STRUCTURE-PROPERTY LINKAGE USING MACHINE LEARNING

A Thesis submitted in partial fulfillment of
the requirements for the degree of

Master of Technology

by

SUBHANSHU RANJAN TIWARI
(Roll No. 183100059)

Under the guidance
of
Prof. Alankar Alankar

DEPARTMENT OF MECHANICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY–BOMBAY

June 30, 2020

Dissertation Approval

The dissertation entitled

STRUCTURE-PROPERTY LINKAGE USING MACHINE LEARNING

by

SUBHANSHU RANJAN TIWARI
(Roll No. 183100059)

is approved for the degree of

Master of Technology

Examiner	Examiner
Guide	Chairman

Date: ________________

Place: ________________

Declaration

I declare that this written submission represents my ideas in my own words and where others ideas or words have been included I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Signature)

Date: June 30, 2020

Name: SUBHANSHU RANJAN TIWARI

Place: IIT Bombay, Mumbai

Roll No: 183100059

Abstract

All Material deform in response to externally applied force. This deformation may be permanent or temporary depending on various factors like type of material, type of stress, shape of material, magnitude of stress etc. Temporary deformation disappears after removal of force while permanent deformation stays after the removal of force. This phenomenon can be explained by dislocation movement and generation in a material. Material science coupled with machine learning tend to reduce the computational efforts and time that involves dislocation dynamics but at the same time provide some good authentic results. Both fundamental and applied research are considerably speeding up with the help of statistical tools. This work deals with the predicting number of dislocations and to create a Machine learning model that can predict number and type of dislocations that are present in a sample among the given set of Dislocation. We collected data of different dislocations by using simulation software and after strained for sufficient amount of time, they are converted to usable data i.e. fingerprint. Fingerprint are used for classification using machine learning. Classification of different dislocation structure was done using Convolution Neural Network. Apart from classification of different initial structure, we tried classifying internal structure at multiple strain using Convolution Neural Network so that we will be confident that our Fingerprint is unique at any point of strain.
After successful classification, we tried mapping of structure with property i.e. given a structure and strain level what should be the stress on that unit cell.
Keywords: Dislocation, Convolution Neural Network, Machine Learning, Stress and Structure property correlation, Strain.

Contents

Abstract **i**

List of Figures **iv**

1 Introduction **1**

1.1 Machine Learning 2

1.1.1 Types of Machine Learning Problems 2

1.1.2 Learning Algorithm 5

1.1.3 Overfitting Problem 5

1.2 Structure Property Correlation 7

2 Literature Review **9**

2.1 Dislocations - Line Defect in crystals 9

2.1.1 Edge Dislocation 10

2.1.2 Screw Dislocation 10

2.1.3 Dislocation Dynamics 12

2.2 K- fold Cross Validation 12

2.3 Regression 14

2.4 Neural Network 15

2.5 Convolution Neural Network 17

2.5.1 ConvNet Architecture 18

3 Methodology **21**

3.1 Data Generation 21

3.2 Data Analysis 23

3.2.1 Stress - Strain Analysis 23

3.2.2 Segment length Analysis . . . 24
3.2.3 Resolved shear stress on different slip system . . . 28
3.3 CNN model . . . 31
3.4 Vanilla Deep Regression . . . 32

4 Result and Discussion **35**
4.1 Understanding of data . . . 35
4.2 Converting 3D image into 2D image . . . 43
4.3 Learning Curves . . . 43
4.4 Prediction Result . . . 45
4.5 Preprocessing of data for VDR . . . 46
4.6 Training of data for VDR . . . 48
4.6.1 Case 1 . . . 51
4.6.2 Case 2 . . . 52
4.6.3 Case 3 . . . 53
4.6.4 Case 4 . . . 54
4.6.5 Case 5 . . . 55

5 Summary **57**

6 Future Work **58**

References **59**

Acknowledgments **61**

List of Figures

1.1 Classical Programming vs Machine Learning [1] 2
1.2 Unsupervised Learning [2] 3
1.3 Supervised Learning [2] 4
1.4 Bias - Variance Trade off [3] 6
1.5 Resolved sheer stress vs Resolved sheer strain in Cu crystal showing hardening stages [4] 8

2.1 Shadowgraph of Slip bands in Aluminium crystal at Room Temperature[17] 9
2.2 Edge dislocation Movement[17] 10
2.3 Screw Dislocation[17] 11
2.4 5- Fold cross validation-1 [5] 13
2.5 5- Fold cross validation-2[5] 13
2.6 Linear regression line[19] 14
2.7 Multiple Linear regression plane [6] 15
2.8 A Simple 3 Layer Neural network [7] 16
2.9 various matrix in neural network[7] 17
2.10 Arrangement of Neurons in 3 dimension(width,height and depth)[7] 18
2.11 Convolution layer [8] 19
2.12 Max pooling layer [9] 19
2.13 Flattening layer [10] 20
2.14 Fully Connected layer acting as input to neural network for classification [11] 20

3.1 Dislocation evolution in a single simulation 21
3.2 Initially undeformed Dislocations in a 3D unit cell 22
3.3 Stress strain curve for 28 Initial dislocation line 23

3.4 Segment length distribution at 0.5% strain on a slip plane for 28 initial dislocation line 24
3.5 Segment length distribution at 1% strain on a slip plane for 28 initial dislocation line 25
3.6 Segment length distribution at 1.5% strain on a slip plane for 28 initial dislocation line 25
3.7 Segment length distribution at 2% strain on a slip plane for 28 initial dislocation line 26
3.8 Segment length distribution at 0.5% strain on a slip plane for 28 initial dislocation line 26
3.9 Segment length distribution at 1% strain on a slip plane for 28 initial dislocation line 27
3.10 Segment length distribution at 1.5% strain on a slip plane for 28 initial dislocation line 27
3.11 Segment length distribution at 2% strain on a slip plane for 30 initial dislocation line 28
3.12 Resolved shear stress on slip system 1 and 2 with strain 29
3.13 Resolved shear stress on slip system 1 and 2 with strain 29
3.14 Resolved shear stress on slip system 1 and 2 with strain 30
3.15 Resolved shear stress on slip system 1 and 2 with strain 30
3.16 Resolved shear stress on slip system 1 and 2 with strain 30
3.17 Resolved shear stress on slip system 1 and 2 with strain 31
3.18 Summary of CNN model 32
3.19 High-level view of the Nvidia's data collection system for self driving cars [20] 33
3.20 Summary of Vanilla regression architecture 34

4.1 Stress strain curves for multiple simulations with 20 initial dislocation line 35
4.2 Stress strain curves for multiple simulations with 28 initial dislocation line 36
4.3 Stress strain curves for multiple simulations with 30 initial dislocation line 36
4.4 Stress strain curves for multiple simulations with 32 initial dislocation line 37
4.5 Stress strain curves for multiple simulations with 34 initial dislocation line 37
4.6 Stress strain curves for multiple simulations with 36 initial dislocation line 38

4.7 Stress strain curves for multiple simulations with 38 initial dislocation line 38
4.8 Stress strain curves for multiple simulations with 40 initial dislocation line 39
4.9 Stress strain curves for multiple simulations with 42 initial dislocation line 39
4.10 Stress strain curves for multiple simulations with 60 initial dislocation line 40
4.11 Stress strain curves for multiple simulations with 80 initial dislocation line 40
4.12 Stress strain curves for multiple simulations with 90 initial dislocation line 41
4.13 Stress strain curves for multiple simulations with 96 initial dislocation line 41
4.14 Stress strain curves for multiple simulations with 100 initial dislocation line 42
4.15 Stress strain curves for multiple simulations with 128 initial dislocation line 42
4.16 A sample converted 3D image 43
4.17 Accuracy of training dataset and validation dataset 44
4.18 Loss in training and validation dataset 44
4.20 Predicted Results 45
4.22 Predicted Results 46
4.23 Mapping of Stress-strain value with file name 47
4.24 Mapping of Stress-strain value with file name and Image(Fingerprint) . . . 48
4.25 NVIDIA Architecture for Self Driving cars 49
4.26 Training and testing loss 50
4.27 Actual and predicted Value of stress 50
4.28 Training and testing loss 51
4.29 Actual and predicted Value of stress 51
4.30 Training and testing loss 52
4.31 Actual and predicted Value of stress 52
4.32 Training and testing loss 53
4.33 Actual and predicted Value of stress 53
4.34 Training and testing loss 54
4.35 Actual and predicted Value of stress 54
4.36 Training and testing loss 55
4.37 Actual and predicted Value of stress 55
4.38 Actual and predicted Value of stress 56

Chapter 1

Introduction

Material science coupled with computational methods can handle complex problems that are hard to solve or take enormous amount of time to solve. Both academia and industrial researchers are now shifting towards solutions that machine itself can solve, generate and propose high quality materials with designed properties that are tailored to fit the needs of specific application.

Movement of dislocation in crystalline materials is one of the primary mechanism of plastic deformation in crystalline materials. These are one- dimensional lattice defects, due to these distortion in the crystallographic lattice occur. When dislocations density is increased they tend to increase the strength of materials but above a certain limit of density, strength of material decreases. When subjected to sufficiently larger stress, dislocations tend to move through crystallographic planes known as slip planes. Apart from their interaction with stress field, they may interact with other type of defect and can result into their movement perpendicular to slip plane, known as climb motion. Dislocations may generate along a slip plane when a crystal is deformed, the source of generation of dislocation when crystal is deformed is known as frank reed source.[13]

To identify the patterns governing the behaviour recent advances in machine learning have been employed, this pattern may include behaviour of grains, dislocations and physical phenomenon. Machine learning algorithms are even predicting results without conducting costly experiments. Rise of materials informatics in material science is driven by database that is being available along with ground breaking advances in deep learning,deep learning has emerged as a game changing technique. [14]

However, given the advancement in machine learning, one cannot be fully sure about the results, they are also designed with some accuracy. Testing of robustness, type of loss functions involve, training data, machine learning architecture used to train model are some bottleneck that need to be properly handle.Despite the challenges, potential social impact of accomplishments is huge; findings may reduce the time of simulations and accelerate material research paving the way for industrial revolution and innovation.

1.1 Machine Learning

It is a method of automating analytical model building. Model is trained with many example relevant to the task and find a statistical structure in those examples that eventually allows the system to come up with rules for automating the task. These extracted patterns or the model can be used to to assist in decision-making processes or automate in predicting unknown data .Machine learning (ML) is trained rather than explicitly programmed. ML algorithms are also applied in the field of finances, speech processing, navigation, game playing, personality profiling, media, health care and many others.

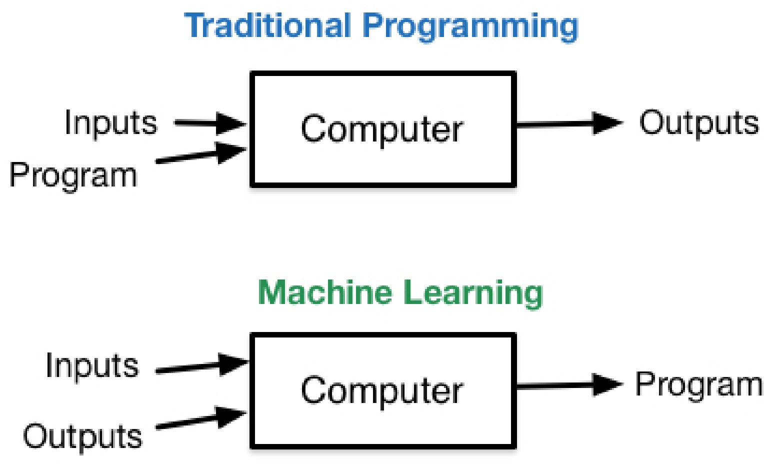


Figure 1.1: Classical Programming vs Machine Learning [1]

1.1.1 Types of Machine Learning Problems

In ML two types of dataset are given, one is known dataset and another in unknown dataset. The known dataset consists both the input and output, while in unknown dataset input's are given and we have to predict or approximate the output, for that purpose function is required like output = f(input). The function is also called model, which is obtain from using the known dataset. To optimize the model or to check the reliability on the model, data scientists divides the known dataset having input and output into two part one is training set and second is testing set. The input data set is presented in form of matrix, where each row is the example and each column is feature vector, same thing will be done on output dataset. Based on the handling of input and output, some categories of ML are decided. Supervised learning and unsupervised learning are two main machine learning approaches.

Unsupervised Learning

Unsupervised learning also known as descriptive in which unlabelled inputs are given where output is unknown. Here focus is establish to find relationship between among the input data itself. If the output obtain from the unlabelled input is finite then it is called clustering, as shown in Fig. the output is a circle geometry, where the input points are spared in a limited distances. On other hand, if output is infinite is called density estimation.

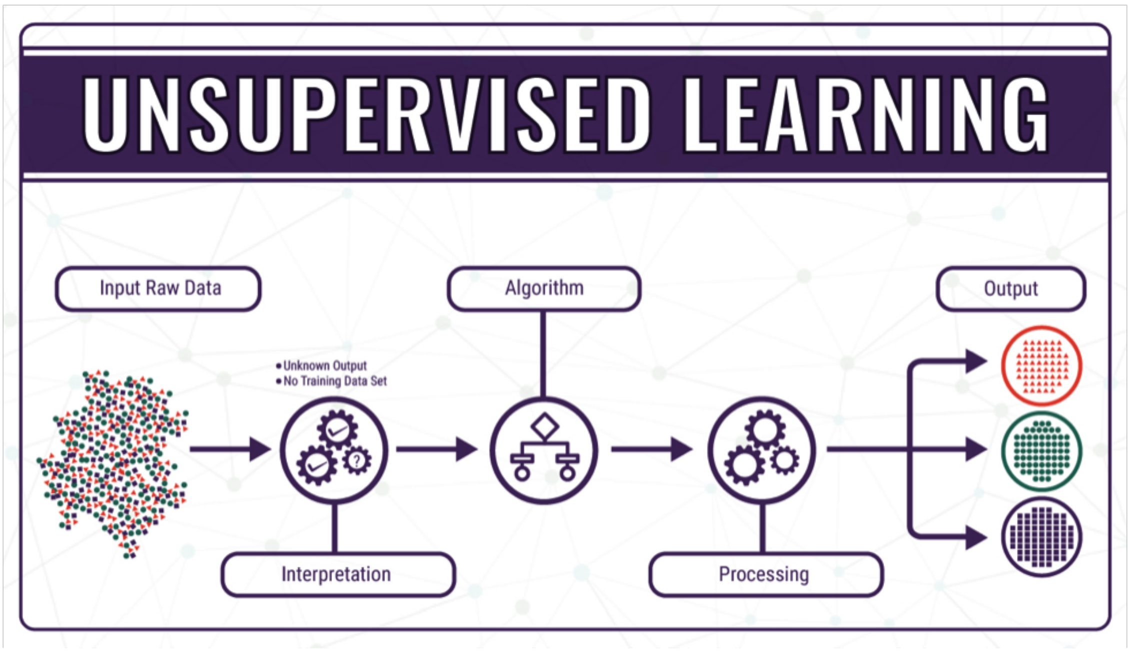


Figure 1.2: Unsupervised Learning [2]

Supervised Learning

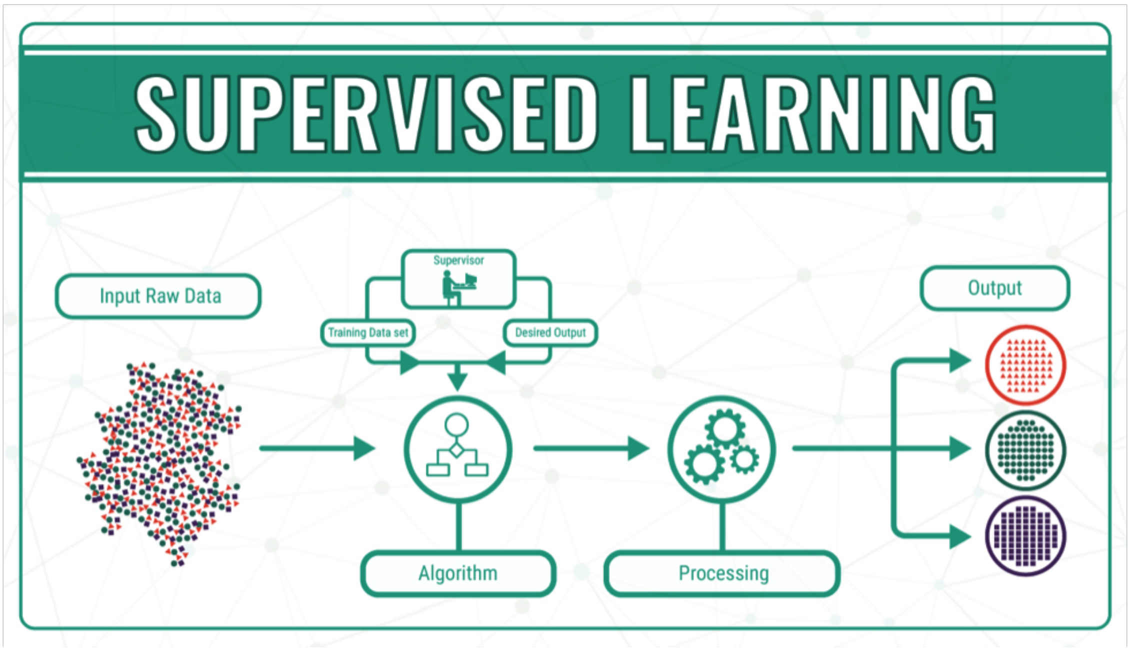


Figure 1.3: Supervised Learning [2]

The goal in supervised learning is to learn the function that leads inputs to outputs, from given a set of labelled data, known as the training set. Supervised learning itself divided in two part, one is regression and another is classification. In regression problems, the output are continuous real value. While in classification problems, multiple output are available where we predict the class in which particular input are settled.

There are other problems in machine learning such as semi-supervised learning, where labelled data is combined with unlabelled ones and reinforcement learning, where no output and input is given but we have to maximise the output and reduce the losses by some actions.

To achieve the best results, best model is required. The model is come up based on what type of data we fed into algorithm. So proper data set is the key requirement to get best results. The data given to us are in any form may be just number or full of string, continuous or discrete, images or binary numbers, may be some data points are missing and some where duplicate. So pre-processing is much important task which lead the whole prediction process.

1.1.2 Learning Algorithm

In the problems where inputs are given with labels, one is facing supervised learning task. Here the data set is written as $\{X^{(i)}\} \to \{y^{(i)}\}$, where 'i' is the index for a particular example. Here $X^{(i)} = X_1^{(i)}, X_2^{(i)}, ..., X_j^{(i)}$, where 'j' is number of feature in the data set. So complete input matrix is $X_{i \times j}$; where i = no. of example written in row of matrix and j = no. of feature written in column of matrix. Similarly $y^{(i)} = y_1^{(i)}, y_2^{(i)}, ..., y_r^{(i)}$, where 'r' is number of output for a particular one input row vector. In most of cases r = 1, so output matrix is a column vector having dimensionality 'i'. Let's consider r = 1, in regression cases, so $y^{(i)}$ is may be discrete value or may be continuous.

The expression to predict the label using regression approach is given by,

$$\widehat{y}^{(i)} = \theta^T X^{(i)} \tag{1.1}$$

Where, $\widehat{y}^{(i)}$ = Predicted label for given X;
$\theta_{j \times 1}$ = Weight vector

Now the task is obtain θ such that $\widehat{y}^{(i)} = y^{(i)}$. In order to obtain the weights θ, cost function is introduced into model, which is given by a sum of least squares error. There are many cost functions are available and in use based on the requirements. Here the cost function is expressed by,

$$J(\theta) = \sum_{i=1}^{n} L[\widehat{y}^{(i)}(X^{(i)}, y^{(i)})] = \frac{1}{2n}\sum_{i=1}^{n}(\widehat{y}^{(i)} - y^{(i)})^2 = \frac{1}{2n}\sum_{i=1}^{n}(X^{(i)}\theta - y^{(i)})^2 \tag{1.2}$$

By minimizing the function $J(\theta)$ with respect to weights, one finds the best set of θ. The gradient decent algorithm is most popular to optimize the value of θ, which is initialize randomly. The expression is written like,

$$\theta_j := \theta_j - \alpha \frac{\partial J(\theta)}{\partial \theta_j} \tag{1.3}$$

Where, α = learning rate

1.1.3 Overfitting Problem

When a "Machine Learning Model" yields a small training error but a large test error, then the model is overfitting. Over fitting means we allow to much flexibility in the model so that training error becomes zero while test error is very high.

To solve overfitting problem bias - variance trade off plays important role. Bias is inversely proportional to flexibility while variance is directly proportional to flexibility. So bias - variance trade off gives optimum flexibility such that training error and testing error lies in range. The graphical view is shown in figure

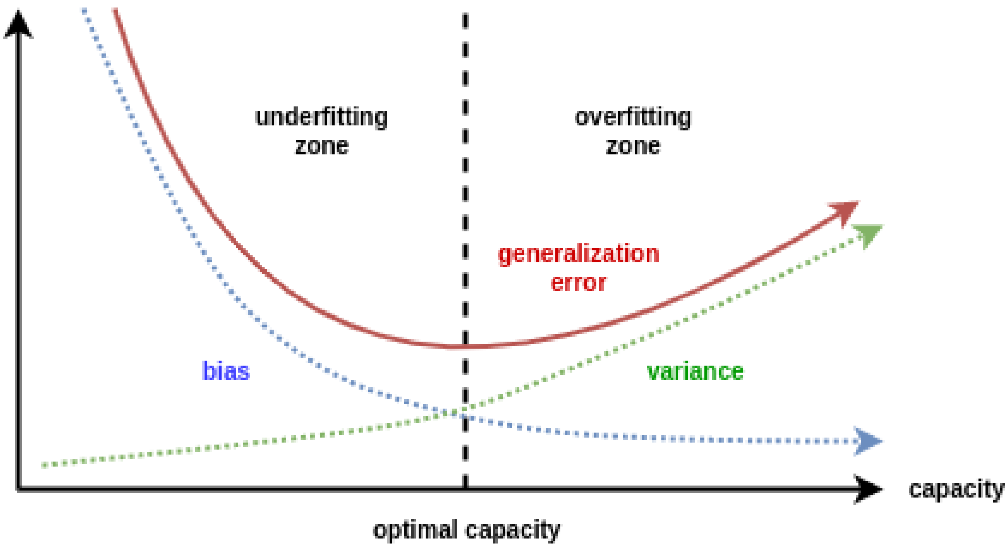


Figure 1.4: Bias - Variance Trade off [3]

Classification Problem

If the labels are discrete then the supervised learning algorithm is classification. A very popular example of classification problem is logistic regression, in which predictions are made by mapping linear regression into [0,1] interval. Let's suppose that the we have to find whether data belongs to a particular class or not. let $X^{(i)}$ belongs to a particular class $y^{(i)} = 1$) or $y^{(i)} = 0$). The expression for such two class or binary prediction is,

$$\widehat{y}^{(i)} = \sigma(\theta^T X^{(i)}) = \frac{1}{1 + exp(-\theta^T X^{(i)})} \tag{1.4}$$

Where, $\widehat{y}^{(i)}$ = Predicted label for given X;

$\theta_{j\times 1}$ = Weight vector

σ = Sigmoid function, it is the activation function.

Activation function converts the output in the range of 0 to 1. There are certain activation function, which are popular in machine learning; those are sigmoid function, step function, tanh function, linear function, Rectified Linear Unit Function (ReLU), Non-linear Cube Activation Function etc.

After applying activation function on $\theta^T X^{(i)}$ the predicted value $\widehat{y}^{(i)} \in [0,1]$. Here one manually decide a value α such that if $\widehat{y}^{(i)} \geqslant \alpha$ then input is belongs to class $y^{(i)} = 1$ otherwise belongs to $y^{(i)} = 0$. Here, the cost function is obtained from the negative log-likelihood. So that parameters θ requires the minimization of the cost function, given by

$$J(\theta) = \sum_{i=1}^{n} [y^{(i)} log(\widehat{y}^{(i)}) + (1 - y^{(i)}) log(1 - \widehat{y}^{(i)})] \tag{1.5}$$

Here, the cost function is looks different compare to regression, but the task is same to minimize $J(\theta)$ using gradient decent approach or similar type approach used in getting minima of convex function.

