CLASSIFICATION OF GROUND DEFORMATION IN InSAR DATA USING CNNs.

Under the supervision of
Prof. PABITRA MITRA, - April 2019
Dept. of Computer Science and Engineering
IIT-Kharagpur

and

Mr.Kaushik Biswas(Ph.D),
Under
Prof.Pabitra Mitra, Dept. CSE
Prof.Debashish Chakravarty
Dept. Mining Engineering

Report by:
P. JAYANTH AVINASH,15EE10029
4th UG student,Dept. of Electrical Engineering

INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR
भारतीय प्रौद्योगिकी संस्थान खड़गपुर
19 51
योगः कर्मसु कौशलम

1 DECLARATION and ACKNOWLEDGEMENT

I certify that
(a) The work contained in this report has been done by me under the guidance of my supervisor.
(b) The work has not been submitted to any other Institute for any degree or diploma.
(c) I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.
(d) Whenever I have used materials (data, theoretical analysis, figures, and text) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the references. Further, I have taken permission from the copyright owners of the sources, whenever necessary.

I would like to express gratitude and indebtedness to my supervisor Prof. Pabitra Mitra, Computer Science and Engineering Department, IIT Kharagpur,Mr. Koushik Biswas and Prof. Debashish Chakravarthy for their guidance and encouragement in the present investigation throughout the duration of this project. It is beyond doubt that this work would not have come into existence in the absence of their guidance.

I express my sincere thanks to all my professors for their encouragement at different stages of my work.

P.JAYANTH AVINASH,15EE10029

CERTIFICATION

This is to certify that the project report entitled “CLASSIFICATION OF GROUND DEFORMATION IN InSAR DATAUSING CNNs.” submitted by P. JAYANTH AVINASH (Roll No. 15EE10029) to Indian Institute of Technology Kharagpur towards partial fulfilment of requirements for the award of degree of Bachelor of Technology in Electrical Engineering is a record of bonafide work carried out by him under my supervision and guidance during spring Semester, 2018-19.

Prof. Pabitra Mitra,
Dept. of Computer Science and Engineering
IIT Kharagpur

Contents

2 INTRODUCTION

Recent improvements in the frequency, type, and availability of satellite images made it now feasible to routinely study volcanoes in remote and inaccessible regions, including those with no ground-based monitoring. In particular, Interferometric Synthetic Aperture Radar data can detect surface deformation, which has a strong statistical link to eruption[10]. However, the data set produced by the recently launched Sentinel-1 satellite is too large to be manually analyzed on a global basis. In this study, we systematically process $\geq$ 30,000 short-term interferograms at over 900 volcanoes and apply machine learning algorithms to automatically detect volcanic ground deformation. Here We use use Different well known CNN Architectures and use them to classify the interferograms. We use transfer learning strategy and study the performance of Networks for some specific learning rates and optimizers.

Globally, 800 million people live within 100 km of a volcano . Improvements in monitoring and forecasting have been shown to reduce fatalities due to volcanic eruptions, but a significant proportion of the $\sim$ 1,500 holocene volcanoes has no ground-based monitoring[10]. Interferometric Synthetic Aperture Radar (InSAR) is a satellite remote sensing technique used to measure ground displacement at the centimeter scale over large geographic areas and has been widely applied to volcanology (e.g., Biggs Pritchard, 2017; Pinel et al., 2014). Furthermore, InSAR measurements of volcanic deformation have a significant statistical link to eruption. Modern satellites provide large coverage with high resolution signals, generating large data sets. For example, the two-satellite constellation, Sentinel-1 A and B, offers a 6-day repeat cycle and acquires data with a 250-km swath at a 5 m by 20 m spatial resolution (single look)[10]. This amounts to$\geq$10-TB per day or about 2 PB collected between its launch in 2014 and June 2017. The explosion in data has brought major challenges associated with manual inspection of imagery and timely dissemination of information.Many volcano observatories lack the expertise needed to exploit satellite data sets, particularly those in developing countries.

Similarly, Deformation occurring due to mining activity will have considerable effect on fatalities . In this project we develop study CNN models and study their effectiveness in Classifying small scale Deformation(due to mining activity). The Major Challenge in our case is that Deformation fringes in Interferograms are much less pronounced than in case of Volcanic deformation. Also the shape and size of Deformation fringes in Small Scale Deformation (SSD) vary largely across the different interferograms since they are consequences of human activity unlike Volcanoes. And most importantly the size of Data available for SSD (for training CNNs) is comparatively much lesser than the case of Vocanic data,so currently the data preparation for our case is in process and Networks are Trained on Volcanic data in order to study their performances beforehand(since , SSD is similar to Large Scale Deformation ;LSD to some Extent).

3 INTERFEROMETRY

Interferometry is a Class of scientific techniques used to Extract information by the phenomenon of interference caused when Electromagnetic waves superimpose on each other. These techniques are use two or more Synthetic Aperture radar (SAR) images to generate maps of surface deformation or digital elevation using differences in phase of the waves returning to the satellite[11].Interferograms help us to determine the location, magnitude and type of an earthquake. They can also help us to improve earthquake models, and investigate the future seismic hazard for an area The most important factor affecting the phase is the interaction with Ground surface . The phase of the wave may change on reflection depending the properties of the material.

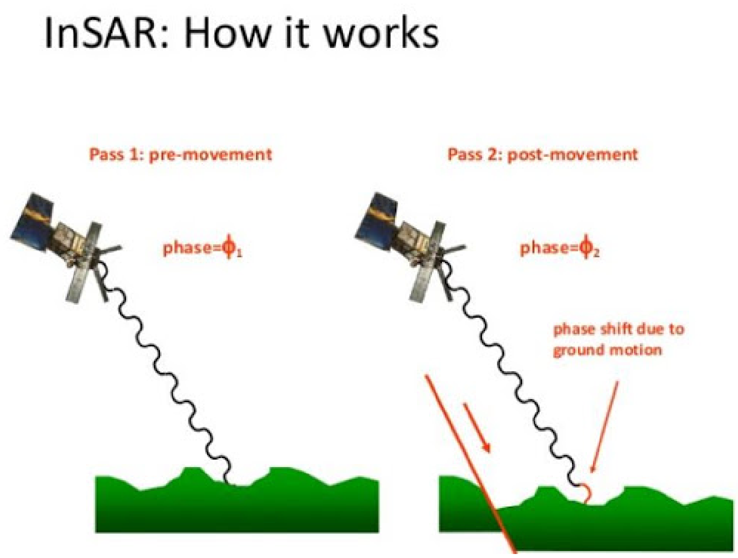


Figure 1: Picture showing Master-slave configuration where Two pictures of same area taken with time certain cycle by two satellites @ source: science direct

There are basically two images obtained which differ by some phase difference(due to deformation, atmospheric noise etc) and are mapped from [ϕ_1-ϕ_2]to 0-256(RGB) for displaying. In case Wrapped phase interferograms [ϕ_1-ϕ_2] is wrapped to [0-2π],which is again mapped to 0-256(RGB).

4 DATA and EXPERIMENTATION

In this project we worked with wrapped phase interferograms. The values of wrapped interferograms vary between $-\pi$ and π and they are typically displayed with colors (red, green, and blue intensities).

