 error is a very deep network using residual connections. The reason why previous architectures are not so deep is that when we continue stacking deeper layers on a “plain” CNN, the model performs worse on both training and test error, which suggested the bad performance is not caused by overfitting. Based on the hypothesis that the problem described above is an optimization problem and deeper models are

harder to optimise, [19] gives a solution—use network layers to fit a residual mapping instead of directly trying to fit a desired underlying mapping.

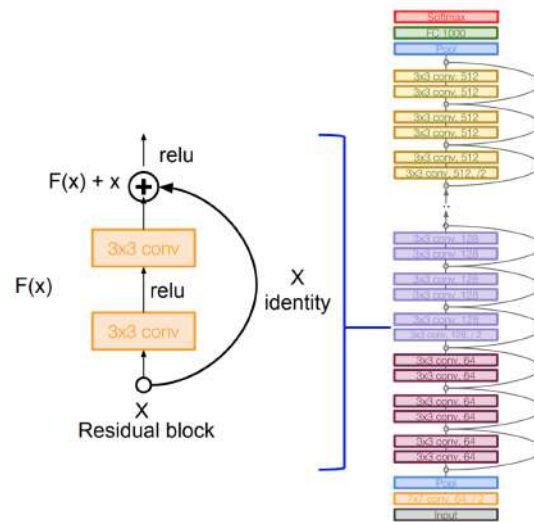


Figure 2.8 Residual Block and Architecture of ResNet

Based on the hypothesis and solution above, [19] proposed a so called “Residual Block”, each of which has two  $3 \times 3$  convolutional layers. ResNet just stack many residual blocks with additional convolutional layer at the beginning. Moreover, ResNet periodically doubles the number of filters and downsample spatially using stride 2.

### 2.3.6 DenseNet

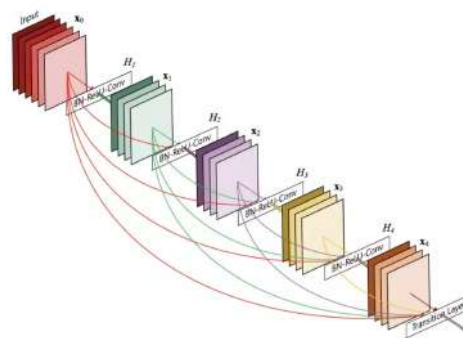


Figure 2.9 Architecture of DenseNet

Basically, DenseNet [20] follows the basic idea of ResNet which builds skip connection between previous layers and back layer. However, each layer in the DenseNet is directly connected to its front layer (dense connection) to improve the utilization of features (feature reuse). What's more, each layer of the network is designed to be narrow, that is, the number of output channels of the convolution is usually small, only a few dozen, which achieves compression of computational cost.

### 2.3.7 SqueezeNet

The basic unit of the SqueezeNet [21] network is a modular convolution operation called the Fire module. The Fire module mainly consists of two layers of convolution operations: one is the squeeze layer with  $1 \times 1$  convolution kernel; the other is the expand layer with  $1 \times 1$  and  $3 \times 3$  convolution kernels. The basic structure of the Fire module is shown in Figure 2.10. The entire SqueezeNet is built using the Fire basic modules.

SqueezeNet achieves AlexNet-level accuracy on ImageNet with 50x fewer parameters. Additionally, with model compression techniques, we are able to compress SqueezeNet to less than 0.5MB (510× smaller than AlexNet).

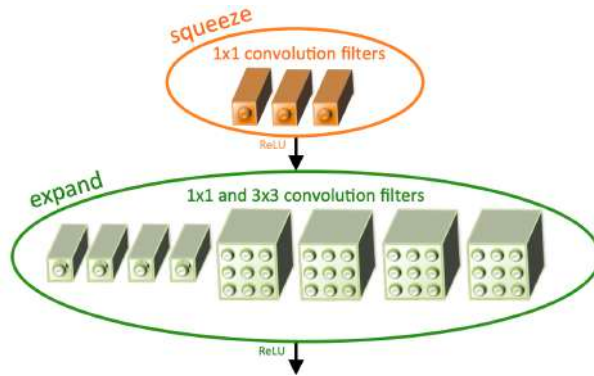


Figure 2.10 Fire Module of SqueezeNet

## 2.4 XAL (Explainable AI)

XAL[22,23,24] tackles the important problem that complex machines and algorithms often cannot provide insights into their behaviour and thought process. XAL allows users and parts of the internal system to be more transparent, providing explanations of their decisions in some level of detail. These explanations are important to ensure algorithmic fairness, identify bias/problems in the training data, and to ensure that the algorithms perform as expected.

Recently, a lot of methods has been proposed to open up the black-box to help people better understanding, visualising and interpreting deep learning models. Most of these methods are post-hoc explainability methods, i.e. we are given a readily trained model and we aim to understand what the model predicts (e.g. categories) in terms what is readily interpretable (e.g. the input variables). Post-hoc explainability methods should be separated with those who just incorporate interpretability into the structure of the model to build so called explainable models, such as interpretable CNNs[25], capsule networks[26], interpretable R-CNNs[27] and the InfoGAN[28].

Among post-doc explainability methods, there are some methods that disentangling CNN representations into explanatory graphs[29,30] and decision trees[31], i.e. they disentangle features in conv-layers of a pre-trained CNN and represent the semantic hierarchy hidden inside a CNN using a graphical model or decision tree model. However, among all the above, the most direct and popular way to understand and explain a CNN model is by visualising its representations. And this report only focus on several important visualisation techniques for CNN which is applied in our project.



## 2.4.1 Back Propagation Methods

In this report, we collectively refer three CNN visualisation methods— Back propagation[32], Deconvolution[33], and Guided Back propagation[34] — as Back Propagation methods. All this three methods aim to visualise the part of a specific input image that most activates a given neuron. In the case of explaining the prediction outcome of a CNN classification model, the “given neuron” should be the neuron corresponding to the output category, thus the visualisation or “reconstructing” of this specific neuron can be regarded as a heat map explaining why the model arrive at this specific output decision.

All these three methods generate the visualisation by using a backward pass of the activation of a single neuron after a forward pass through the network. However, the difference across them is how

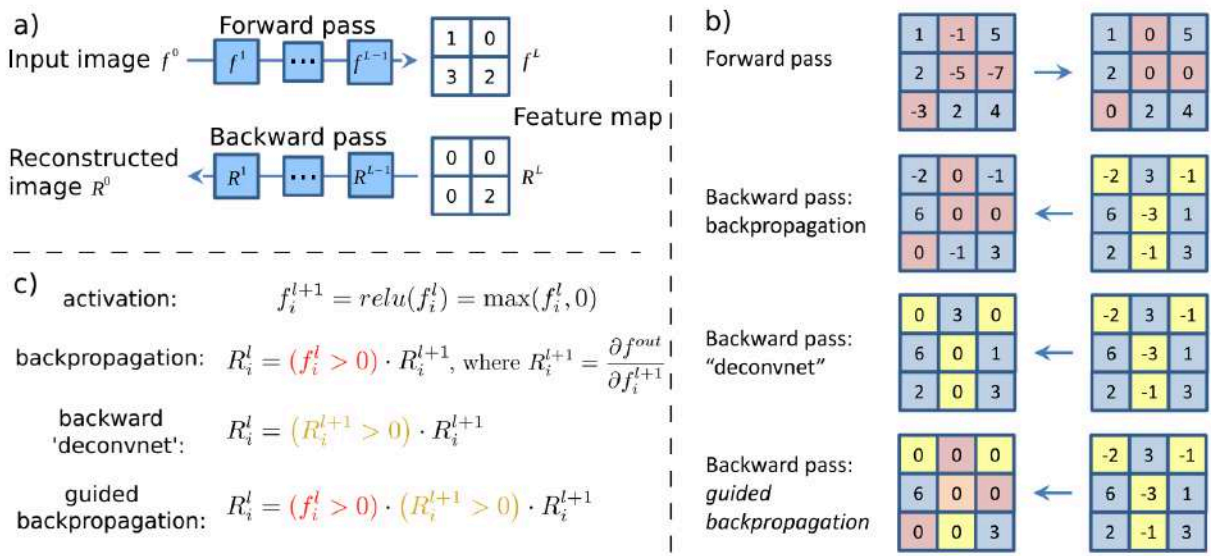


Figure 2.11 three methods differ in how they handle back propagation through ReLU

they handle backpropagation through rectified linear (ReLU) nonlinearity, the detailed difference are shown clearly in the figure 2.11.

When we use these three different methods to generate heat maps for the same input image with the same output category, we will find from Backprop, Deconvnet to Guided BP, the heat map become much sharper and less noisy, as is shown figure 2.12

The reason for this is that in Guided-Backprop, during ReLU backward in addition to what is suppressed in simple backprop, we additionally suppress the negative gradients. Negative gradients at a particular ReLU neuron, state that this neuron has a negative influence on the class that we are trying to visualize.

