Recent Advances in Machine Learning

DeepFake Detection

Chair: Dr.-Ing. Margret Keuper
Team members: Soham Kalghatgi
Pruthvi Radadiya

What are DeepFakes?

Deepfakes referes to fake content created using Deep Learning **(Deep Learning + Fake)**. It is process where an existing image or video of a person is taken and replaced with someone else's likeness using artificial neural networks.

How are Deepfakes created?

Machine Learning techniques like encoders and Generative Adversarial Networks (GAN) are used to create DeepFakes

Image dataset

Real Images: ImageWoof dataset
Fake Images: generated using SNGAN and unpampled using-

- bilinear interpolation upsampling
- bicubic interpolation
- pixel shuffle upsampling

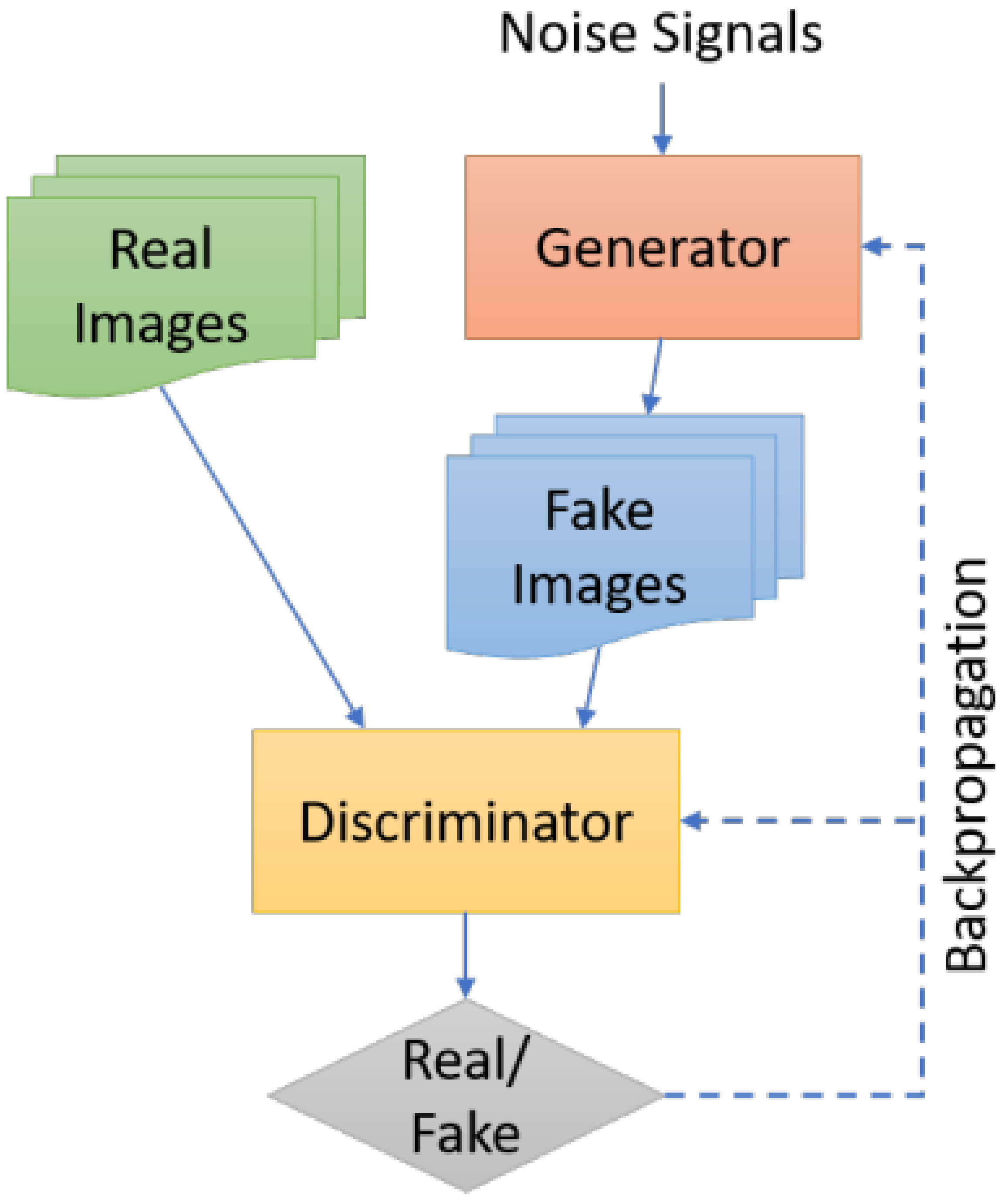
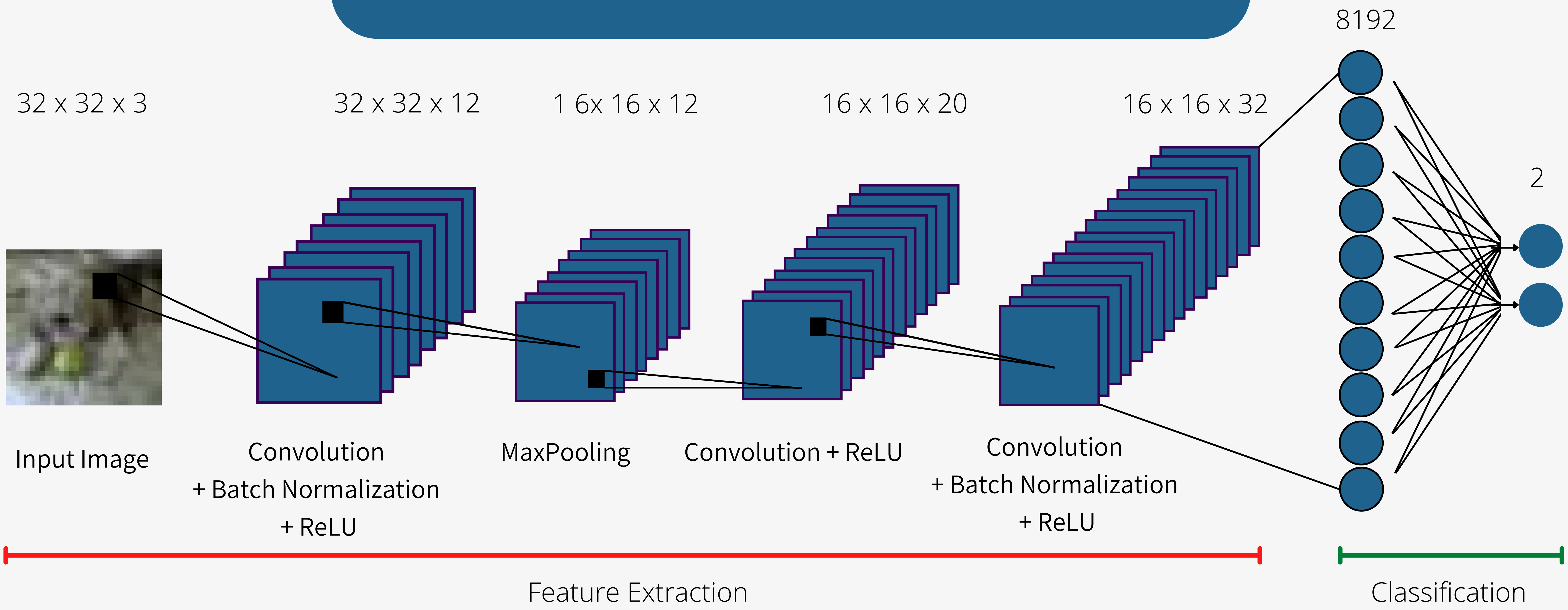