1.2 Structure Property Correlation

In a perfect crystal, stress required to cause deformation is much higher as compared to when dislocations are present. Understanding the impact of dislocation on structure property is a key to understand deformation. Dislocation motion is analogous to the motion of a caterpillar. It moves only a small part of the body instead of completely moving the whole plane of the body.[15]

Dislocations can only be removed from the material when dislocations of opposite sign come closer to each other. Also,we know that Plastic deformation is an irreversible, path dependent phenomenon hence dislocation density cannot be determined by equilibrium thermodynamics. Earlier, scientist have used discrete dislocation simulations to express the mean free path in terms of the critical shear stress, elastic moduli, density of junctions, and the number of active slip planes inside a crystal.

While experimenting on copper single crystal,J. Diehl proposed that material harden in a number of stages. In stage 1 also known as "easy glide" does not depends on multiple slip in the system but only on the orientation of crystal. In stage 2 which is the most steepest meaning has higher strain rate is material insensitive, whereas stage 3 strongly depends on material, temperature and rate of deformation.[16]

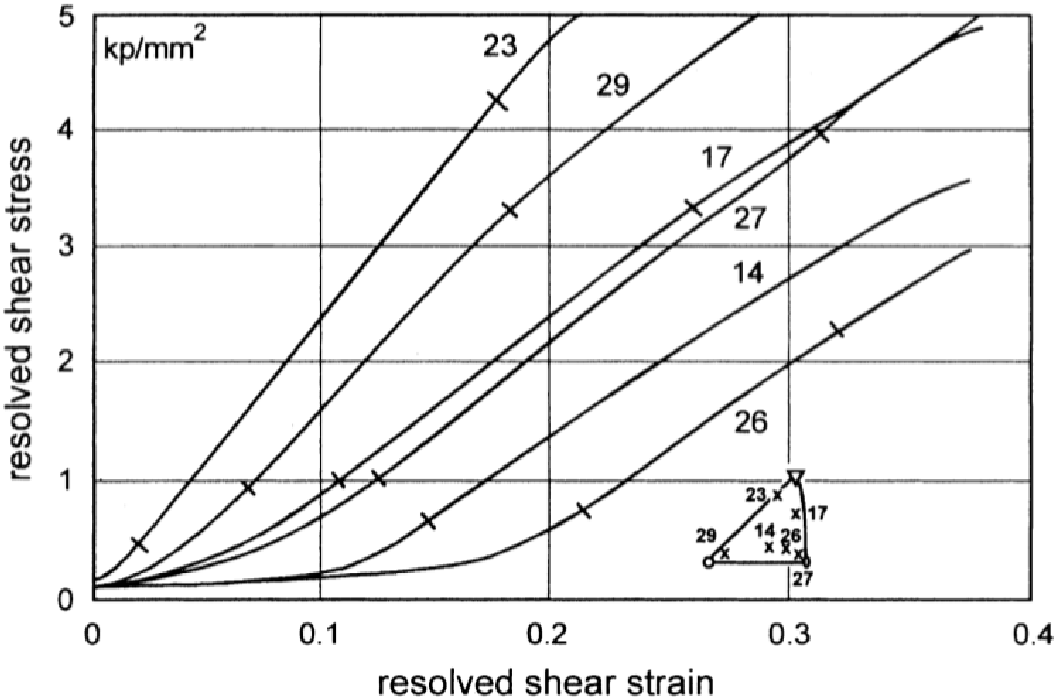


Figure 1.5: Resolved sheer stress vs Resolved sheer strain in Cu crystal showing hardening stages [4]

Chapter 2

Literature Review

2.1 Dislocations - Line Defect in crystals

Line defect or 1- dimension defect in crystal are called Dislocations. These are called 1-D defect or line defect because these occur at the areas where a line of atoms are displaced or out of position in their crystal structure.Upon stress application they move and generate. This motion of dislocation allows plastic deformation to occur.

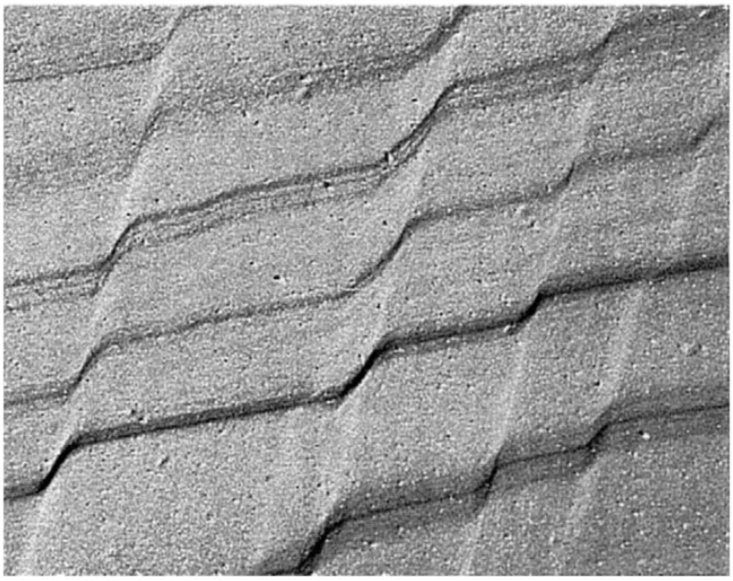


Figure 2.1: Shadowgraph of Slip bands in Aluminium crystal at Room Temperature[17]

After the development of transmission electron microscope, dislocation theory was confirmed which says ductility and strength of metals are controlled by dislocations and their plastic properties greatly changes during forming operations. Dislocations are mainly of three type namely, edge dislocation, screw dislocation and mixed dislocation. Both edge and screw dislocations are extreme forms of the possible dislocation structures that can occur inside a crystal. Most dislocations occur in crystal are hybrid in nature.[17]

2.1.1 Edge Dislocation

In edge dislocation one extra half plane of atoms exist above or below the slip plane. The locus of defective points produced in the lattice is a line which runs along the bottom or top of extra half plane on the slip plane. In the vicinity of dislocation line atomic bonds are distorted.

As shown in the Fig.2.2, when stress is applied, dislocation moves a small amount at a time along the slip plane. This movement continues along the slip plane till it comes out of the crystal causing the top half of the crystal to move with respect to bottom half. Note that not all bonds are broken at a time, instead fraction of bond is broken. Hence this require lower energy meaning lower forces in comparison to breaking all the bonds simultaneously.

Here, Burger's vector is defined as the smallest atomic unit displacement of the lattice distortion resulting from a dislocation in a crystal lattice. Burger's vector lies along the slip plane and direction is generally along the closed crystallographic plane.

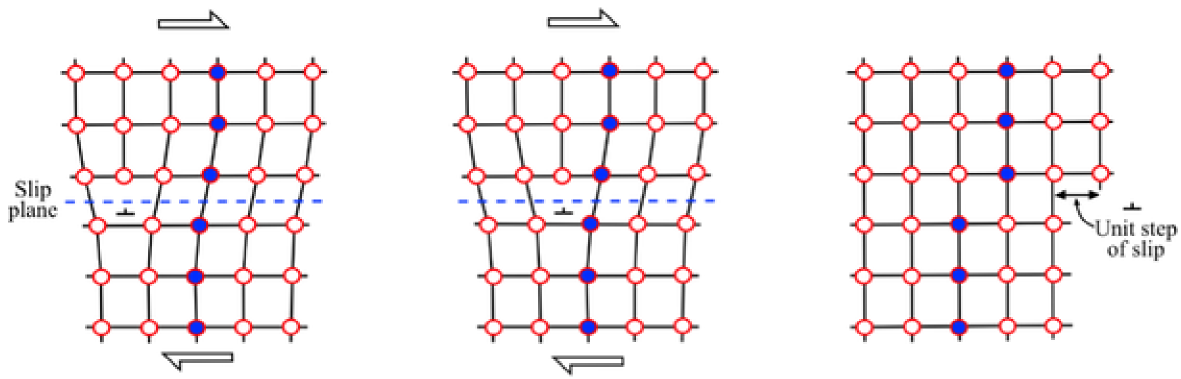


Figure 2.2: Edge dislocation Movement[17]

2.1.2 Screw Dislocation

When the movement of dislocation line is perpendicular to the stress direction then the type of dislocation is known as screw dislocation. Motion of screw dislocation is also due to presence of shear stress in the crystal. In this when stress is applied motion of a certain fraction of atoms occur perpendicular to the slip plane hence require lesser energy to cause slip.

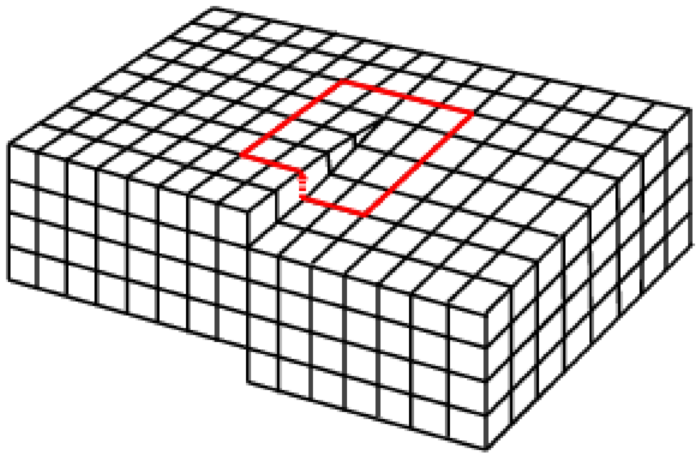


Figure 2.3: Screw Dislocation[17]

Generally, dislocations move along the closed packed planes because as the spacing between the planes increases, energy required to move dislocation also increases. In FCC there are 12 slip planes, hence dislocations move easily in FCC and we can say that most of the crystals that exhibit FCC are ductile in nature. Similarly BCC has 48 slip systems and show ductile behaviour. To improve the strength of materials, we make them alloys by introducing solid solute particles that oppose the dislocation movement. Similarly grain boundaries also does not allow dislocations to move and pin the dislocations. This means finer grain structure has higher hardness compared to coarser grain structure.

In ionic materials,slip is difficult because ion must move past an area that has repulsive charge and a repulsive force act which is higher than applied stress. So, this makes ionic materials brittle. Similarly, low density packing is the reason of covalent material being brittle compared with metals.

2.1.3 Dislocation Dynamics

Discrete dislocation dynamics is the most suitable computer simulation technique to generate good result for materials at nano and micron scale. Material is treated as continuum that has dislocations along with grain boundary as in case of polycrystalline materials. Some laws are defined to nucleate dislocation, move dislocation and even destroy dislocation in a crystal and this Dislocation dynamics is govern by those laws only.

Computationally demanding 3-D dislocation dynamics models has strong physics base and show similarity with real materials but these are applicable to single crystal at low strain, low volume of material and low dislocation densities not to polycrystalline materials(Does not show much similarity). Similar to real materials, egde dislocation can climb as well as glide. In dislocation climb, dislocation are stopped along slip plane and they start moving in perpendicular direction along the next closed packed plane. Hence, generally softening in material occur due to climb mechanism.[18]

A material is modelled as an elastic solid that contain dislocations and then simulations are carried out. At any time we have information about displacement and stress field because these are considered (assumed) to be in equilibrium. As the simulation progress, it automatically update positions of dislocation, stress, displacement field inside material and strain.

2.2 K- fold Cross Validation

Variety of machine learning models and architecture are available for unsupervised learning. Among those models, K-fold cross validation method split the complete dataset into k parts/sections in which (k-1) sets are used for training and 1 set is used for testing alternatively till each section becomes a test section hence the name K-fold cross validation. Following is the approach for k=5 i.e. 5 fold cross validation:

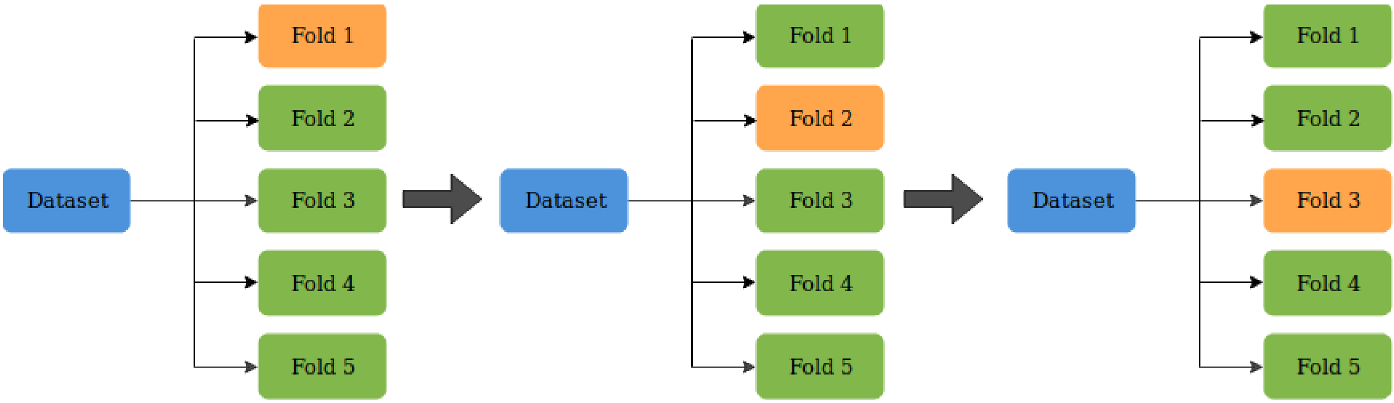


Figure 2.4: 5- Fold cross validation-1 [5]

In 5 fold problem, dataset is split into 5 folds and in first iteration, first fold is treated as test set while rest all are training set. In second iteration, second set is treated as test set while rest serves as training. In this way this process is repeated till each set becomes a test set.

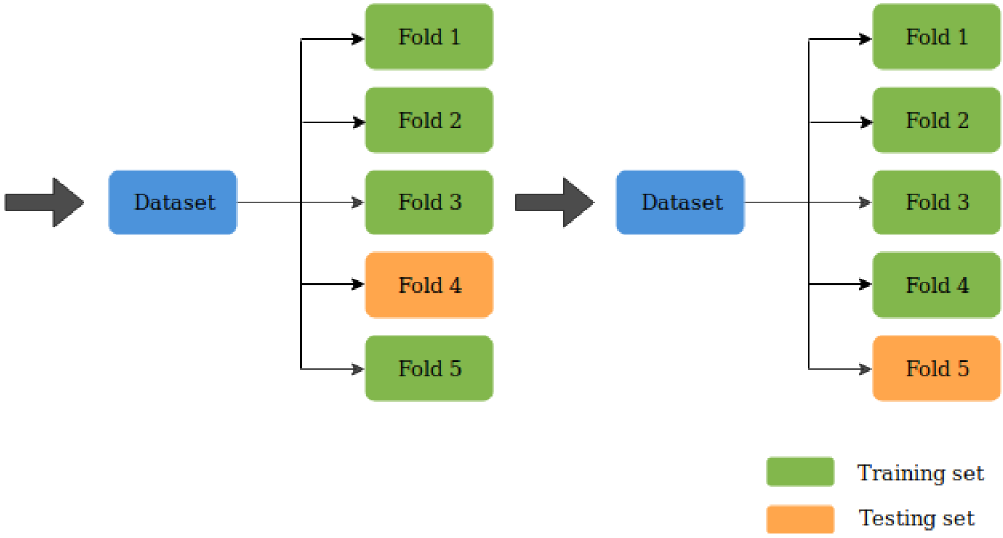


Figure 2.5: 5- Fold cross validation-2[5]

K-fold cross validation is generally used when training data is less and to ensure that each data is used in training. This makes model robust, but this makes computation cost to increase and overall computation time increases by k times.

2.3 Regression

A simple linear regression try to model the relationship between two variables one is dependent and other is independent. This is done by fitting a linear equation to observed data(Independent variable) and a line is generated which can be extrapolated or interpolated depending on the requirement of the user. Similarly, multiple linear regression is a technique that uses several explanatory variables to predict the outcome of a dependent variable, explanatory variables are also called independent variables and dependent variable is also called response variable.

Before applying linear regression we plot scatter plot to see if there is any significant relationship curve between the two variables. If we see scatter plot to show no relationship between the two variables then fitting a linear regression model is not a good choice and hence in future predictions will not be that accurate and may deviate from actual results. Before fitting a line, we can also calculate correlation coefficient as an indicator of strength of association. A simple linear regression line look a below:

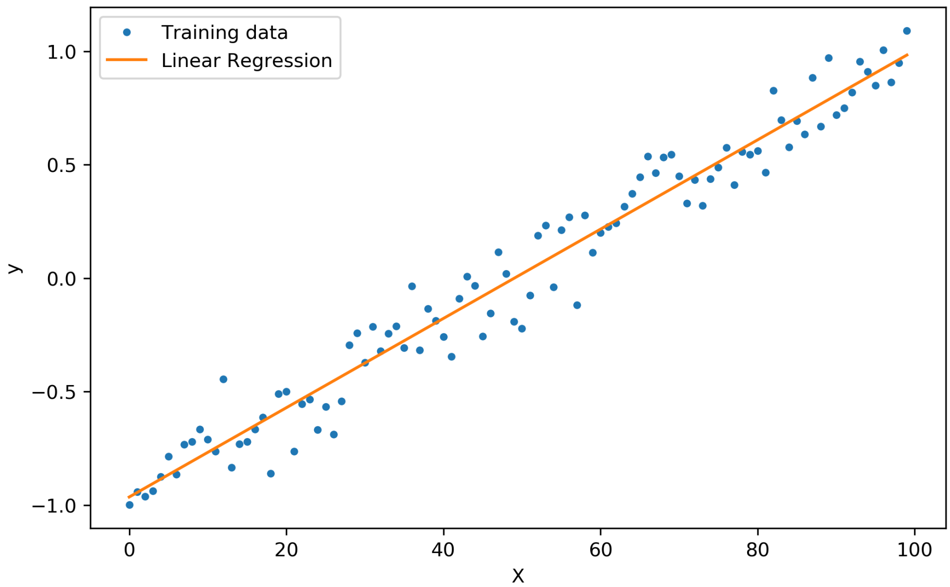


Figure 2.6: Linear regression line[19]

A linear regression line has an equation of the form

$$Y = a + bX \tag{2.1}$$

where, X is the explanatory variable and Y is the dependent variable. a and b are constants.

A multiple linear regression line has equation of the form

$$Y = a + b_1X_1 + b_2X_2 + b_3X_3 \;\; .... + b_nX_n \tag{2.2}$$

where, X_i is i^{th} independent variable and $b_i's$ are constants.

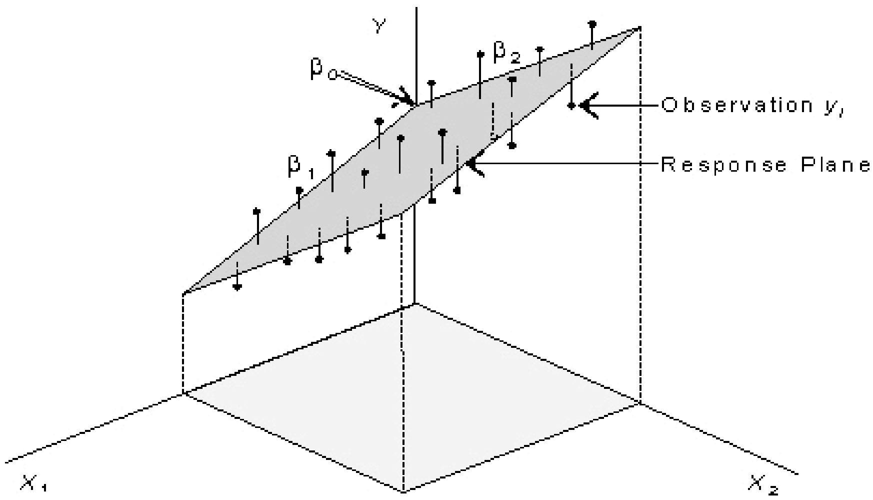


Figure 2.7: Multiple Linear regression plane [6]

To examine the validity of the line/plane, residuals(deviation from actual line) are calculated. Plotting the residual on Y-axis against independent variable will reveal any possible non-linear relationship and alert to investigate other possibilities.

2.4 Neural Network

Neural networks are just like human brain (Neurons inside human body) and in mathematical terms are multi-layer networks of neurons (nodes) that we use to classify things, make predictions, etc. We have one input layer and one output layer and in between these two layers are the Hidden layers. Depending upon the requirement of problem number of hidden layers can be set.

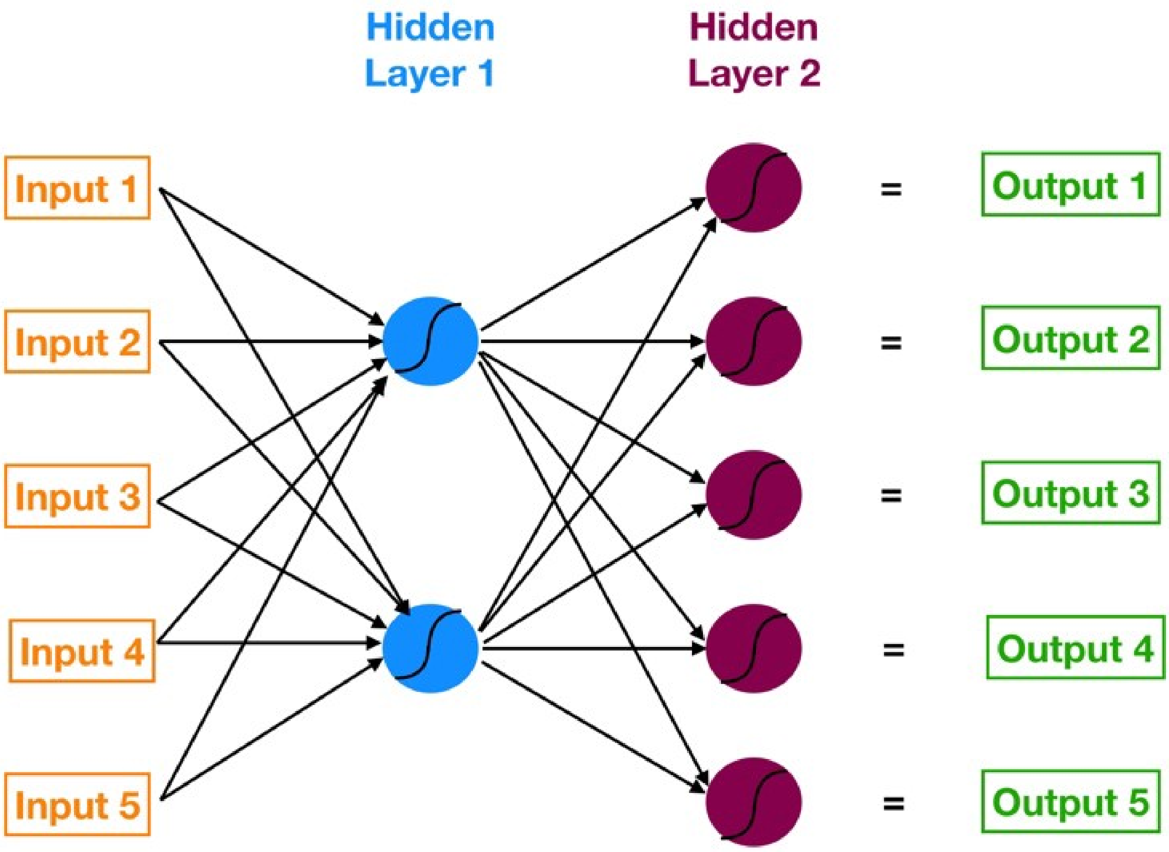


Figure 2.8: A Simple 3 Layer Neural network [7]

At each neuron input are weighted and added with bias term, this neuron when acted by activation function will further result into an input to other neuron in next layer. We denote X as input matrix, W as weight matrix and B as bias matrix. We calculate Z as:

$$[Z] = [W]^T * [X] + [B] \tag{2.3}$$

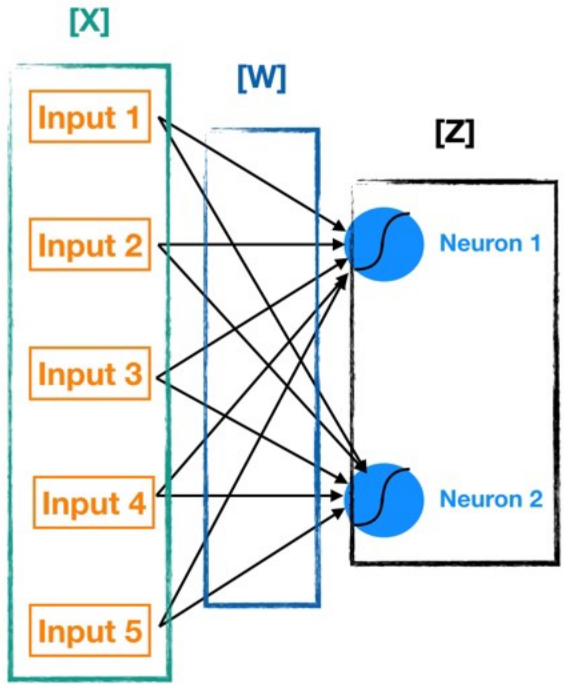


Figure 2.9: various matrix in neural network[7]

By repeatedly calculating $[Z]$ for each neuron and applying the activation function to it for each successive layer, we can move from input to output.