During backpropagation there are paths that have positive influence and some that have negative influence, and these end up cancelling out in a weird interference pattern, causing gradients to seem noisy. Whereas in Guided Backpropagation, we only keep paths that lead to positive influence on the class score, and suppress the ones that have negative influence, leading to much cleaner looking images.

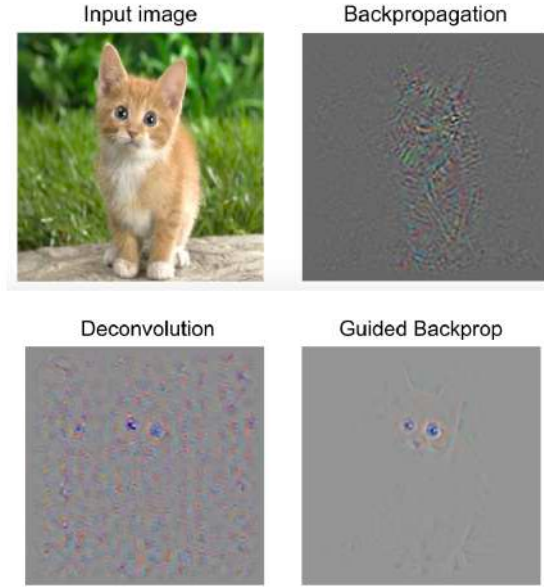


Figure 2.12 Backprop is much noisy, Guided Backprop is much sharper

## 2.4.2 Grad-CAM and Guided Grad-CAM

Selvaraju et al. proposed a method called Gradient-weighted Class Activation Mapping (Grad-CAM) [35], that can be used to explain and visualise the regions that contributed to each of the output class of a neural network. They propose to generate a coarse heat map to highlight the pixel that makes the neural network predict the class based on the last convolutional layer in the neural network.

While Grad-CAM visualisations are class-discriminative and localise relevant image regions well, they lack the ability to show fine-grained importance like pixel-space gradient visualisation methods (Guided Backpropagation and Deconvolution). In order to combine the best aspects of both, [35] create a new visualisation technique called Guided Grad-CAM by fusing Guided Backpropagation and the Grad-CAM visualisations via a point-wise multiplication. This visualisation is both high-resolution and class-discriminative.

## 2.4.3 Layer-wise Relevance propagation ( LRP )

A recent technique called Layer-wise relevance propagation (LRP) [36] explains the classifier's decisions by decomposition. Mathematically, it redistributes the prediction  $f(\mathbf{x})$  backwards using local redistribution rules until it assigns a relevance score  $R_i$  to each input variable (e.g., image pixel). Basically, this method can suggest that how much does each pixel contribute to prediction. The key property of this redistribution process is referred to as relevance conservation and can be summarised as

$$\sum_i R_i = \dots = \sum_j R_j = \sum_k R_k = \dots = f(\mathbf{x})$$

This property says that at every step of the redistribution process (e.g., at every layer of a deep neural network), the total amount of relevance (i.e., the prediction  $f(\mathbf{x})$ ) is conserved. No relevance



is artificially added or removed during redistribution. The relevance scores  $R_i$  of each input variable determines how much this variable has contributed to the prediction. Thus, in contrast to sensitivity analysis, LRP truly decomposes the function value  $f(x)$ .

## 3 Workflow of Research

### 3.1 Derive Heat maps from different CNN architecture

To investigate the relationships between the explainability and architectures of CNN model, we have to include three variables in our experiment: Explanation Methods (CNN visualising techniques), CNN Architectures and Sample Images. More specifically, after defining a certain explanation method and a specific input sample image, we derived heat maps for different CNN architectures. Then we change a different explanation method and a different kind of sample image and repeat this process for many times.

#### 3.1.1 CNN architectures

Six different CNN architectures with different layers were used in our experiments, we will either use 13 models or 6 architectures in our experiments according to our purpose.

The total 13 models are listed in table 3.1

Architectures	Models
AlexNet	AlexNet
Inception V3	Inception V3
VGG	VGG16, VGG19
ResNet	ResNet50, ResNet101 ResNet152
DenseNet	DenseNet121, DenseNet169, DenseNet201
SqueezeNet	SqueezeNet1_0, SqueezeNet1_1

Table 3.1 CNN architectures and models used in experiment

We don't train CNN models by ourselves because there are already some pre-trained model available online to faster our research process. In our research, we utilize pre-trained CNN models from pytorch TORCHVISION.MODELS package, which are trained on ImageNet dataset and includes 21 different CNN models.

#### 3.1.2 Explanation methods

In our experiment, we applied 5 different explanation methods to evaluate the explainability of different CNN architectures — Vanilla Backpropagation, Grad-CAM, Guided Backpropagation, Guided Grad-CAM and Deconvnet.

Because we are able to get heat maps from any layer we want, so when performing GradCAM we have to specify the layer we want to visualise. For different architectures we get heat maps from different layers. For VGG, AlexNet, DenseNet and SqueezeNet, we get our heat maps from the last feature layer instead of classifier layer. For Inception and ResNet, because their architectures are not explicitly divided into feature and classifier two parts, we derived their heat maps from the layer rightly before the Fully Connected Layer or Average Pooling Layer—‘layer4’ for ResNet and ‘mixed\_7c’ for Inception V3.

Architecture	AlexNet	Inception V3	ResNet	DenseNet	VGG	SqueezeNet
<b>Layers to derive GradCam heatmap from</b>	features	Mixed_7c	layer4	features	features	features

Table 3.2 different layers we derived heat maps from for different architectures

### 3.1.3 Sample images

We totally used 20 images which falls into 7 categories as our input sample images, we derive heat maps for each of these 20 samples images with different explanation methods and different CNN architecture. Table 3.3 shows all sample images we use in our experiment.

In the first round of our experiment, three sample images were used—King Snake 0, King Snake 1, Sea Snake—to derive heat maps from all 13 CNN models that are mentioned in 3.1.1.

In the second round of our experiment, all 16 sample images from 4 categories — cab, kelpie, kit fox and minibus will be our target image to derive heat maps from, however this time, we only look at 6 different CNN architectures we mentioned in 3.1.1, i.e. AlexNet, Inception V3, ResNet 152, DenseNet, VGG19, SqueezeNet1\_0.

In the third round of our experiment, we want to propose a subjective standard to assess the explainability of a CNN model, so this time we only look at the sample image “Broom” and derived heat maps from all 13 CNN models because on this specific image “Broom” several CNN models give different class prediction outcome, which will help us illustrate our idea.

## 3.2 Compare between heat maps across CNN architectures

### 3.2.1 Calculate similarity matrix across CNN architectures

We definitely can compare just by going through heat maps one by one and observing similarities and differences by our human eyes, but the patterns given by this subjective observation is not convincing enough. Thus, a objective similarity measures between images is needed, moreover, this measure should be consistent with human intuition because it is to human that heat map explain its prediction decision.








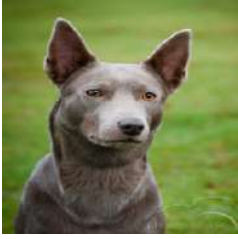












	Round 1	Round 2	Round 3	Round 4
cab				
kelpie				
kit fox				
minibus				
	King Snake 0	King Snake 1	Sea Snake	Broom
Snake & Broom				

Table 3.3 sample images we use in our experiment

In our work, SSIM (Structural Similarity Index) [37] was applied to measure the similarity of two heat maps. SSIM is an image similarity evaluation standard based on the degradation of structural information. SSIM is in line with human intuition, because human visual perception is highly adapted for extracting structural information from a scene.

After obtaining certain heat maps of different CNN architectures for a specific sample image, we applied SSIM to assign each pair of architectures a similarity score which is between 0 and 1. Then we will obtain a similarity matrix across all CNN architectures taken into consider.

### 3.2.2 Perform clustering based on the similarity matrix

Before clustering, we have to transform the similarity matrix to distance matrix because all clustering methods is based on distance instead of similarity. Based on the hypothesis that the more similar two images are, less distance is between them, we calculate the distance matrix for our heat maps derived from different CNN architectures as follow:

$$\text{distance\_matrix}(i,j)=1-\text{similarity\_matrix}(i,j)$$

We then performed hierarchical/agglomerative clustering to our architectures, during which we applied “ward” for calculating the distance between the newly formed cluster because this method gives us good clustering results. Finally, We use dendrogram graph to show the result of our hierarchical clustering.

### 3.3 Evaluate the quality of heat maps given by different CNN architectures

After the experiment of 3.1 & 3.2, we may have a deeper understanding of the difference across heat maps derived from different CNN architectures, however, in this section, we want to take one step further to determine which CNN architecture can give us a better heat map.

We transferred a method [38] proposed to compare the quality of heat maps from different explanation methods to our task to compare heat maps from different CNN architectures.