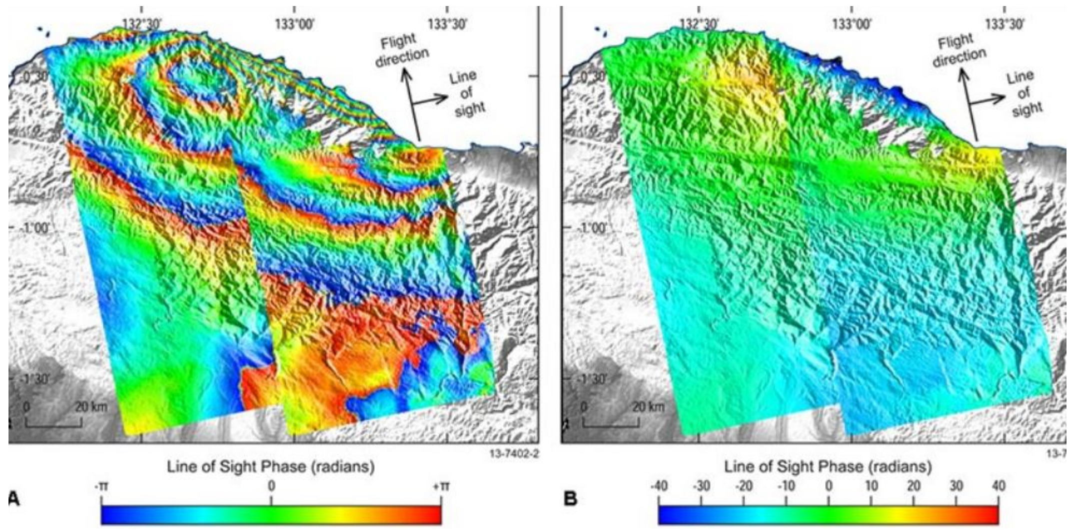


Figure 2: (A)wrapped and (B)unwrapped-phase Interferograms, source:Geoscience Australia

For sake of using Convolutional Neural Nets(CNNs),we first convert the wrapped interferogram into grayscale image, that is, the pixel value in the range of $[-\pi\ ,\pi]$ is scaled to [0, 255] or [125, 125] if zero-center normalization is required. Subsequently, each training image is divided into patches equal to the input size of the CNNs (mostly $224\times$ 224). In case of Volcanic deformation Data(LSD),Canny edge detection is applied, where a Gaussian filter is first applied to remove noise, and then double thresholding is applied to the intensity gradients of the image. As the wrapped-phase interferograms shows strong edges where the phase jumps between $-\pi$ and π the Canny operator can straightforwardly extract fringes occurring from volcano deformation[10] .As the number of background areas (negative samples) is significantly larger than those associated with volcano deformation (positive samples), only the patches in which strong edges have been detected are used. Since areas without strong edges are unlikely to contain volcanic deformation, they are instantly defined as background without classification by the CNN.The training process starts with data with ground truth (labeled as 1 or positive, where deformation is present; and 0 or negative in other areas eg:background). Data augmentation has been done to balance number of Positive and negative samples for training CNNs.

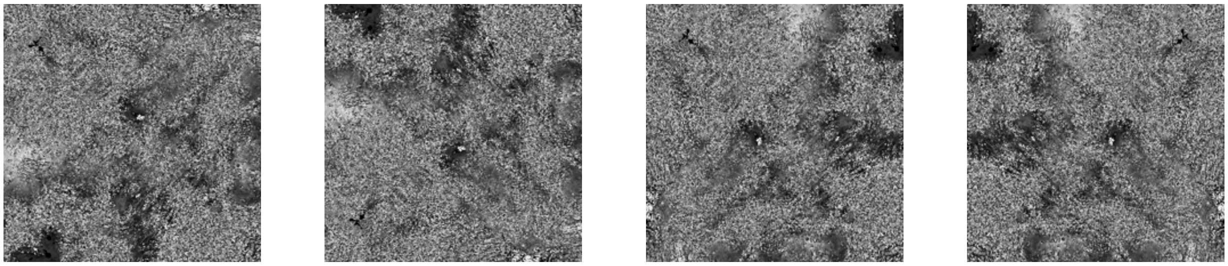


Figure 3: images from background class @source:Dataset

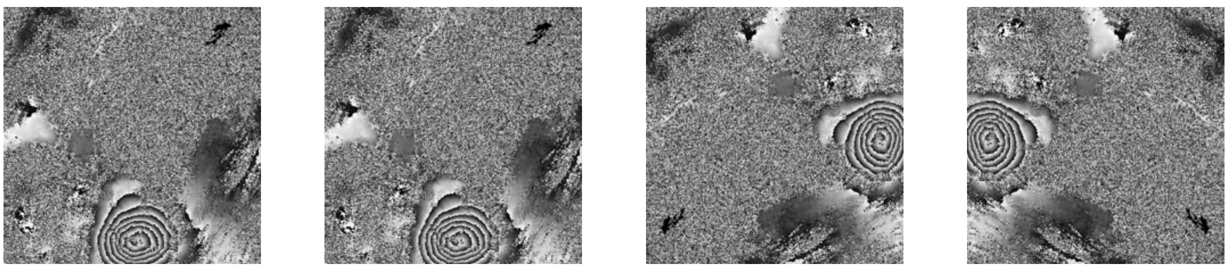


Figure 4: images from volcano deformation class @source:Dataset

4.1 Our data and Preparation

In our case data we are dealing with is from mining deformation(SSD). Unlike volcano data, our data has less Signal to noise ratio. Since the scale of Deformation is small, fringes are less significant and weak.Since The amount of data available is quite less compared to volcanic deformation,it is being augmented. Machine learning models are better when trained on sufficient amount of training data, and CNN models are designed aiming to work well in case small training data and better accuracy with less number of training parameters.

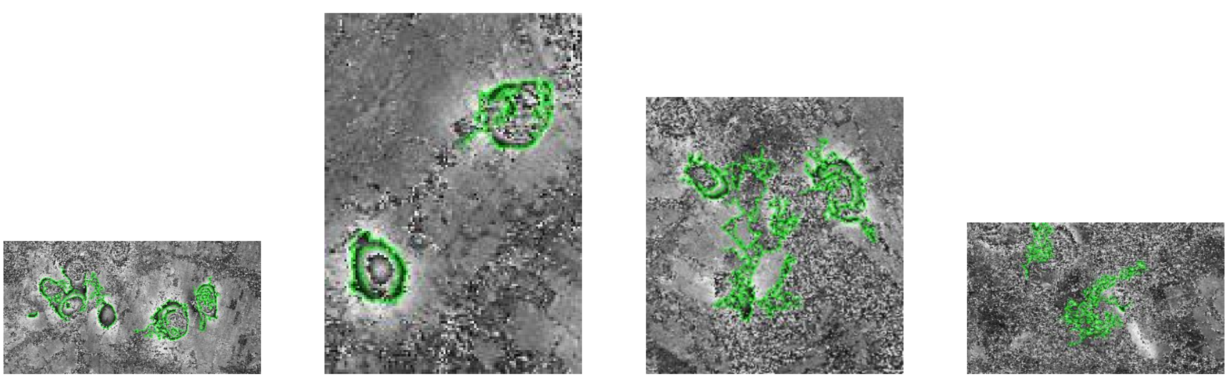


Figure 5: images from mining Deformation interferograms using canny edge detection , The left most image patch is Noise that is detected as a fringe,Also sizes are varying from images to image

4.2 Challenges involved

In Mining data set, we have applied Gaussian filter to remove noise and canny edges detection algorithms to detect edges . Two challenges involved in this process are
1.The amount of training data available is low
2. Fringes in low SNR region remain undetected for some threshold threshold levels which varies from image to image
3. Some Noisy regions are detected as fringe.(eg: left most image)
4.Size and Dimension of croppings vary significantly, on the other hand CNNs can take images of any size but the fully connected layers should have fixed size input,(to over come this issue SPPnets are used)
5. Unlike Volcanic Deformation, Fringe patterns and shapes vary widely across the images and it is comparatively difficult For CNNs to learn those patters.
6. These patterns sometimes are connected to each other and are Merged together unlike Volcanic Deformation fringes.

