



Impact Location Prediction using Neural Networks

Group 4

Group: 4

Names: Renuka Misal, Soji Jacob, Rushikesh Shende, Soham Bhute, Pratik Tatipamula

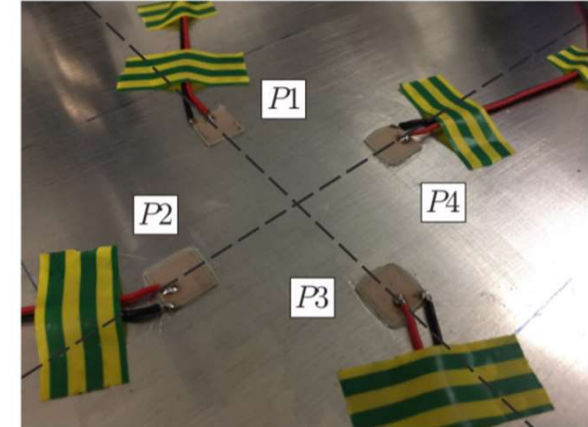
Date: 19.01.2022

Team Introduction

- Renuka Misal: - CNN Data Map, ANN Data Processing
- Soji Jacob: - Numerical Data Preprocessing, CNN Data Map & Image Map
- Rushikesh Shende: - ANN & KNN Architectures, PPT
- Soham Bhute: - Data Extraction, PPT
- Pratik Tatipamula: - Experimental Data Processing

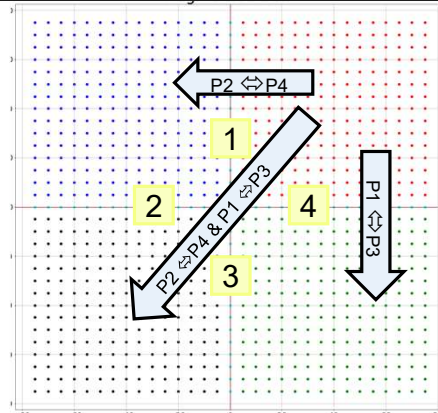
Motivation and Aim

- Motivation
 - Bad weather conditions during an air flight (E.g. Hailstorms) can cause extensive damage to aircraft structure and compromise its structural integrity
 - Predicting the sudden impact of hailstorms is a major problem in the aviation industry
- Problem Statement
 - Aim of the experiment is to predict the impact location and/or its characteristics in a controlled experimental setup.
 - The output of the neural network should predict the location of impact of the ball



