FYS-STK4155 - Project 3

Markus Bjørklund, Magnus Kristoffersen

December 2023

Abstract

We conduct a study of several neural network architectures applied to the *RadioML 2016.10A* data set, with the purpose of automatic modulation classification. We apply seven different deep neural network models to the data set; five convolutional neural networks and two long short-term memory models, which is a specialized recurrent neural network. We find that a parallel convolutional neural network with two fully connected layers performs the best, achieving an average correct classification of over 75 % for samples with positive signal-to-noise ratio, where several modulation types reach above 95 % accuracy. We find that the two long short-term memory models perform the worst, although this might be due to the models not fully converging after training for the chosen number of epochs. Finally, we find that the performance of the models decreases significantly for increasingly negative signal-to-noise ratios, converging to the accuracy expected from random predictions.

Code, figures and documentation can be viewed on GitHub[1].

1 Introduction

Automatic modulation classification (AMC) describes the process of automatically detecting the type of modulation used to encode information over the air. AMC has many use cases both for cooperative and non-cooperative communication systems, such as a more efficient utilization of the frequency spectrum, adaptive and multipurpose sensors, and fingerprinting. The usage of machine learning in the radio frequency (RF) domain is a relatively new field of study, especially the application of deep neural networks (DNN). Radio frequency is here taken to be the spectrum up to roughly 300 GHz, encompassing most man-made communication systems.

Because of the novelty of this field, there exists a scarcity in large, labeled and high quality data sets which are openly available. To remedy this, a few such data sets have been released by DeepSig, the first in 2016 [1]. This first data set, called *RadioML 2016.04C*, was later revised to the updated *RadioML 2016.10A*, which is the data set we will consider in this work. As this is considered a benchmark in the field, a significant amount of research has been published using this data set.

The application of DNNs on this data set was first shown by O'Shea et al., who implemented a convolutional neural network (CNN) for AMC [2]. They showed a significant

[1] `https://github.com/marbjo/FYS-STK4155/tree/master/project3`

increase in classification accuracy compared to more traditional non-ML methods, such as cyclic spectral features. Hong et al. implemented a recurrent neural network (RNN), and reported an increase in classification accuracy from 80 % to 91 % when comparing with CNN based methods for high signal-to-noise ratio (SNR) [3]. Furthermore, Rajendran et al. utilized a long short-term memory (LSTM) model, which is a special case of an RNN [4]. They report an average accuracy score of close to 90 % in the $0 - 20$ dB SNR range. Ramjee et al. provided a comparison between many different DNN architectures and ways to reduce training time using this data set [5]. Tu et al. reported a superior accuracy for AMC on this data set when employing complex valued networks, while Krzyston et al. reports a peak classification accuracy of 92.4 % using residual and fully connected CNN architectures with complex convolutions [6, 7].

To improve classification accuracy for low SNR, Zhang et al. proposed using a spectral correlation function and a CNN denoising model, in conjunction with a DNN classifier [8]. Further efforts to improve accuracy in the low SNR regime was carried out by Hou et al., transforming the IQ data into an image by a smooth pseudo Wigner–Ville distribution before extracting image features [9]. Qiao et al. proposed a joint denoising and modulation classification method based on multitask learning, Ke and Vikalo presented a learning framework based on a LSTM denoising auto-encoder, and Han et al. employed an attention based multi-layer perceptron along with a smoothed pseudo Wigner-Ville distribution and a convolution auto-encoder to denoise and extract features[10, 11, 12].

In this work, we will apply several deep neural network architectures to the *RadioML 2016.10A* data set, attempting to reproduce some of the results presented in the literature. We will evaluate and compare the accuracy and computational efficiency of the models. We present our method and relevant theory in Section 2, our results and discussion in Section 3 and finally our concluding remarks and future work in Section 4.

2 Method

2.1 Data set availability

The *RadioML 2016.10A* data set is publicly available from DeepSig, and was first presented at the 6th Annual GNU Radio Conference [1]. More details on the generation of the data set can be found in [13].

2.2 Theory

2.2.1 Modulation

In communication systems, sending signals directly at their intrinsic, or baseband, frequency is often unfeasible, due to reasons such as spectrum interference, signal environment and the relation between receiver antenna size and wavelength of the signal, to name a few. Instead, a carrier wave of a certain (generally much higher) frequency is modulated by, or mixed with, the information carrying signal. The signal then has to be demodulated at the receiver to extract the information again. This process allows for efficient spectrum allocation and a higher information load over the air, but depends on a successful demodulation process, or

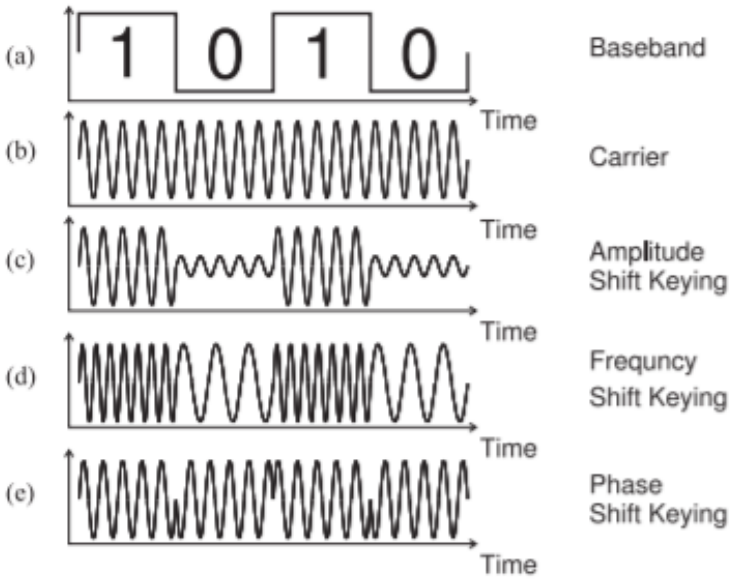


Figure 1: A pedagogical example of how digital data, the baseband signal shown in (a), will modulate a carrier signal (b), for both ASK (c), FSK (d) and PSK (e). Image credit: Michael Steer[15].

else the information is unusable. To account for a constantly growing amount of information being exchanged and the advent of digital data, new modulation types are being employed for different purposes. An instructive example how a carrier wave can be modulated by an information carrying signal is shown in Fig. 1. This section will present some of the many modulation types currently in use, as well as a clever way to represent modulated signals, namely as In-Phase and Quadrature (IQ) signals [14, p. 170-171].

Consider a complex decomposition of a signal

$$c\left(t\right) = e^{j\omega_0 t} = \cos\omega_0 t + j\sin\omega_0 t, \tag{1}$$

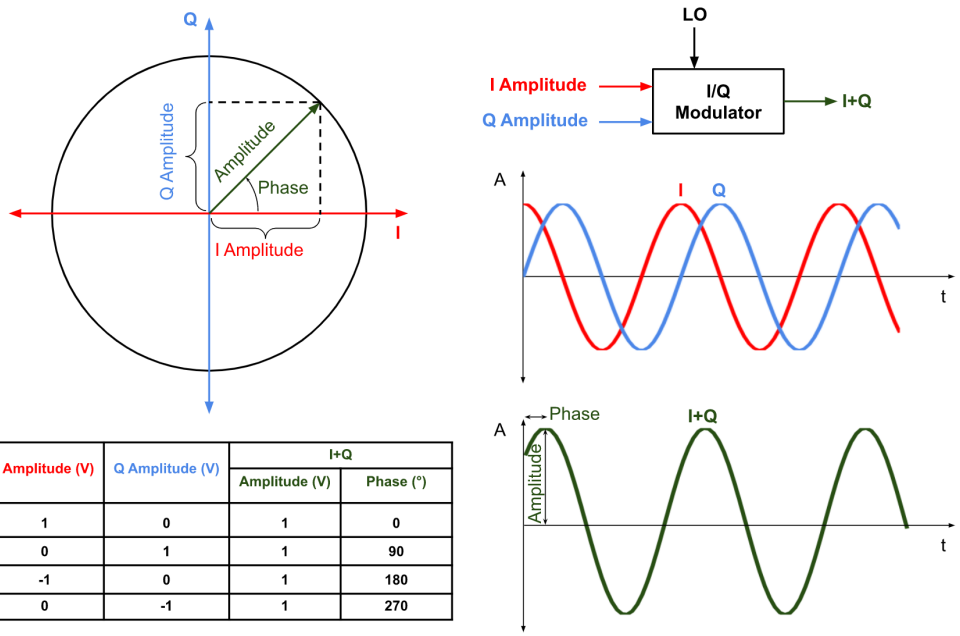
where j represents the imaginary unit. These sinusoids are in quadrature, meaning they have a phase offset of 90° degrees relative to each other. We denote the first term as I, in-phase, and the second as Q, quadrature, corresponding to the real and imaginary parts, respectively. Now, data can be modulated onto this carrier waveform by a new complex function

$$g\left(t\right) = x\left(t\right) + jy\left(t\right), \tag{2}$$

also known as the complex envelope, such that a real world transmitted waveform, or signal, $s\left(t\right)$ is given by

$$\begin{aligned} s\left(t\right) &= \mathrm{Re}\{g\left(t\right)c\left(t\right)\} \\ &= x\left(t\right)\cos\omega_0 t - y\left(t\right)\sin\omega_0 t. \end{aligned} \tag{3}$$

