Application of mmWave Rador Sensor for People Identification and Classifition 논문 리뷰

류영교

Abstract

: 높은 정확도의 실내 Device-free 사람 인식 서비스가 필요한 상황. 카메라 같은걸 이용한 visual methods는 밝기 조건이나 clear한 view가 제공되지 않으면 사용하기 어렵다. 또한 privacy 문제도 있다. 이 논문에서는 mmWave 레이더와 dbscan, 양 방향 LSTM을 사용한 사람 identification과 classification system을 소개한다.

1. Introduction

: 실내에서 사람을 인식하는 일은 보안이나 에너지 관리 등의 면에서 중요한 일이 되었다.
사람을 인식하는 기기에는 ID card나 스마트폰, 스마트 팔찌 같은 것들이 나와있지만 이것은 사람과 떼 놓을 수 없다는 한계점이 있다. 이를 해결하기 위해 카메라를 사용하는 것은 보안상의 문제가 있다. 따라서 radio signal frequency method가 제안되었지만 두 사람을 잘 구별하지 못한다거나 하는 한계가 있다. Point cloud data를 사용한 LiDar같은 것도 제안되었는데 cost-effective하지 않다는 점에서 한계가 있었다.
따라서 여기서는 mmWave와 LSTM, random forest classifier를 사용해서 사람을 인식하고 구별하는 시스템을 제안한다.

2. Related Works

- Abdu, F.J.; Zhang, Y.; Fu, M.; Li, Y.; Deng, Z. Application of deep learning on millimeter-wave radar signals: A review. Sensor 2021, 21, 1951. : 딥러닝에 사용될 레이더 신호의 input representation이 나오는 논문

- Lombacher, J.; Laudt, K.; Hahn, M.; Dickmann, J.; Wöhler, C. Semantic radar grids. In Proceedings of the 2017 IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, USA, 11-14 June 2017; pp. 1170-1175.: CNN을 사용해 rador 신호에서 static object classification 방법을 제안한 글, static한 object는 특성상 데이터가 많이 손실되기 때문에 unstable classification으로 이어진다.

- Major, B.; Fontijne, D.; Ansari, A.; Teja Sukhavasi, R.; Gowaikar, R.; Hamilton, M.; Lee, S.; Grzechnik, S.; Subramanian, S. Vehicle detection with automotive radar using deep learning on range-azimuth-doppler tensors. In Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops, Seoul, Republic of Korea, 27 October-2 November 2019; pp. 1-9 : LSTM 딥러닝 기반 vehicle classification 소개, Range, Velocity, Azimuth를 사용했고 LSTM을 사용했다.

- Guo,Y.; Bennamoun, M.; Sohel, F.; Lu, M.; Wan, J. 3D object recognition in

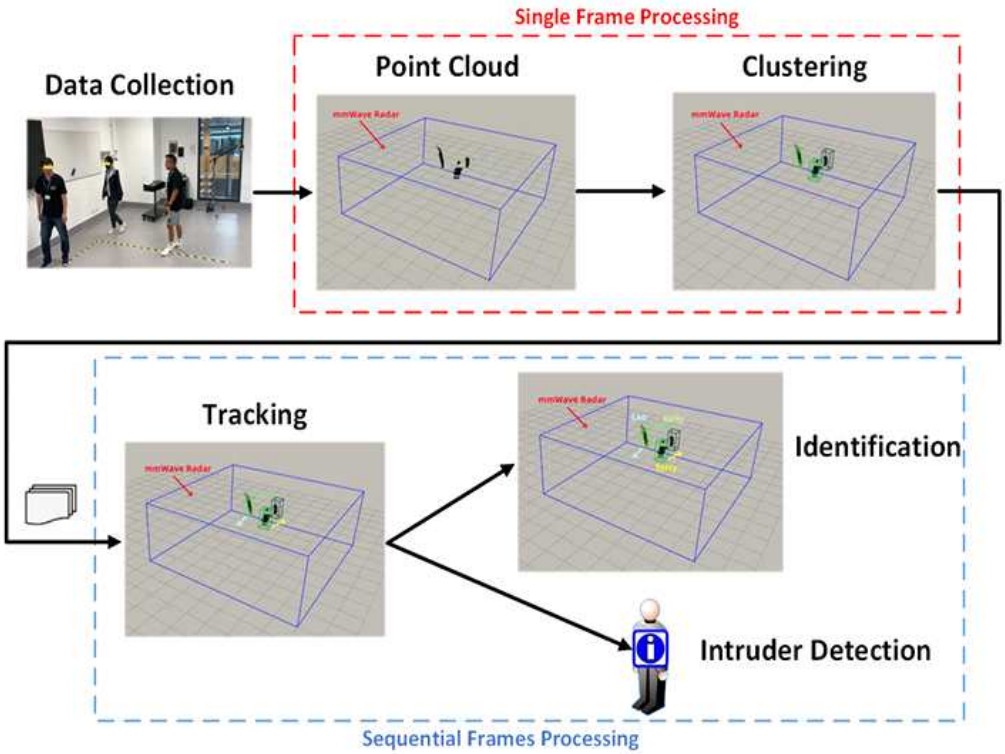
cluttered scenes with local surface features: A survey. IEEE Trans. Pattern Anal. Mach. Intell. 2014, 36, 2270-2287. : point cloud를 geometric information을 유지하면서 3D plotting하는 것에 대한 내용의 글

- Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. Adv. Neural Inf Process. Syst. 2017, 30, 1-10. : LiDAR point cloud를 처리하기 위해 사용한 PointNet을 PointNet++로 만드는 내용의 글. 2D 물체 identification과 bounding box estimation을 원래 3D 모델에 적용했다.

- Lee, S. Deep learning on radar centric 3d object detection. arXiv 2020, arXiv:2003.00851. : public radar 데이터 셋에 추가 레이블을 붙여서 학습을 진행한 애용의 글, overfitting 문제를 해결했다고 한다.

- Simony, M.; Milzy, S.; Amendey, K.; Gross, H.M. Complex-yolo: An euler-region-proposal for real-time 3d object detection on point clouds. In Proceedings of the European Conference on Computer Vision(ECCV) Workshops, Munich, Germany, 8-14 September 2018; pp. 1-13. : LiDAR point cloud 데이터를 rador points dataset으로 변형하고 Complex-YOLO에 사용한 내용의 글

3. System Design



point cloud 데이터는 range FFT, Doppler-FFT, 각도 추정을 통해 TLV 데이터 포맷으로 저장된다. 이후 데이터를 가지고 데이터에 맞게 변형된 DBSCAN 클러스터링 알고리즘을 돌린다. 그리고 시계열 데이터를 분석하기 위해 deep recurrent neural network을 사용하고 이

군집이 intruder인지 insider인지 구분하기 위해 Random Forest classifier를 사용한다. 이것은 feature distance to centers가 얼마나 되는지에 따라 구분되는데 softmax-loss와 center-loss의 균형을 맞춰서 계산된다. 아래에 자세한 설명이 있다.

3.1. Point Cloud Genereation and Static Clutter Removal

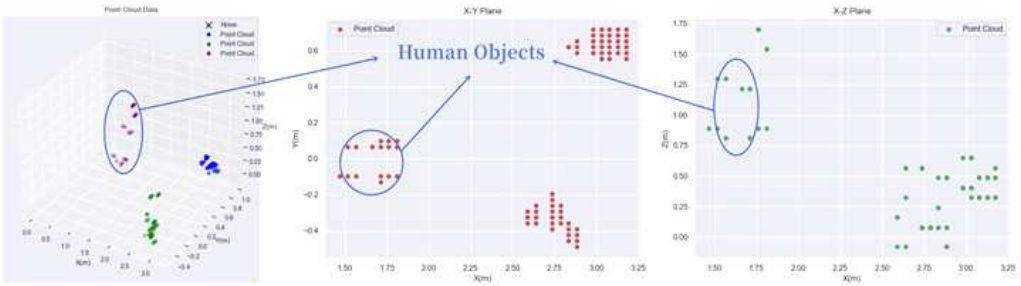
point cloud 데이터는 TLV 포맷으로 저장된다. 데이터 파싱은 프레임 헤더에서 packet length, frame number, number of TLV를 읽는 것으로 시작된다. TLV size도 헤더에 들어있는데 point들의 개수는 이것에 관련되어있고 data frame의 정확한 위치에 값을 읽어오는데 사용된다.

