CS 498 Machine Perception Final Project: Millimeter Wave Radar Gait Recognition

Xin Jin
Univeristy of Illinois at Urbana Champaign
xinj3@illinois.edu

Abstract: Gait recognization is a special kind of identity recognization task gains more and more attentions recently. Traditioanl computer vision approaches use videos and images to capture the walking postures of a person, but they are often compromised by the privacy issues and lighting conditions. Researchers found alternative solutions by using millimeter wave radar to detect the 3D locations of a person in point clouds form, and use machine learning models to identify the person. In this project, gait recognition was implemented using two different neural network structures: Feature Sequence Model and Time Sequence Model. The experiments showed similar performance between these two model structures. Furthermore, the performace of different data representations, feature array representation and voxelized point cloud representation, were also tested on Time Sequence Model. The experiments showed what feature array representation had lower accuracy but being much more space efficient.

1. Introduction

Just like any other biological features, gait information can be used for identity recognition. Gait describes a person's walking postures, and due to the difference in weights, heights, and walking habbits, people can often recognize a person in real life solely based on the gait of a person without knowning the face. Gait recognition is a popular task in computer vision [6], where videos or images of a person walking need to be taken as dataset. However, there are two major liminations with computer vision approaches. First, because videos and images of a person must be taken for the recognition task, personal privacy will be greatly threatened. Second, to capture accurate information, images and videos must be taken under ideal lighting condition with minimal blocking or occlusion in the scene.

Recently, researchers explored alternative solutions by using millimeter wave radar technology to capture human gait postures. Milimeter wave radar transmits wireless signals into space and captures the unique walking postures of a person. The recieved signal can be translated into different representations, and unique patterns can be learned by various machine leanring models for recognition and identification tasks. Due to the physical nature of the millimeter wave, millimeter wave radars do not capture the physical appearance, which greatly preserve the personal privacy. Also, millimeter wave is able to penatrate through space and obstacles, thus it can capture the gait information under any lighting conditions, and in various scene settings.

In this project, I replicated a gait recognition experiment, and compared the performace of different data representations and model structures. The dataset was aquired from an open source dataset [8], which recorded the raw data from two millimeter wave sensors. The dataset was first been preprocessed into desired format. Then, two models were constructed from scratch for feature learning. The first model was inspired by [7], which was more feature sequence oriented neural network model. The model was adjusted from the original paper for the purpose of this project. The second model was inspired by [10], which was more time sequence oriented neural network model. Similarly, necessary adjustments were applied for the purpose of this project.

The outline of this paper is as follow: Section 2 explains the background knowledge of the millimeter wave radar, and the related work. Section 3 explains the dataset used in this project, and the data preprocessing step. It also explains two different models used in this project. Section 4 shows the implementation details and experiment results followed by analysis. Section 5 is the conclusion.

2. Background & Related Work

2.1. Millimeter Wave Radar

The milimeter wave radars that used to collected the dataset were special types of radar called Frequency-Modulated Continuous Wave radar (FMCW radar). FMCW

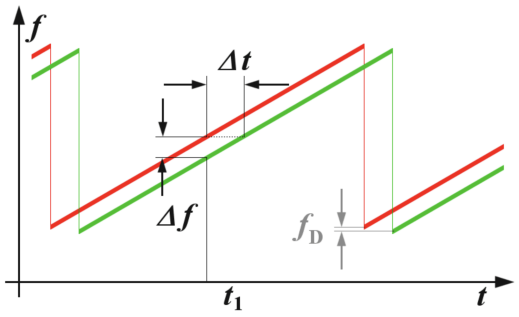
radars transmit continuous waves whose frequency increases linearly over time, and the waves are transmited periodically. A graph of a segment of the transmited signal is shown in Figure 1. The green slope is the transmited signal, and the red slope is the received signal. The frequence of the transmited signal increase linearly over time in an interval. Each signal in a interval is called a cherp. The transmitted signal hits an object, reflects back, and will be captured by the radar receivers. Due to the reflection and transmission in space, comparing to the transmitted signal, the received signal will have a frequency shift Δf and time delay Δt. Δt then can be used to calculate the distance between the object and the sensor antenna:

$$distance = \frac{c * |\Delta t|}{2} \quad (1)$$

Where c is the speed of the light.

