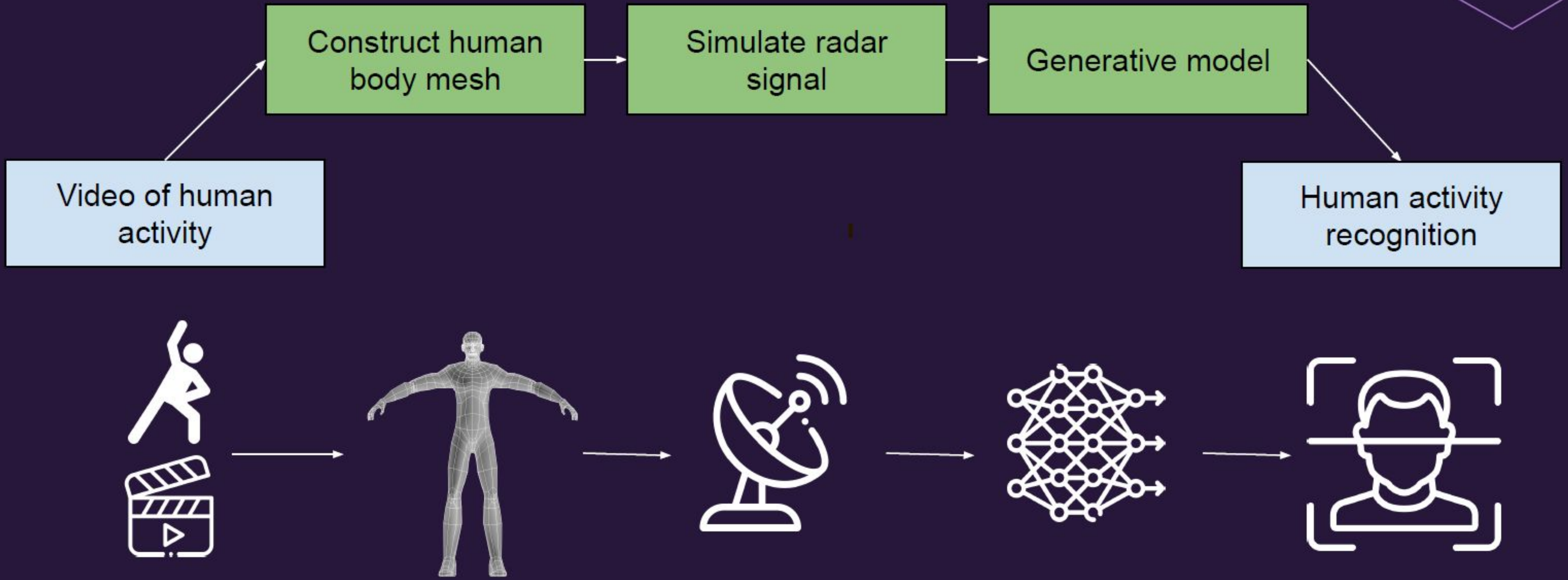
Final Design Review

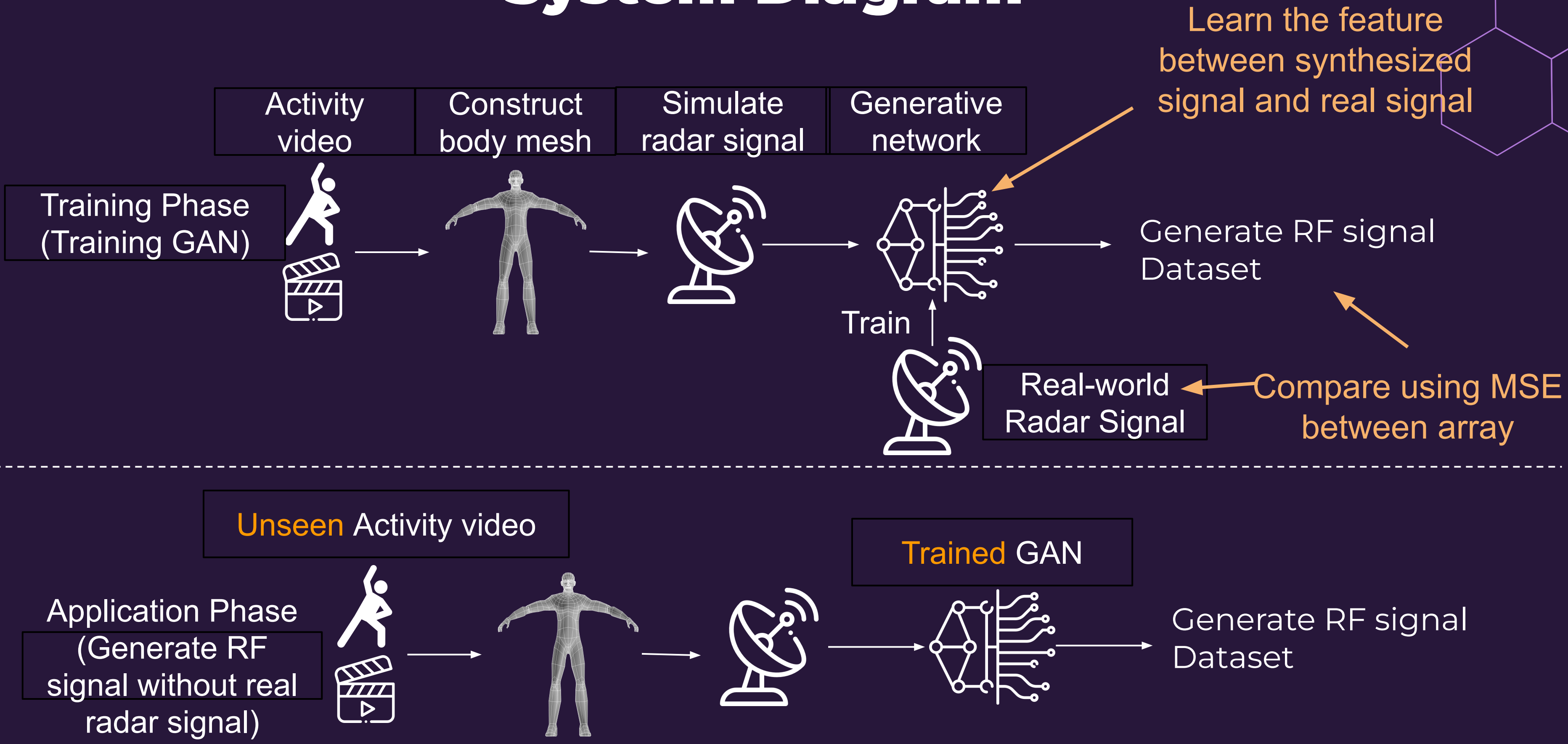
Video to RF Team

A. Overview of Proposed Prototype

Brief overview of your PROJECT focus

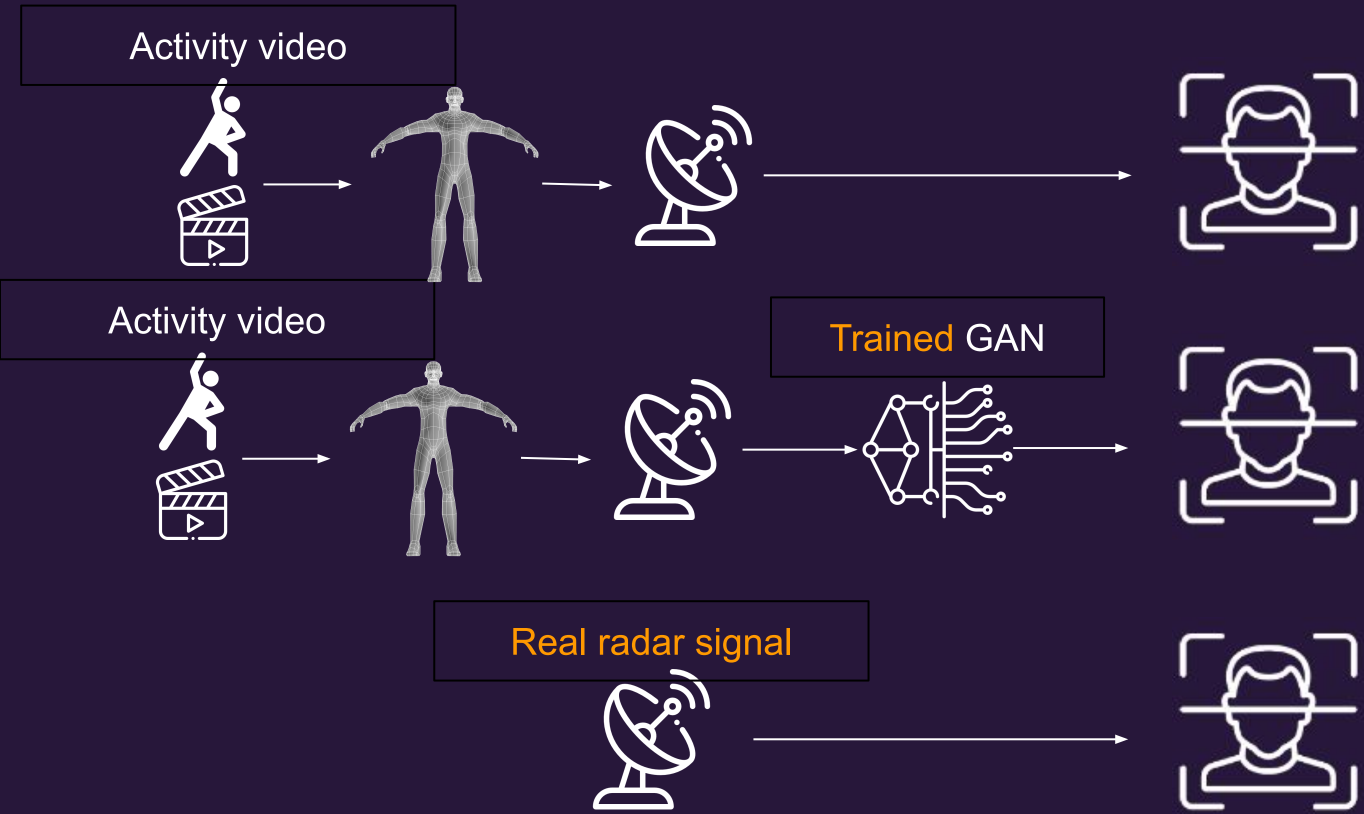


System Diagram



Learn the feature between synthesized signal and real signal
Activity video
Construct body mesh
Simulate radar signal
Generative network
Training Phase (Training GAN)
Generate RF signal Dataset
Train
Real-world Radar Signal
Compare using MSE between array
Unseen Activity video
Trained GAN
Application Phase (Generate RF signal without real radar signal)
Generate RF signal Dataset

Additional task-HAR

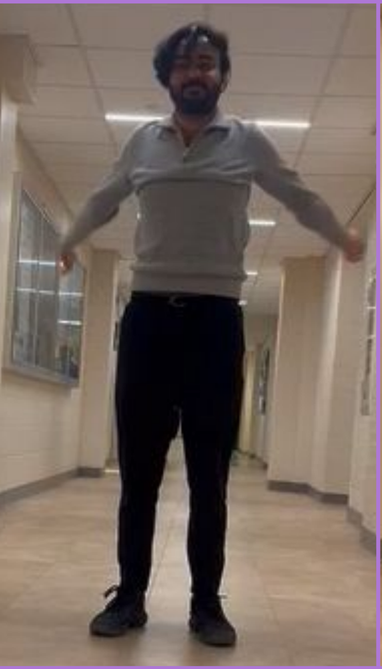


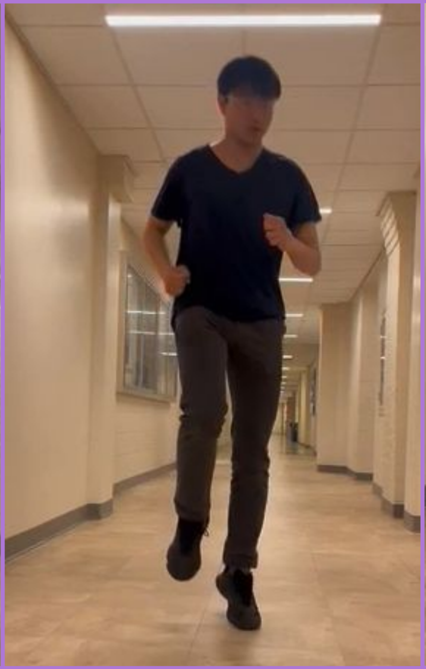
Train the same HAR network using data from 3 different resource

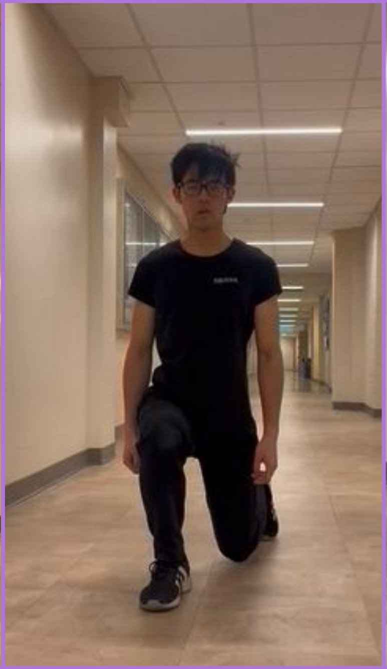
B and C. Technical progress and Schematics

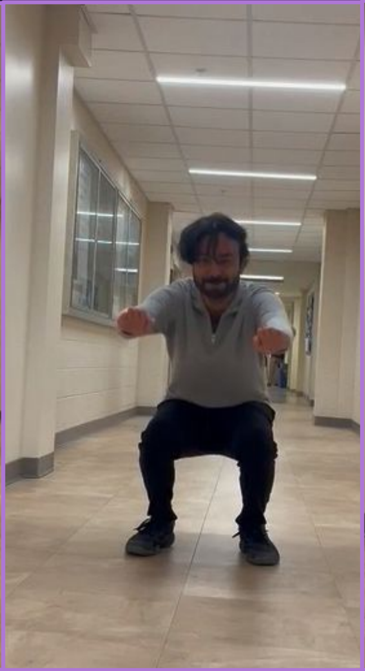
GAN for Signal Improvement

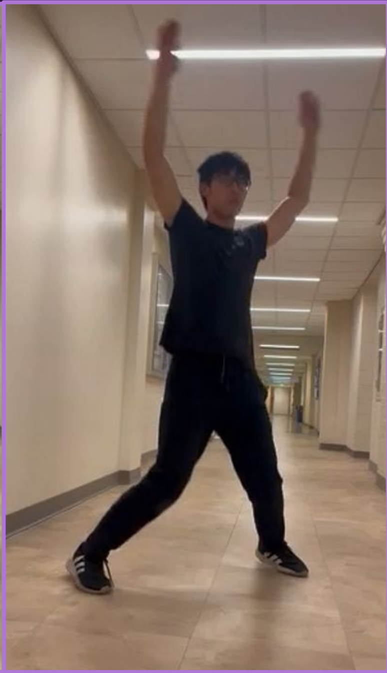
Training Data Collection-5 Motion each for 5 min


Hand Swing


Running


Lunges


Squats


Jumping Jack

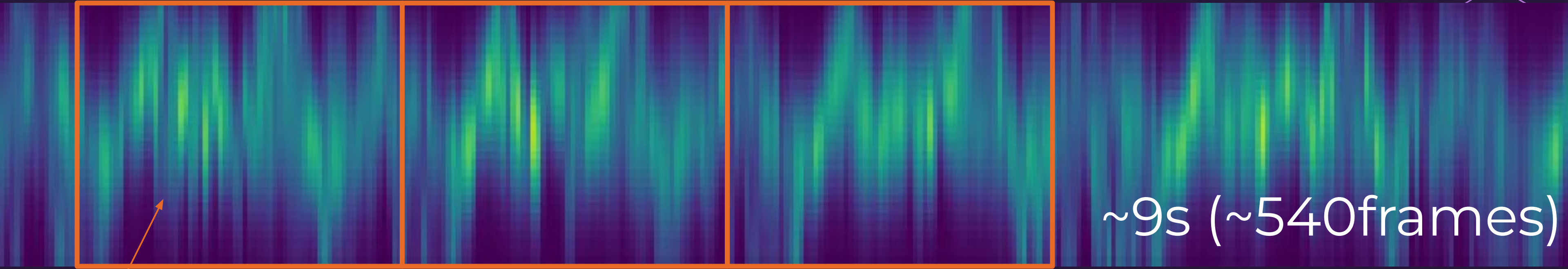
Test Data collection-2 Motion each for 1 min

Hand Clipping

Torso Turning

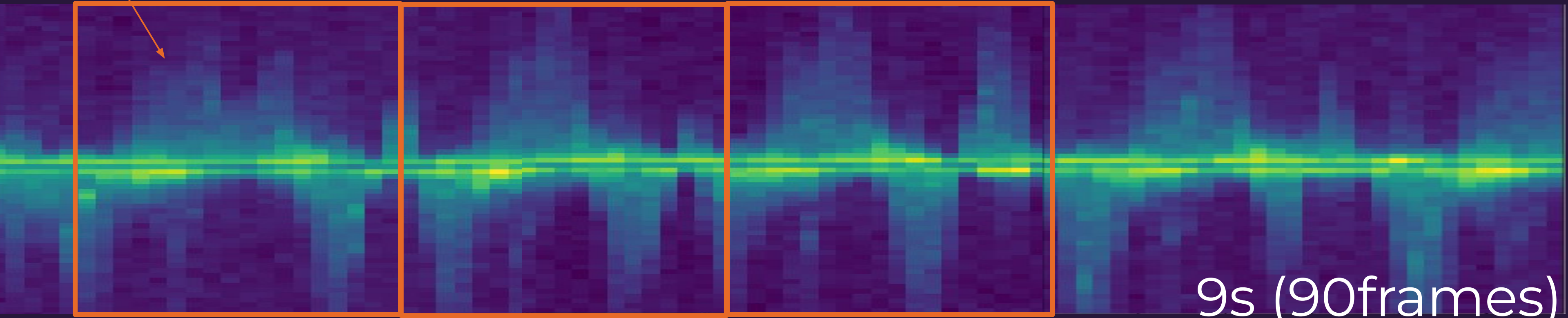
Velocity-Time

Synthesized Signal



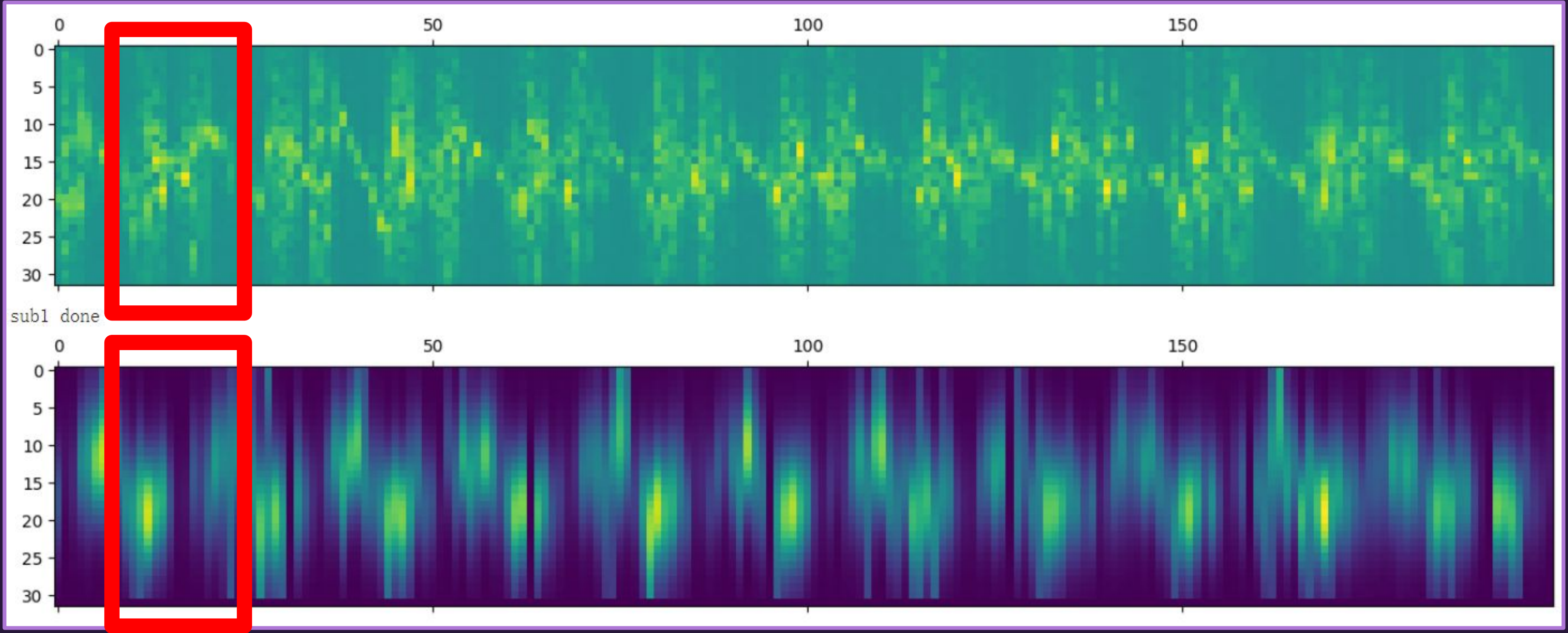
Hand Waving Motion

Real Signal



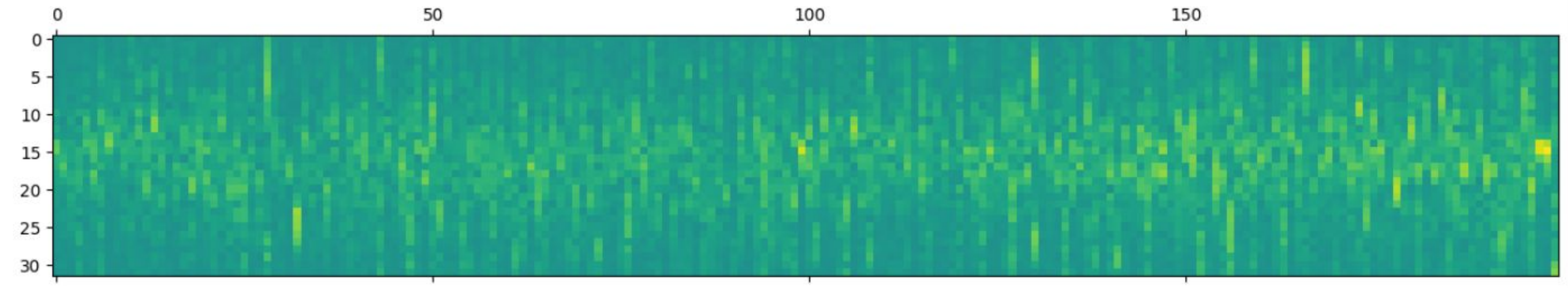
Real Radar Signal

Squat:

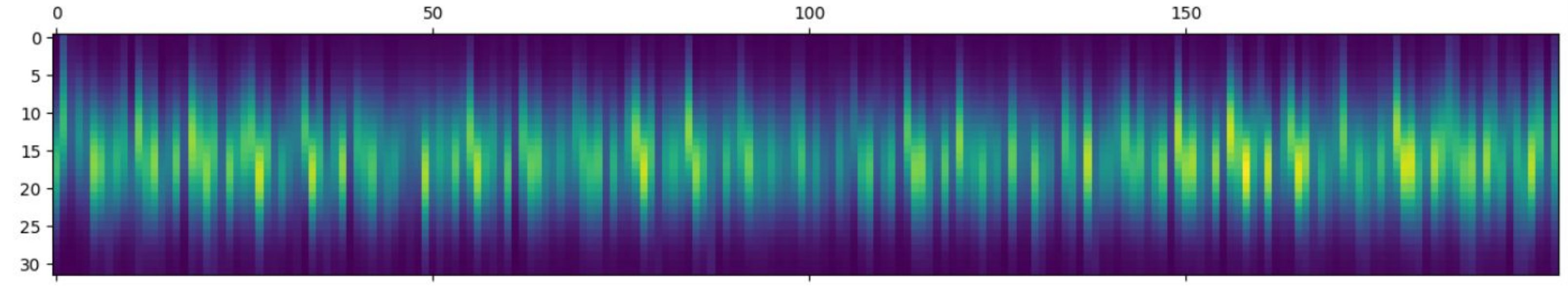


Real Radar Signal

Running:

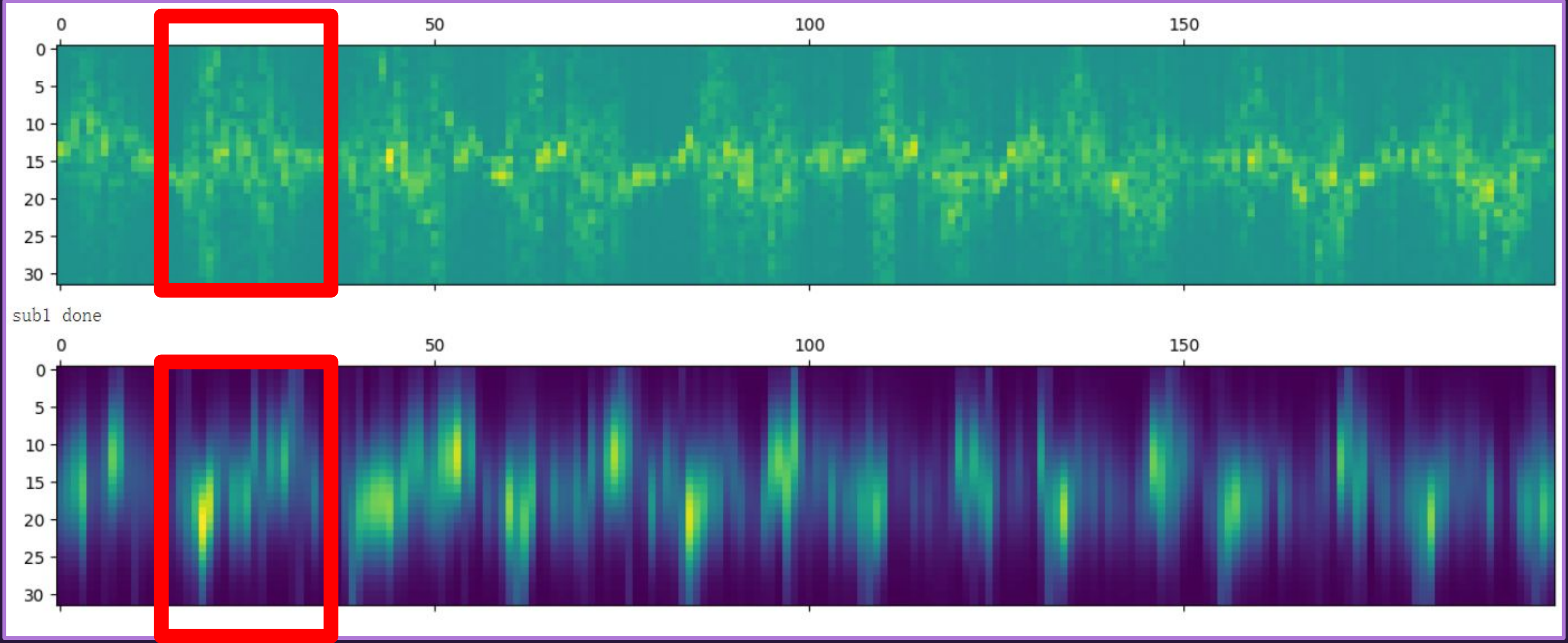


```
sub1 done
```



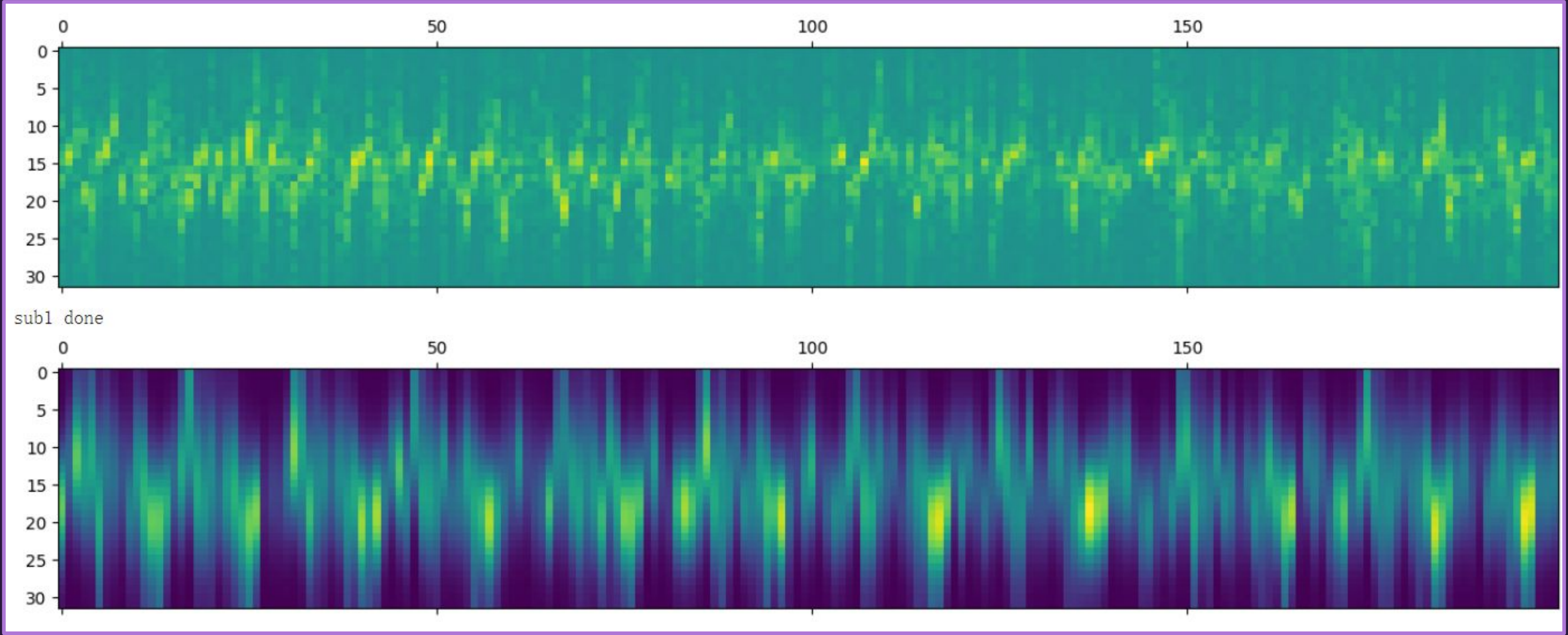
Real Radar Signal

Lunging:



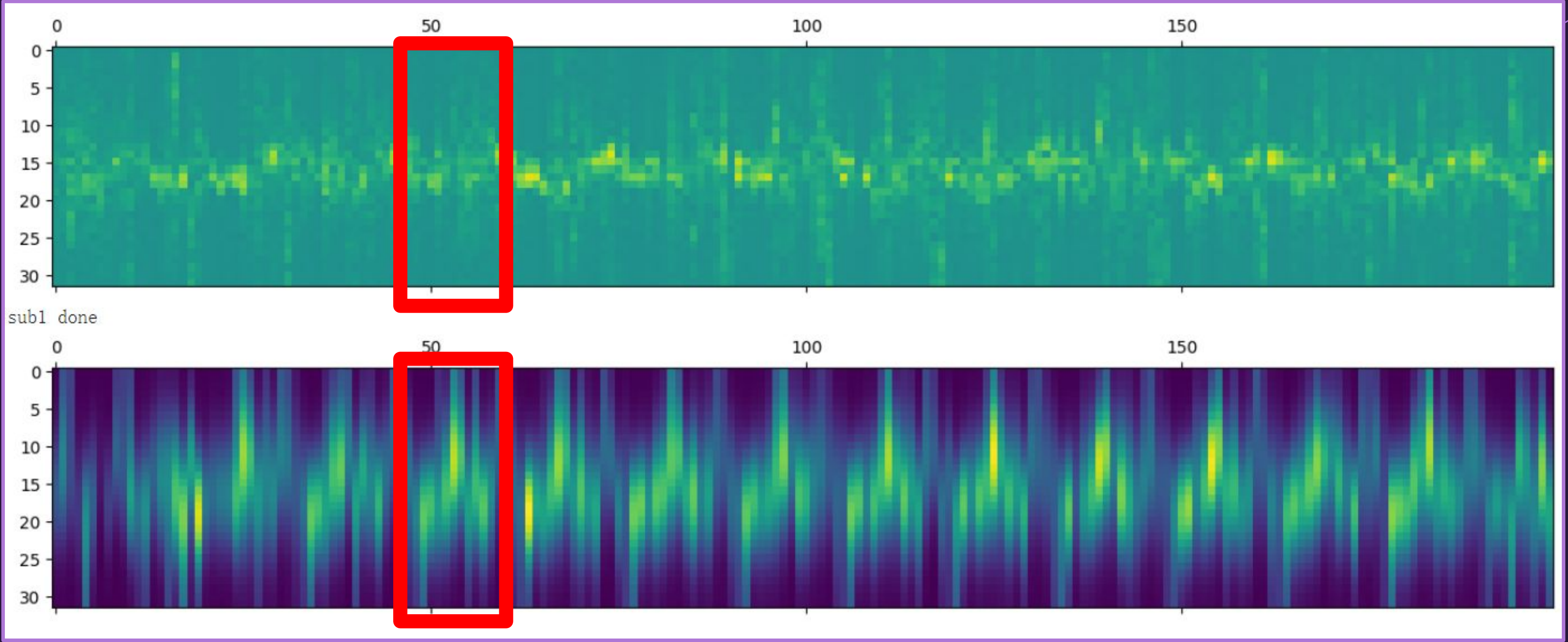
Real Radar Signal

Jumping Jack:



Real Radar Signal

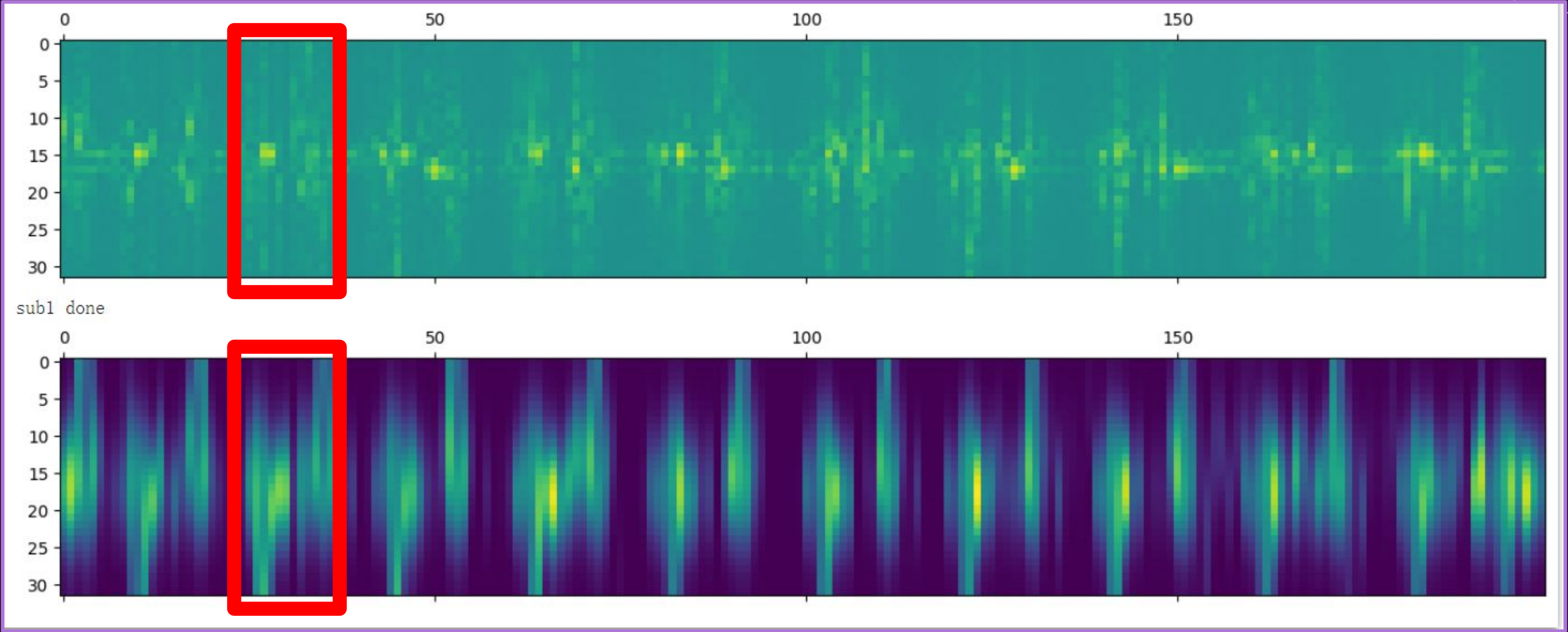
Hand Swing:



```
sub1 done
```

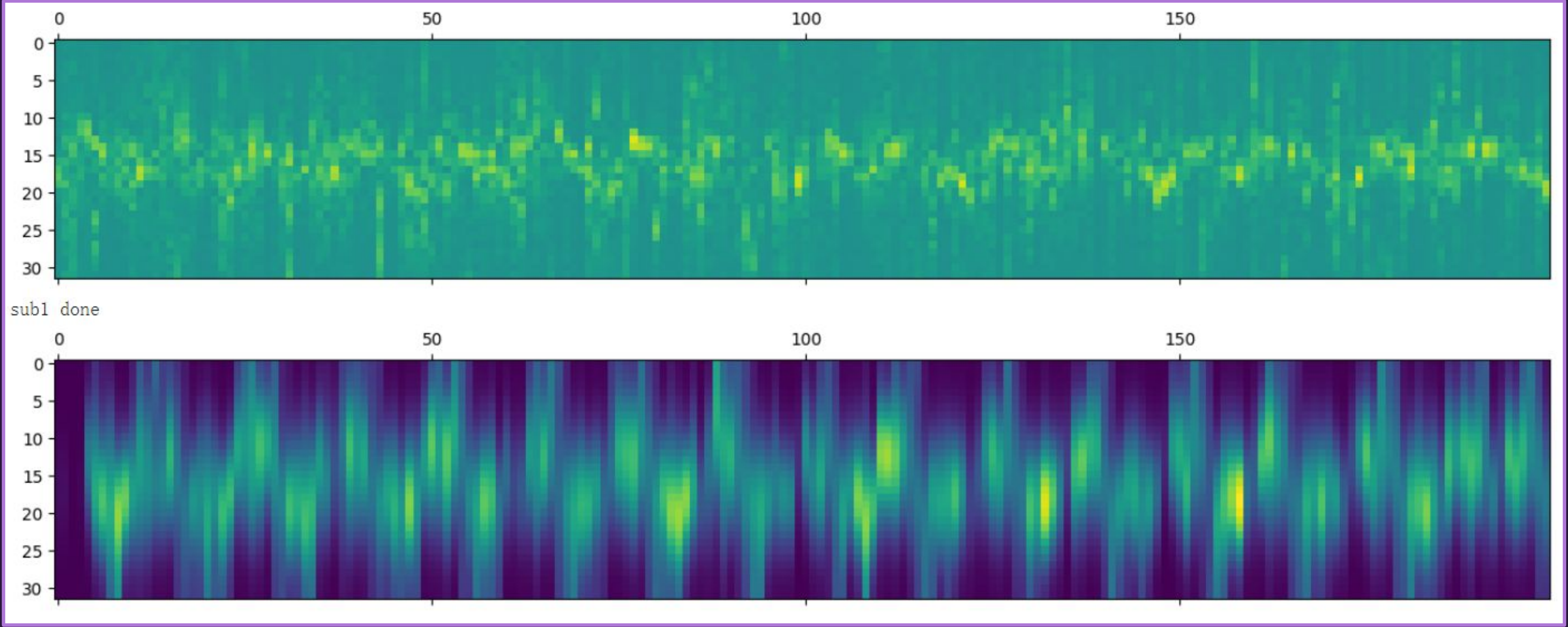
Real Radar Signal

Hand Clipping:



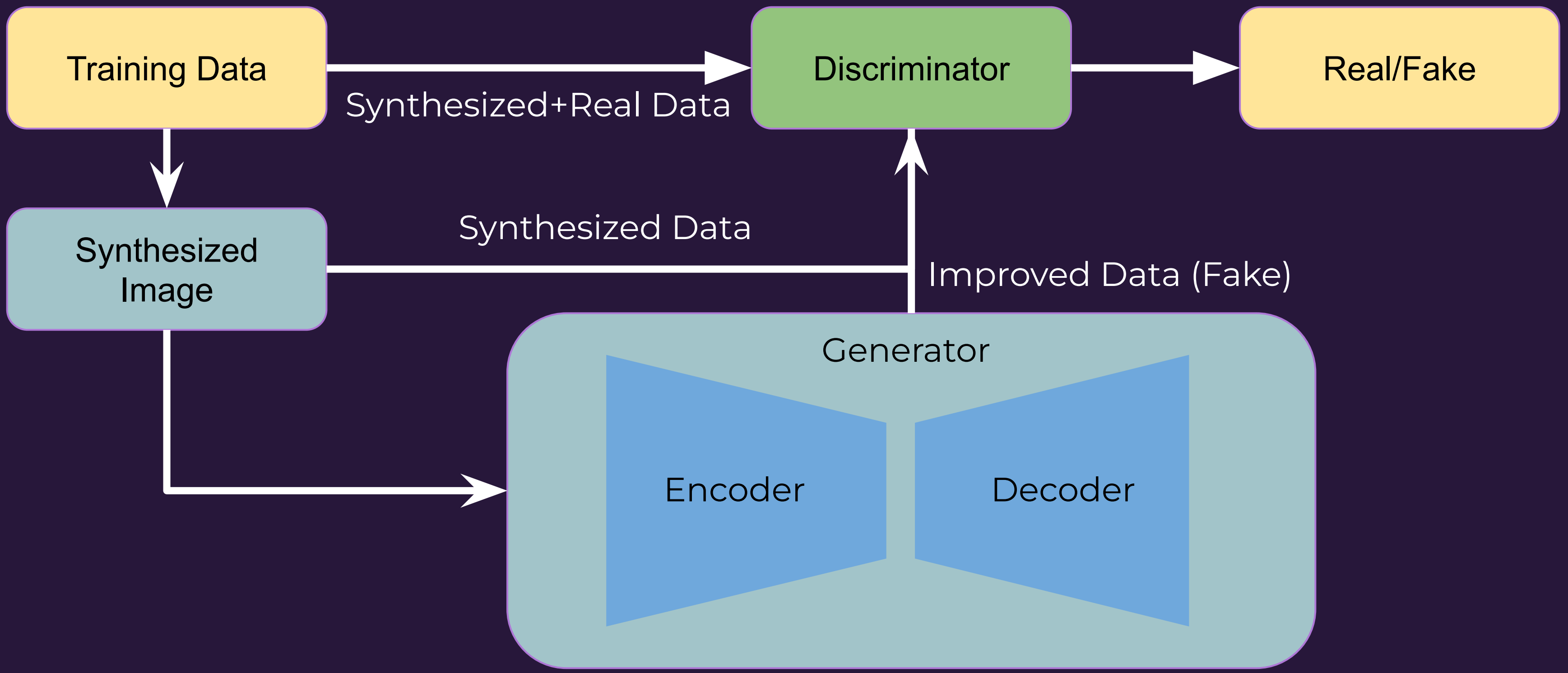
Real Radar Signal

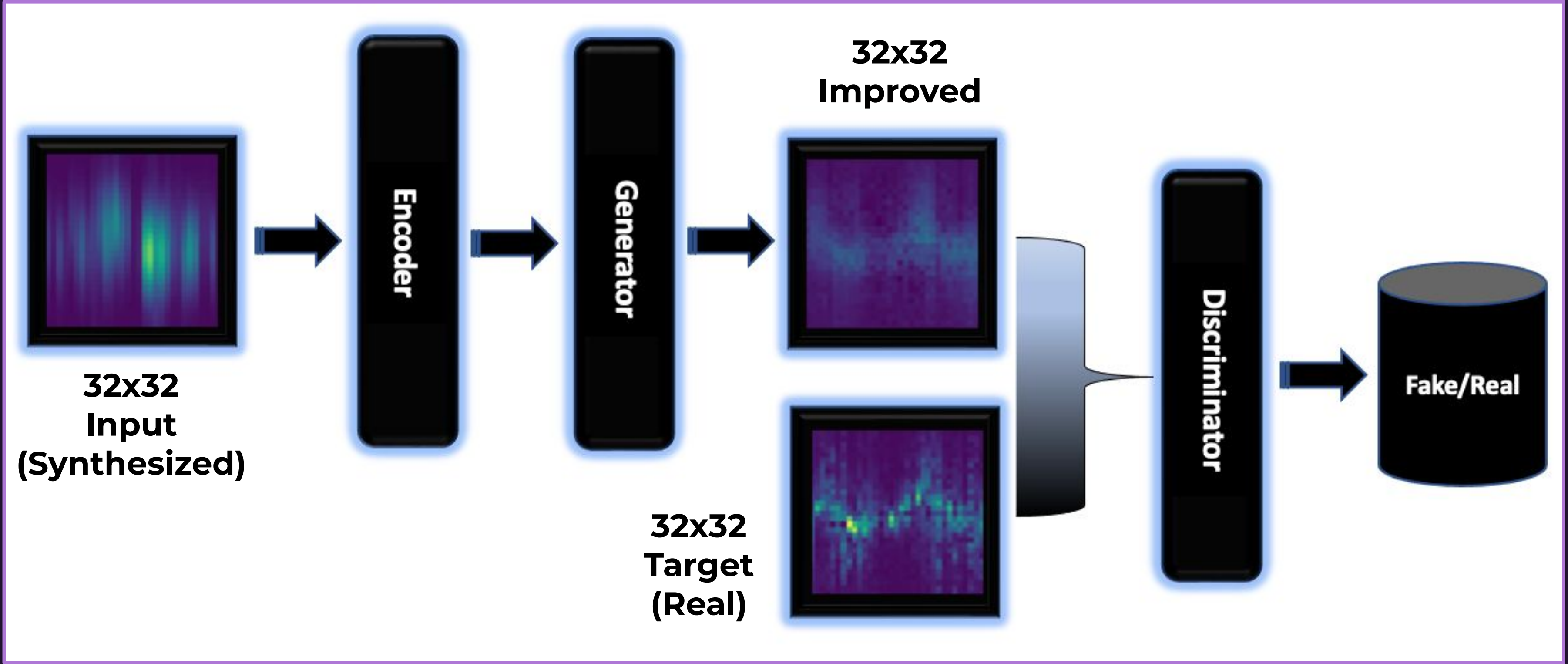
Torso:



```
sub1 done
```

Generative Adversarial Network




32x32
Improved
Encoder
Generator
Discriminator
Fake/Real
32x32
Input
(Synthesized)
32x32
Target
(Real)