Data Pre Processing

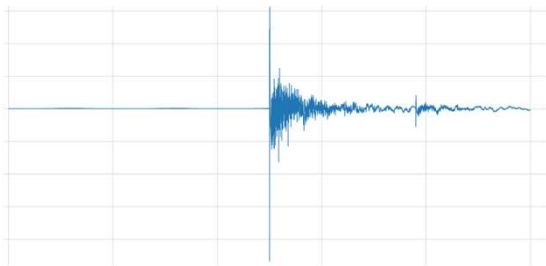
1. Data Augmentation



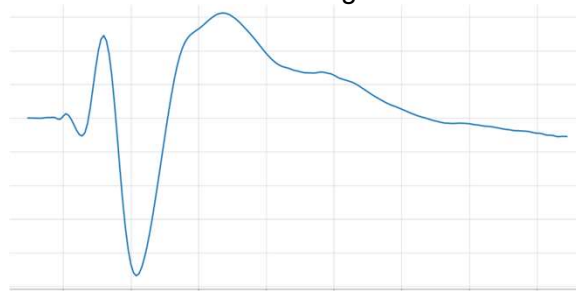
Data Points after Augmentation: 925

2. Data Cutting

Before Cutting

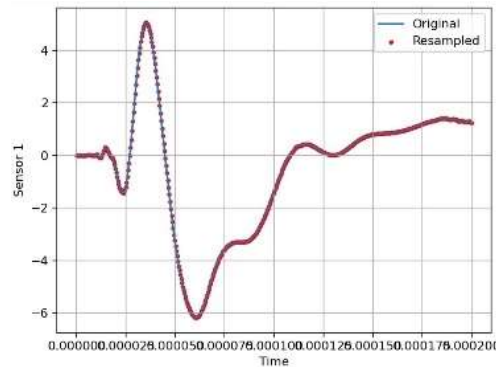


After Cutting

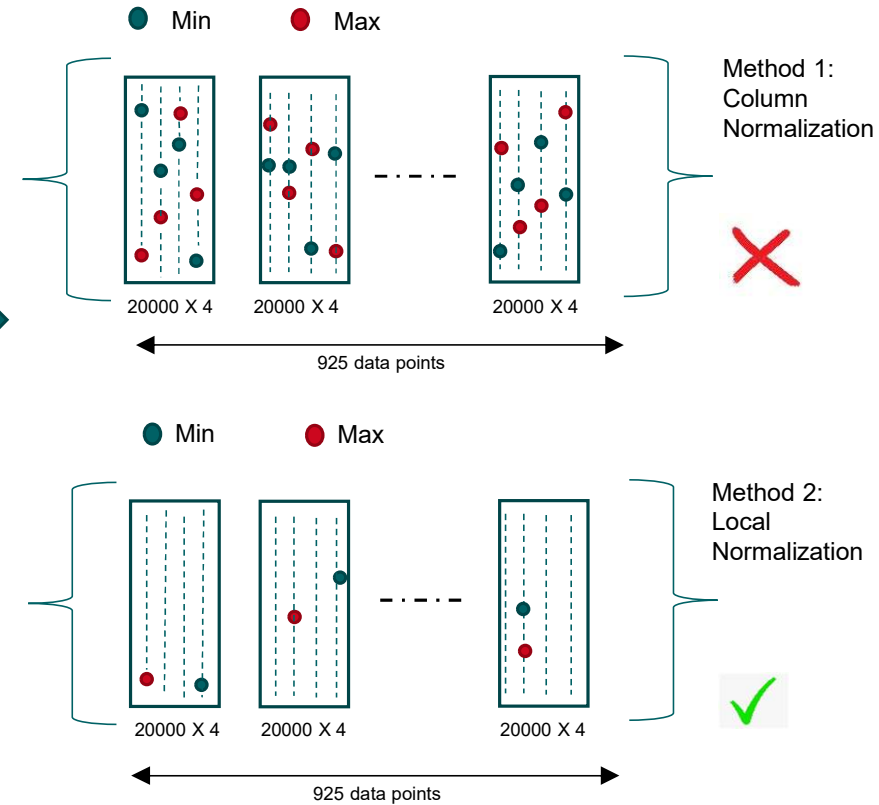


3. Resampling

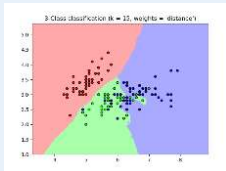
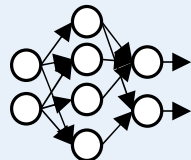
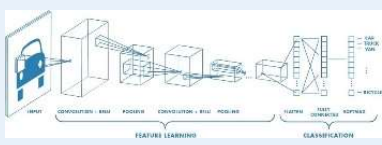
Number of Resampling Points: 500



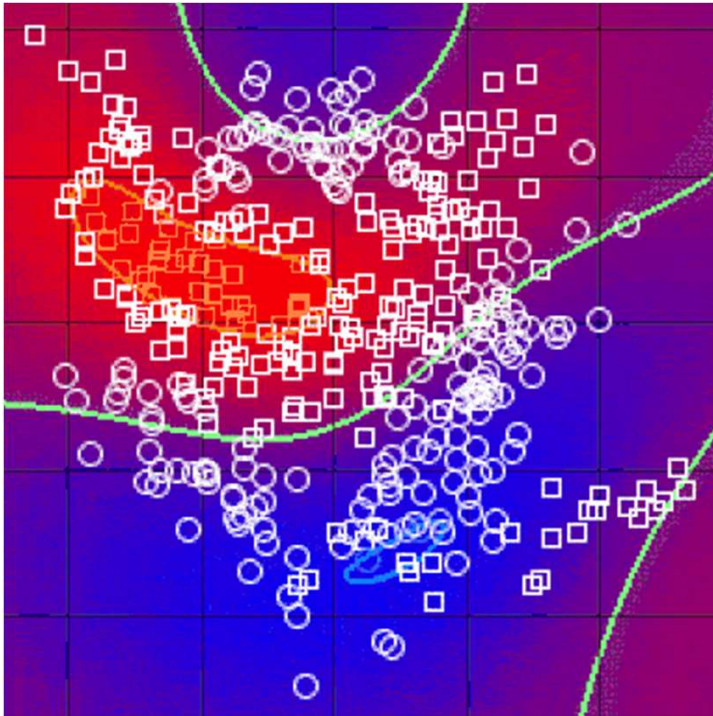
4. Normalizing



Model Comparison

Metric \ Models	KNN	ANN	CNN
			
R2 Score	0.9846	0.8238	0.9976
MSE	0.0098	0.019	0.00082
Complexity	Low	Medium	High
Trainable Parameters	-	2,184,958	97,714