FMCW radars usually have multiple antennas placed in certain patterns. The pattern allows the sensor to capsture the small time delay between different antennas of the same received signal. The small time delay between the antenna can then be used to calculate the angle of the received signal. By knowing the distance of the object and the angle of the received siganl, we can calculate the relative 3D position of the object with respect to the radar.

Figure 1. A segment of the FMCW radar signals. The green slope is the transmitted signalm, and the red slope is the received signal. Δf is the frequency shift and Δt is the time delay.



2.2. Related Work

FMCW radars were widely used in various applications. In vital signal detection, FMCW radars are used to detect the displacement of the human chest wall, and signal processing techniques can be applied to the received signal to estimate the heart beat rate and respiration rate [1, 3]. This shows a lot of potentials in health care and monitoring since the detection devices do not need to be in physically contact with the users. Also, researchers often use the point cloud representation of the FMCW radar for human activity detection [10, 11], such as walking, jumping, and boxing. Point cloud representation can also be used for room map scanning and reconstruction [5]. Alghough, the point cloud data is often too sparse for high resolution scanning, researchers were able to leverage the deep neural network to reconstruct the high resolution map using the sparse point cloud dataset [5], even if the room is visually unclear. Another popular FMCW radar application is to use the doppler-range heat map representation. Different from the point cloud, doppler-range heat map only recoreds the range and velocity information. This kind of representation is found useful in human activity detection [4], and also hand gesture recognition [2]. The hand gesture recognition with FMCW is already applied on some of the smart phones for functions like touchless control [9].

3. Methods

3.1. Dataset

The gaits 3D point cloud dataset is aquired from [7]. The data was gather through experiments shown in Figure 2. There were two scenes in the experiment. In each scene, single or multiple participant(s) were asked to walk in the designated experimental area for multiple times. The participants were asked to walk in either fix route, or free route. The data gather from both scene with either fixed or free route were stored seperately. A summary of the dataset is shown in Figure 3. For example, the bold **25** in the table means that there are 25 unqiue participants in the scene 2 experiment walking in fixed route, and there are three participants walking simutaneously in this experiment.

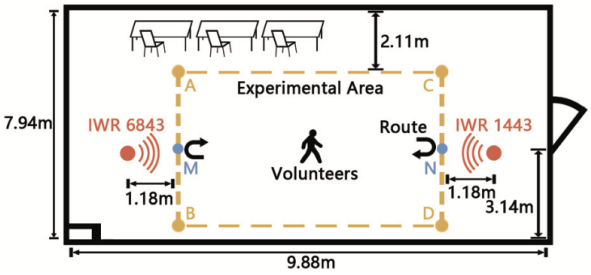
The dataset was collected with two millimeter wave sensors: TI IWR1443 and TI IWR6843. Each device is a FMCW rader with multiple antennas. The FMCW radar sends out sinusoidal waves with linearly increasing frequency periodically. The transmited signal reflects back from objects, and the received signal is then comparied with the transmited signal to estimate the object's 3D location as well as velocity. Each frame will capsture multiple points in the space. The configureation details can be found in [7].

The main reason to use two sensors instead of one was that the resolution of a single sensor was relatively low. In average, TI IWR1443 sensor was only able to detect upto 64 points in a frame, and TI IWR6843 was only able to detect up to 100 points in a frame. The situation would be worse if there were multiple persons in the scene and occlusion was also likely to happen. By placing two sensors at opposite location, it could greatly increase the resolution (point cloud density) of a frame.