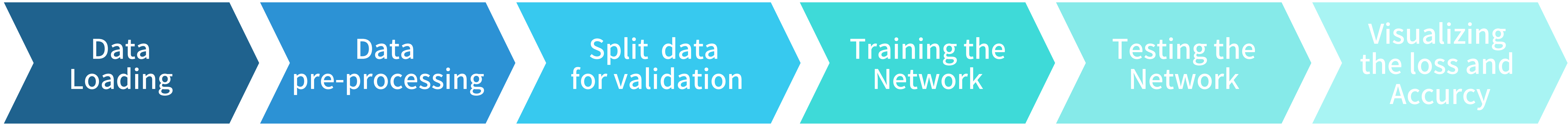
Figure 1. The GAN architecture for Fake image generation.

CNN model archicture



The proposed classifier consists of CNN model as its base which is then appended with batch normalization, max pooling and a two node dense layer. The two nodes in last dense layer in the architecture proposed are used for two final classes (real and fake). Batch normalization layer is used for normalization and scaling for inputs from previous layer.

Network Pipeline



Data Loading

Data pre-processing

Split data for validation

Training the Network

Testing the Network

Visualizing the loss and Accury

- The model is trained with a max number of 50 epochs, for a batch size of 40.
- The loss function used is Cross entropy Loss.
- The optimizer used is Adam.
- In order to prevent over-fitting, early stop is implemented.

