# Galaxy Image Classification

By Group 1 - Jyoti Sharma, Mishkin khunger, Tanvi Hindwan

# Road Map of Galaxy Zoo

- Introduction
- Dataset Detail
- Problem Statement
- Data Preprocessing
- Models
- Results
- Conclusion/Future Work

# About Galaxy Zoo Challenge

Galaxy Zoo - Classify the morphologies of distant galaxies in our Universe.

Dataset asks to analyze the JPG images of galaxies and  find an automated metrics that reproduce the probability distributions derived from human classifications.

For each galaxy, determine the probability that it belongs in a particular class.

# Dataset Details

Images_training : JPG images of 61578 galaxies

Images_test : JPG images of 79975 galaxies.

Targets : 37 classes with Probability distributions for the classifications for each of the training images. (Classes represent the morphology (or shape) of the galaxy in 37 different categories).

URL: https://www.kaggle.com/c/galaxy-zoo-the-galaxy-challenge/data
https://www.kaggle.com/c/galaxy-zoo-the-galaxy-challenge/data

# Problem Statement

- To develop a Neural Network to classify Galaxy Images for 37 different categories with probabilities ranging between 0 to 1.

- Multi label Classification Problem.

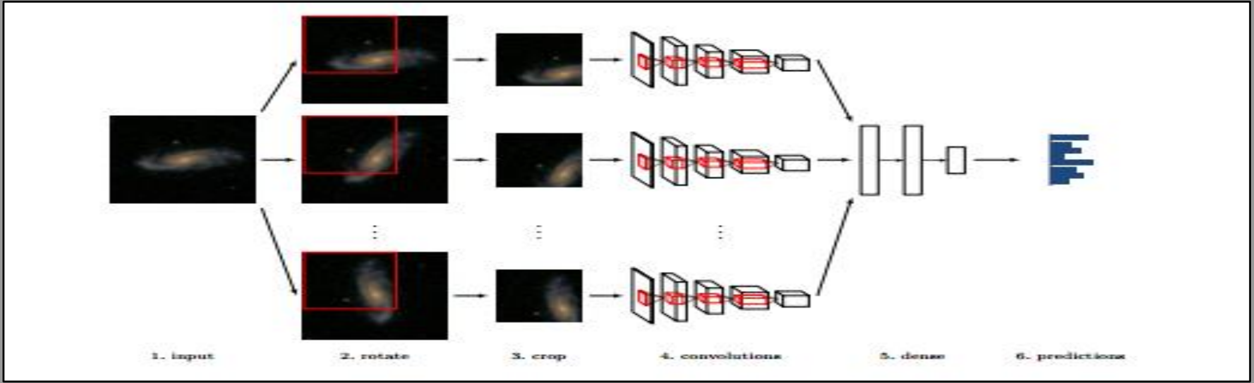
# Data Preprocessing

Original Image - 424 x 424

Center Crop - 256 x 256

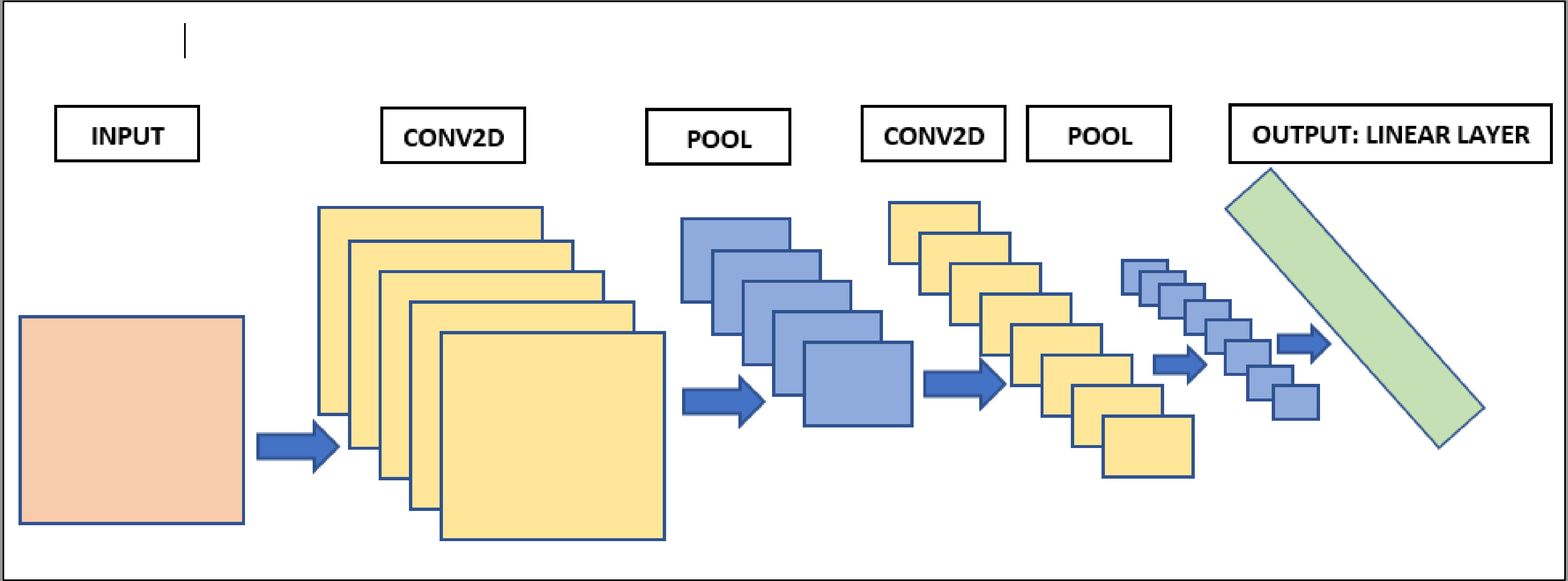
Image Shape - 3 x 64 x 64  (RGB)