5 LITERATURE REVIEW

To learn about thousands of objects from millions of images, we need a model with a large learning capacity. However, the immense complexity of the object recognition task implies that this problem cannot be specified even by a dataset as large as ImageNet, so a model should also have lots of prior knowledge to compensate for all the data we don't have[9]

Convolutional Neural Networks (CNNs) constitute one such class of models. Their learning capability and capacity can be controlled by varying their depth and breadth, and they also make strong and more accurate assumptions about the nature of images (namely, stationarity of statistics and locality of pixel dependencies)[9].Thus, compared to standard feed forward Neural Networks(NN) with similarly-sized layers, CNNs have much fewer connections and parameters and so they are easier to train, while their theoretically-best performance is likely to be only slightly worse[7,9].Pooling layers in CNNs summarize the outputs of neighboring groups of neurons in the same kernel map.To some extent this helps the models to overcome object localization problem[7,9].

5.1 Hyper-parameters

A common problem faced is choosing a learning rate and optimizer (the hyper-parameters). hyper-parameters are crucial to training success, yet can be hard to find.It should be noted that the results below are for one specific model and dataset. The ideal hyper-parameters for other models and datasets might differ.Gradient descent is one of the most popular algorithms to perform optimization and by far the most common way to optimize neural networks. The Optimizers vailable in Keras API include Stochastic gradient descent(SGD),Root Mean Square Propagation(RMSprop),Adaptive Gradient descent(Adagrad),Adadelta,Adam,Adamax and Nadam optimizers. It has to be noted that the choice of optimizers and learning rate for that particular optimizer has a great effect on training time and average accuracy on validation data. In our case we tested the Models performances with Adadelta,and SGD Optimizers with learning initial Learning rates(LR) of 0.1,0.01,0.001,0.0002,0.0005,0.0001. while default LR of Adadelta is 1.0,we tested it for the case of 0.01,0.1. It was observed that SGD is taking comparatively less time for updating each gradient in an epoch. So, SGD Optimizer was chosen later for training all models. SGD performs frequent updates with a high variance that cause the objective function to fluctuate heavily. So the Testing loss is often seen fluctuating, while the fluctuations in Validation accuracy can be due to over fitting sometimes. SGD's fluctuation, on the one hand, enables it to jump to new and potentially better local minima[2]. On the other hand, this ultimately complicates convergence to the exact minimum, as SGD will keep overshooting.

lr=0.001, decay=1e-6, momentum=0.9, nesterov=True were given as input parameters for SGD in all cases and each model is trained for a 20 epochs, (Alexnet for 50 epochs). A kernel Regulizer with L_2),(0.01 as input parameter) is chosen on a Denselayer with 200 Neurons on in Fully Connected Layer(FCL)on top of CNN block for each model. This helps in reducing Fluctuations in training Loss,accuracy and chance of over fitting.

5.2 Activations

The standard way to model a neuron's output f as a function of its input x is with $f(x) = tanh(x)$ or $f(x) = (1 + e^x)^{-1}$(sigmoid). In terms of training time with gradient descent, these saturating nonlinearities are much slower than the non-saturating nonlinearity $f(x) = max(0, x)$(ReLU)[9]. Deep convolutional neural networks with ReLUs train several times faster than their equivalents with $tanh$ or $Sigmoid$ units.ReLUs have the desirable property that they do not require input normalization to prevent them from saturating. If at least some training examples produce a positive input to a ReLU, learning will happen in that neuron. In our training we often used combination of these activations together to test the model on Volcanic deformation dataset.At fully connected layers on the top we used $softmax$ activation which pushes all the outputs in the range of [0,1] as class probabilities.

5.3 DropOut

Combining the predictions of many different models is a very successful way to improve test accuracy, but it can computationally too expensive for big neural networks (with More parameters to train and Deeper)to train. There is, however, a very efficient way of model combination that only costs about a factor of two during training, called "dropout", consists of setting the output of each hidden neuron to zero with a probability p given as parameter to Dropout layer. The neurons which are "dropped out" in this way do not contribute to the forward pass and do not participate in back propagation as well. So every time an input is given to Network, the neural network effectively samples a different architecture[9], but all these architectures share weights. This technique reduces complex co-adaptations of neurons,since a neuron cannot rely on the presence of particular other neurons. It is, therefore, forced to learn more robust features that are useful in conjunction with many different random subsets of the other neurons[9]. At test time, we use all the neurons but multiply their outputs by p, which is a reasonable approximation to taking the geometric mean of the predictive distributions produced by the exponentially-many dropout networks. In our work, we chose Droput as 0.5 in 2nd last layer of FCLs and 0.2,0.3 in initial experimentation with some Models.

5.4 Transfer learning

Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task.For image classification of problems, it is common to use a deep learning model pre-trained for a large and challenging image classification task such as the ImageNet 1000-class photograph classification competition.Many deep neural networks trained on natural images are known to exhibit a curious phenomenon in common: on the first layer they learn features similar to Gabor filters and color blobs[3]. Such first-layer features appear not to be specific to a particular dataset or task, but general in that they are applicable to many datasets and tasks.In our problem we used and tested transfer learning strategy with different models whose weights are loaded from models trained on Imagenet data set. Transfer learning worked well with VGG16, InceptionV3 but failed in case of ResNet and DenseNets for a Learning rate of 0.001 and 20epochs.Performance of these models are shown in Results section.

5.5 SPPNet

The most important challenge with our Dataset is overcoming the issue of varying input size. Convolutional Blocks can take inputs of any dimension but the Top layers of these models i.e The FCLs can only take iput vector of fixed size. To deal with this issue, we used Spatial Pyramidal pooling (SPP) layer on top of CNN block.These work by fixing the number of bins per image with Varying bin sizes. Fixing Number bins per a feature map ensures the input shape to Dense layers of the network to be invariant[4]. SPPlayer takes input a list(eg:[2,4,6])as input argument and outputs a one dimensional

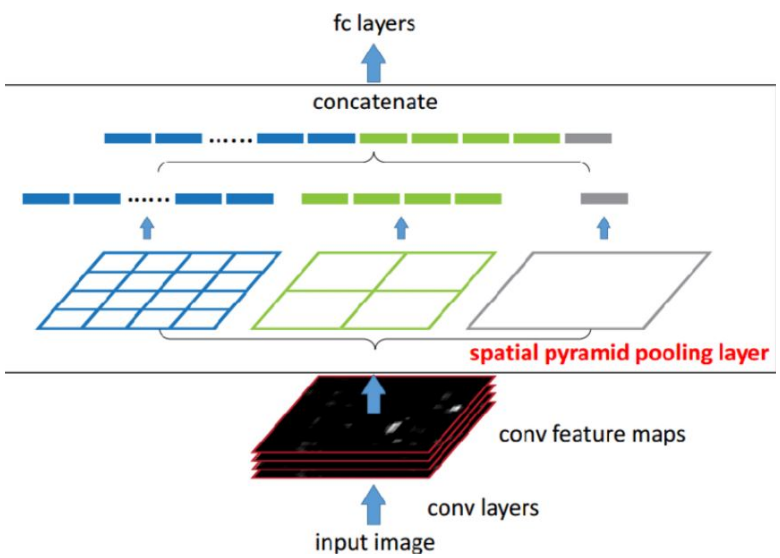


Figure 6: Figure showing SPP Layer working @source: [4]

vector of size sum of the squares of input list(i.e 56=$2^2 + 4^2 + 6^2$ in our example)

