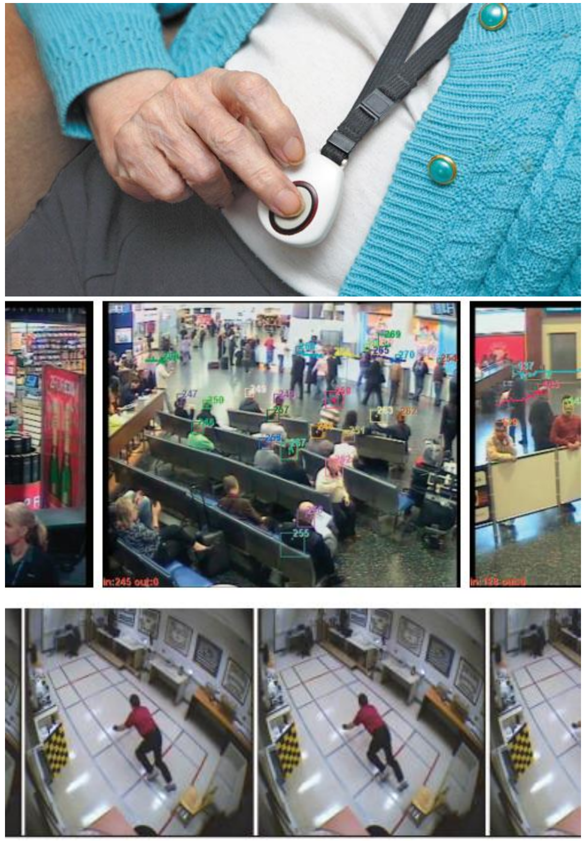
Privacy-ensuring fall detection using mmWave technology

Abanob Tawfik

z5075490

Recap

- One of the biggest causes of fatal and non-fatal injuries amongst people
- According to the World Health Organisation
  - Falls are the second leading cause of accidental or unintentional injury deaths worldwide
  - Each year approximately 646,000 people die from falls globally
  - Adults over the age over 65 tend to suffer the greatest number of fatal falls
  - 37.3 million falls occur each year that require medical attention



Problems in Fall Detection

- Phone and sensor based fall detection methods can be problematic
- Camera-based monitoring requires data for training which is often unacquirable as:
  - Falls are unlikely events
  - It is unethical to ask elderly people to fall
  - Training data from healthier younger people cannot simulate a real natural fall
- There is still no standard for fall detection that performs consistently well
- Camera-based monitoring raises privacy concerns

Aim For Thesis B

	Week 1 Term 2	Week 2 Term 2	Week 3 Term 2	Week 4 Term 2	Week 5 Term 2	Week 6 Term 2	Week 7 Term 2	Week 8 Term 2	Week 9 Term 2	Week 10 Term 2	Week 11 Term 2
Meet with team for weekly standup											
Develop a more complex generative model											
Develop Fall Detection Logic											
Compute confusion matrix											
Analyse result, determine how to improve the model											
Thesis B Demonstration											
Thesis B Report											

Progress During Thesis B

- Progress for thesis B began early
- First task was to setup a codebase in katana
- After code base was set up in katana issues had arisen
  - Data storage (OneDrive)

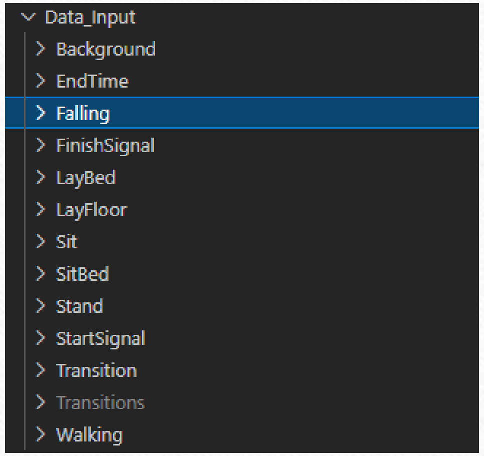
Progress During Thesis B

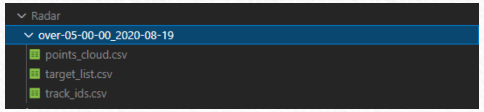
- After successfully setting up katana I created a manual guide for others to use
  - Set up katana
  - Integrate one drive
- I had also gotten a shared drive to store all the data for the team to use
- Began coding the complex model

Data Preprocessing

- Data pre-processing had been problematic since mmWave data was stored as a csv dump in raw point cloud form
- Data had no structure, making the model unable to distinguish between activities
- Developed a data cleaning script to link training data to activities and provide structure to the data

Start_Time	Location_Class	Activity_Class
00:00.98334	Room	StartSignal
00:05.15000	Room	Walking
00:11.11300	Room	Stand
00:13.61362	Room	Walking
00:20.49994	Room	Stand
00:23.66667	Room	Walking
00:30.07923	Room	Stand
00:33.69551	Room	Walking
00:39.45000	Room	Stand
00:45.41667	Room	Walking
00:51.16137	Room	Stand
00:56.78334	Room	Walking
01:08.15000	Room	Stand
01:12.28198	Room	Walking
01:14.85954	Room	FinishSignal
01:17.78335	Room	EndTime