Target labels - One-hot encoded using Multi-label Binarizer

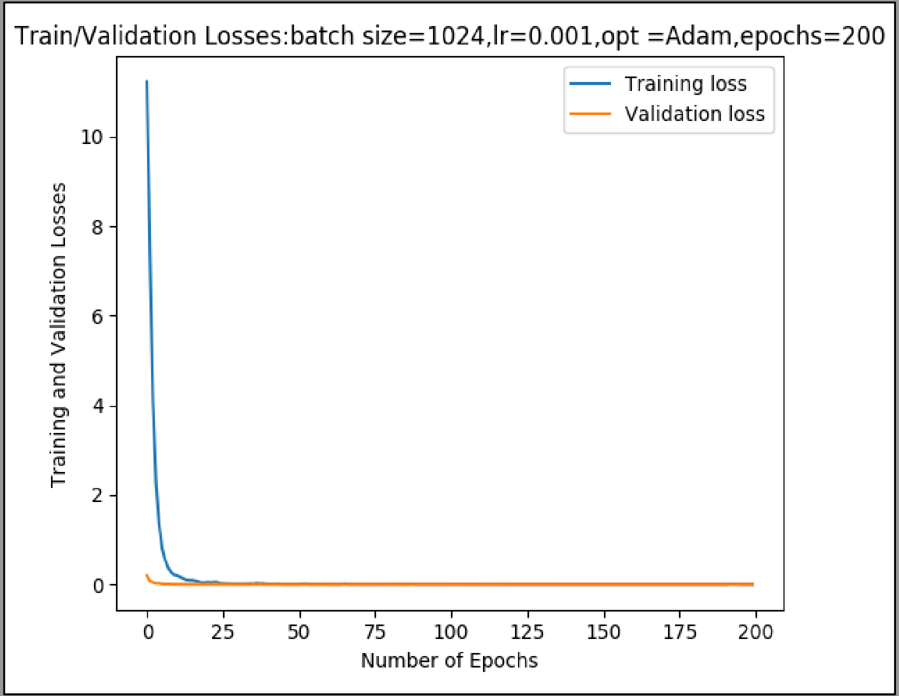



https://www.technologyreview.com/2015/04/03/11482/how-machine-vision-is-reinventing-the-study-of-galaxies/