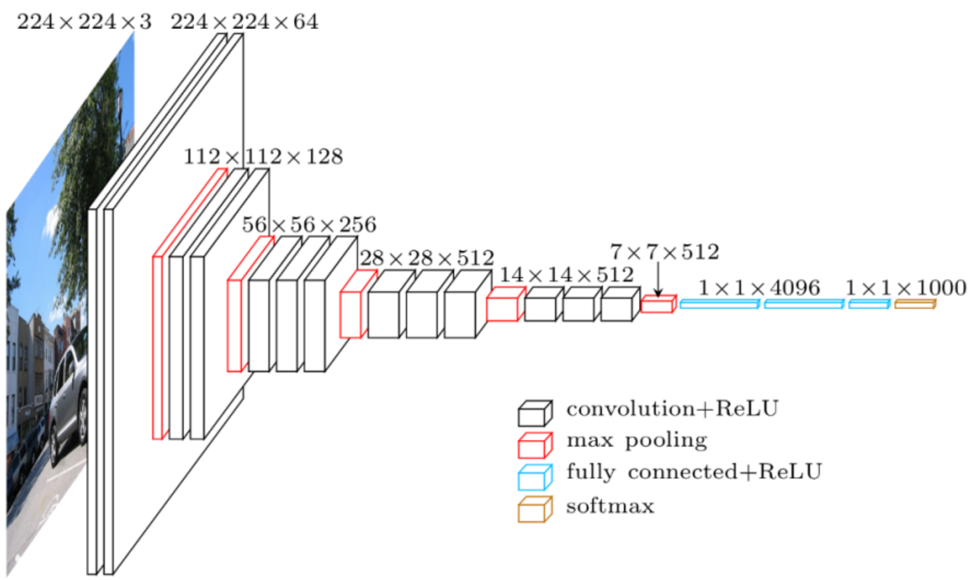


Figure 7: Figure showing VGG16 architecture @source :[1]

5.6 CNN Models used and their Architectures

VGG16 is a convolutional neural network model proposed by K. Simonyan and A. Zisserman from the University of Oxford in the paper “Very Deep Convolutional Networks for Large-Scale Image Recognition”. The model achieves 92.7(%) top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes.
Unfortunately, there are two major drawbacks with VGGNet:
1.It is painfully slow to train.
2.The network architecture weights themselves are quite large (concerning disk/bandwidth).

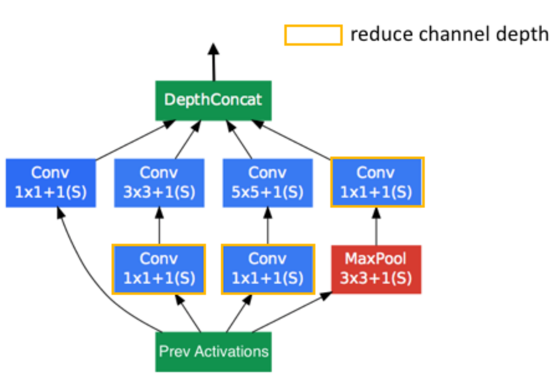


Figure 8: Inception Cell @source : [1]

In 2014, researchers at Google introduced the Inception network.The model is comprised of a basic unit referred to as an ”Inception cell” in which we perform a series of convolutions at different scales and subsequently aggregate the results.. For each cell, we learn a set of 1x1, 3x3, and 5x5 filters which can learn to extract features at different scales from the input. Max pooling is also used, albeit with ”same” padding to preserve the dimensions so that the output can be properly concatenated.

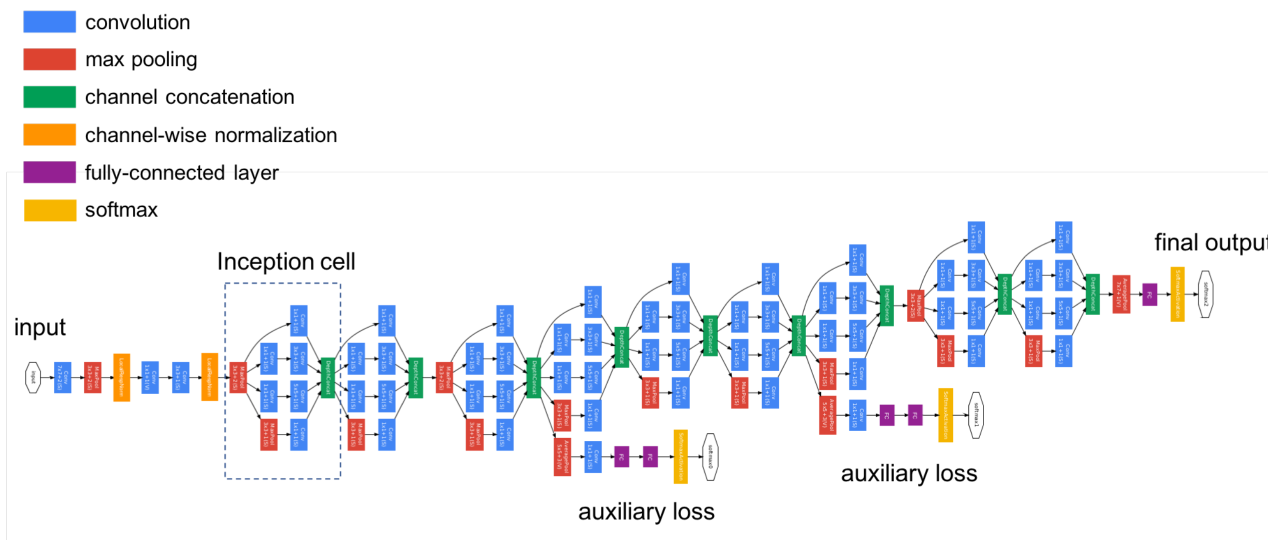
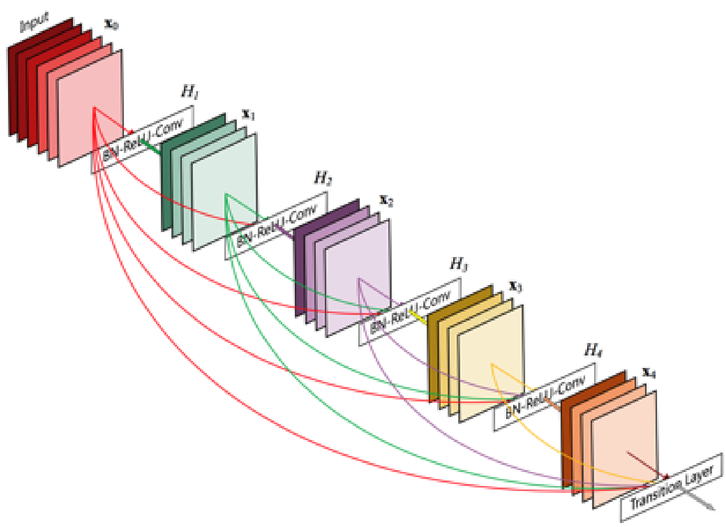


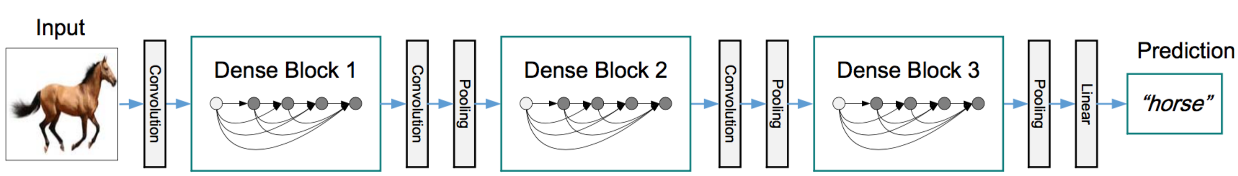
Figure 9: Figure showing CNN block of InceptionV3 @source: [1]

Figure 10: Figure showing CNN block of ResNet,ResNets are easy to optimize, but the "plain" networks (that simply stack layers) shows higher training error when the depth increases.ResNets can easily gain accuracy from greatly increased depth, producing results which are better than previous networks It is worth noticing that the ResNet model has fewer filters and lower complexity than VGG nets



ht

Figure 11: Figure showing Dense block of DenseNet121,To further improve the information flow between layers a different connectivity pattern has been proposed . Dense blocks are Designed to have direct connections from any layer to all subsequent layers.Traditional convolutional feed-forward networks connect the output of the L^{th} layer as input to the $(L + 1)^{th}$ layer, which gives rise to the following layer transition: $x_l = H_l(x_{l-1})$. ResNet add a skip-connection that bypasses the non-linear transformations with an identity function: $x_l = H_l(x_{l1}) + x_{l-1}$ [8]. In case of DenseNets it is$x_l = H_l([x_0, x_1, ..., x_{l-1}])$,where$[x_0, x_1, ., x_{l-1}]$ is concatenation of all previous layers [5].



ht

Figure 12: Figure showing CNN block of DenseNet121 @source : [1]

5.6.1 AlexNet

In AlexNet(A plain neural net),Relu activation function is used instead of $Tanh$ to add non-linearity. It accelerates the speed by 6 times at the same accuracy. Uses dropout instead of regularisation to deal with overfitting. However, the training time is doubled with the dropout rate of 0.5.overlap pooling to reduce the size of the network. It reduces the top-1 and top-5 error rates by 0.4% and 0.3%, respectivel(on a dataset with enough number of classes). It has 5 Convolutional layers and 3 FCL and a total of 62,378,344 parameters to train [4].