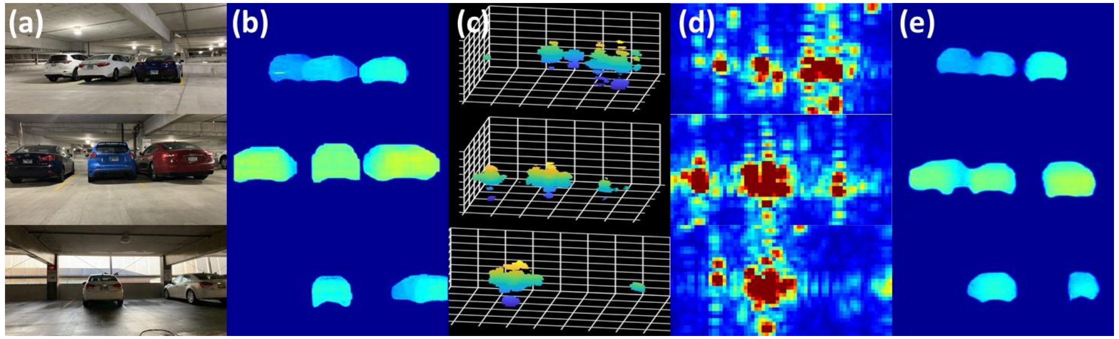




Data Preprocessing

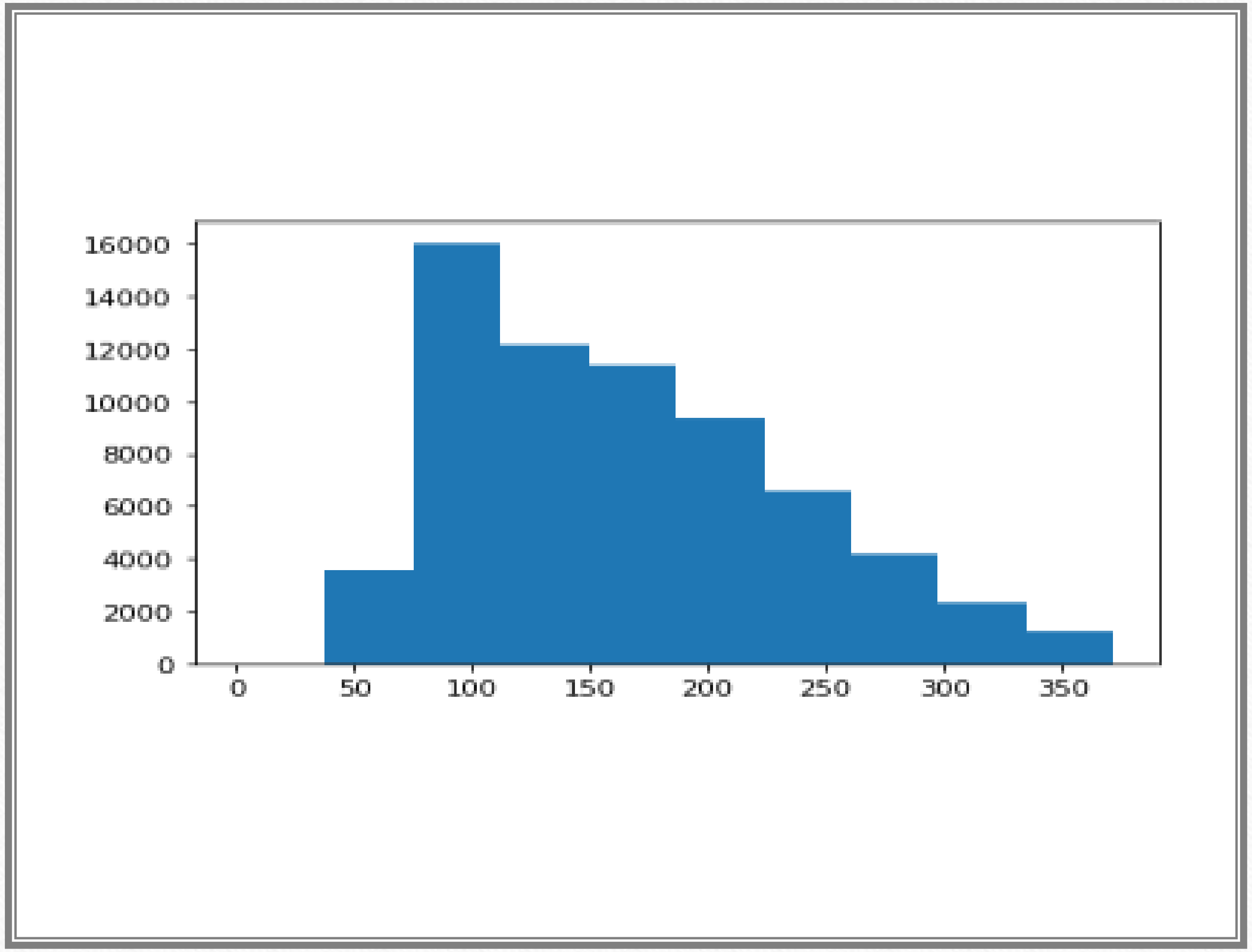
- Data is initially a point cloud of mmWave data, where one frame contains a random amount of points
- Data needs to be grouped into frames
- Activities are then defined by their sequence of frames
- Samples can be categorised into normal and abnormal data


```
timestamp,frame_number,x,y,z,doppler,snr,range,azimuth,elevation
2020-08-24T11:01:47.039506,1.0,-0.1080533307908504,1.197349239511554,0.5082883046921254,-0.1310400034999475,2.879999935626984,1.3052500619960483,-0.08999999798834324,0.3999999910593033
2020-08-24T11:01:47.039506,1.0,-0.0692182620928221,1.152253029760344,0.4610541555747438,-0.1310400034999475,3.359999924898148,1.243000059039332,-0.059999998658895486,0.3799999915063381
2020-08-24T11:01:47.039506,1.0,-0.07208962188715548,1.2000515864203756,0.5082883046921254,-0.1310400034999475,3.1199999302625656,1.3052500619960483,-0.059999998658895486,0.3999999910593
2020-08-24T11:01:47.039506,1.0,-0.03635873212307607,1.2115941553950245,0.4841439554013953,-0.1310400034999475,3.2399999275803566,1.3052500619960483,-0.029999999329447743,0.379999991506
2020-08-24T11:01:47.039506,1.0,-0.03809275324903776,1.2693775196343198,0.5072337552280467,0.1310400034999475,1.9999999552965164,1.3675000649527649,-0.029999999329447743,0.37999999150633
2020-08-24T11:01:47.039506,1.0,-0.012260958326089314,1.2260549898816675,0.4475673748474947,-0.1310400034999475,3.1199999302625656,1.3052500619960483,-0.009999999776482582,0.349999992176
2020-08-24T11:01:47.039506,1.0,-0.02971375530698232,2.9712765506045296,0.2681526582913460 5,0.1310400034999475,1.1199999749660492,2.9835001417086464,-0.009999999776482582,0.0899999979883
2020-08-24T11:01:47.039506,1.0,0.0288563305290406,1.4426241780387234,0.5763176944684297,-0.1310400034999475,0.8799999803304672,1.553750073799165,0.019999999552965164,0.3799999915063381
2020-08-24T11:01:47.039506,1.0,0.029766707647556474,1.4881369656394374,0.6293000577532846,-6.2834801678254735,0.6399999856948853,1.6160000767558813,0.019999999552965164,0.39999999105930
2020-08-24T11:01:47.039506,1.0,0.059519901509975534,2.9755983321012023,0.2086744902864175,0.1310400034999475,1.3599999696016312,2.9835001417086464,0.019999999552965164,0.0699999984353780
2020-08-24T11:01:47.039506,1.0,0.0606594383530973 1,3.032567578416113,0.2737251284820829,0.1310400034999475,1.1199999749660492,3.0455001446534884,0.019999999552965164,0.08999999798834324
2020-08-24T11:01:47.039506,1.0,0.07211557901232094,1.4411094858258415,0.5763176944684297,-0.1310400034999475,0.9999999776482582,1.553750073799165,0.04999999888241291,0.3799999915063381
2020-08-24T11:01:47.039506,1.0,0.07439072529106447,1.4865744869926285,0.6293000577532846,-6.2834801678254735,0.7599999830126762,1.6160000767558813,0.04999999888241291,0.3999999910593033
2020-08-24T11:01:47.039506,1.0,0.07658765711946827,1.5304765029091132,0.6843239984651877,-0.1310400034999475,0.6399999856948853,1.6782500797125977,0.04999999888241291,0.4199999906122685
2020-08-24T11:01:47.039506,1.0,0.1190159958539994,2.973812913653126,0.2086744902864175,0.1310400034999475,1.5199999660253525,2.9835001417086464,0.03999999910593033,0.06999999843537807
2020-08-24T11:01:47.039506,1.0,0.1212946137407197,3.0307479772820725,0.2737251284820829,0.1310400034999475,1.2399999722838402,3.0455001446534884,0.03999999910593033,0.08999999798834324
2020-08-24T11:01:47.039506,1.0,0.09689431006668664,1.3819428543843812,0.553320624761564,-0.1310400034999475,1.6399999633431437,1.4917500708543232,0.06999999843537807,0.3799999915063381
2020-08-24T11:01:47.039506,1.0,0.1009214240094616,1.4393790581529962,0.5763176944684297,-0.1310400034999475,1.2399999722838402,1.553750073799165,0.06999999843537807,0.3799999915063381
2020-08-24T11:01:47.039506,1.0,0.10496477631490908,1.4970468595174524,0.5994074942950812,-6.2834801678254735,0.8799999803304672,1.6160000767558813,0.06999999843537807,0.3799999915063381
2020-08-24T11:01:47.039506,1.0,0.10811560114814253,1.5419851005834868,0.6535413502007733,-0.1310400034999475,0.7599999830126762,1.6782500797125977,0.06999999843537807,0.3999999910593033