Static clutter removal 과정:

1. 고속 푸리에 변환으로 Range를 process한다. output은 range bins이다.
2. estimated Direct Current(추정되는 직류 성분)을 각 range bin에 빼서 static clutter removal을 수행한다.

3.2. Clustering

이렇게 나온 데이터는 DBSCAN으로 클러스터링 된다. 하지만 수집된 데이터의 밀도는 거리 측면에서나 시간에 따라서나 매번 다를 수 있다. 따라서 정확도가 낮아지기 때문에 여기서는 변형된 DBSCAN을 사용한다.



Z축의 (vertical) 특징이 inconsistency를 보였기 때문에 클러스터링에서 사용하는 유클리디안 거리에서 z축에 대한 가중치를 낮췄다. (아래 식에 알파가 가중치다)

$$D^2(p^i, p^j) = (p_x^i - p_x^j)^2 + (p_y^i - p_y^j)^2 + \alpha(p_z^i - p_z^j)^2$$

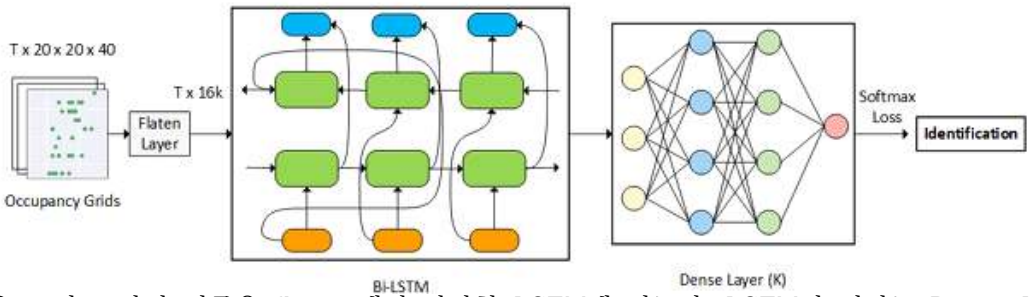
3.3. Tracking

클러스터링을 통해서 detect된 클러스터들의 centroid들은 프레임 별로 이어져야 한다. 이를 위해서 simplified Global Nearest Neighbour 알고리즘을 사용했고 각 트랙(클러스터)은 연속적인 프레임에서 이어진다. 트랙들은 프레임이 진행되면서 inactive 해질 수도 있고 deleted 될 수도 있다. 정확한 트랙을 예측하기 위해 재귀적인 칼만 필터를 사용했는데 이 논문에서는 data association은 다루지 않는다고 한다.

3.4 identification

detection과 tracking 과정을 통해서 track된 points들이 나왔을 것이다. 만약 각 객체가 경계 박스 안에 있다고 하고 이 안에 그 객체의 점들이 있다고 하면 객체를 각 trajectory

frame에서 점유 그리드안에 있다고 할 수 있다. 이 점유 그리드는 예를 들면 키나 중심의 밀도같은 사람의 정보를 담고 있을 것이다. 여기에 사람 몸의 정보같은 내용을 함께 분류기에 입력해서(레이블링을 뜻하는 듯 하다) tracked object의 identity를 결정할 수 있다. 하지만 직접 이 점유 그리드에서 사람의 정보를 추출하는 것은 비효율적이다. 따라서 여기서 양방향 LSTM을 사용한다.



점유 그리드 안의 점들을 flatten해서 양방향 LSTM에 넣는다. LSTM의 결과는 Dense Layer에 들어가고 softmax loss를 통해 classification 결과가 나오게 된다. 여기서 k는 클러스터 개수고 T는 data frame의 개수다. K는 layer size이다.

3.5 Intruder Detection

out-of-set samples는 intruder로 처리해야 하는데 방금 identification에서 사용한 softmax 층은 다항분포로 밖에 intruder를 예측하지 못한다. scattered features에 대한 softmax loss는 intruder에 구별에 악영향을 미친다, 따라서 center loss를 사용한다.
방법은 아까의 신경망에서 샘플 특징을 뽑아내서 intra-class 거리를 center loss로 최소화하는 것이다.

softmax와 center loss의 조합은 아래 식과 같다.

