Sequential Convolutional Recurrent Neural Networks for Fast Automatic Modulation Classification

Kaisheng Liao, Guanhong Tao, Yi Zhong, Yaping Zhang, and Zhenghong Zhang

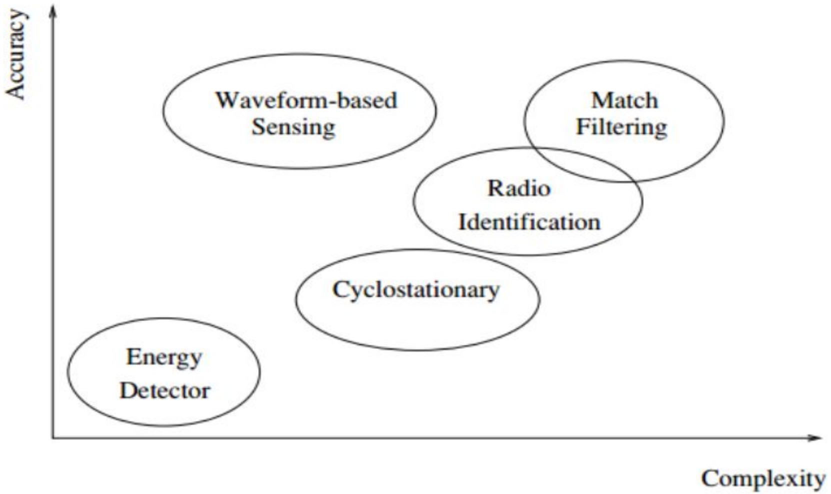
Jeel Bhavsar: es16btech11005
Vignatha Vinjam: es17btech11024

Introduction

Why do we need Automatic Modulation Classification(AMC)?

Growing spectral resource scarcity;

Improving the performance of Cognitive Radios (Core in non-cooperative communication)



Can we stick to the existing methods?

Likelihood-based approach

- Not ideal for unknown channel conditions and heavy computational load

Feature-based approach(traditional)

- Easy practical implementation hence widely studied.

- Current FB AMC methods are commonly designed for a limited set of modulation and lack of generalization ability.

- Very difficult to manually design features for new modulation types in non-cooperative scenarios.

Using Deep Neural Networks

Automatic feature extraction is required for new modulation types and complex environments.

Results from previous research works:

- Convolutional neural networks (CNNs) trained on time domain in-phase and quadrature (IQ) data significantly outperform conventional approaches.

- Using Recurrent neural networks (RNNs) for learning temporal representations achieve higher accuracies than CNNs but suffer from computational complexity and long training time.

- Adopting convolutional long short-term deep neural networks (CLDNNs) has shown some success.

- Improved accuracy by using 2 CNNs (trained on different datasets).

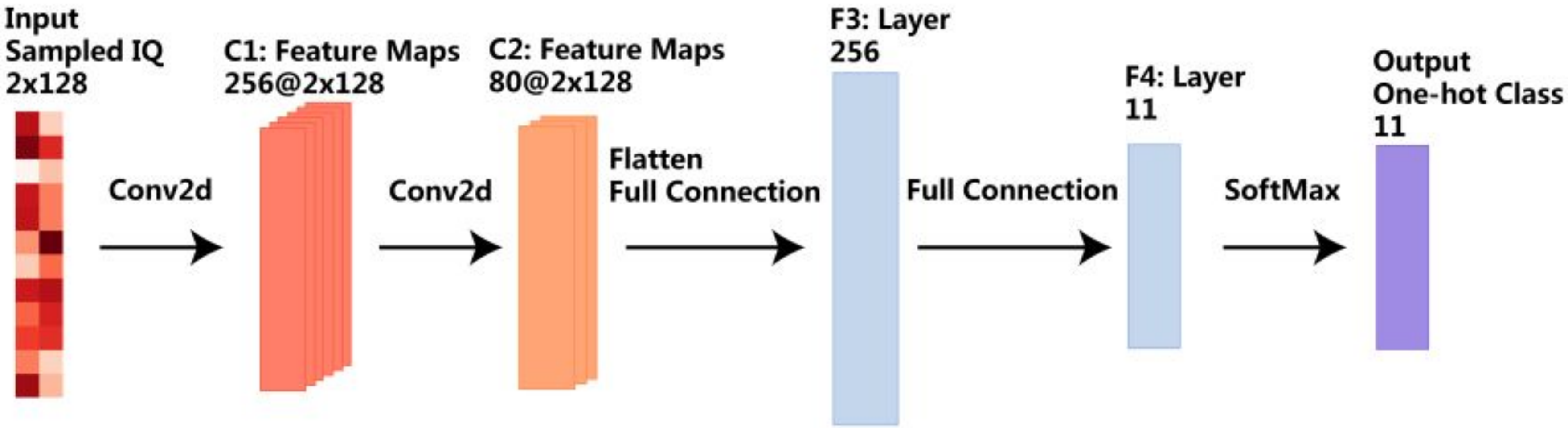
- For various DNNs, reducing input dimensionality (with subsampling techniques) reduced the training complexity (A comparative study performed by Sharan Ramjee, Diyu Yang, Aly El Gamal in the paper- 'Fast Deep Learning for Automatic Modulation Classification')