2020-08-24T11:01:47.039506,1.0,0.20829898889138665,2.9708380097430718,0.17890261783485562,0.1310400034999475,1.5199999660253525,2.9835001417086464,0.06999999843537807,0.0599999986588954
2020-08-24T11:01:47.039506,1.0,0.21248928319877525,3.0306015528435672,0.21301094693054612,0.1310400034999475,1.3599999696016312,3.0455001446534884,0.06999999843537807,0.0699999984353780
2020-08-24T11:01:47.039506,1.0,0.21697374986666892,3.09456069206604,0.18635314582747528,0.1310400034999475,0.8799999803304672,3.1077501476102043,0.06999999843537807,0.059999998658895486
2020-08-24T11:01:47.039506,1.0,0.14405090664223474,1.4357041976566949,0.5763176944684297,-0.1310400034999475,1.3599999696016312,1.553750073799165,0.09999999776482582,0.3799999915063381
2020-08-24T11:01:47.039506,1.0,0.1498222140845383,1.4932247680857402,0.5994074942950812,-0.6546400174847804,0.9999999776482582,1.6160000767558813,0.09999999776482582,0.3799999915063381
2020-08-24T11:01:47.039506,1.0,0.15559352152684186,1.5507453385147858,0.6224972941217327,-0.1310400034999475,0.8799999803304672,1.6782500797125977,0.09999999776482582,0.3799999915063381
2020-08-24T11:01:47.039506,1.0,0.26767012831304626,2.9660781600147064,0.17890261783485562,0.1310400034999475,1.5199999660253525,2.9835001417086464,0.08999999798834324,0.0599999986588954
2020-08-24T11:01:47.039506,1.0,0.2732325710666608,3.027716117420743,0.18262038633016686,0.1310400034999475,1.3599999696016312,3.0455001446534884,0.08999999798834324,0.059999998658895486
2020-08-24T11:01:47.039506,1.0,0.18920756388976026,1.4472345850054138,0.5327774822212564,0.1310400034999475,1.5199999660253525,1.553750073799165,0.12999999709427354,0.349999921768904
2020-08-24T11:01:47.039506,1.0,0.19678804392331625,1.50521711302896 1,0.554122871291746,0.1310400034999475,0.9999999776482582,1.6160000767558813,0.12999999709427354,0.349999921768904
2020-08-24T11:01:47.039506,1.0,0.2020388907724597,1.5453804500775516,0.6224972941217327,-0.1310400034999475,0.8799999803304672,1.6782500797125977,0.12999999709427354,0.3799999915063381
2020-08-24T11:01:47.039506,1.0,0.3568756884703017,2.9596753869926733,0.1193081815473341,-0.1310400034999475,1.3599999696016312,2.9835001417086464,0.119999997317791,0.03999999910593033
2020-08-24T11:01:47.039506,1.0,0.3642919085759355,3.0211802886161165,0.1217875203292797,-0.1310400034999475,1.3599999696016312,3.0455001446534884,0.119999997317791,0.03999999910593033
2020-08-24T11:01:47.039506,1.0,0.2181124850857934,1.4431612488279295,0.5327774822212564,0.1310400034999475,1.5199999660253525,1.553750073799165,0.14999999664723873,0.349999921768904
2020-08-24T11:01:47.039506,1.0,0.2284616848698453,1.5116376777625105,0.5236535474817391,0.1310400034999475,0.9999999776482582,1.6160000767558813,0.14999999664723873,0.3299999926239252
2020-08-24T11:01:47.039506,1.0,0.2355895595142286 6,1.5587999136575852,0.5754682603622356,-0.1310400034999475,0.8799999803304672,1.6782500797125977,0.14999999664723873,0.349999921768904
2020-08-24T11:01:47.039506,1.0,0.4159938759770523,2.951946433887473,0.1193081815473341,-0.1310400034999475,1.2399999722838402,2.9835001417086464,0.13999999687075615,0.03999999910593033
2020-08-24T11:01:47.039506,1.0,0.4246386288882563,3.0132907204304664,0.1217875203292797,-0.1310400034999475,1.2399999722838402,3.0455001446534884,0.13999999687075615,0.03999999910593033
2020-08-24T11:01:47.039506,1.0,0.26315795685255383,1.4461649939504906,0.5034818684404407,0.1310400034999475,1.5199999660253525,1.553750073799165,0.17999999597668648,0.3299999926239252
2020-08-24T11:01:47.039506,1.0,0.27370121208284914,1.504104669492513,0.5236535474817391,0.1310400034999475,1.1199999749660492,1.6160000767558813,0.17999999597668648,0.3299999926239252
2020-08-24T11:01:47.039506,1.0,0.28424446731314457,1.5620443450345365,0.5438252265230376,0.1310400034999475,0.8799999803304672,1.6782500797125977,0.17999999597668648,0.3299999926239252
2020-08-24T11:01:47.039506,1.0,0.3208137596119784,1.5051613390621656,0.4929747692440568,0.1310400034999475,0.9999999776482582,1.6160000767558813,0.20999999530613425,0.30999999307096004
2020-08-24T11:01:47.039506,1.0,3.7327639344237484,2.5580874611507873,-0.8234465438774342,0.0,74.11999834328891,4.595002184645 27,0.9699999783188104,-0.17999999597668648
```