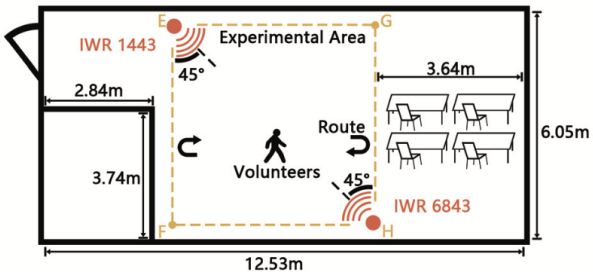
Each sensor was connected to a laptop. To solve the time synchronization issue, one of the laptop was made as a host,

and another was made as a client. While the laptop was recording the sensor data, it also recorded the time stamp of the current frame using Network Time Protocal(NTP) along with the sensor data in a csv file. Figure 4 shows a screenshot of the actual dataset. **Frame #** is the default frame number generated by the board. **# Obj** is the number of points detected in the current frame. **X**, **Y**, and **Z** are the 3D coordinates of the point with respect to the sensor. **Doppler** is the doppler velocity of the point. **Intensity** is the intensity of the received signal. **y**, **m**, **d**, **h**, **m.1**, and **s** represent the year, month, date, hour, minute, and second time stamp of the current frame. In each experiment, the data was stored in two csv files collected by each sensor.

Figure 2. The experiments setup of two scenes. Each scene has a designated experimental area with two sensors placed in opposite location.) [7]



(a) scene1



(b) scene2

3.2. Dataset Preprocessing

To merge the data collected by two sensors into one conhesive dataset, two preprocessing steps were necessary.

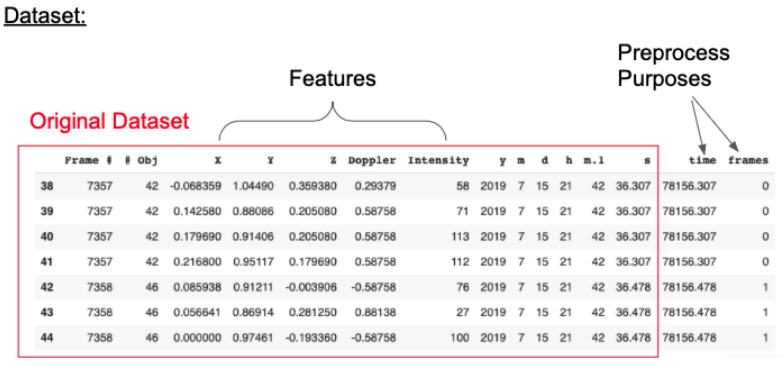
First, because the dataset was collected by two sensors at different location, the 3D points collected by each sensor was in different coordiate system. To merge the data collected by two sensors into one, the coordiante system of the data collected by one sensor needed to be transformed into the coordinate system of another sensor. More specifically, a homogenous transformation was apply on the data collected by IWR1443, which rotated the **X**, **Y**, and **Z** coordinates 180 degrees and translated to the location of IWR6843 based on the experiment scene shown in Figure 2.

Figure 3. Summary of the dataset. For example, the bold **25** in the table means that there are 25 unqiue participants in the scene 2 experiment walking in fixed route, and there are three participants walking simutaneously at a time. [7]

Scene	scene1		scene2	
Route	Fixed	Free	Fixed	Free
1vol-sim	20	20	30	25
2vol-sim	20	20	30	30
3vol-sim	15	10	**25**	25
4vol-sim	15	10	15	15
5vol-sim	10	10	10	10

Second, time synchronization issue needed to be address in order to merge two dataset. The time stamps of each data frame collected by two sensors did not match precisely, and two frames collected at about the same time by two sensors should be considered as one frame in order to increase the point cloud denisty of a frame. Given that the all data was collected on the same date, a **time** column was added to the dataset shown in Figure 4. The **time** column was the time in seconds since the start of the day the data was collected. Equivalently, **time** = (**h** * 60 + **m.1**) * 60 + **s**. The data was then sorted in increasing order based on **time** column. Two frames would be considered as one frame if two frames were collected within 50ms. The grouped frames were assigned with a new frame id was shown in the last column in Figure 4.

Figure 4. A screenshot of the dataset (after merging). The red box shows the original dataset, and two extra columns (time and frames) are added for preprocessing purpose.



In this project, only the **X**, **Y**, **Z**, **Doppler**, and **Intensity** features were used. Each frame was either padded or

downsampled to 128 points. Majority of the frames has less than 128 points. When doing padding, same points in the current frame were replicated for padding. Every 32 frames were grouped into a block, which was about 3s. Thus, the dimension of one data block was 32 (frames) x 128 (points) x 5(features). The dataset was shuffled, and 80 percent of the merged dataset was used for training and 20 percent was used for testing.