5.6.2 VGG16

Layer (type)	Output Shape	Param #
VGG16 (Model)	(None, 200)	15445000
dense_8 (Dense)	(None, 200)	40200
batch_normalization_4 (Batch	(None, 200)	800
activation_4 (Activation)	(None, 200)	0
dense_9 (Dense)	(None, 200)	40200
batch_normalization_5 (Batch	(None, 200)	800
activation_5 (Activation)	(None, 200)	0
dropout_2 (Dropout)	(None, 200)	0
dense_10 (Dense)	(None, 2)	402
batch_normalization_6 (Batch	(None, 2)	8
activation_6 (Activation)	(None, 2)	0

Total params: 15,527,410
Trainable params: 15,526,606
Non-trainable params: 804

Figure 13: Table showing VGG16's summary;On top of CNN block custom dense layers(FCls) were added. @source:output log

Layer (type)	Output Shape	Param #
model_2 (Model)	(None, 200)	15445000
dense_8 (Dense)	(None, 200)	40200
batch_normalization_4 (Batch	(None, 200)	800
activation_4 (Activation)	(None, 200)	0
dense_9 (Dense)	(None, 200)	40200
batch_normalization_5 (Batch	(None, 200)	800
activation_5 (Activation)	(None, 200)	0
dropout_2 (Dropout)	(None, 200)	0
dense_10 (Dense)	(None, 2)	402
batch_normalization_6 (Batch	(None, 2)	8
activation_6 (Activation)	(None, 2)	0

Total params: 15,527,410
Trainable params: 12,610,958
Non-trainable params: 2,916,452

Figure 14: Table showing Pretrained VGG16's summary;On top of CNN block custom dense layes(FCls) were added,some CNN layers were Freezed from learning. @source:output log

5.6.3 InceptionV3

Layer (type)	Output Shape	Param #
InceptionV3 (Model)	(None, 200)	24105960
dense_98 (Dense)	(None, 200)	40200
batch_normalization_246 (Bat	(None, 200)	800
activation_1128 (Activation)	(None, 200)	0
dense_99 (Dense)	(None, 200)	40200
batch_normalization_247 (Bat	(None, 200)	800
activation_1129 (Activation)	(None, 200)	0
dropout_20 (Dropout)	(None, 200)	0
dense_100 (Dense)	(None, 2)	402
batch_normalization_248 (Bat	(None, 2)	8
activation_1130 (Activation)	(None, 2)	0

Total params: 24,188,370
Trainable params: 24,153,134
Non-trainable params: 35,236

Figure 15: Table showing InceptionV3 summary.On top of CNN block custom dense layes(FCls) were added. @source:output log

Layer (type)	Output Shape	Param #
InceptionV3 (Model)	(None, 200)	24105960
dense_5 (Dense)	(None, 200)	40200
batch_normalization_189 (Bat	(None, 200)	800
activation_189 (Activation)	(None, 200)	0
dense_6 (Dense)	(None, 200)	40200
batch_normalization_190 (Bat	(None, 200)	800
activation_190 (Activation)	(None, 200)	0
dropout_1 (Dropout)	(None, 200)	0
dense_7 (Dense)	(None, 2)	402
batch_normalization_191 (Bat	(None, 2)	8
activation_191 (Activation)	(None, 2)	0

Total params: 24,188,370
Trainable params: 13,499,662
Non-trainable params: 10,688,708

Figure 16: Table showing Pretrained InceptionV3 summary.On top of CNN block custom dense layes(FCls) were added.some CNN layers were Freezed from learning. @source:output log

5.6.4 ResNet50

Layer (type)	Output Shape	Param #
Resnet50 (Model)	(None, 200)	25890888
dense_18 (Dense)	(None, 200)	40200
batch_normalization_195 (Bat	(None, 200)	800
activation_729 (Activation)	(None, 200)	0
dense_19 (Dense)	(None, 200)	40200
batch_normalization_196 (Bat	(None, 200)	800
activation_730 (Activation)	(None, 200)	0
dropout_3 (Dropout)	(None, 200)	0
dense_20 (Dense)	(None, 2)	402
batch_normalization_197 (Bat	(None, 2)	8
activation_731 (Activation)	(None, 2)	0

```
Total params: 25,973,298
Trainable params: 25,919,374
Non-trainable params: 53,924
```

Figure 17: Table showing ResNet50 summary,On top of CNN block custom dense layers(FCls) were added. @source:output log

Layer (type)	Output Shape	Param #
resnet50 (Model)	(None, 7, 7, 2048)	23587712
flatten_1 (Flatten)	(None, 100352)	0
dense_1 (Dense)	(None, 200)	20070600
batch_normalization_1 (Batch	(None, 200)	800
activation_50 (Activation)	(None, 200)	0
dense_2 (Dense)	(None, 200)	40200
batch_normalization_2 (Batch	(None, 200)	800
activation_51 (Activation)	(None, 200)	0
dropout_1 (Dropout)	(None, 200)	0
dense_3 (Dense)	(None, 2)	402
batch_normalization_3 (Batch	(None, 2)	8
activation_52 (Activation)	(None, 2)	0

Total params: 43,700,522
Trainable params: 23,527,558
Non-trainable params: 20,172,964

Figure 18: Table showing Pretrained ResNet50 summary,On top of CNN block custom dense layes(FCls) were added.some CNN layers were Freezed from learning @source:output log