This representation of a signal lends itself well to graphical interpretation through a phasor diagram, shown in the top left pane of Fig. 2. We can thus treat the modulation



I Amplitude (V)	Q Amplitude (V)	I+Q	
		Amplitude (V)	Phase (°)
1	0	1	0
0	1	1	90
-1	0	1	180
0	-1	1	270

Figure 2: IQ signals, in both a phasor representation and as a time-amplitude plot. The plot shows that you can represent different modulation schemes by modulating amplitude of the component I and Q signals. Image credit: Vigneshdm1990 [16].

separately from the carrier signal by looking at the constituent I/Q parts and interpreting the modulation as a perturbation around the carrier phasor [14, p. 173-174, 206–207]. In a quadrature modulator, we can create both amplitude, phase and frequency (frequency is the time derivative of the instantaneous phase) modulation simply by amplitude modulating the individual I and Q signals [14, p. 209-210].

With this phasor representation in mind, we can represent different states of the I and Q modulations in a constellation diagram. An example of a constellation diagram for 16-QAM is shown in Fig. 3.

The *RadioML 2016.10A* data set contains 11 different modulation types, 8 digital and 3 analog, which are:

BPSK Binary phase-shift keying is the simplest form of phase shift keying (PSK), where the modulating signal shifts the phase of the carrier wave between two values 180° apart [14, p. 176].

QPSK Quadrature phase-shift keying is similar to BPSK, only now with four distinct phase values.

8PSK 8PSK denotes phase-shift keying with eight different phases.

16QAM 16QAM represents quadrature amplitude modulation (QAM) with sixteen distinct constellation points. QAM can be viewed as changing both the phase, as in PSK, and

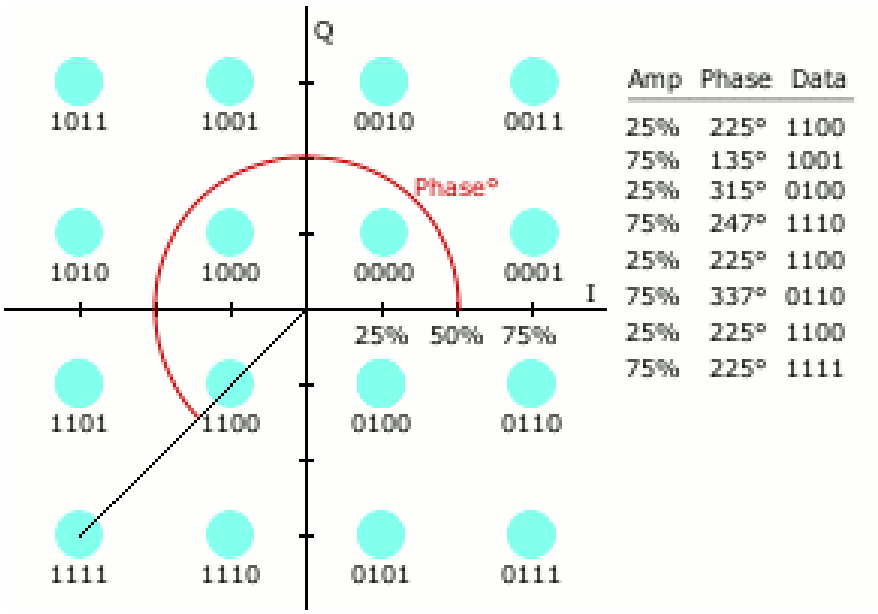


Figure 3: An example of a constellation diagram for 16-QAM modulation. Sixteen distinct constellation points can represent sixteen distinct waveforms, and therefore send four bits ($\log_2(16) = 4$) per waveform, or symbol, for digital data. Image credit: Chris Watts[17].

the amplitude as in amplitude shift keying (ASK) at the same time [14, p. 564]. This results in a constellation diagram as shown in Fig. 3.

64QAM 64QAM is QAM with sixty-four constellation points.

GFSK GFSK is gaussian frequency shift-keying. Frequency shift keying is a modulation scheme where each symbol, or waveform, changes the frequency of the carrier wave between different values. GFSK is then a special case of FSK where the data first has been pulse-shaped through the use of a gaussian filter, such that the transitions between the different frequencies corresponding to the different symbols is smoother. GFSK is used in technologies such as Bluetooth [18].

CPFSK CPFSK is known as continuous-phase frequency-shift keying is a special case of FSK with an added requirement that the phase should be continuous, meaning the change from one frequency to another is a smooth transition [14, p. 177]. GFSK and CPFSK are closely related, and might differ only in the type of filtering applied beforehand.

PAM4 PAM4 is four-level pulse-amplitude modulation. Pulse-amplitude modulation means that the amplitude of a pulse train carrier is modulated by the data you wish to send [14, p. 91].

WB-FM Wideband FM is the analog equivalent of FSK, and typically used for commercial radio systems. The term wideband refers to the maximum frequency deviation of the

modulated signal being much greater than the frequency of the *modulating* signal, resulting in a need for greater bandwidth [15, ch. 2.4.4-2.4.5].

AM-SSB Single-sideband amplitude modulation, an analog equivalent to amplitude shift keying. The carriers amplitude is modulated according to the data signal. In the case of AM modulation, the two information carrying sidebands around the carrier frequency are identical. Single-sideband transmissions is the process of transmitting only a single sideband. This does not lose any information, but puts slightly stricter requirements on the receiving side [15, ch. 2.4.1-2.4.3].

AM-DSB Double sideband amplitude modulation works just like AM-SSB, only now both sidebands are transmitted.

We show the I/Q samples from the *RadioML 2016.10A* data set, one of each modulation type, in Fig. 4.

2.2.2 Convolutional Neural Networks

A convolutional neural network (CNN) is a specialized type of neural network utilizing the operation of convolution in one or more layers. A CNN layer generally consists of a convolution operation, an activation function, and a pooling stage. Sometimes, these operations are viewed at three separate layers, sometimes as a single layer [19, p. 335-336].

There are three important ideas in CNNs called local receptive fields, shared weights, and pooling. The local receptive field tries to take advantage of spatial information, meaning it will consider potential relationships between data points in close proximity. We can think of the receptive field as how much of the input a node in a CNN sees at a time. How many steps across the input data this receptive field moves for the adjacent node in the convolutional layer is denoted by the stride length. The mapping, from the input nodes the receptive field sees, to the hidden nodes is called the feature map, and we move this mapping across the input nodes. The weights defining this feature map are then called shared weights, since they are the same for each mapping. Finally, a pooling stage (or layer) simplifies the output from the hidden convolution layer to the output. A common choice is *Max Pooling*, where the pooling stage outputs the maximum activation from a region in the previous hidden layer [20].

The basic convolution operation can be defined for the one-dimensional case as

$$s\left(t\right) = \left(x * w\right)\left(t\right) = \sum_{a=-\infty}^{\infty} x\left(a\right) w\left(t-a\right), \tag{4}$$

although this is not the exact implementation often used in practice in CNN applications due to techniques like zero-padding, strides and higher dimensional representations of the data [19, p. 342]. This type of network has proven to be very effective for two-dimensional grid-like data, perhaps most famously for image recognition [19, p. 326, 365–366].