3.3. Models

This project experimented two model strcutrues: Feature Sequence Model and Time Sequence Model. The first model was inspired by [7], in which each feature was seperatedly learned by a convolution nueral network. Then, the extracted features were concatenated together and learned by a fusion layer. At the end, a fully connected layer was used to yield desired label probabilities. The second model was inspired by [10], in which the data was first split into four time sequences, and each time sequence was learned by a convolution nueral network. Then, features extracted from each time sequence were concatenated together and learned by another layer of neural network, and finally a fully connected layer was used to yield the label probabilities.

3.4. Feature Sequence Model

The structure of the first model was shown in Figure 5. The feature channels were split into five individual channels. Each feature was then processed by a series of neural network. The assumption was that the change of each feature over time might reveal the gait charactristics of a person. This model tried to learn the charactristic of each feature (3D coordinates, doppler velocity, and intensity) over time and then fuse the extracted features at the end for the recognition.

Conv1 was a layer of 3x3 spatio-temporal convolution kernel with stride of 1 and padding of 1. The input channel for **Conv1** was 1 and output channel was 3. It was follow by a normalization layer and a ReLu activation layer. Then, a 3x3 max pooling layer with stride of 2 and padding of 1 was applied to the result. The result of **Conv1** was feed to both **Resnet** and **Conv2**. **Resnet** was the first 8 layer of the pretrained ResNet18. The input channel for **Resnet** was 3 and output channel was 64. It was also follow by a normalization layer and a ReLu activation layer. **Conv2** was a layer of 3x3 spatio-temporal convolution kernel with stride of 1 and padding of 1, and similarly was followed by a normalization layer and a ReLu activation layer. The input channel for **Conv2** was 3 and output channel was 64. The results of **Resnet** and **Conv2** were concatenated together and passed to **Conv3**. **Conv3** was a layer of 3x3 spatio-temporal convolution kernel with stride of 1 and padding of 1, and similarly was followed by a normalization layer and a ReLu activation layer. The input channel for **Conv3** was 128 and output channel was 64.

The results of five features were then concatenated together and passed to **Conv4**. **Conv4** was a layer of 3x3 spatio-temporal convolution kernel with stride of 1 and padding of 1, and was followed by a normalization layer and a ReLu activation layer. The input channel for **Conv4** was 320 and output channel was 1. Finally, the result was passed to a fully connected layer to get the desired output dimension.

Figure 5. Feature Sequence Model Structure

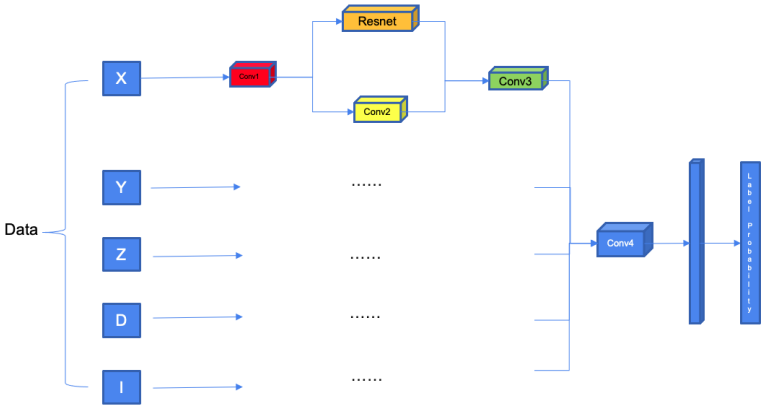
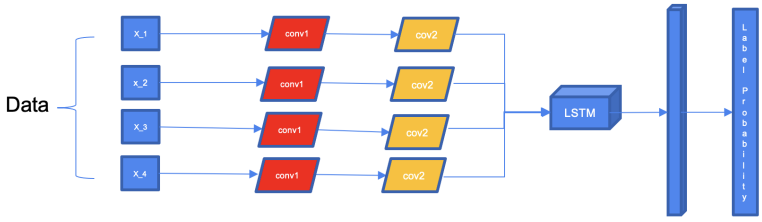


Figure 6. Time Sequence Model Structure



3.5. Time Sequence Model

The structure of the model was shown in Figure 6. Instead of spliting the features, Time Sequence Model splited the data based on the time sequences. Each input data was consisted of 32 consecutive frames, and 32 frames was split into 4 groups. Each group was then processed by two layers of convolutional neural network. Different from the previous model, this model assumed that five features should be considered as a conhesive bundle, and the gait charactristics were hidden in time.