K-Nearest Neighbor Regression

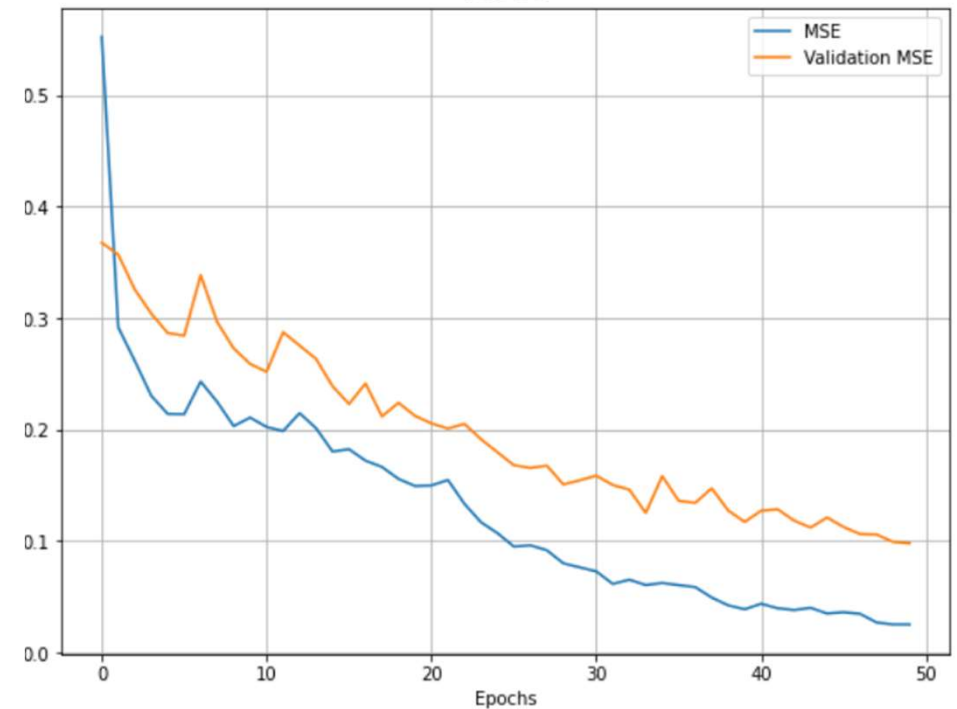
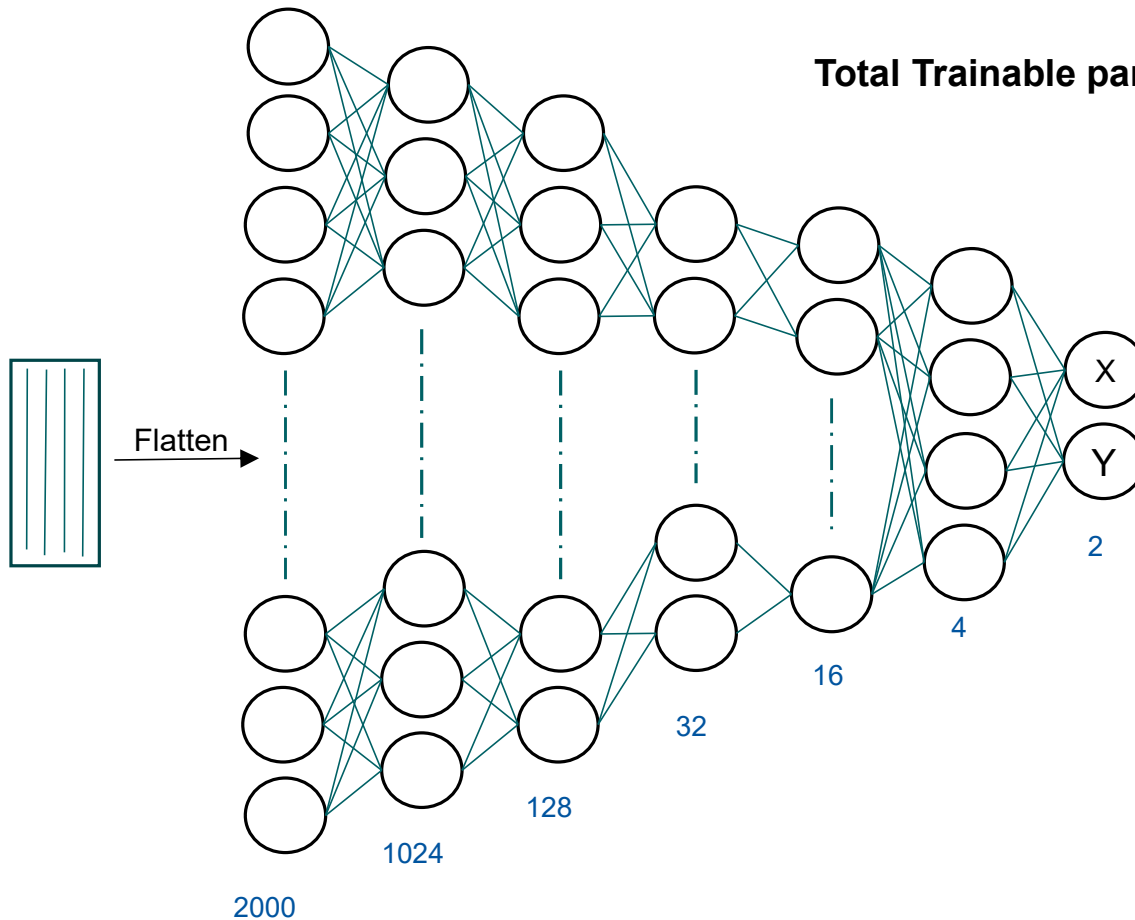


