Review on Current Papers:

1. Paper by Kim, S. mainly predicts building and floor given a bunch of input signals. For the referenced points labels, the prediction accuracy is rather low. Thus, he tracked down the first k possible points’ information as predicted results. Then, he calculated the average value of their latitude and longitude. His prediction on building-floor looks good, however, when I modified and re-ran his code (from his GitHub), the accuracy on referenced point prediction is actually very low. Averaging k neighbors is indeed one solution. However, enhanced prediction accuracy is still necessary.

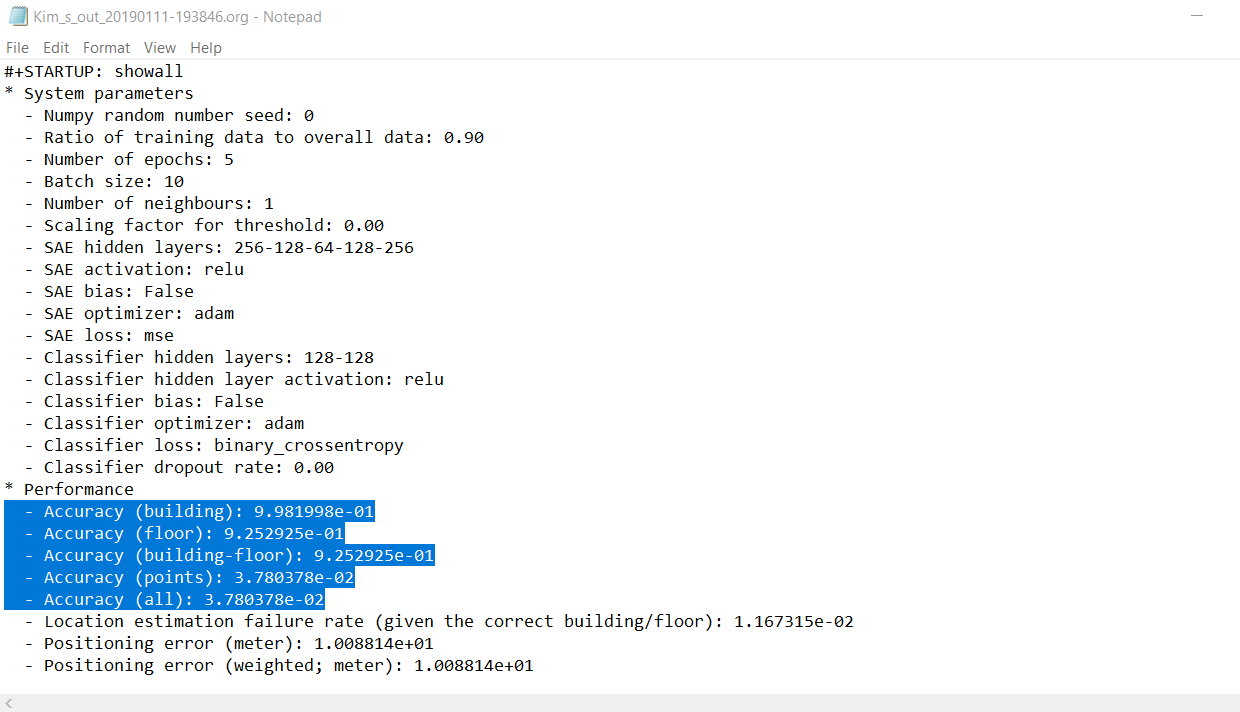


Figure 1: reproduce Kim's work

The accuracy of building-floor prediction reaches 91% on average. He claims that this takes advantage of the Stacked Autoencoder (SAE) (or Deep Autoencoder). This is reasonable. To find whether the data is sparse input, I checked its distribution:

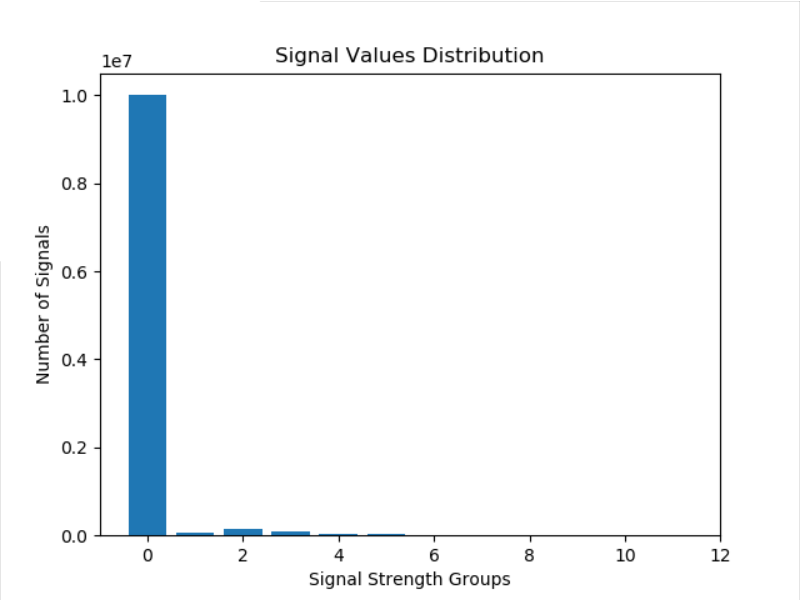


Figure 2: Data Distribution

From Figure 2, we can see that 0 inputs are far more than non-zero inputs, which makes the dataset a sparse input. Hence, theoretically reducing its dimensions or extracting key features will be helpful for later training.

To testify how well the SAE performs, I first designed a simple (520-256-128-118) DNN with combined one-hot labels, without any autoencoders:

For 1111 test data:

building accuracy: 99.9099909990999 %

building + floor prediction accuracy: 90.36903690369037 %

Without using any autoencoder, the model has already reached building-floor accuracy over 90%.

Next, I added a single layer autoencoder with 256 nodes:

For 1111 test data:

building accuracy: 99.63996399639964 %

building + floor prediction accuracy: 82.8982898289829 %

Next, I used a 520-256-128-256-520 autoencoder:

For 1111 test data:

building accuracy: 97.65976597659765 %

building + floor prediction accuracy: 55.80558055805581 %

The model did not get any enhancement as expected. On the contrary it performed worse. Kim uses 520-256-128-64-128-256-520 SAE together with two 128-node fully connected layer. This model does not work for me and I get rather low prediction accuracy. Meanwhile, I did not find any big differences between our codes except I uses two model objects, which should not be the main reason.

Although reproducing SAE did not work, reducing sparse input dimensions or extracting key features should still be the main target for enhancing overall prediction accuracy. However, because compressing dimensions will cause inevitable information loss, and because the original input dimension is not that high as traditional sparse input problem, it’s also understandable if no obvious enhancement is achieved.

2. CNN based classification by Jang and Hong.

Convolutional neural networks showed great performance in image processing tasks. Due to its learning characteristics, this model performs well to noisy images, twisted and modified images. We can also generate reinforced dataset to offset the scarce of training data. For example, change grey scales, translate and stretch. The RSSI-classification problem is not suitable for a CNN model at the first glance. Even if we can reshape the input to a 2d image, we can hardly do data reinforcement due to its data characteristics. Unlike normal images, translation, change of grey scales of the reshaped image both may lead to the change of allocated labels. With such suspicion, I created a CNN model and find how it works.

(1) Reshape of input array to 23 \* 23 images.

Like Jang and Hong’s paper, I added 9 zeros to the original (,520) inputs, thus satisfies 23 \* 23 condition. An abstract of first 100 images are shown below.