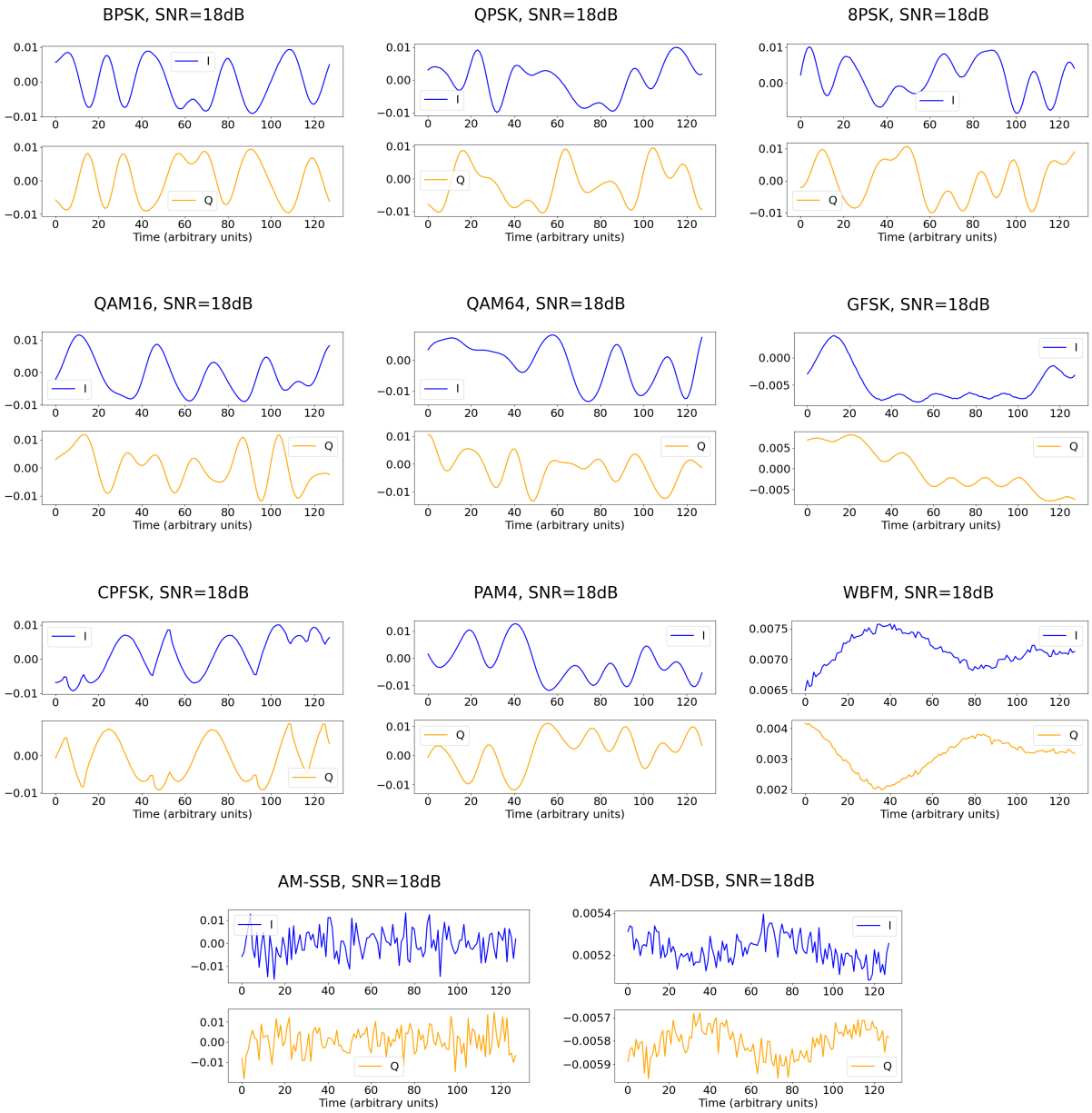


Figure 4: IQ samples with signal-to-noise ratio of 18 dB for each modulation type in the *RadioML 2016.10A* data set.

2.2.3 Recurrent Neural Networks

While CNNs have proven especially successful at handling two-dimensional grid-like data, a recurrent neural networks is a specialized network for handling one-dimensional sequential data. It is also adept at generalizing to input of variable length. A recurrent neural net's hidden nodes are defined by a recurrence relation, such as

$$\mathbf{h}^{(t)} = f\left(\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}; \theta\right), \tag{5}$$

where $\mathbf{h}^{(t)}$ represents the state at position t, and θ is shared parameter across positions [19, p. 367-370]. Recurrent neural networks suffer from the problem of vanishing or exploding gradients, due to the repeating nature of a shared weight. This can be illustrated by the simplified scalar case of a recurrence relation, where the state $h^{(t)}$ is given by

$$h^{(t)} = w^t h^{(0)}, \tag{6}$$

where w is the weight. For a weight below 1, this product would quickly vanish, while a weight above 1 would explode [19, p. 396-398]. One of the most effective types of RNNs equipped to handle the vanishing or exploding gradient problem are gated RNNs, of which the long short-term memory (LSTM) model belongs to. An LSTM consists of a forget gate, input gate and an output gate. These gates control the amount to propagate to the next cell and hidden states, C_{t+1} and h_{t+1} respectively, via a sigmoid function. The activation function for the input itself can be any squashing function [19, p. 404-407], although the hyperbolic tangent is a common choice. An illustration of the LSTM cell can be seen in Fig. 5.

2.2.4 Dropout regularization

Dropout can be thought of as a computationally inexpensive approximation to bagging. Dropout trains the subset of models resulting from removing a certain number of (hidden) nodes from the mother network [19, p. 255]. Dropout utilizes parameter sharing, reducing the memory load in comparison to bagging [19, p. 257].

A reasoning for the use of dropout, given by the winning submission paper of the ImageNet object recognition challenge, reads: "This technique reduces complex co-adaptations of neurons, since a neuron cannot rely on the presence of particular other neurons. It is, therefore, forced to learn more robust features that are useful in conjunction with many different random subsets of the other neurons" [22]. Dropout has proven itself empirically as a very successful method of regularization.

2.2.5 Loss, activation and gradient clipping

Since this is a multi-class classification problem, we make use of the (categorical) cross-entropy cost function in conjunction with a softmax activation layer in the output. The softmax function can represent the probability distribution over multiple classes, and can be formally written as

$$\operatorname{softmax}\left(\mathbf{z}_{\mathrm{i}}\right) = \frac{\exp \mathrm{z}_{\mathrm{i}}}{\sum_{\mathrm{j}} \exp \mathrm{z}_{\mathrm{j}}}, \tag{7}$$

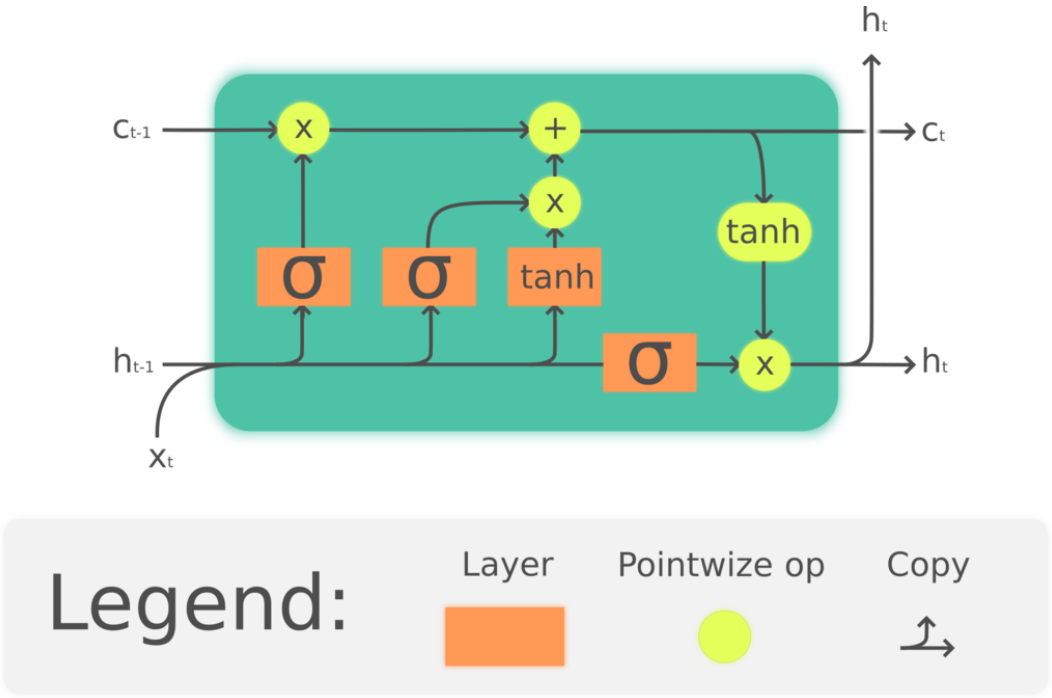


Figure 5: The structure of an LSTM node. The long term memory can be seen propagating as the cell state C, while the short term memory propagates as the hidden state h. Image credit: Guillaume Chevalier[21].

for a discrete variable $\mathbf{z}$ [19, p. 180-183]. In short, the softmax function turns a vector of unbounded (prediction) values into a probability distribution, with their relative ordering maintained.