The method is based on the following three ideas:

- (1) The perturbation of input variables which are highly important for the prediction leads to a steeper decline of the prediction score than the perturbation of input dimensions which are of lesser importance.
- (2) Explanation methods such as Sensitivity Analysis and LRP provide a score for every input variable. Thus, the input variables can be sorted according to this relevance score.
- (3) One can iteratively perturb input variables (starting from the most relevant ones) and track the prediction score after every perturbation step. The average decline of the prediction score (or the decline of the prediction accuracy) can be used as an objective measure of explanation quality of this specific heat map, because a large decline indicates that the heat map as successful in identifying the truly relevant input variables.

## 4 Results and Discussion

### 4.1 Explainability of CNN Model is Influenced by its Architecture

We gained this result from the first part of our experiment, that is, by applying 5 different saliency methods, we derived heat maps of 13 different CNN models for 3 “snake” sample images. However, we cannot include all our results from 3 sample images in our report, so for each explanation method, we only choose the heat maps and clustering results of one sample image to show our findings. And all these findings hold true for all three sample images.

#### 4.1.1 Guided Backprop

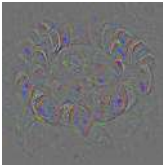
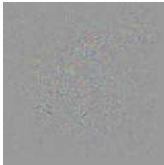












Saliency Maps derived from GuidedBP — King Snake 1						
AlexNet	Inception_V3	SqueezeNet 1_0	SqueezeNet 1_1	VGG13	VGG16	VGG19
						
DenseNet 121	DenseNet 169	DenseNet 201	ResNet 50	ResNet 101	ResNet 152	Original
						

Table 4.1 Heat Maps derived from Guided BP for King Snake 1

From our subjective inspection on heat map of sample image king snake 1 (shown in Table 4.1), we may find that different architectures give us roughly different style of heat maps. We then perform Hierarchical Clustering to support our observation. From the clustering results as shown in Figure 4.1, we can observe that the distances of heat maps given by the same architecture (but different layers) are short while those given by different architectures are relatively long, which is in line with our intuition.

Another interesting finding from clustering results given by guided BP is : although the heat maps of VGG, ResNet, DenseNet, SqueezeNet are similar to each other in some specific sample image, the heat maps of inception V3 and AlexNet are quite different from those of any other architectures for all three sample images. When we go back to observe the original heat maps, we have indeed found that the heating maps of these two architectures hold certain kinds of stable properties, which can to some extent prove that the explainability of CNN model is indeed influenced by CNN architecture.

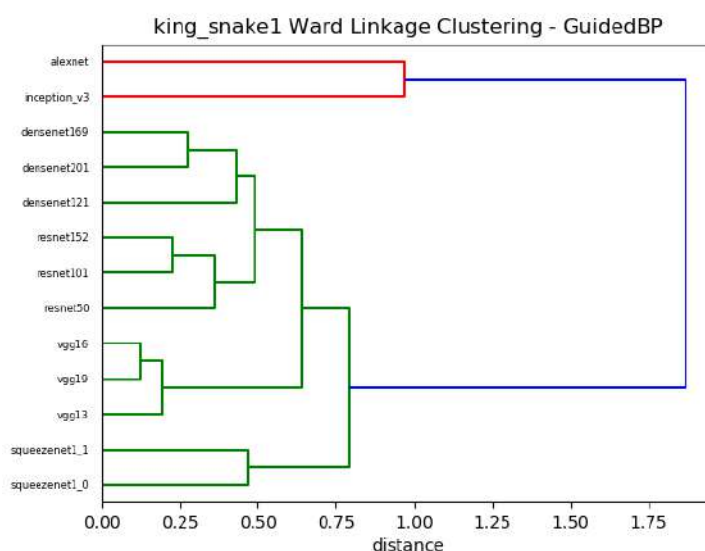


Figure 4.1 Clustering Results of heat maps derived from Guided BP for King Snake 1

#### 4.1.2 Deconvnet

### Saliency Maps derived from DeconvNet—KingSnake 0

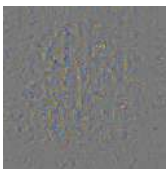













AlexNet	Inception_V3	SqueezeNet 1_0	SqueezeNet 1_1	VGG13	VGG16	VGG19
						
DenseNet 121	DenseNet 169	DenseNet 201	ResNet 50	ResNet 101	ResNet 152	Original
						

Table 4.2 Heat Maps derived from DeconvNet for King Snake 0

By inspecting the heat maps given by Table 4.2 by eyes, we have a surprising finding that each of 6 CNN architectures seem to have their own patterns in their deconvnet heat maps, and the clustering results strongly support our findings by subjective observation—distances of heat maps given by the same architecture are short while those given by different architectures are significantly long. The clustering results (Figure 4.2) strongly support our findings.



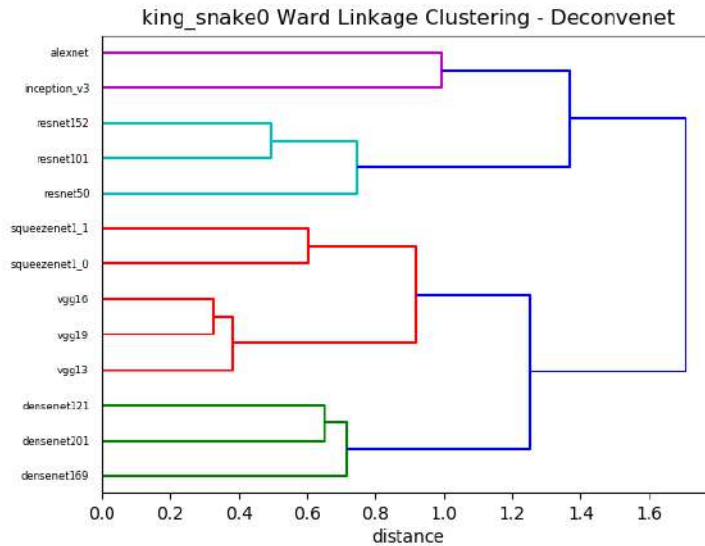


Figure 4.2 Clustering Results of heat maps derived from DeconveNet for King Snake 0

### 4.1.3 Grad-CAM

Saliency Maps derived from GradCAM – Sea Snake						
AlexNet	Inception_V3	SqueezeNet 1_0	SqueezeNet 1_1	VGG13	VGG16	VGG19
DenseNet 121	DenseNet 169	DenseNet 201	ResNet 50	ResNet 101	ResNet 152	Original

Table 4.3 Heat Maps derived from GradCAM for Sea Snake

The Grad-CAM heat maps derived from Sea Snake shown in Table 4.3 contains much more visible information than other kinds explanation methods we investigate, from which we get obtain patterns like this:

1. It is obvious that models with the same architecture give similar heat map, although VGG13 is an exception.

2. DenseNet, ResNet and Inception V3 highlights almost the same position of the image—middle section of the snake, which may suggest the explainability of those three architectures is similar, although the highlight area of Inception V3 is much bigger than ResNet and DenseNet.

3. The area SqueezeNet highlight is much smaller but gives us a very convincing explanation to its prediction—SqueezeNet classifies this image as sea snake by detecting the head and tail of the snake.

4. AlexNet highlights the head of the snake and gives a significantly different heat maps from other architectures.

The clustering results also positively support our observations as show in Figure 4.3.

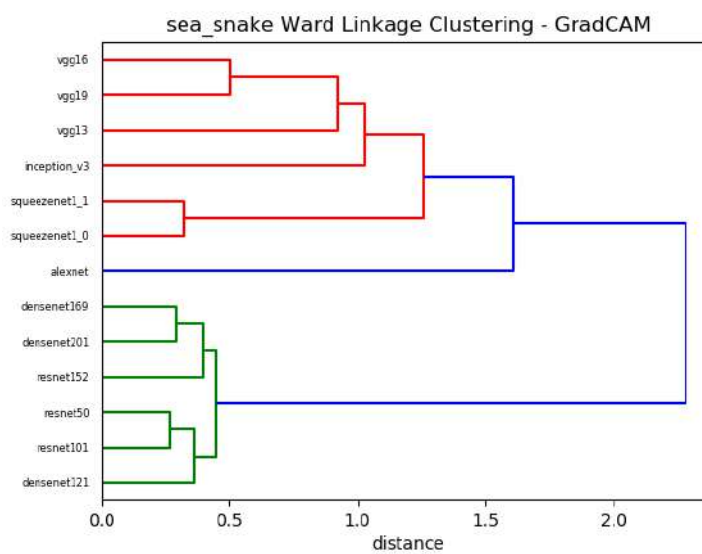


Figure 4.3 Clustering Results of heat maps derived from Grad-CAM for Sea Snake

#### 4.1.4 Guided Grad-CAM

From Visual Observation from Table 4.4 and Clustering Results Figure 4.4 the patterns behind heat maps of Guided GradCAM are as follows:

1. Different architectures, different heat maps; Same architecture, Similar heat maps.

2. DenseNet, ResNet and VGG give similar heat maps

3. AlexNet and Inception V3 is quite different other architectures.

#### 4.1.5 Vanilla BP

The vanilla BP heat map does not suggest any obvious patterns either from the perspective of observation by eyes Table 4.5 or from the clustering results Figure 4.5. Even models belonging to the same CNN architecture with different layers such as DenseNet 121 and DensNet 169 have so



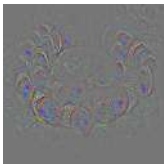













Saliency Maps derived from Guided GradCAM— King Snake 1						
AlexNet	Inception_V3	SqueezeNet 1_0	SqueezeNet 1_1	VGG13	VGG16	VGG19
						
DenseNet 121	DenseNet 169	DenseNet 201	ResNet 50	ResNet 101	ResNet 152	Original
						

Table 4.4 Heat Maps derived from Guided Grad-CAM for King Snake 1

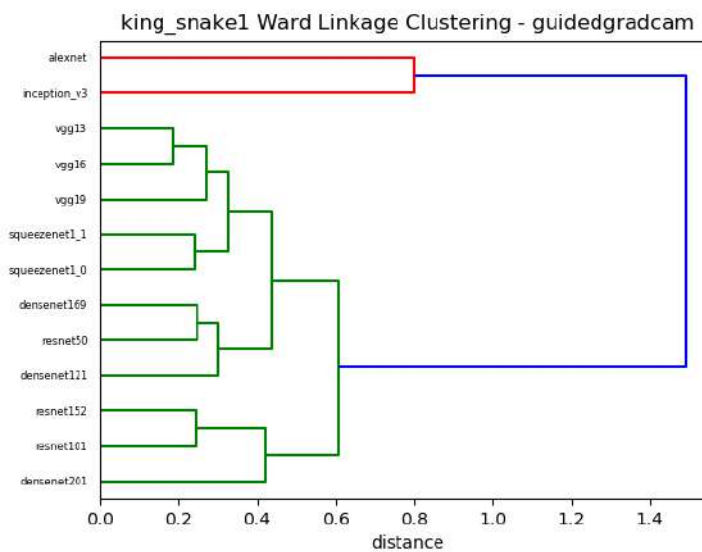


Figure 4.4 Clustering Results of heat maps derived from Guided Grad-CAM for King Snake 1

much difference at all. One possible reason for this phenomena is that Vanilla BP is the first explanation method proposed in 2013 by [32], which has been attacked by many researchers because it always give too noisy heat map ,thus it is not a good method to suggest the explainability of CNN models. Based on this fact, we discard this explanation method in our following experiment.

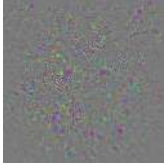



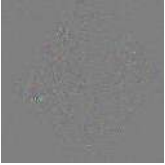



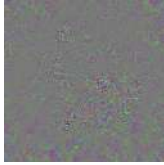
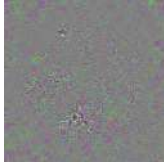




Saliency Maps derived from VanillaBP — King Snake 0						
AlexNet	Inception_V3	SqueezeNet1_0	SqueezeNet1_1	VGG13	VGG16	VGG19
						
DenseNet_121	DenseNet_169	DenseNet_201	ResNet_50	ResNet_101	ResNet_152	Original
						

Table 4.5 Heat Maps derived from Vanilla BP for King Snake 0

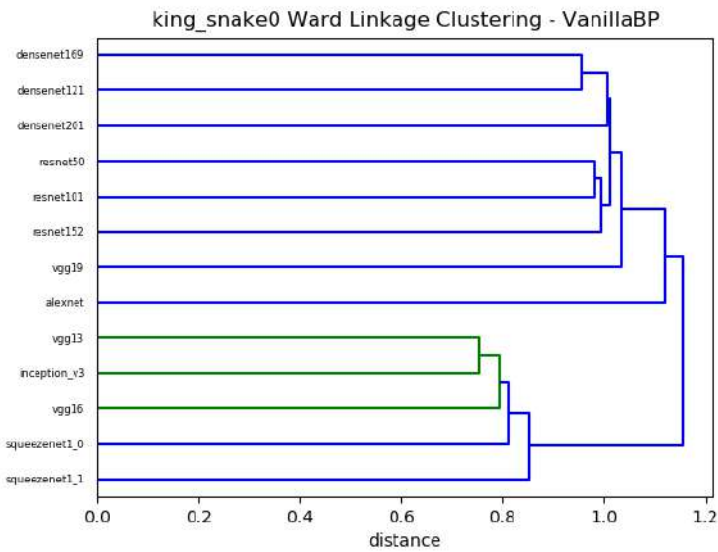


Figure 4.5 Clustering Results of heat maps derived from Vanilla BP for King Snake 0

#### 4.1.6 Results and Summary of Findings

In the first part of our experiment, almost all explanation methods generate similar heat maps for models from the same architecture while produce different heat maps for models from different architectures, based on which, we can come to a conclusion that explainability of CNN model is indeed influenced by its architecture.

What is more, we found some interesting patterns of heat maps for different explanation methods:

For Guided Backprop, AlexNet and Inception V3 give a significantly different heat maps from any other architectures. (AlexNet and Inception V3 are also vastly different)

For DeconvNet, each architecture has a special heat map style, i.e. the distance from one architecture to any other one is always long, there are no two architectures give us similar heat maps.

For GradCAM, 1) Inception V3, ResNet and DenseNet seem to give similar heat maps  
2 ) AlexNet highlight a vastly different position from any other architectures.  
3) SqueezeNet tend to highlight much smaller space than any other architectures.

For Guided GradCAM, 1) DenseNet and ResNet give similar heat maps.  
2) AlexNet and Inception V3 are quite different from other architectures.

For Vanilla Backprop, because its heat maps are quite noisy and models in the same architectures don't give similar heat maps, which is not in line with our intuition, thus we suppose that this method is not good enough to measure the explainability of CNN models, thus we discard this explanation method in our follow-up experiments.

## **4.2 Explore the Relationships of Explainability across Different Architectures**

In the first part of our experiment, we have concluded that there do exists some relationships between the architecture and the explainability of CNN model. However, in this part ,we want to explore the relationships across different CNN architectures in the aspect of explainability.

In this section, we are no longer interested in the relationships of explainability of models from the same architecture with different layers, because the result of last section tells us that explainability doesn't vary a lot between models in the same architectures. Thus we only keep one most representative one in each architecture—VGG19, ResNet152, SqueezeNet1\_0, DenseNet161, AlexNet, Inception V3—to facilitate us to explore relationships between different architectures in terms of explainability.

From 4.1, we do find patterns for different explanation methods, however, all these patterns are derived from only 3 “snake” sample images which make our findings not so convincing. So in this section, we take more sample images into consider, i.e. 16 sample images which fall into 4 categories—cab, kelpie, kit fox, minibus. However because 4 sample images (coloured with red in the heat map in Table 4.7 - Table 4.12) have been assigned wrong labels by at least one architecture, we finally use 12 sample images to support or reject our patterns proposed in 4.1.

### **4.2.1 Guided Backprop**

11/12 sample images show us that Inception V3 was the last and the only one to be incorporated into the cluster formed by other 5 architectures, which strongly support that Inception V3 does generate a vastly different heat map from any other architectures.

AlexNet was the second to last to be incorporated into cluster formed by other 4 architectures in 6/12 sample images, rightly before Inception V3, which to some extent suggests AlexNet generate notably different heat maps from DenseNet, ResNet, VGG and SqueezeNet.

**Saliency Maps derived from GuidedBP — cab**


















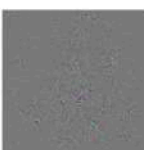










	Original	AlexNet	DenseNet16 1	Inception_V 3	ResNet152	SqueezeNet 1_0	VGG19
ca b0							
ca b1							
ca b2							
ca b3							

Table 4.6 Heat Maps derived from Guided BP for cab

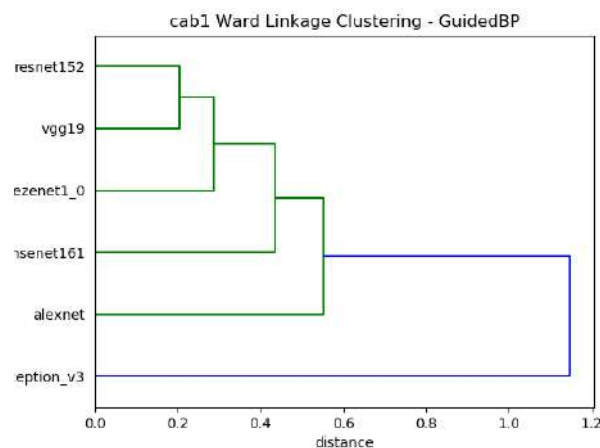


Figure 4.6 Clustering Results of heat maps derived from Guided BP for cab1

## 4.2.2 Deconvnet

We make the hypothesis that if the first clustering process occurs at the distance which is longer than half of the distance at which all architectures were clustered together in one group, we have the confidence to say that there are no two architectures give us similar heatmaps. Based on this

criteria, we went through the clustering results of deconvnet one by one, and found that only 1/12 sample image rejected our proposed patterns.