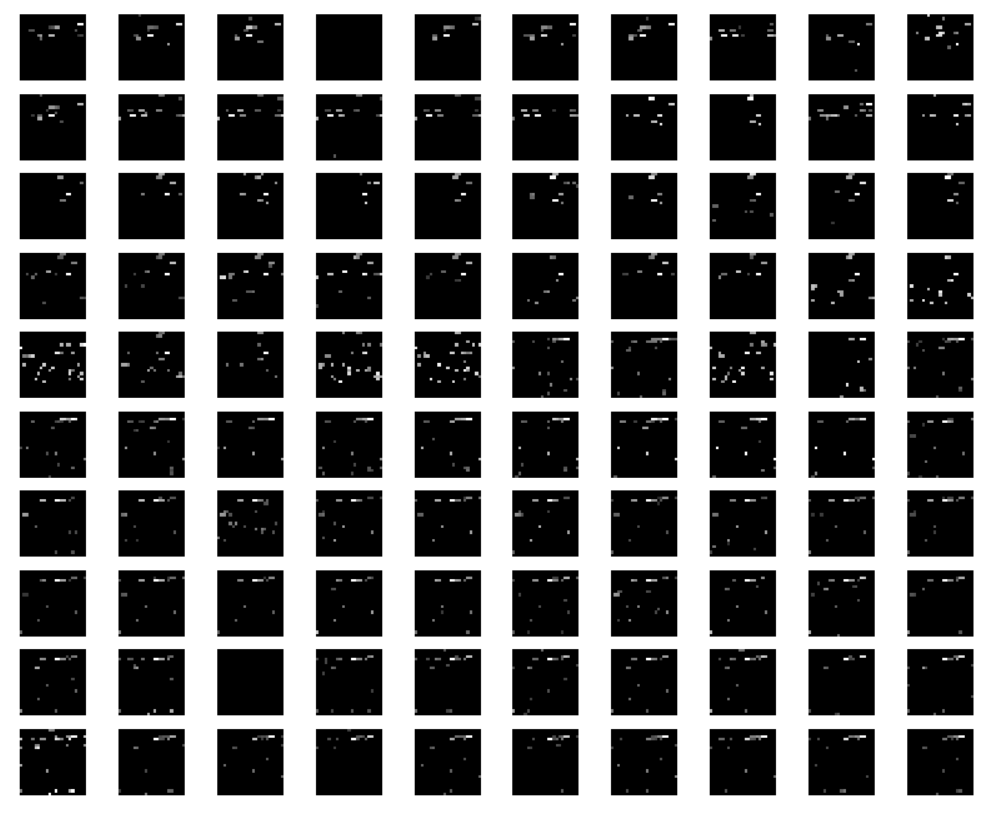


Figure 3: 2d representation of inputs

Details for the CNN model are as follows:

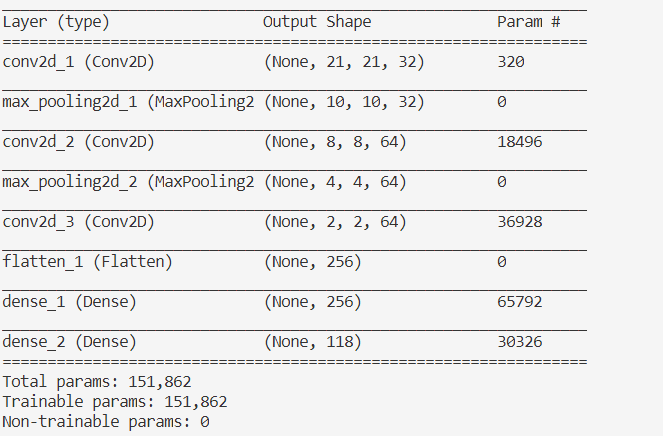


Figure 4: CNN model information

Results:

For 1111 test data:

building accuracy: 98.37983798379838 %

building + floor prediction accuracy: 70.83708370837084 %

This model fails to provide high accuracy as expected. If we reinforce input data, it is possible that we can improve the prediction accuracy. However, as stated above, this CNN model is not perfectly suitable for this problem because there are no obvious features for the signal images.

Additional Works:

1. Rectify errors in previous codes, optimize code structures.

(1). Re-split data as: train, validation & test. In previous version, validation is mistakenly used as test, which triggers incorrect high accuracy.