5.6.5 DenseNet121

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
DenseNet121 (Model)          (None, 200)               8292104
_________________________________________________________________
dense_3 (Dense)              (None, 200)               40200
_________________________________________________________________
batch_normalization_1 (Batch (None, 200)               800
_________________________________________________________________
activation_1 (Activation)    (None, 200)               0
_________________________________________________________________
dense_4 (Dense)              (None, 200)               40200
_________________________________________________________________
batch_normalization_2 (Batch (None, 200)               800
_________________________________________________________________
activation_2 (Activation)    (None, 200)               0
_________________________________________________________________
dropout_1 (Dropout)          (None, 200)               0
_________________________________________________________________
dense_5 (Dense)              (None, 2)                 402
_________________________________________________________________
batch_normalization_3 (Batch (None, 2)                 8
_________________________________________________________________
activation_3 (Activation)    (None, 2)                 0
=================================================================
Total params: 8,374,514
Trainable params: 8,290,062
Non-trainable params: 84,452
_________________________________________________________________
```

Figure 19: Table showing DenseNet121 summary,On top of CNN block custom dense layers(FCls) were added.source:output log

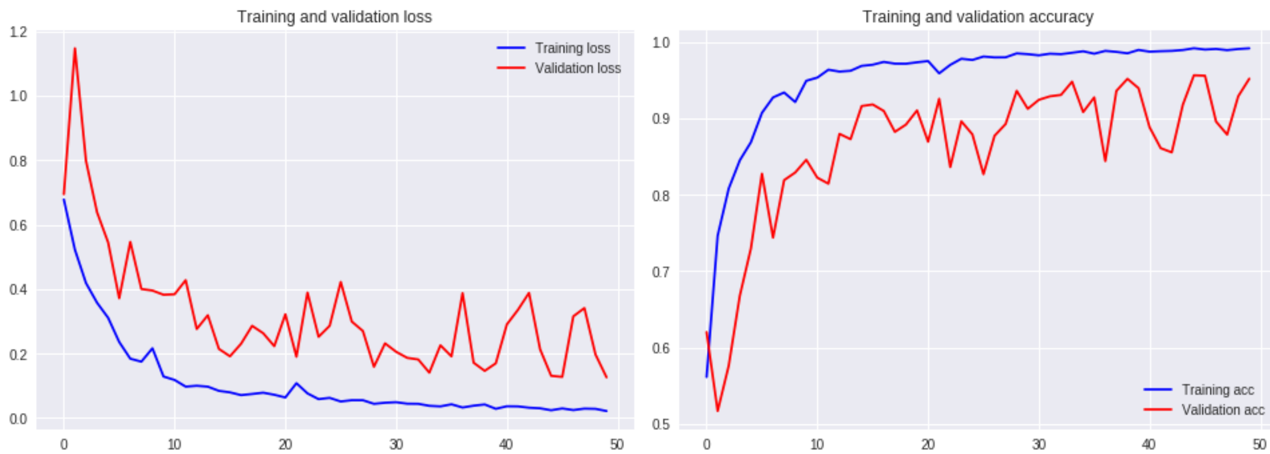
Layer (type)	Output Shape	Param #
DenseNet121 (Model)	(None, 200)	8292104
dense_3 (Dense)	(None, 200)	40200
batch_normalization_1 (Batch	(None, 200)	800
activation_1 (Activation)	(None, 200)	0
dense_4 (Dense)	(None, 200)	40200
batch_normalization_2 (Batch	(None, 200)	800
activation_2 (Activation)	(None, 200)	0
dropout_1 (Dropout)	(None, 200)	0
dense_5 (Dense)	(None, 2)	402
batch_normalization_3 (Batch	(None, 2)	8
activation_3 (Activation)	(None, 2)	0

Total params: 8,374,514
Trainable params: 7,278,606
Non-trainable params: 1,095,908

Figure 20: Table showing Pretrained DenseNet121 summary,On top of CNN block custom dense layes(Fcls) were added.some CNN layers were Freezed from learning @source:output log

6 HARWARE ACCELERATORS and LIBRARIES

Keras API is extensively used for building All CNN models. All the models were trained on Google colab,colab provides a vitual machine for each session that can last upto 12hrs ,and 90 mins of idle time.For every 12hrs Disk, RAM, VRAM, CPU cache etc data that is on our allotted virtual machine will get erased
GPU: 1xTesla T4 , 2560 CUDA cores, 320 Tensor Cores, compute 3.7, 16GB of (14.56 usable)GDDR6 VRAM,8.1 TFLOPS single-precision floating-point performance