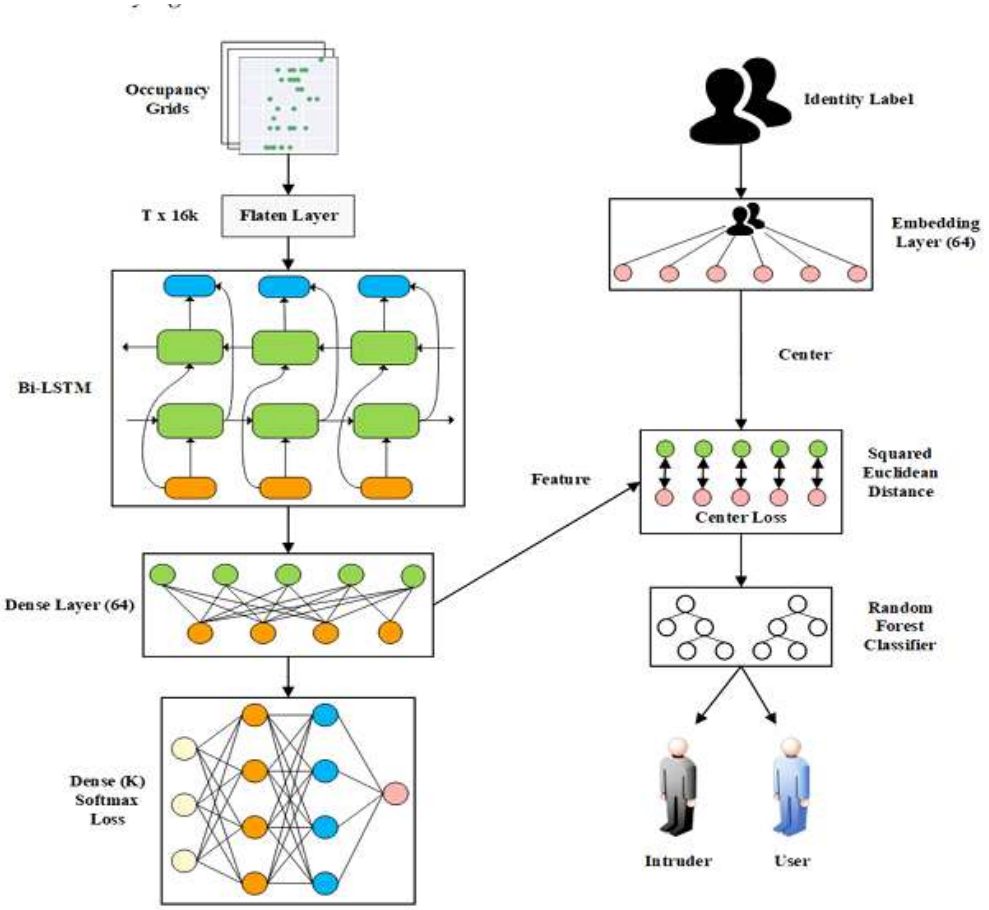
$$\ell = \ell_{softmax} + \lambda \ell_{center}$$

람다는 softmax와 center loss의 균형을 맞춰주는데 여기서는 0~0.1 사이에서 모델을 학습했다고 한다.

$$\ell_{softmax} = -\sum_{i=1}^{m} \log \frac{e^{W_{y_i}^T \mathbf{f}_i + b_{y_i}}}{\sum_{j=1}^{g} e^{W_j{}^T \mathbf{f}_i + b_j}},$$

$$\ell_{center} = \sum_{i=1}^{m} \|\mathbf{f}_i - \mathbf{c}_{y_i}\|_2^2$$

softmax와 center loss에 대한 식이다. g는 m개 샘플의 학습 데이터중 객체의 개수를 의미하고 f는 특징이다. c는 특징의 중심을 나타내고 W,b는 가중치와 bias이다.