Issue Arises

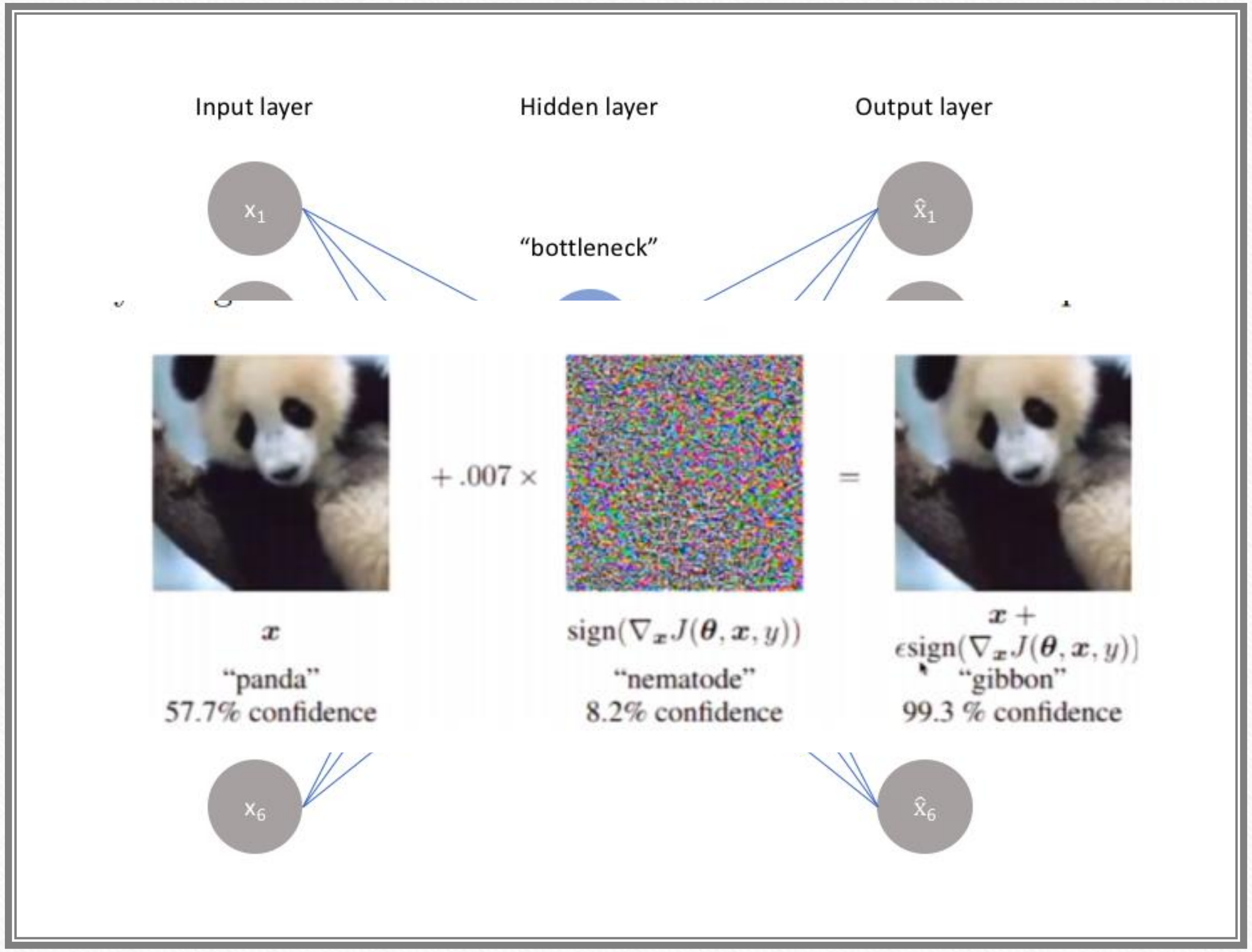
- Due to the real-time requirement of the project at hand, data input became problematic
- Point cloud frames have a random number of points due to nature of point cloud data
- Machine learning models work with fixed input size