Benchmark Dataset Description

Representation of received signal:
$r(t) = s(t) * h(t) + n(t)$

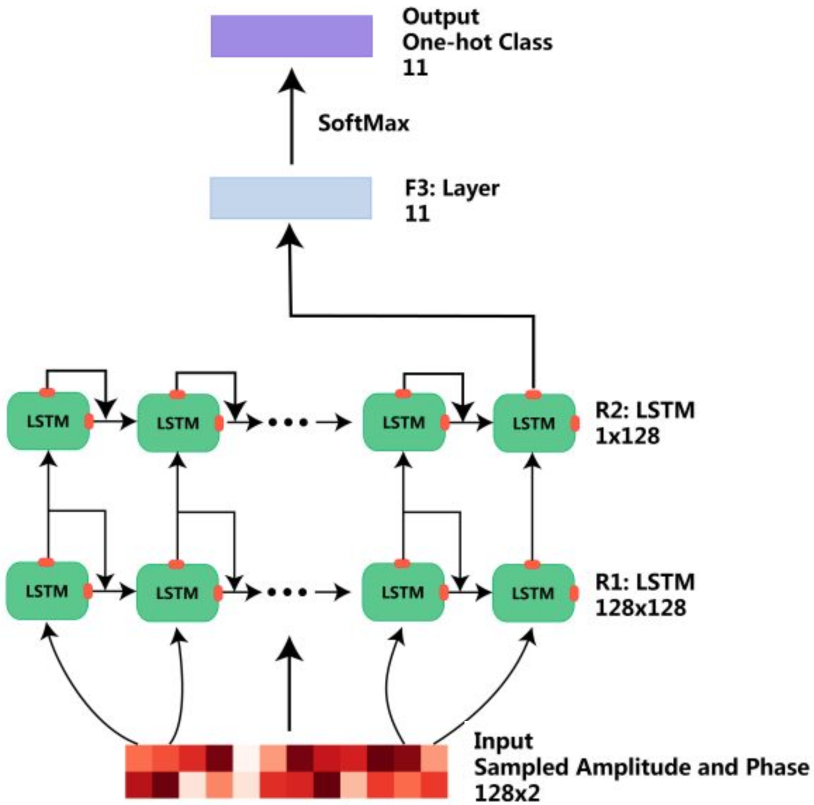
Dataset	RadioML2016.10a
Modulations	8 Digital Modulations: BPSK, QPSK, 8PSK, 16QAM, 64QAM, BFSK, CPFSK, and PAM4 3 Analog Modulations: WBFM, AM-SSB, and AM-DSB
Length per sample	128
Signal format	In-phase and quadrature (IQ)
Signal dimension	2×128 per Sample
Duration per sample	128 μs
Sampling frequency	1 MHz
Samples per symbol	8
SNR Range	[-20 dB, -18 dB, -16 dB, . . ., 18 dB]
Total number of samples	220000 vectors
Number of training samples	198000 vectors
Number of test samples	22000 vectors

Baseline Approaches



CNN architecture proposed by **Timothy J O'Shea, Johnathan Corgan, T. Charles Clancy** in their paper **Convolutional Radio Modulation Recognition Networks**