아까 그림에서와 다르게 양방향 LSTM 바로 직후의 Dense layer에서 클러스터 특징을 뽑아낸다. 객체의 ID를 객체의 중심에 레이블링 하기 위해서 임베딩을 사용하는데, inner 클러스터의 임베딩된 특징과 이 뽑아낸 특징의 유클리드 거리를 통해 center loss를 알아낸다. 이를 softmax와 조합한 loss를 이용해서 Random forest classifier에 넣어서 이것이 일정 값보다 크면 intruder로 판단할 것이다. 이렇게 intruder를 알아낼 수 있다.

4. System Implementation

4.1. Experiment Setup

물체 identification은 레이더 센서와 laptop control terminal로 구성된다. 센서는 IWR6843ISK, 60GHz mmWave sensor kit를 썼고 데이터는 HD 카메라 tracking system과 함께 교차 검증되었다. 레이더 센서로 얻어진 point cloud는 laptop terminal로 옮겨져 processing 되었다.

샘플들의 identity들은 직접 레이블링 했고 classifier는 케라스와 텐서플로 라이브러리를 사용했다. 성능은 이 직접 레이블링 한 ground truth와 예측 결과를 비교해서 평가했다.

4.2. Data Collection

데이터는 연구실에서 12명의 실험자들이 지정된 곳에서 20분동안 랜덤하게 걷도록 함으로써 진행됐다. 참가자들의 키는 167~186cm였고 몸의 크기는 실제 실내 환경을 고려해서 다르게 하였다.

5. System Configuration

5.1. Sensor Setup

IWR6843ISK 60GHz mmWave 센서 디바이스는 60~64GHz 주파수 range와 4GHz 대역폭에서 사용되었다. 3개의 전송 안테나와 4개의 수신 안테나가 있고 point cloud 데이터를 173.6ms 주기로 생성하고 54.725 MHZ/ms slope로 변화한다. 0,084m의 해상도를 가지고 있고 최대 7.2m까지 구분할 수 있다. 센서의 속도 측정 최대치는 8.38m/s이다. 이 장치는 프레임당 288개의 Chirp를 송신하도록 설정 되었고, 샘플링 속도는 초당 20프레임이다.

5.2. Clustering Parameter Configuration

minClusterSize와 maxDistance 파라미터는 데이터들을 사용하는 휴리스틱한 방법을 통해 얻어졌으며 minClusterSize = 10, maxDistance = 0.8m가 선정되었다.

5.3. Classifier Training Configuration

input data frame은 flatten층에서 16000차원으로 변화된다. 128개의 은닉층을 가진 양방향 LSTM unit들에는 Adam 옵티마이저가 사용되었고 드랍아웃은 0.5가 적용되었다. 훈련/테스트 데이터 비율은 9:1이고 overfitting을 막기위해 데이터는 X,Y축으로 움직이고 회전된다. 모델은 에포크 32번을 돌린다.

5.4. Intruder Detection Configuration

특징을 추출하는 부분은 64개 unit의 dense layer로 한다. overfitting을 막고 성능 유지를 위해 softmax와 center loss의 가중치는 1:0.5로 한다. LSTM과 특징 추출 layer의 드랍아웃은 0.5로 해서 성능과 overfitting을 막았고 에포크는 32번으로 모델을 학습했다. Random Forest Classifier의 파라미터는 k-fold 교차검증(k = 5)와 grid search로 구했고 그렇게 구해진 파라미터는 아래와 같다.

```
# Train/Test split
x_train, x_test, y_train, y_test =
train_test_split(X, y, test_size = 0.25, random_state = 18)
# Model training
clf = RandomForestClassifier(n_estimators = 500, max_depth = 4, max_features = 3,
bootstrap = True, random_state = 18).fit(x_train, y_train)
```

6. System Evaluation

6.1. Tracking

이 시스템의 tracking system과 Kinect v2 카메라와의 정확도를 비교했다. 두 시스템 모두 연구실 환경에서 평가했고 ground truth는 HD 카메라로 intruder과 inner를 직접 마킹했다. 결과 RMSE가 Kinect v2 카메라는 1.025m인데 이 시스템의 tracking에서는 0.2665m가 나왔다.