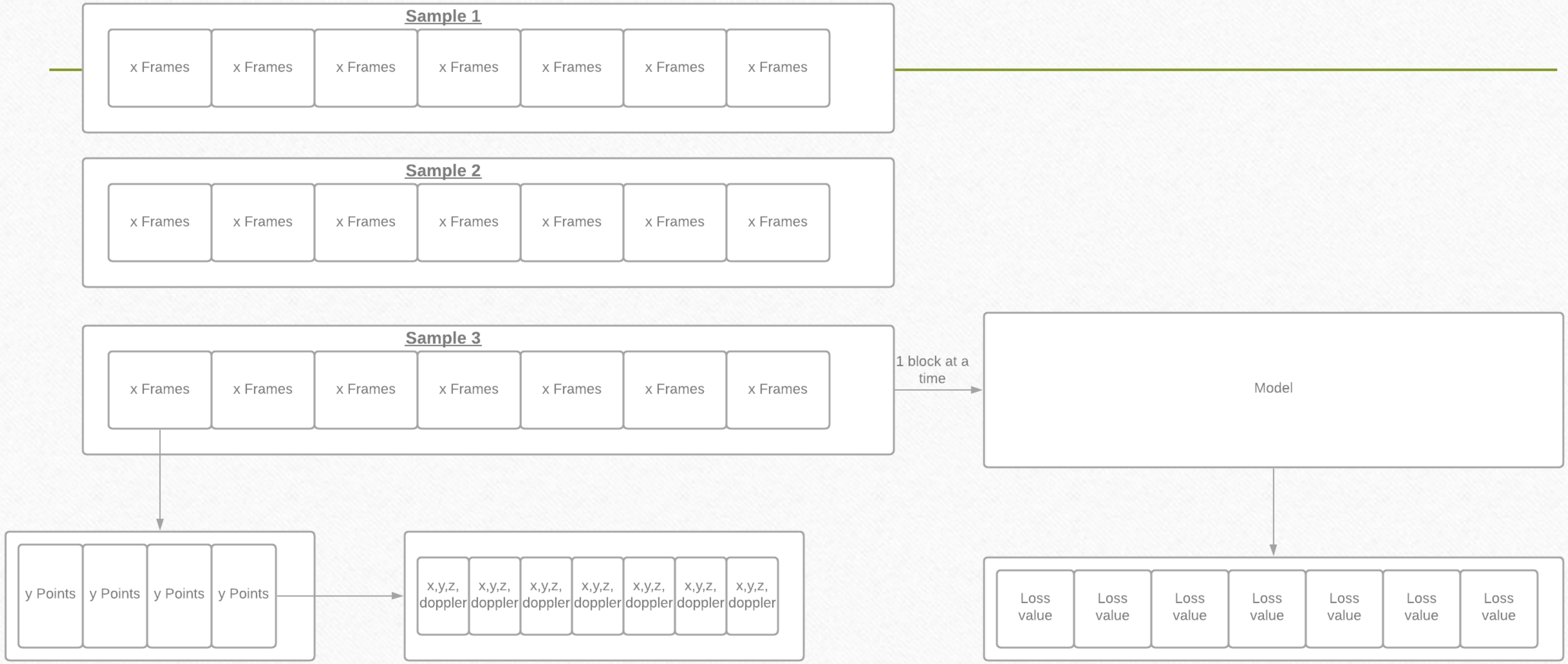
Solution

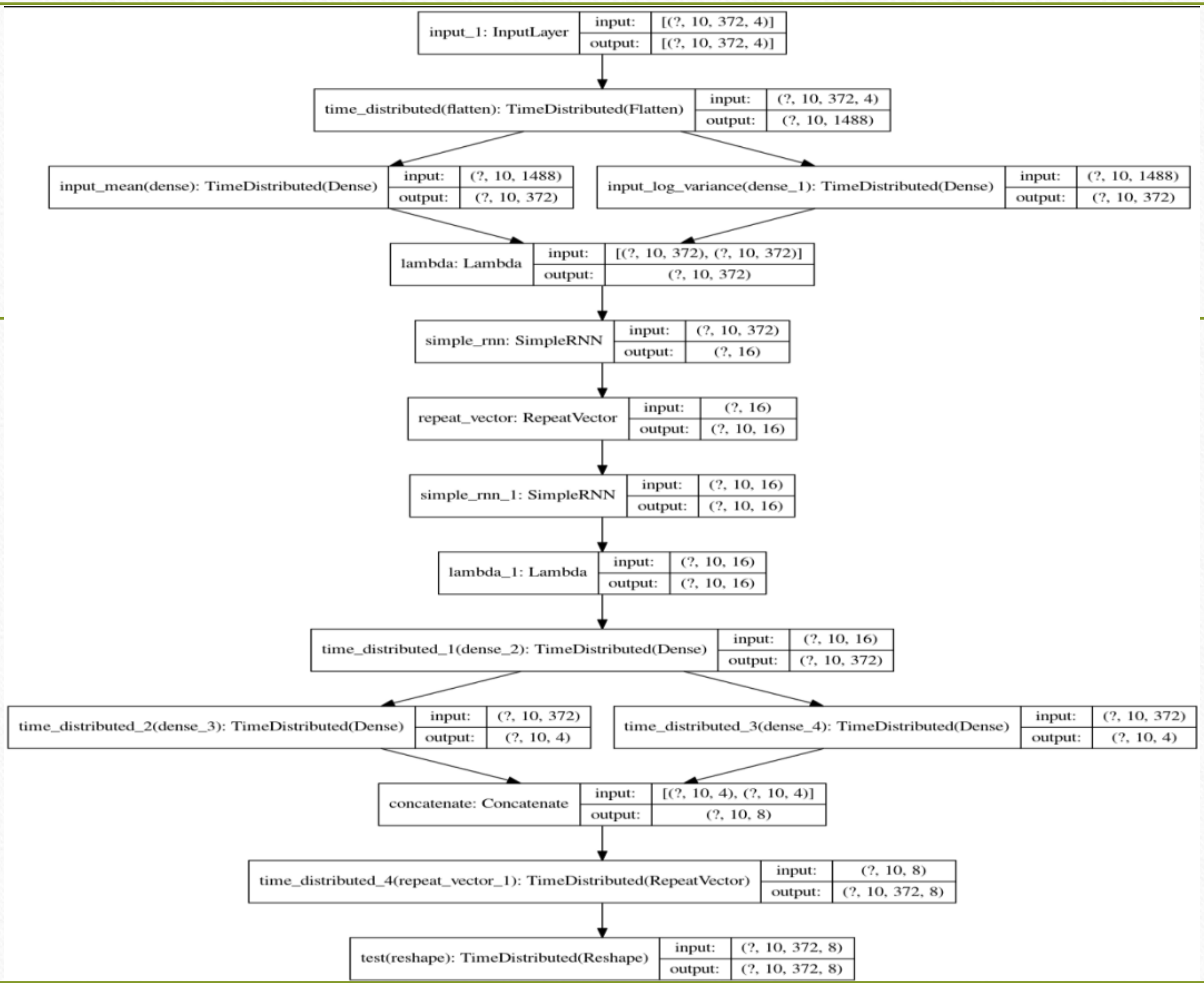
- Frame statistics were gathered
  - Training set
  - Testing set
- Using this result, all input was oversampled to match the maximum possible frame size