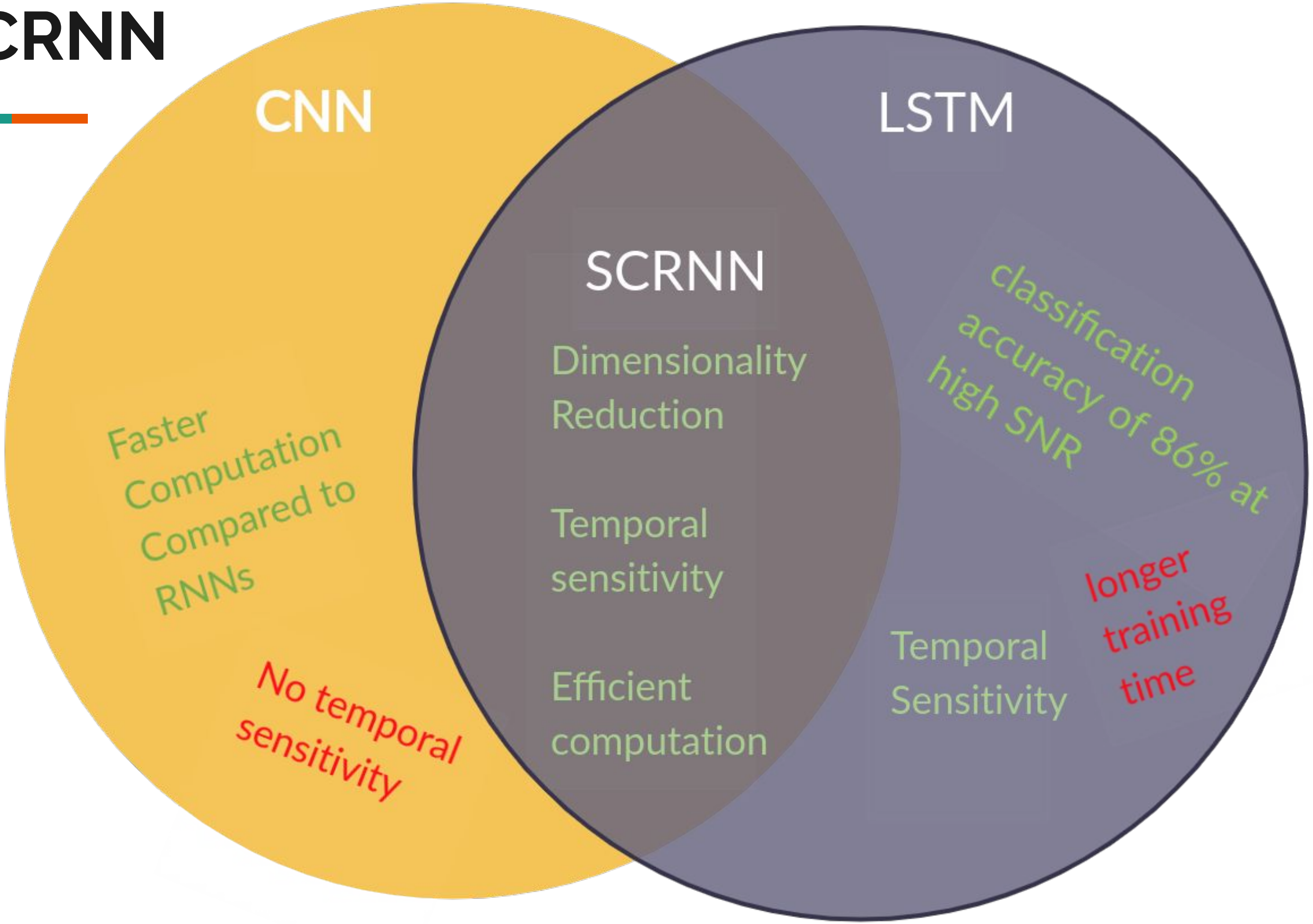
Baseline Approaches



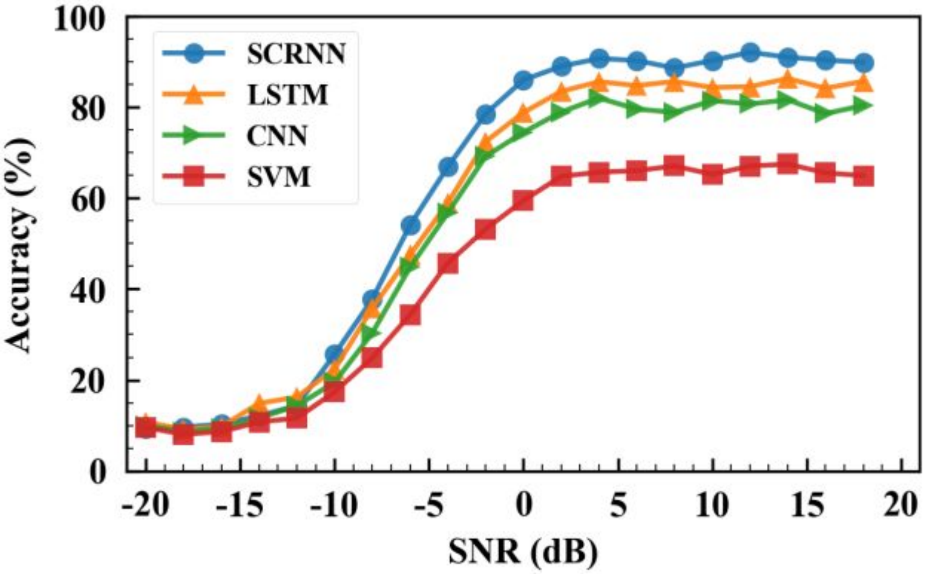
LSTM architecture proposed by **Sreeraj Rajendran, Wannes Meert, Domenico Giustiniano, Vincent Lenders, and Sofie Pollin** in their paper **Deep Learning Models for Wireless Signal Classification with Distributed Low-Cost Spectrum Sensors**

Note that this model learns from the time domain information of the modulation schemes using amplitude-phase format, instead of IQ format.