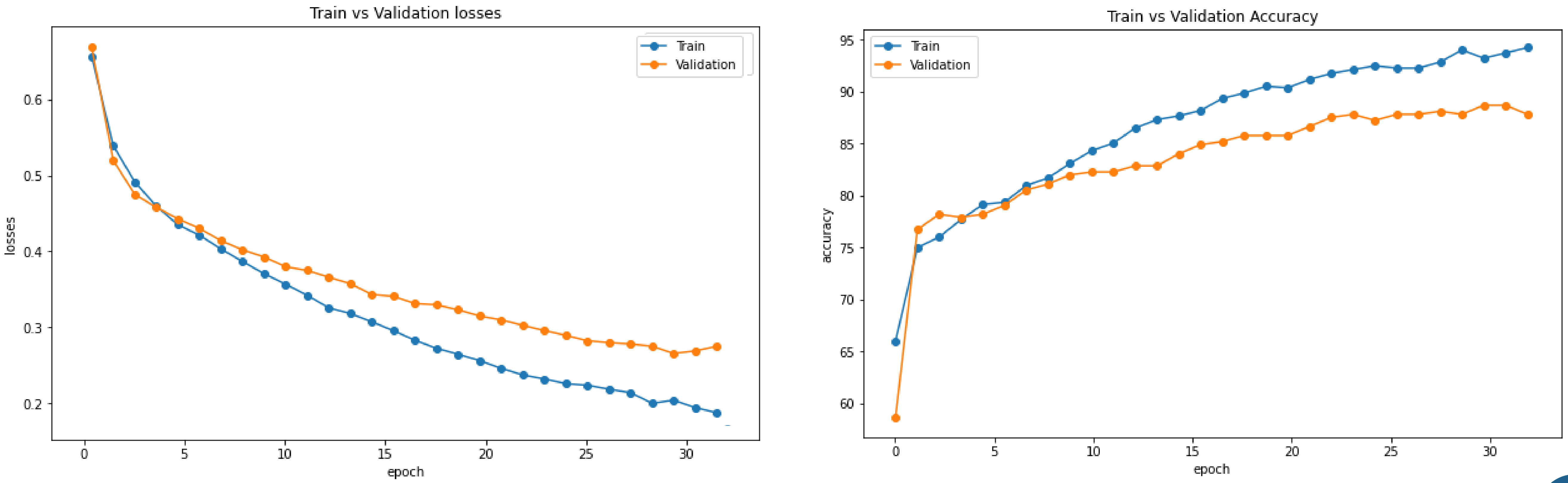


Figure 2. Loss and Accuracy plots for Bilinear data.

Results

Training Data	Testing Data	Training Loss	Training Accuracy	Testing Loss	Testing Accuracy
Bicubic	Bicubic	0.418707	92.0312	0.484526	**82.165**
	Bilinear	0.418707	92.0312	0.557916	75.759
	Pixel Shuffle	0.418707	92.0312	0.542024	75.542
	Combined Data	0.418707	92.0312	0.999258	23.625
Bilinear	Bicubic	0.381744	96.25	0.594669	68.75
	Bilinear	0.381744	96.25	0.428791	**90.462**
	Pixel Shuffle	0.381744	96.25	0.591304	70.846
	Combined Data	0.381744	96.25	0.938759	29.757
Pixel Shuffle	Bicubic	0.445331	87.9688	0.54742	75.523
	Bilinear	0.445331	87.9688	0.59253	70.756
	Pixel Shuffle	0.445331	87.9688	0.526125	**78.25**
	Combined Data	0.445331	87.9688	1.02465	23.1667
Combined Data	Bicubic	0.464932	75.154	0.888184	23.254
	Bilinear	0.464932	75.154	0.903998	22.751
	Pixel Shuffle	0.464932	75.154	0.877364	23.167
	Combined Data	0.464932	75.154	0.49304	**83.756**

Observations

The network was constructed using a CNN with three convolution layers, increase or decrease in the convolution layers results in decrease in accuracy.

CNN layers

The accuracy of the network drastically reduced when trained and tested for combined data when the entire dataset of Bilinear, Bicubic and Pixel shuffle was balanced. (1:1 ratio for real and fake data).

Data Imbalance

The network was designed with one fully connected layer with ReLu activation. Addition of hidden layers decreased the accuracy.

Dense layers

Maximum accuracy was observed when the network was trained for Bilinear data and tested with Bilinear data.

Max Accuracy

Adam optimizer provided us with higher accuracy than the Stochastic Gradient Descent.

Optimizer

Greyscale transformation resulted in decreased accuracy.

Data Pre-processing

Challenges

PYTHON LIBRARIES

DATA LOADING

Unawareness about Data loading method and the structure of the loaded data.

CNN LAYERS AND ACTIVATION FUNCTION

Number of convolution and fully connected layers and the activation function to be used

OVERFITTING

Initially, our model was overfitting, later we implemented early stop to prevent overfitting

Timeline

1

APRIL

- Project Introduction
- Understanding the project

2

MAY

- Literature Review
- Python
- Data Preprocessing and Loading

3

JUNE

- Building the network
- Training
- Testing
- Plotting the observations

4

JULY

- Training and testing for different data samples and tuining the model.
- Final Presentation
- Report writing

Thank you