The cross-entropy cost function is given by the expression

$$H\left(p,q\right)=-\sum_{\text{classes}}p\left(x\right)\log q\left(x\right), \tag{8}$$

where p and q are the probability distributions for the true and predicted labels respectively. In practice, this means that p is the one-hot vector for the true label and q is the output from softmax.

We also make use of the rectified linear unit (ReLU) activation function, which outputs the input value for positive input values, but returns the value 0 for negative inputs. This can be expressed as

$$f(x)=\begin{cases}x, & \text{if } x>0\\ 0, & \text{else}\end{cases}. \tag{9}$$

Inside the LSTM units, the hyperbolic tangent activation function is used on the input itself, while the sigmoid function is used elsewhere, as is standard.

Gradient clipping seeks to avoid taking too large steps following the gradient, to not overshoot the solution in a gradient descent scenario. One implementation of gradient clipping can be summarized in the pseudo code snippet,

```
if norm(gradient) > threshold:
    gradient = gradient * threshold / norm(gradient)
```

meaning we normalize the gradient to the threshold value if the gradient exceeds the threshold, while stile following the gradient direction, since the gradient is a vector and the threshold is a scalar [19, p. 409-411].

2.3 Our model architectures

We will consider five different CNN models (four different architectures) and two RNN models for classifying the modulation types of samples in the *RadioML 2016.10A* data set. The reason for considering different models and neural network types is to observe which models are most suitable for classifying a time series of IQ data. All the models are trained using a dropout rate of 50 % in all the hidden layers except the final fully connected layer giving the output. This choice is made to reduce overfitting of the models to the training data set.

All the models will be trained using the ADAM optimizer and categorical cross-entropy as cost function. For more information about ADAM, see [19, p. 305-306] or our previous project. To reduce problems associated with exploding gradients, we apply gradient clipping to the optimizers for the RNN models.

The total data set containing 220 000 samples of IQ time series of 128 time steps will be randomly shuffled, and then split into a training set (60 %), a validation set (20 %) and a test set (20 %). The training will be performed for 200 epochs with a batch size of 1024 samples. The final performance of the models will be measured by the classification accuracy achieved on the test set.

The simplest CNN model we consider is defined using three hidden layers in sequence. The input data, which is given by a sequence of 128 time steps with an in-phase and a quadrature value for each step, is sent through a one-dimensional convolution layer with 64 filters of size 3. The next layer is a two-dimensional convolution layer with 64 filters of dimensions 2×4, which combines the convolved sequences of the in-phase and quadrature data into a single sequence. The resulting sequence is finally converted to a modulation prediction using a fully connected layer with 11 nodes, corresponding to the number of modulation types in the data set.

The output from each of the convolution layers is activated using the ReLU activation function, while the fully connected layer is activated using softmax activation in order to obtain classification probabilities for the different modulation types.

The next CNN model architecture is used for two different models (different number of filters/nodes in the hidden layers). Similar to the previous model, these models have two convolution layers (a one- and a two-dimesional layer). The difference is that we add an additional fully connected layer with ReLU activation before the final fully connected layer.

By using 256 filters in the one-dimesional convolution layer, 80 filters of dimensions 2×3 in the two-dimensional convolution layer, and 256 nodes in the fully connected layer, we have recreated the model used by O'Shea et al. [2]. This model will be used to verify our implementation and as a benchmark for comparing the performance of our other models against.

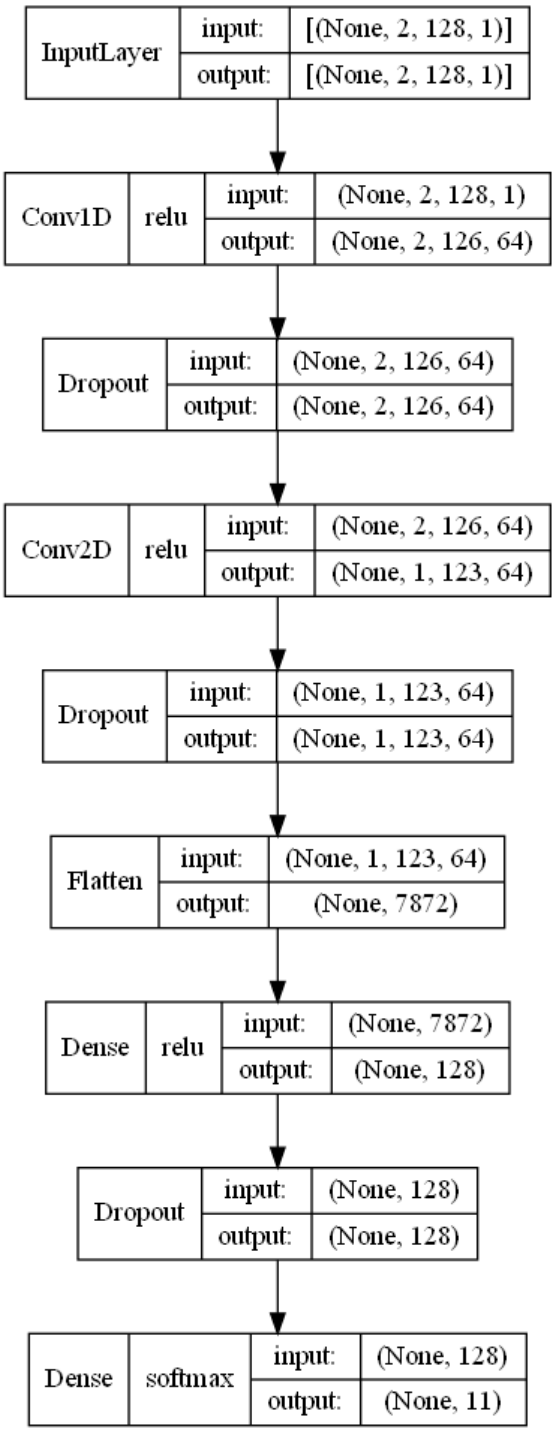


Figure 6: Architecture of a convolutional neural network with two convolution layers (with 1- and a 2-dimensional filters, respectively) used to classify the *RadioML 2016.10A* data set. The shape of inputs and outputs for the different layers, as well as the activation functions are shown in this illustration

We also define a different model from this architecture by using 64 filters in both of the convolution layers (filters of sizes 3 and 2×4 respectively) and 128 nodes in the fully connected layer. The structure of this model is shown in Fig. 6. Due to the decrease in the number of filters/nodes in the hidden layers, this model has less complexity then the model used by O'Shea et al. [2]. By comparing the performance of these two models, we can observe how the decreased complexity affects the classification accuracy.

In the previous models, we used filters of size 3 in the one-dimensional convolution layer, which is only able to extract small features from the input sequence. Our two final CNN models attempt to extract both small and larger features in the initial layer by replacing the one-dimensional layer in the previous models with three convolution layers with different filter sizes in parallel. This means that we send the input IQ sequence into all three convolution layers, and then concatenate the output from these layers, before sending it into the two-dimensional convolution layer. The difference between the two final models using parallel convolution layers is that one of the models has a single fully connected layer, while the other has two fully connected layers, similar to the previous models.

Both of the models use 64 filters in each of the parallel convolution layers, and the filter sizes of the filters in the different convolution layers are 3, 5 and 8. The two-dimensional convolution layer contains 80 filters of size 2×4. For the model with an additional fully connected layer, shown in Fig. 7, the fully connected layer has 256 nodes.

The two recurrent neural networks (RNNs) we consider are based upon using long short-term memory (LSTM) layers. The first model sends the input IQ sequence through an LSTM layer with 64 units using the tanh activation function, and then sends the final outputs of each unit into a fully connected layer with 128 nodes. The output from this layer is activated with the ReLU function before being sent into the final fully connected layer with softmax activation.

The second RNN model is similar to the first, except that an additional LSTM layer is added between the input layer and the LSTM layer, which sends the output from all steps of the LSTM units into the next LSTM. This LSTM layer also contains 64 units.