Generator Design

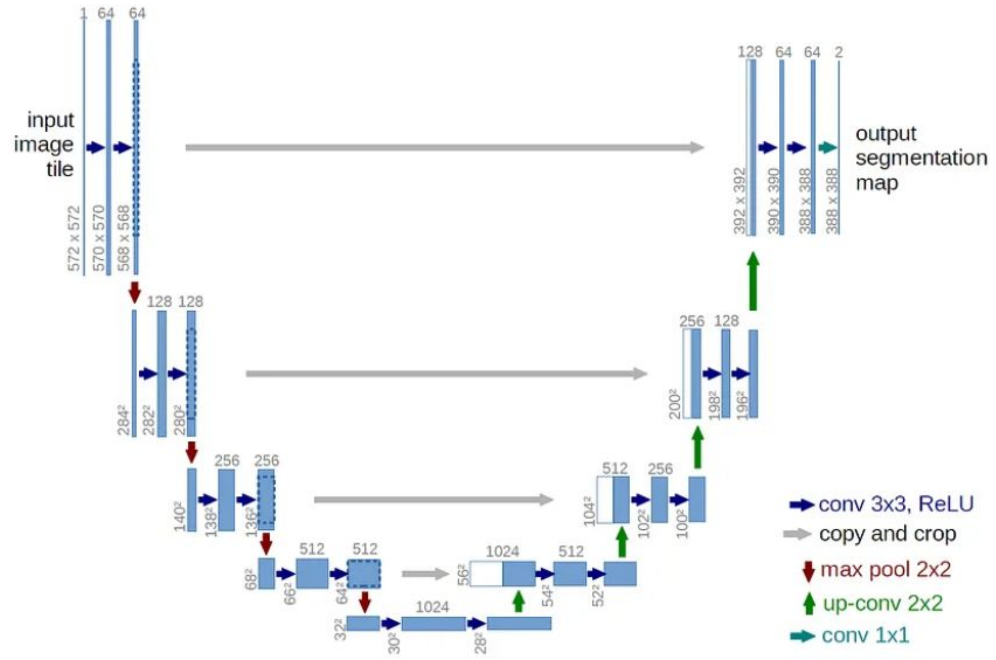


Fig — 4: Example of a Basic U-Net Architecture (Ref:- https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/u-net-architecture.png)

Input_size: (1,1,32,32)
Downsize layer: 5
Upsize layer: 5
# of feature: 4
kernel size: 4
stride: 2
Dropout: 0.5
activation: Leaky relu
Output: (1,1,32,32)

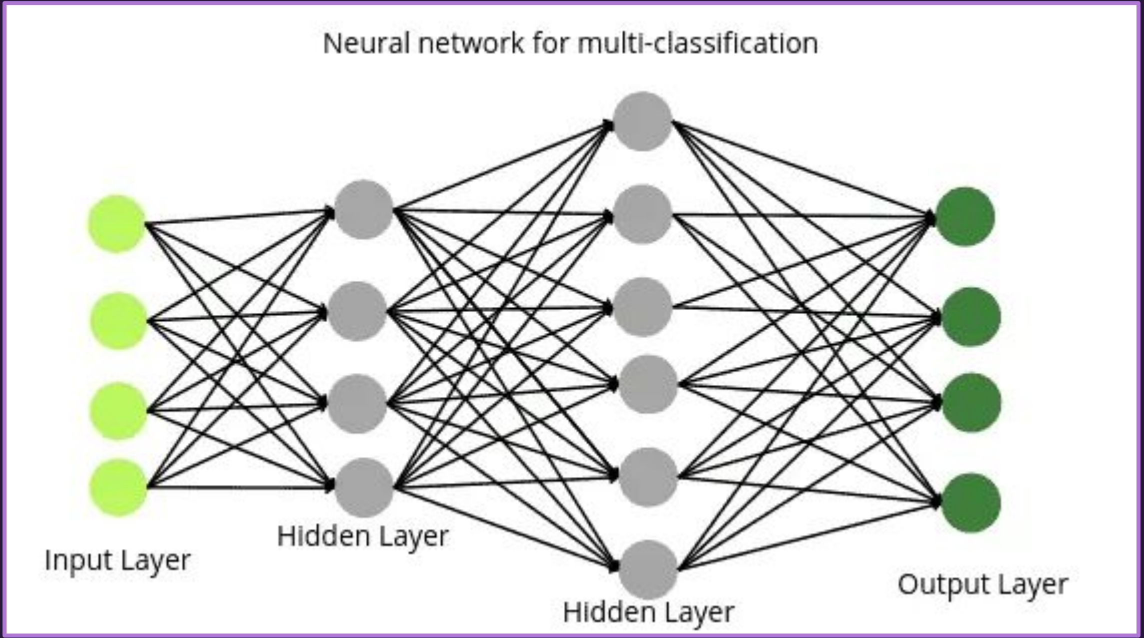
Input Size (1, 1, 32, 32)

Output Size (1, 1, 32, 32)

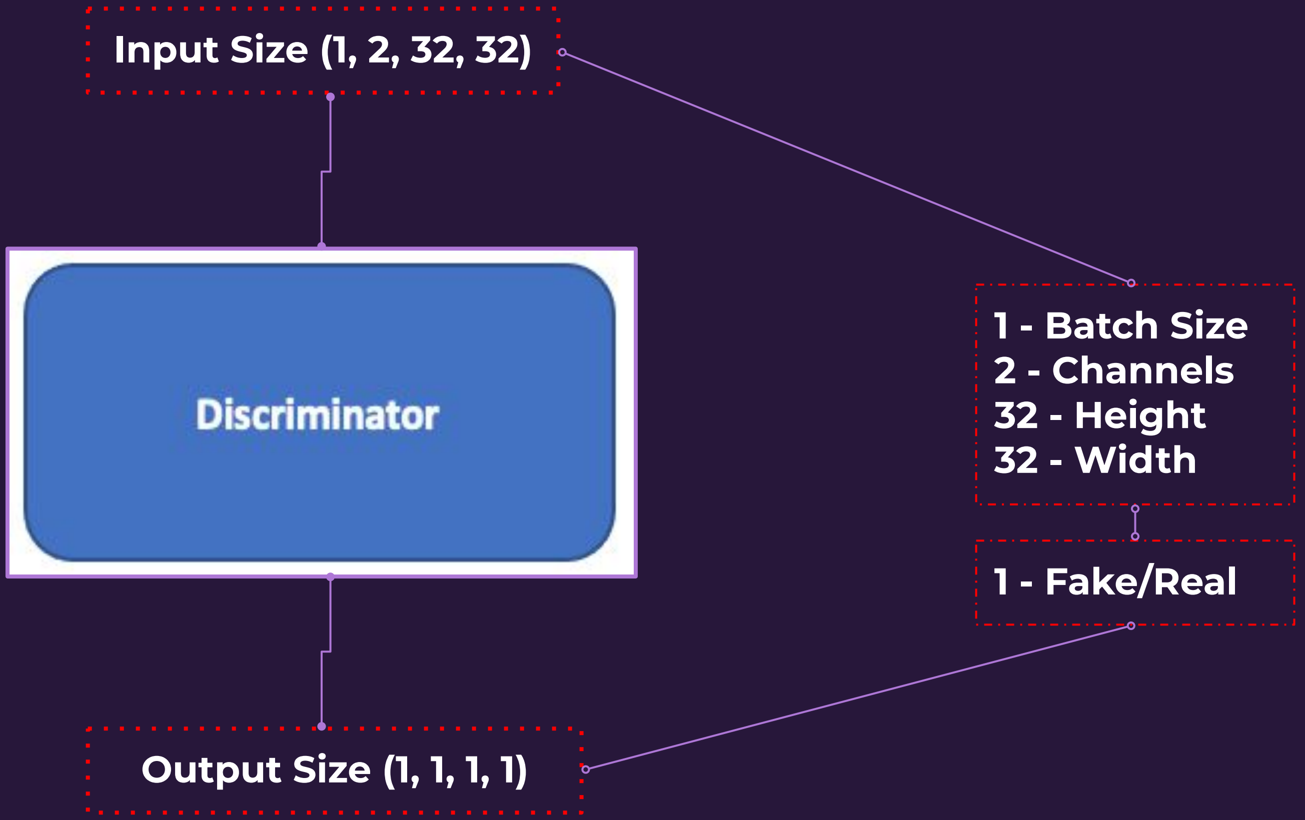
1 - Batch Size
1 - Channel
32 - Height
32 - Width

Matching

Discriminator Design



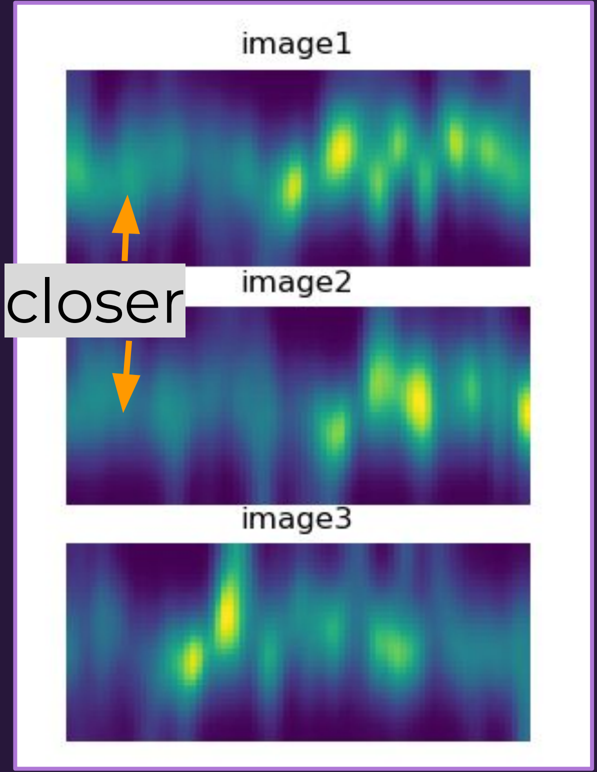
Input_size = (1,2,32,32)
Hidden layer: 4
Layer type: Cov2d
Activation: Leaky Relu
Output: (1,1,1,1)

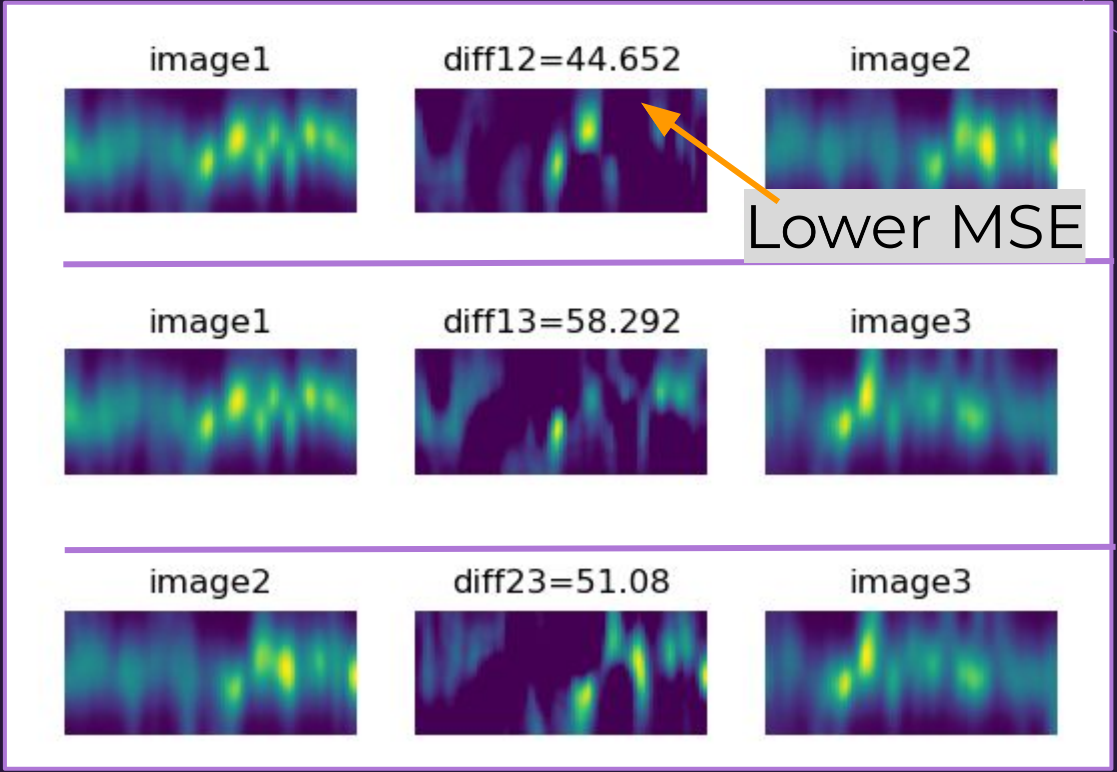

Input Size (1, 2, 32, 32)
Discriminator
Output Size (1, 1, 1, 1)
1 - Batch Size
2 - Channels
32 - Height
32 - Width
1 - Fake/Real

Dataset and Training Configuration

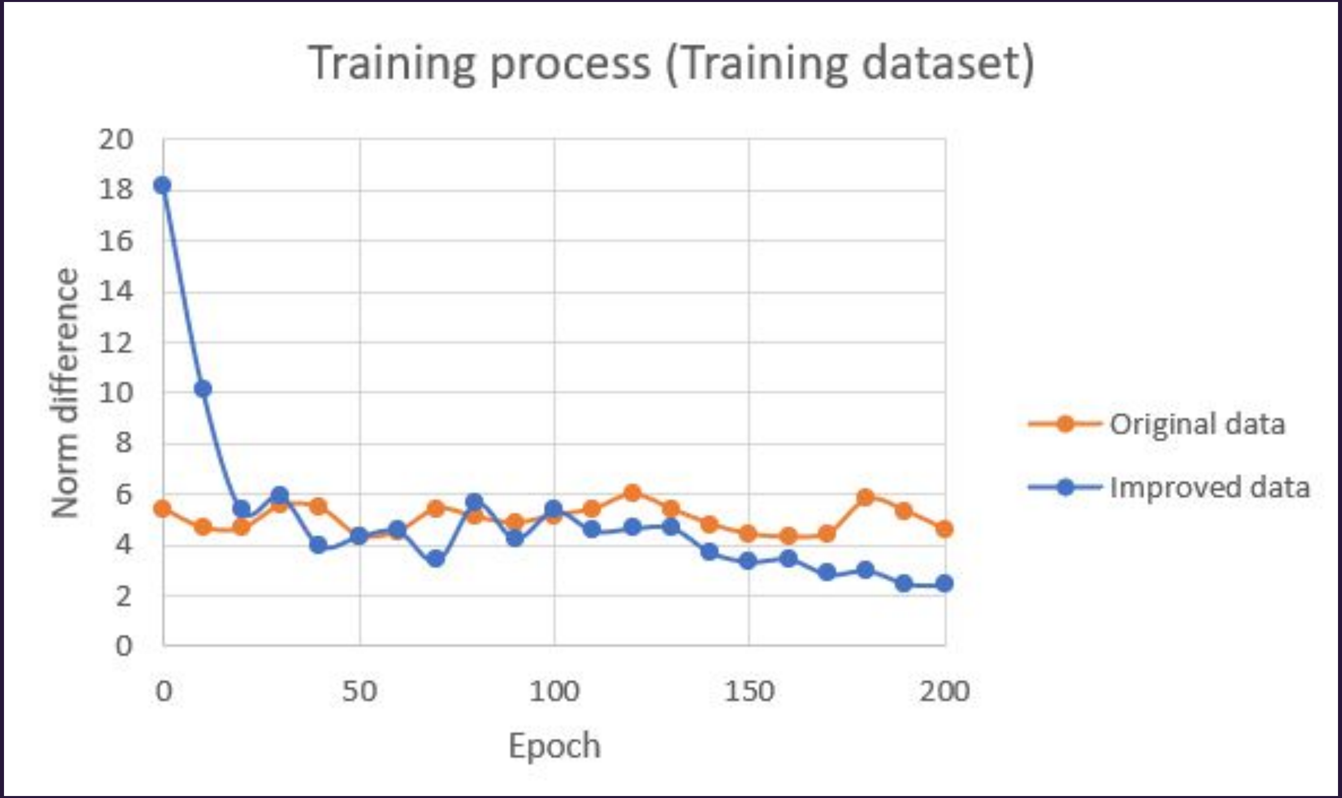
- Training dataset: (5 motion)
    - Total data length: 414 (cuts)
    - Training and validation data split : (400,14)
- Testing dataset: (2 unseen motion)
    - Total data length: 36 (cuts)
- Training config.
    - Batch size: 16
    - Epoch: 200
    - Platform: Local CPU

Metrics-MSE (L2 Norm)