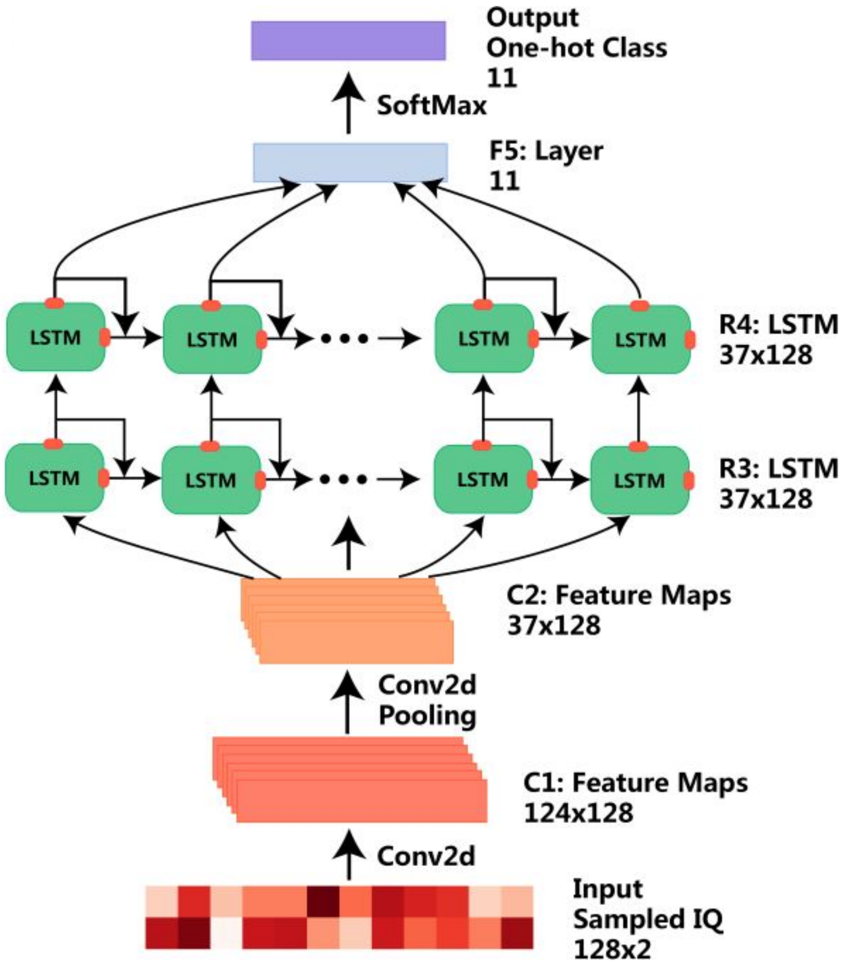
SCRNN


CNN
LSTM
SCRNN
Faster Computation Compared to RNNs
No temporal sensitivity
Dimensionality Reduction
Temporal sensitivity
Efficient computation
classification accuracy of 86% at high SNR
Temporal Sensitivity
longer training time

SCRNN

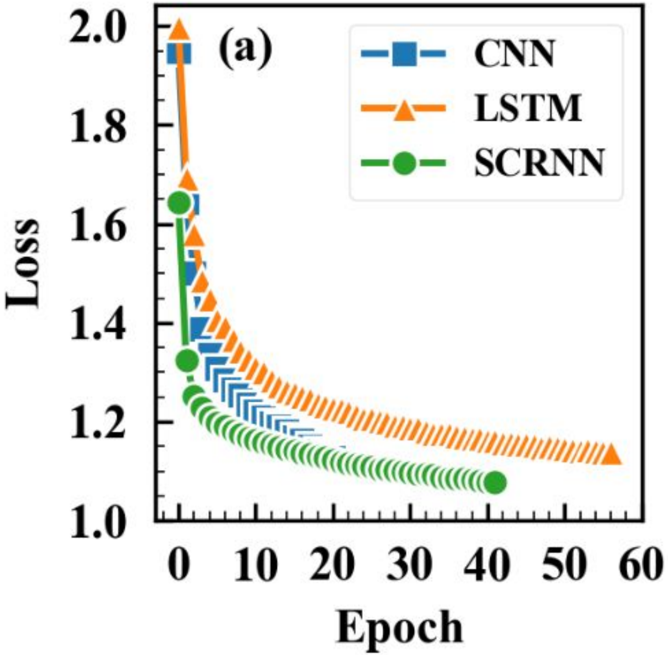


SCRNN
LSTM
CNN
SVM
Accuracy (%)
SNR (dB)
100
80
60
40
20
0
-20
-15
-10
-5
0
5
10
15
20