3 Results and discussion

The training of the neural networks that were described in the previous section was performed on the training set for 200 epochs, with a batch size of 1024. After each epoch, the model was applied to the validation data set, containing 44 000 samples, to give an estimate of the performance of the model. The performance estimates used in this report is based on the accuracy metric, denoting the fraction of samples where the model predicts the correct modulation type. The validation accuracies after each epoch of training for the different neural network models are shown in Fig. 8.

After completing the 200 epochs of training, the final accuracies of the models are measured by applying them to samples that have never seen by the models. This is achieved by using the 44 000 IQ samples in the test set. Considering that the samples in the *RadioML 2016.10A* data set are labeled with modulation type and signal-to-noise ratio (SNR), it is interesting to observe the dependency between SNR and classification accuracy for the models. A plot showing this dependency is presented in Fig. 9, where we observe that the models

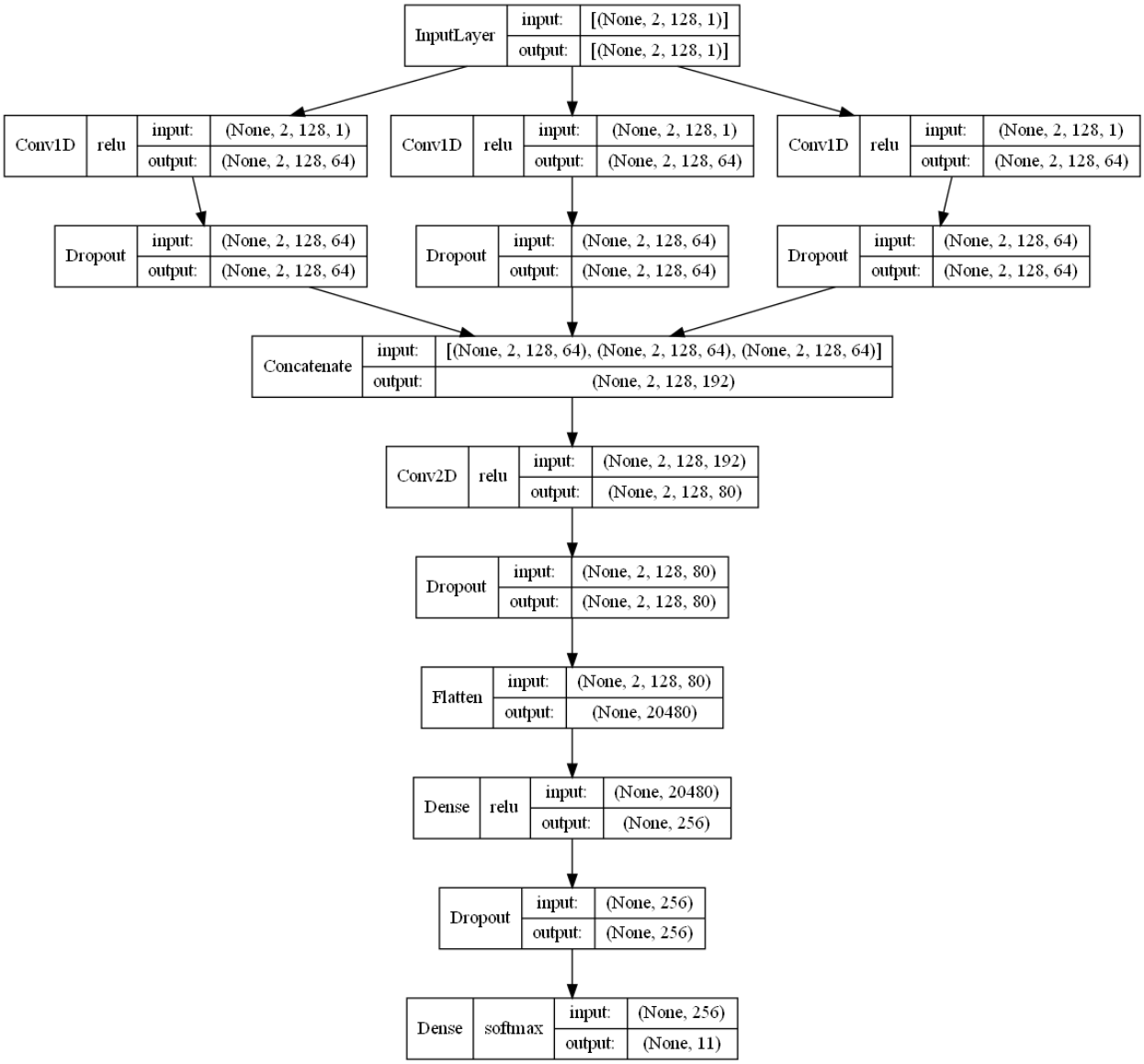


Figure 7: Architecture of a convolutional neural network based upon the model in Fig. 6, where the first convolution layer has been replaced by three convolution layers in parallel.

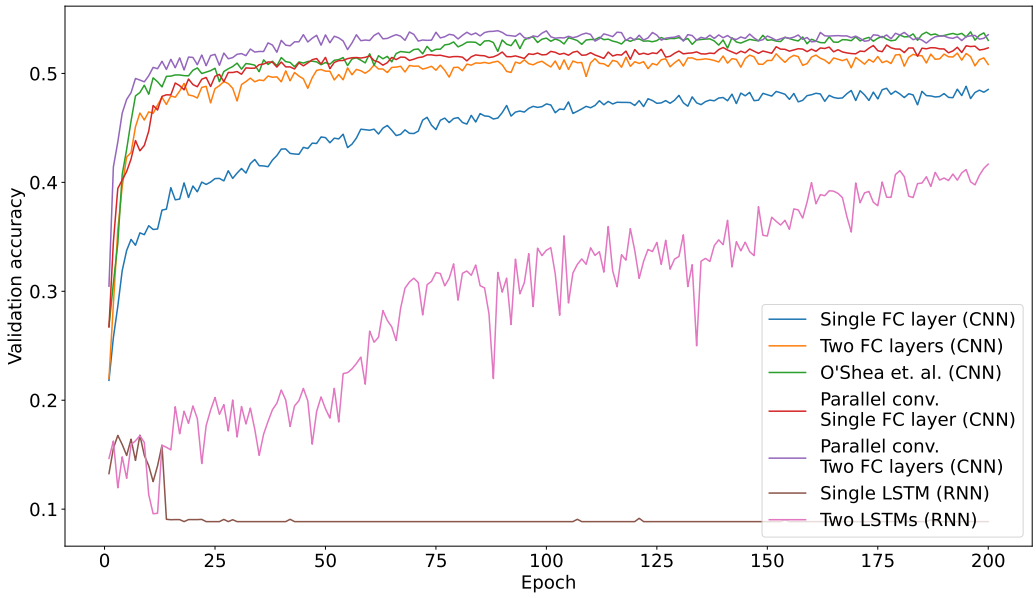


Figure 8: Classification accuracy for our 7 neural network models on the validation set after each epoch during training.

give low accuracies for the lowest SNRs, increasing accuracies for SNRs up to approximately 0 dB, before plateauing around the accuracies obtained at 0 dB. When the SNR becomes larger than 0 dB, the signals are easily discerned from the noise in the IQ samples. Thus, increasing the SNR gives little additional information to the classification models, which can explain the reason for the models reaching an accuracy plateau.

All of the models have accuracies around $1/11 \approx 0.091$ for the lowest SNRs in the data set, which is the accuracy expected by randomly guessing the labels. This is to be expected, because the signal drowns in the noise for such low SNRs, making the model unable to extract any features of the signal from an IQ sample.

By comparing the accuracy scores of the different trained neural network models in Fig. 9, we observe that the RNN models have the worst classification performance. The single layer LSTM model is unable to learn anything from the data, resulting in accuracies around $1/11 \approx 0.091$ for all SNRs. Meanwhile, the model with two LSTM layers is able to learn something about the data, but does not achieve higher than approximately 60 % accuracy for high SNR samples. By looking at the validation loss during training of this model in Fig. 8, we observe that this model learns significantly slower than the CNN models. This plot may indicate that the model could be able to achieve higher accuracy by increasing the training time (number of epochs).