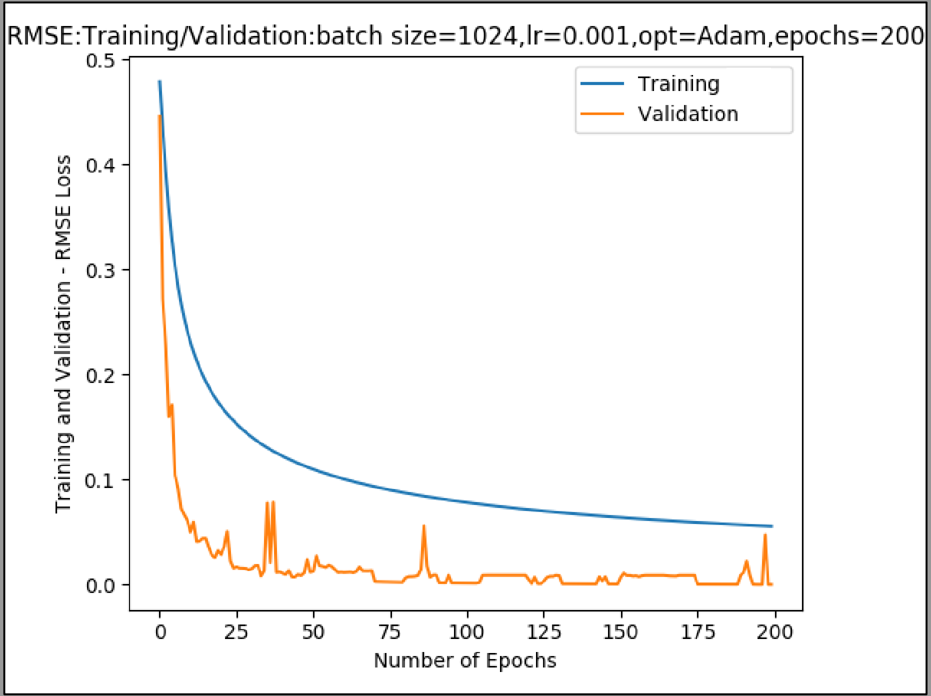
# CNN Architecture



# Training-Validation Set Results




Train/Validate Set Losses



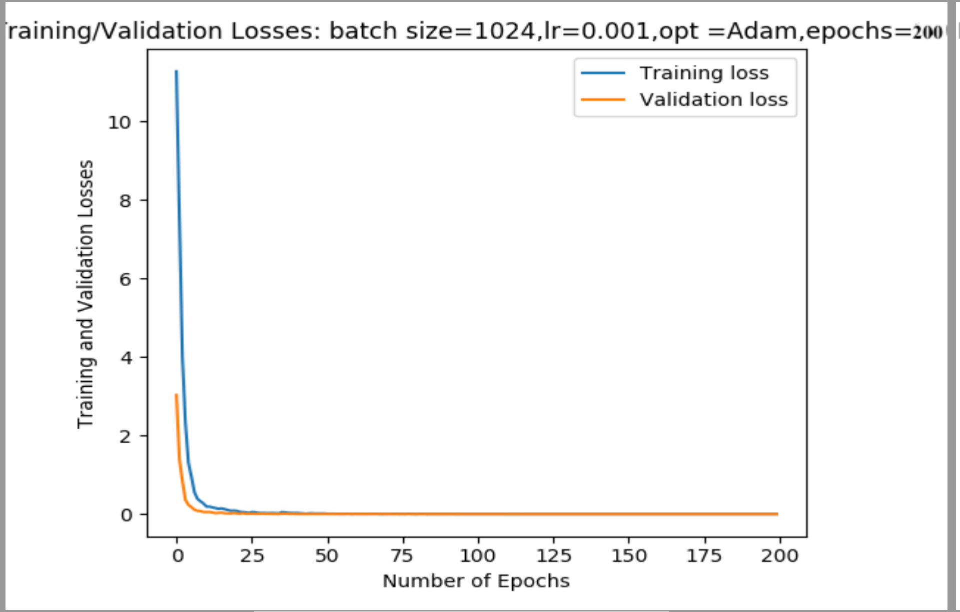

RMSE on Train/validate Set

# Data Augmentation

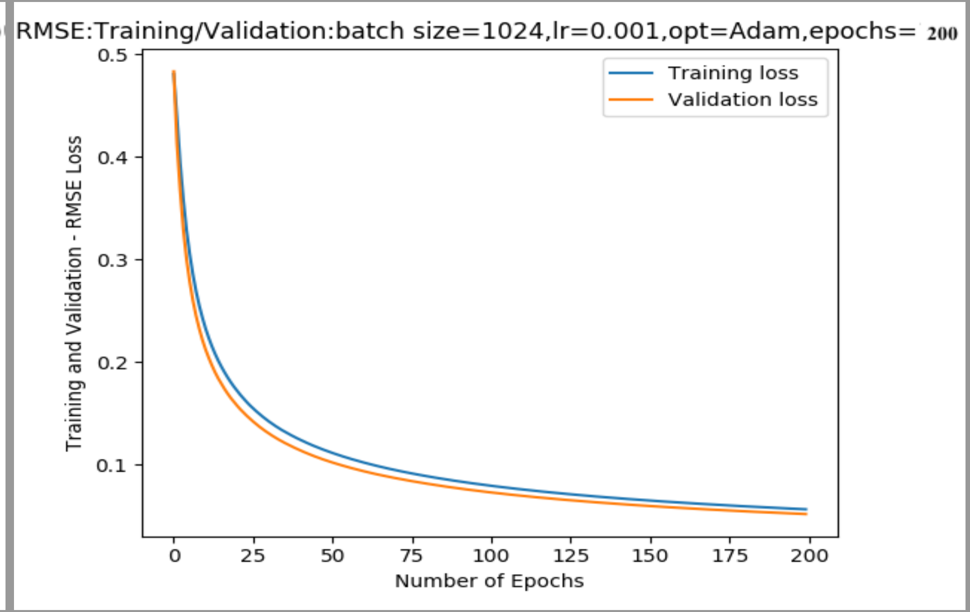
Two types of augmentation functionality used:

- Rotation: random with angle between 0° and 90° (uniform)
- Random Flip: flip the given Image randomly with a given probability.

# Data Augmentation-Training Validation Set Results




Train/Validate Set Losses




RMSE on Train/validate Set

# Output for CNN Model

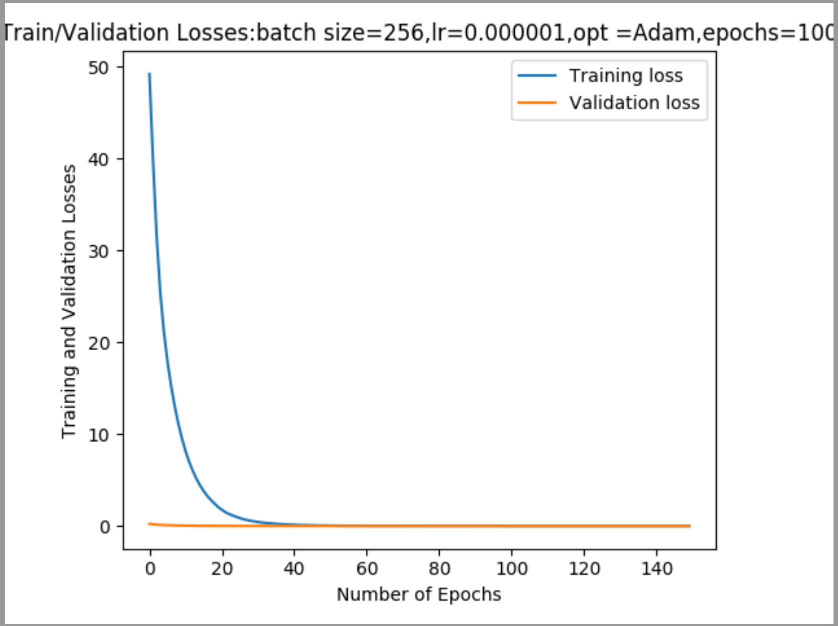
tensor([[0.0135, 0.0133, 0.0133, 0.0123, 0.0125, 0.0141, 0.0145, 0.0120, 0.0121, 0.0131, 0.0138, 0.0130, 0.0146, 0.0154, 0.0122, 0.0137, 0.0128, 0.0135, 0.0132, 0.0135, 0.0121, 0.0131, 0.0133, 0.0145, 0.0138, 0.0116, 0.0158, 0.0139, 0.0131, 0.0137, 0.0127, 0.0137, 0.0131, 0.0133, 0.0124, 0.0135, 0.0110]])