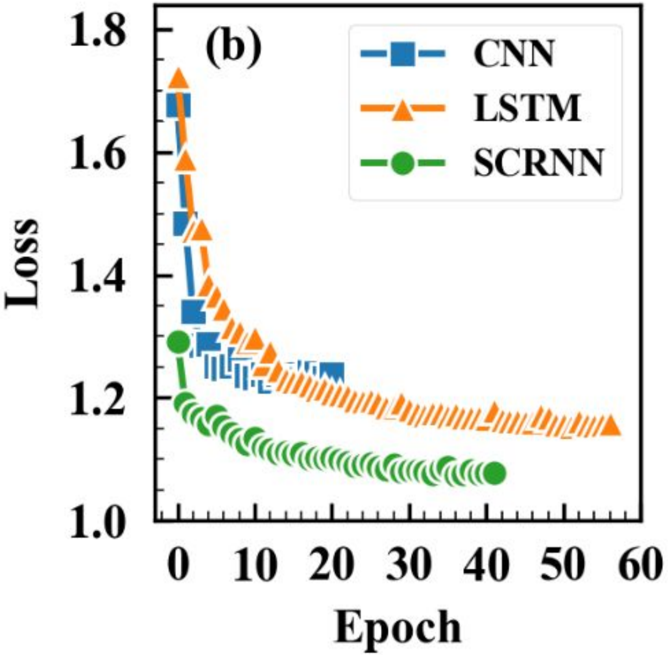


Output
One-hot Class
11
SoftMax
F5: Layer
11
LSTM
R4: LSTM
37x128
R3: LSTM
37x128
C2: Feature Maps
37x128
Conv2d
Pooling
C1: Feature Maps
124x128
Conv2d
Input
Sampled IQ
128x2