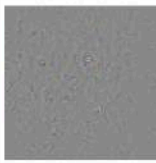

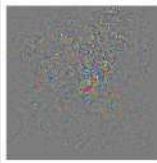
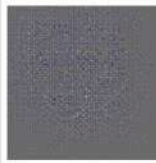



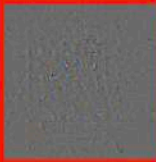

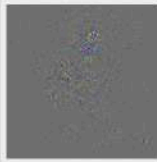
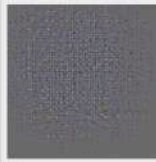
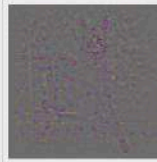

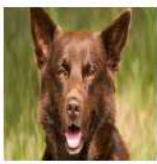


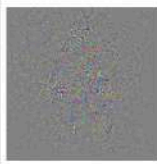
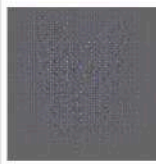





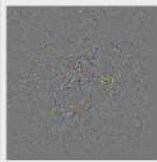



Saliency Maps derived from Deconvnet — kelpie							
	Original	AlexNet	DenseNet161	Inception_V3	ResNet152	SqueezeNet1_0	VGG19
kelpie0							
kelpie1							
kelpie2							
kelpie3							

Table 4.7 Heat Maps derived from DeconvNet for kelpie

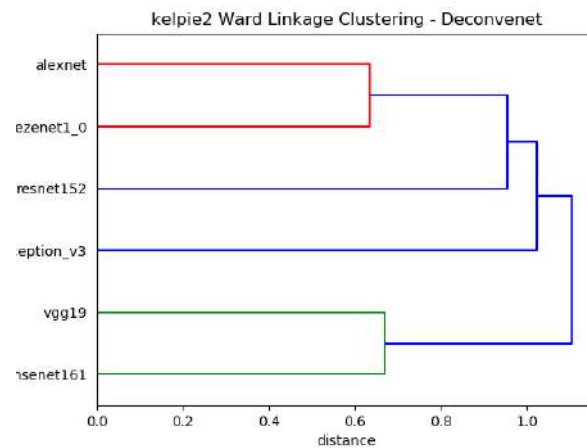


Figure 4.7 Clustering Results of heat maps derived from DeconvNet for kelpie 2

### 4.2.3 Grad-CAM

Inception V3, ResNet and DenseNet were the first three architectures clustered together for 9/12 sample images, which strongly supports our guess. From direct observation of heatmaps shown in figure x-x, we can also easily make this conclusion.



AlexNet seems to give very different heating maps in some special sample images, and the pixels which AlexNet highlight sometimes disobey our intuition. For example, in Kelpie 1, AlexNet highlight the legs of the dog, which seems not an obvious characteristic of a kelpie; in kit fox 0,2,3 and minibus 1, AlexNet mainly highlight the background instead of the kit fox or the minibus. The clustering results also support our findings, AlexNet was the last architecture to be clustered with others in 8 sample images out of 12.

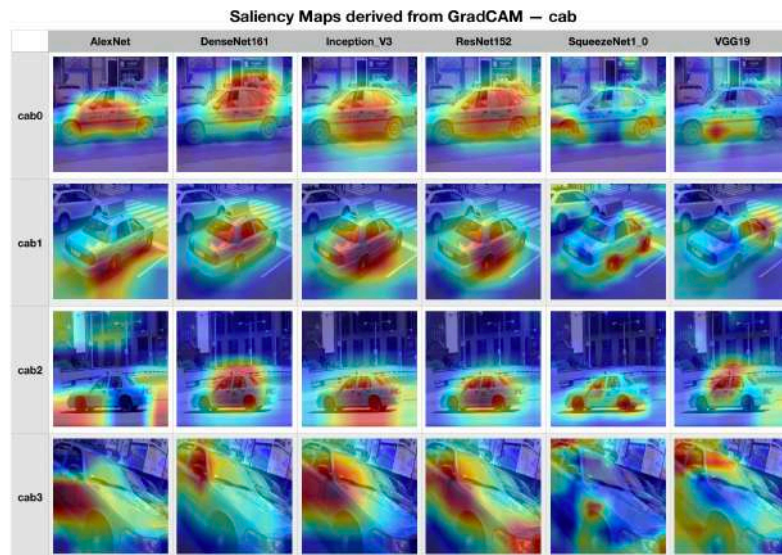


Table 4.8 heat maps derived from GradCAM for cab

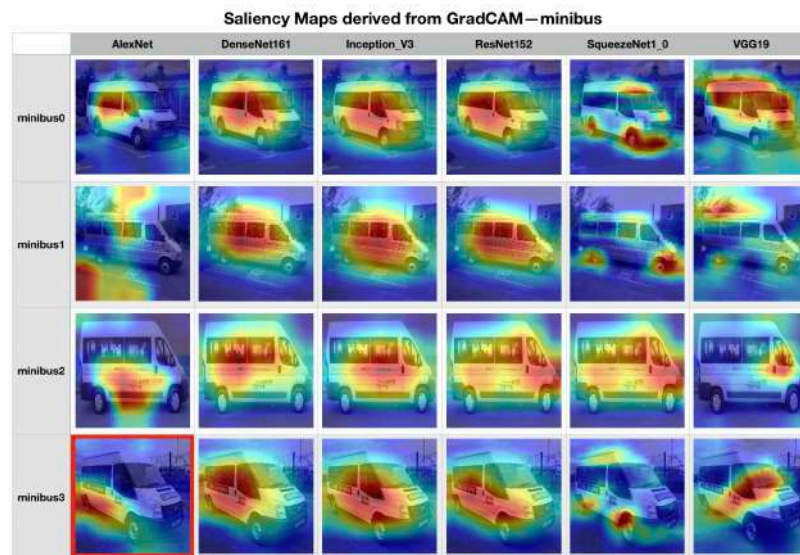


Table 4.9 heat maps derived from GradCAM for minibus

Another interesting lessons we obtained from direct observation is that, SqueezeNet always tend to highlight much smaller area than other architectures. For example, in sample image “Kelpie” and “Kit fox”, SqueezeNet only highlight the eyes, noses, ears and mouths of the dog instead of the whole face; in “minibus” and “cab”, SqueezeNet tend to only highlight the wheels in stead of the whole body of the vehicle.

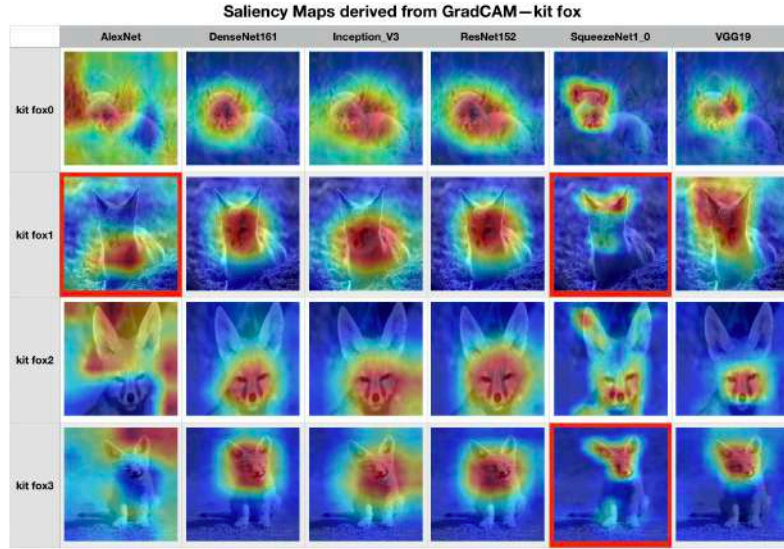


Table 4.10 heat maps derived from GradCAM for kit fox

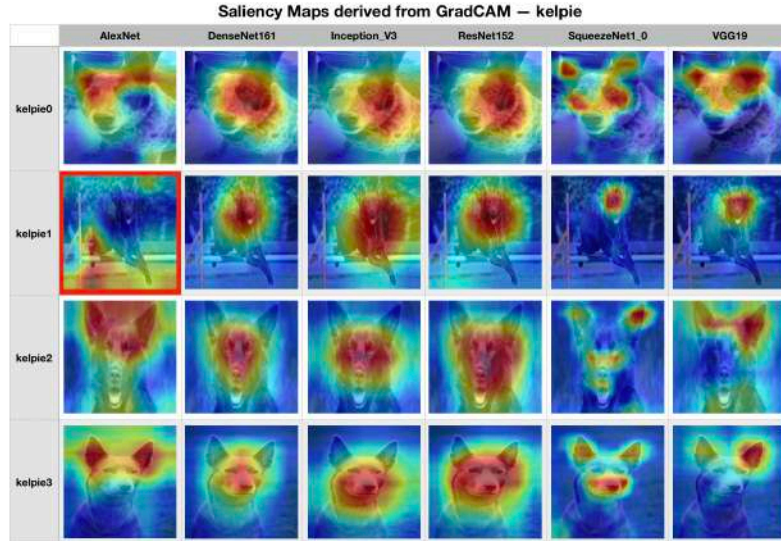


Table 4.11 heat maps derived from GradCAM for kelpie

#### 4.2.4 Guided Grad-CAM

In the first round of our experiment, we only found that heatmaps of ResNet and DenseNet are similar, however, in this round of our experiment, we found a much more frequent cluster formed by ResNet, DenseNet and VGG. These three architectures were the first three architectures to be cluster together in 8/12 our sample images, which suggests it is a stable and significant cluster.