(2). Rewrite prediction accuracy counting code. In previous version, wrong predictions of Referenced Points were only counted when Building and Floor are predicted correctly, which caused incorrect high accuracy.

(3). Shuffle data before processing, which avoids data similar to test data has never been trained before.

2. Enhancing referenced points prediction:

(1). Manually separate building. Treat every group as multiclass classification.

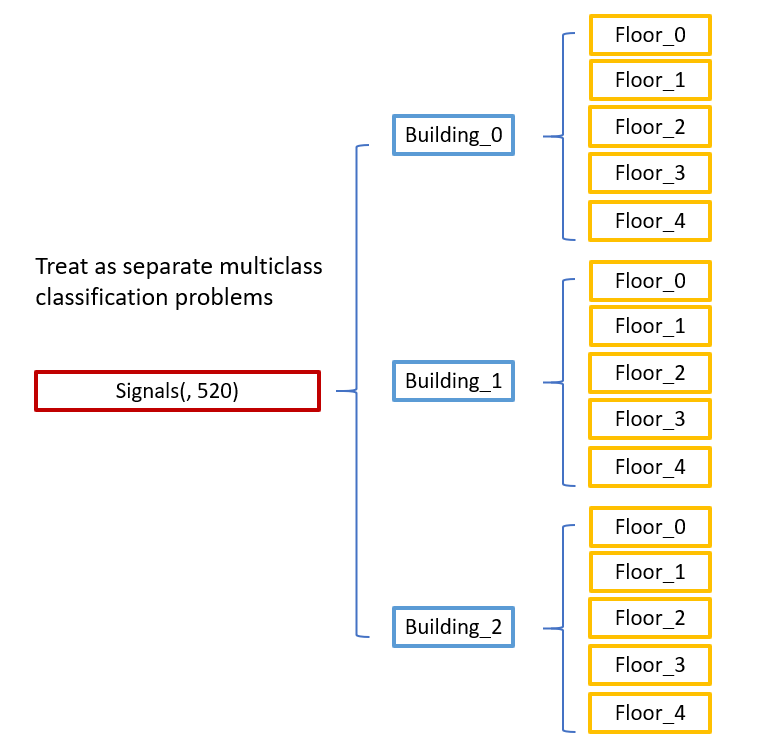


Figure 5: Separated Building

One advantage of separate training is that we can choose different training epochs for different buildings to prevent them from overfitting. In fact, this is necessary as some building may overfit after epoch 10 whereas another may overfit around 20.