Hyperparameters	
Nearest Neighbours	4

Results	
R2 Score	0.9846

Coordinate Predictions		
1	240	247.5
2	206.25	261.25
3	250	255
4	236.25	277.5
5	271.25	257.5
9	310	257.5
13	272.5	280
24	250	245
25	233.75	222.5
26	271.25	287.5
35	212.5	246.25
36	250	260
42	298.75	240
61	215	242.5
62	240	247.5
64	275	295
69	260	250
85	225	302.5

Artificial Neural Network (MLP) – Architecture



Artificial Neural Network (MLP) – Results & Predictions

Hyperparameters	
Initial Learning Rate	0.0015
Decay Steps	100
Decay Rate	0.9
L2 Regularizer	0.01
Activation Function	tanh
Optimizer	Adam
Loss Metric	MSE
Batch Size	32
Train Test Split	0.2
K-Folds	8

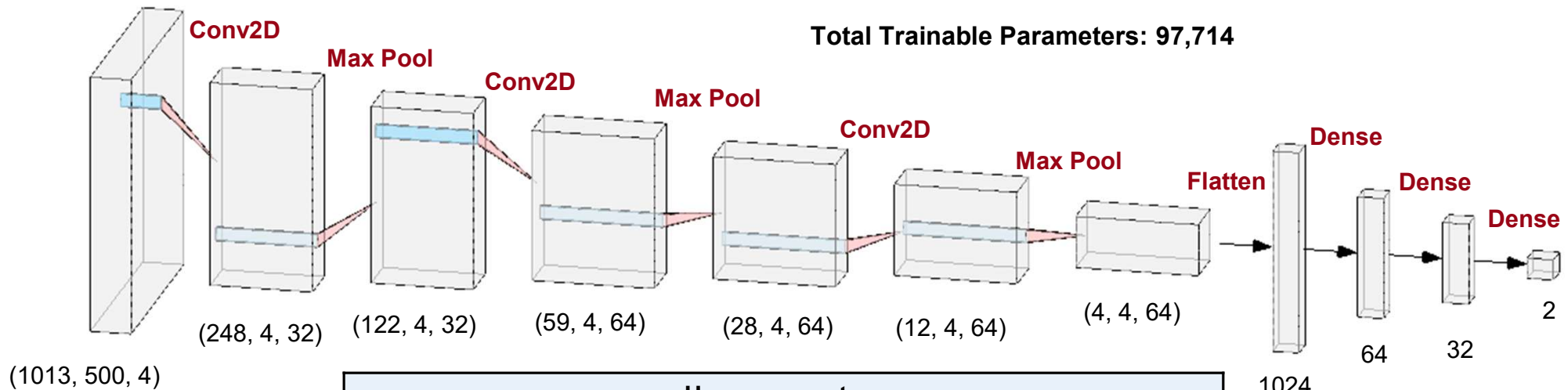
Results	
Cross-validated MSE	0.0636
Holdout Set MSE	0.019
R2 Score	0.8238

- Processor: Intel i5 10th Gen @ 1.00GHz, 8GB DDR4 RAM
- Total Training Time = 152.03s
- Average Training Time per K-Fold = 19s