Autoencoders

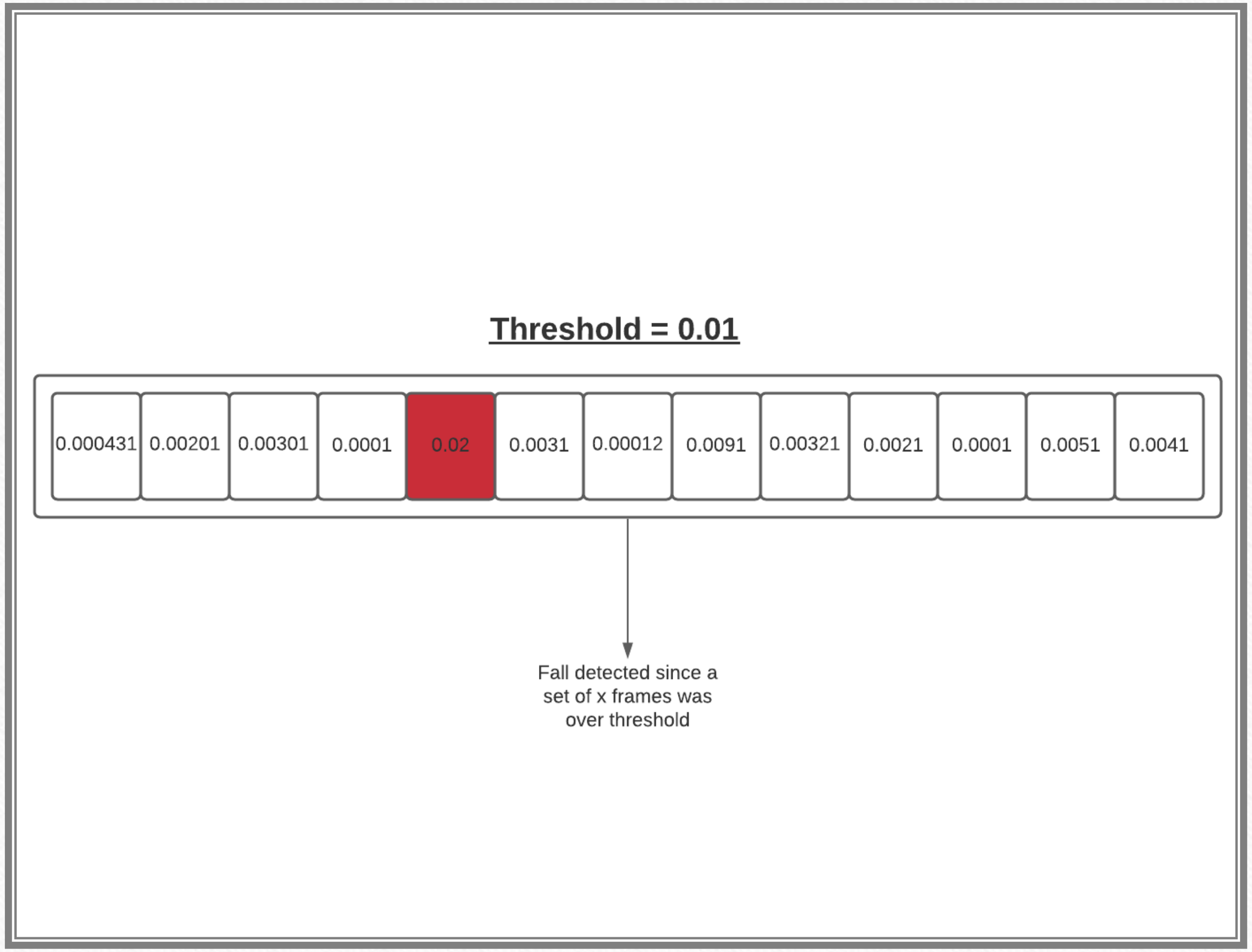


My Model Design – Input Output

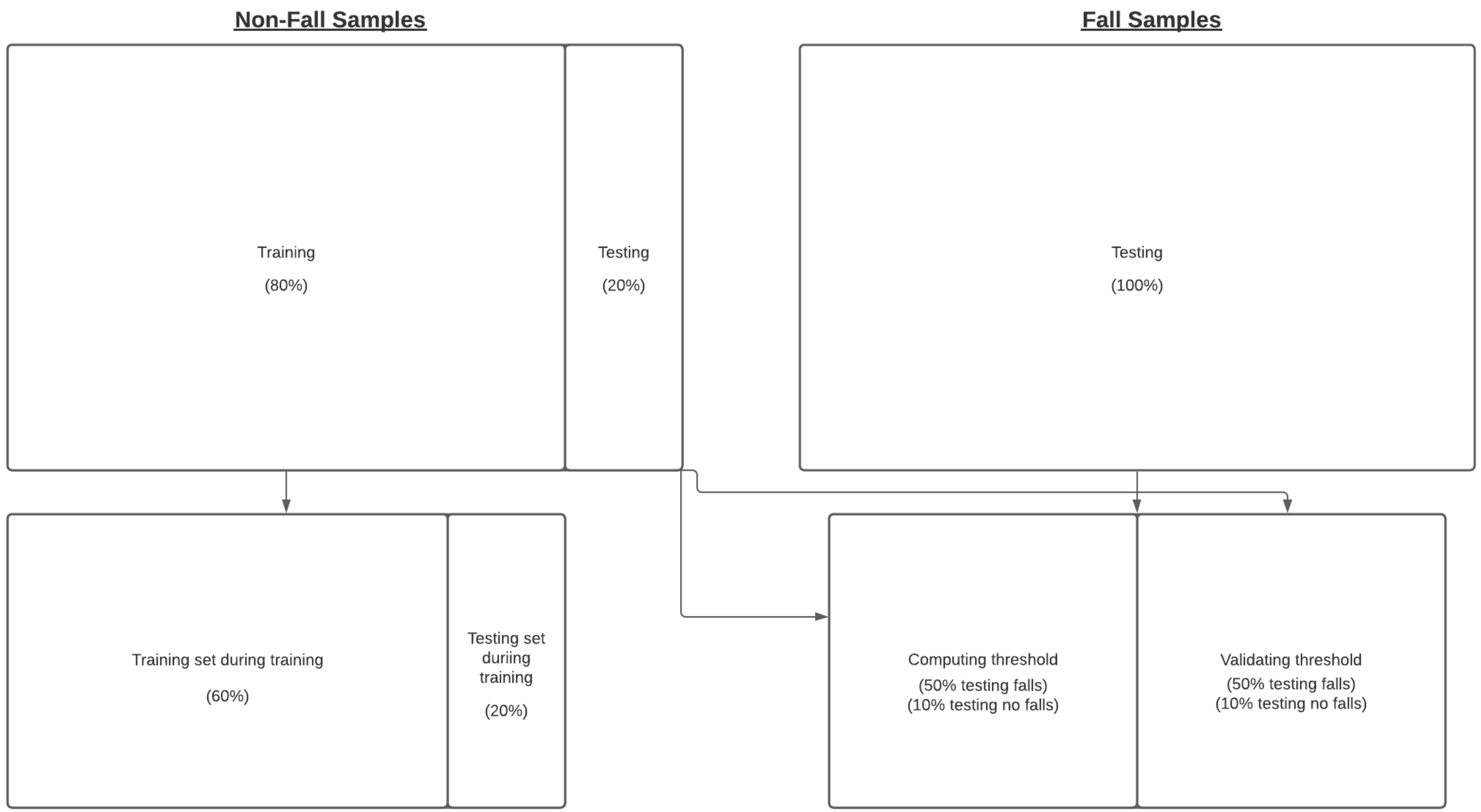



input_1: InputLayer
input: [(?, 10, 372, 4)]
output: [(?, 10, 372, 4)]
time_distributed(flatten): TimeDistributed(Flatten)
input: (?, 10, 372, 4)
output: (?, 10, 1488)
input_mean(dense): TimeDistributed(Dense)
input: (?, 10, 1488)
output: (?, 10, 372)
input_log_variance(dense_1): TimeDistributed(Dense)
input: (?, 10, 1488)
output: (?, 10, 372)
lambda: Lambda
input: [(?, 10, 372), (?, 10, 372)]
output: (?, 10, 372)
simple_rnn: SimpleRNN
input: (?, 10, 372)
output: (?, 16)
repeat_vector: RepeatVector
input: (?, 16)
output: (?, 10, 16)
simple_rnn_1: SimpleRNN
input: (?, 10, 16)
output: (?, 10, 16)
lambda_1: Lambda
input: (?, 10, 16)
output: (?, 10, 16)
time_distributed_1(dense_2): TimeDistributed(Dense)
input: (?, 10, 16)
output: (?, 10, 372)
time_distributed_2(dense_3): TimeDistributed(Dense)
input: (?, 10, 372)
output: (?, 10, 4)
time_distributed_3(dense_4): TimeDistributed(Dense)
input: (?, 10, 372)
output: (?, 10, 4)
concatenate: Concatenate
input: [(?, 10, 4), (?, 10, 4)]
output: (?, 10, 8)
time_distributed_4(repeat_vector_1): TimeDistributed(RepeatVector)
input: (?, 10, 8)
output: (?, 10, 372, 8)
test(reshape): TimeDistributed(Reshape)
input: (?, 10, 372, 8)
output: (?, 10, 372, 8)

Thresholding



- To determine if a fall occurred, we check if a loss value is above a threshold
- Finding an optimal threshold to both minimise false positives and maximise true positives is a hard task
- Youden index can be used to determine this threshold
- Threshold must be validated itself
    - Test set split into two


Non-Fall Samples
Fall Samples
Training
(80%)
Testing
(20%)
Testing
(100%)
Training set during training
(60%)
Testing set duriing training
(20%)
Computing threshold
(50% testing falls)
(10% testing no falls)
Validating threshold
(50% testing falls)
(10% testing no falls)