Results and Conclusions



Training Loss vs Epochs



Validation Loss vs Epochs

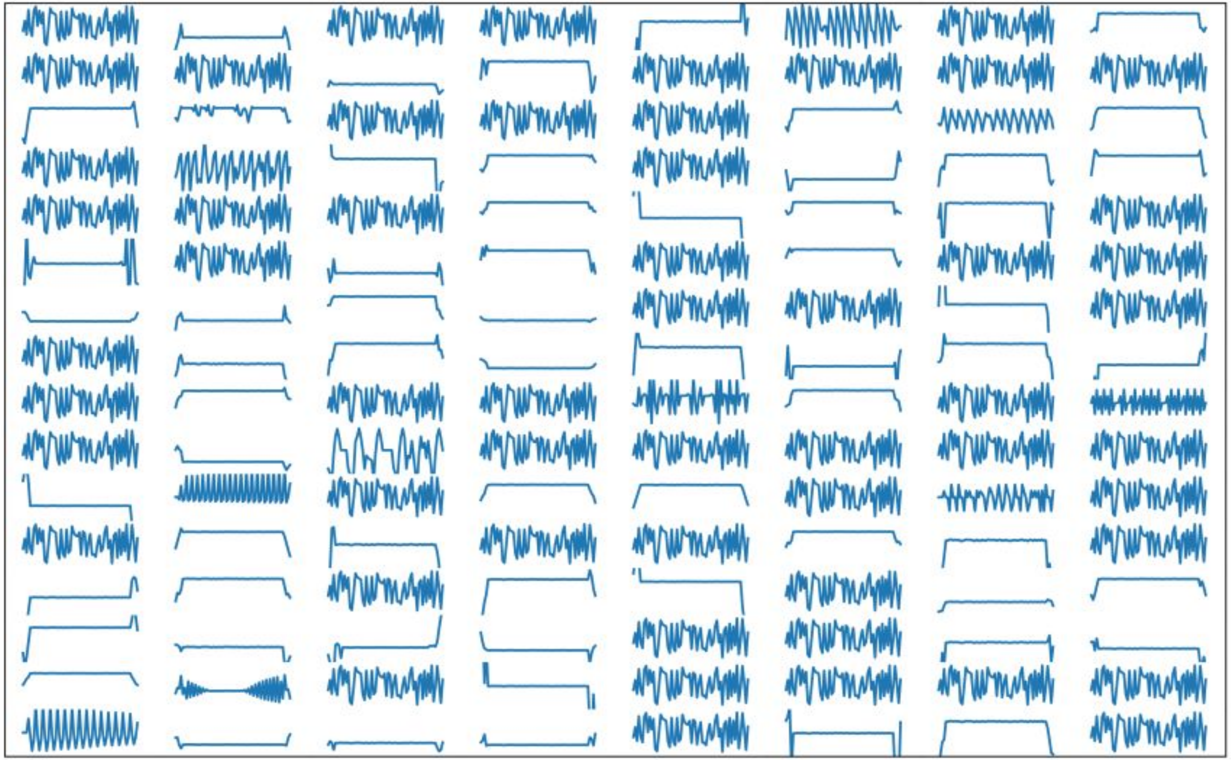
Results and Conclusions

TRAINING AND PREDICTION TIME COMPARISON BETWEEN THE TWO BASELINE MODELS AND THE THREE SCRNN MODELS.

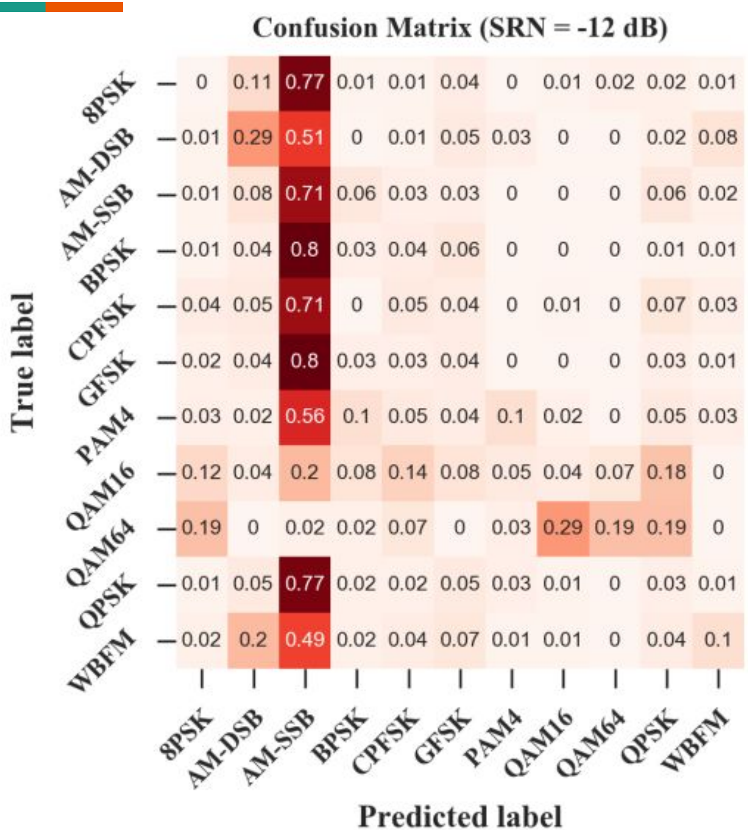
Models	CNN	LSTM	1-Layer SCRNN	2-Layer SCRNN	3-Layer SCRNN
Training Time per Epoch (s)	30	800	800	280	90
Training Epochs	20	56	57	41	57
Total Training Time (s)	600	44800	45600	11480	5130
Prediction Time (μs/sample)	1000	2000	641	661	751