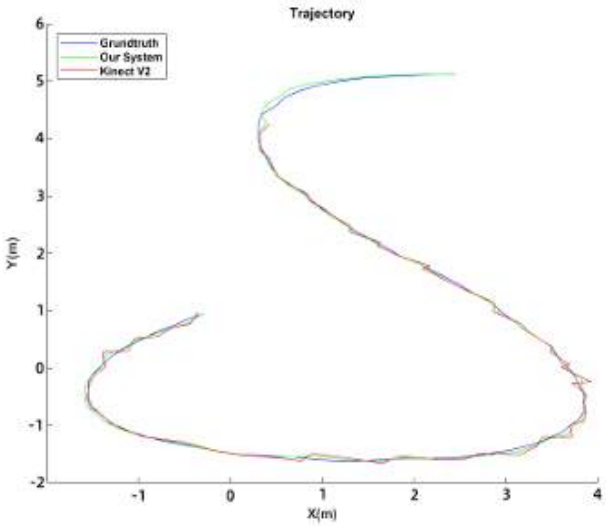


Figure 6. Tacking Trajectory.

Table 1. Tracking Error Comparison Between Kinect v2 and Proposed System.

System	Total Frames	Object Number	RMSE
Kinect v2	5611	1	1.025
Proposed System	5611	1	0.2665

6.2. Identification

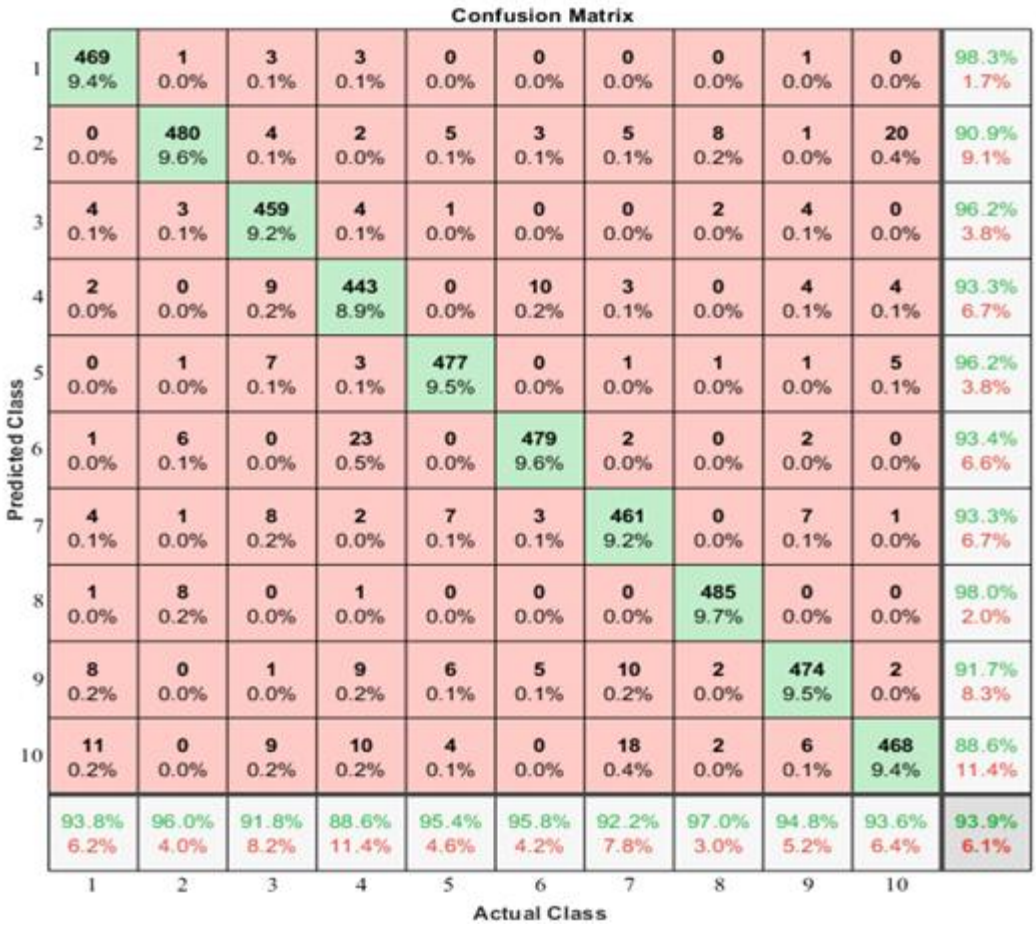
point cloud 데이터에서 traditional vision 방법으로 인간의 보행을 분석하는 것은 인간의 부분을 알아내기 어려운 부분이 있었다. 이걸 극복하기 위해 양방향 LSTM으로 feature을 추출해서 다른 모델 아키텍쳐보다 모델 수렴 속도와 정확도 면에서 이점을 가졌다.