Conv1 was a layer of 3x3 1D convolution kernel with stride of 1 and padding of 1. The input channel for **Conv1** was 5 and output channel was 5. It was follow by a normalization layer and a ReLu activation layer. **Conv2** was also a layer of 3x3 1D convolution kernel with stride of 1 and padding of 1. The input channel for **Conv2** was 5 and output channel was 5. Similarly, it was follow by a normalization layer and a ReLu activation layer.

The results of four time series were the concatenated together and passed to a **LSTM** model. The hidden size of the **LSTM** model was 5 and the number of layers was 2. Finally, the result was passed to a fully connected layer to get the desired output dimension.

4. Experiments

4.1. Implementation

Due to the size of the original datatst, this project only took a subset from the original dataset. The subset used in this project was the scene2 when one participant walking in the scene, and five unique participants were been recognized in this project. The dataset for each participant after preprocessing was relatively small. Thus, augmentation technique was been used to increase the size of the dataset. A rotation martix was been applied on the 3D coordiantes of the dataset. Given that the antennas of the radar was able to scan $\pm$ 20 degrees horizontally [7], a random angle in range of -20 to 20 degrees was used to rotate the coordinates in y axie. The **Doppler** and **Intensity** features remained the same. Since the rotation matrix was only applied on the coordinate features, it prevented the linear transformation issue in the data augmentation. The dataset was been augmented to twice the size for both models.

Both models were trained with the same hyperparameters. The training batch size was 64. Adam optimizer was been used along with cross entropy loss. Both models were trained for 40 epochs, and the learning rate was 0.01. Both models were implemented using Pytorch.

Both models returned arrays of label probabilities. The max probabilities were been used as the prediction, and predictions were compared against the true labels to evaluate the accuracy.

4.2. Results & Analysis

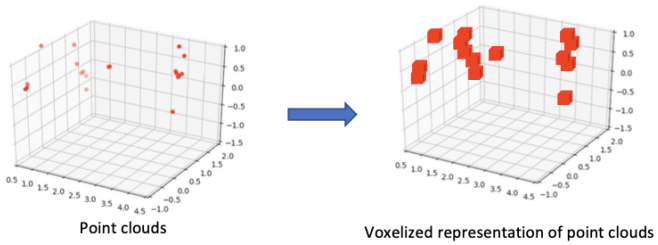
The Feature Sequence Model was able to achieve an accuracy of 80.86%. The Time Sequence Model was able to achieve an accuracy of 82.03%.

Comparing the results of Feature Sequence Model with [7], which used similar model structure, the model was able to achive an accuracy of 86% when there were 10 unique participants. The accuracy of the Feature Sequence Model was slightly lower than that of [7]. The potential reason could be that the Feature Sequence Model was more complicated than that of [7], which required more data samples for training, but the dataset in this project was limited.

For Time Sequence Model, it was similar to the model proposed in [10]. However, the data representation used in this project was different. Authors of [10] only used the 3D coordiantes of the point cloud and converted the 3D coordinates into a voxelized representation. The voxelized representation was a voxelized 3D space that was made of several 3D blocks. Each 3D block was a small 3D region with initial values of all zeros. If there were n points fall in a 3D block range, the value of that 3D block was increamented by n. An expample of the voxelized representation was shown in Figure 7, where red points in point cloud representation were sampled into a voxelized representation.

Instead of gait recognition, [10] was trying to identity 5 human activities(Boxing, Jumping Jacks, Jumping, Squats, and Walking). The recognition accuracy achived by [10] was 90.47%. The advantage of voxelized representation was that it was able to reconstruct the 3D position of a person, and the model could learn the features directly in 3D space. As for the data representation used in this project, five features were pass to the model in time sequence, and the model needed to exract features from xyz coordinates as well as doppler velocity and intenisty on its own. The major disadvantage of the voxelized representation was that majority of the data in voxelized representation was zeros, and useful information was only stored very sparsely in 3D matrices, which could greatly increase size of the dataset and load for the model.