To train a neural network, we need to apply an optimization algorithm such as gradient descent and then define a cost function such as MSE (mean square error) to calculate accuracy of our model. After a successful number of iterations we get weights and biases for which MSE is very low and that define our model.

2.5 Convolution Neural Network

Convolutional Neural Networks (ConvNet) are very similar to ordinary Neural Networks as they are made up of neurons that are interconnected to each other and learnable weights and biases are updated as the training of model occur. At Each neuron, we receives some inputs, performs a dot product of weights and input,result are followed by passing through an activation function.

From other neural network architecture Convolution Neural Network architectures explicitly assume input to be images, which allows programmers to put certain properties into the architecture and use it for Machine learning. These properties make forward function more efficient to implement and reduce the amount of parameters in the network and in turn training time. Weights are updated while training through input images multiple times during both forward and backward propagation.

2.5.1 ConvNet Architecture

The layers of a Convolution neural Network have neurons arranged in 3 dimensions: depth, height, width. As the convolution neural network progresses, we reduce the full image into smaller arrays and at the end we have a single vector of class scores(their probability). Scores are single vector arranged along the depth dimension. A sample convolution neural architecture:

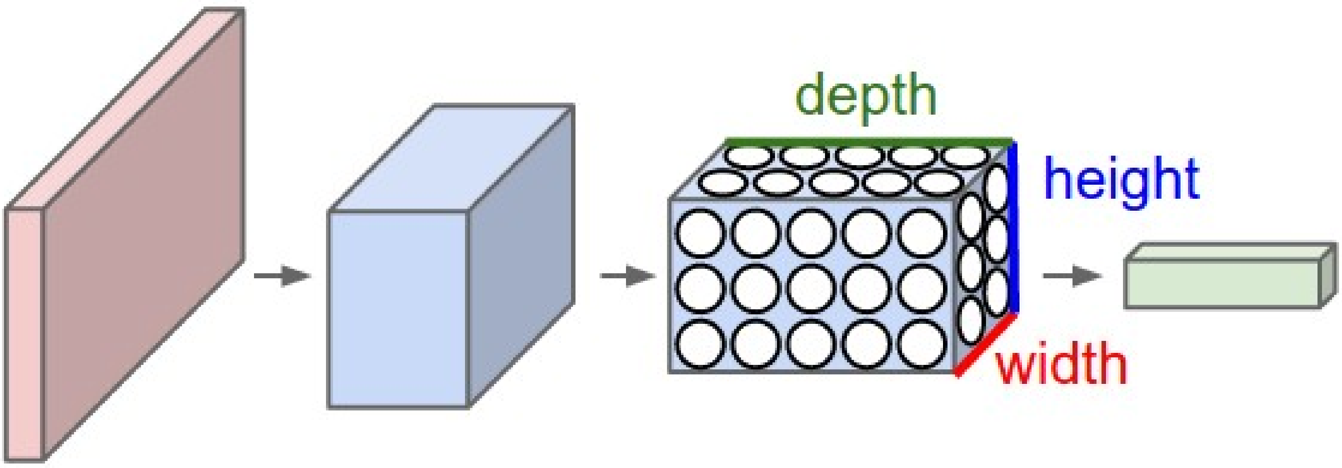


Figure 2.10: Arrangement of Neurons in 3 dimension(width,height and depth)[7]

Layer 1: Convolutional layer

A simple input image will be of LxHx3 size where L is length, H is height and 3 shows 3 layers of colour channels viz. Red,Blue and Green colour. This input will contain raw pixel values of image. When it passes through convolution layer, neurons that are connected in the vicinity of the studying neuron also take part. we compute dot product of small region with the weight matrix.

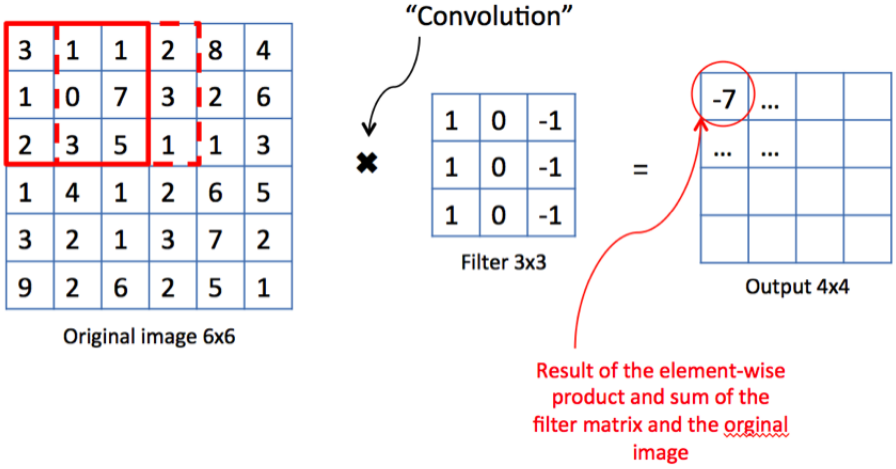


Figure 2.11: Convolution layer [8]

Layer 2: Max pooling layer

Max polling layer reduces parameters by progressively reducing the spatial size of representation hence computation reduces significantly inside the network. At each feature map pooling layer operates independently. In max pooling, depending upon the size of filter maximum value among the matrix is selected as shown in figure.

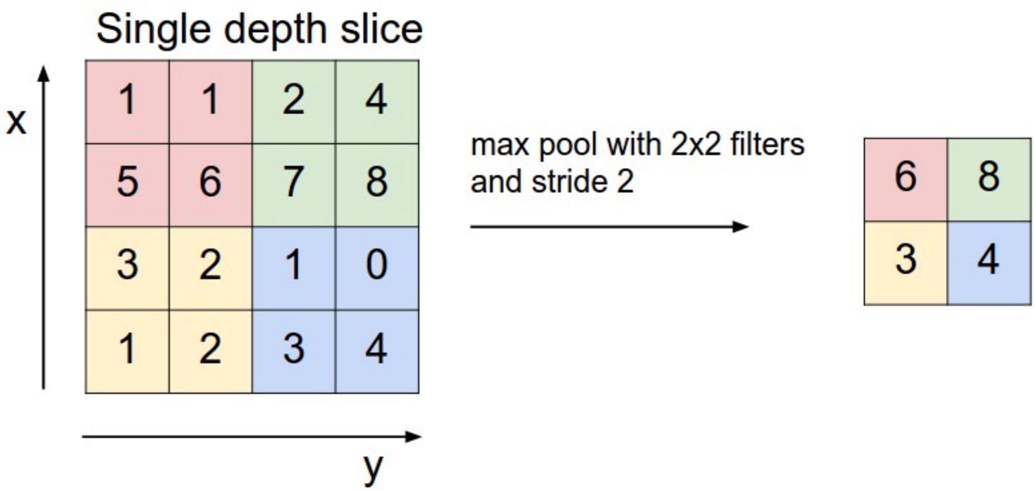


Figure 2.12: Max pooling layer [9]

Layer 3: Flattening Layer

Flattening layer lies in between convolution layer and fully connected layer. It transform the 2 dimensional matrix into a single vector that feeds to fully connected layer hence to a dense fully connected classifier.

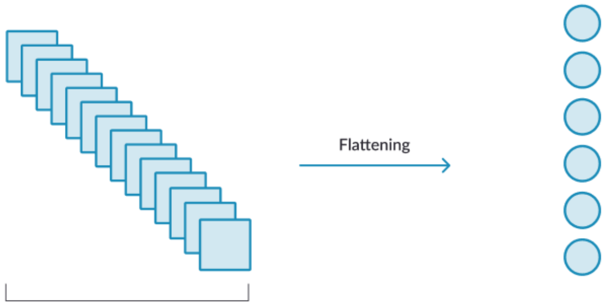


Figure 2.13: Flattening layer [10]

Layer 4: Fully connected Layer

Output of fattening layer is considered as input to the fully connected layer and depending on number of hidden layers each neuron from i^{th} layer is connected to the $i+1^{th}$ layer to form a dense matrix

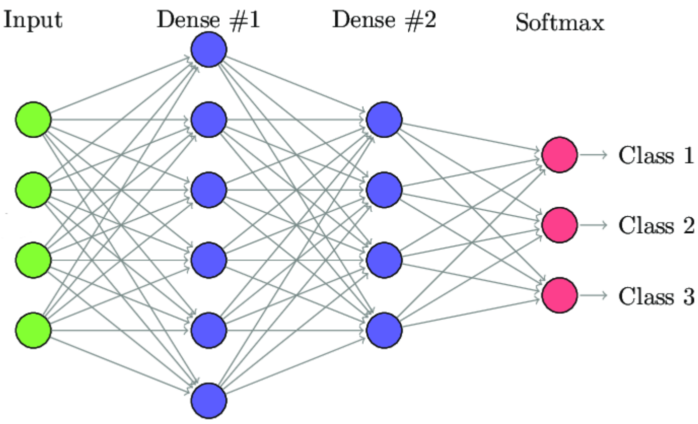


Figure 2.14: Fully Connected layer acting as input to neural network for classification [11]

Layer 5: Classification layer

To classify images into different classes, Convolution neural Networks transform the original image layer by layer from the original pixel values to the final probability of a class. In this layer activation function like softmax and ReLU are used to compute the class scores. The parameters in the all previous layers will be updated by calculating loss function using gradient descent or ADAM so that the class probability that the Convolution Neural Network computes are consistent with the labels in the training set for each image. Generally we use softmax for classifying classes count more than 2 and for binary we can use sigmoid/ReLu.

Chapter 3

Methodology

3.1 Data Generation

In Dislocation dynamics, a finite number of frank reed sources were given inside a unit cell, uni-axial loading was applied and evolution of dislocation was observed. Data was collected for different type of dislocations in a unit cell.

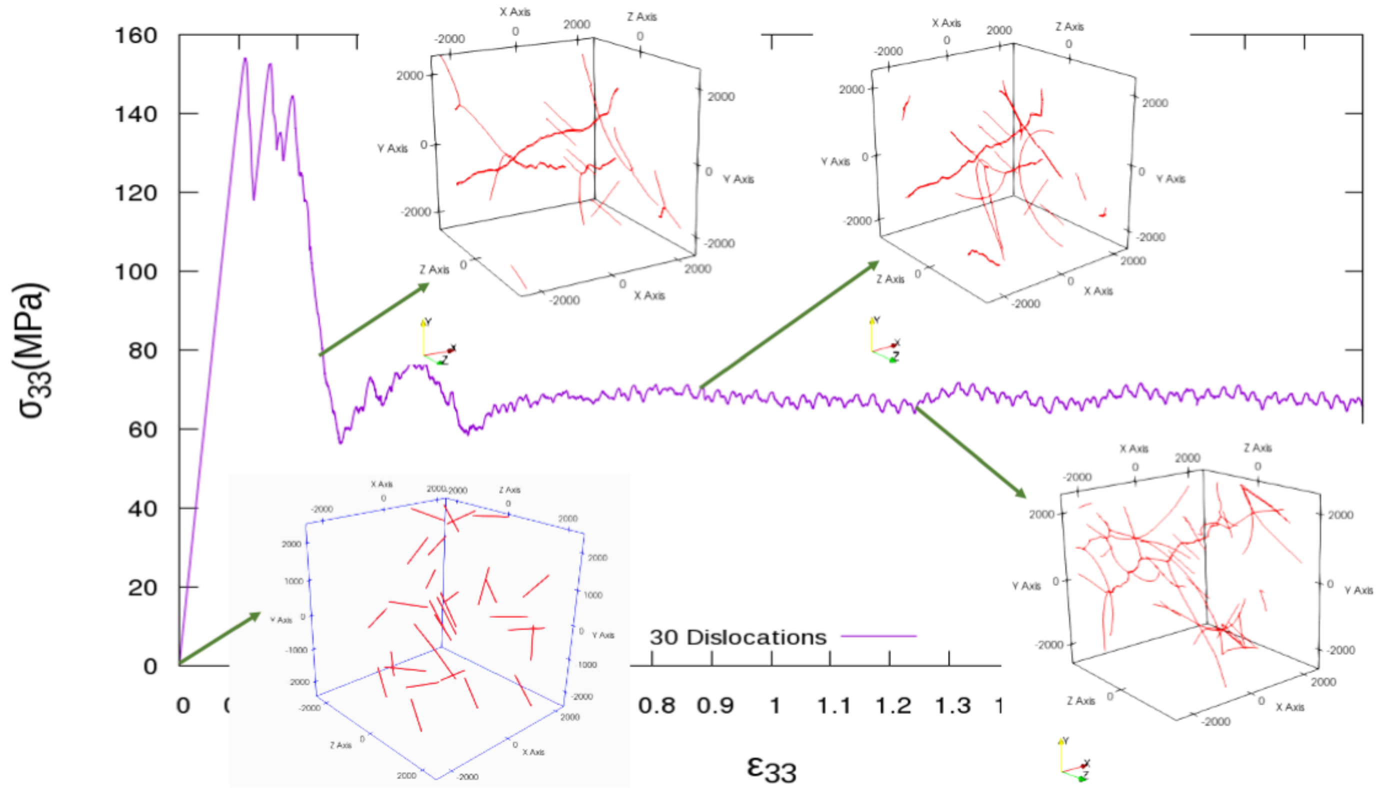


Figure 3.1: Dislocation evolution in a single simulation

We collected data of different dislocations by using simulation and letting them run for 2% dislocation and then the data collected is processed to an usable form for machine learning application. Starting number of dislocation in a unit cell are 28,30, 32, 34, 36, 40, 42, 44, 60, 80, 90, 96 and 128. As the stress is applied, dislocation started generating from frank reed sources.

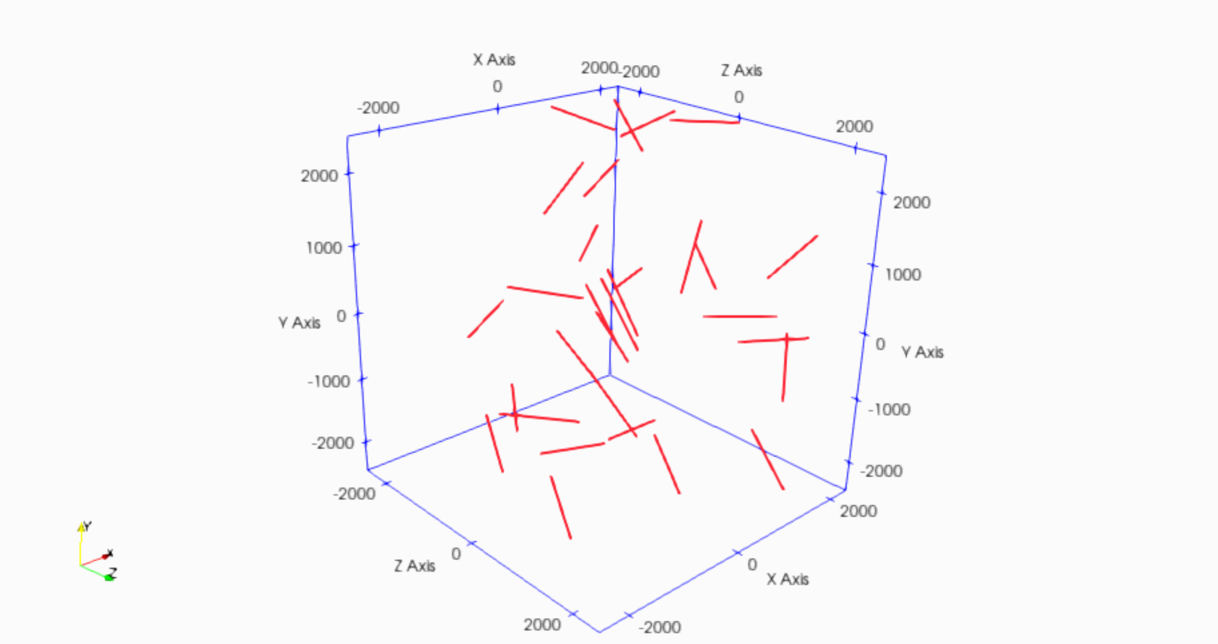


Figure 3.2: Initially undeformed Dislocations in a 3D unit cell

Data was collected has the following information about the structure of unit cell:

- Size of the unit cell (Pre-determined)
- X,Y,Z co-ordinates of the nodes of dislocation line.
- Burgers vector of each dislocation line.
- Angle between dislocations line
- Slip plane angle
- Segment length on each slip plane
- Dislocation density on each slip plane

We used this data to generate fingerprint for the Machine Learning model such that each structure property should be there at any point of strain so we devised a syntax to generate 2-Dimensional images from 3-Dimensional structure.

For machine learning problem, we used grey scale images and whole matrix is normalised between 0-255 value where 255 is pure white and 0 is dark black color on gray scale and intensity varies linearly from 0 to 255.

3.2 Data Analysis

We analysed effect of deformation on structure properties like segment length on a particular slip plane, Dislocation density on a slip plane and stress values. We did this study for multiple simulations but here showing only simulations that had initial configuration of 28 Dislocations and 30 dislocations.

3.2.1 Stress - Strain Analysis

As the stress is applied to material it start deforming, material behave elastically upto yield point and then after yield point it comes into the region of flow stress where it deform plastically. These curves reveal many of the properties of a material such as the Young's modulus, the yield strength and the ultimate tensile strength.

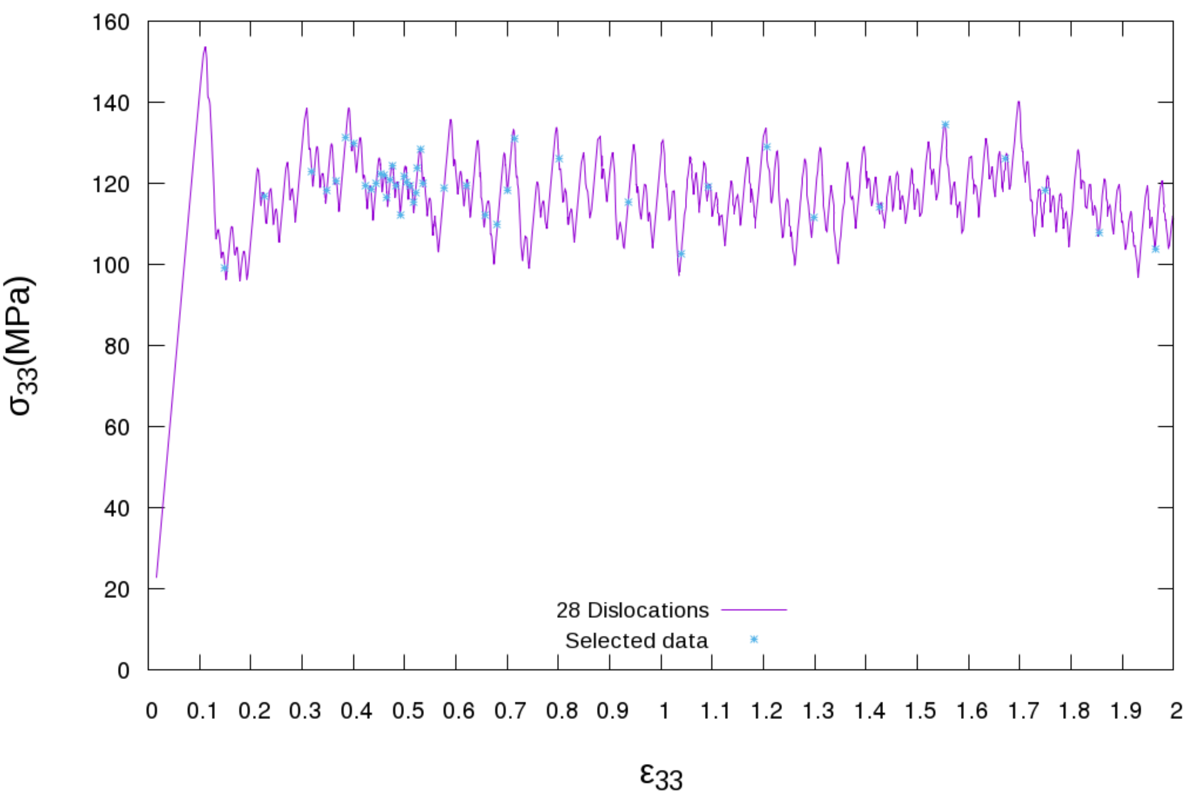


Figure 3.3: Stress strain curve for 28 Initial dislocation line

Both curve show clearly different stages, first stage is the linear elastic region. In elastic region material obeys Hooke's law and the slope of curve gives Elastic modulus of material. It can be easily concluded that material with different initial structure shows different Elastic modulus hence once can say for same material, material properties depends on internal structure.
The end of first stage is the initiation point of plastic deformation. The second stage is the strain hardening region. This region starts as the strain goes beyond the yielding point.

We took FCC(Face Centered Cubic) material(Copper) for simulations and all results are shown for that only.