Another finding is that in 9/12 sample images, Inception V3 was the last and the only one to be incorporated into the cluster formed by other 5 architectures, while in the other 3 sample image, Inception together with AlexNet was the last two architectures to be included into cluster. All these clustering results suggest that Inception V3 generate a vastly different heat map from any other architectures.

Saliency Maps derived from Guided GradCAM — kelpie							
	Original	AlexNet	DenseNet161	Inception_V3	ResNet152	SqueezeNet1_0	VGG19
kelpie0							
kelpie1							
kelpie2							
kelpie3							

Table 4.12 heat maps derived from Guided GradCAM for kelpie

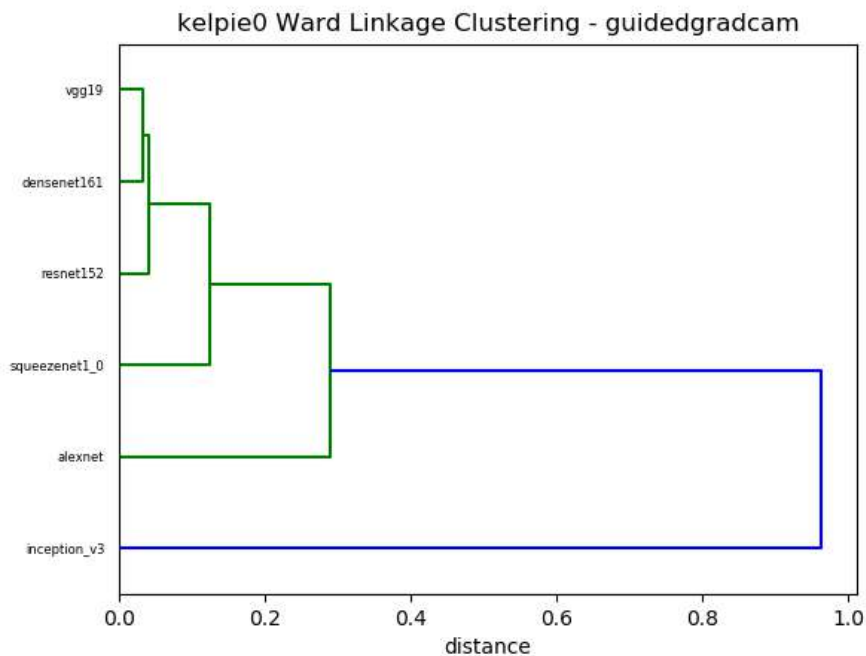


Figure 4.8 Clustering Results of heat maps derived from Guided GradCAM for kelpie 0

AlexNet is also quite different from any other architectures, it is either the second to last to be incorporated into clusters formed by VGG, DenseNet, ResNet and SqueezeNet or together with Inception to be the last two architectures to be included into cluster in all 12/12 sample images.



### 4.2.5 Summary

We draw the following table to illustrate the patterns we found for different explanation methods:

Explanation Methods	Patterns of heatmaps between different CNN architectures	Confidence
<b>Guided BP</b>	Inception V3 are significantly different from others.	11/12
	AlexNet are quite different from others.	6/12
<b>Deconvnet</b>	Each architecture has heat map of its own style.	11/12
<b>Grad-CAM</b>	Cluster: Inception V3, ResNet and DenseNet	9/12
	AlexNet are quite different from others.	8/12
	AlexNet tend to highlight pixels irrelevant to the prediction category.	By observation
	SqueezeNet tend to highlight much smaller area than others.	By observation
<b>Guided Grad-CAM</b>	Cluster: ResNet, DenseNet and VGG	8/12
	Inception V3 are significantly different from others.	9/12
	AlexNet are quite different from others.	12/12

Table 4.13 Patterns found through experiment 2

### 4.2.6 Discussion of Patterns

#### a) Inception V3 has heat maps of its own style regardless of explanation methods

We are curious about why Inception V3 always generate so much different heat maps in three explanation methods based on back propagation—Guided BP, DeconvNet and Guided Grad-CAM. So we build Table 4.13 to facilitate our observation, we are surprising to find that Whatever the explanation methods is used, the heat map of Inception V3 for the same sample image seems to be almost the same. We speculated that this may be caused by the existence of Inception Module.

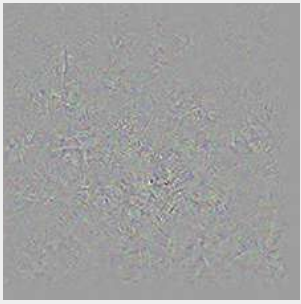
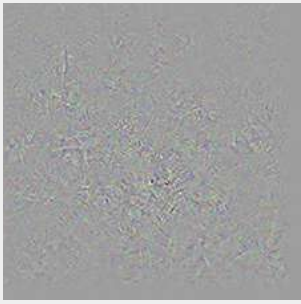
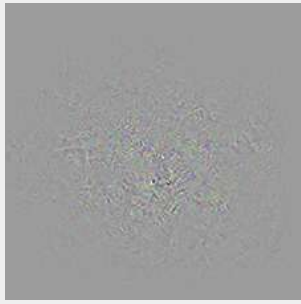
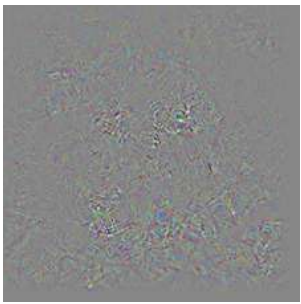
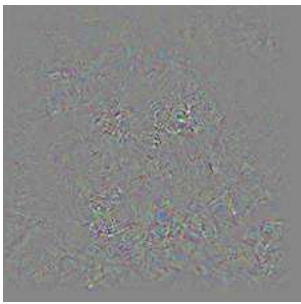
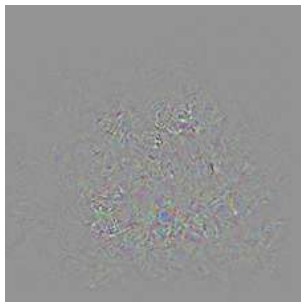
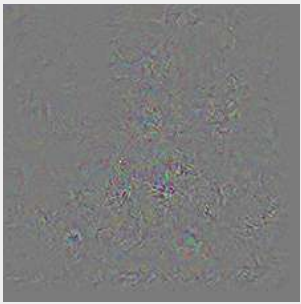
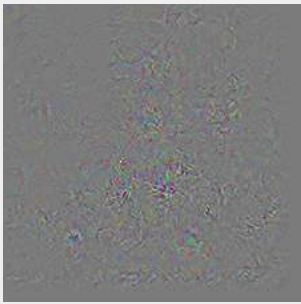
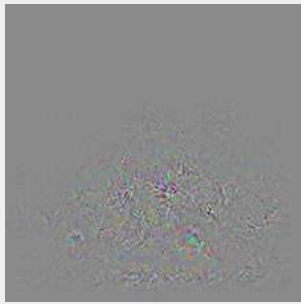
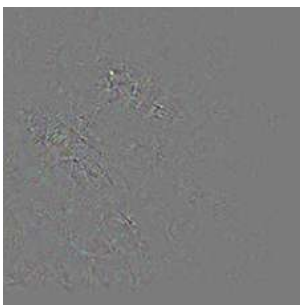
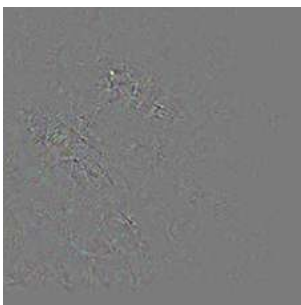
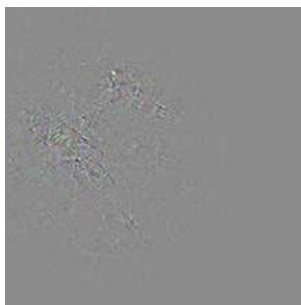
	Guided BP	Deconvnet	Guided Grad-CAM
Cab 0			
Cab 1			
Cab 2			
Cab 3			

Table 4.14 Heat maps of Inception V3 given by three explanation methods

#### b) Propose a subjective standard to assess the explainability of a CNN model

In all four explanation methods we used, AlexNet always give a quite different heat maps from any other architectures. Meantime, in the heat map given by GradCAM, we found AlexNet tend to highlight pixels irrelevant to the prediction category, which may suggest the heat map given by AlexNet has poor explainability. So here comes the question, what is poor explainability for a model or an architecture? In next few paragraphs, we try to prepose a subjective standard to assess the explainability of a CNN model.