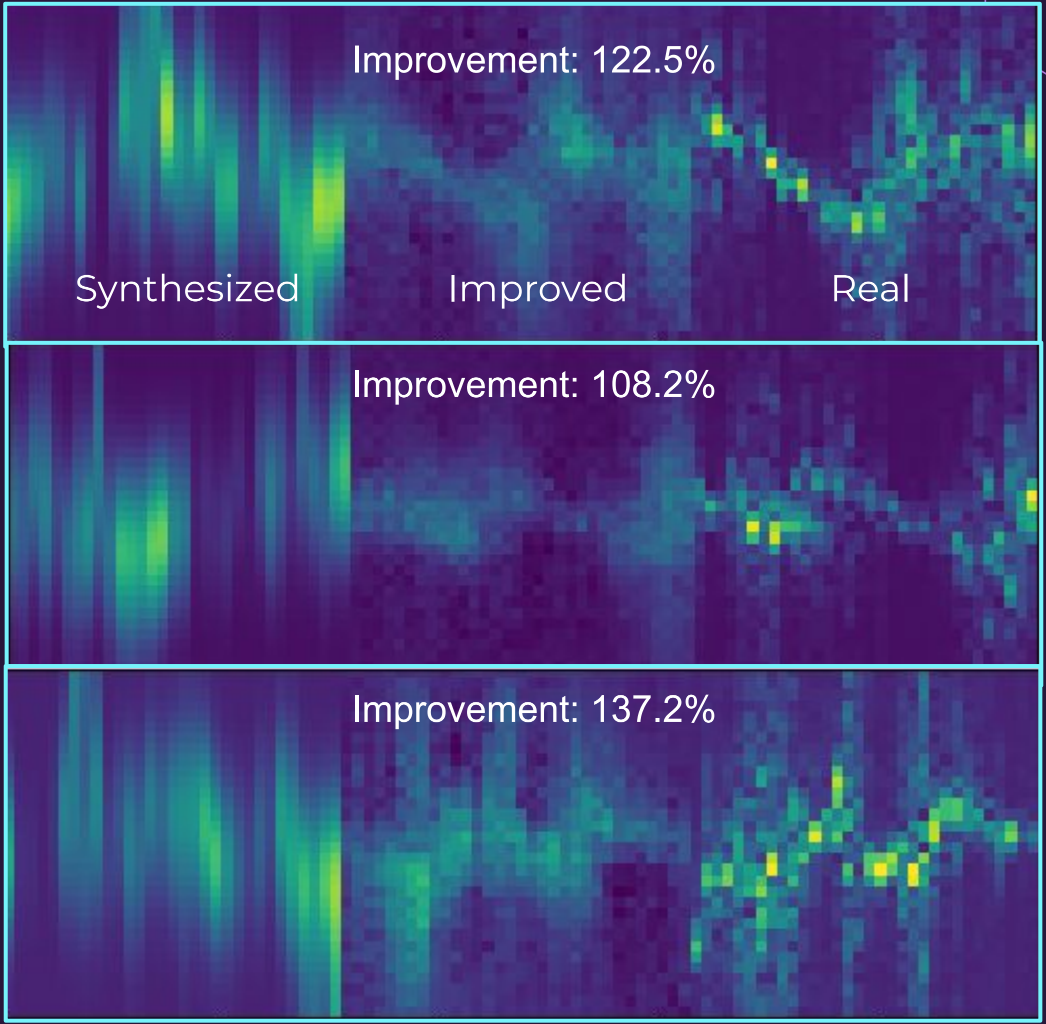




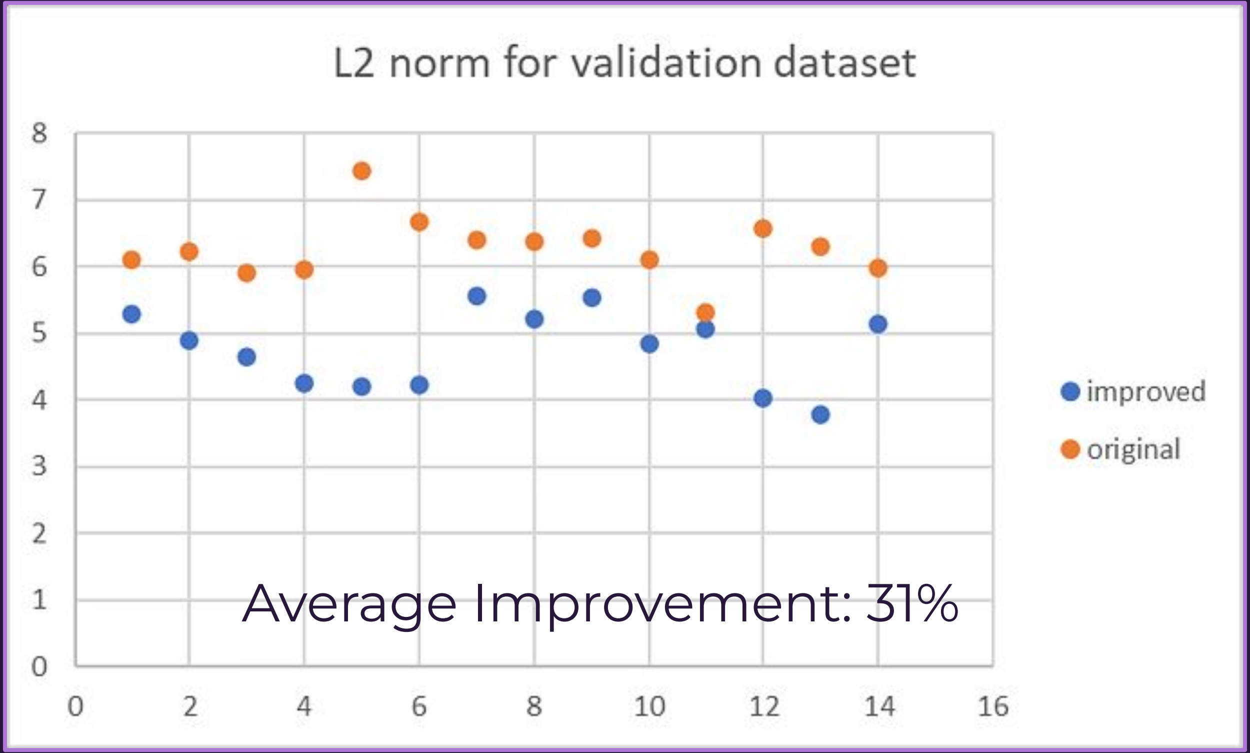
Training Result



Training Data Result

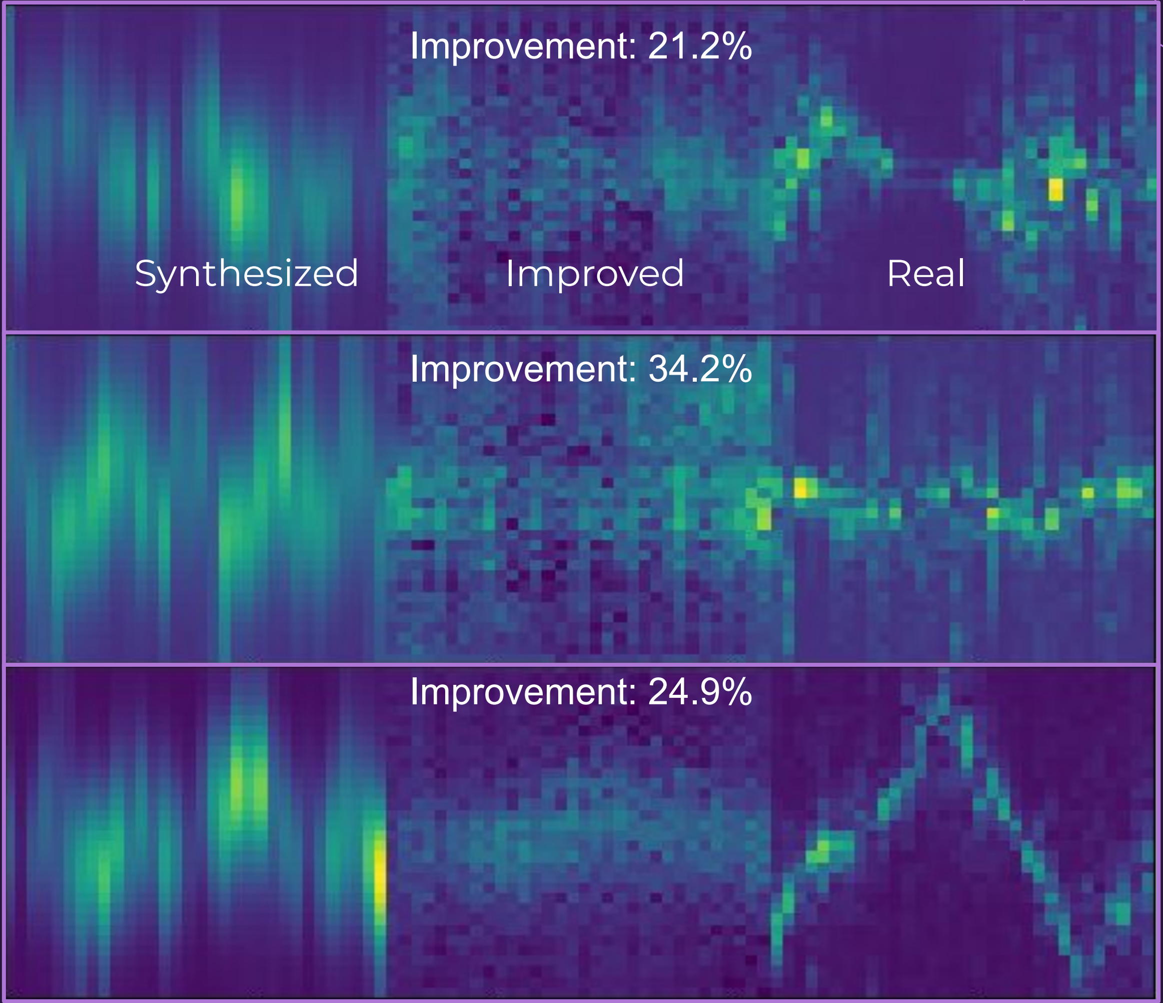


Validation Result

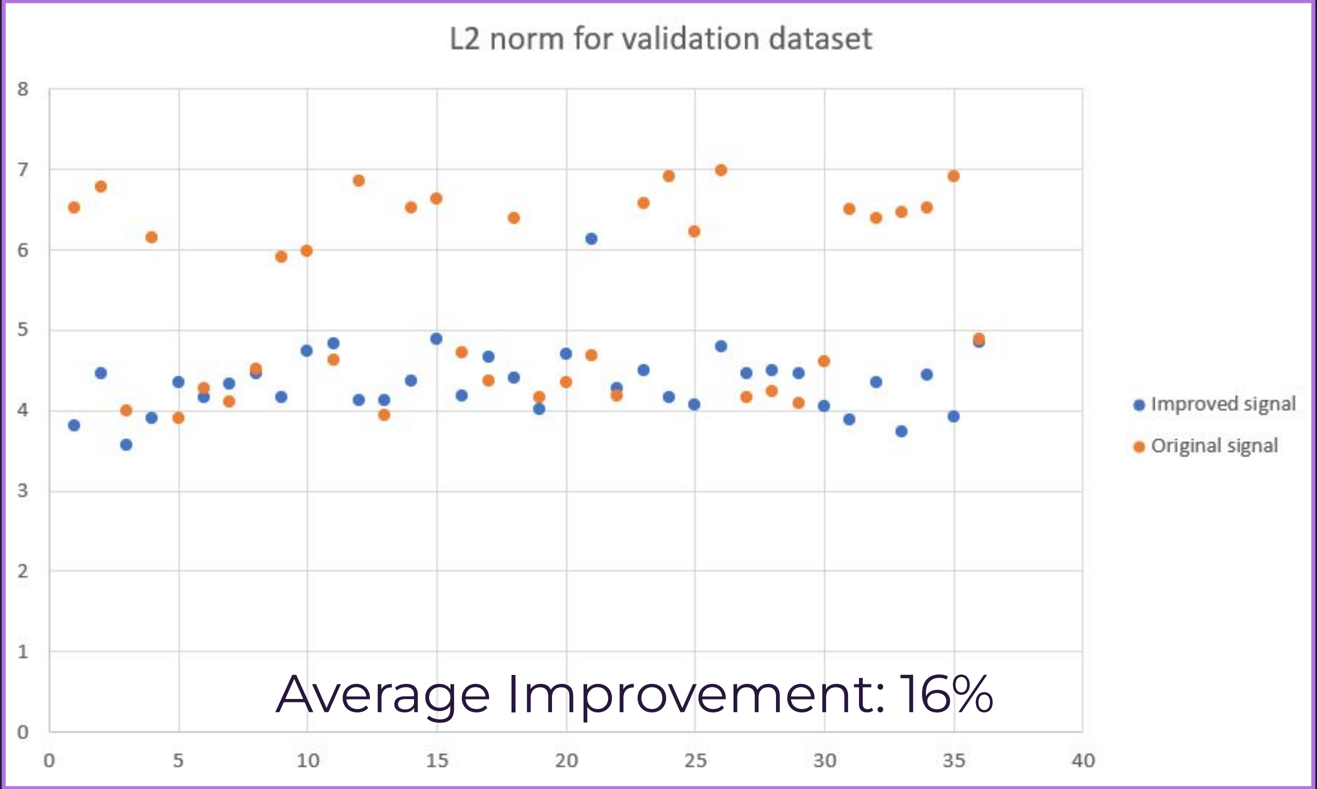


Average Improvement: 31%

Validation Data Result

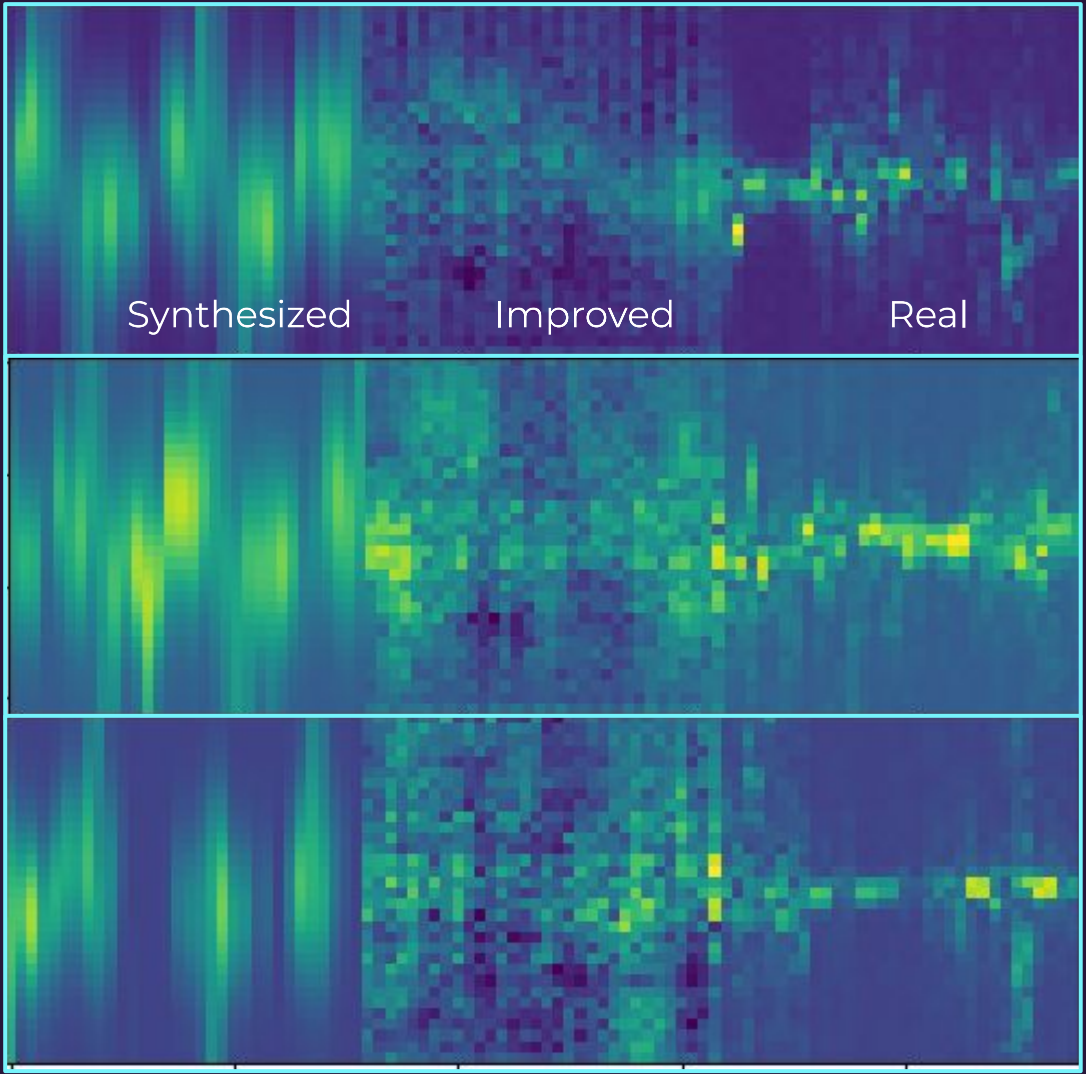


Testing Result



Average Improvement: 16%

Testing Data Result



Conclusion

- We proved GAN can be used for improving signal quality synthesized from video, which make it possible to take advantage of huge activity video dataset online
- For the unseen videos, the quality improvement is not as clear, this can due to lack of training data

Future Work

- Adapting data agumentation method to increase the training dataset
- Implement time alignment algorithm to avoid time difference between two source
- Test different generative network architecture
- Including “range” dimension
- Implementing HAR network on unseen activity

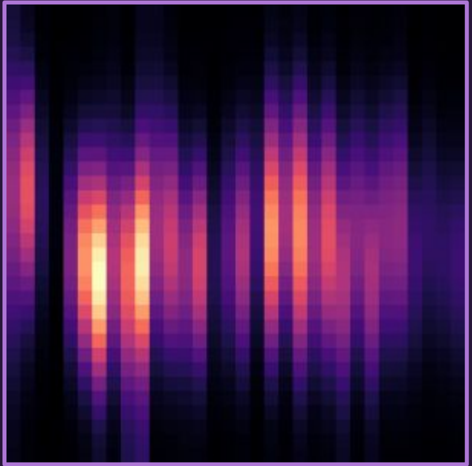
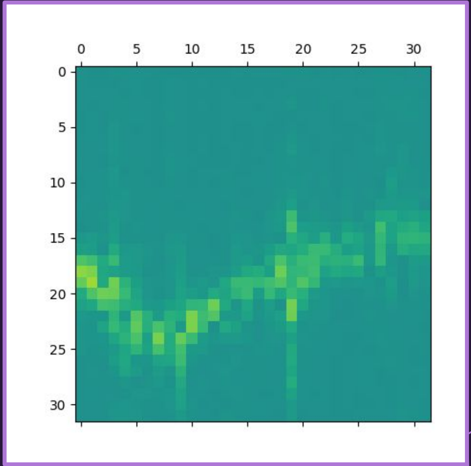
Comparison the Accuracy between trained CNN model with Sythesized Data,

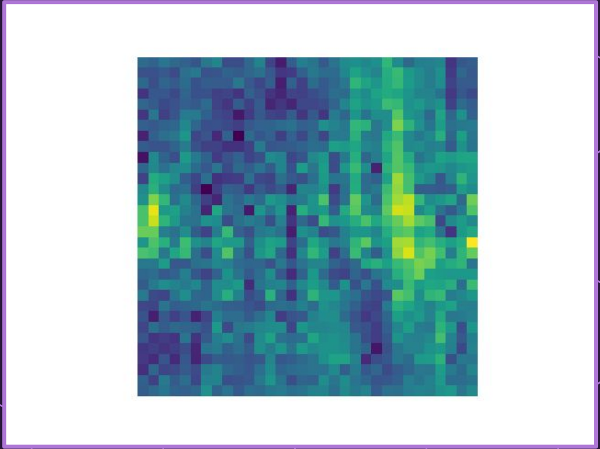
Real Radar Signal and GAN model

Sythesized Data:

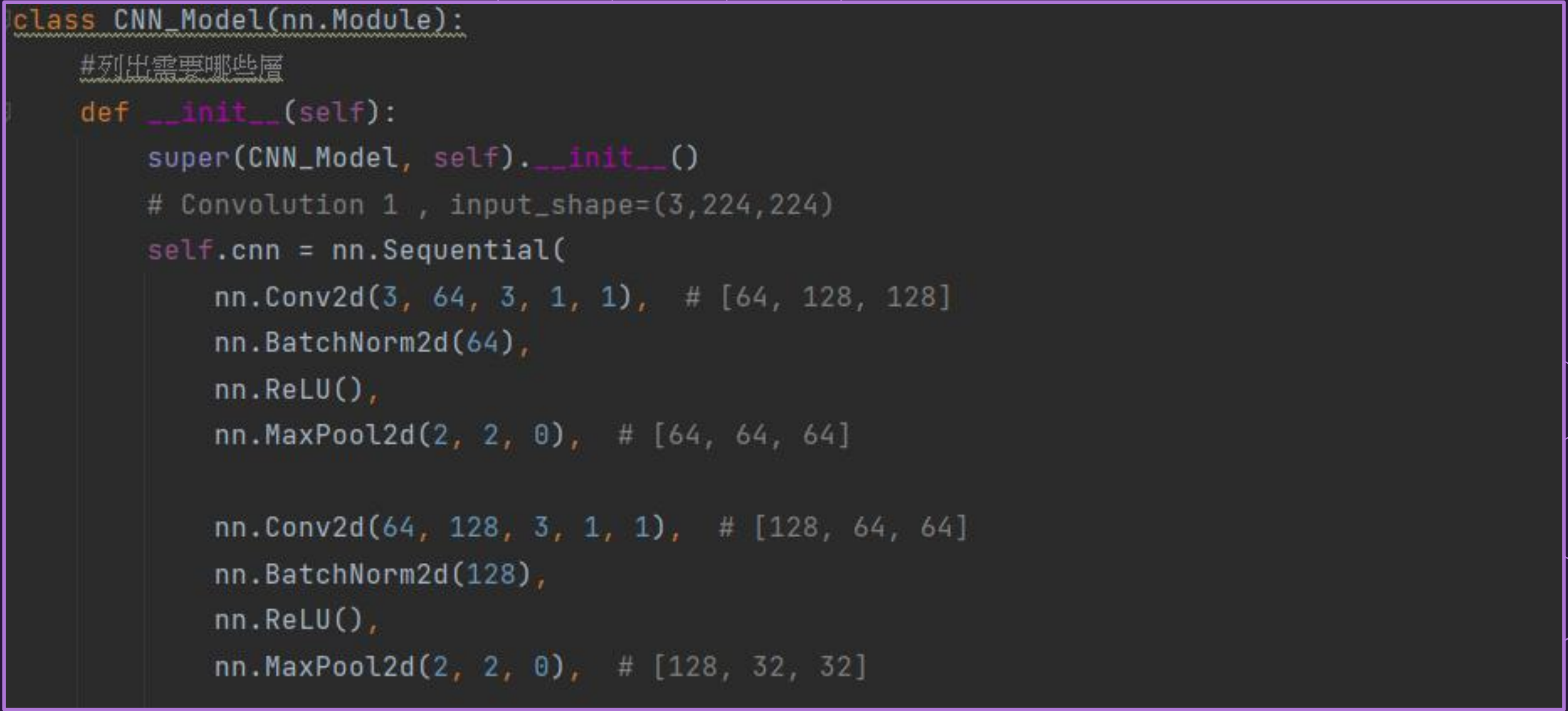
Real Radar Data:

GAN Model Data:



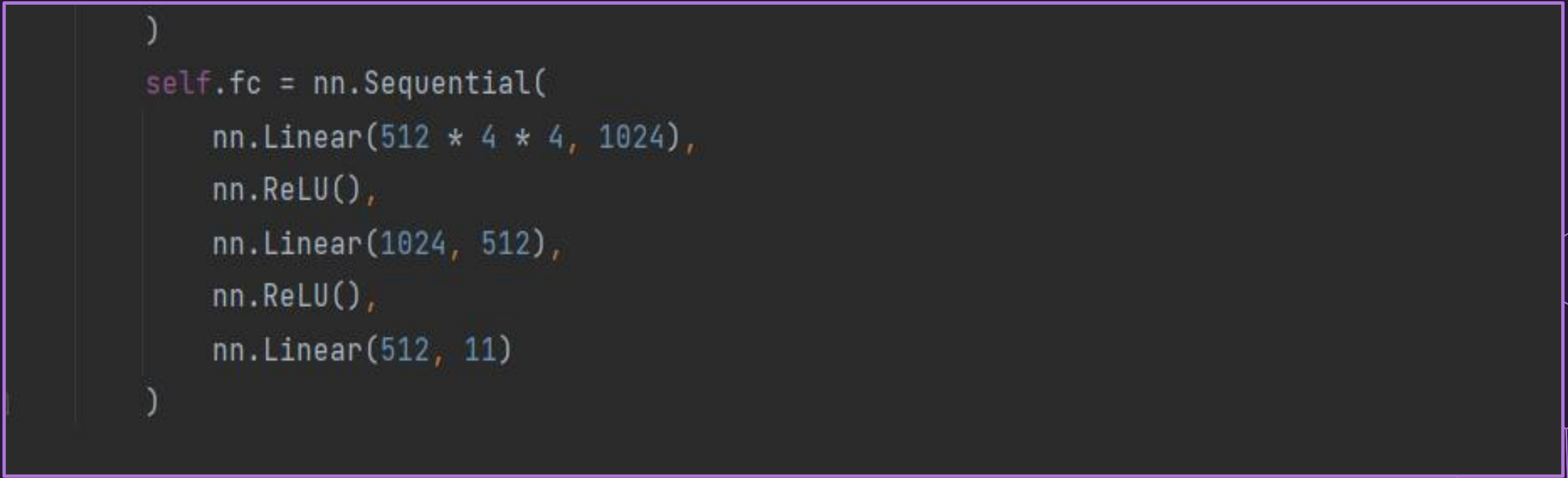
Using the same CNN architecture layers to train these three types of Data Sources:



```
class CNN_Model(nn.Module):
    #列出需要哪些層
    def __init__(self):
        super(CNN_Model, self).__init__()
        # Convolution 1 , input_shape=(3,224,224)
        self.cnn = nn.Sequential(
            nn.Conv2d(3, 64, 3, 1, 1),  # [64, 128, 128]
            nn.BatchNorm2d(64),
            nn.ReLU(),
            nn.MaxPool2d(2, 2, 0),  # [64, 64, 64]

            nn.Conv2d(64, 128, 3, 1, 1),  # [128, 64, 64]
            nn.BatchNorm2d(128),
            nn.ReLU(),
            nn.MaxPool2d(2, 2, 0),  # [128, 32, 32]
```

Using the same CNN architecture layers to train these three types of Data Sources:

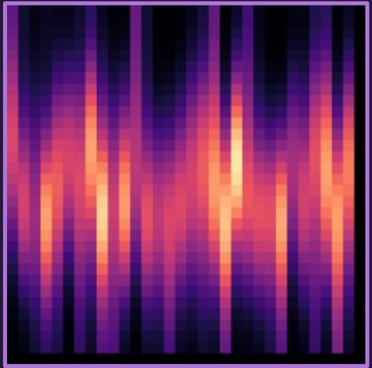


```
    )
    self.fc = nn.Sequential(
        nn.Linear(512 * 4 * 4, 1024),
        nn.ReLU(),
        nn.Linear(1024, 512),
        nn.ReLU(),
        nn.Linear(512, 11)
    )
```

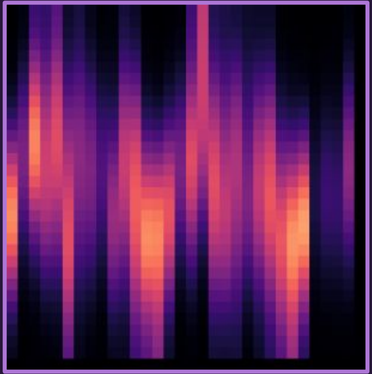
Training Result (using Sythesized Data)

Total five Actions to Recognize:

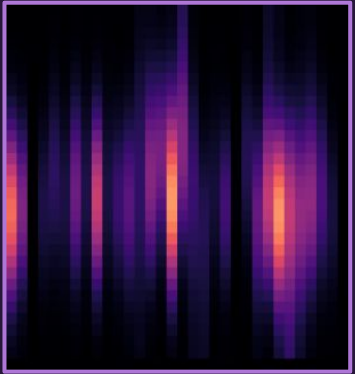
Handmotion:



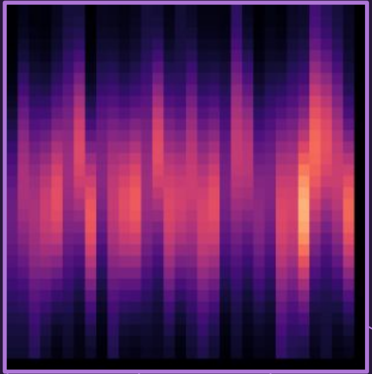
JJ:



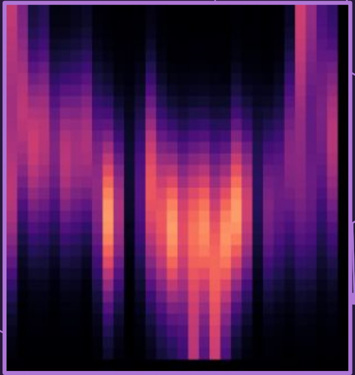
Lunge:



Run:

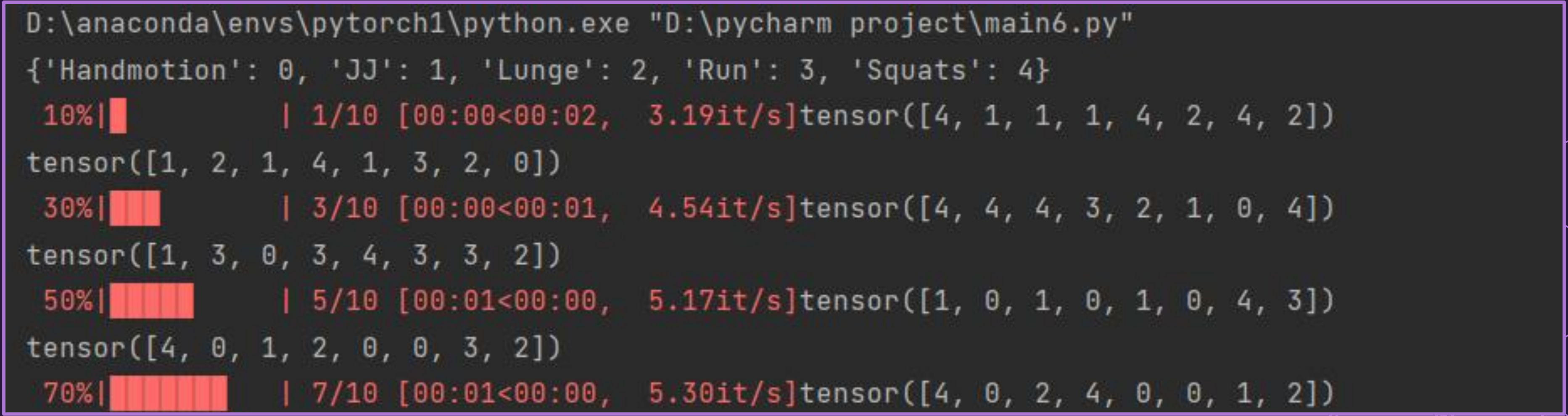


Squats:



Labeling:

Handmotion: 0 , JJ : 1 , Lunge : 2 , Run : 3, Squats : 4

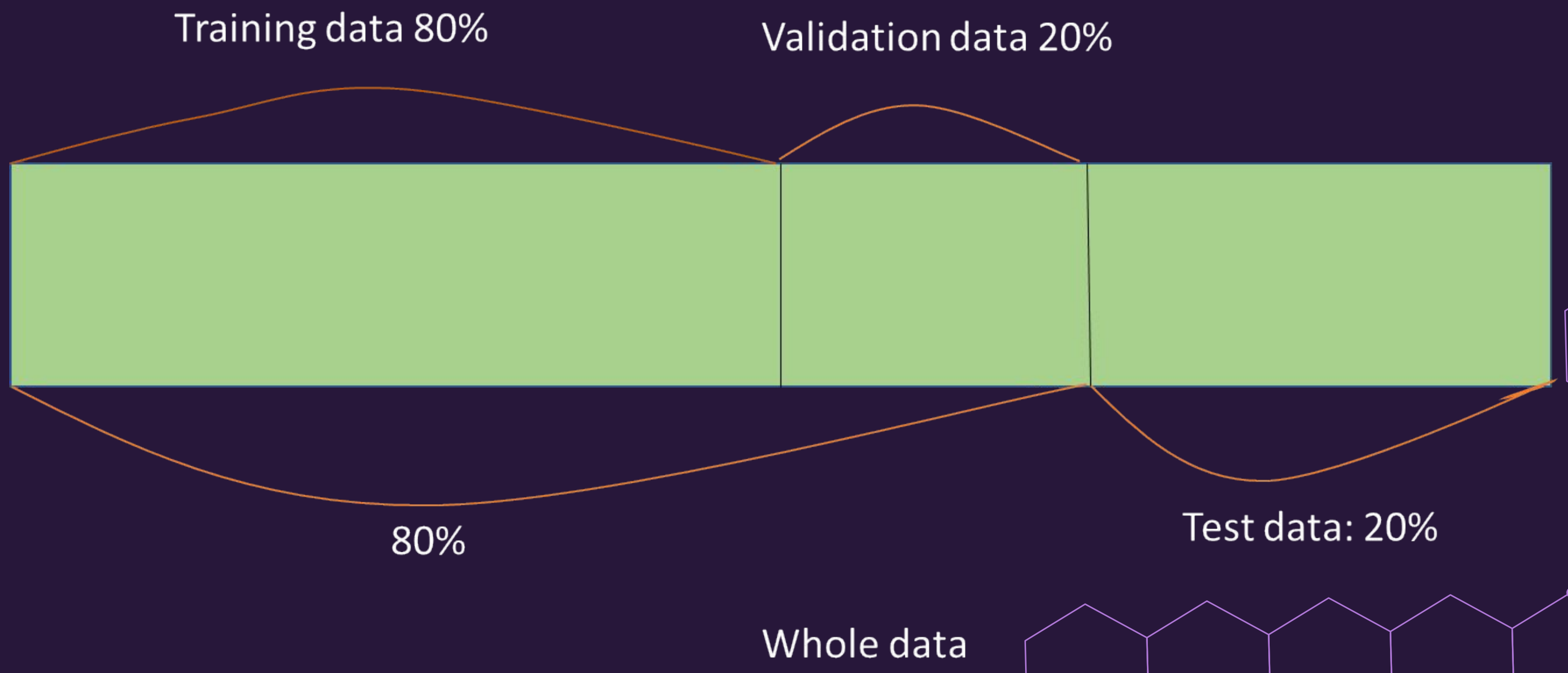


```
D:\anaconda\envs\pytorch1\python.exe "D:\pycharm project\main6.py"
{'Handmotion': 0, 'JJ': 1, 'Lunge': 2, 'Run': 3, 'Squats': 4}
 10%|█         | 1/10 [00:00<00:02,  3.19it/s]tensor([4, 1, 1, 1, 4, 2, 4, 2])
tensor([1, 2, 1, 4, 1, 3, 2, 0])
 30%|███       | 3/10 [00:00<00:01,  4.54it/s]tensor([4, 4, 4, 3, 2, 1, 0, 4])
tensor([1, 3, 0, 3, 4, 3, 3, 2])
 50%|█████     | 5/10 [00:01<00:00,  5.17it/s]tensor([1, 0, 1, 0, 1, 0, 4, 3])
tensor([4, 0, 1, 2, 0, 0, 3, 2])
 70%|███████   | 7/10 [00:01<00:00,  5.30it/s]tensor([4, 0, 2, 4, 0, 0, 1, 2])
```

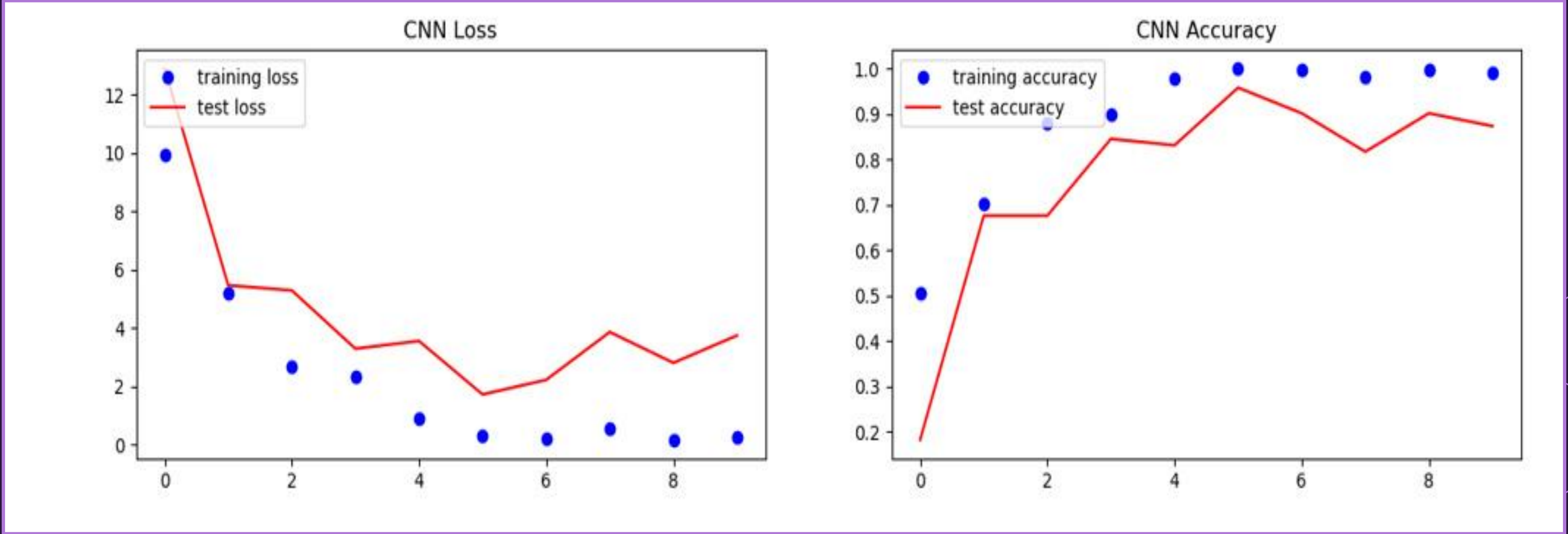
Training Result:

We trained the sythesized CNN model using 20% of the data as test data. For the remaining 80% data sets, we split 20% of those training data into validation data and use the remaining 80% for training.

Splitting the Data:

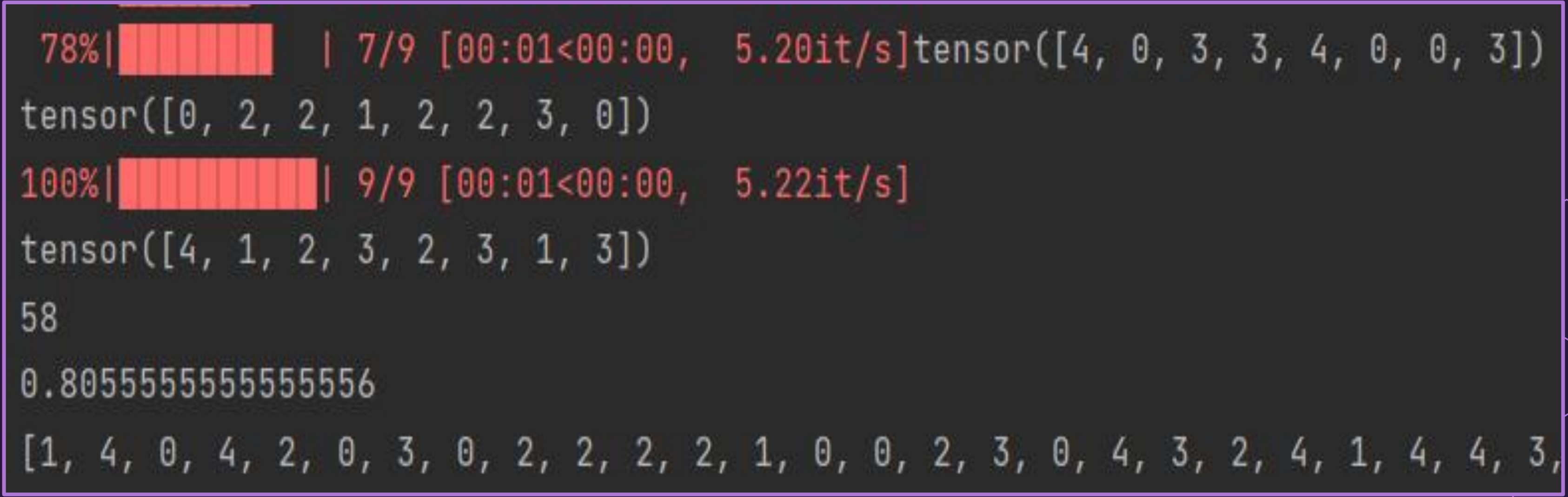


Training Result:



Testing Accuracy:

Testing accuracy about: 80.5 %

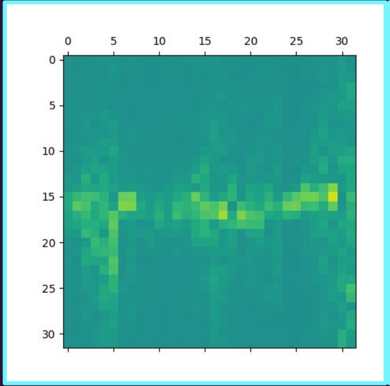


```
 78%|███████   | 7/9 [00:01<00:00,  5.20it/s]tensor([4, 0, 3, 3, 4, 0, 0, 3])
tensor([0, 2, 2, 1, 2, 2, 3, 0])
100%|██████████| 9/9 [00:01<00:00,  5.22it/s]
tensor([4, 1, 2, 3, 2, 3, 1, 3])
58
0.8055555555555556
[1, 4, 0, 4, 2, 0, 3, 0, 2, 2, 2, 2, 1, 0, 0, 2, 3, 0, 4, 3, 2, 4, 1, 4, 4, 3,
```

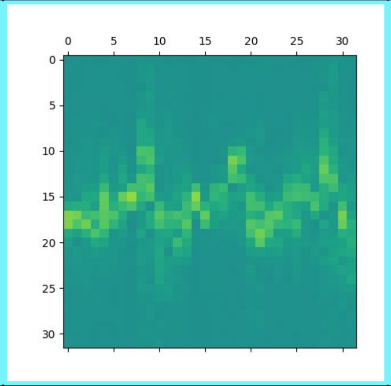
Training Result (using Real Radar Signal Data)

Total five Actions to Recognize:

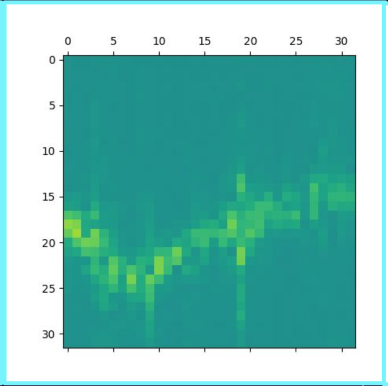
Handmotion:



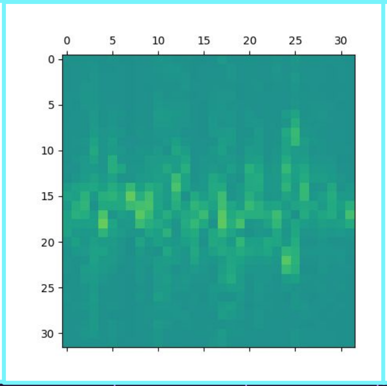
JJ:



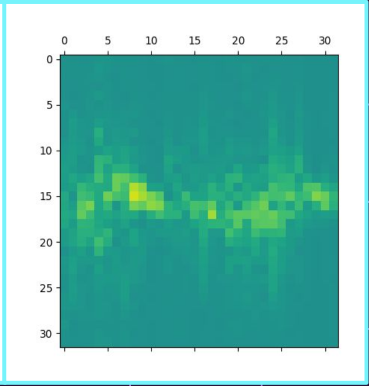
Lunge:



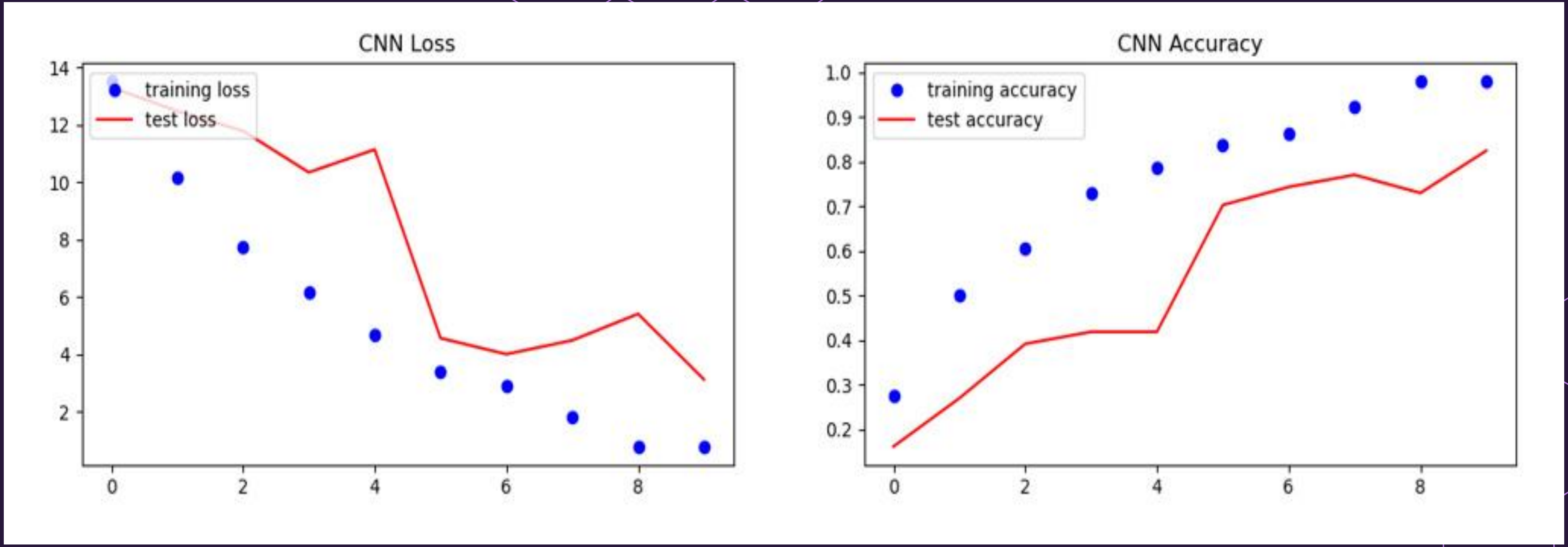
Run:



Squats:

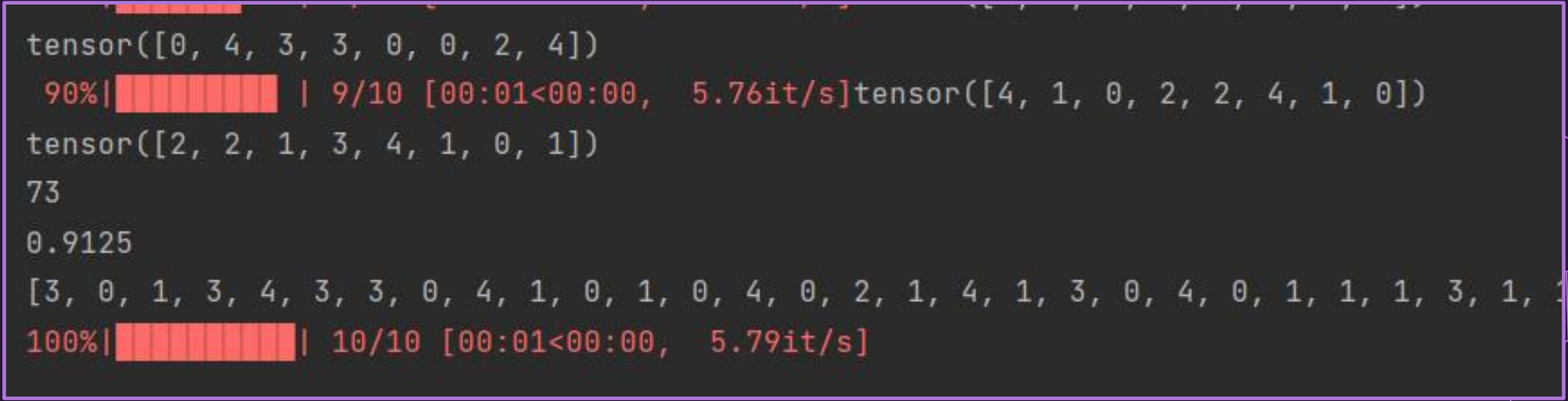


Training Result:



Testing Accuracy (Real Radar):

Testing accuracy about: 91.25 %

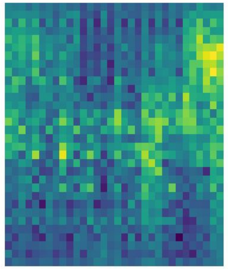
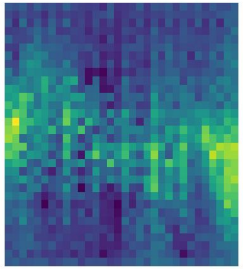
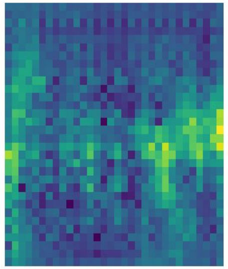
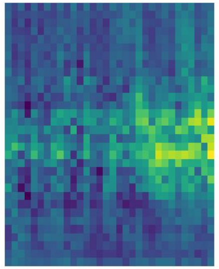
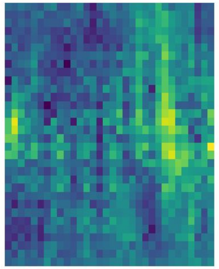


```
tensor([0, 4, 3, 3, 0, 0, 2, 4])
 90%|█████████ | 9/10 [00:01<00:00,  5.76it/s]tensor([4, 1, 0, 2, 2, 4, 1, 0])
tensor([2, 2, 1, 3, 4, 1, 0, 1])
73
0.9125
[3, 0, 1, 3, 4, 3, 3, 0, 4, 1, 0, 1, 0, 4, 0, 2, 1, 4, 1, 3, 0, 4, 0, 1, 1, 1, 3, 1,
100%|██████████| 10/10 [00:01<00:00,  5.79it/s]
```

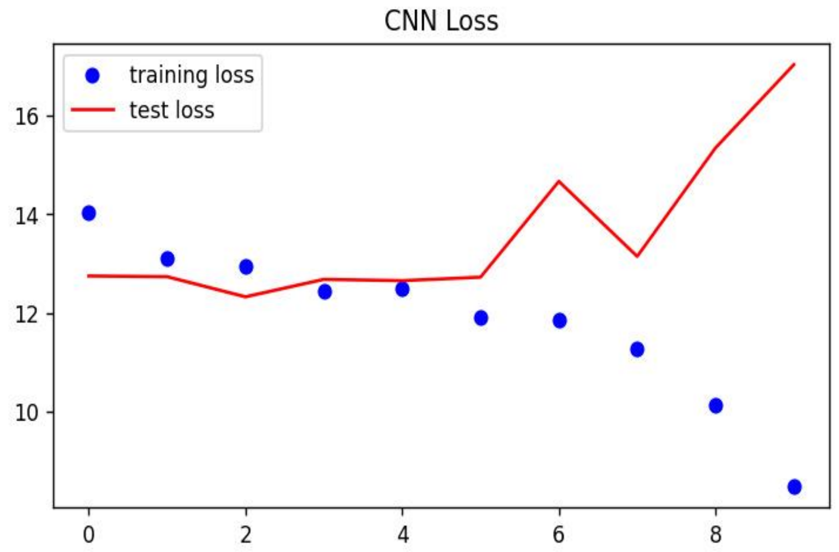
Training Result (using GAN Data)

