ITU-ML5G-PS-009: RF-Sensor Based Human Activity Recognition Using Hierarchical Classifier

Team Name: Orion

Team Members:

1. Pedamalli Saikrishna
2. Ashwini Kumar
3. Ashok Kumar Reddy Chavva
4. Sripada Kadambar

Introduction

RF sensors are widely being used for a lot of activities like activity and gesture recognition, vital signs monitoring and border intrusion control among many others. Their use has increased manifold over the years owing to the availability of RF-sensor data which has become easily accessible due to the low cost of RF-sensors. Nevertheless, it is still a daunting task to develop deep learning models on such a small dataset, with model overfitting on the training data still remaining as one of the challenging tasks along with validation accuracy remaining (~82-86%) across all datasets.

To overcome these challenges, we propose a hierarchical classifier which exploits the innate similarity between certain activities such as (picking/bending) or (limping/short step/ scissor gait) and clubs them together as a single class and then uses a new classifier to classify among these classes thus creating a layer of classifiers on top of each other and hence the name hierarchical classifier.

Experimental Setup and Datasets

The datasets were taken from [https://github.com/ci4r/CI4R-Activity-Recognition-datasets]. It consisted of data from 3 different sensors. They are:

- 77 GHz FMCW radar: The Texas Instruments IWR1443 Frequency Modulated Continuous Wave (FMCW) transceiver was set at a 77 GHz center frequency and 750 MHz bandwidth.
- 24 GHz FMCW radar: An Ancortek SDR-Kit was set to transmit an FMCW waveform at 24 GHz center frequency and 1500 MHz bandwidth.
- XeThru UWB impulse radar: The XeThru X4 sensor transmits across a band of roughly 7 GHz - 10 GHz.

Proposed Hierarchical Classifier

For each dataset, we found through experimentations that clubbing classes (2&3) and (8,9,10) together yielded the best results. Thus, we have 3 classifiers in place:

1. To classify between classes – [0,1,{2/3},4,5,6,7,{8/9/10}].
2. To classify between classes – [2,3].
3. To classify between classes – [8,9,10].

If the prediction of Stage 1 classifier is '{2/3}' then the image is sent to Stage 2_1 classifier for further classification. Similarly, if the prediction of Stage 1 classifier is '{8/9/10}' then the image is sent to Stage 2_2 classifier for further classification.

Network Architecture for the Proposed Hierarchical Classifiers:

We resized the input images to size (128,128) and passed them through an inception layer with kernel sizes (3,11), (5,21) and (9,9). The output from these filters were concatenated and returned as an output of the inception layer. This is then fed to MaxPooling(MP) (Strides= (2,2)) and Batch Normalization(BN) and then again passed through another inception layer whose hyper-parameters are same as previous inception layer. The output thus obtained was passed through MP layer (Strides= (4, 4)) followed by a BN layer. Then the output was passed through a convolutional layer with 128 filters of kernel size (5,5) which is followed by MP (Strides= (2, 2)) and BN layer. The output was then flattened and passed through two dense layers of size=80 with dropout=0.2 and then the final output was obtained from the softmax layer. For Stage 1 classifier we used the filter size of inception layer to be 16 and for Stage 2 classifier we set them as 48.

The model was run for 100 epochs with batch_size=32 and adam optimizer and 'sparse_categorical_crossentropy' as the loss function.

Training

All the sensors were trained similarly. For training Stage 1 classifier, images belonging to classes 2 and 3 were put together in a single folder corresponding to output class {2/3}, the same was done for classes 8, 9 and 10 corresponding to output class {8/9/10}. Thus, reducing the number of classes to 8 while training Stage 1 classifier.

For second stage we have two classifiers. Stage 2_1 classifier is trained to classify between classes 2 and 3 and Stage 2_2 classifier is trained to classify between 8, 9 and 10.

All the classifiers are trained independently using a validation split of 50%.

Results

Performance - Accuracy

The following table benchmarks the proposed method with CNN and CAE architectures present in the literature. The entire dataset accuracy in the table corresponds to (training + validation) data together. For the CNN and CAE, the validation accuracy corresponds to the accuracy on validation dataset. For the proposed method as we have trained each of the networks independently , we deduce the validation accuracy from the entire dataset accuracy.

Entire dataset accuracy = (percentage of data for training * training accuracy + percentage of data for validation * validation accuracy). Here, percentage of data for training = percentage of data for validation = 50% = 0.5. Hence, entire dataset accuracy = (0.5 * training accuracy + 0.5 * validation accuracy). For a given entire dataset accuracy (constant), the worst case validation accuracy is when training accuracy is highest (maximum possible = 100%).

Entire dataset accuracy = (0.5 * 100 + 0.5 * validation accuracy).
Entire dataset accuracy = (50 + (0.5 * validation accuracy)).
Entire dataset accuracy - 50 = 0.5 * validation accuracy.
Validation accuracy = 2*(Entire dataset accuracy - 50).
Validation accuracy = (2*Entire dataset accuracy) - 100.

Sensor	Entire dataset Accuracy (%)			Validation Accuracy (%)		
	CNN	CAE	Hierarchical	CNN	CAE	Hierarchical
77 GHz	93	92.9	**95.67**	86.11	85.8	**91.34**
24 GHz	92.27	90.4	**94.15**	84.64	82.86	**88.3**
Xethru	91.15	90.8	**93.19**	82.3	81.74	**86.38**

Performance - Complexity Analysis

The table provides the complexity analysis of the proposed architecture. For obtaining the average complexity we assumed 1 sample from each class. This requires Stage 1 classifier alone for 6 samples, Stage 1 + Stage 2_1 classifiers for 2 samples and Stage 1 + Stage 2_2 classifiers for 3 samples. The average of all these are taken to report the average complexity.

Architecture	Complexity (Giga FLOPs)	No. of Parameters		
		Trainable	Non-trainable	Total
CNN	~8.577	3,261,235	-	3,261,235
CAE	~17.152	3,247,243	-	3,247,243
Stage 1	~1.795	708,824	448	709,272
Stage 2_1	~13.648	2,382,674	832	2,383,506
Stage 2_2	~13.648	2,382,755	832	2,383,587

CNN (Giga FLOPs)	CAE (Giga FLOPs)	Proposed – Hierarchical (Giga FLOPs)		
		Best Case	Worst Case	Average
~8.577	~17.152	~1.795	~15.443	~7.999

Conclusion

We observed ~4-5% improvement in validation accuracy as compared to the next best solution while having the least average complexity compared to solutions available in literature.

References

1. Sevgi Z. Gurbuz, M. Mahbubur Rahman, Emre Kurtoglu, Trevor Macks, and Francesco Fioranelli "Cross-frequency training with adversarial learning for radar micro-Doppler signature classification (Rising Researcher)", Proc. SPIE 11408, Radar Sensor Technology XXIV, 114080A (11 May 2020); https://doi.org/10.1117/12.2559155.
2. C. Szegedy *et al.*, "Going deeper with convolutions," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1-9, doi: 10.1109/CVPR.2015.729859