The model that seems to give the best accuracy overall for the test set is the parallel convolution model with two fully connected layers, closely followed by the model proposed by O'Shea et al. [2]. These are also the two models with highest complexity, where both models has 256 nodes in the first fully connected layer. The difference between the models is that the

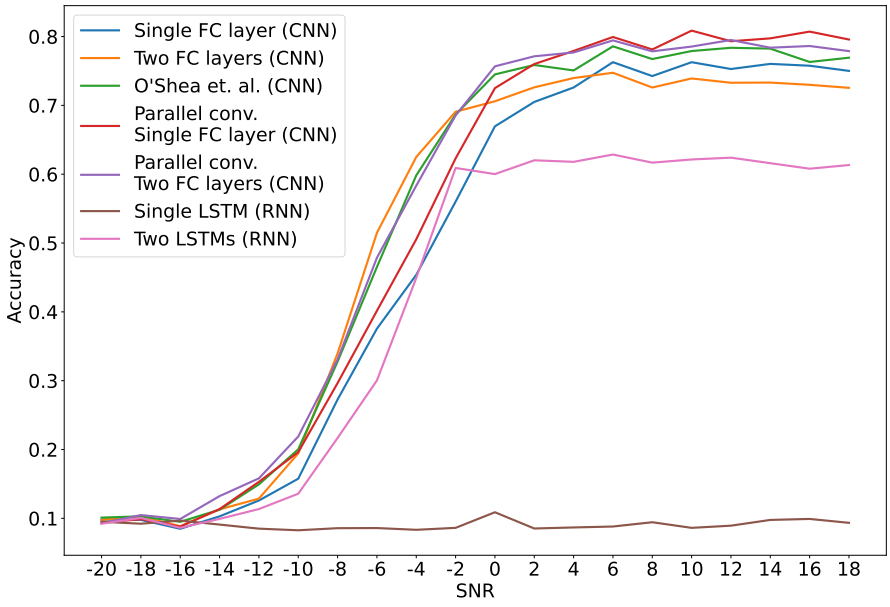


Figure 9: Classification accuracy for our 7 neural network models on samples in the test set at different signal-to-noise ratios (SNRs). The models have been trained during 200 epochs with a batch size of 1024.

parallel model has three convolution layers with 64 filters of sizes 3, 5 and 8, respectively, while the other model has a single convolution layer with 256 filters of size 3. This may indicate that the parallel model was able to learn more about the signal modulations by observing larger parts of the in-phase and quadrature sequences during the convolutions, despite having fewer filters in each of the convolution layers.

In addition to looking at how the accuracies of the models depend on the SNR of the samples, it is also natural to look at how accurate the models are for each of the individual classification labels. This can be presented in a structured manner through a confusion matrix, which shows the distribution of predicted labels against true labels. This gives the opportunity to observe which labels a model has problems with correctly classifying and which labels the model predicts instead in these situations.

For samples with negative SNR, meaning that the signal is weaker than the noise level, we saw a significant decrease of classification accuracies for the models in Fig. 9. For the lowest SNR ($-20\,\mathrm{dB}$), we observe close to similar confusion matrices for the various trained models. An example is shown for the convolutional neural network with parallel convolution layers and two fully connected layers in Fig. 10. In this figure, we observe that the model is biased towards classifying low SNR samples as AM-SSB, and in general seems unable to find distinguishing features of the different modulation types in these samples.

A possible explanation to why a large fraction of the low SNR samples were classified as AM-SSB is that signals with this modulation type look more similar to noise, compared to the other modulation types in the data set. This can be seen by comparing the IQ data

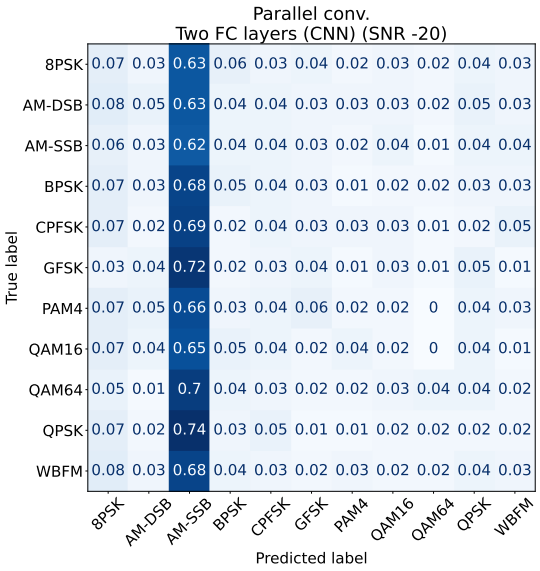


Figure 10: Confusion matrix for classifications of samples with −20 dB SNR using the parallel convolution model with two fully connected layers

of the high SNR samples shown in Fig. 4, where most of the samples have smooth curves, while the IQ data of the AM-SSB sample has a discontinuous nature. The two other analog modulation types, AM-DSB and WB-FM, can also be seen having a noise-like nature, but these sequences can be observed following underlying waves of shorter frequencies, making it easier to distinguish the sequences from random noise.

By considering the classification of samples with higher SNR values using the parallel convolution model with two fully connected layers, we can observe an increasing classification accuracy in Fig. 11. The confusion matrix for samples of −8 dB SNR shows that the classification accuracy is improving, but some of the modulation types still have a slight bias towards AM-SSB classifications. Meanwhile, when we consider the samples with the highest SNR values of 18 dB, the resulting confusion matrix shows high correlation between the predicted and true classification labels.

Since all the trained models seem to reach a plateau accuracy around 0 dB SNR in Fig. 9, we present the confusion matrices of our models on all the samples in the test set with non-negative SNR in Figs. 12, 13 and 14.

The confusion matrices of the convolution models in Fig. 12 has several similar features. We observe that all the models achieve above 90 % accuracy for BPSK, CPFSK, GFSK and PAM4 modulated samples. Meanwhile, all the models seem to struggle with differentiating between QAM16 and QAM64 modulated samples and between 8PSK and QPSK modulated samples, and WB-FM modulated samples are often misclassified as AM-DSB.

We note that we were unable to replicate the classification accuracy presented by O'Shea et al. despite using the model described in their article [2]. Their confusion matrix also shows a significant amount of misclassifications of WB-FM as AM-DSB, 8PSK as QPSK and QPSK as 8PSK, but unlike our results, they obtain high accuracy for the QAM16 and QAM64 modulations. This may be caused by better tuned hyperparameter values or aspects

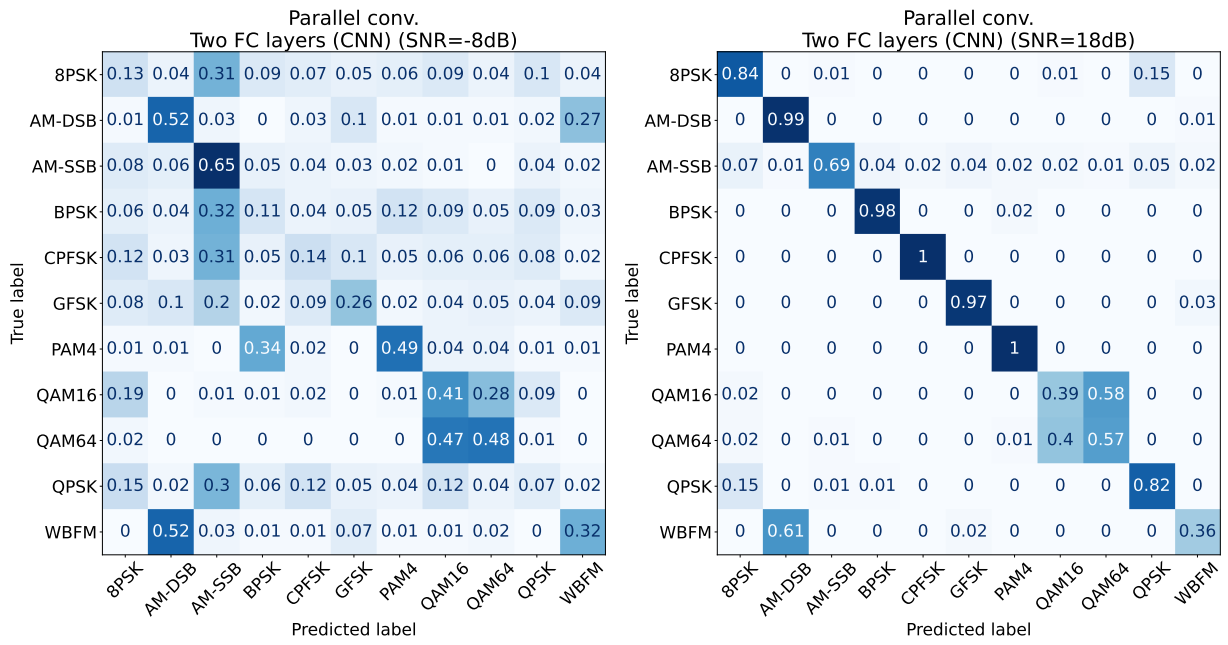


Figure 11: Two confusion matrices for classifications of samples with SNR values of -8 dB and 18 dB, respectively, using the parallel convolution model with two fully connected layers.

of the model that were not described in the article.