BiLSTM, LSTM, CNN 아키텍쳐를 비교하였고 모두 32 에포크와 0.5 드랍아웃을 적용했다. BiLSTM과 LSTM은 128개의 hidden unit을 사용했고 CNN은 max pooling을 사용한 2개의 컨볼루션 층을 사용했다. 성능 지표는 아래와 같다.

Table 2. The Accuracy Differences Between Architectures.

Architectures	Frames	Accuracy	Loss	Validation Accuracy	Validation Loss
BiLSTM	12,490	0.9522	0.1308	0.8763	0.3218
LSTM	12,490	0.9241	0.1704	0.8581	0.3497
CNN	12,490	0.8819	0.2068	0.8391	0.3518

오버피팅을 막기 위해 cross-validation을 사용했고 prediction은 validation data로 평가되었다. BiLSTM이 다른 두 모델의 성능을 압도했고 따라서 이 모델을 사용했다. 그냥 LSTM은 긴 sequence에 대해 정보를 전달하는 능력이 떨어져서 성능이 떨어지는 것을 알 수 있었다. 아래는 예측 클래스에 대한 confusion matrix이다. 행이 예측 클래스고 열이 실제 클래스다.



그림의 제일 오른쪽 열은 어떤 클래스로 예측 했을 때 그 클래스로 예측한 데이터 전체 분의 실제로 그 클래스인 데이터이고 제일 아래 행은 실제로 데이터가 어떤 클래스일 때 그 클래스로 예측한 모든 데이터 분의 실제로 그 클래스인 데이터이다. 제일 오른쪽 제일 아래 셀은 전체 데이터에 대해 맞힌 비율을 나타낸다. 따라서 93.9퍼센트의 정확도가 나오고 10번째 사람을 identify하는 것은 정확도가 93.6퍼센트가 나온다는 걸 알 수 있다.

6.3. Intruder Detection

intruder detection은 intruder를 모델에 넣었을 때 나온 softmax loss와 특징 추출후에 계산한 center loss를 가지고 평가했다. training, test 데이터 셋은 1:1이다.

아래는 성능 평가 결과다.

Intruders	Precision	Recall	F1	Accuracy
2	0.9352	0.9863	0.9522	0.9681
4	0.8719	0.9581	0.9080	0.9278
6	0.8290	0.9021	0.8570	0.8825
8	0.8068	0.8941	0.8430	0.8720
10	0.7879	0.8777	0.8319	0.8287

intruder 수를 2~10으로 늘려가면서 평가했는데 precision은 intruder를 intruder로 classify한 비율, recall은 identify된 intruder의 비율, F1은 그냥 F1 score, Accuracy는

정확하게 categorized된 샘플의 비율이다.
intruder 수가 늘면 고차원 feature에 공간에서 샘플이 너무 흩어져버려서 잘못 classification을 하게 돼서 정확도가 떨어진다고 한다. 향후 목표는 이것을 개선하는 것이다.

6.4. Discussion on Robustness
실험 준비를 할 때, 센서의 최대 감지 거리를 6m로 했다. 원칙적으로는 30m까지 가능하지만 거리가 늘어나면 정확도가 떨어지고 SNR이 늘어나게 될 것이다.

측정을 하는 동안 mmWave 센서가 창문이나 거울의 반사에 영향을 받는걸 발견했다. 이런 환경에서 노이즈로 볼 수 있는 object가 갑자기 나타나는 현상이 일어난다. 실험 환경에서는 창문이 하나밖에 없고 벽만 있었기 때문에 상관이 없었지만 이런걸 고려해 보는것도 좋다고 생각한다.