CPU: 1xsingle core hyper threaded i.e(1 core, 2 threads) Xeon Processors @2.3Ghz (No Turbo Boost), 45MB Cache
RAM: 12.6 GB Available
Disk: 320 GB Available



b

Figure 21: Alexnet performance, LR=0.01,SGD ,50 epochs;8:2 validation split(22745+5685 images)

7 RESULTS

The following plots were obtained by plotting history of trained models. Sparse categorical cross entropy was chosen as loss function, (binary classification proble, binary cross entropy is same as this for our case). All the models were trained for 20 epochs ,on Google Colab. Dataset contains 28430 images of 2 classes(Background and Volcanic deformation) rougly equal number of images in each class. Transfer learning strategy was used with Pretrained models. pretrained weights are from model weights of Imagenet Data. Transfer learning stategy works perfectly well with ,VGG16, InceptionV3 but not in case of ResNet50,and DenseNet121.

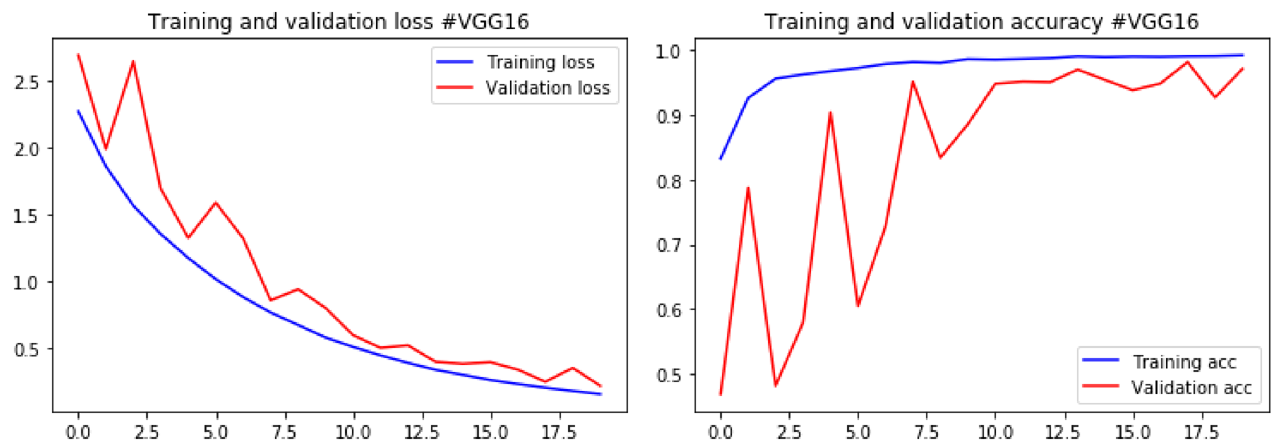


Figure 22: VGG16 performance, LR=0.001,SGD ,20 epochs;8:2 validation split(22745+5685 images)

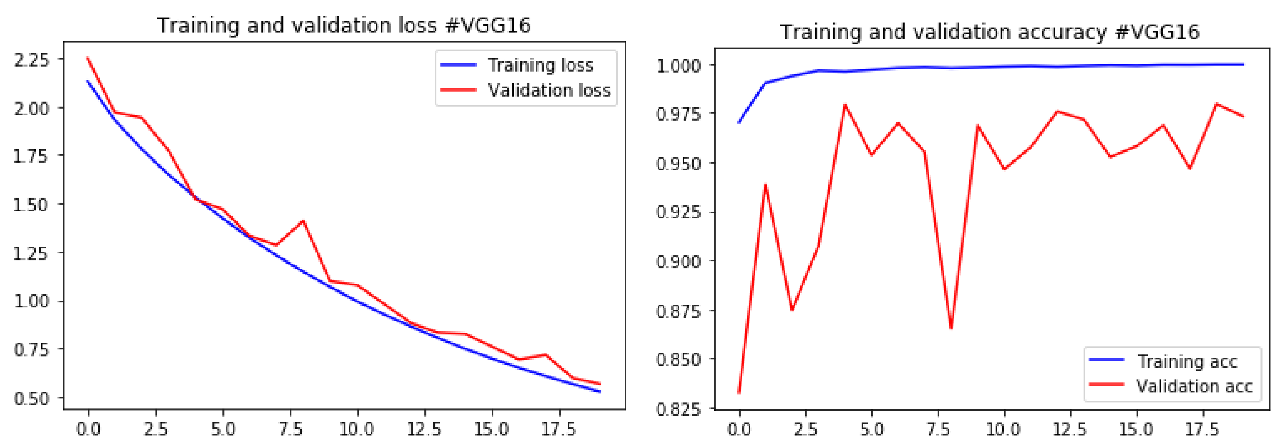


Figure 23: Pretrained VGG16 performance, LR=0.001,SGD ,20 epochs;8:2 validation split(22745+5685 images)

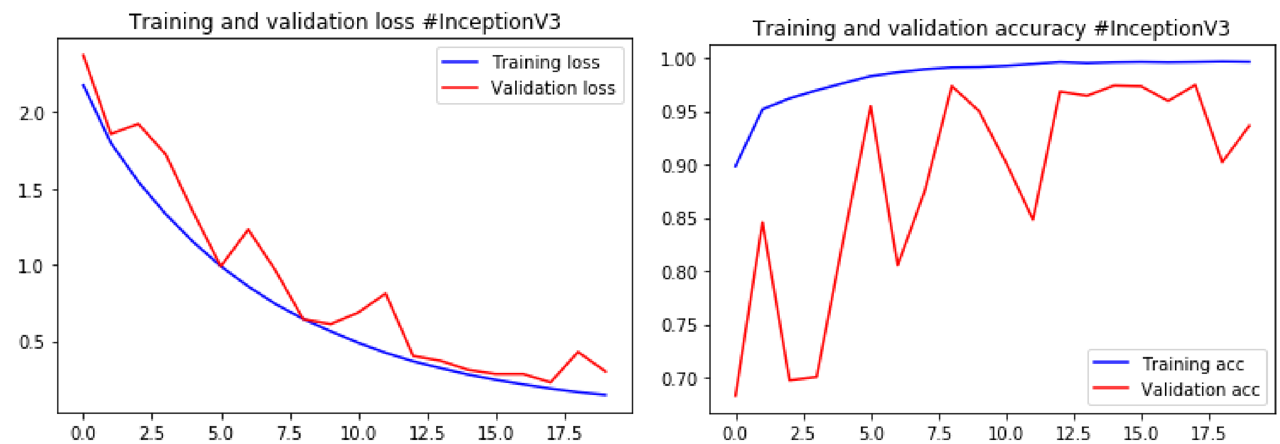


Figure 24: InceptionV3 Performance, LR=0.001,SGD ,20 epochs;8:2 validation split(22745+5685 images)

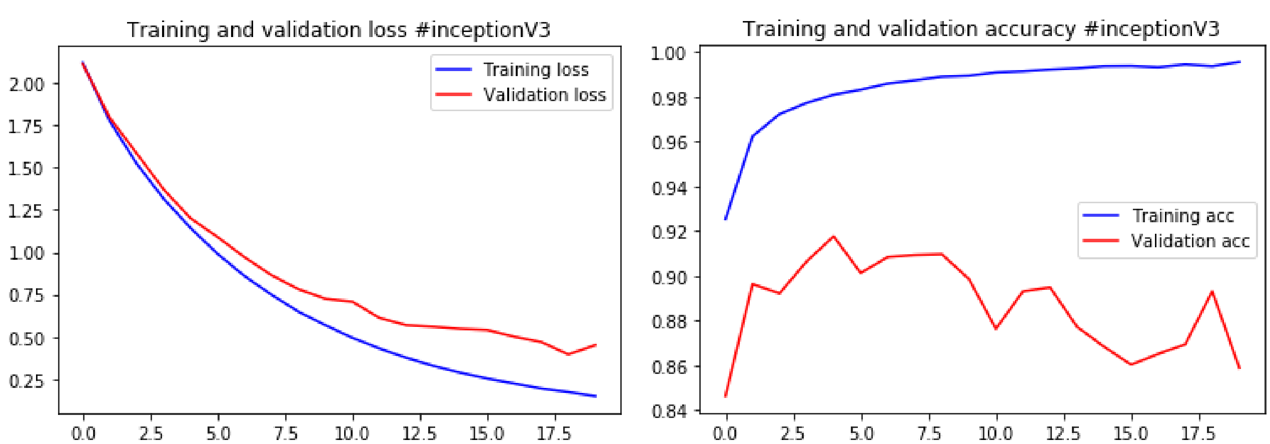


Figure 25: Pretrained InceptionV3 Performance, LR=0.001,SGD ,20 epochs;8:2 validation split(22745+5685 images), Pretrained on Imagenet dataset

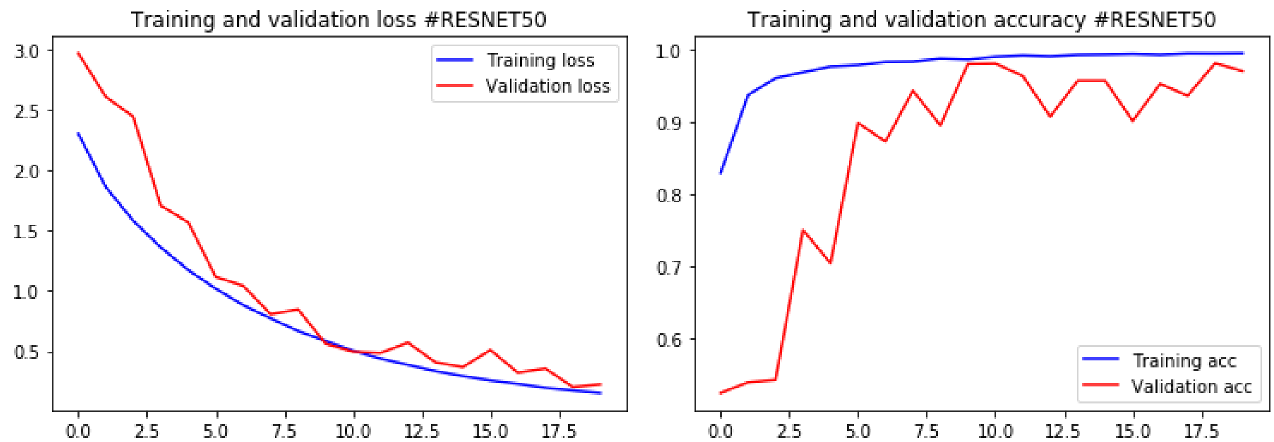


Figure 26: ResNet50 Performance, LR=0.001,SGD ,20 epochs;8:2 validation split(22745+5685 images)

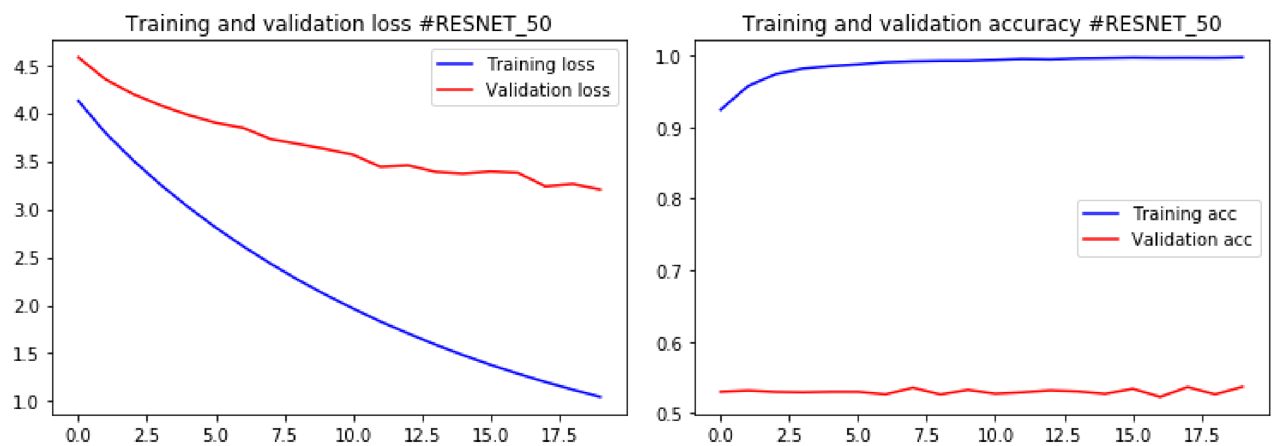


Figure 27: ResNet50 Performance, LR=0.001,SGD ,20 epochs;8:2 validation split(22745+5685 images),Pretrained on Imagenet dataset

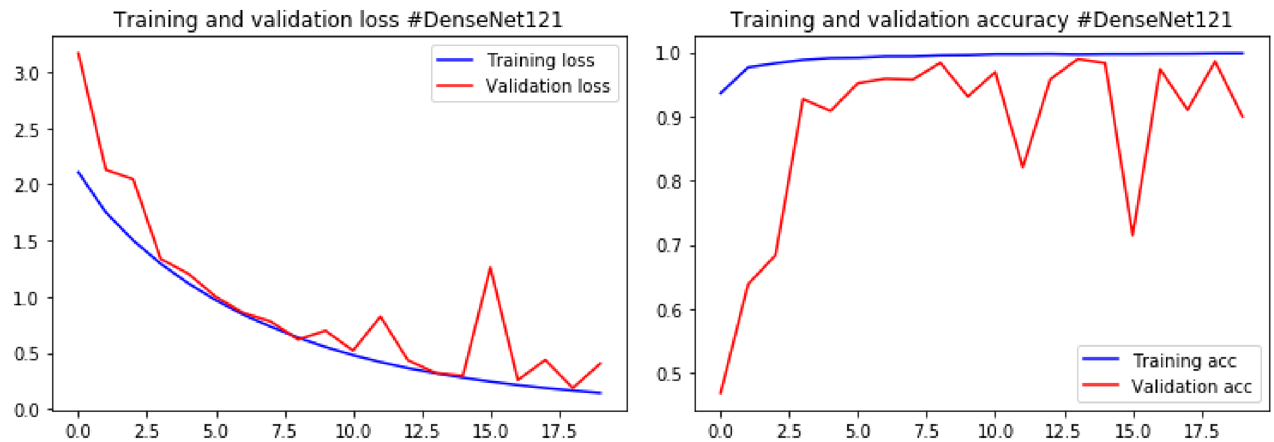


Figure 28: DenseNet Performance, LR=0.001,SGD ,20 epochs;8:2 validation split(22745+5685 images)

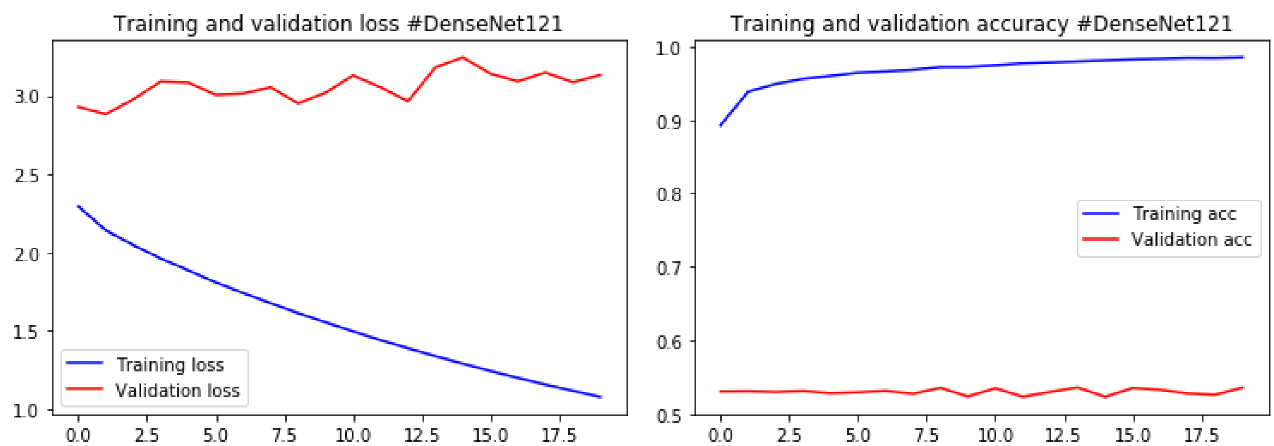


Figure 29: Pretrained DenseNet121 Performance, LR=0.001,SGD ,20 epochs;8:2 validation split(22745+5685 images),Pretrained on Imagenet dataset

Summary of Results,trained on volcano dataset,28430 images ($224 \times 224 \times 3$)							
CNN	Epochs	Learning Rate	Time per epoch	validation split	Trainable parameters	Non trainable parameters	validation accuracy (%) avg
AlexNet	50	0.01	~220s	0.2	62,378,344	—	93
VGG16	20	0.001	390s	0.2	15,526,606	804	95
VGG16-pretrained	20	0.001	312s	0.2	12,610,958	2,916,452	97
InceptionV3	20	0.001	354s	0.2	24,153,134	35,236	93
InceptionV3-Pretrained	20	0.001	300s	0.2	13,499,662	10,688,708	85
ResNet50	20	0.001	384s	0.2	25,919,374	53,924	97
ResNet50-pretrained	20	0.001	284s	0.2	23,527,558	20,172,964	53
DensenNet121	20	0.001	390s	0.2	8,290,062	84,452	94
DenseNet121-pretrained	20	0.0005	335s	0.2	7,278,606	1,095,908	53