Explanations for the misclassifications of WB-FM, 8PSK and QPSK samples are given by O'Shea et al., presenting specific cases where the modulation types are indiscernible. For signals of the analog modulation types WB-FM and AM-DSB, there are periods of silence, where only a carrier tone is present, resulting in similar IQ data for the two types. For the digital modulations, the constellation points in QPSK are a subset of the points in 8PSK, causing an 8PSK sample to be indiscernible from QPSK if the sample only uses constellation points from the subset of QPSK.

The confusion matrix for the single LSTM model in Fig. 14 supports our previous claim that the model was unable to learn from the training data, as every sample is classified as QAM16 modulation. Meanwhile, the confusion matrix of the two-layer LSTM model obtains similar characteristics as the convolution models, where the model has high accuracy for classifying AM-DSB, AM-SSB, BPSK, CPFSK, GFSK and QPSK samples, while the classifications of 8PSK, WB-FM, QAM16 and QAM64 samples have low classification accuracies. Additionally, we observe a lower accuracy for PAM4 modulations.

From the results presented in this section, we found that the convolutional neural networks achieved higher accuracy and faster convergence than the recurrent neural network based on two LSTM layers. Considering that RNNs are created specifically to process temporal/sequential data, while CNNs are often used to look for spatial connections in data, such as in images, it is surprising to see that CNNs outperform the RNNs.

The slow training of RNNs is a known problem for these networks, and is associated with vanishing gradient problems when performing backpropagation, making the model unable to learn long-term dependencies in the data. Even though LSTMs are meant to reduce

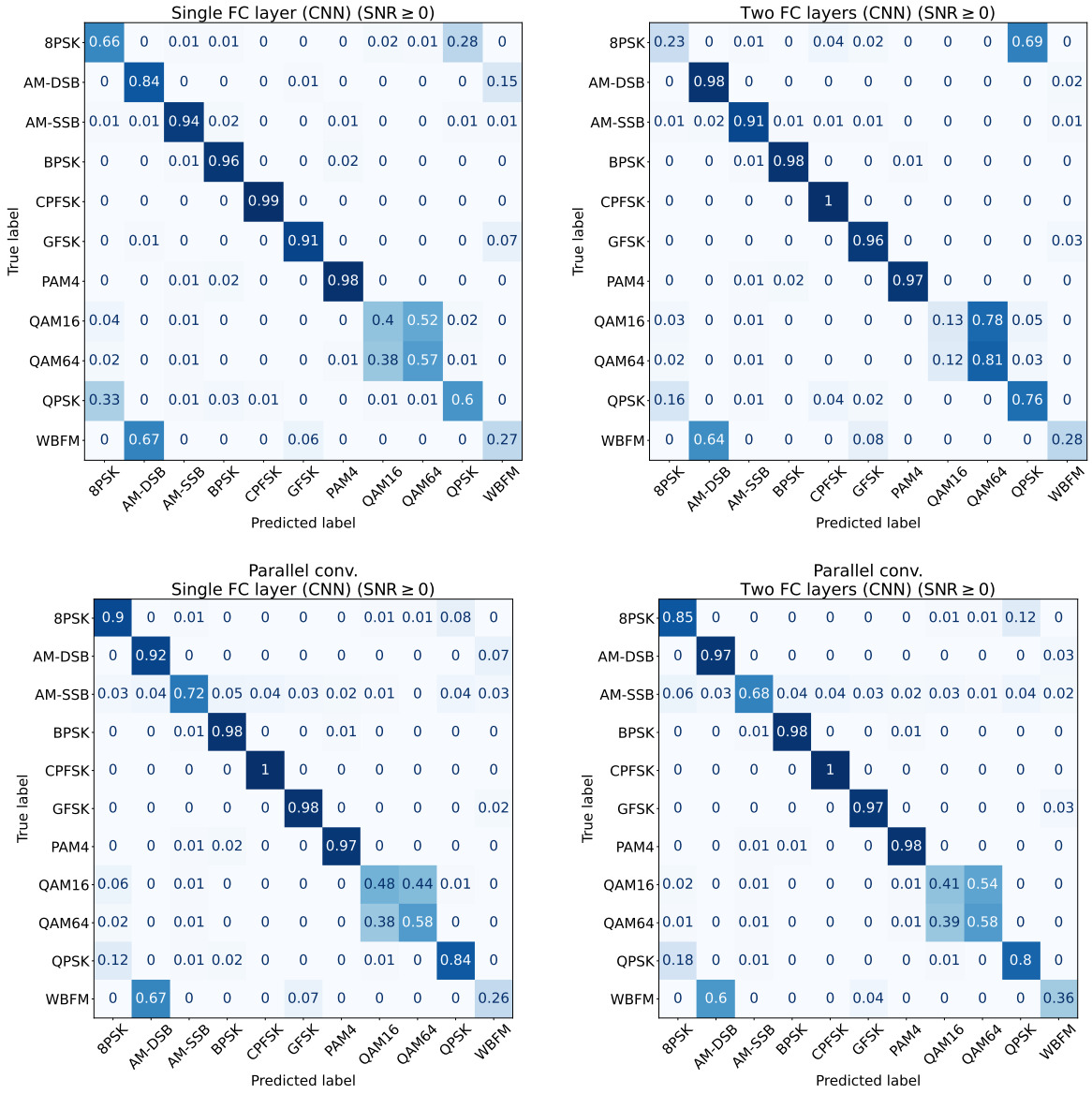


Figure 12: Confusion matrices for classifications of samples with non-negative SNR for our CNN models.

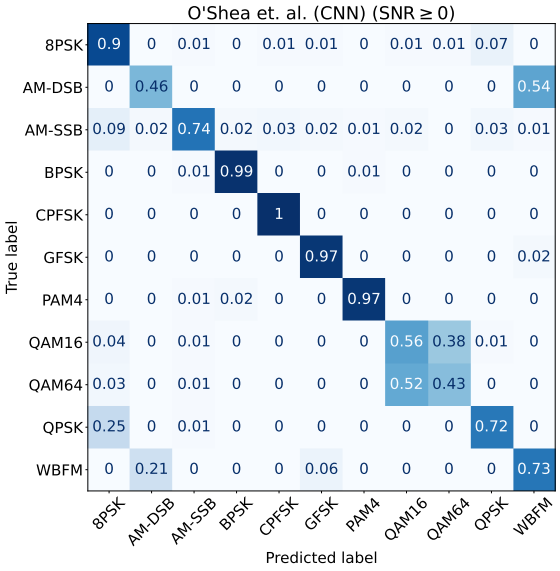


Figure 13: Confusion matrix for classifications of samples with non-negative SNR values using the CNN model proposed by O'Shea et al. [2].