3.2.2 Segment length Analysis

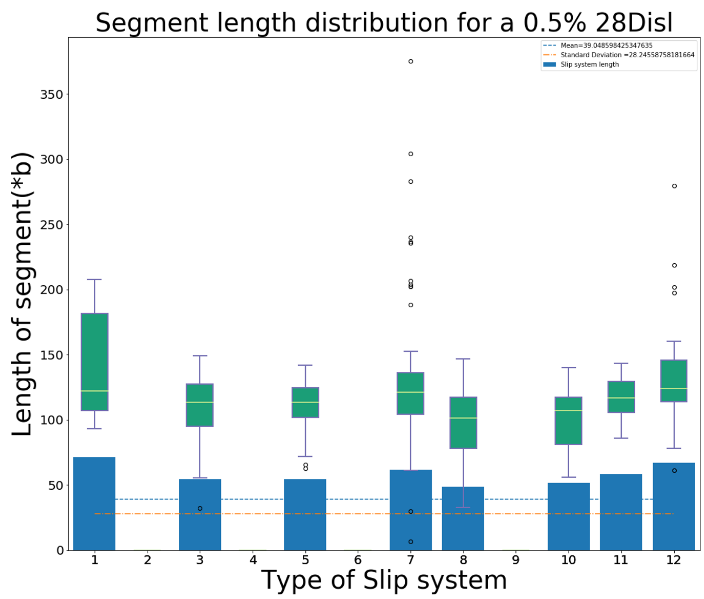


Figure 3.4: Segment length distribution at 0.5% strain on a slip plane for 28 initial dislocation line

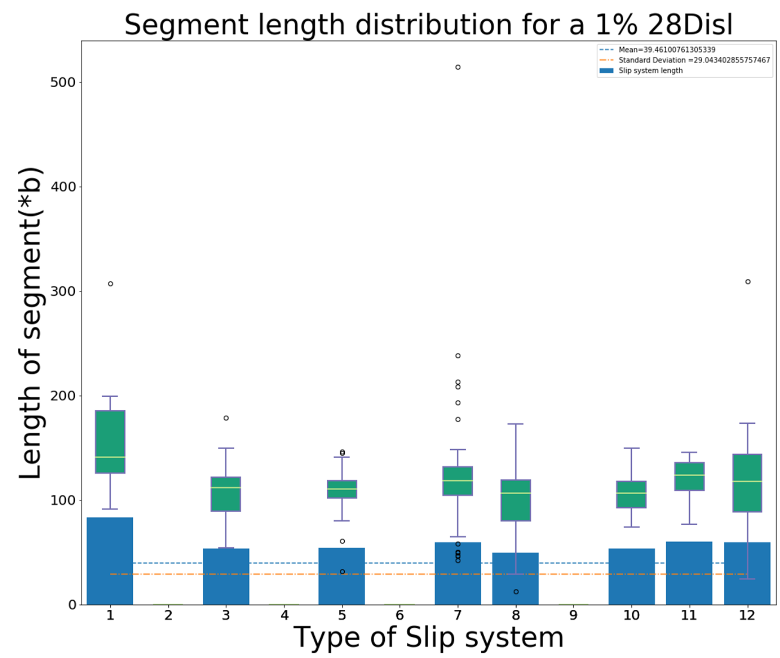


Figure 3.5: Segment length distribution at 1% strain on a slip plane for 28 initial dislocation line

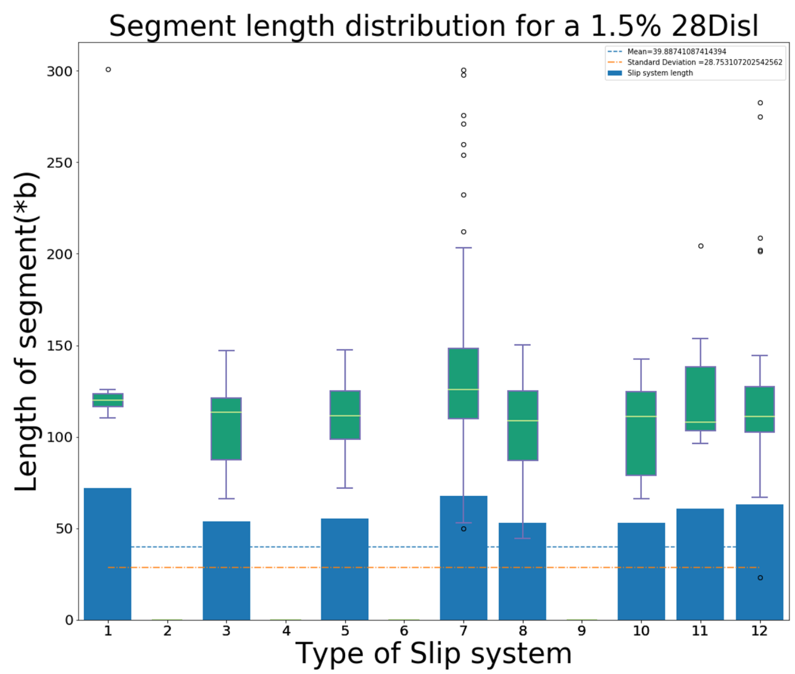


Figure 3.6: Segment length distribution at 1.5% strain on a slip plane for 28 initial dislocation line

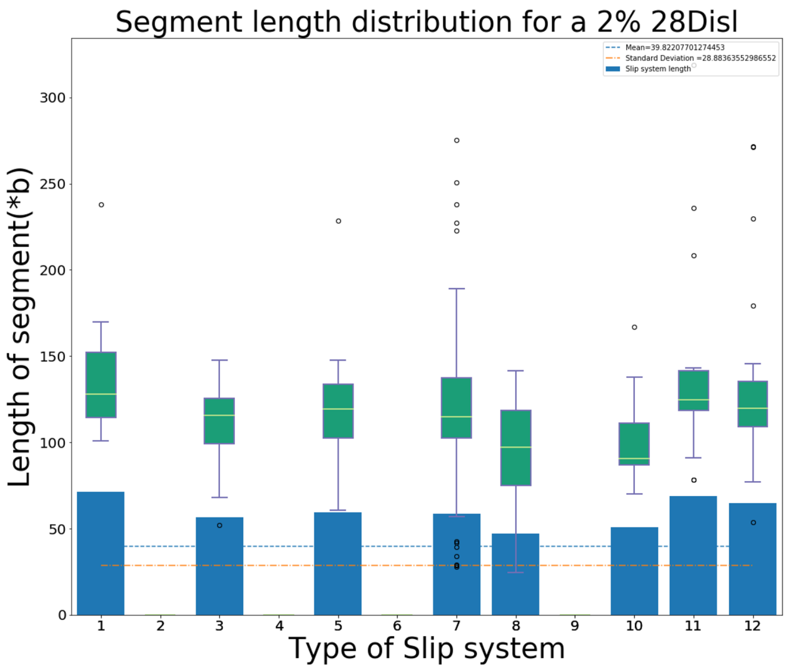


Figure 3.7: Segment length distribution at 2% strain on a slip plane for 28 initial dislocation line

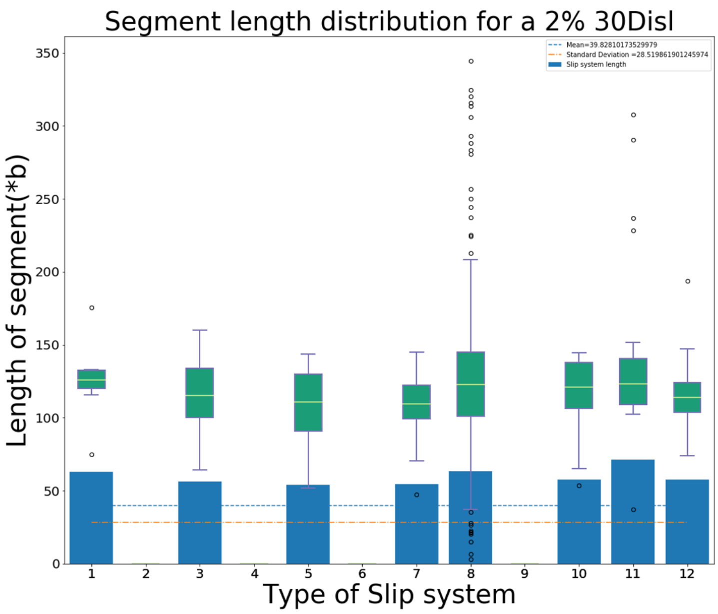


Figure 3.8: Segment length distribution at 0.5% strain on a slip plane for 28 initial dislocation line

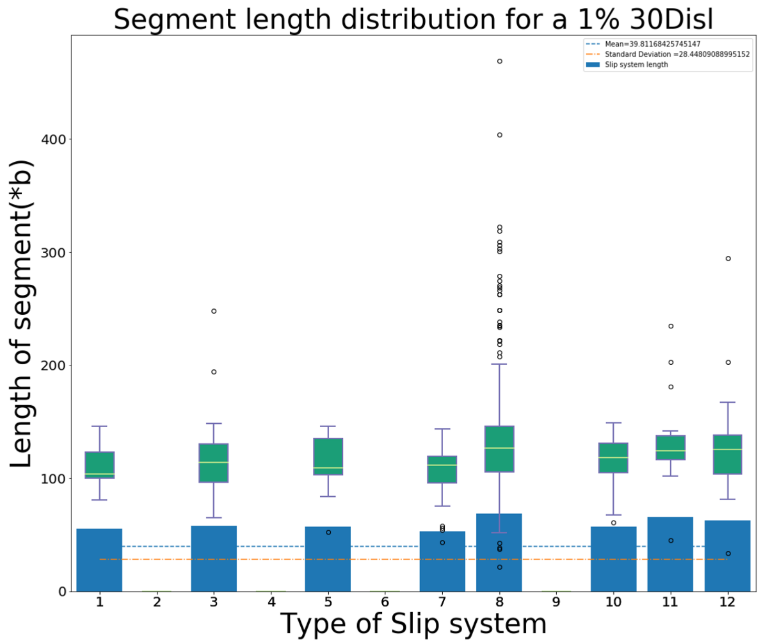


Figure 3.9: Segment length distribution at 1% strain on a slip plane for 28 initial dislocation line

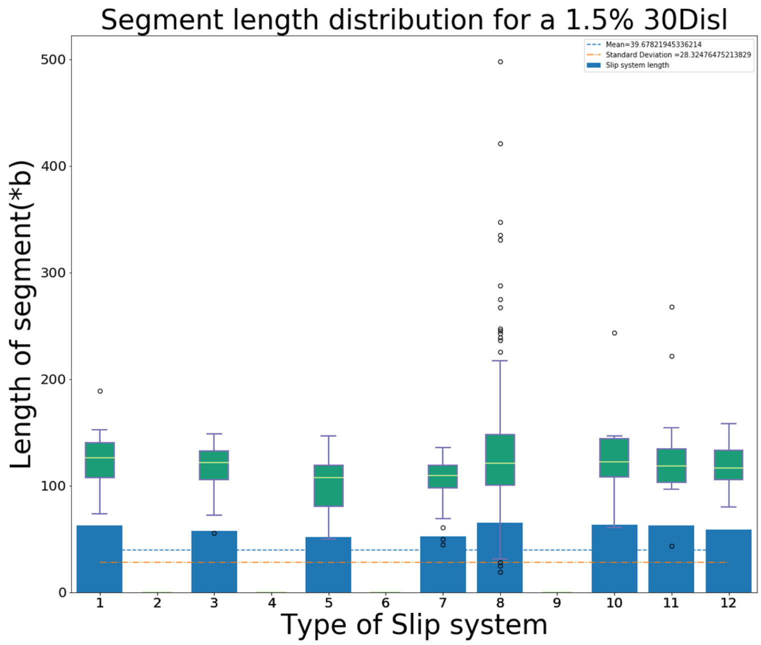


Figure 3.10: Segment length distribution at 1.5% strain on a slip plane for 28 initial dislocation line

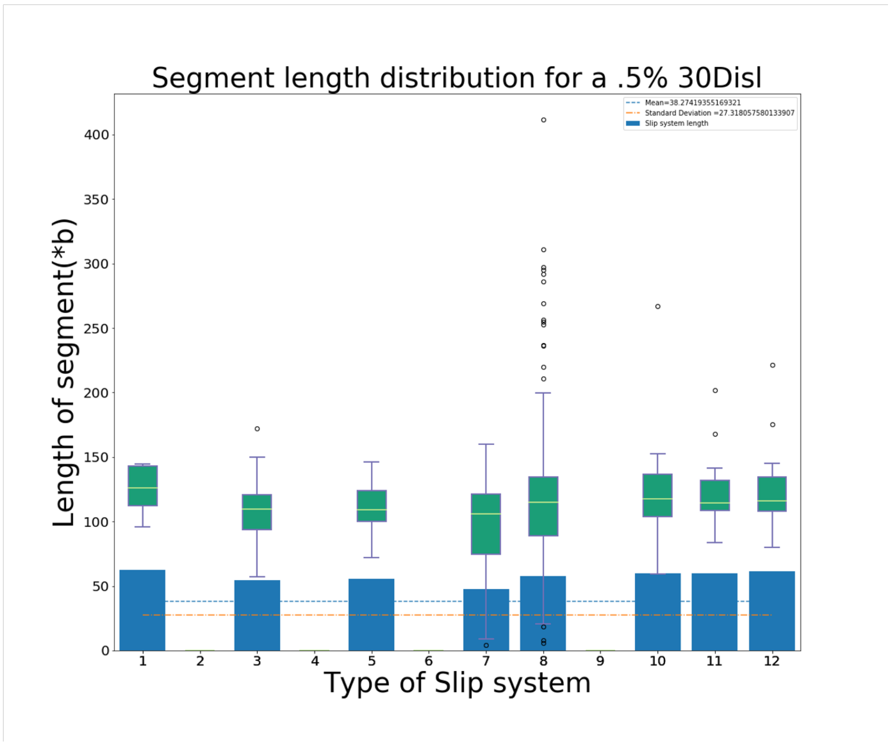


Figure 3.11: Segment length distribution at 2% strain on a slip plane for 30 initial dislocation line

Deformation of a crystal occur when the resolved shear stress exceeds a critical value given by schmid law and said to have slip occurred on a slip plane.

3.2.3 Resolved shear stress on different slip system

There are twelve slip systems in an FCC crystal and from the above analysis of segment length we can say that eight slip systems are active and 4 are inactive. Moreover active and non-active slip systems depends on various other external factors such as strain rate, temperature and various other parameters. We have defined 12 slip systems based on their combination of slip plane and Burger's vecor direction as follows:

FCC Slip System	Plane	BV	PLANE	BV
1	3	1	(-1 -1 1)	[-1 1 0]
2	4	1	(1 1 1)	[-1 1 0]
3	2	2	(-1 1 1)	[0 1 -1]
4	4	2	(1 1 1)	[0 1 -1]
5	2	3	(-1 1 1)	[-1 0 -1]
6	3	3	(-1 -1 1)	[-1 0 -1]
7	1	4	(1 -1 1)	[1 0 -1]
8	4	4	(1 1 1)	[1 0 -1]
9	1	5	(1 -1 1)	[0 -1 -1]
10	3	5	(-1 -1 1)	[0 -1 -1]
11	1	6	(1 -1 1)	[-1 -1 0]
12	2	6	(-1 1 1)	[-1 -1 0]

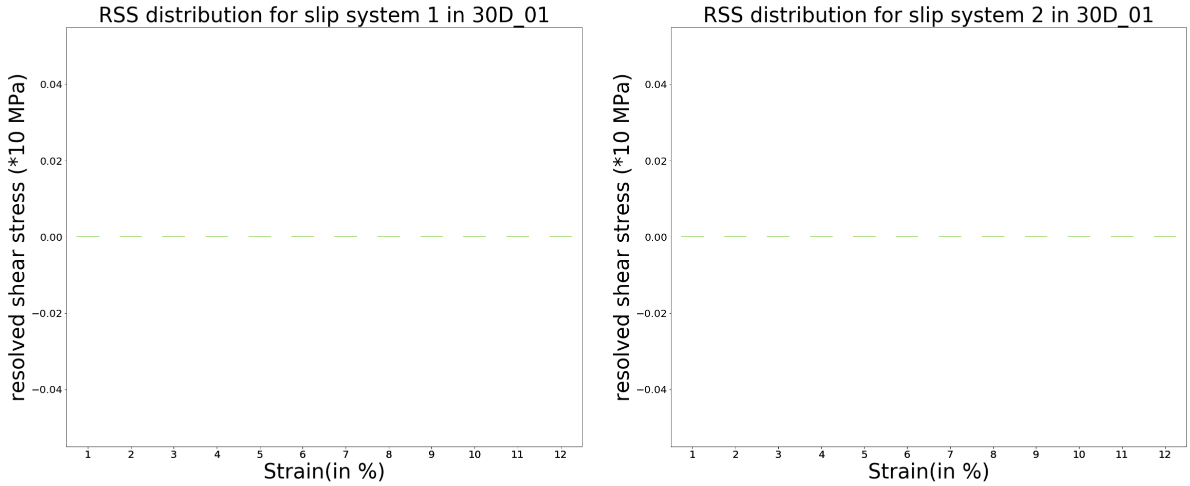


Figure 3.12: Resolved shear stress on slip system 1 and 2 with strain

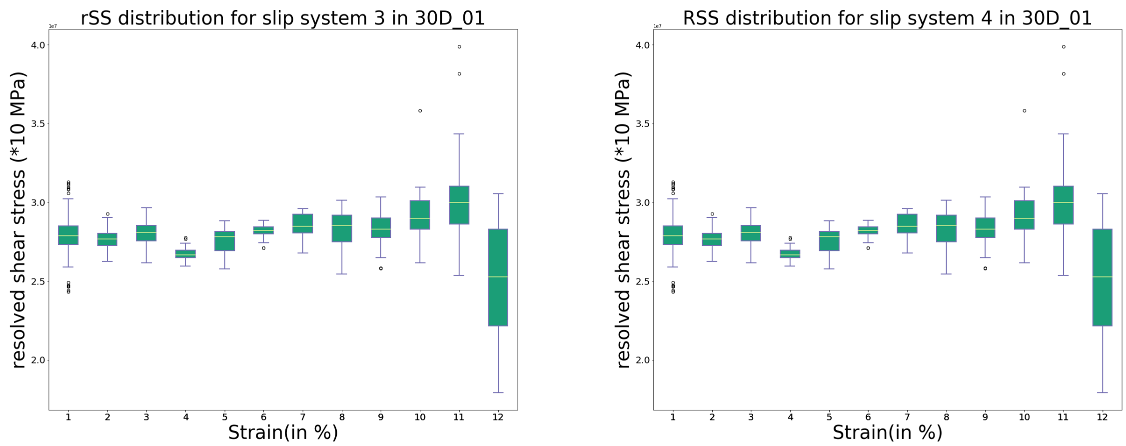


Figure 3.13: Resolved shear stress on slip system 1 and 2 with strain

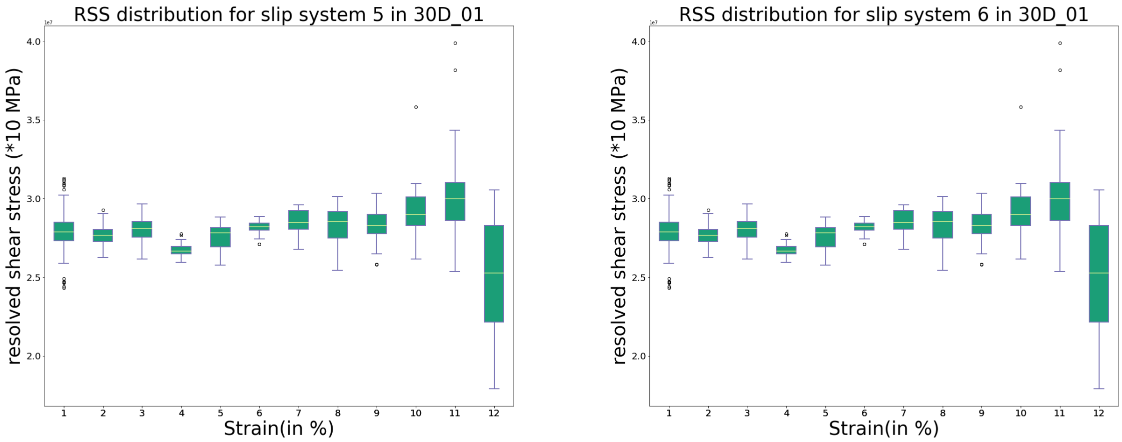


Figure 3.14: Resolved shear stress on slip system 1 and 2 with strain

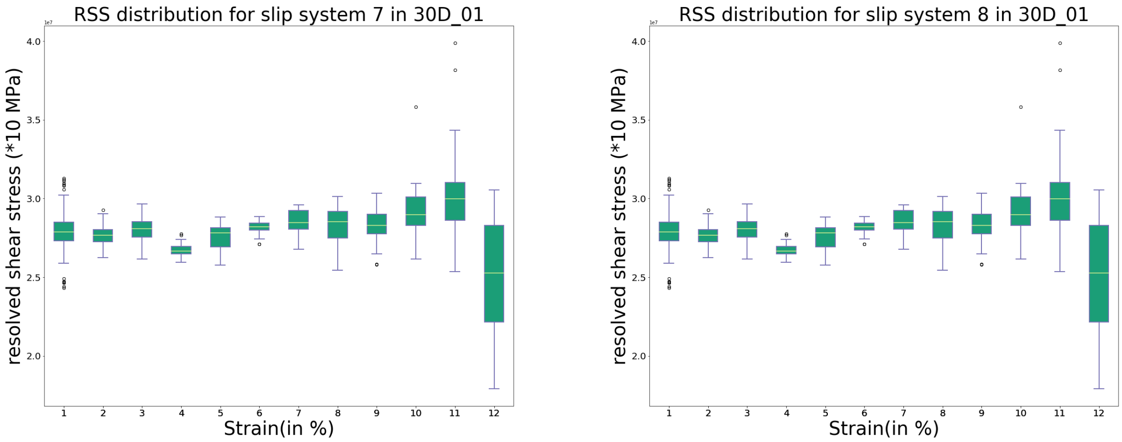


Figure 3.15: Resolved shear stress on slip system 1 and 2 with strain

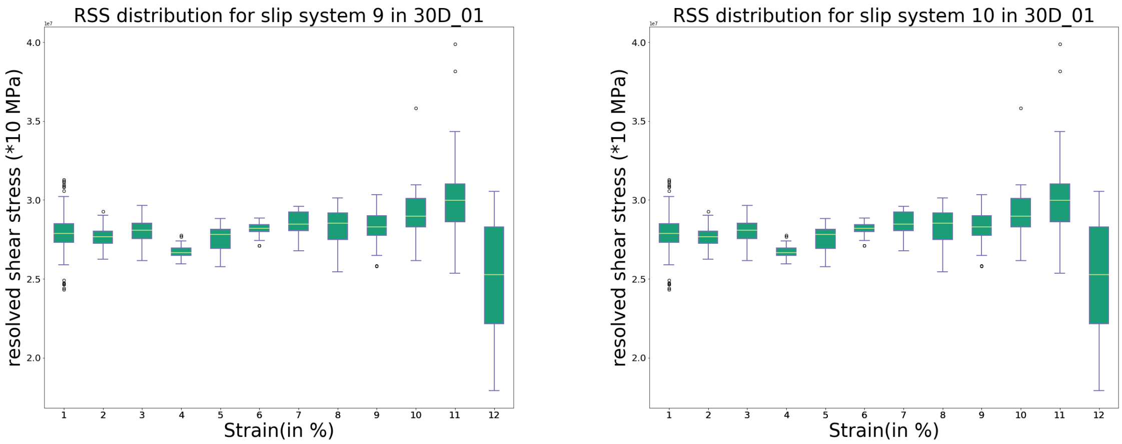


Figure 3.16: Resolved shear stress on slip system 1 and 2 with strain

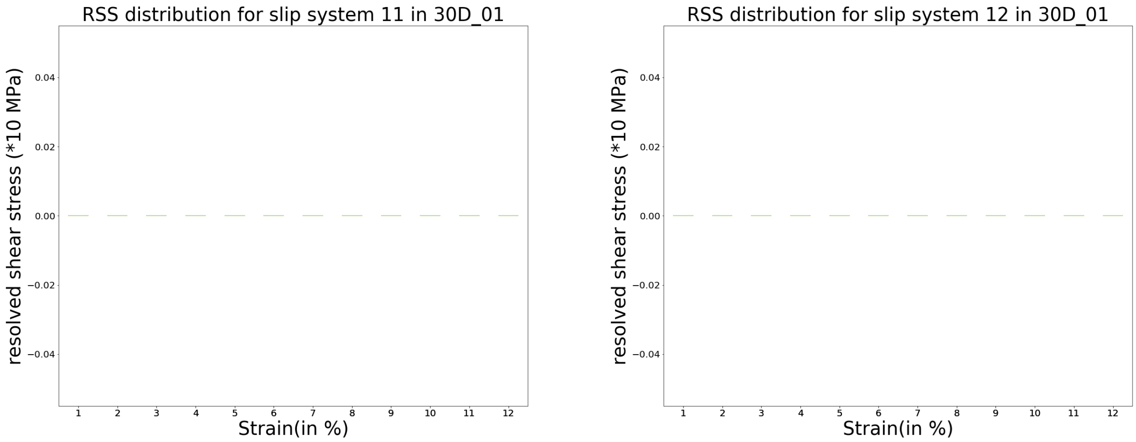


Figure 3.17: Resolved shear stress on slip system 1 and 2 with strain

From the above box plot it can be seen that at very low strain value value of resolved shear stress is almost equal on every slip plane while there is a large variance in resolved shear stress at higher strain. Also we can show that slip system 1, 2, 11 and 12 are not active during the deformation while rest all other are active.