Coordinate Predictions		
1	214.81	249.76
2	255.53	246.77
3	314.93	264.39
4	283.64	277.05
5	268.92	272.29
9	271.68	241.84
13	278.32	292.54
24	283.19	241.07
25	240.4	212.21
26	240.64	281.95
35	234.79	243.44
36	231.2	264.25
42	270.39	251.2
61	237.47	220.75
62	242.64	248.27
64	238.55	290.91
69	249.58	270.28
85	273.51	286.14

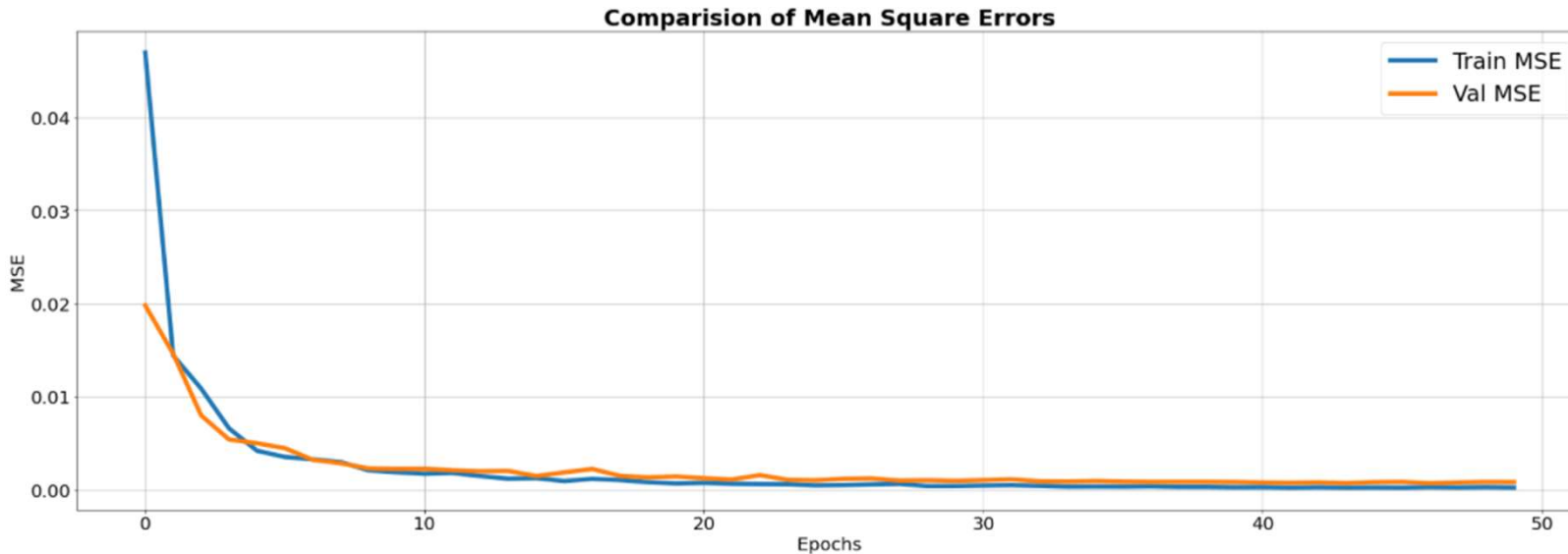
Convolution Neural Network – Architecture

Total Trainable Parameters: 97,714



Hyperparameters			
Initial Learning Rate	0.001	Number of Batches	100
Train-Val-Test Split	70-20-10	Activation	tanh
Decay Steps	1000	Conv Filter Size	(5,1)
Decay Rate	0.7	Conv Stride	(2,1)
Epochs	50	Pool Filter Size	(5,1)
Regularizer, Dropout	Iterated	Pool Stride	(2,1)
Optimizer	Adam	Initializer	GlorotNormal
Loss Metric	MSE	No. of Conv Filters	(32, 64, 64)

Convolution Neural Network – Results



- Processor: Intel i5 10th Gen @ 1.00GHz, 8GB DDR4 RAM
- Total Training Time = 35.0733 s

Results			
Validation MSE	0.00082	RMSE on Experimental Validation Data	1.9959 mm
Training MSE	0.00022	RMSE on Test Data	2.0927 mm
Max Deviation in Coordinates for whole data set	10.1696 mm	RMSE on Whole Data	1.4789 mm
Max Deviation in Coordinates for test data set	8.754 mm	R2 Score on Test Data	0.9976

Convolution Neural Network – Comparison

Effect of including Experimental Validation Data in Training

	When Included in Training	When excluded from Training
RMSE on Experimental Validation Data	1.9959 mm	10.552 mm
R2 Score Experimental Validation Data	0.9982	0.8207

- Since the given Numerical Data is close to ideal conditions, the model would fail to predict accurately the Experimental Data
- Inclusion of Experimental Data makes the model more robust.
- To increase the number of Experimental Datasets for training, augmentation for the given Experimental Validation Dataset was also done.