# Pretrained Model - ResNet50

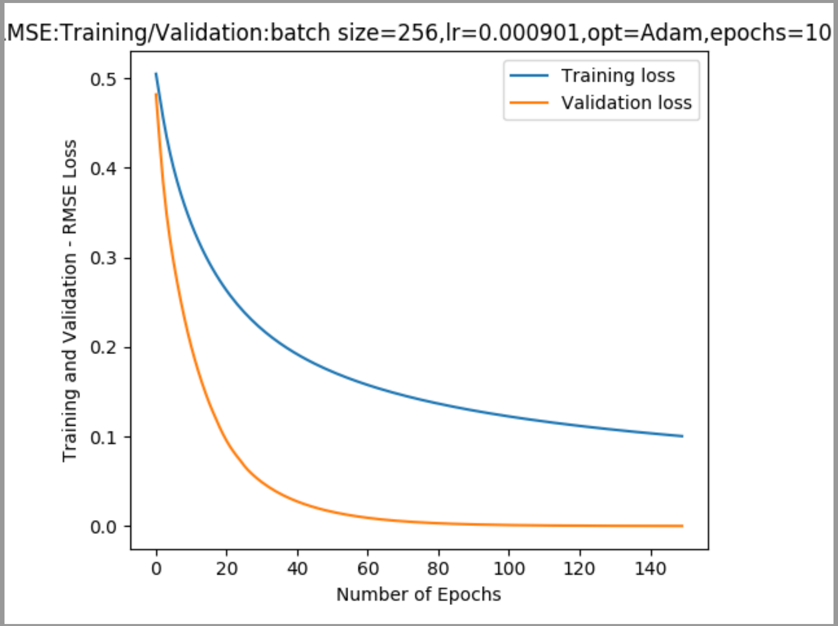
- 50 layers deep convolutional neural network.
- Architecture:

| layer name | output size | 18-layer | 34-layer | 50-layer | 101-layer | 152-layer |
|---|---|---|---|---|---|---|
| conv1 | $112\times112$ | $7\times7$, 64, stride 2 | | | | |
| conv2_x | $56\times56$ | $3\times3$ max pool, stride 2 | | | | |
| | | $\begin{bmatrix}3\times3, 64\\ 3\times3, 64\end{bmatrix}\times2$ | $\begin{bmatrix}3\times3, 64\\ 3\times3, 64\end{bmatrix}\times3$ | $\begin{bmatrix}1\times1, 64\\ 3\times3, 64\\ 1\times1, 256\end{bmatrix}\times3$ | $\begin{bmatrix}1\times1, 64\\ 3\times3, 64\\ 1\times1, 256\end{bmatrix}\times3$ | $\begin{bmatrix}1\times1, 64\\ 3\times3, 64\\ 1\times1, 256\end{bmatrix}\times3$ |
| conv3_x | $28\times28$ | $\begin{bmatrix}3\times3, 128\\ 3\times3, 128\end{bmatrix}\times2$ | $\begin{bmatrix}3\times3, 128\\ 3\times3, 128\end{bmatrix}\times4$ | $\begin{bmatrix}1\times1, 128\\ 3\times3, 128\\ 1\times1, 512\end{bmatrix}\times4$ | $\begin{bmatrix}1\times1, 128\\ 3\times3, 128\\ 1\times1, 512\end{bmatrix}\times4$ | $\begin{bmatrix}1\times1, 128\\ 3\times3, 128\\ 1\times1, 512\end{bmatrix}\times8$ |
| conv4_x | $14\times14$ | $\begin{bmatrix}3\times3, 256\\ 3\times3, 256\end{bmatrix}\times2$ | $\begin{bmatrix}3\times3, 256\\ 3\times3, 256\end{bmatrix}\times6$ | $\begin{bmatrix}1\times1, 256\\ 3\times3, 256\\ 1\times1, 1024\end{bmatrix}\times6$ | $\begin{bmatrix}1\times1, 256\\ 3\times3, 256\\ 1\times1, 1024\end{bmatrix}\times23$ | $\begin{bmatrix}1\times1, 256\\ 3\times3, 256\\ 1\times1, 1024\end{bmatrix}\times36$ |
| conv5_x | $7\times7$ | $\begin{bmatrix}3\times3, 512\\ 3\times3, 512\end{bmatrix}\times2$ | $\begin{bmatrix}3\times3, 512\\ 3\times3, 512\end{bmatrix}\times3$ | $\begin{bmatrix}1\times1, 512\\ 3\times3, 512\\ 1\times1, 2048\end{bmatrix}\times3$ | $\begin{bmatrix}1\times1, 512\\ 3\times3, 512\\ 1\times1, 2048\end{bmatrix}\times3$ | $\begin{bmatrix}1\times1, 512\\ 3\times3, 512\\ 1\times1, 2048\end{bmatrix}\times3$ |
| | $1\times1$ | average pool, 1000-d fc, softmax | | | | |
| FLOPs | | $1.8\times10^9$ | $3.6\times10^9$ | $3.8\times10^9$ | $7.6\times10^9$ | $11.3\times10^9$ |

# ResNet50 Results




Train-test Loss




Train test RMSE

# Conclusion and Future work

- On comparing all the three models, RMSE values for CNN model with Data augmentation was the best.
- RMSE can be improved by trying more pre trained model except resnet-18 and resnet 34.
- Another approach for data augmentation can be tried.