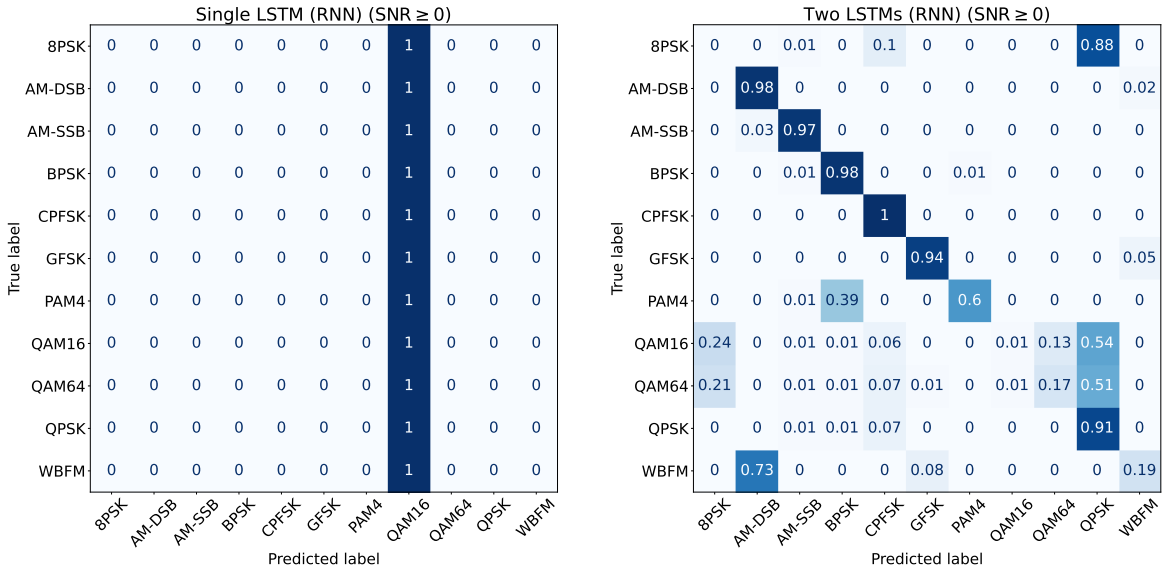


Figure 14: Confusion matrices for classifications of samples with non-negative SNR values using RNN models.

this issue, we still observe that the training time of CNNs is significantly faster. Given that the validation accuracy curve of the two-layer LSTM model in Fig. 9 has not reached convergence after 200 epochs, the model may have the ability to achieve higher test accuracy than the results presented for our trained model. And while an LSTM should be adept at capturing larger structure in the data due to propagating information from previous data points, it seems the context window for the CNN models in this scenario is sufficiently large to capture enough structure to still outperform the LSTM models. The input from the *RadioML 2016.10A* samples are also fixed length, while an RNN would be a more natural choice than a CNN with fully connected layers for handling time series of variable input length.

4 Conclusion

We have trained various convolutional and recurrent neural networks to classify the modulation type of a signal, given an IQ sample containing 128 time steps. The models were trained and tested on the *RadioML 2016.10A* data set, which contains 220 000 IQ samples, equally distributed across 11 modulations types. The convolutional neural network models gave better test accuracies than the recurrent networks. The model achieving the highest accuracy was created by three parallel one-dimensional convolution layers with different filter sizes, a two-dimensional convolution layer which combined the in-phase and quadrature sequences, and finally two fully connected layers. This model obtained a classification accuracy of 53 % across the entire test data set, and around 75 % accuracy on samples with positive signal-to-noise ratio. Some of the misclassified samples can be explained by the modulation of the signals being indiscernible from a different modulation type.

4.1 Future work

A continuation of this work could be to optimize the model parameters such as number and sizes of the convolution filters, number of nodes in the fully connected layers, choice of activation functions, and tuning hyper parameters such as learning rate and dropout rate. Additionally, it would be natural to consider other model architectures that might be better suited for classifying this type of data.

Further, it would be interesting to perform a similar analysis on other data sets. An additional data set was released by DeepSig, called *RadioML 2018.01A*. This data set is even larger, and contains 24 different validated modulation types, both digital and analog. The *RadioML 2018.01A* data set was generated and used for the paper by O'Shea et al. released in 2017 [23], and poses a more complex and challenging classification task. It is also considered a benchmark data set in the field, and is therefore also well referenced in the literature.

Another interesting open data set to consider is the *HisarMod* data set, available from IEEEDataPort [24], and was used in the work presented in the article by Tekbiyik et al. [25]. This data set contains 26 different modulation types, which are all subsets of five modulation families: analog, FSK, PAM, PSK, and QAM.

References

[1] DeepSig Inc. *RF Datasets For Machine Learning*. 2023. URL: `https://www.deepsig.ai/datasets` (visited on 22/11/2023).

[2] Timothy J. O'Shea, Johnathan Corgan and T. Charles Clancy. 'Convolutional Radio Modulation Recognition Networks'. In: *Engineering Applications of Neural Networks*. Ed. by Chrisina Jayne and Lazaros Iliadis. Cham: Springer International Publishing, 2016, pp. 213–226.

[3] Dehua Hong, Zilong Zhang and Xiaodong Xu. 'Automatic modulation classification using recurrent neural networks'. In: *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*. 2017, pp. 695–700.

[4] Sreeraj Rajendran et al. 'Deep Learning Models for Wireless Signal Classification With Distributed Low-Cost Spectrum Sensors'. In: *IEEE Transactions on Cognitive Communications and Networking* 4.3 (2018), pp. 433–445.

[5] Sharan Ramjee et al. *Fast Deep Learning for Automatic Modulation Classification*. 2019.

[6] Ya Tu et al. 'Complex-Valued Networks for Automatic Modulation Classification'. In: *IEEE Transactions on Vehicular Technology* 69.9 (2020), pp. 10085–10089.

[7] Jakob Krzyston, Rajib Bhattacharjea and Andrew Stark. *High-Capacity Complex Convolutional Neural Networks For I/Q Modulation Classification*. 2020.

[8] Lin Zhang et al. 'Intelligent Denoising-Aided Deep Learning Modulation Recognition With Cyclic Spectrum Features for Higher Accuracy'. In: *IEEE Transactions on Aerospace and Electronic Systems* 57.6 (2021), pp. 3749–3757.

[9] Changbo Hou et al. 'Automatic modulation classification using KELM with joint features of CNN and LBP'. In: *Physical Communication* 45 (2021), p. 101259.

[10] Jiansen Qiao et al. 'Blind Modulation Classification Under Uncertain Noise Conditions: A Multitask Learning Approach'. In: *IEEE Communications Letters* 26.5 (2022), pp. 1027–1031.

[11] Ziqi Ke and Haris Vikalo. 'Real-Time Radio Technology and Modulation Classification via an LSTM Auto-Encoder'. In: *IEEE Transactions on Wireless Communications* 21.1 (2022), pp. 370–382.

[12] Jia Han, Zhiyong Yu and Jian Yang. 'Multimodal attention-based deep learning for automatic modulation classification'. In: *Frontiers in Energy Research* 10 (2022).

[13] Timothy O'Shea and Nathan West. 'Radio Machine Learning Dataset Generation with GNU Radio'. In: *Proceedings of the GNU Radio Conference* 1.1 (2016).

[14] Bernard Sklar. *Digital communications : fundamentals and applications*. eng. 2nd ed. Harlow: Pearson, 2014.

[15] Michael Steer. *MICROWAVE AND RF DESIGN I - RADIO SYSTEMS*. Accessed: 2023-12-1. CC BY NC-4.0 https://creativecommons.org/licenses/by-nc/4.0/. North Carolina State University. Available online at https://eng.libretexts.org/Bookshelves/Electrical_Engineering/Electronics/Microwave_and_RF_Design_I_-_Radio_Systems_(Steer).

[16] Vigneshdm1990. *IQ phasor diagram*. Own work, CC BY-SA 4.0. Mar. 2022. URL: commons.wikimedia.org/w/index.php?curid=86332943.

[17] Chris Watts. *QAM16 Demonstration*. Own work, CC BY-SA 3.0. Mar. 2022. URL: commons.wikimedia.org/w/index.php?curid=15781908.

[18] M.C. Valenti, M. Robert and J.H. Reed. 'On the throughput of Bluetooth data transmissions'. In: *2002 IEEE Wireless Communications and Networking Conference Record. WCNC 2002 (Cat. No.02TH8609)*. Vol. 1. 2002, 119–123 vol.1.

[19] Ian Goodfellow, Yoshua Bengio and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[20] Micheal A. Nielsen. *Neural Networks And Deep Learning*. http://neuralnetworksanddeeplearning.com/chap6.html. Determination Press, 2015. Chap. 6.

[21] Guillaume Chevalier. *The LSTM cell*. CC BY 4.0 https://creativecommons.org/licenses/by/4.0, via Wikimedia Commons. 2018.

[22] Alex Krizhevsky, Ilya Sutskever and Geoffrey E Hinton. 'ImageNet Classification with Deep Convolutional Neural Networks'. In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira et al. Vol. 25. Curran Associates, Inc., 2012.

[23] Timothy James O'Shea, Tamoghna Roy and T. Charles Clancy. 'Over-the-Air Deep Learning Based Radio Signal Classification'. In: *IEEE Journal of Selected Topics in Signal Processing* 12.1 (2018), pp. 168–179.

[24] Kürşat Tekbıyık et al. *HisarMod: A new challenging modulated signals dataset*. 2019.

[25] Kürsat Tekbiyik et al. 'Robust and Fast Automatic Modulation Classification with CNN under Multipath Fading Channels'. In: *CoRR* abs/1911.04970 (2019).