Convolution Neural Network – Comparison

Coordinate Predictions on Cut and Uncut Numerical Data

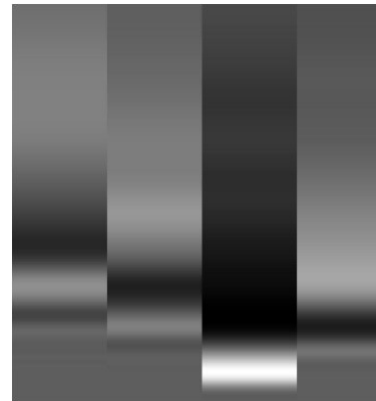
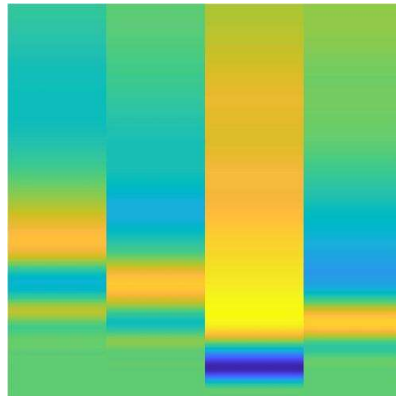
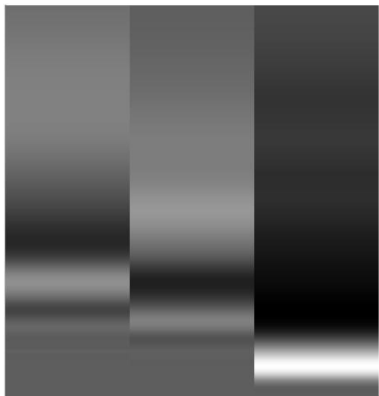
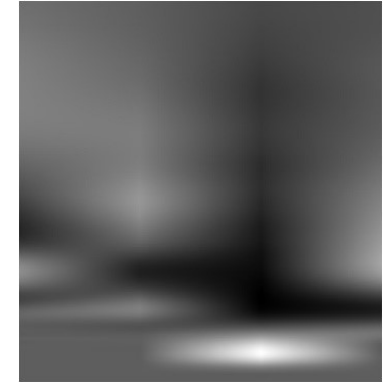
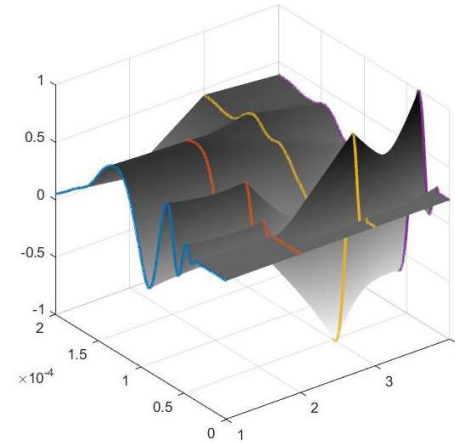
Coordinate Predictions on Cut Numerical Data		
1	243.8124	242.9785
2	233.8909	239.9393
3	256.2994	254.5943
4	249.7443	259.3821
5	248.8161	255.3065
9	271.0378	229.4757
13	294.1216	286.6793
24	252.4848	241.9401
25	220.4420	201.5592
26	250.4021	313.7439
35	228.7617	229.2879
36	245.9761	262.0672
42	267.9358	232.1219
61	203.3516	221.8577
62	241.2504	250.5545
64	261.8284	291.8987
69	262.4064	253.5770
85	229.6025	296.3911

Coordinate Predictions on Uncut Numerical Data		
1	233.1345	250.6272
2	232.9496	233.5038
3	256.1684	254.3674
4	250.7997	257.0374
5	241.5154	246.6749
9	268.6111	236.7853
13	293.191	284.1542
24	251.5374	243.3652
25	224.0412	209.7135
26	252.7176	307.7839
35	226.5766	232.1158
36	244.1192	257.5513
42	266.9145	238.9055
61	210.6283	223.6966
62	240.5161	249.8425
64	269.0522	291.9933
69	262.6551	254.3559
85	230.5353	291.1704