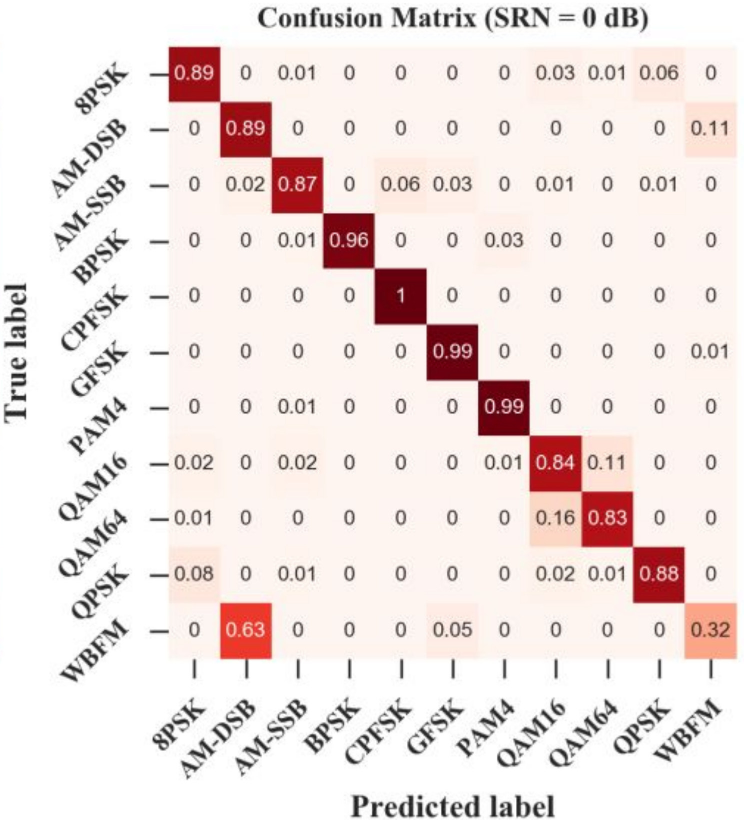
Results and Conclusions

Response patterns of the 128 filters learned by the first convolutional layer of the SCRNN.