Optimal Epochs for the 3 building respectively are 20, 11, 16

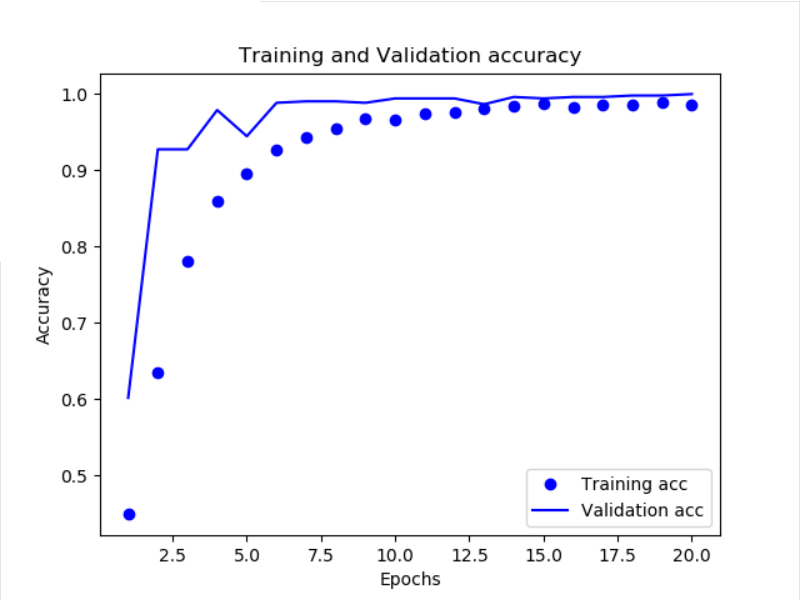
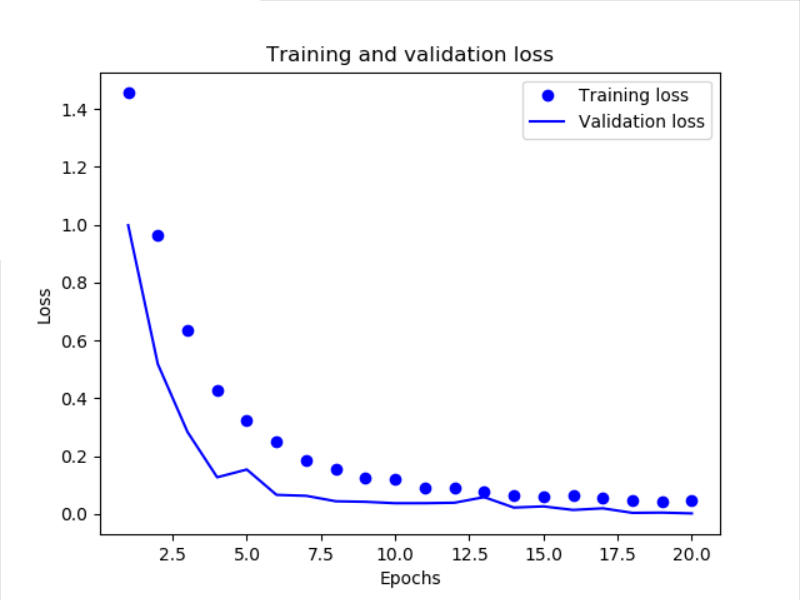


Figure 6: Building 1's validation loss and accuracy

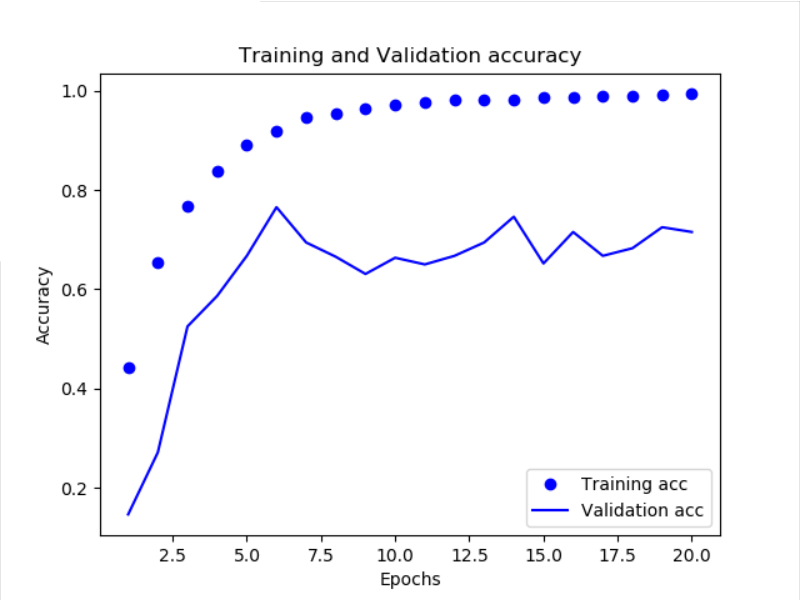
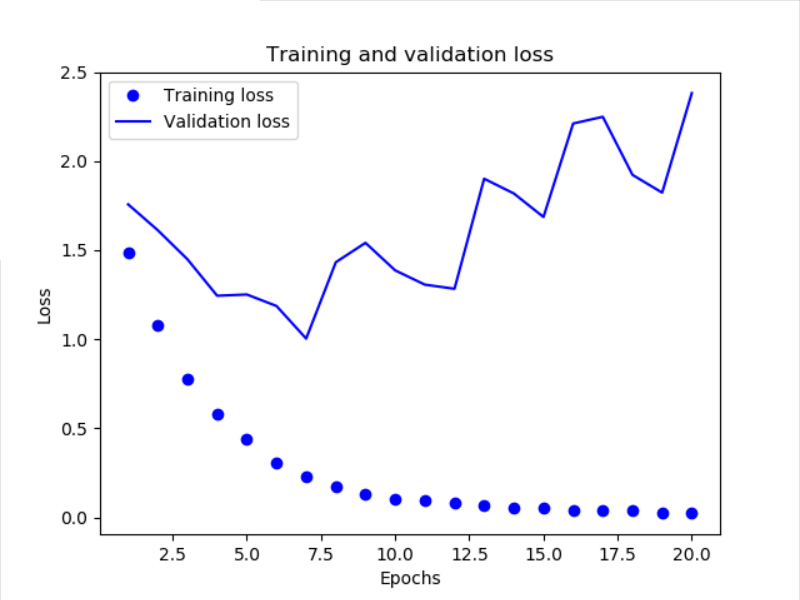