We use Figure 4.9 as an example to illustrate our idea. AlexNet has classified this image as minibus correctly. However, from the heat map we can tell that AlexNet mainly makes this decision by the



Figure 4.9 Grad-CAM heat map of AlexNet for mini bus 1

pixel of ground instead of minibus itself, which may suggest AlexNet model has a generally bad explainability.

To furthermore illustrate the subjective standard we proposed to assess the explainability of a CNN model, we add one more experiment using the sample image “broom” mentioned in 3.2.1.

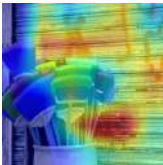
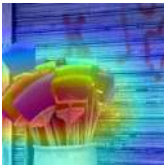
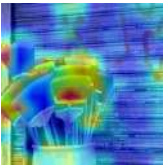
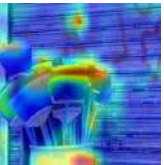

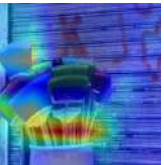
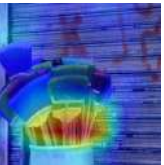



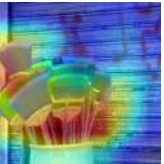

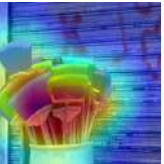
Saliency Maps derived from GradCAM – Broom						
AlexNet (macaw)	Inception_V3	SqueezeNet 1_0 (Swab)	SqueezeNet 1_1 (Shopping basket)	VGG13	VGG16	VGG19
						
DenseNet 121 (paint brush)	DenseNet 169	DenseNet 201	ResNet 50	ResNet 101	ResNet 152	
						

Table 4.15 Heat maps derived from GradCAM for Broom

AlexNet classifies broom as macaw, which is far away from the right answer, thus give a heat map with strange highlight space which is hard to understand for us human.

Now Let’s look at SqueezeNet, we still say it has good explainability because SqueezeNet 1\_1 highlight the basket-like object containing brooms when it classifies this image as shopping basket.

So a model or architecture with good explainability should have the following characteristic: **the classification results should be consistent to the heat map it gives in a way that is understandable to us humans**. In this sense, AlexNet has a generally bad explainability, which may be partly due to its low classification accuracy.

We want to argue that it seems that this way of evaluating the explainability of a CNN model seems to be subjective, but we want to point out that it is to human the CNN model explains, so the subjective participation of human should be included into the assessment.

#### **c) DenseNet and ResNet may have similarity in terms of its explainability**

In our case, having similar explainability means when they make predictions for the same input image, the pixel on which they make decisions based is similar. The similar of explainability of DenseNet and ResNet may partly because they all have Residual Block.

#### **d) SqueezeNet has good explainability**

The “Good Explainability” here means the decisions made by SqueezeNet depend on some pixels which is understandable and precise to us human. And the area they highlight is small, accurate and precise, instead of highlighting a big space for us human to guess from. For example, SqueezeNet predict an image as a car precisely by its two wheels, which make a lot sense to us human beings.

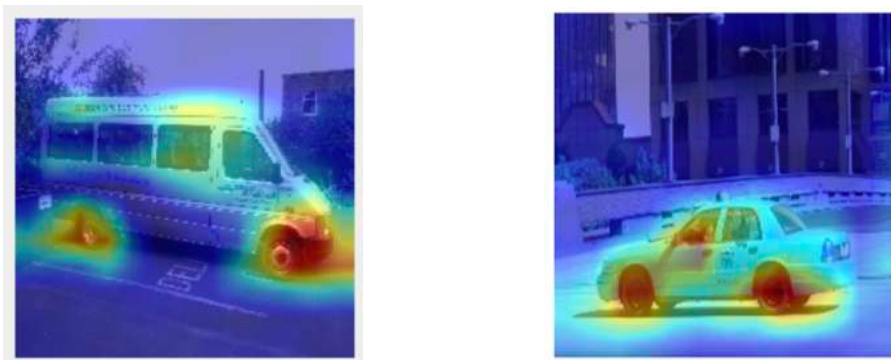


Figure 4.10 SqueezeNet only highlight two wheels of a car, which is regarded with good explainability

### **4.3 Evaluate the Quality of Heat Maps Derived from Different Architectures**

Originally, we came up with a hypothesis that the quality of the heat maps generated by a satisfying explanation methods can to some extent suggest the explainability of a certain architecture. Additionally, we suppose that the “input perturbation analysis”, a method proposed in [15] to quantitatively and objectively evaluate the quality of heat maps given by different explanation methods can be transferred to our case to help us assess the quality of heat maps derived from different architectures.

Based on this two hypothesis, we perform assessment to the explainability among different CNN architectures. The input image sample on which we perform perturbation analysis is 100 validation images from ImageNet dataset. The models in this experiment we use is the readily pre-trained model from Keras.

First we want to figure out what is the best explain method in terms of their explainability, we hold on to one model—Inception V3 and perturbation 15 steps and calculate the AOPC (Area of the most relevant first perturbation curve) for each explanation methods. The bigger AOPC, the better explainability, so it turns out that LRP-Epsilon [36], is the method with the best explainability.

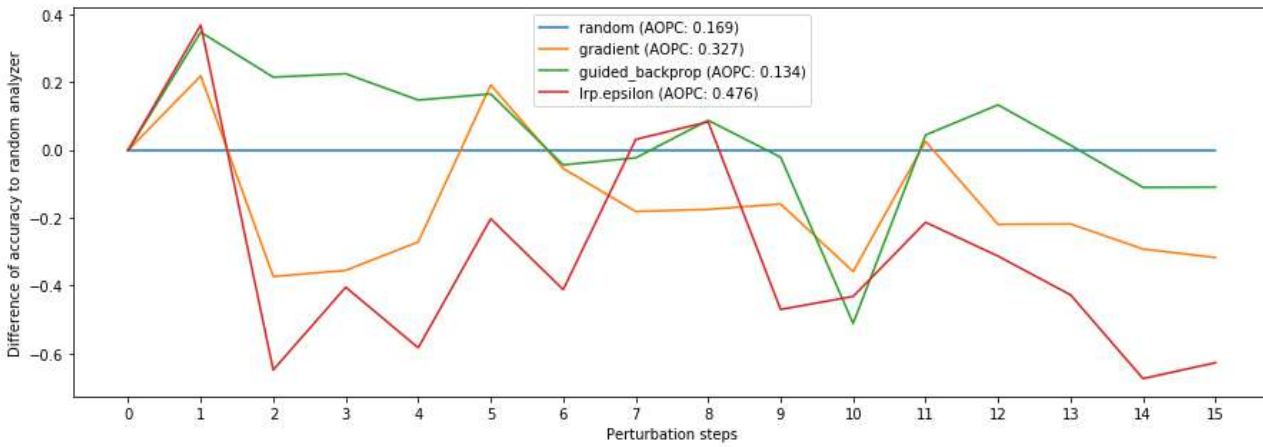


Figure 4.11 Perturbation test across different explanation methods with fixed model Inception V3

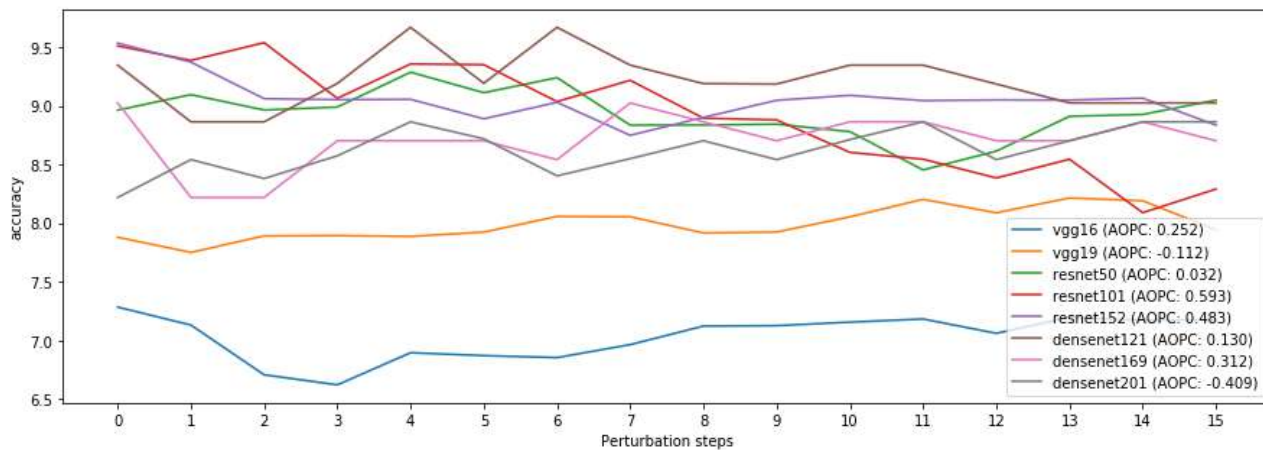


Figure 4.12 Perturbation test across different models with fixed explanation method LRP-Epsilon

Then we fixed LRP-Epsilon method, changed different models and then calculate AOPC for each of them. We found roughly architecture with relatively higher accuracy ,such as ResNet and Inception V3, have bigger AOPC, thus have better explainability. Those with lower accuracy like VGG don't perform well in our perturbation test.