3.3 CNN model

After multiple simulations, we collected initial fingerprint of the data into different classes meaning initial fingerprint of 28 dislocation simulation represent class 1, 30 dislocation represent class 2 and so on. Now we have initial structure of 10 classes(30D, 32D, 34D, 36D, 38D,40D, 42D, 60D,96D,128D) and we have to classify them using CNN. In CNN model following layers are used in their respective order:

1. Convolution Layer with ReLu activation
2. Max Pooling Layer
3. Again Convolution Layer followed by Max pooling layer.
4. Flatten layer
5. 2 Fully connected layer followed by activation function
6. Finally Softmax activation function is used to assign to compute the class scores.

Using keras and tensorflow as backend we trained our model for 10 different classes. Adam optimizer was used and loss was calculated using cross entropy function. Following is the summary of the model for classification.

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 62, 62, 64)        1792
_________________________________________________________________
activation_4 (Activation)    (None, 62, 62, 64)        0
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 31, 31, 64)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 29, 29, 32)        18464
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 14, 14, 32)        0
_________________________________________________________________
flatten_2 (Flatten)          (None, 6272)              0
_________________________________________________________________
dense_3 (Dense)              (None, 64)                401472
_________________________________________________________________
activation_5 (Activation)    (None, 64)                0
_________________________________________________________________
dense_4 (Dense)              (None, 21)                1365
_________________________________________________________________
activation_6 (Activation)    (None, 21)                0
=================================================================
Total params: 423,093
Trainable params: 423,093
Non-trainable params: 0
```

Figure 3.18: Summary of CNN model

3.4 Vanilla Deep Regression

With the evolution of deep learning, Artificial Intelligence and data science has been revolutionised. Many vision tasks such as Human pose estimation and autonomous self driving cars have reshaped the understanding of neural network. In self driving car,problem remain a supervised regression problem between the the road images and car steering angles in real-time from the cameras of a car. Input being the Images and output is car steering angle and speed.[20]

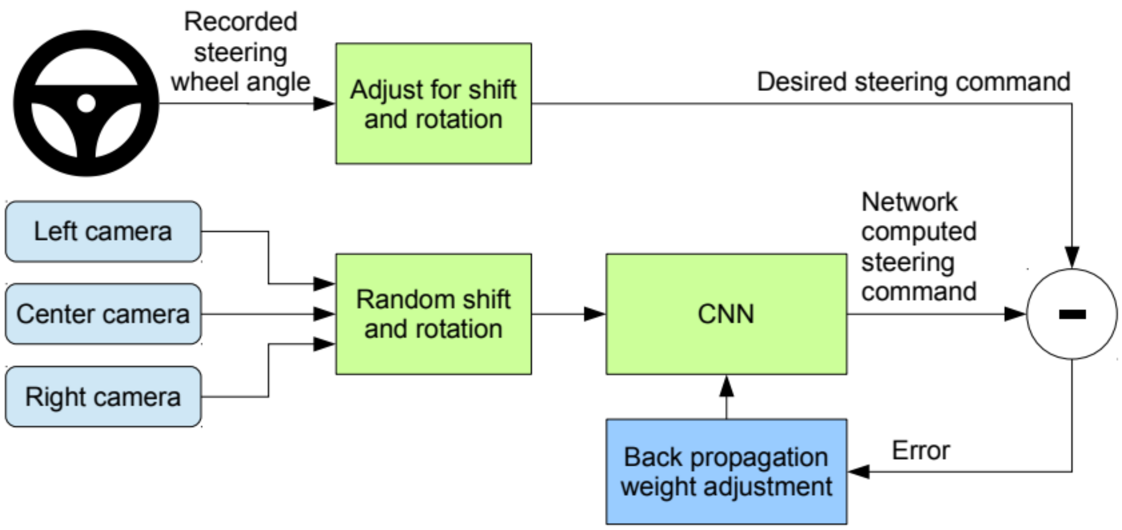


Figure 3.19: High-level view of the Nvidia's data collection system for self driving cars [20]

Three front-facing "cameras" record data from the car's point of view and send it to driving simulator; apart from it throttle, speed and steering angles are also taken as input. This is very much similar to our main problem i.e. we have fingerprint images of our data and for a particular strain value we have to predict what should be the stress.

The goal is to predict a set of continuous interdependent values. We tried using the similar architecture with some modifications. We changed some steps in data preprocessing and we used convolution neural network with a linear regression top layer. Final model was tried fitted using keras with tensorflow as back end. Work is also inpired from Vanilla deep regression problems that are crucial in the context of human-robot interaction i.e. robot must have good head and body posture while facing a person.[21]

Following is the summary of the architecture that we applied:

```
Model: "sequential_11"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lambda_11 (Lambda)           (None, 231, 231, 4)       0
_________________________________________________________________
conv2d_51 (Conv2D)           (None, 114, 114, 24)      2424
_________________________________________________________________
conv2d_52 (Conv2D)           (None, 55, 55, 36)        21636
_________________________________________________________________
conv2d_53 (Conv2D)           (None, 26, 26, 48)        43248
_________________________________________________________________
conv2d_54 (Conv2D)           (None, 24, 24, 64)        27712
_________________________________________________________________
conv2d_55 (Conv2D)           (None, 22, 22, 64)        36928
_________________________________________________________________
max_pooling2d_11 (MaxPooling (None, 11, 11, 64)        0
_________________________________________________________________
flatten_11 (Flatten)         (None, 7744)              0
_________________________________________________________________
dense_41 (Dense)             (None, 100)               774500
_________________________________________________________________
dropout_31 (Dropout)         (None, 100)               0
_________________________________________________________________
dense_42 (Dense)             (None, 50)                5050
_________________________________________________________________
dropout_32 (Dropout)         (None, 50)                0
_________________________________________________________________
dense_43 (Dense)             (None, 25)                1275
_________________________________________________________________
dropout_33 (Dropout)         (None, 25)                0
_________________________________________________________________
dense_44 (Dense)             (None, 1)                 26
=================================================================
Total params: 912,799
Trainable params: 912,799
Non-trainable params: 0
```

Figure 3.20: Summary of Vanilla regression architecture

Chapter 4

Result and Discussion

4.1 Understanding of data

We ran multiple simulations of various initial dislocations and allowed to deform internal cell structure for a certain period of time. Some simulations ran for days, some for weeks and even some for months to get upto a certain value of strain. Following are the stress strain curve of simulations:

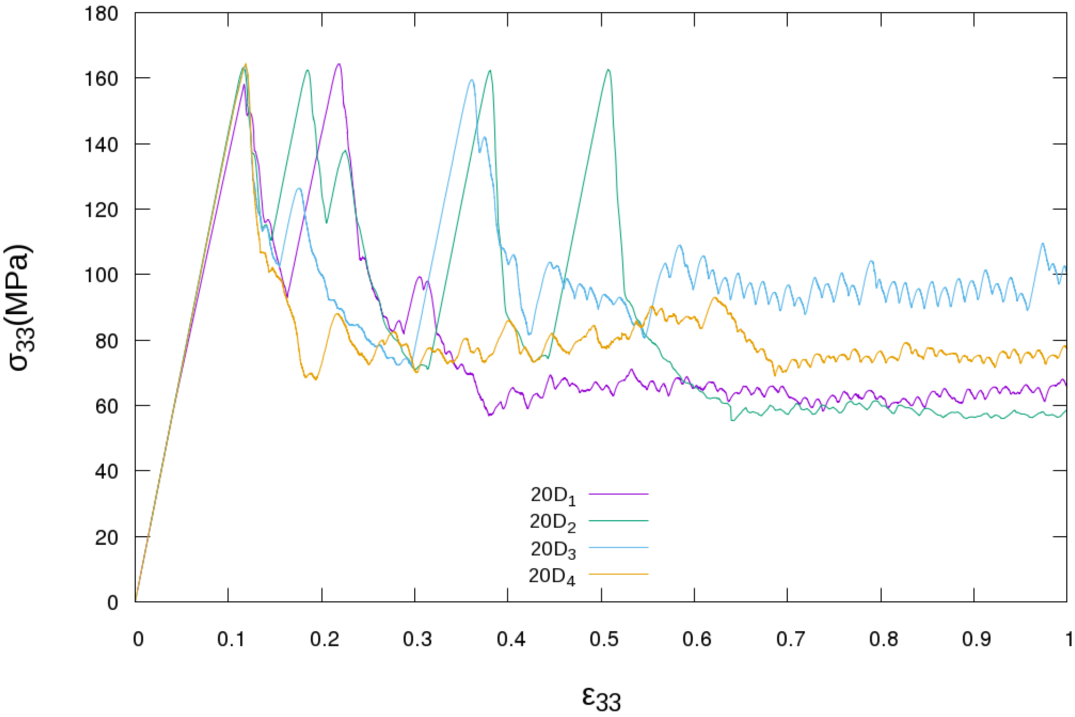


Figure 4.1: Stress strain curves for multiple simulations with 20 initial dislocation line

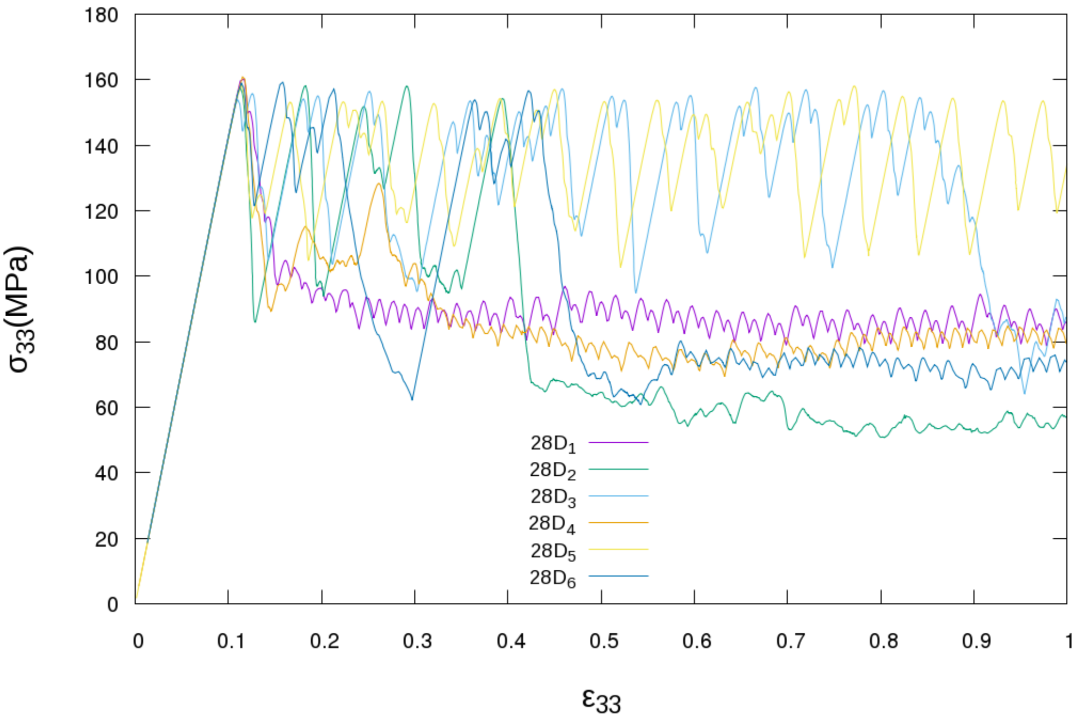


Figure 4.2: Stress strain curves for multiple simulations with 28 initial dislocation line

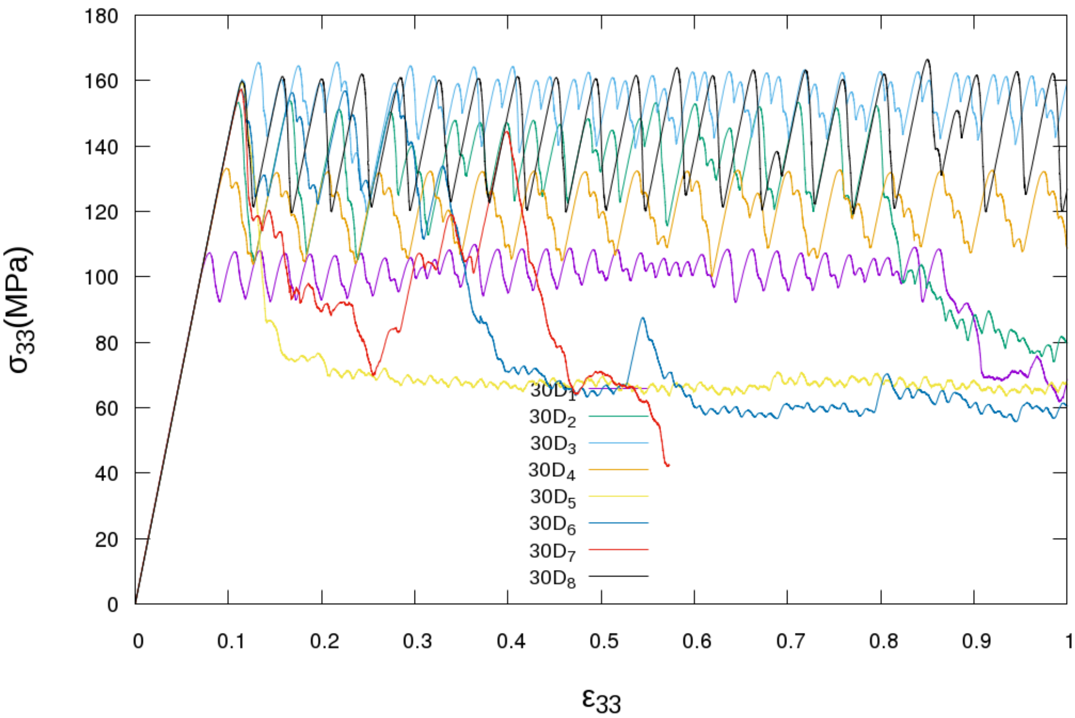


Figure 4.3: Stress strain curves for multiple simulations with 30 initial dislocation line

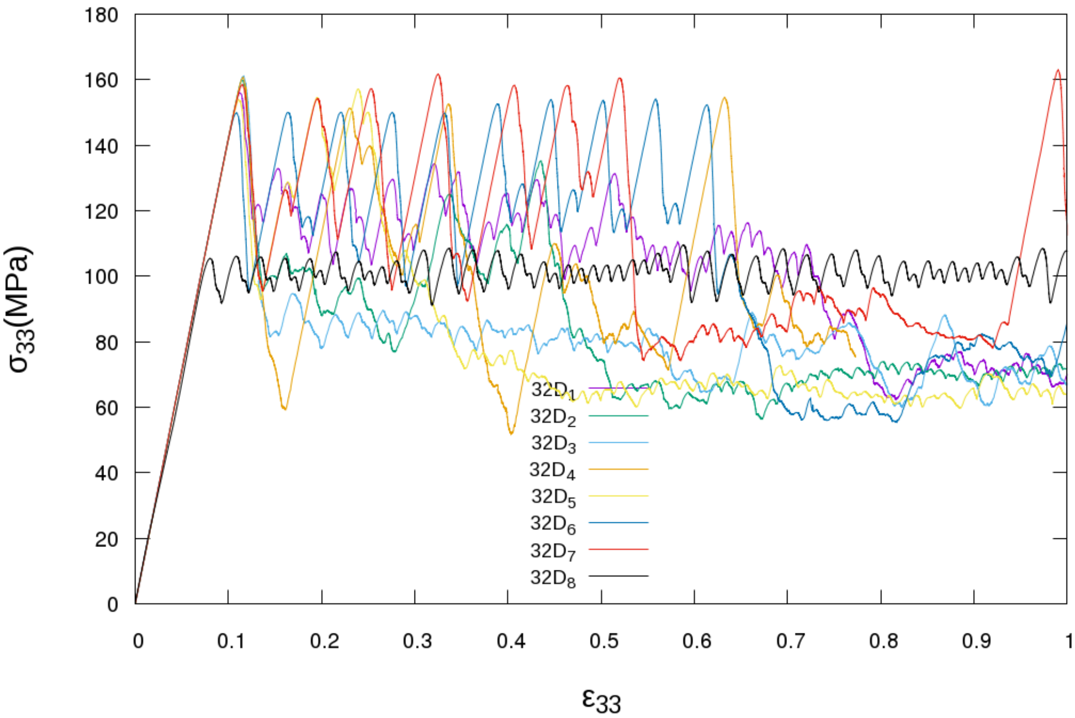