8 Inference and Continuation of work..

Now we have seen the performance of various CNN models on volcanic dataset, both incase of Pretrained and learning from scratch. Following statements can be infered from traing the models

1. Transfer learning works most often but not always, and it particularly in case of ResNet and DenseNets(both of which have connections from their previous layers in their fundamental blocks. It is believed that Transfer learning strategy works well with data sets from similar domain but our case imagenet dataset and Volcanic data are completely different domains but still VGG16 and Inception worked pretty well.this could be due to fact that CNN layers in bottom learn most generalized feature of a natural image.
2.Choice Hyperparameters greatly effects model's overall performance.
3.Size of the network or Number of parameters always does not always gaurantee better performance. (DenseNets performed better than Resnets)
4.Higer batch size helps model to better generalize. Note that an ideal model's performance should not depend of order of inputs that we give. lower the batch size per epoch more possible ways of giving inputs,less generalized is the model.
These trained models were saved in modelname.h5 file, and will be used for transfer learning in Mining Dataset. Mining dataset should be trained with SPPlayer on top of CNN block.And Multisized training should be done to make the model learn better.

9 References

[1]Jeremy Jordan,19 APRIL 2018,Atricle title:"Common architectures in convolutional neural networks."

[2]Sebastian Ruder,19 JANUARY 2016,Article title:"An overview of gradient descent optimization algorithms"

[3]Jason Yosinski,1 Jeff Clune,2 Yoshua Bengio,3 and Hod Lipson4,"How transferable are features in deep neural networks?"arxiv:1411.1792

[4]Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition"arXiv:1406.4729

[5] Gao Huang Cornell University,Zhuang Liu Tsinghua University,Laurens van der Maaten,Kilian Q. Weinberger "Densely Connected Convolutional Networks"arXiv:1608.06993

[6]Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna "Rethinking the Inception Architecture for Computer Vision"arXiv:1512.00567

[7]Karen Simonyan, Andrew Zisserman,"Very Deep Convolutional Networks for Large-Scale Image Recognition"arXiv:1409.1556

[8]Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun"Deep Residual Learning for Image Recognition" arXiv:1512.03385

[9]Alex Krizhevsky,Ilya Sutskever,Geoffrey E. Hinton"ImageNet Classification with Deep Convolutional Neural Networks"[2012]

[10]N. Anantrasirichai1 , J. Biggs2 , F. Albino2 , P. Hill1, and D. Bull1 "Application of Machine Learning to Classification of Volcanic Deformation in Routinely Generated InSAR Data"AGU100 Research article:10.1029/2018JB015911

[11] Ullaby book,Chapter 14, Real Synthetic Aperture radar side looking airborne Radar