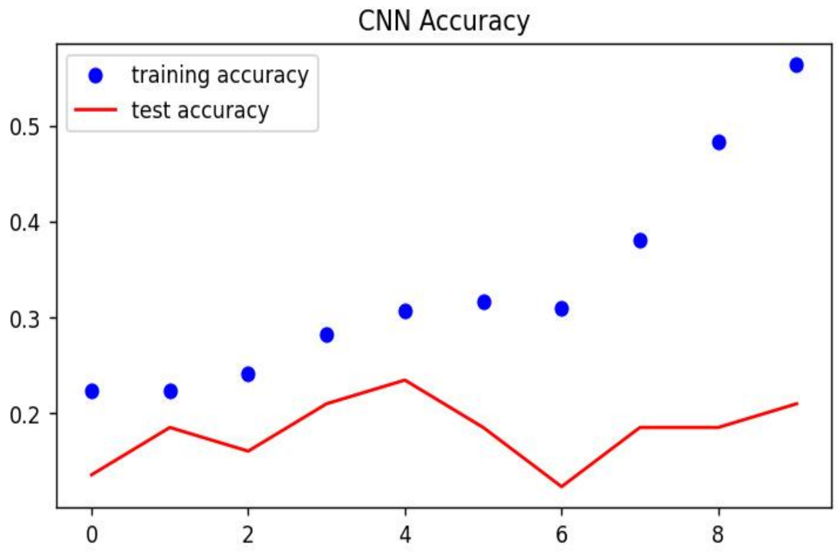
Total five Actions to Recognize:

Handmotion: JJ: Lunge: Run: Squats:

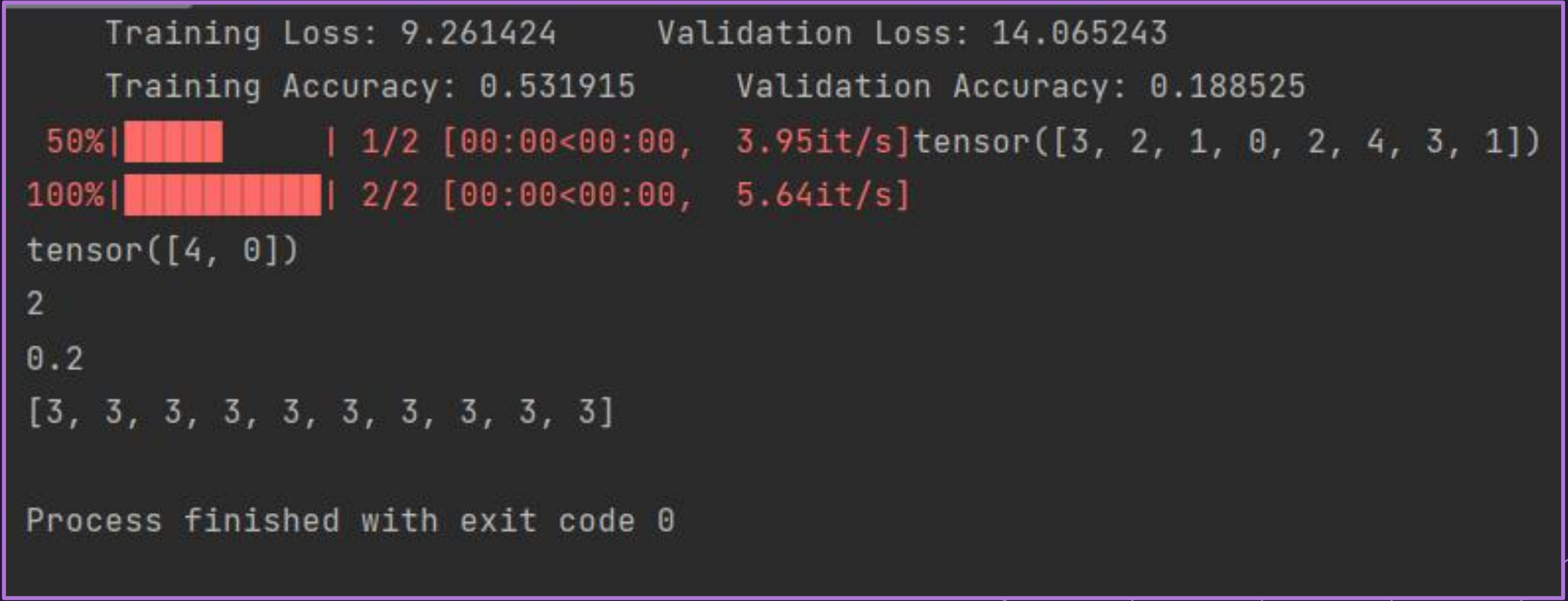
Training Result:





Testing Accuracy (GAN):

Testing accuracy about: 20 %



```
    Training Loss: 9.261424     Validation Loss: 14.065243
    Training Accuracy: 0.531915     Validation Accuracy: 0.188525
 50%|█████     | 1/2 [00:00<00:00,  3.95it/s]tensor([3, 2, 1, 0, 2, 4, 3, 1])
100%|██████████| 2/2 [00:00<00:00,  5.64it/s]
tensor([4, 0])
2
0.2
[3, 3, 3, 3, 3, 3, 3, 3, 3, 3]

Process finished with exit code 0
```

Comparison Accuracy:

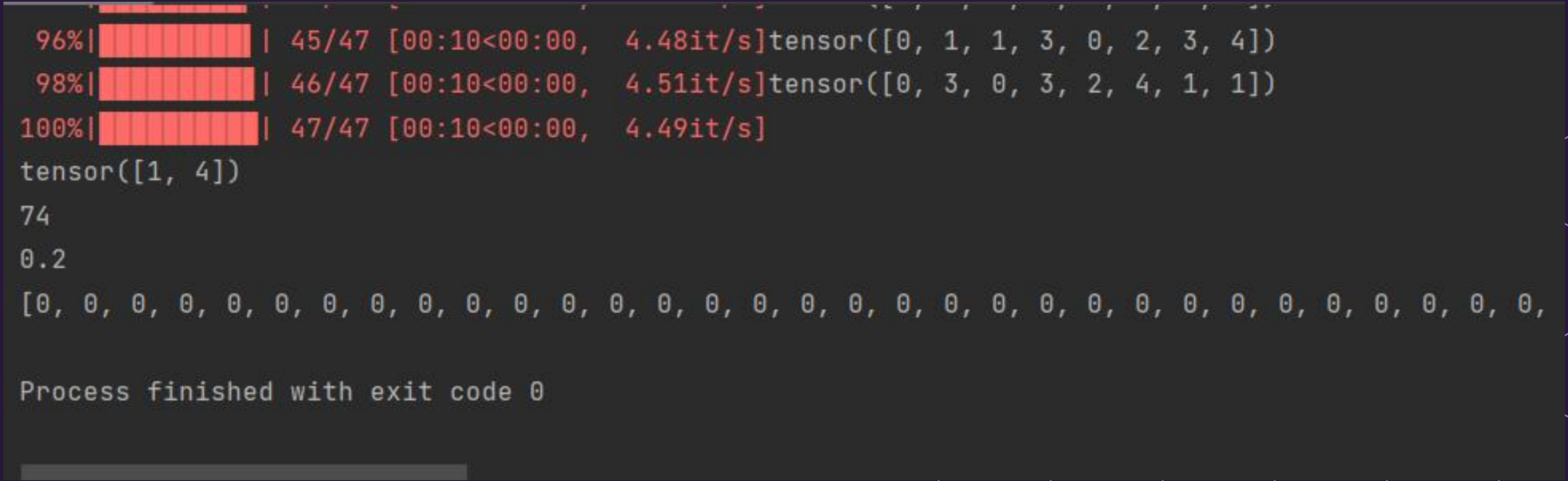
CNN model trained by Sythesized Data: 80.5%

CNN model trained by Real Radar Signal Data: 91.25%

CNN model trained by GAN Data: 20%

Using trained CNN model with Synthesized Data to test the Real Radar Signal

Accuracy: 20%

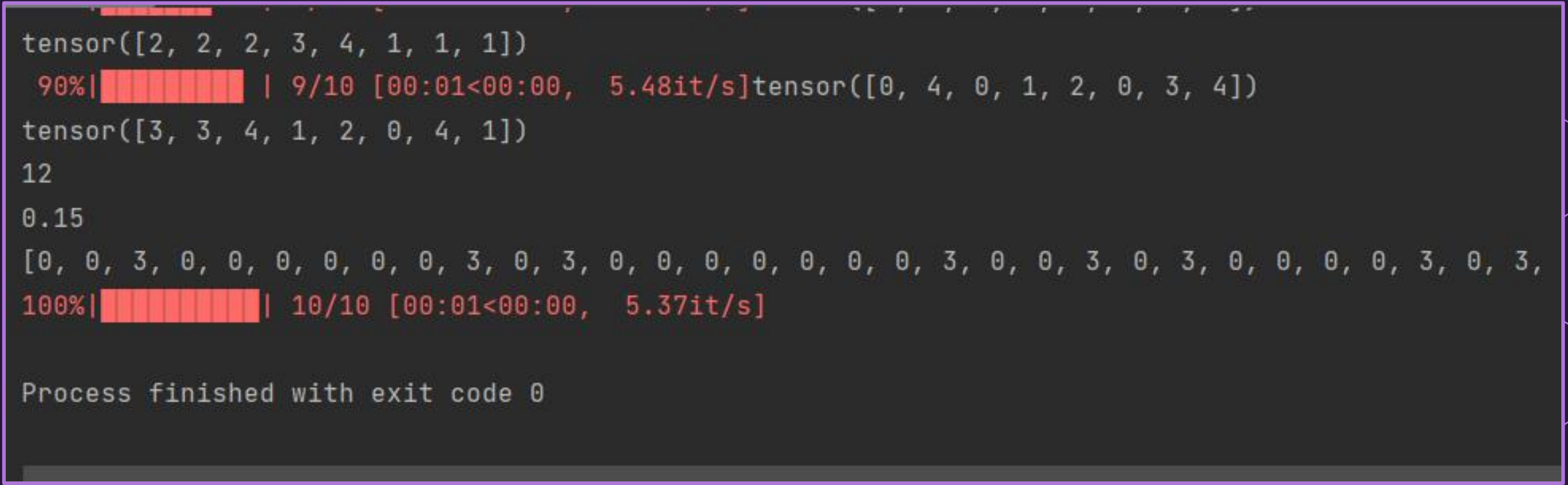


```
 96%|██████████ | 45/47 [00:10<00:00,  4.48it/s]tensor([0, 1, 1, 3, 0, 2, 3, 4])
 98%|██████████| 46/47 [00:10<00:00,  4.51it/s]tensor([0, 3, 0, 3, 2, 4, 1, 1])
100%|██████████| 47/47 [00:10<00:00,  4.49it/s]
tensor([1, 4])
74
0.2
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

Process finished with exit code 0
```

Using trained CNN model with GAN Data to test the Real Radar Signal

Accuracy: 15%



```
tensor([2, 2, 2, 3, 4, 1, 1, 1])
 90%|█████████ | 9/10 [00:01<00:00,  5.48it/s]tensor([0, 4, 0, 1, 2, 0, 3, 4])
tensor([3, 3, 4, 1, 2, 0, 4, 1])
12
0.15
[0, 0, 3, 0, 0, 0, 0, 0, 0, 3, 0, 3, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 3, 0, 3, 0, 0, 0, 0, 3, 0, 3,
100%|██████████| 10/10 [00:01<00:00,  5.37it/s]

Process finished with exit code 0
```

Conclusion:

When training our CNN model with GAN data, our CNN model overfits. The reason maybe lack of training data.

Another reason maybe that the data generated by GAN has more features added to the PNG image, which makes it harder to distinguish the different actions compared to the data generated by the other two sources.

Future work: to improve and adjust our CNN model layers to train the data produced by GAN successfully.

Issues/Challenges/Roadblocks

- Channel mismatch during Training
- Operands Broadcast Error with Shapes
- Matching Tensor Size in accordance with Input-Output Combinations including Batch Size, Channels, Height and Width for Discriminator-Generator Blocks
- Adjusting Conv2D layers for efficient results

D. Project Status Update

Update of Team Member’s Technical Responsibilities

- **Upanshu Srivastava**
  - Collect and Label Video and Radar Data
  - Build and Train Generative Model (Pix2Pix)
- **Yinchien Haung**
  - Construct Mesh from Video and Simulate Radar Signal
  - Collect and Process the Signals from the Real Radar
  - Build and Train Generative Model
- **Ruijie Song**
  - Process the Signals from the Real Radar
- **Jin-Hau Yeh**
  - Build and Train CNN Model for Human Activity Recognition
- **All group members**
  - Study Radar Signal Formulation
  - Visualizing Neural Network and Training Process

Update of Team Member's Management Responsibilities

- **Technical Project Manager (TPM) — Yin chien Huang**
    - Divide the whole project into specific tasks and assign to every team members
    - Track the progress of each tasks
    - Coordinate meetings
- **Document Coordinator (DC) — Upanshu Srivastava**
    - Restore and manage the video and radar data
    - Edit and proofread materials for reports/presentations
- **Business Office Liaison (BOL) — Ruijie Song**
    - Help purchasing and managing hardware equipments

Update of Project Plan (Milestones)

1. Collecting videos of motion and real-world radar signals, label those data for training. ✔
Milestone 1: Obtain labelled motion videos and real-world radar signals from motions. ✔

2a. Constructing mesh and Simulate radar signal ✔
Milestone 2a: Obtain simulated velocity Histogram plot. ✔

2b. Process the real signals from radar ✔
Milestone 2b: Obtain real velocity Histogram plot. ✔

3. Building basic generative model ✔
Milestone 3: Finish the code for the generative model structure and loss function validation. ✔

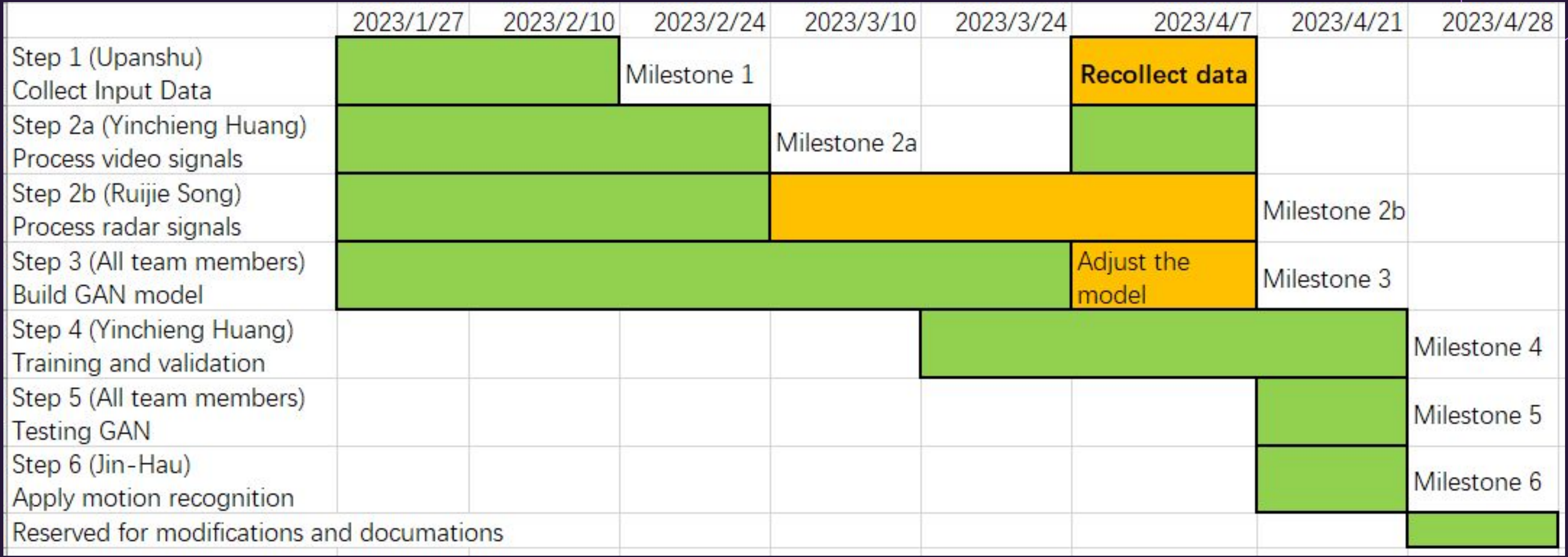
Update of Project Plan (Milestones)

4. Training and cross validation of the generativemodel. ✔
Milestone 4: Obtain the accuracy of our generative model. ✔

5. Testing, debugging and improving our generative model.
Milestone 5: Obtain refined radar signals which are ready for the motion recognition.

6. Apply the motions recognition model to the refined signals.
Milestone 6: Observe correct rate of each NN

Update of Project Plan (GANTT Chart)



	2023/1/27	2023/2/10	2023/2/24	2023/3/10	2023/3/24	2023/4/7	2023/4/21	2023/4/28
Step 1 (Upanshu) Collect Input Data	🟩	🟩	Milestone 1			🟧 **Recollect data**		
Step 2a (Yinchieng Huang) Process video signals	🟩	🟩	🟩	Milestone 2a		🟩		
Step 2b (Ruijie Song) Process radar signals	🟩	🟩	🟩	🟧	🟧	🟧	Milestone 2b	
Step 3 (All team members) Build GAN model	🟩	🟩	🟩	🟩	🟩	🟧 Adjust the model	Milestone 3	
Step 4 (Yinchieng Huang) Training and validation					🟩	🟩	🟩	Milestone 4
Step 5 (All team members) Testing GAN							🟩	Milestone 5
Step 6 (Jin-Hau) Apply motion recognition							🟩	Milestone 6
Reserved for modifications and documations								🟩

Issues/Challenges/Roadblocks

1. Synchronize real and synthesized radar signal
   Solution: We start the camera recording and radar recording at the same time
2. Adjust the input size for both real and synthesized signal to be the same
   Solution: We format the processed real and synthesized signals to 32*32 matrices.
3. ~~Estimate camera angle from video~~
4. ~~Create a cascade training dataset of 32x32~~
5. ~~Design Pix2Pix GAN model in accordance to input-output of 32x32 matrix~~

E. Contributions

- Upanshu Srivastava (PPT Section - A, B, C, E; Developing and Training Pix2Pix GAN Model)
- Ruijie Song (PPT Section - B, C, D; Project Status Update)
- Yinchien Huang (PPT Section - B, C; Operating Radar system, Convert Video to Radar Signal, Signal Processing of Radar Signal, Training the GAN Model)
- Jin-Hau Yeh (PPT Section - B, C; Developing Human Activity Machine Learning Model)

Ideas to Innovation – Spring ‘23 Final Report

A. Technical Plan for Prototype

The functionalities of the prototype revolve around the Generative Adversarial Network (GAN), where the model imitates the synthesized RADAR signal to the real RADAR signal. Creating the minimum loss between the former and the latter, utilizing the L2 Norm. Hence, delivering the improved signal. Moreover, using the trained Generative Adversarial Network (GAN) generates the RF Signal dataset through unseen activity video dodging the Real RADAR Signal. The above process leads to three different datasets of Synthesized RADAR Signal, Real RADAR Signal and GAN Generated Dataset, which are further used for training the Human Activity Recognition (HAR) System. Resulting in adequate identification of activity recognition.

The benchmark to test the designed systems includes utilizing human activity videos from YouTube or other online sources to categorize activity and accuracy for the same.

Description of proposed Prototype/Demonstration:

The development of the Generative Adversarial Network (GAN) remains unchanged. Though there were modifications in the list of functionalities, the Human Activity Recognition (HAR) System was added to the project in the mid-semester.

The ***key functionalities*** include the following:

1) Pipeline of generating Synthesized RADAR Signals from Video:
   a) Generating 3D Human Structure (Mesh) from Video
   b) Computing synthesized Radar signal from body mesh

2) Improving simulated RADAR Signals using Deep Learning Techniques:
   a) Feed the Synthesized RADAR Signals and the Real RADAR Signal through Generative Adversarial Network (GAN), leading to improvement of the former
   b) Minimizing the loss between the Synthesized RADAR Signals and the Real RADAR Signal through L2 norm

3) Augment efficiency of Human Activity Recognition (HAR) Systems using three different datasets and identify various activities

Proposed Goals/Deliverables for the Prototype Design/Demonstration:

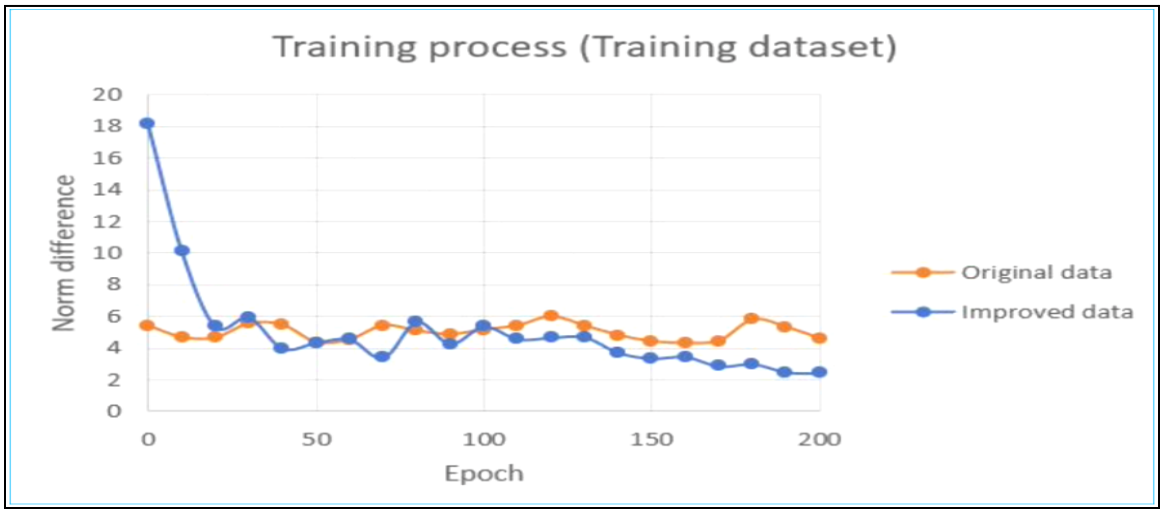
- ***Goals of Proposed Prototype:***
  - Develop Generative Adversarial Network (GAN) Model
  - Obtain loss of the Generative Model
  - Obtain Refined RADAR Signals, ready for the Motion Recognition
  - Building and Training CNN for Human Activity Recognition (HAR) System
  - Feed CNN Model with Refined RADAR Signals, test the accuracy

- ***Scale of Proposed Prototype:***
  - Create a GAN Model with Minimum Loss between the Synthesized RADAR Signal and the Real RADAR Signal using L2 Norm
  - Feeding the refined RADAR signal dataset to the Motion Recognition Model
  - The proposed accuracy for Human Activity Recognition (HAR) System was 80%

Metrics:

❖ L2 Norm for different datasets for GAN Model

- ***Training Dataset*:** A training data set is a dataset of examples used during the learning process and is used to fit the parameters of a GAN.



The above results reflect Norm Difference converges as the number of Epoch increases. Making improved data closer to the real data.