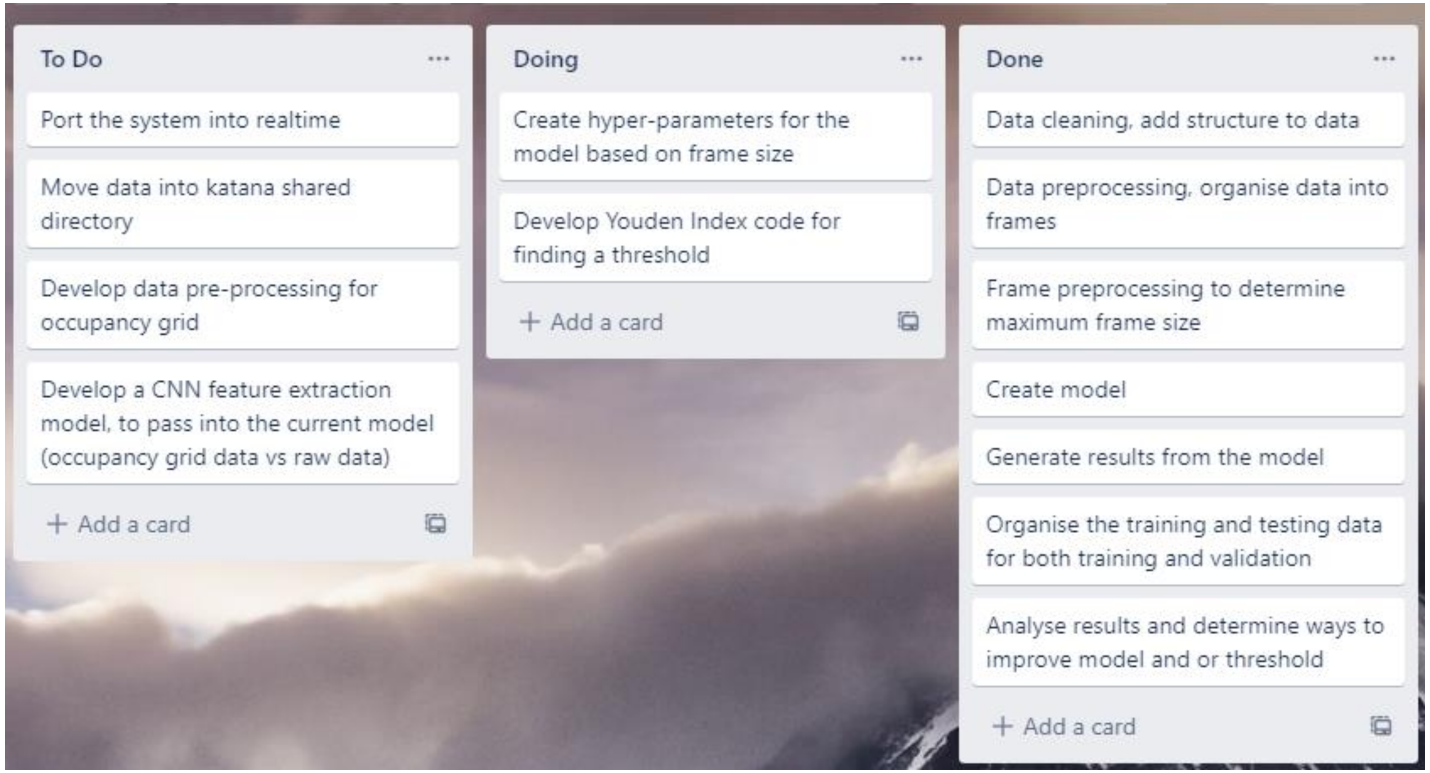
Results of Model From Thesis B

- The current model has yet to implement the youden's index to find an optimal threshold
- This term was more focused on pre-processing and model design
  - Training hasn't been completed yet to a satisfactory level
- Due to the losses in the data, it made it impossible to find a threshold
  - Using 10 frames to determine a loss is not enough
- Despite no great results this term so far, I've set the ball rolling for Thesis C

Outcome of Thesis B

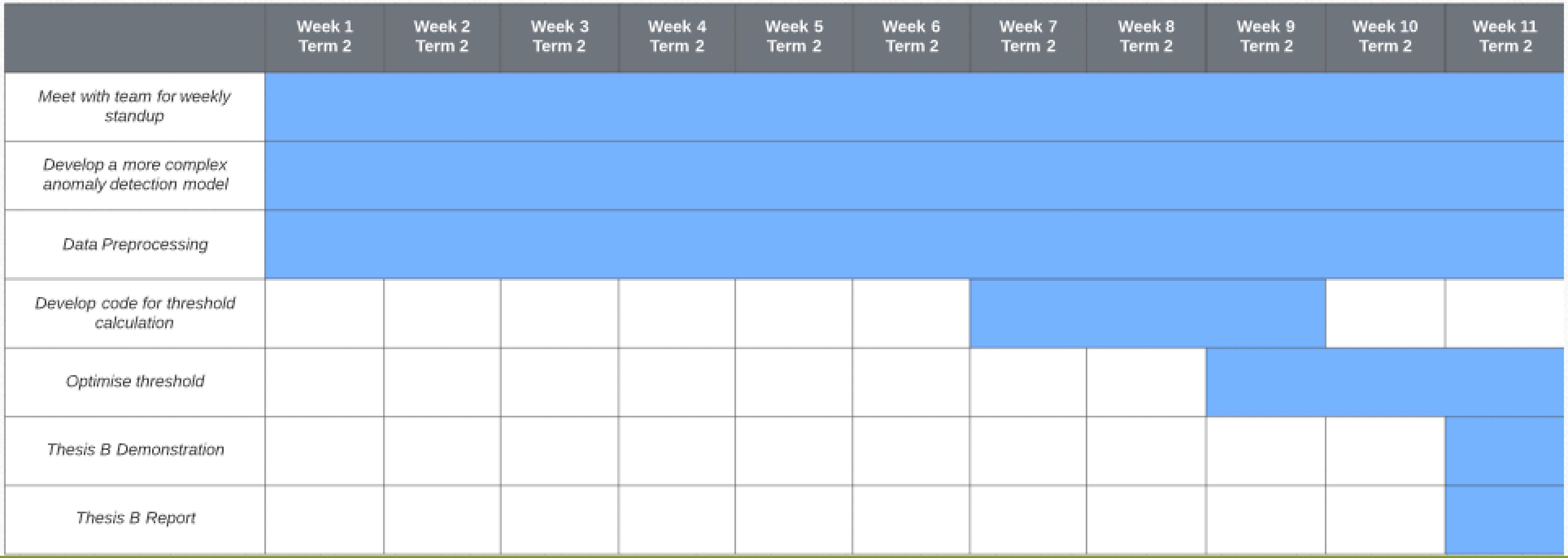
- The model planned for development was completed in its initial form
- All data preprocessing issues overcome
- Data is now processed automatically with many customizable options
  - Still adding more
- The model generated is flexible with many hyper-parameters
- Model can be easily ported into a real time solution