- Other result metrics as mentioned on slide 9 are of the same order for both these cases, hence our network is robust in the sense that it **would not require cutting of Numerical Data**

Convolution Neural Network – Image Map Trials

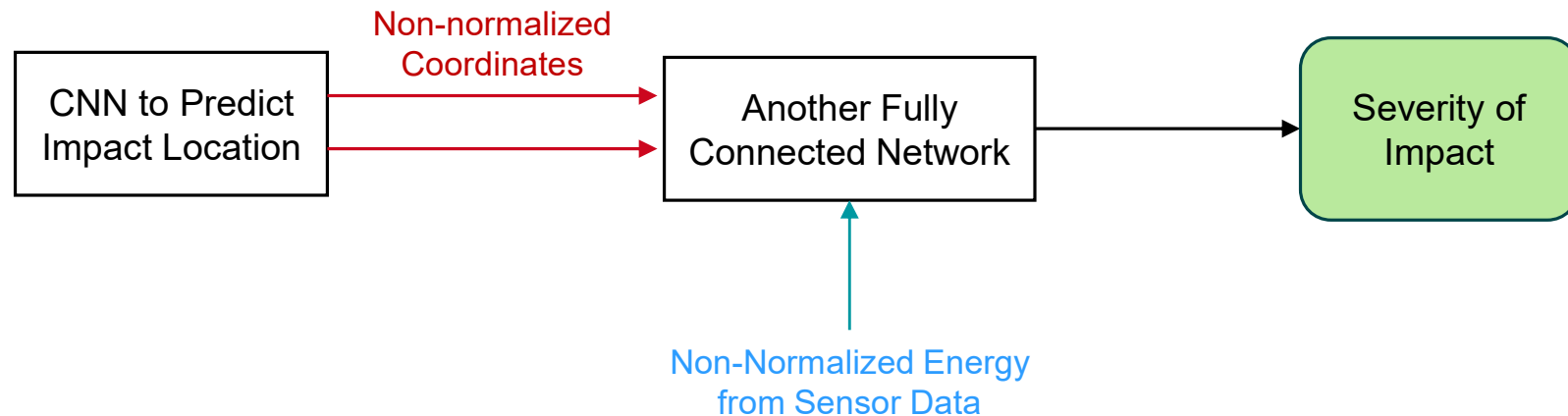
- Idea of the image map:
 - Generate a 3D grayscale surface from the 4 sensor data (done in MATLAB)
 - Capture and save the top view image of the surface
 - Importing these images for input to CNN in Python



- Generating an image map, and training and optimization of CNN is computationally expensive and time consuming!

Future Prospects

- Using advanced architectures implementing Ensemble Learning Methods to predict the impact coordinates
- Impact detection and energy evaluation to classify the severity of impact. The combination of energy at the sensor locations along with the non-linear energy damping relationship can be given as an input to another network for predicting the impact energy