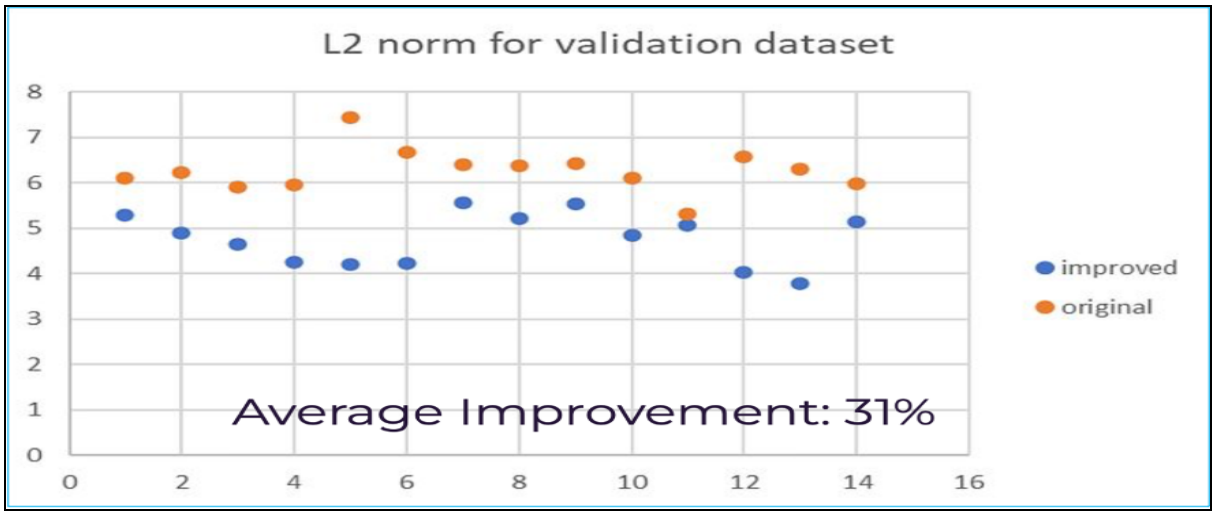
- ***Mathematical Expression:***

Real Data – R
Improved Data – I
Δ Norm Difference = R – I

Δ Norm Difference = R – I ∝ Number of Epochs

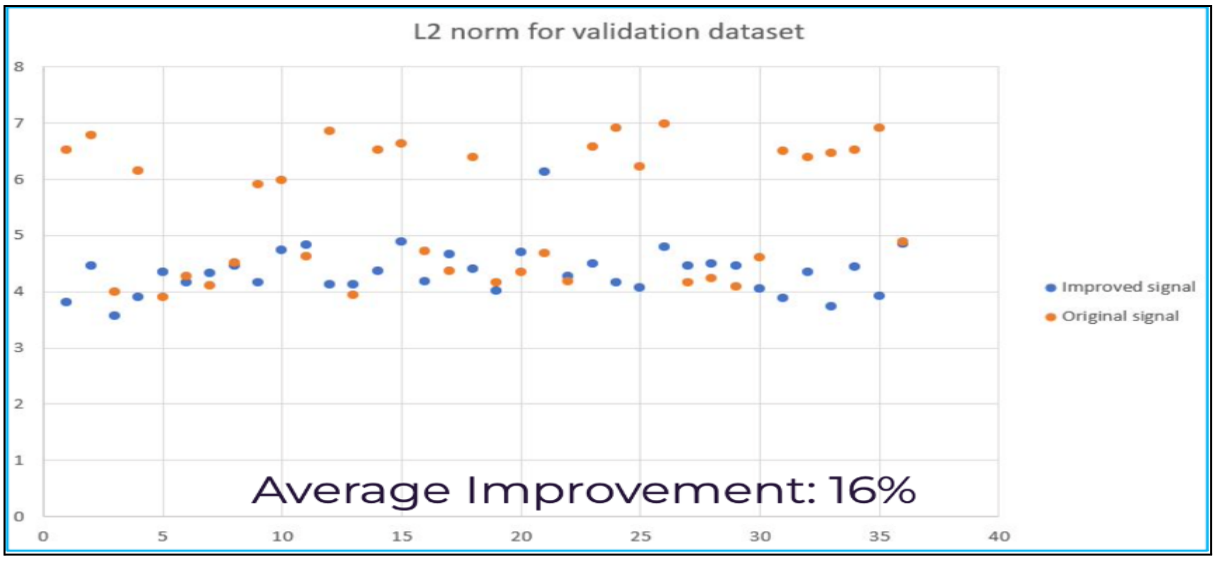
Δ Norm Difference	**Number of Epochs**
5 - 18 = - 13	0
5 – 6 = - 1	75
4 – 3 = 1	150
5 -3 = 2	200

- ***Validation Dataset*:** A validation data set is a dataset of examples used to tune the hyperparameters of a GAN. Datasets belong to the same activity as training data but are not fed into the training phase.



The above graph reflects the average improvement of 31% of Improved Signal from the Original Signal.

- ***Testing Dataset***: A test dataset is a set of examples used only to assess the performance of a GAN. Datasets are from unseen activities.



The above graph reflects the average improvement of 16% of Improved Signal from the Original Signal.

L2 Norm for Different Datasets GAN	Improvement (%)
Training Dataset	113%
Testing Dataset	16%
Validation Dataset	31%

CNN Model accuracy for training and testing using Synthesized Radar, Real Radar, and GAN Datasets.

Training and testing Datasets CNN	Testing Accuracy (%)
Synthesized RADAR Data	80.5%
Real RADAR Data	91.25%
GAN Generated Data	20%

Dropped Functionalities:

- Usage of LSTM for Human Activity Recognition (HAR) System efficacy check

The rationale behind dropping the above functionalities was the feasibility and inadequate timeline to achieve the same. Though the team's fourth member was added, this led to the addition of some partial features for Human Activity Recognition (System) using the traditional CNN model.

||| **Date Dropped:** January 20th, 2023 ||| **Date Added:** March 24th, 2023 |||

B. Design approach and final status of your prototype design/subsystem implementation/integration

Description of the Subsystems which Comprise your Prototype:

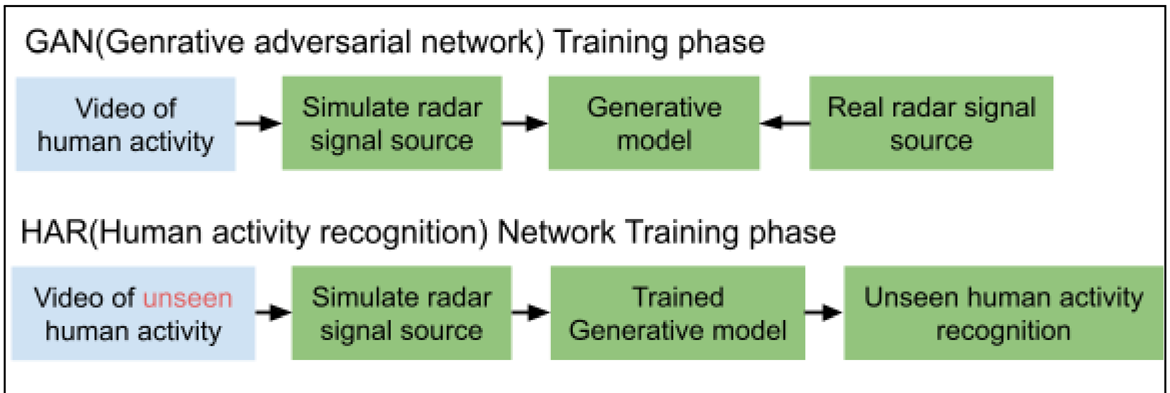

GAN(Genrative adversarial network) Training phase
Video of human activity
Simulate radar signal source
Generative model
Real radar signal source
HAR(Human activity recognition) Network Training phase
Video of unseen human activity
Simulate radar signal source
Trained Generative model
Unseen human activity recognition

Fig. System diagram of the prototype

There are four key functionality to be achieved in this prototype as shown in Fig. above.

- ***Simulate Radar Signal Source*:**
  a) Function: Produce the synthesized radar signal in doppler-time form from human activity video.
  b) Input: Human activity video
  c) output: Synthesized radar signal in doppler-time to feed into Generative network.

- ***Real Radar Signal Source*:**
  a) Function: Produce real radar signal in doppler-time form collected from mm-wave radar.
  b) Input: Mm-wave raw data
  c) Output: Doppler-time signal to feed into Generative network

- ***Generative Model*:**
  a) Function: Enhancing the synthesized radar signal to emulate the characteristics of a real radar signal and appear more authentic.

b) Input:
   i) GAN Training phase: Synthesized radar signal and real radar signal
   ii) HAR Training phase: Only synthesized radar signal
c) Output: Improved Radar Signal

- ***Human Activity Recognition Model*:**
  a) Function: Recognizing human activity using radar signal
  b) Input: Radar signal
  c) Output: Activity categories

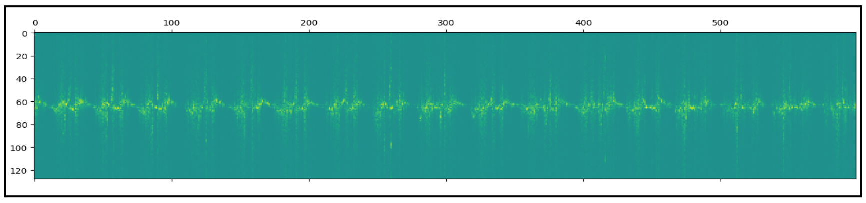
Description of the Main Approaches which you Utilized within each Subsystem:

- ***Simulate Radar Signal Source:***

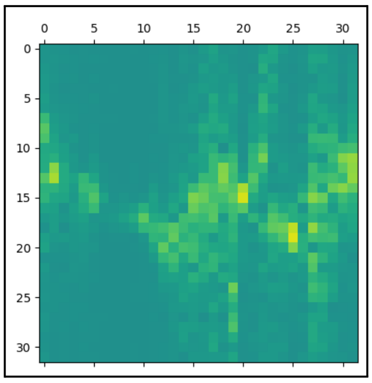
  We utilized a github resource and the original paper as references, which comprise functions such as detecting individuals in videos, constructing a mesh of the human body, and calculating the reflective radar signal. To streamline the process, we simplified the architecture to only capture a fixed camera angle, resulting in reduced computation time. Additionally, we incorporated a short-time Fourier transform to generate the doppler-time data within a length of 3.2 seconds.

- ***Real Radar Signal Source:***

  The raw signal obtained by mmWave radar is several binary files. We first transferred the binary file to complex number matrices. Then, we applied short-time fourier transforms to them to obtain the whole velocity vs. time plots. The radar signal has 10 frames per second, and our recording duration is 1 minute for each motion. Therefore, we had 600 columns of data, each column representing the velocity information during 0.1s. Since the radar signal had 128 chirps, the data had 128 rows.


(Fig. Example of a Complete Velocity vs. Time Plot)

In order to transfer the whole data to 32*32 matrices to fit the GAN model, we divided it into 18 cuts by columns. Also, we observed that there isn't much information at the top and bottom of the plot, we clip them and only keep the 32 rows of data in the middle.


*(Fig. 32*32 Matrix obtained from the above Plot)*

- ***Generative model*:**

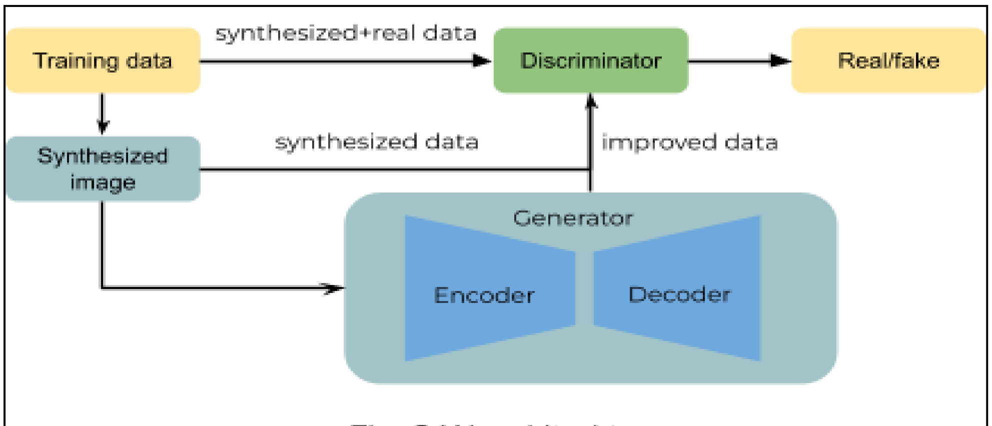


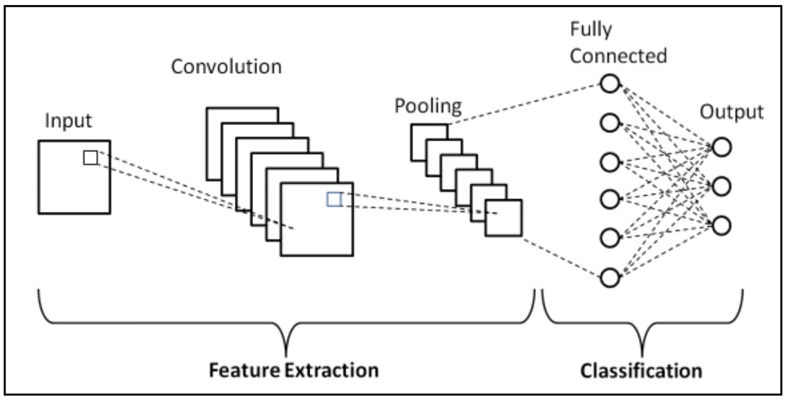
Fig. GAN architechture

We applied the Pix2Pix conditional GAN model to produce improved radar signals. The model comprises of two components: a generator and a discriminator. The generator uses a multi-layer U-shaped encoder-decoder architecture that captures the image's contour and details, resulting in improved synthesized radar signals. The discriminator, on the other hand, is a classification network that evaluates both the real and generated radar signals to calculate the conditional probability of the image being real.

- ***Human Activity Recognition Model*:**

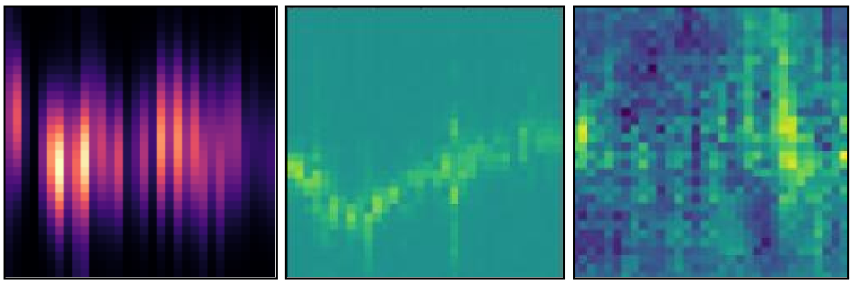
To demonstrate that the data sets produced by our GAN model can be used in the real situation. We implemented a HAR system using the GAN-generated data to train our CNN model and test on the real radar signal to check whether the accuracy achieved our goal. We refer to one paper [3] to implement our HAR model using CNN model. Their experiments have shown that the proposed CSI-based HAR outperforms other competitor methods including 1D-CNN, Long Short-Term Memory (LSTM), and Bi-directional LSTM, and achieves an accuracy of around 95% for seven activities.

They transfer human activity input data into PNG format and take it as input to the CNN model. Similarly, we construct our project's CNN model by ourself.



(Fig. CNN model)

There are five Convolutionlayers, five Batch layers, five ReLu layers and five MaxPool layers in the Convolution part. In the Fully connected layers part, there are three Linear layers and two ReLU layers. As mentioned above we transfer our input synthesized data, real radar signal data and GAN model data into PNG format (Fig. 1) as our input data to train our CNN model.


(*Fig. 1 - Left to Right, Synthesized Data, Real Radar Signal and GAN-Generated Data)*

<u>Summarization of the Final Status of the Prototype Design</u>:

- ***<u>Simulate Radar Signal Source</u>*:**
  We successfully achieved the primary objective; however, we constrained the video to feature only one individual.
- ***<u>Real Radar Signal Source</u>*:** Successfully achieved
- ***<u>Generative Mode</u>*l:**
  Although we attained the primary function which is to enhance the synthesized signal, the enhancement in unseen activities is restricted. This limitation could be resolved by testing various hyperparameters and by acquiring more data.
- ***<u>Human Activity Recognition Model</u>*:**
  We have succeeded in identifying various activities using real, synthesized and enhanced data. Nevertheless, since the signal for unseen activity is not distinct enough, we are currently unable to differentiate unseen activity using a network trained from enhanced signal.

<u>C. Final Demonstration</u>

The results of our system will be demonstrated across three categories, namely training data, validation data, and testing data. The training and validation data will comprise the same activity, while the testing data will consist of unseen activities.

In the following figure, improvement metrics is calculated as,

$$\frac{||I-R||-||S-R||}{||S-R||} \times 100\%$$

where S represents synthesized signal, R represents real signal and I represent Improved (GAN-generated) signal. All the norms are L2 norms.

Dataset	Training data	Validation data	Test data (Unseen activity)
Dataset size	400	14	36
Average improvement	113.4%	30.8%	16.2%

Table. Comparison of performance in different dataset

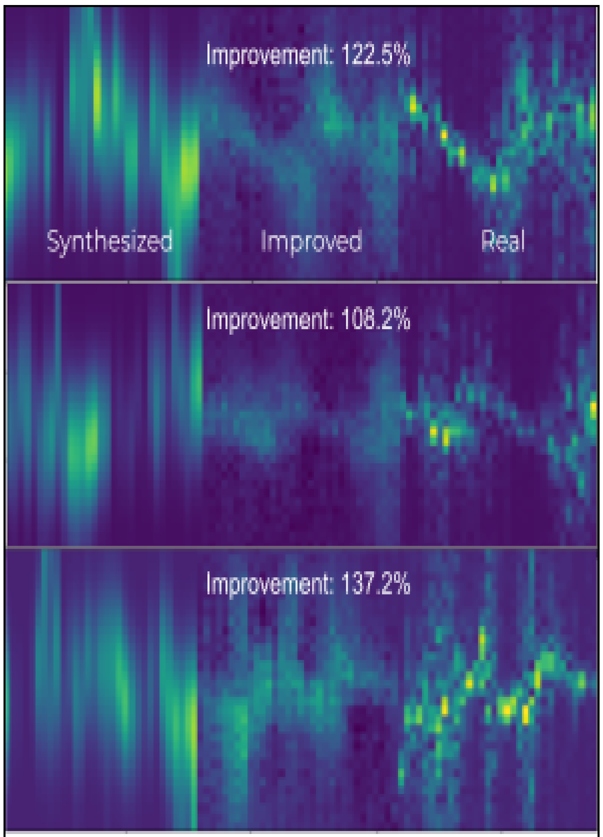


Fig. Training data result

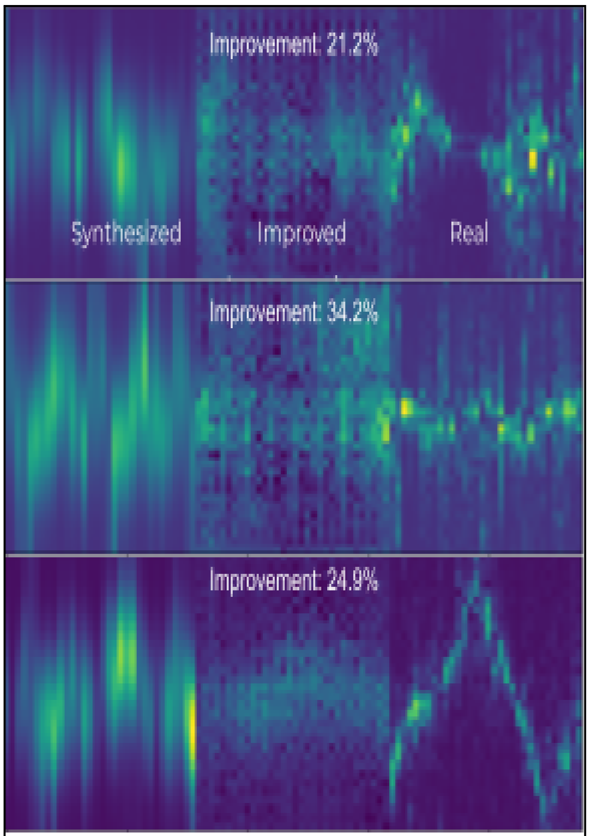


Fig. Validation data result

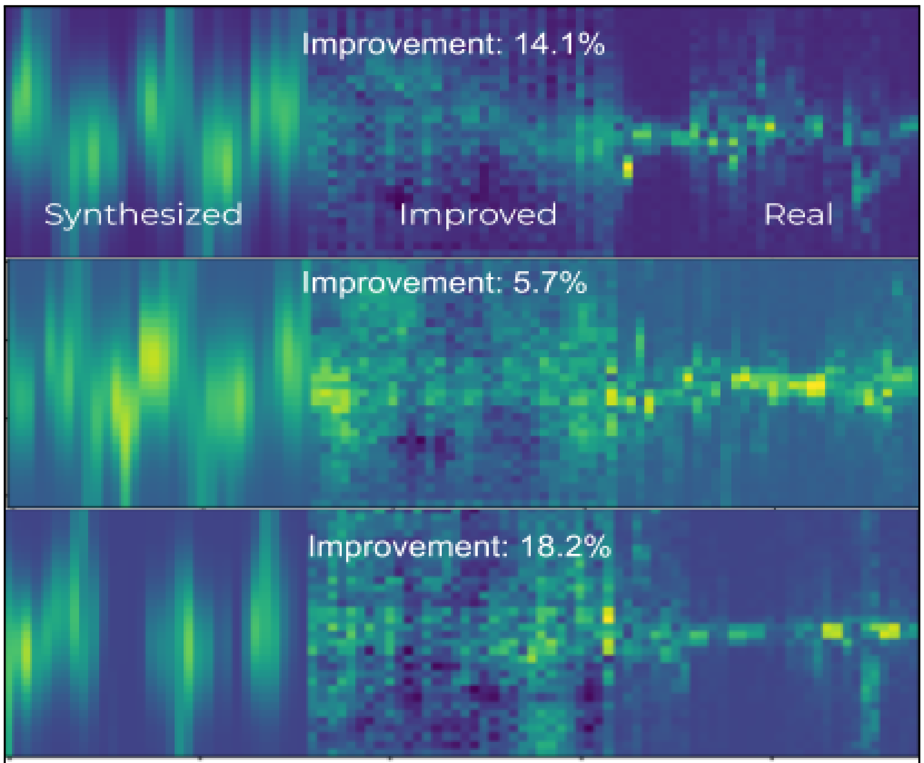


Fig. Testing data result

Based on the table and figures presented, we can infer that our GAN model can efficiently enhance the training data. However, its performance declined in the validation and test datasets, possibly due to overfitting on our relatively small training dataset. In conclusion, our study demonstrates that utilizing the Pix2Pix GAN model can enhance the quality of data synthesized from human activity videos, allowing us to leverage the vast video datasets already available.