Checklist

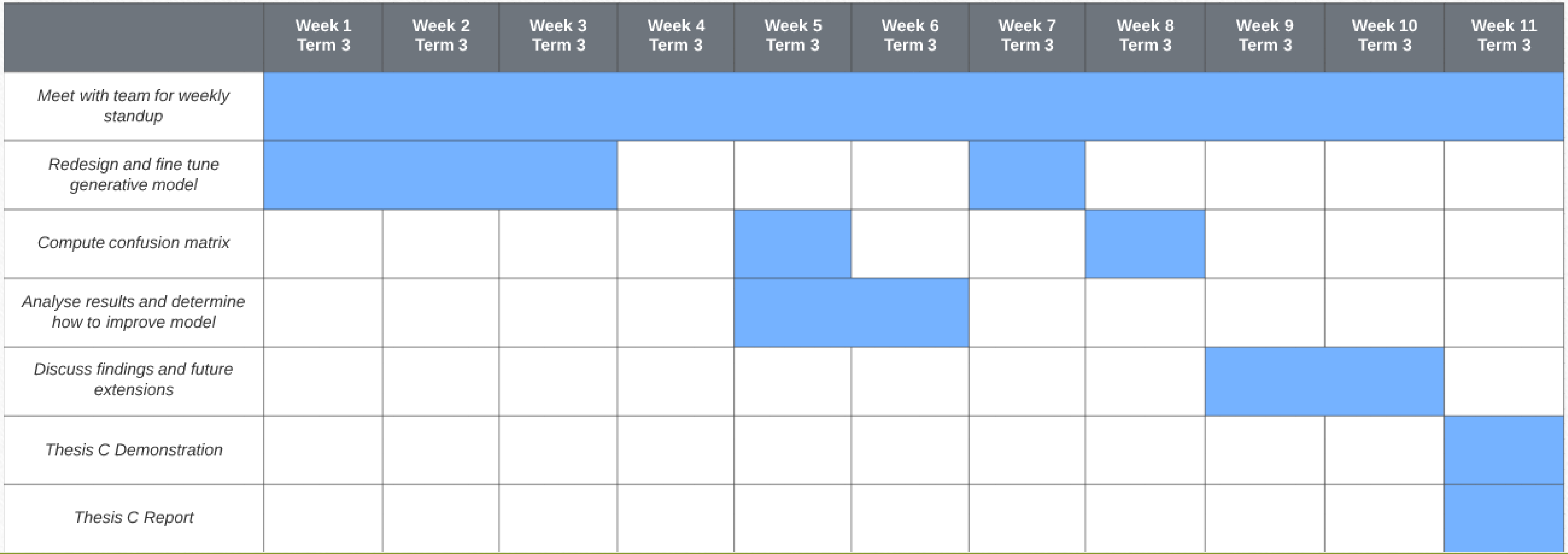


To Do
Port the system into realtime
Move data into katana shared directory
Develop data pre-processing for occupancy grid
Develop a CNN feature extraction model, to pass into the current model (occupancy grid data vs raw data)
+ Add a card
Doing
Create hyper-parameters for the model based on frame size
Develop Youden Index code for finding a threshold
+ Add a card
Done
Data cleaning, add structure to data
Data preprocessing, organise data into frames
Frame preprocessing to determine maximum frame size
Create model
Generate results from the model
Organise the training and testing data for both training and validation
Analyse results and determine ways to improve model and or threshold
+ Add a card

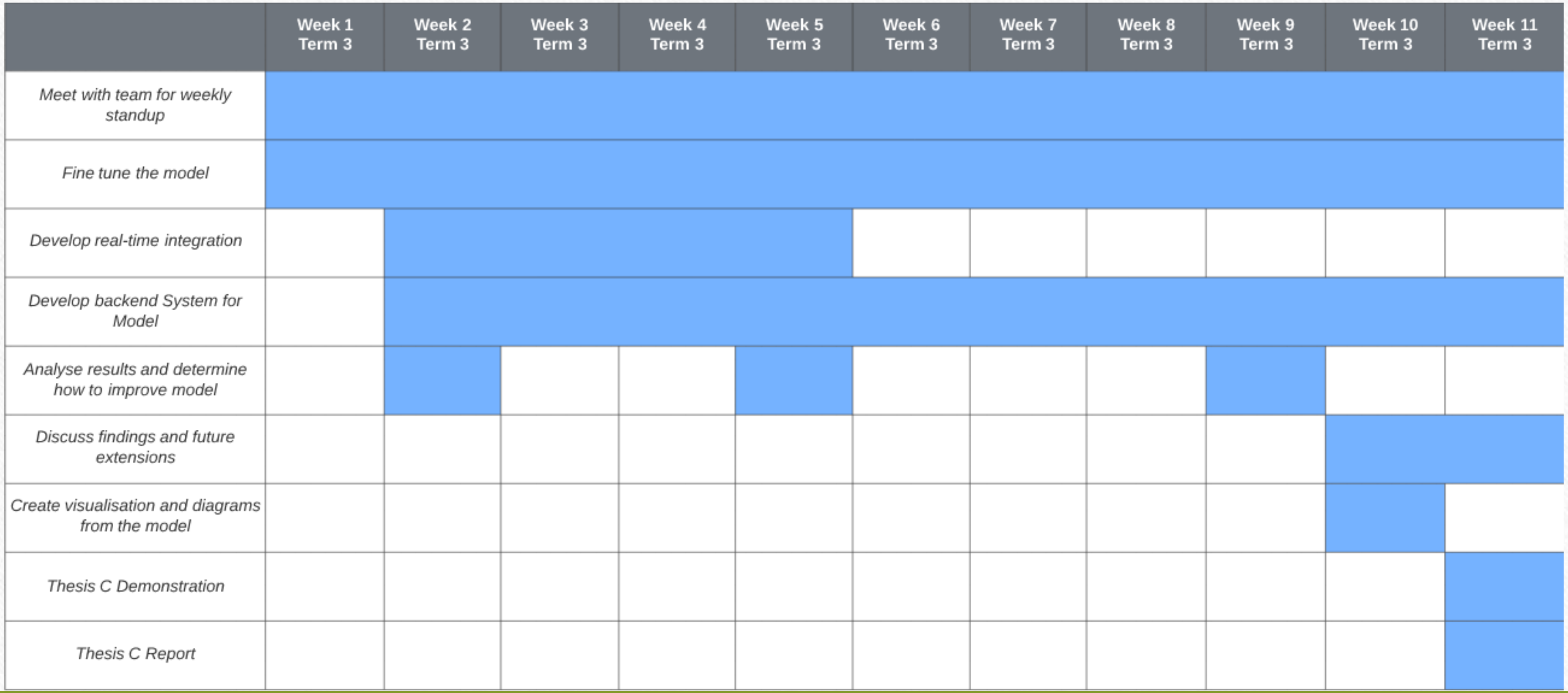
Revised Gantt Chart For Thesis B



Old Gantt Chart For Thesis C



Revised Gantt Chart For Thesis C



References

- [1] R. Luque, E. Casilari, M.-J. Mor´on, G. Redondo, "Comparison and characterisation of android-based fall systems, sensors 14", IEEE, 2014, 18543-18574
- [2] WHO, Falls, World Health Organization, 2018, [Online]. Available: https://www.who.int/news-roo#m/fact-sheets/detail/falls
- [3] S. S. Khan, J. Hoey, "Review of Fall Detection Techniques: A Data Availability Perspective", University of Waterloo, 2016, pp.6, 8-9, 12, 19
- [4] L. Schwickert, C. Becker, U. Lindemann, C. Mar´echal, A. Bourke, L. Chiari, J. Helbostad, W. Zijlstra, K. Aminian, C. Todd, S. Bandinelli, J. Klenk, "Fall detection with body-worn sensors: A systematic review, Zeitschrift f¨ur Gerontologie und Geriatrie", IEEE, 2013, 706–719.

References

- [5] E. Stone, M. Skubic, “Fall detection in homes of older adults using the microsoft kinect, Biomedical and Health Informatics”, IEEE, 2015, pp. 290–301.
- [6] J. Perry, S. Kellog, S. Vaidya, J.-H. Youn, H. Ali, H. Sharif, “Survey and evaluation of real-time fall detection approaches, in: High-Capacity Optical Networks and Enabling Technologies (HONET)”, 2009 6th International Symposium on, 2009, pp. 158–164.
- [7] F. Jin, A. Sengupta, S. Cao, “mmFall: Fall Detection using 4D MmWave Radar and a Hybrid Variational RNN AutoEncoder”, IEEE, 2020, pp. 1

References

- [8] H. Haibo, E. Garcia, Learning from imbalanced data, “Knowledge and Data Engineering”, IEEE, 2009, 1263–1284.
- [9] W. Klement, S. Wilk, W. Michalowski, S. Matwin, “Classifying severely imbalanced data, in: Advances in Artificial Intelligence”, Springer, 2011, pp. 258–264.