## 4.4 Further Discussion for the Perturbation Test

In this section we want to discuss the effectiveness of our perturbation test in 4.3.

The perturbation test was originally proposed to objectively and quantitatively evaluate the explainability of different explanation methods. An explanation method with good explainability should generate heat map highlighting the really important pixels contributing the most to help the model arrive at certain decision. So the perturbation test basically just evaluate the real importance



of pixels certain heat map highlight, the more important pixels highlighted by a certain heat map are, the better the explanation method is.

Now let's consider examining CNN model or architecture with good explainability, as discussed in 4.2.6 (b), we hold that evaluating the explainability of a CNN model needs human's participation, because it is human to whom model explains. So we cannot assess a model's explainability only by objective approach with human's absence.

After deeply thinking about the difference of two task, we can conclude that this evaluation method proposed by [15] cannot be transferred to the evaluation of the explainability of CNN models and architectures.

**So the perturbation test to evaluate models and architectures is invalid.**

## 5 Conclusions

### 5.1 Summary of Findings

After deriving heat maps across CNN models and architectures by applying different explanation methods, we have the following exciting findings:

1. The explainability of CNN model is indeed influenced by its architecture
2. Inception V3 has heat maps of its own style regardless of explanation methods used.
3. DenseNet and ResNet have much similarity in terms of its explainability.
4. We have proposed a subjective standard to assess the explainability of a CNN model:

a model or architecture with good explainability should have the following characteristic: the classification results should be consistent to the heat map it gives in a way that is understandable to us humans.

5. AlexNet has a relatively bad explainability because it tends to highlight pixels irrelevant to the prediction category.
6. SqueezeNet has a relatively good explainability because tend to highlight much smaller area than others.
7. For DeconvNet, each architecture has heat maps of its own style.
8. The "input perturbation analysis" proposed by [15] can only be used to compare the quality of heat maps obtained by different explanation methods and cannot be transferred to compare heat maps across different CNN architectures.

### 5.2 Open Questions

1. Why Inception V3 has heat maps of its own style regardless of explanation methods? Is it because of its special Inception module?
2. Why SqueezeNet tends to highlight a smaller region than any other architectures in Grad-CAM?
3. Why DenseNet and ResNet always generate similar heat maps? Is it because their Residual Block?

4. Can we devise an objective measure to assess the explainability of CNN models and architectures?
5. Why Deconvnet give models of the same architecture heat Maps with its own style?

### **5.3 Possible Future Work**

We can try out more state-of- art explanation methods such as LRP and more newly proposed CNN architectures to find more interesting patterns.

We also can try different types of images on SqueezeNet to find for specific object, what part do SqueezeNet highlight and whether the explanation is convincing enough for us human.

## Bibliography

- [1] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[C]//Advances in neural information processing systems. 2012: 1097-1105.
- [2] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.
- [3] Long J, Shelhamer E, Darrell T. Fully convolutional networks for semantic segmentation[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2015: 3431-3440.
- [4] You Q, Jin H, Wang Z, et al. Image captioning with semantic attention[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 4651-4659.
- [5] Antol S, Agrawal A, Lu J, et al. Vqa: Visual question answering[C]//Proceedings of the IEEE international conference on computer vision. 2015: 2425-2433.
- [6] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," in Proc. of ICML, 2010, pp. 495–502.
- [7] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng, "Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis," in Proc. of CVPR, 2011, pp. 3361–3368.
- [8] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. P. Kuksa, "Natural language processing (almost) from scratch," Journal of Machine Learning Research, vol. 12, pp. 2493–2537, 2011.
- [9] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in Proc. of EMNLP, 2013, pp. 1631–1642.)
- [10] G. Montavon, M. Rupp, V. Gobre, A. Vazquez-Mayagoitia, K. Hansen, A. Tkatchenko, K.-R. Müller, and O. A. von Lilienfeld, "Machine learning of molecular electronic properties in chemical compound space," New Journal of Physics, vol. 15, no. 9, p. 095003, 2013.
- [11] Adebayo J, Gilmer J, Muelly M, et al. Sanity checks for saliency maps[C]//Advances in Neural Information Processing Systems. 2018: 9505-9515.
- [12] Andrew M Saxe, Pang Wei Koh, Zhenghao Chen, Maneesh Bhand, Bipin Suresh, and Andrew Y Ng. On random weights and unsupervised feature learning. In ICML, pages 1089–1096, 2011.
- [13] Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. arXiv preprint arXiv:1610.01644, 2016.
- [14] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. arXiv preprint arXiv:1711.10925, 2017.
- [15] Samek W, Wiegand T, Müller K R. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models[J]. arXiv preprint arXiv:1708.08296, 2017.
- [16] LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.
- [17] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition[J]. arXiv preprint arXiv:1409.1556, 2014.
- [18] Szegedy C, Liu W, Jia Y, et al. Going deeper with convolutions[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2015: 1-9.
- [19] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.
- [20] Huang G, Liu Z, Van Der Maaten L, et al. Densely connected convolutional networks[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 4700-4708.
- [21] Iandola F N, Han S, Moskewicz M W, et al. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and < 0.5 MB model size[J]. arXiv preprint arXiv:1602.07360, 2016.



- [22] Lipton Z C. The mythos of model interpretability[J]. arXiv preprint arXiv:1606.03490, 2016.
- [23] Vellido A, Martín-Guerrero J D, Lisboa P J G. Making machine learning models interpretable[C]//ESANN. 2012, 12: 163-172.
- [24] Adadi A, Berrada M. Peeking inside the black-box: A survey on Explainable Artificial Intelligence (XAI)[J]. IEEE Access, 2018, 6: 52138-52160.
- [25] Quanshi Zhang, Ying Nian Wu, and Song-Chun Zhu. Interpretable convolutional neural network. In arXiv:1710.00935, 2017.
- [26] Sara Sabour, Nicholas Frosst, and Geoffrey E. Hinton. Dynamic routing between capsules. In NIPS, 2017.
- [27] Tianfu Wu, Xilai Li, Xi Song, Wei Sun, Liang Dong, and Bo Li. Interpretable r-cnn. In arXiv: 1711.05226, 2017.
- [28] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In NIPS, 2016.
- [29] Q. Zhang, R. Cao, Y. N. Wu, and S.-C. Zhu. Growing interpretable part graphs on convnets via multi-shot learning. In AAAI, 2016.
- [30] Q. Zhang, R. Cao, F. Shi, Y.N. Wu, and S.-C. Zhu. Interpreting cnn knowledge via an explanatory graph. In AAAI, 2018.
- [31] Quanshi Zhang, Yu Yang, Ying Nian Wu, and Song-Chun Zhu. Interpreting cnns via decision trees. arXiv:1802.00121, 2018.
- [32] Simonyan K, Vedaldi A, Zisserman A. Deep inside convolutional networks: Visualising image classification models and saliency maps[J]. arXiv preprint arXiv:1312.6034, 2013.
- [33] Zeiler M D, Fergus R. Visualizing and understanding convolutional networks[C]//European conference on computer vision. springer, Cham, 2014: 818-833.
- [34] Springenberg J T, Dosovitskiy A, Brox T, et al. Striving for simplicity: The all convolutional net[J]. arXiv preprint arXiv:1412.6806, 2014.
- [35] Selvaraju R R, Cogswell M, Das A, et al. Grad-cam: Visual explanations from deep networks via gradient-based localization[C]//Proceedings of the IEEE International Conference on Computer Vision. 2017: 618-626.
- [36] Bach S, Binder A, Montavon G, et al. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation[J]. PloS one, 2015, 10(7): e0130140.
- [37] Wang Z, Bovik A C, Sheikh H R, et al. Image quality assessment: from error visibility to structural similarity[J]. IEEE transactions on image processing, 2004, 13(4): 600-612.
- [38] Samek W, Binder A, Montavon G, et al. Evaluating the visualization of what a deep neural network has learned[J]. IEEE transactions on neural networks and learning systems, 2017, 28(11): 2660-2673.
- [39] Lim Y K. An evaluation of the explainability of models from transfer learning[D]. UTAR, 2018.