Figure 4.4: Stress strain curves for multiple simulations with 32 initial dislocation line

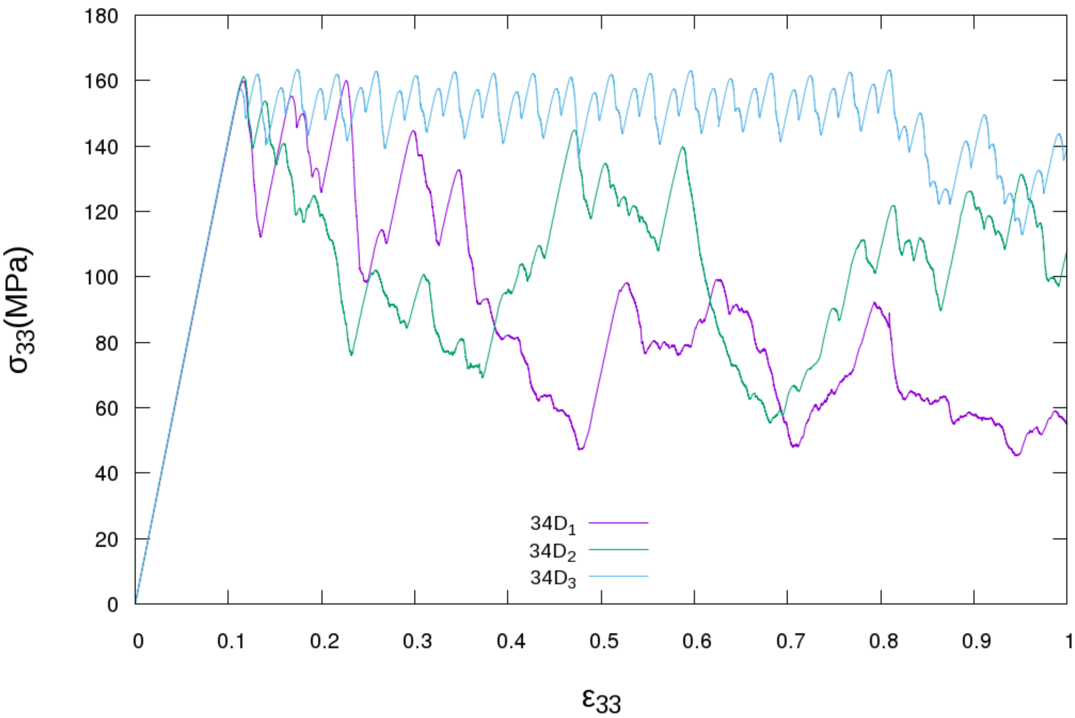


Figure 4.5: Stress strain curves for multiple simulations with 34 initial dislocation line

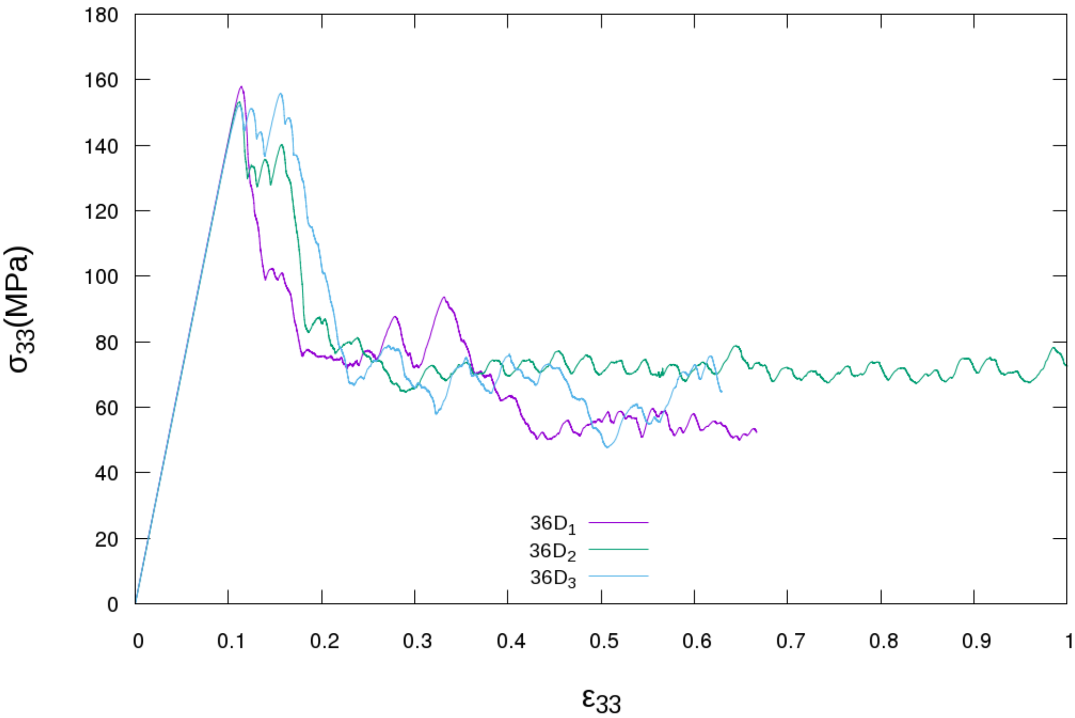


Figure 4.6: Stress strain curves for multiple simulations with 36 initial dislocation line

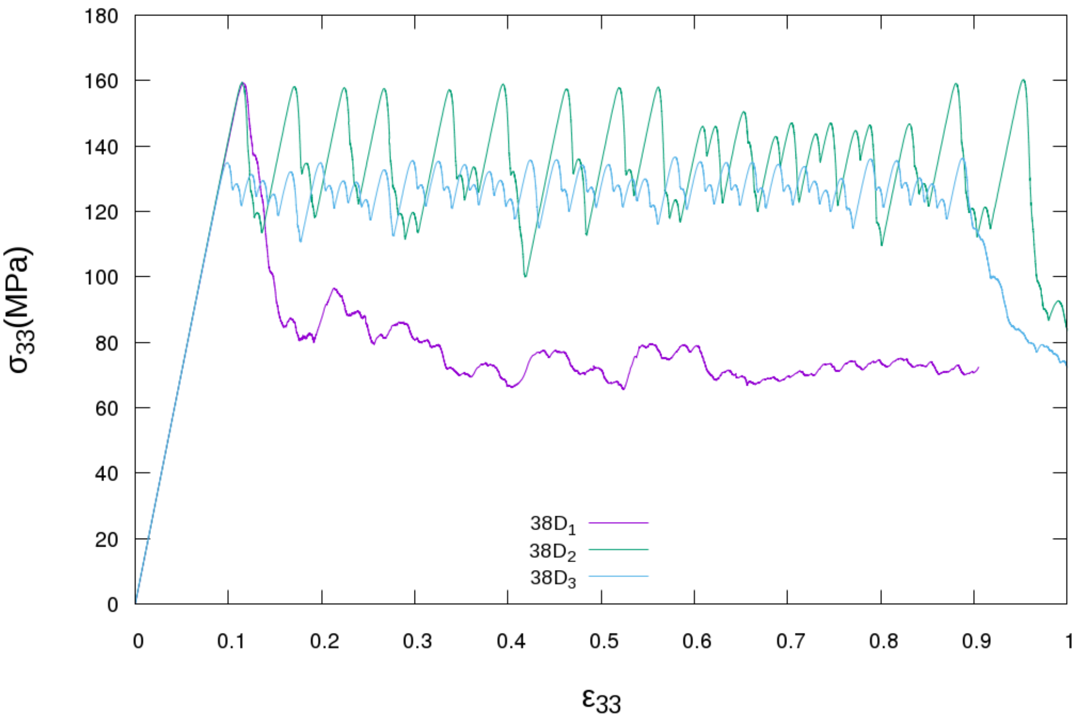


Figure 4.7: Stress strain curves for multiple simulations with 38 initial dislocation line

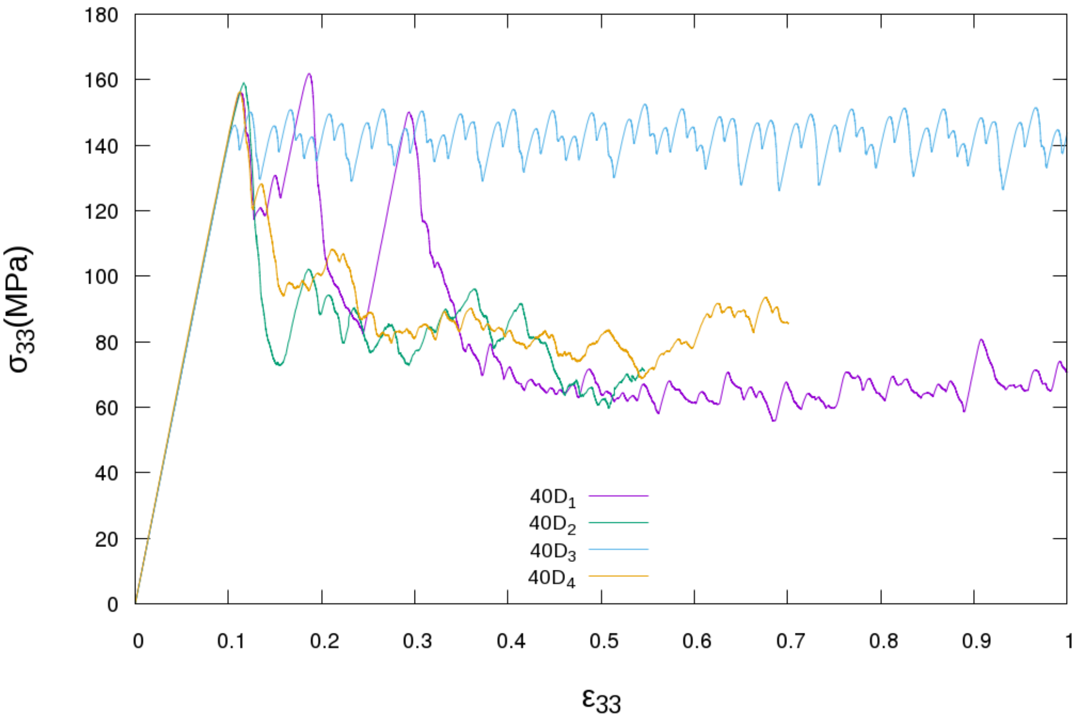


Figure 4.8: Stress strain curves for multiple simulations with 40 initial dislocation line

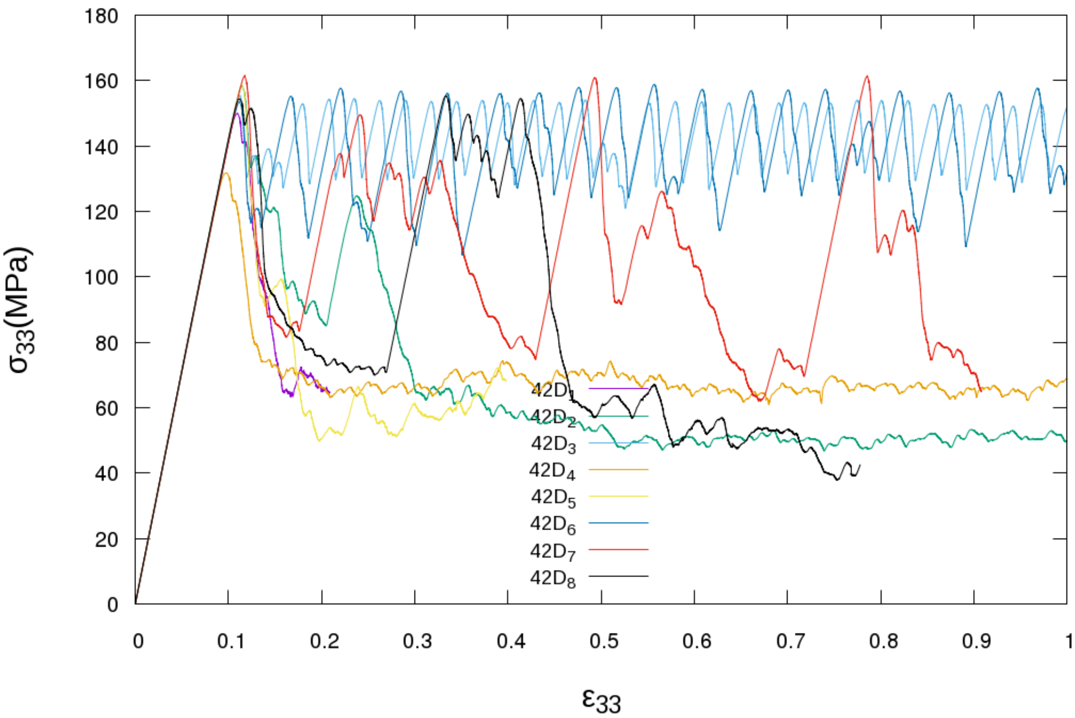


Figure 4.9: Stress strain curves for multiple simulations with 42 initial dislocation line

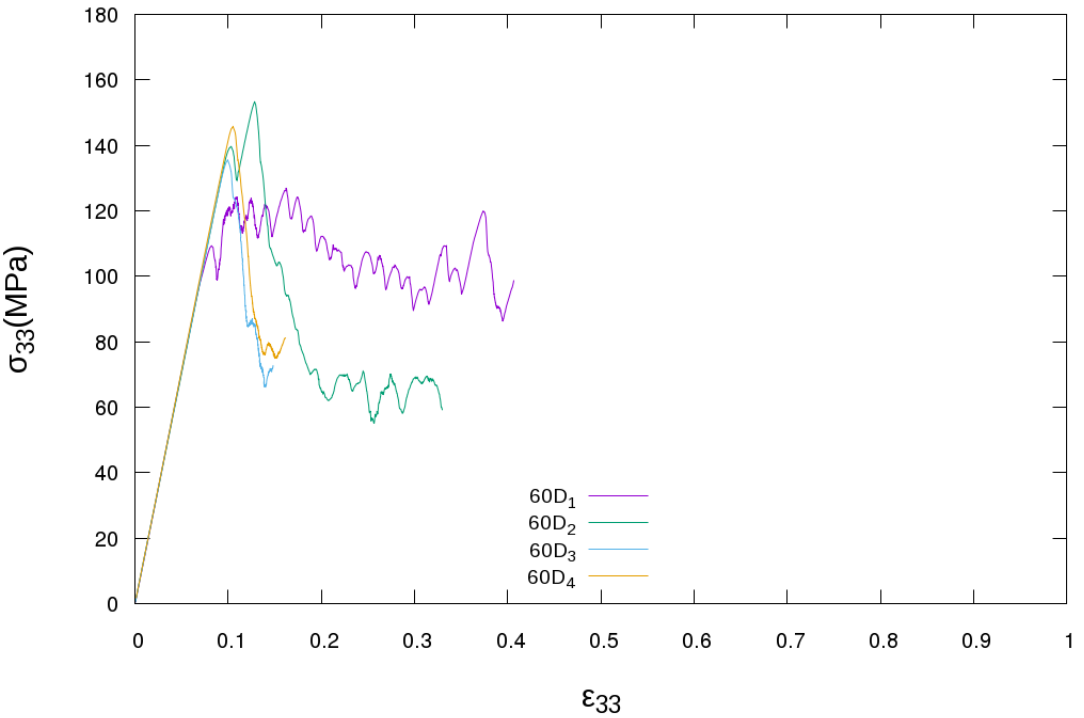


Figure 4.10: Stress strain curves for multiple simulations with 60 initial dislocation line

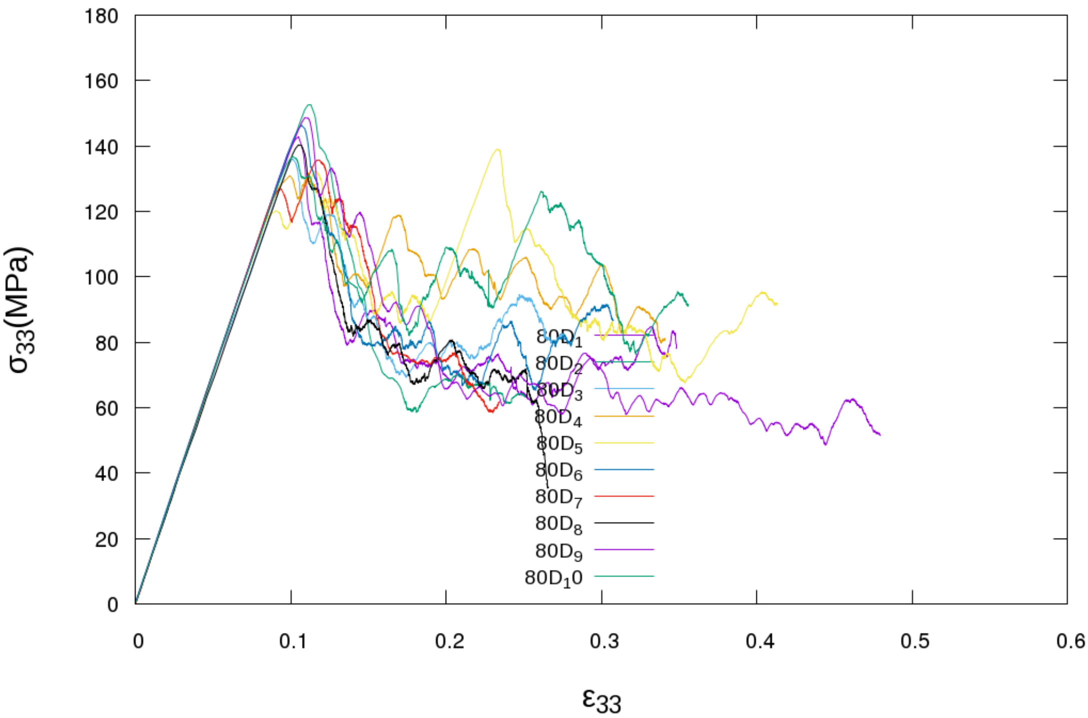


Figure 4.11: Stress strain curves for multiple simulations with 80 initial dislocation line

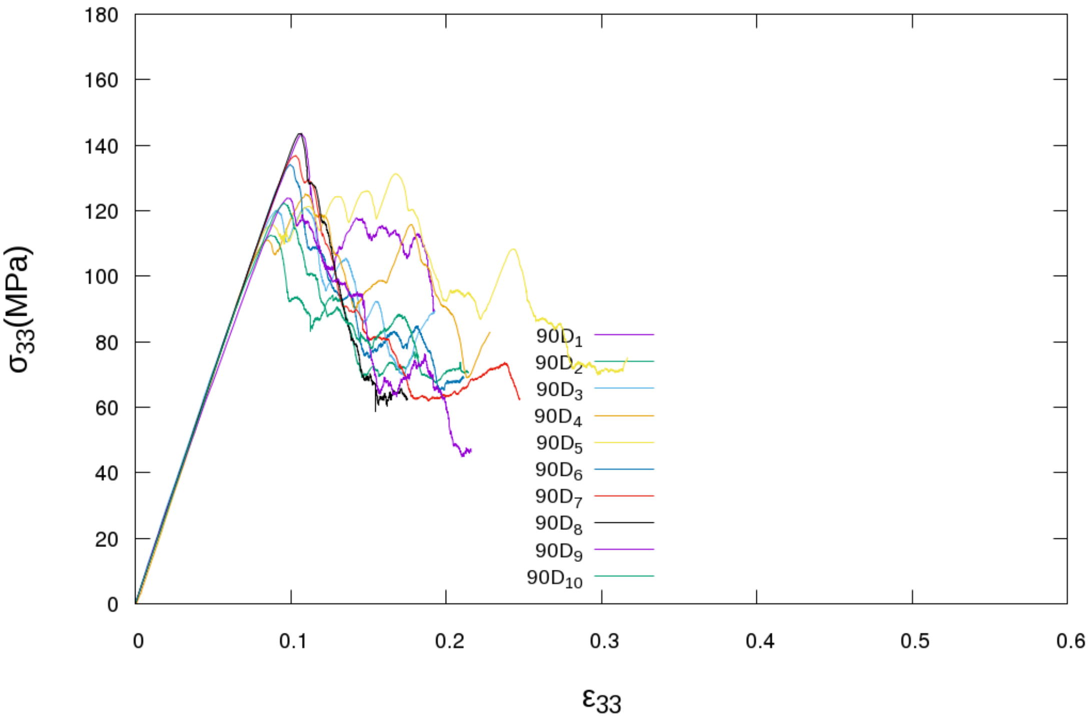


Figure 4.12: Stress strain curves for multiple simulations with 90 initial dislocation line

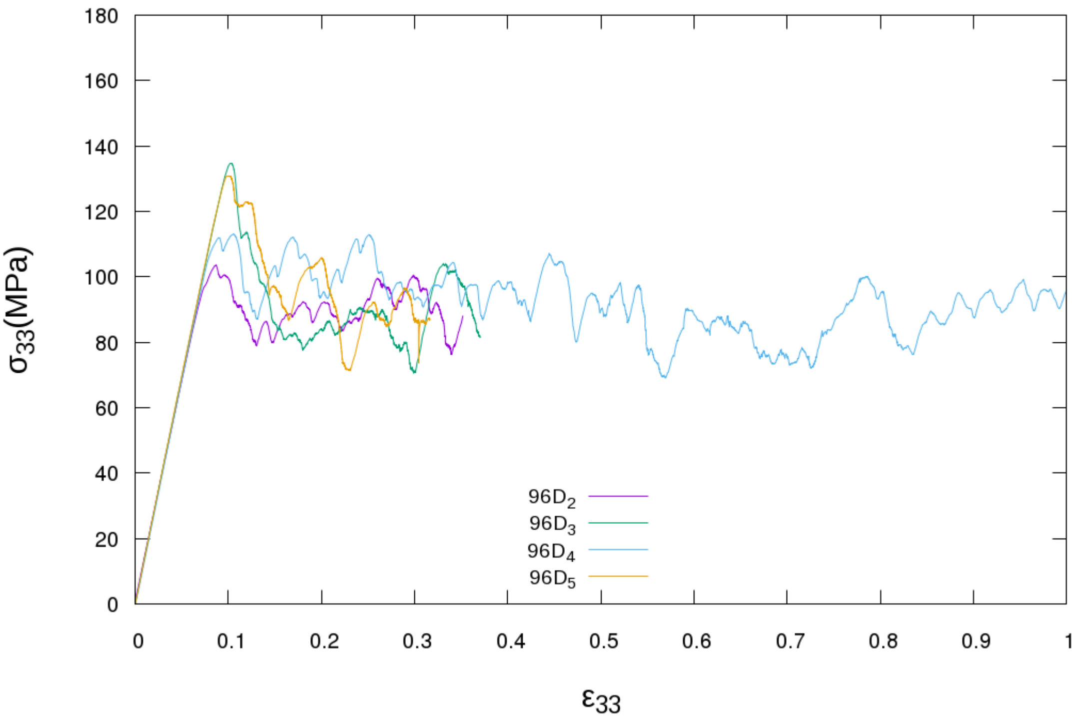


Figure 4.13: Stress strain curves for multiple simulations with 96 initial dislocation line

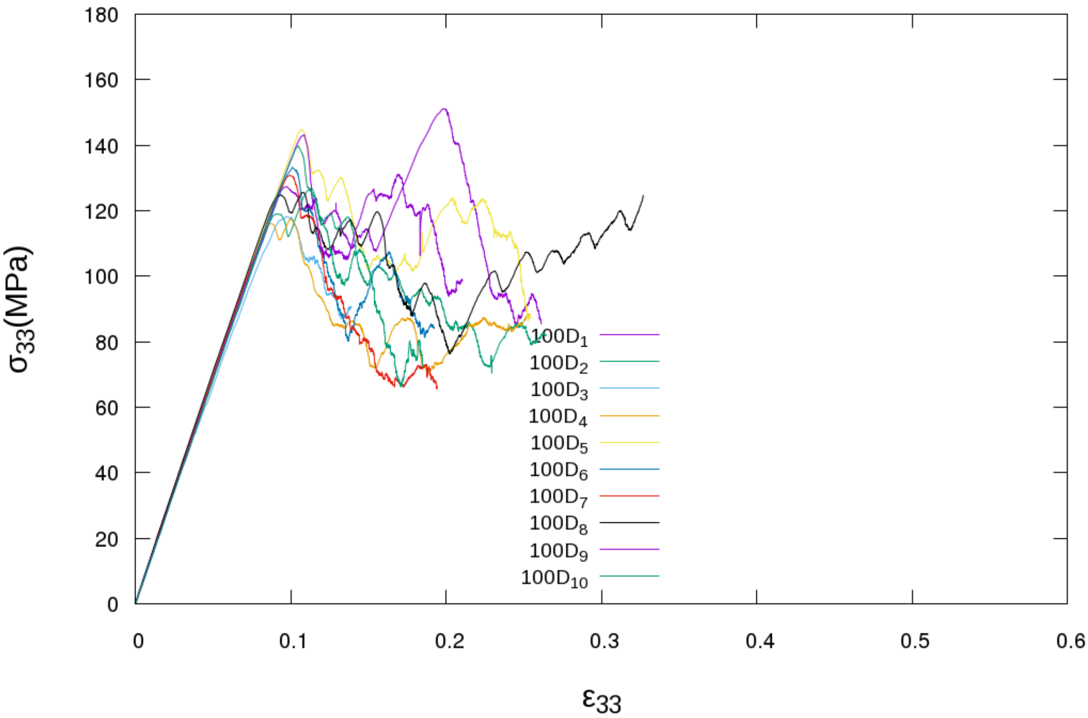


Figure 4.14: Stress strain curves for multiple simulations with 100 initial dislocation line

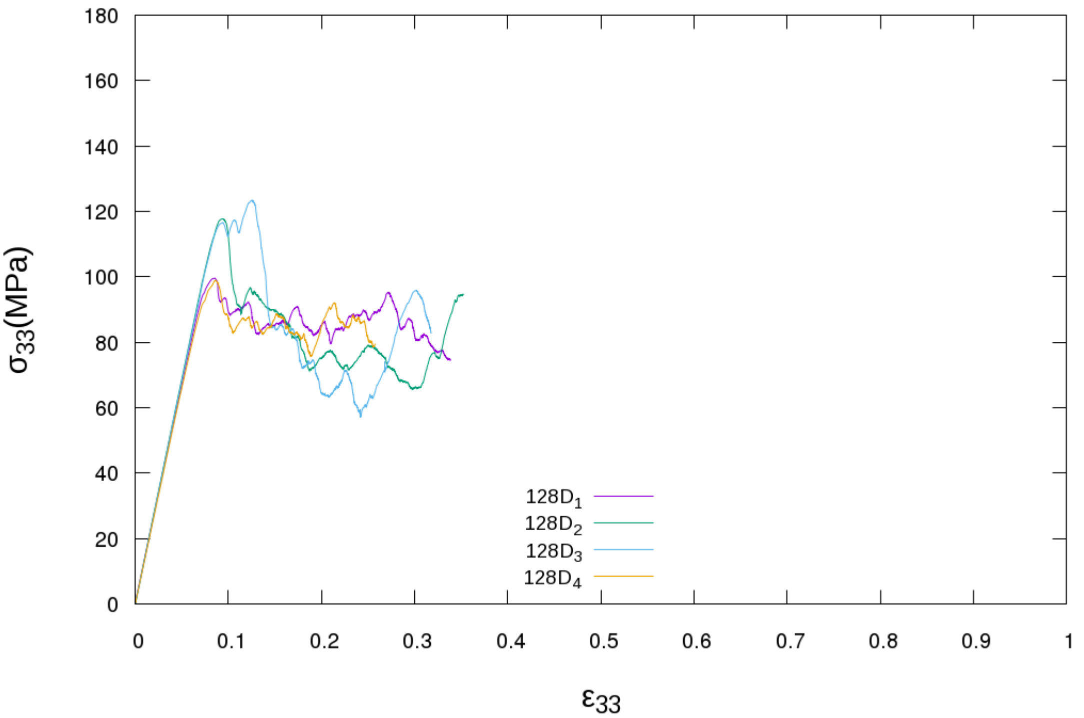


Figure 4.15: Stress strain curves for multiple simulations with 128 initial dislocation line

4.2 Converting 3D image into 2D image

Using Matplotlib, Numpy, Pandas library in python data from dislocation dynamics was converted into an image such that each pixel contain some input parameter.

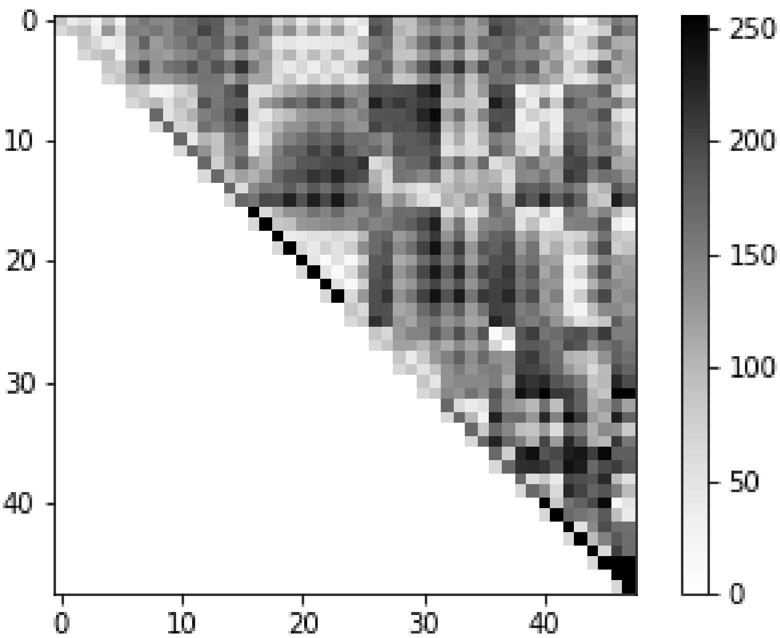


Figure 4.16: A sample converted 3D image

Now, since CNN require huge number of data set to train its model, we changed each pixel intensity to 0 , 50 and 100 on a scale between 0-255 to get multiple images in each class.

4.3 Learning Curves

Complete data was split into training:test:validation as 70:15:15 ratio and Model was trained for 100 epochs.Loss was calculated by cross entropy loss function and model was optimised using ADAM optimiser. Following accuracy and loss were observed.

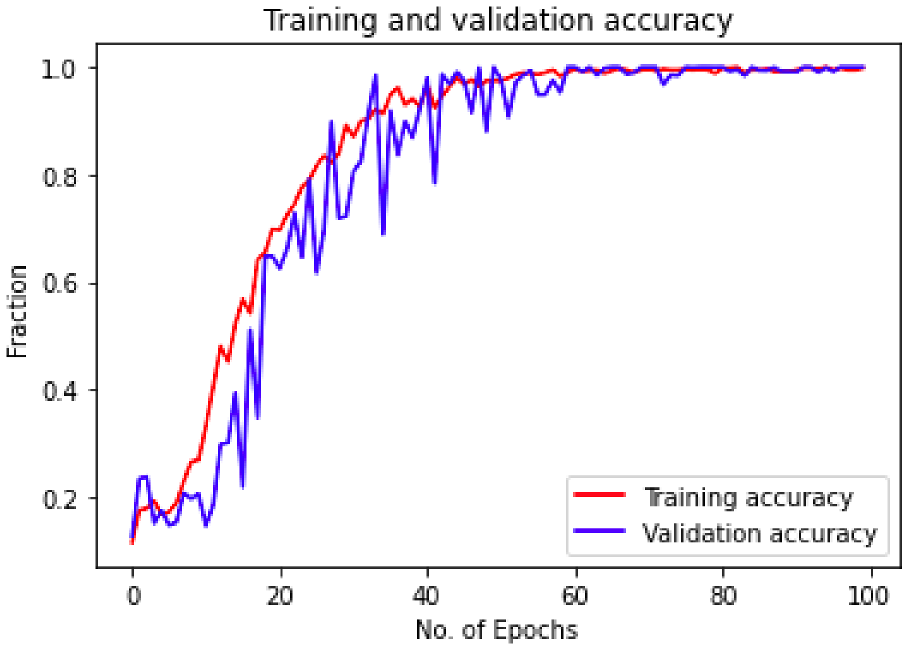


Figure 4.17: Accuracy of training dataset and validation dataset

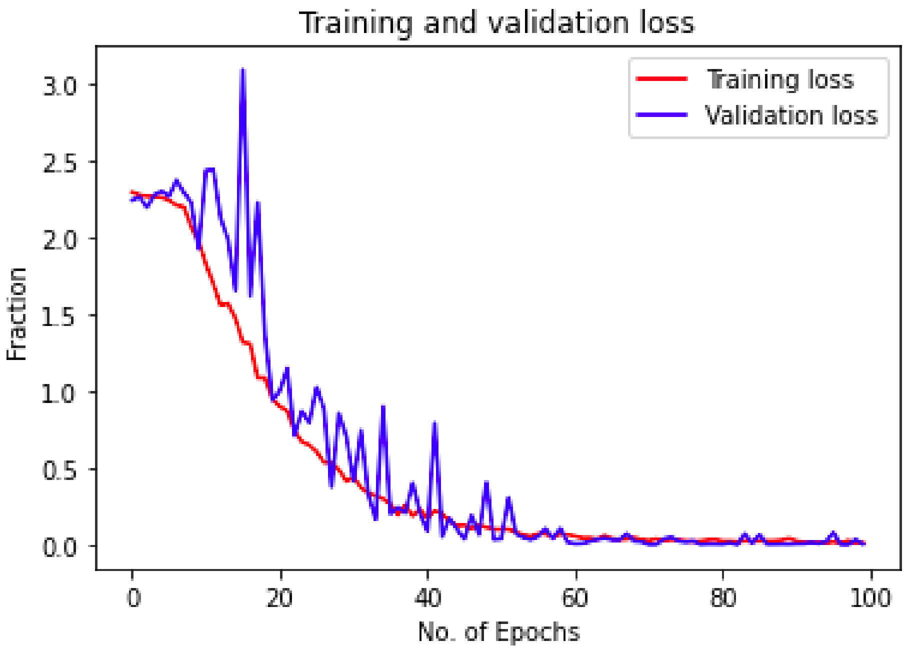


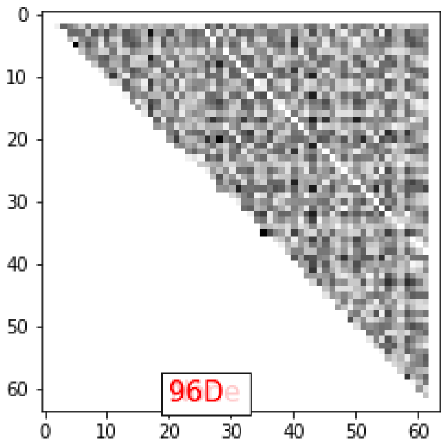
Figure 4.18: Loss in training and validation dataset

4.4 Prediction Result

We took random images from test data which were initially not used to train model and got following result:

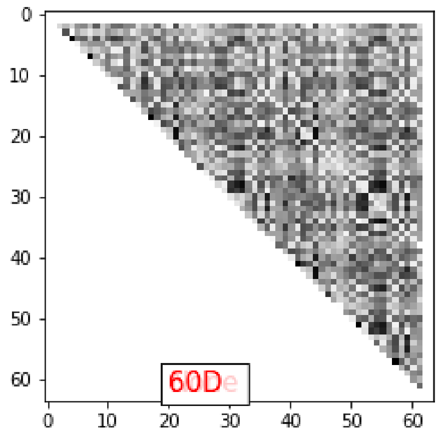
```
a = "Split_image/test/96D/96D_1_39.png'
pred(a)
```

[[0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]]



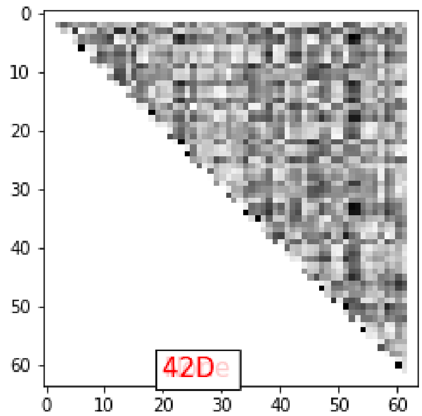