Figure 7: Building 2's validation loss and accuracy

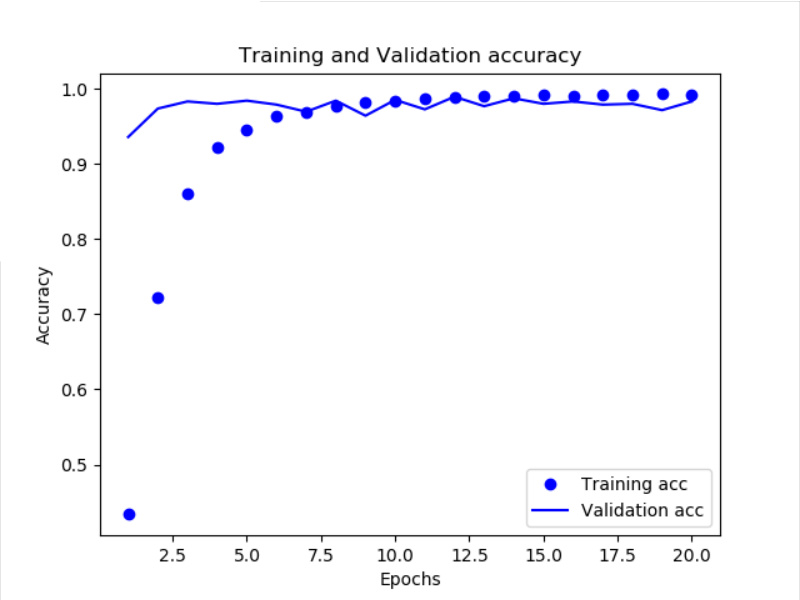
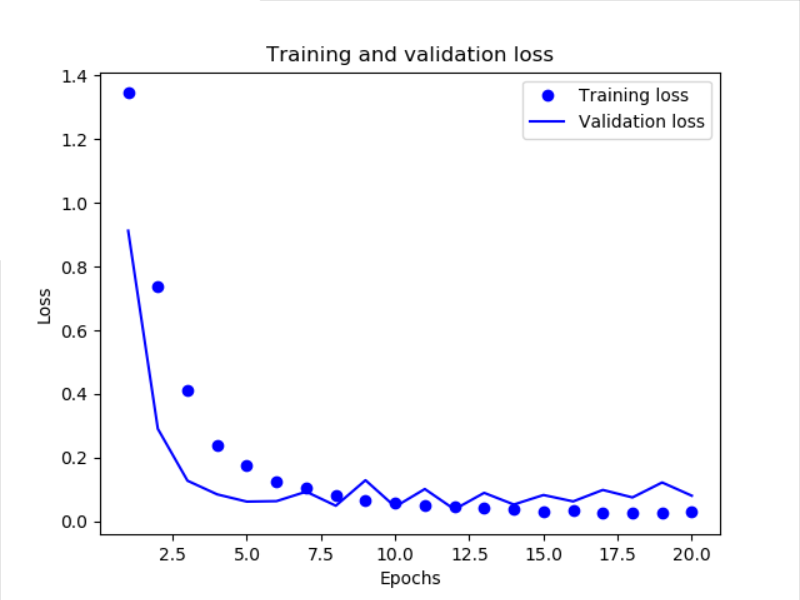