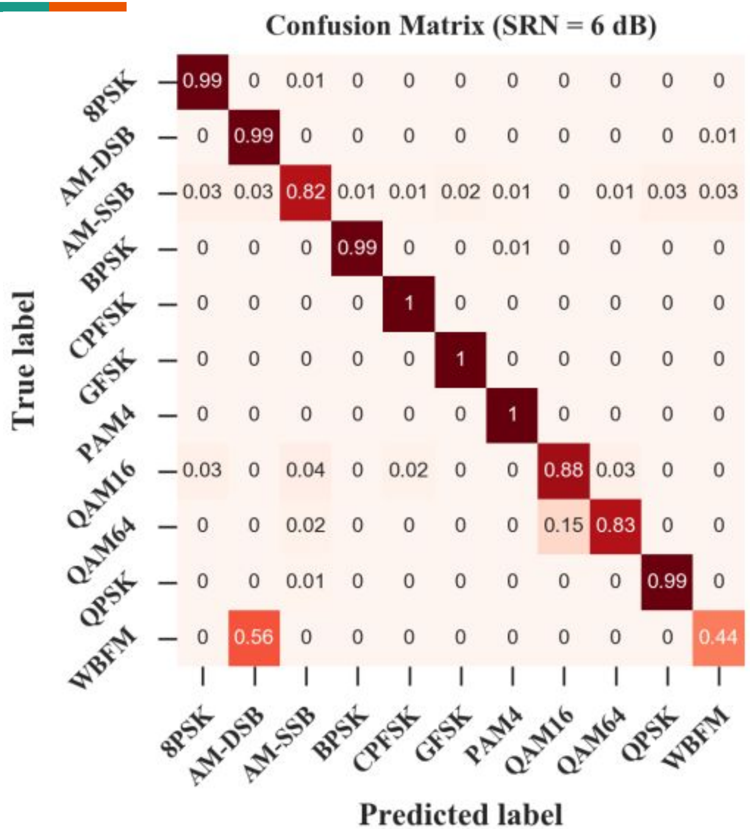


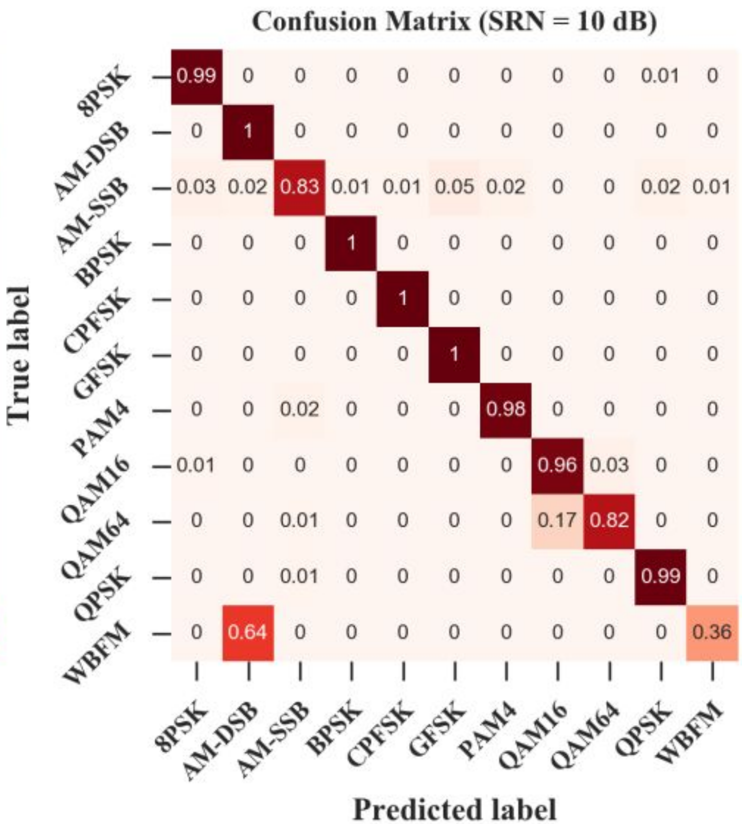
Results and Conclusions





Results and Conclusions

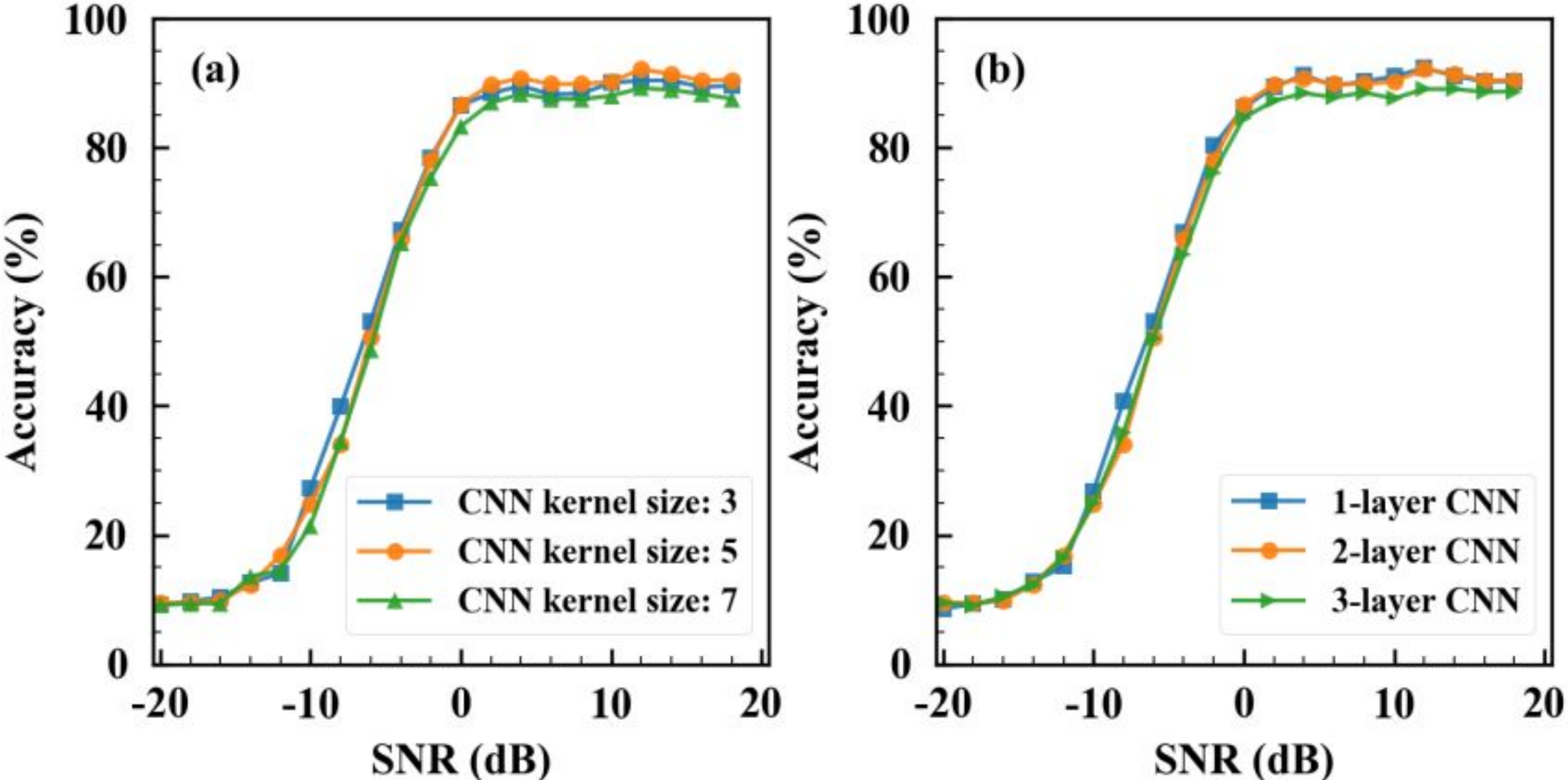




Everything's better with a little Deep Learning Garnish

Thank You!

Results and Conclusions



Results and Conclusions