```
a = "Split_image/test/60D/60D_1_35.png"
pred(a)
```

[[0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]]




```
a = "Split_image/test/42D/42D_1_26.png"
pred(a)
```

[[0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]]




```
a = "Split_image/test/40D/40D_1_12.png"
pred(a)
```

[[0. 0. 0. 0. 0. 0. 1. 0. 0. 0.]]

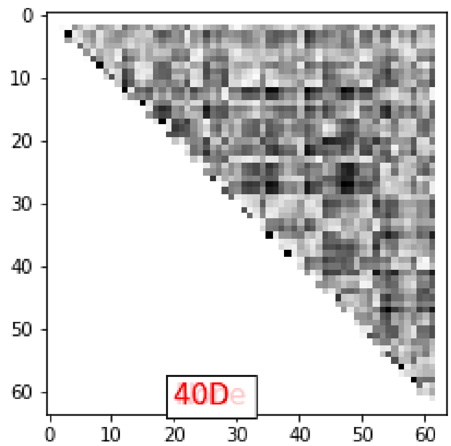


Figure 4.20: Predicted Results

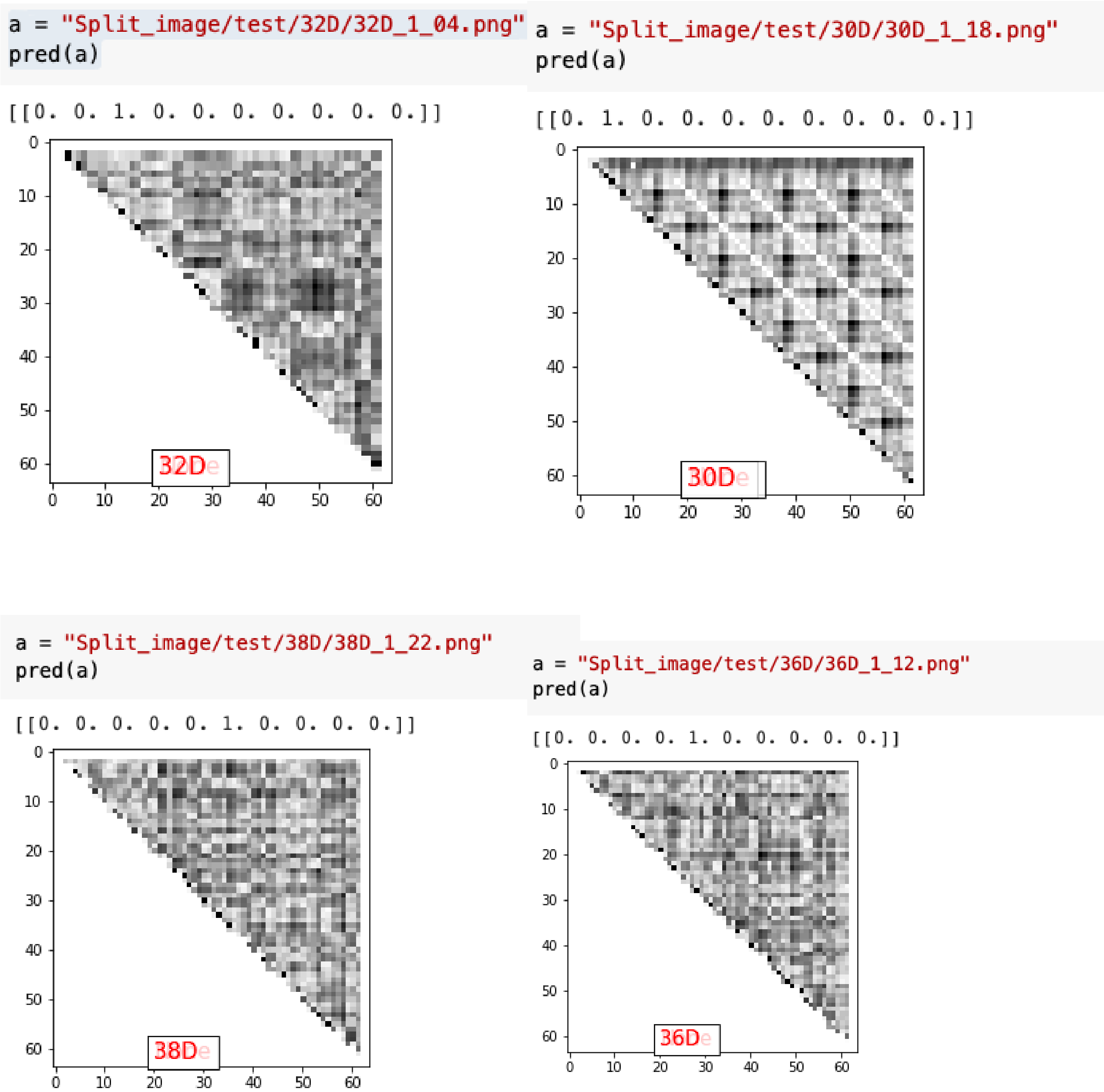


Figure 4.22: Predicted Results

Our Convolution neural network model was successfully predicting test images.

4.5 Preprocessing of data for VDR

We have three different datasets,namely, Image dataset, stress-strain data and file name/id for that particular data value for single simulation and we performed multiple simulation. We first tried to map stress strain data with file name. Following are the result of mapping:

	stress	strain
30D_1_01.csv	53800000.0	0.000377
30D_1_02.csv	86300000.0	0.000607
30D_1_03.csv	110000000.0	0.0007740000000000001
30D_1_04.csv	126000000.0	0.000891
30D_1_05.csv	154000000.0	0.0011099999999999999
30D_1_06.csv	124000000.0	0.0020399999999999997
30D_1_07.csv	99800000.0	0.00225
30D_1_08.csv	77900000.0	0.00243
30D_1_09.csv	65100000.0	0.00259
30D_1_010.csv	56900000.0	0.00271
30D_1_011.csv	60400000.0	0.00281
30D_1_012.csv	63400000.0	0.00294
30D_1_013.csv	64800000.0	0.00301

Figure 4.23: Mapping of Stress-strain value with file name

Now this data is incomplete without the fingerprint of internal structure i.e. 2-D images. We then combined all the data into one file and stored it as pickle file.

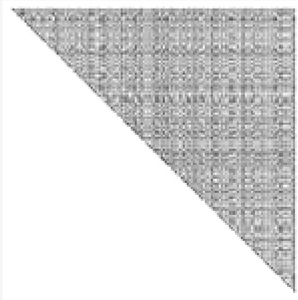
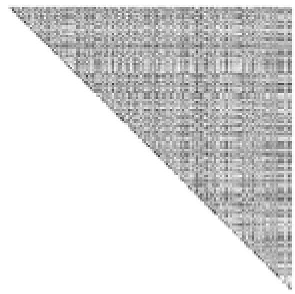
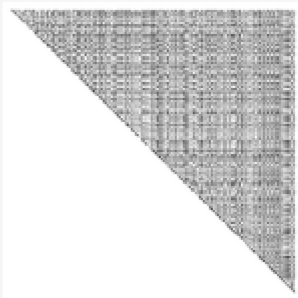
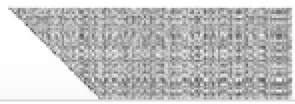
	lm_id	stress	strain	image	image_th
0	100D_1_01.csv	65770000.0	0.000470	100D_1/100D_1.png	
1	100D_1_02.csv	105400000.0	0.000771	100D_1/100D_2.png	
2	100D_1_03.csv	115300000.0	0.000854	100D_1/100D_3.png	
					

Figure 4.24: Mapping of Stress-strain value with file name and Image(Fingerprint)

4.6 Training of data for VDR

We compiled the data for vanilla deep regression(VDR) with mean square error as loss function and ADAM as optimiser. With validation split to be 25% of total training data, we compiled the data using keras and tensorflow as backend for 30 epochs. We performed multiple iteration of data preprocessing, training and testing with different architecture is required. From the curve below, we can say that predicted value of stress was oscillating between a range in spite of the fact that training and testing losses are nearly zero.

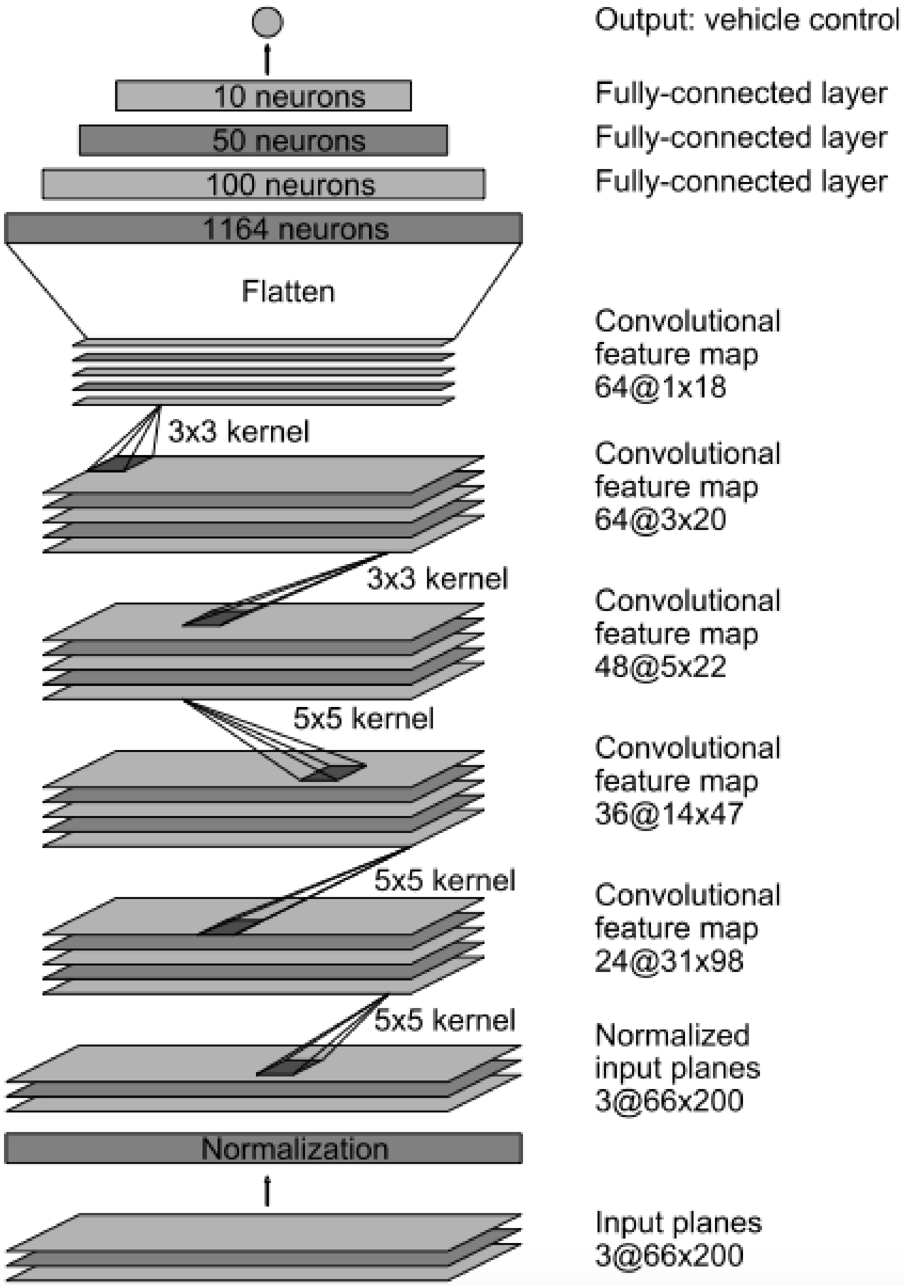


Figure 4.25: NVIDIA Architecture for Self Driving cars

The architecture that we tried is similar to NVIDIA architecture for self driving cars with some modifications and tested it for multiple cases. It starts with convolution layers that have 24, 36 and 48 filters of size 5x5 filters respectively in the first 3 layers and a stride of 2 in both the directions. These layers are followed by two convolution layers each having 64 filters and 3x3 size with strides of 1 in both the directions. Exponential Linear Unit (ELU) activation function is used in all the five convolution layers. A max pooling layer follows. The layer is flattened and fed into four fully connected layers with first two having ELU activation. Dropout is applied in each layer after the first one with a probability of retaining the weights equal to 0.5. According to this, we get following result:

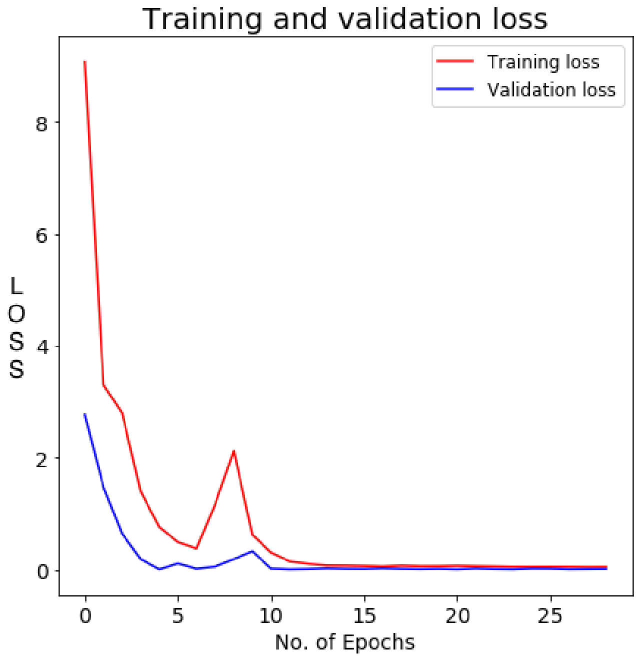


Figure 4.26: Training and testing loss

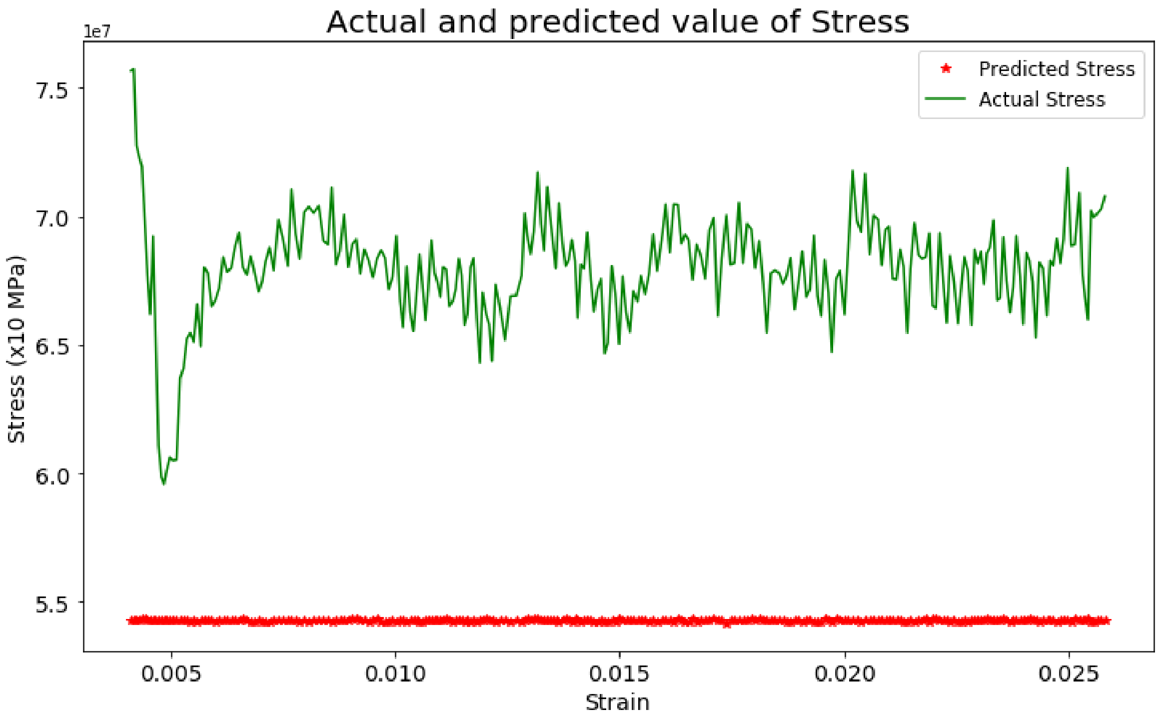


Figure 4.27: Actual and predicted Value of stress

4.6.1 Case 1

In this case, we applied some modifications in architecture by removing one dropout layer and one dense layer. Also changed all convolution size to 3x3 and with stride 2x2 in first three layer and stride of 1x1 in last two convolution layer. We have kept all other things same. After training and fitting, we get following result:

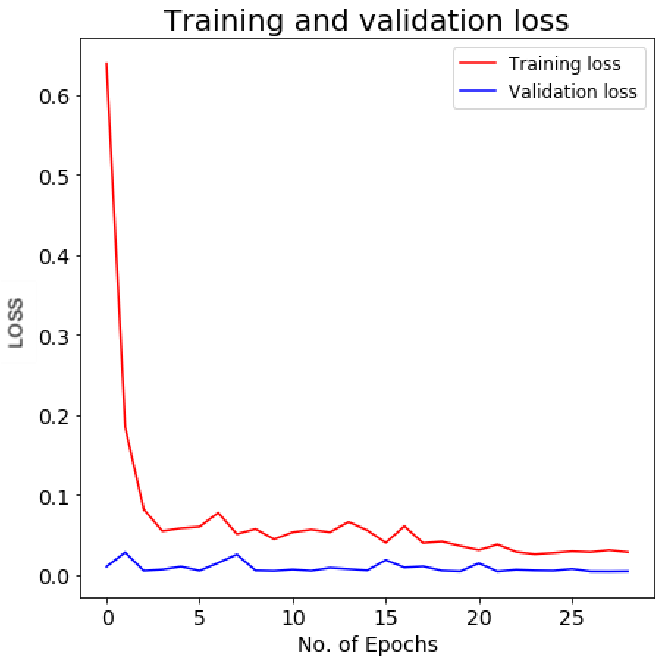


Figure 4.28: Training and testing loss

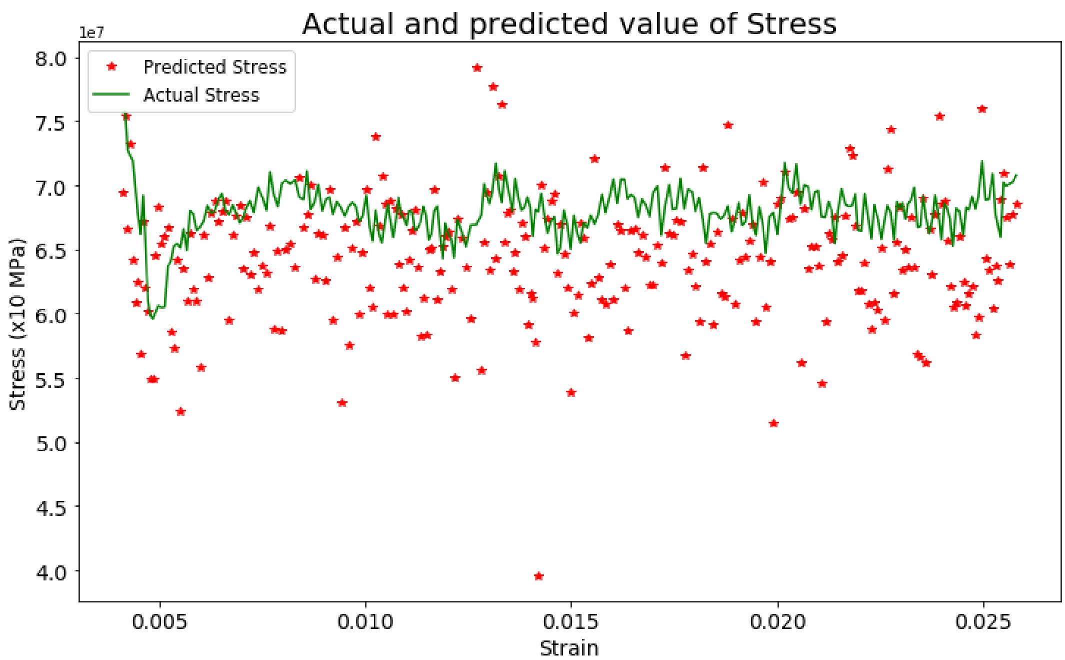


Figure 4.29: Actual and predicted Value of stress

4.6.2 Case 2

In this case, we have removed one dropout layer and one dense layer. Also changed first two convolution layer filter size as 2x2 and kept all other things same. After training and fitting, we get following result:

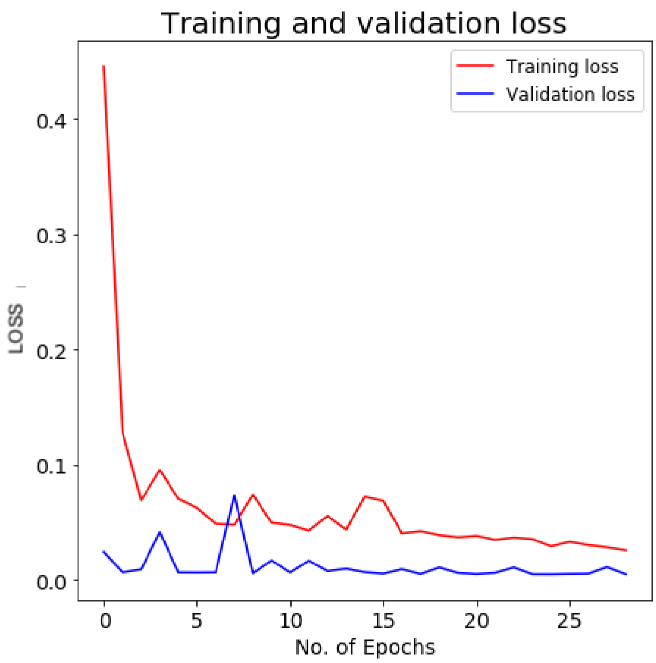


Figure 4.30: Training and testing loss

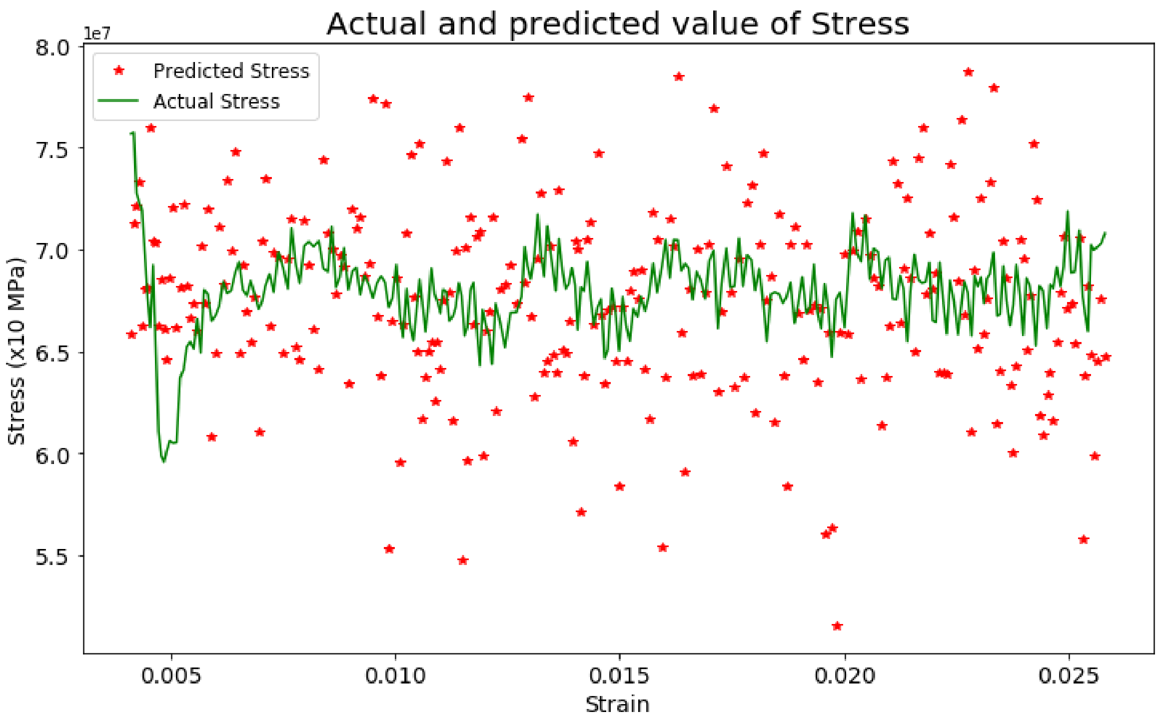


Figure 4.31: Actual and predicted Value of stress

4.6.3 Case 3

In this case only change is done in learning rate. We changed learning rate of ADAM optimizer to 0.002 from 0.001. All other parameters are kept same as that of case 2. After training and fitting, we get following result:

Figure 4.32: Training and testing loss

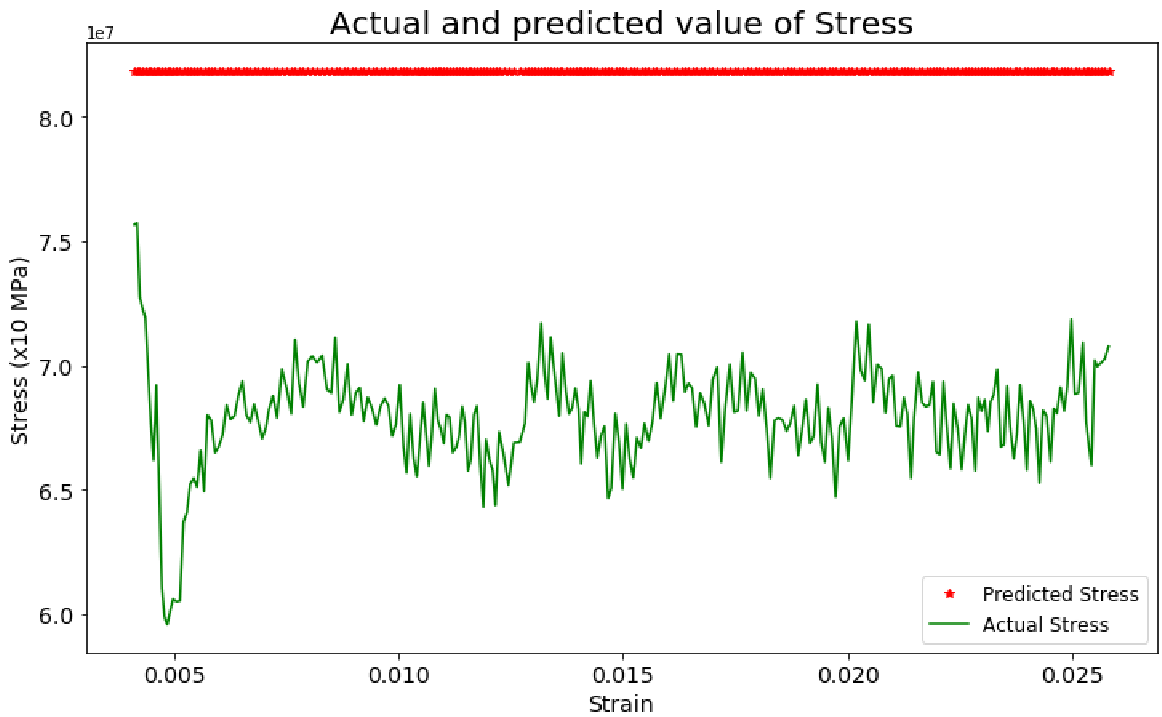


Figure 4.33: Actual and predicted Value of stress

4.6.4 Case 4

In this case, we changed optimizer to stochastic gradient descent (SGD) with learning rate 0.001. All other parameters are kept same as that of case 2. After training and fitting, we get following result:

Figure 4.34: Training and testing loss

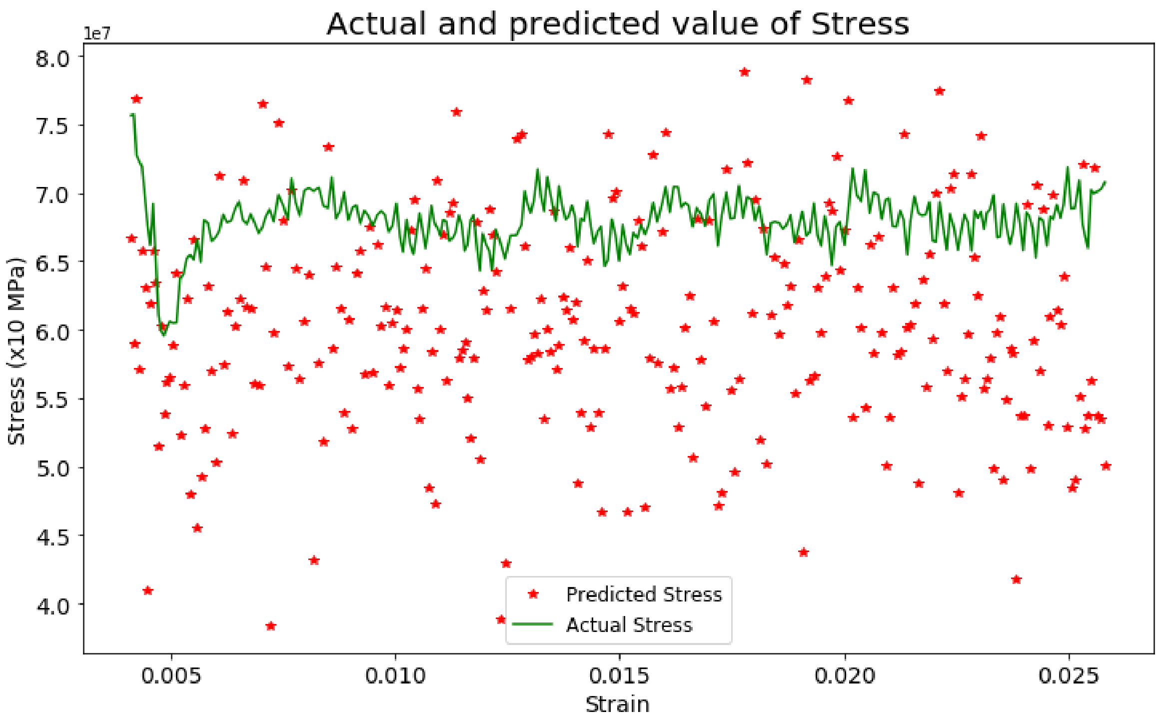


Figure 4.35: Actual and predicted Value of stress

4.6.5 Case 5

In this case, we mixed the data for multiple type of initial dislocation line. We kept all parameters as same as case 4 and trained for 70 epochs. After training and fitting, we get following result:

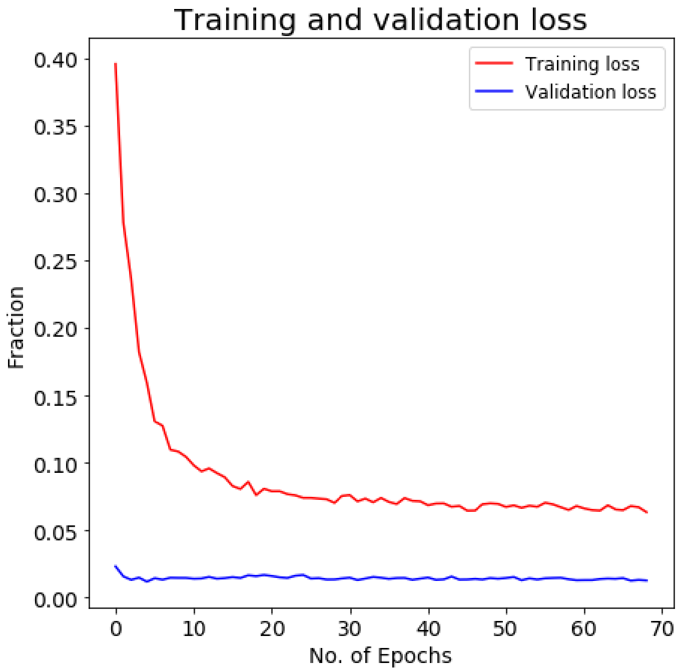


Figure 4.36: Training and testing loss

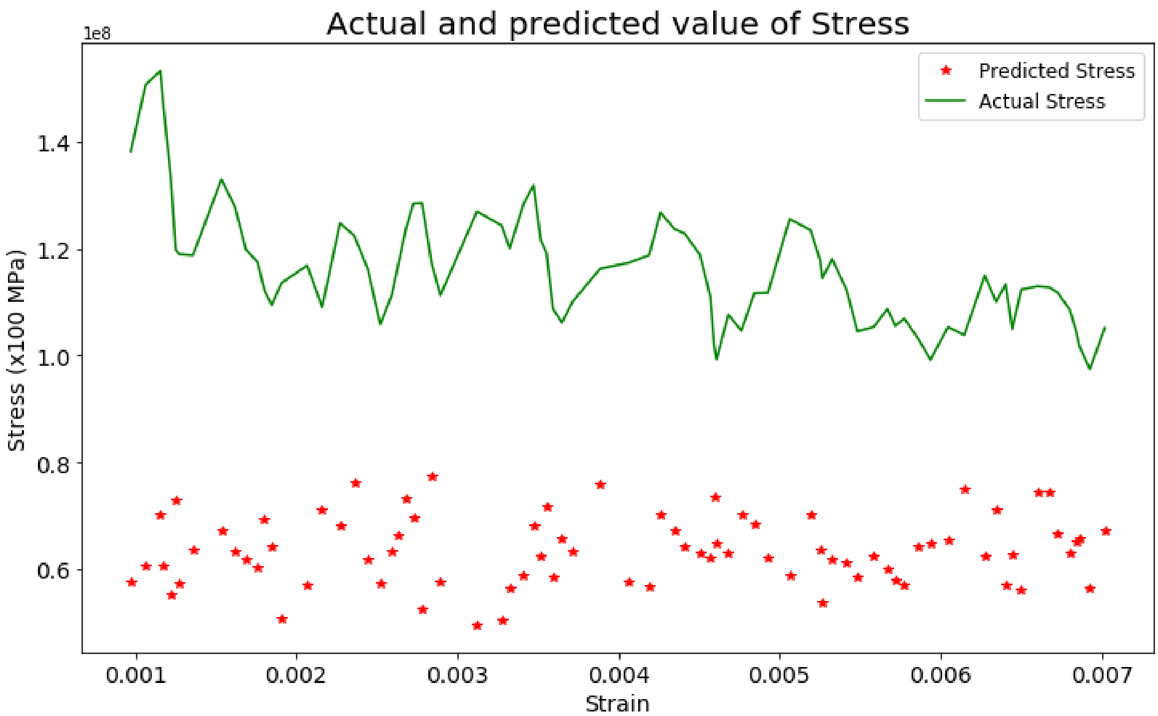


Figure 4.37: Actual and predicted Value of stress

Calculation of R-square value and plot between predicted and actual value:

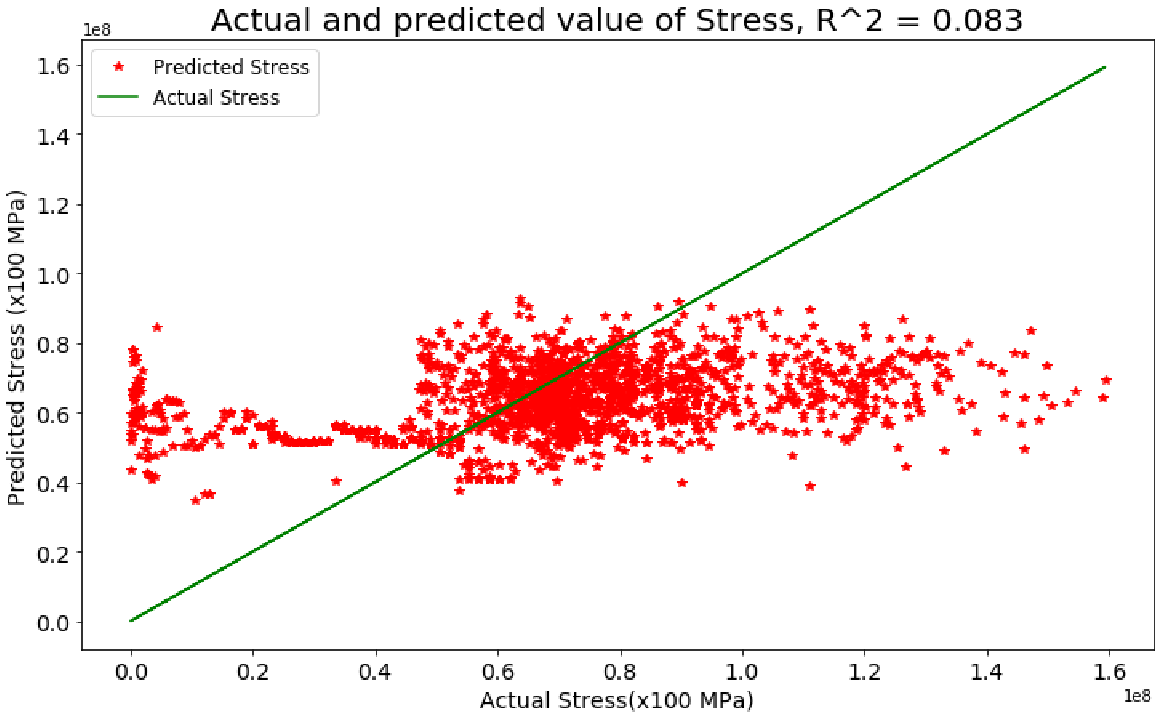


Figure 4.38: Actual and predicted Value of stress

Low R-square value indicated that model is not fitting all the points correctly.

Chapter 5

Summary

Data of different type of dislocations was collected from multiscale dislocation dynamics plasticity model. After data preprocessing images were formed into a 2-D image that contain information such as distance between nodes,burgers vector, slip plane and angles between dislocation lines using python. Since neural network require large number of training dataset so we created it by changing pixel values in images to 0,50 and 100 such that we can generate approximately 257000 images. Data for 10 classes was classified using Convolution Neural Network where we split data into 70:15:15 between training:validation:test set. By manipulating hyper-parameters we avoided over-fitting as validation and training accuracy are of same order. Over-fitting occur when difference between training and validation accuracy is high which is not seen in our case. Model was correctly predicting the test set which was not used in training. We can use K-fold cross validation since we have limited number of data set but that will increase the duration of computation. Accuracy of our model was above 98 percent after 100 epochs and further epochs were not increased because that will create over-fitting problem. Since our model has different size of images with respect to dislocation and still it is classifying, we can say that classification does not depend upon size generated and not even on pixel count.

Chapter 6

Future Work

1. We have successfully trained and validated CNN model to perform classification. We can use this model to perform classification on a particular material going through uniaxial stress and taking data at a particular interval using DD simulation.

2. Once data is collected and model is trained, we can successfully tell by looking at dislocation structure that material has been subjected to a particular amount of strain as our model can classify the dislocation structure at various level of strain.

3. We can work it into a Image Regression problem for further implementing structure property correlation using machine learning.

4. Instead of going through Images, we can directly look into the data and try to come up with a machine learning model that will predict the correct outcome.

References

[1] Mahmoud parsian. https://github.com/mahmoudparsian/data-algorithms-book/tree/master/src/main/java/org/dataalgorithms/machinelearning/.

[2] Ronald Van Loon. https://bigdata-madesimple.com/machine-learning-explained-understanding-supervised-unsupervised-and-reinforcement-learning/.

[3] Deepika Garg. https://www.digitalvidya.com/blog/bias-variance-tradeoff/.

[4] http://inspirehep.net/record/1094489/plots. *Microcontinuum field theories: I. Foundations and solids.*

[5] https://medium.com/datadriveninvestor/k-fold-cross-validation-6b8518070833/.

[6] Arnav Oberoi. https://towardsdatascience.com/multiple-linear-regression-beginners-guide-5b602d716aa3.

[7] Tony Yiu. https://towardsdatascience.com/understanding-neural-networks-19020b758230/.

[8] https://www.superdatascience.com/blogs/convolutional-neural-networks-cnn-step-4-full-connection/.

[9] Karpathy Soumith. http://cs231n.github.io/convolutional-networks//.

[10] PhD Jason Brownlee. https://machinelearningmastery.com/keras-functional-api-deep-learning//.

[11] Charlotte Pelletier, Geoffrey Webb, and François Petitjean. Temporal convolutional neural network for the classification of satellite image time series. *Remote Sensing*, 11:523, 03 2019.

[12] A. ALANKAR. https://www.me.iitb.ac.in/ alankar/index.html/.

[13] Shafique MA Khan, Hussein M Zbib, and Darcy A Hughes. Modeling planar dislocation boundaries using multi-scale dislocation dynamics plasticity. *International Journal of Plasticity*, 20(6):1059–1092, 2004.

[14] Jonathan Schmidt, Mário RG Marques, Silvana Botti, and Miguel AL Marques. Recent advances and applications of machine learning in solid-state materials science. *npj Computational Materials*, 5(1):1–36, 2019.

[15] John P Hirth and Ladislas Kubin. *Dislocations in solids*, volume 15. Elsevier, 2009.

[16] James D Livingston. The density and distribution of dislocations in deformed copper crystals. *Acta metallurgica*, 10(3):229–239, 1962.

[17] M Yu Gutkin and EC Aifantis. Edge dislocation in gradient elasticity. *Scripta Materialia*, 36(1), 1997.

[18] Benoît Devincre, Ladislas Kubin, and Thierry Hoc. Physical analyses of crystal plasticity by dd simulations. *Scripta Materialia*, 54(5):741–746, 2006.

[19] CASPER HANSEN. https://mlfromscratch.com/machine-learning introduction-8-linear-regression-and-logistic-regression/.

[20] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars. *ArXiv*, abs/1604.07316, 2016.

[21] S. Lathuiliére, P. Mesejo, X. Alameda-Pineda, and R. Horaud. A comprehensive analysis of deep regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2019.

Acknowledgments

I would first like to thank my guide Prof. Alankar Alankar of the mechanical department for constantly guiding me and giving his valuable time. The door to Prof. A. Alankar office was always open whenever I ran into a trouble spot or had a question about my research topic. Even during COVID-19 lock-down he was constantly in touch with me and really push me hard to my limits. I would also like to thank Mr. Saurabh Kasodariya for his support. At the end I would like to thank IIT Bombay for providing me this platform to showcase my skills.

Date: ____________ SUBHANSHU RANJAN TIWARI