Figure 7. Example of transforming point cloud representation into voxelized representation. [10]



Comparing the Feature Sequnce Model with the Time Sequence Model, the Featrue Sequence Model assumed that recognition was done by observing the change of each

feature over time individually. Thus, Feature Sequnce Model extracted each five features independently before fuse them together for the final step. On the other hand, Time Sequnce Model assumed that the recognition was done by observing the change of five features over time as a whole. Thus, it extracted features for each sequential time intervals first before fusing them together for the final step.

Based on the experiments, two models could achieve very similar results in term of accuracy with slightly better performance in Time Sequence Model. Also, comparing to the voxelized representation used in [10], Time Sequence Model was able to acheive close performance by only using arrays of five features which was much more space efficient.

5. Conclusion

In conclusion, this project experimented the gait recognition task using millimeter wave sensor collected dataset. Two different model structures were used to implement the gait recognition task. Performace of Featrue Sequence Model was compared with that of Time Sequence Model. The experiments showed that both models were able to effectively recognize gait charactristics. This project also compared the performace of different data representations. Using the same Time Sequence Model, the feature array representation was able to achieve close performance comparing to the voxelized representation with much smaller model input size.

For some future works, same models and data representations can be tested on dataset with multiple persons in the scene walking simutaneously. Other data representations such Doppler-Range map are also worth testing under the same test settings.

References

[1] Keisuke Edanami and Guanghao Sun. Medical radar signal dataset for non-contact respiration and heart rate measurement. *Data in brief*, 40:107724, 2022. 2

[2] Piotr Grobelny and Adam Narbudowicz. Mm-wave radar-based recognition of multiple hand gestures using long short-term memory (lstm) neural network. *Electronics*, 11(5):787, 2022. 2

[3] Mi He, Yongjian Nian, and Yushun Gong. Novel signal processing method for vital sign monitoring using fmcw radar. *Biomedical Signal Processing and Control*, 33:335–345, 2017. 2

[4] Xinyu Li, Yuan He, and Xiaojun Jing. A survey of deep learning-based human activity recognition in radar. *Remote Sensing*, 11(9):1068, 2019. 2

[5] Chris Xiaoxuan Lu, Stefano Rosa, Peijun Zhao, Bing Wang, Changhao Chen, John A Stankovic, Niki Trigoni, and Andrew Markham. See through smoke: robust indoor mapping with low-cost mmwave radar. In *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*, pages 14–27, 2020. 2

[6] Yasushi Makihara, Atsuyuki Suzuki, Daigo Muramatsu, Xiang Li, and Yasushi Yagi. Joint intensity and spatial metric learning for robust gait recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5705–5715, 2017. 1

[7] Zhen Meng, Song Fu, Jie Yan, Hongyuan Liang, Anfu Zhou, Shilin Zhu, Huadong Ma, Jianhua Liu, and Ning Yang. Gait recognition for co-existing multiple people using millimeter wave sensing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 849–856, 2020. 1, 2, 3, 4, 5

[8] mmGait. people-gait. https://github.com/mmGait/people-gait, 2020. 1

[9] Samsung. Use motions and gestures to control your galaxy phone. https://www.samsung.com/us/support/answer/ANS00086302/, 2022. 2

[10] Akash Deep Singh, Sandeep Singh Sandha, Luis Garcia, and Mani Srivastava. Radhar: Human activity recognition from point clouds generated through a millimeter-wave radar. In *Proceedings of the 3rd ACM Workshop on Millimeter-wave Networks and Sensing Systems*, pages 51–56, 2019. 1, 2, 4, 5, 6

[11] Yuheng Wang, Haipeng Liu, Kening Cui, Anfu Zhou, Wensheng Li, and Huadong Ma. m-activity: Accurate and real-time human activity recognition via millimeter wave radar. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8298–8302. IEEE, 2021. 2