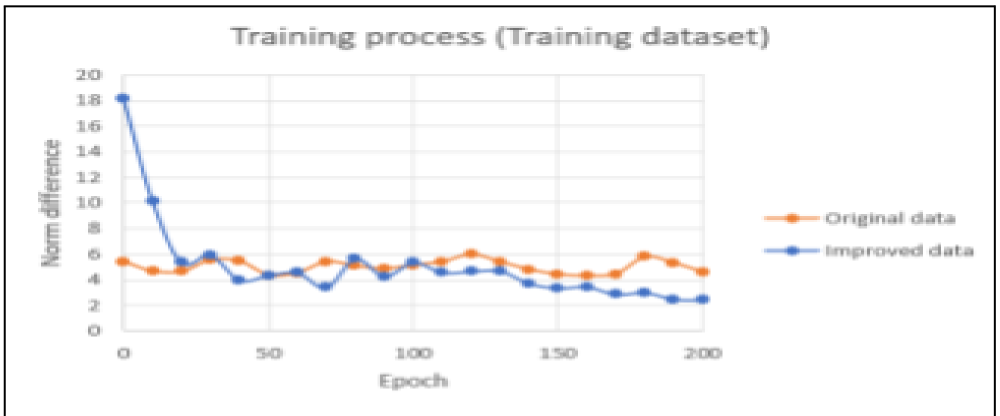
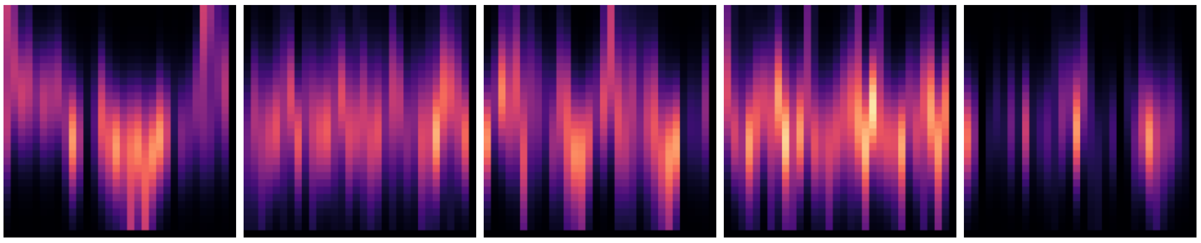


Fig. Training process of norm difference vs epoch

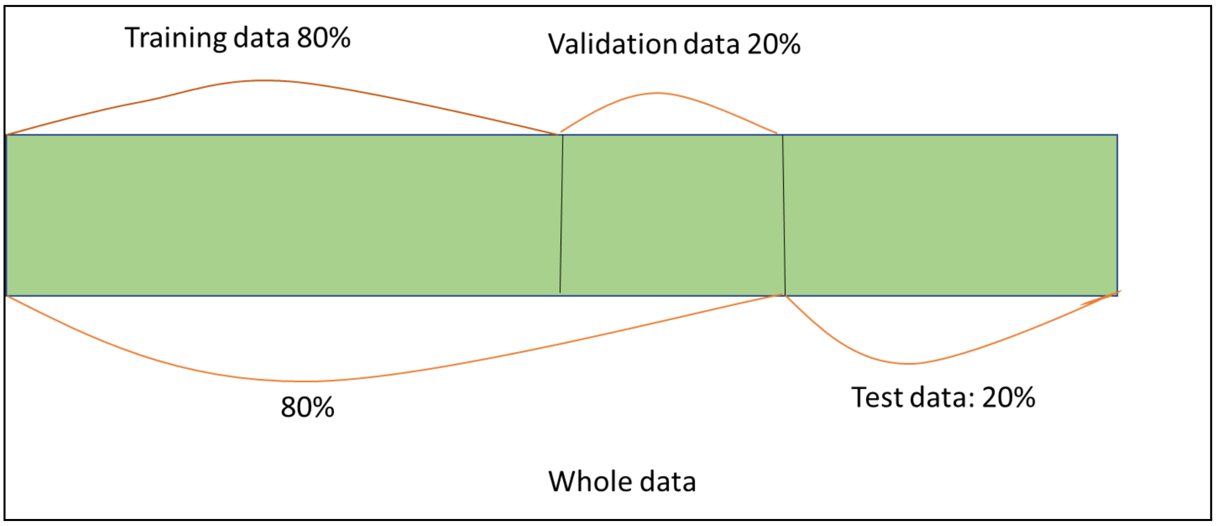
***For Human Acitivity Recognition*:**

We train our CNN model using Synthesized data, real radar signal and GAN data seperately to recognize five human activities (Hand Motion, Jumping Jack, Lunge, Run and Squats). We label them as Hand Motion: 0, JJ: 1, Lunge: 2, Run: 3, Squats: 4.

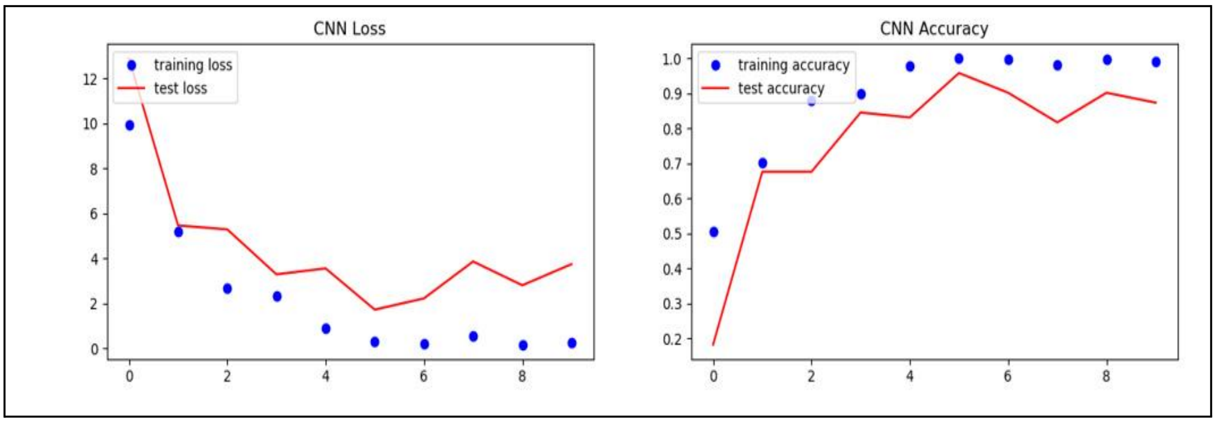


(Figure from left to right, Synthesized data for Hand Motion, JJ, Lunge, Run and Squats)

We trained the synthesized CNN model using 20% of the data as test data. For the remaining 80% data sets, we split 20% of those training data into validation data and use the remaining 80% for training (as shown in Fig.).

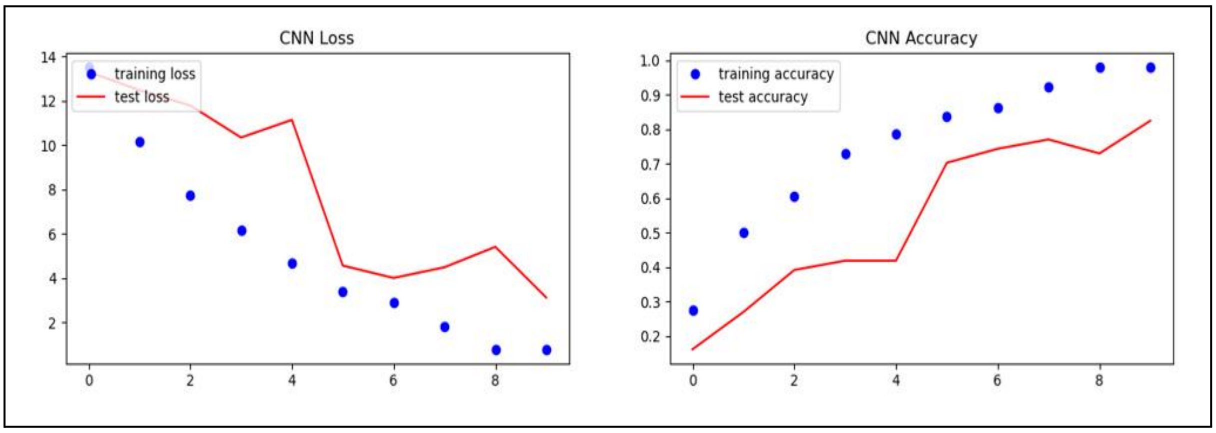


The training result shows that the CNN model trained from Synthesized data has accuracy at about 80%.

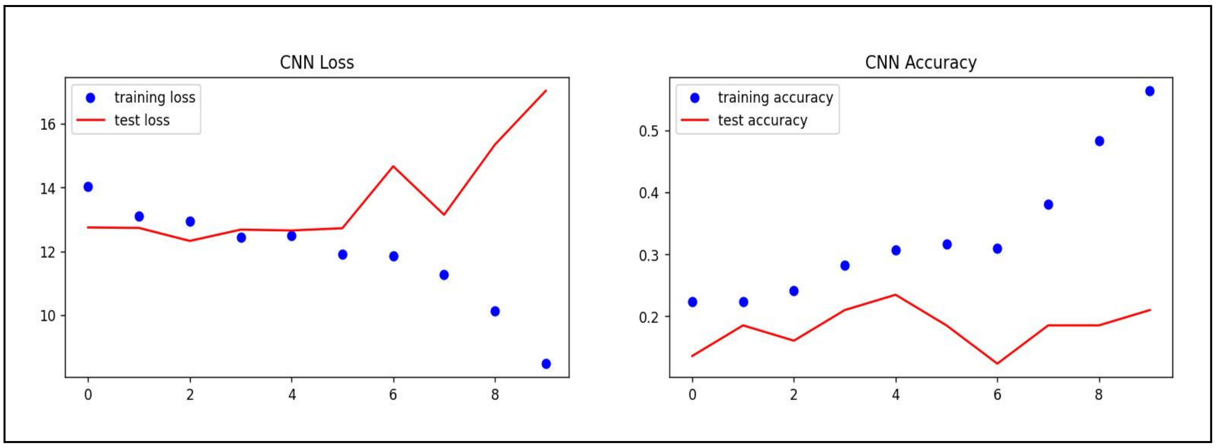


(Fig. Training Result for Synthesized Data)

For Real Radar Signals, the accuracy is at about 90% while for the GAN model it is at about 20%.

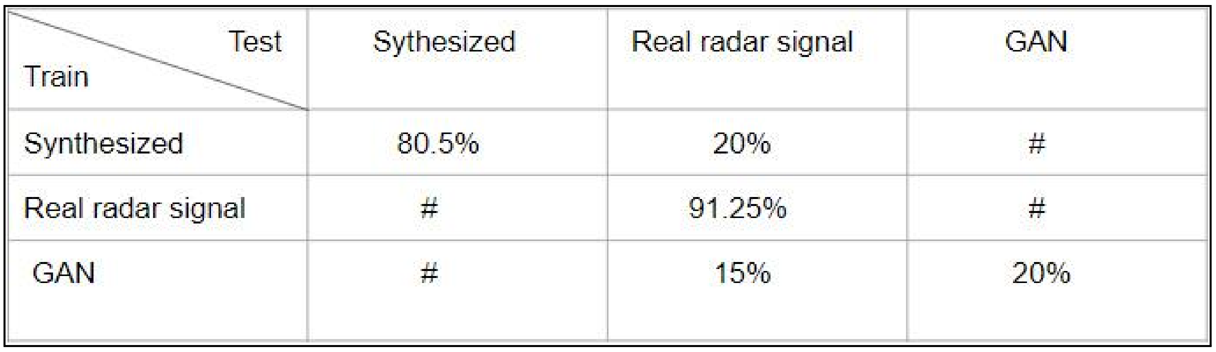


(Fig. Training Result for Real Radar Signal)



(Fig. Training Result for GAN Data)

We also tested our model trained by GAN data on the real radar signal and the result accuracy is at about 15%. As we can see, when training our CNN model with GAN data, our CNN model overfits. The reason may be lack of training data. Another reason may be that the data generated by Gan has more features added to the PNG image, which makes it harder to distinguish the different actions compared to the data generated by the other two sources. Our future work is to improve and adjust our CNN model layers to train the data produced by GAN successfully.



Train \ Test	Sythesized	Real radar signal	GAN
Synthesized	80.5%	20%	#
Real radar signal	#	91.25%	#
GAN	#	15%	20%

D. Functionalities and Performance Metrics of Prototype – Testing and Benchmarking

Comparison of your prototype functionalities/metrics to your proposed values:

The L2 Norm has been the standard way to evaluate the accuracy and efficiency of the system. The GAN Model and CNN Human Recognition Model were tested through L2 Norm to keep up the accuracy and minimum Loss for refined RADAR Signals. We proved GAN can be utilized to improve the signal quality synthesized from video, which makes it possible to take advantage of huge activity video dataset online. The quality improvement of unseen videos could be more precise, and this is due to a lack of training data.

The second table includes a qualitative comparison of what has been achieved compared to what was proposed. The development part of GAN and the CNN for Human Activity Recognition was completed. The accuracy and results could have been better and needs more standardization. The first table approves the need for improvement as there is a massive gap between the proposed and resulting values.

PARAMETERS	PROPOSED ACCURACY	MEASURED ACCURACY
GAN Testing Dataset	Approx 80%	**16%**
GAN Validation Dataset	Approx 80%	**20%**
Synthesized RADAR Data*	Achieved	**80.5%**
Real RADAR Data*	Achieved	**91.25%**
Improved RADAR Data*	Approx 80%	**20%**

* HAR Centric Accuracy

Proposed	Realized
Develop GAN Model	Achieved
Accurate Generative Model	Partial
Refined RADAR Signal	Partial
Building CNN Model	Achieved

Training CNN Model	Achieved
HAR Accuracy	Partial

Benchmarking against existing Approaches:

The above tables represent the low accuracy of the detection or the poor gain of the improved signal delivered by the generator. The convergence of two lines in the graph for the training dataset between improved and original data can be seen. The gap between the two lines is reduced as there is an increase in the number of epochs. In that case, probably an increase in the dataset with more epochs could lead to a decrease in the loss leading to better results. Moreover, the CNN Model for Motion Recognition results were quite good for the Synthesized and the Real RADAR signal but were very low for the GAN Data.

The quality improvement is unclear for unseen videos reflecting a lack of training data. The current GAN has more features added to the PNG Image, making it harder to distinguish the different actions than the data generated by the other two sources. More focus could be made on improving and adjusting the CNN Model layers for getting trained successfully on the GAN data.

Points of Improvement:

- Increase in Training Datasets
- Introduction of Time Synchronous Algorithm to avoid Time difference between two sources
- Introduction of Range dimension
- Testing various generative network architecture
- HAR Network implementation for unseen activities

There are HAR products in the market[1] that work through different approaches for activity recognition. It might include a specific feature extraction from the body through various sensors and tools. Like, Recurrent Neural Networks (RNN) instead of Convolutional Neural Networks (CNN). Though, our approach included the latter one. Like, CNN-LSTM approaches boast an accuracy of approx 91% for activity recognition.

The current HAR in the market holds higher accuracy. Different inductions of sensors such as cameras, and other bodily sensors bring better feature extraction.

Our Approach	Existing Approach
CNN*	CNN-LSTM[2], RNN*
Data Collection - RADAR	Data Collection – Biosensors, Inertial Sensors, Smart Wearables
GAN for Refined Signals	-
Low Accuracy - 16%	High Accuracy < 80%

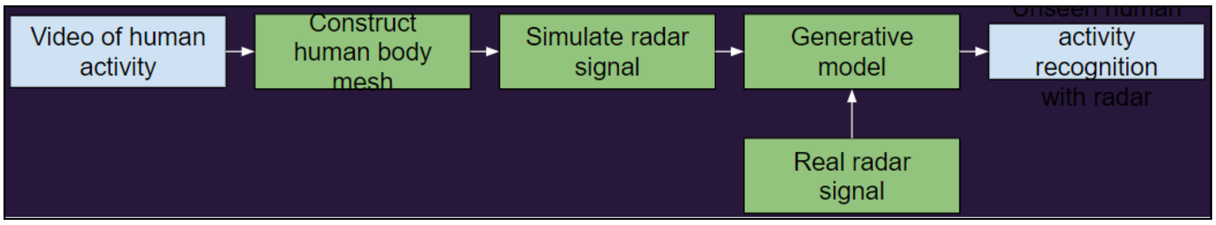
* HAR Centric

E. Relationship of Prototype to Final Product

The prototype proves the whole flow of subsystem blocks works. First, we used the Vid2doppler modules to generate simulated radar signals from videos, and developed our own algorithm to process the real radar signals. We can observe the correspondence relationship between those two signals.

[1] HAR Methods
[2] CNN LSTM Approach

Then, we proved that the GAN is able to improve the simulated signals. Finally, the activity recognition neural network can discriminate between various human motions.



In order to make the prototype scalable to the final product, several improvements can be made. First, since our prototype proves the ability to discriminate 5 motions and 2 unseen motions, more motions can be added. Also, to train more powerful generative models and activity recognition neural networks , we can also improve our hardware, such as PCs or GPUs. We can also switch to a more accurate mmWave radar, so it can obtain much more accurate real radar signals.

For the key remaining challenges, the most important thing is improving accuracy of the generative model. The better it improves the simulated signal, the more accurate the activity recognition can be. Another challenge is we need to make a GUI for customers. Currently we are using code and commands to run our product, we need to make an interface to make it easy to use by our customers. The other 2 challenges are real time recognition and multi-people recognition. The prototype processes the radar signals and does motion-discrimination after it finishes the radar signal recording. However, in our final product, we need to do the recording and discriminating simultaneously, which means we need to make the whole process running in real time. In the real application scene, the final product would face multi-people's motions-recognition tasks. Since our prototype can only recognize a single person's motions, it is also a big challenge.

F. Next Steps/Follow-Up Topics

A revised set of key functionalities/goals/metrics:

Revised Key function:
- Able to train a neural network using the enhanced data to recognize human activity.
- Capable of preprocessing video data involving multiple individuals and feeding it into the pipeline for further processing.

Revised metrics:
- Accuracy of the Human Activity Recognition (HAR) model trained using the enhanced radar signals.

Key limitations in the capabilities of our approach:

- Currently, time alignment is achieved manually, which may result in temporal misalignment of data from different resources. Even though the neural network can still extract features under this scenario, a time-aligned dataset would decrease the amount of data required to train the GAN model.
- Our Human Activity Recognition (HAR) model employs a conventional convolutional neural network that processes fixed-sized inputs. Consequently, it may not be feasible to implement this model in real-time scenarios. To overcome this limitation, we may need to utilize a Recurrent Neural Network (RNN) instead.

Potential changes in scale to more closely approximate required scale for final system:

As our goal is to enhance any human activity, the GAN model must learn features from as many activities as possible. We estimate that the model will require at least ten times the current data size to achieve this. Although we can use data augmentation techniques to expand the dataset, we may also need to collect data on "smaller" activities, such as nodding the head, to ensure that the GAN model and the radar have a high enough resolution to distinguish between activities and learn their detailed features.

G. Technical Contributions

Tasks	Team Members
Collect Input Data	Ruijie, Upanshu, Yinchien
Process Video Signals	Yinchien
Process RADAR Signals	Ruijie
Build GAN and Demonstrate	Upanshu
Training, Validation and Testing GAN	Yinchien
CNN Development and Training for HAR	Jin Hau

The collection of RADAR data can't be done individually as it involves the parallel operation of a Smartphone, RADAR and Laptop linked for processing the data. Collecting and recollecting the data for various activities is exhausting as it involves different exercises. If performed alone, multiple people can finish the task in a few hours rather than several days. Moreover, installing the correct software in the right environment makes the process faster.

The project was indeed a joint effort, as the team was stuck with many barriers and came in to rescue to get over the situation. Designing and developing the GAN Model in accordance with the RADAR dataset was one of the challenging parts. Yinchein must be appreciated for clearing the deadlock error generated during the GAN modification. Ruijie worked on his part with consistency and delivered the real RADAR Signal on time, even after data recollection. The same can be said for Yinchein for delivering the synthesized RADAR signal. Upanshu did his job for data collection and development of the GAN Model. Whereas Jin Hau took a challenge to design and develop the CNN model for Human Activity Recognition (HAR) on short notice as he joined the team mid-semester. The team did a commendable job of being strict about the deadlines and working coherently. Though the team implemented the whole project, the results weren't expected and standard. Improvement could have been made, and it's the work for the future to elevate the efficiency of the GAN Model.

H. Key Milestones/Timeline

Milestone 1: Obtain labeled motion videos and real-world radar signals from motions.
Milestone 2a: Obtain simulated velocity Histogram plot.
Milestone 2b: Obtain real velocity Histogram plot.
Milestone 3: Finish the code for the generative model structure and loss function validation.
Milestone 4: Obtain the accuracy of our generative model.
Milestone 5: Obtain refined radar signals which are ready for motion recognition.
Milestone 6: Observe correct rate of each NN.
Our milestones are almost the same as the project plan. The only difference is the milestone 2. The tasks of processing simulated signals and real radar signals are taken by different members. For better project progress management, we divided milestone 2 into two parts.

	2023/1/27	2023/2/10	2023/2/24	2023/3/10	2023/3/24	2023/4/7	2023/4/21	2023/4/28
Step 1 (Upanshu) Collect Input Data	🟩	🟩	Milestone 1			🟧 **Recollect data**		
Step 2a (Yinchieng Huang) Process video signals	🟩	🟩	🟩	Milestone 2a		🟩		
Step 2b (Ruijie Song) Process radar signals	🟩	🟩	🟩	🟧	🟧	🟧	Milestone 2b	
Step 3 (All team members) Build GAN model	🟩	🟩	🟩	🟩	🟩	🟧 Adjust the model	Milestone 3	
Step 4 (Yinchieng Huang) Training and validation					🟩	🟩	🟩	Milestone 4
Step 5 (All team members) Testing GAN							🟩	Milestone 5
Step 6 (Jin-Hau) Apply motion recognition							🟩	Milestone 6
Reserved for modifications and documations								🟩

For our project timeline, it differs a lot from the project plan. Since our technical advisor found some errors in our data collected in January, we had to recollect data at the beginning of April. Then, we had to process the simulated and real signals again in April. Thus, most of the milestones are postponed.

I. Project Management

Technical:

Upanshu Srivastava
- Collect and Label Video and Radar Data
- Build and Train Generative Model (Pix2Pix)

Yinchien Haung
- Construct Mesh from Video and Simulate Radar Signal
- Build and Train Generative Model

Ruijie Song
- Process the Signals from the Real Radar

Jin-Hau Yeh
- Build and Train CNN Model for Human Activity Recognition

All group members
- Study Radar Signal Formulation
- Visualizing Neural Network and Training Process

Management:

Technical Project Manager (TPM) — Yin chien Huang
- Divide the whole project into specific tasks and assign to every team members
- Track the progress of each tasks
- Coordinate meetings

Document Coordinator (DC) — Upanshu Srivastava
- Restore and manage the video and radar data
- Edit and proofread materials for reports/presentations

Business Office Liaison (BOL) — Ruijie Song
- Help purchasing and managing hardware equipments

In order to monitor the project progress, we hold 2 meetings every week. During every meeting, the team would communicate updates on team members' tasks. We will discuss if our results satisfy every milestone, and what to do before the next meeting.
For preparing materials for reports/presentations, the team would first assign different parts to members during the weekly meeting. Then, each team member works on their own parts first, so we would get a first draft. Next, we exchange opinions on the whole slides/reports and do revision during the next meeting. Finally, the document coordinator does the final proofread and submit the reports/slides.

Appendix:

I. Team:
   A. Yinchien Huang (huan1903@purdue.edu) - Machine Learning
   B. Ruijie Song (song690@purdue.edu) - Signal Processing
   C. Upanshu Srivastava (usrivas@purdue.edu) - Machine Learning
   D. Jin-Hau Yeh (yeh79@purdue.edu) - Machine Learning

II. Individuals:
   A. Qiming Cao - Related codes for Signal Processing and Extraction

III. References:

A. Design Reviews
B. Different Sensor Data Collection Approaches for Human Activity Recognition
C. HAR using CNN LSTM
D. HAR Accuracies
E. HAR for Different Applications
F. Other HAR References
G. Synthesizing Doppler Radar Data from Videos for Training Privacy
H. Pix2Pix GAN Demonstration
I. HAR Using Deep Learning
J. CNN Architecture
K. Basecamp Code Reference