Figure 8: Building 3's validation loss and accuracy

Results for three buildings:

For 536 test data:

accuracy at building 1: 96.45522388059702 %

For 307 test data:

accuracy at building 2: 83.38762214983713 %

For 268 test data:

accuracy at building 3: 92.53731343283582 %

From the images, we can see that validation loss and accuracy are low for Building No.2. Thus, it is reasonable that we separate different buildings and do predictions respectively. One problem of this method is that it is tedious. Every time when we deal with a dataset, we must manually group the data to expected buildings and floors, which takes a lot of time and efforts, and is not suitable for scaling.

Another experiment, adopting CNN after manually classified buildings is also carried out:

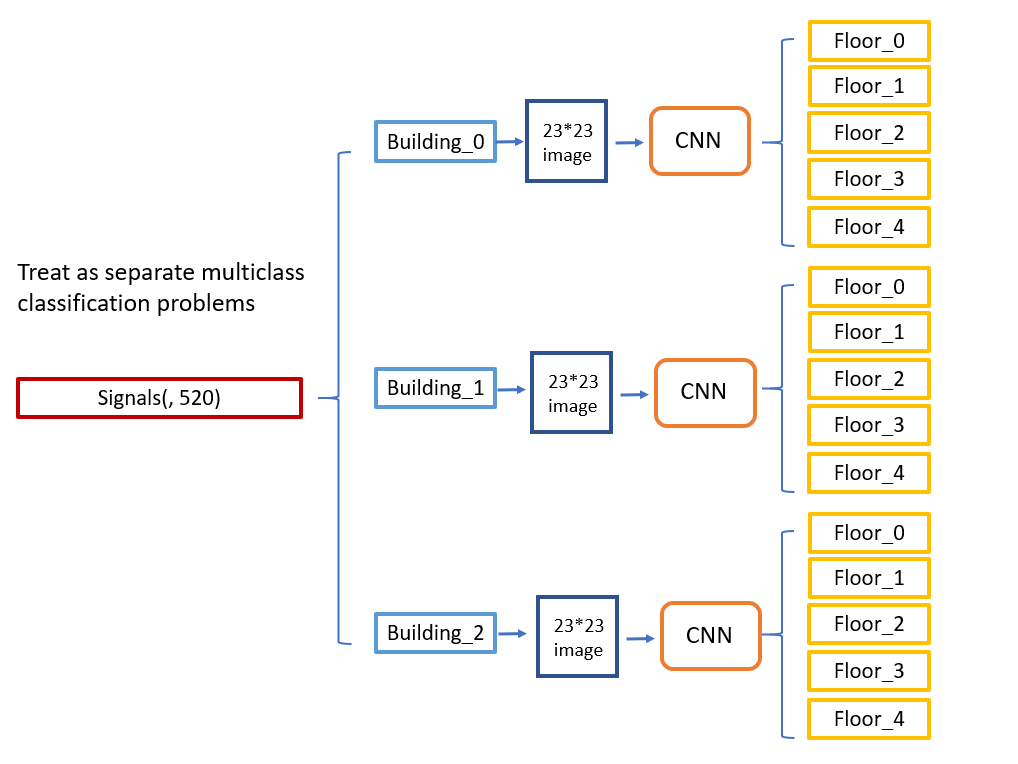


Figure 9: CNN in manually classified buildings

For 536 test data:

floor accuracy: 94.21641791044776 %

For 307 test data:

floor accuracy: 82.41042345276873 %

For 268 test data:

floor accuracy: 92.91044776119404 %

No significant improvements happened.