- Image maps could be suitable for complex network of piezoelectric sensors

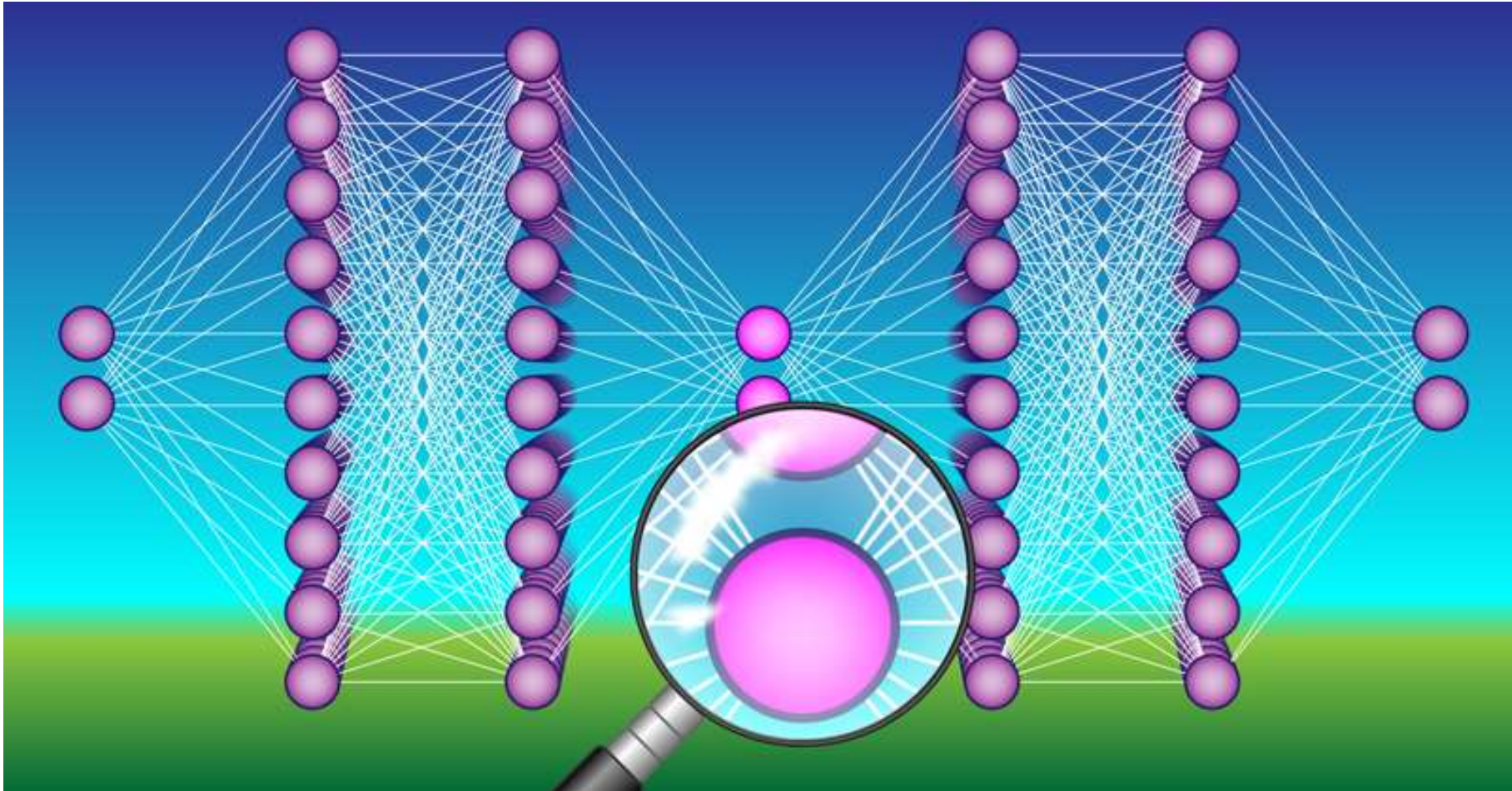
Discussion and Conclusion

- Applying Regularizer and Dropout to simple network architectures results in underfitting & unreasonable output values. The 70-20-10 (Train-Validate-Test) split of the data helps identify under/over-fitting
- Choice of weight initializer (E.g.: Glorot, He, Random, Lecun) has an important impact on the output.
- Activation Functions (tanh, ReLU, LeakyReLU, eLU, SELU, GELU) change the output considerably
 - ReLU was faced a problem called “Dying ReLU”, in which several neurons did not respond anymore resulting in a partially passive network. LeakyReLU performs better than ReLU
 - Regardless, tanh is used here since it performed better than LeakyReLU across several iterations
- Accuracy metric for regression problem has no physical interpretation, hence should not be used to evaluate the model
- Hyperparameter tuning plays an important role for achieving less deviation in output values. Accessibility to a powerful compute cluster (RWTH CLAIX 18) helps in iterating over various combinations of hyperparameters across different models
- Progressing from the simplest (linear regression) to complex architecture streamlined our architecture design process

References

1. Prof. Dr. -Ing. Bernd Markert; Dr. Franz Bamer; Dr. sc. ETH, Dipl.-Ing. Georg Kocur; Denny Thaler (M.Sc.), Computational Intelligence in Engineering Lecture Notes, IAM, RWTH Aachen University
2. Prof. Dr. Bastian Leibe, Machine Learning Lecture Notes, Visual Computing Institute, RWTH Aachen University
3. Iuliana T.; Hailing Fu.; Sharif Khodaei Z., A Convolutional Neural Network for Impact Detection and Characterization of Complex Composite Structures, *Sensors* 2019, 19, 4933; doi:10.3390/s19224933
4. *RWTH Compute Cluster*. Retrieved from <https://help.itc.rwth-aachen.de/en/service/rhr4fjjutttf/>
5. *TensorFlow*. Retrieved from <https://www.tensorflow.org/>
6. *Keras*. Retrieved from <https://keras.io/>

Any Questions?



Thank You

