Analysis of Open Set Deep Neural Network Variants towards Classification of Known and Unknown Signals

Srihari K. Kompella
Montgomery Blair High School
Silver Spring, MD, USA
srihari.kompella@gmail.com

Sastry Kompella
SPARLab Inc.
North Potomac, MD, USA
sastryk@vt.edu

***Abstract*—In environments crowded with electromagnetic activity, cognitive radios (CR) must have the ability to differentiate between different signals, so as to understand their origins if needed, and decide whether they are friendly or unfriendly signals. This is extremely important in the case of military and intelligence applications, but is beginning to become important in the commercial world as well, given the recent discussion around 5G and ORAN. Previous methods for signal classification were limited to identifying specific signal features in supervised settings. In this paper, we develop and compare three open and three closed-set signal classification models based on various Machine Learning architectures. This is achieved by training the models first with common analog and digital signal modulations, such as AM and QPSK. The purpose of an open set model is to classify known signals, as well as unknown signals whose information is not included in the training and validation sets. For the purpose of this project we are considering the following signal modulations: 8PSK, AM-DSB, BPSK, CPFSK, GFSK, PAM4, QAM16, QAM64, QPSK, WBFM and AM-SSB. Results show that open-set networks are able to perform very close to their closed-set counterparts while also being able to classify unknown signals that they have not been trained on.**

***Index Terms*—Open Set, Closed Set, Convolutional Neural Network, Residual Network, Long-Short Term Memory, LSTM Network**

I. INTRODUCTION

Cognitive Radios (CR) [1] use dynamic spectrum access (DSA) [2] techniques to increase their efficiency in crowded environments, by switching to idle channels when their current operating channel is interfered with, or if the primary user of the spectrum channel switches from idle to active. While early DSA applications used simple energy detection to decide whether to move to a different channel, current and future DSA systems need to be more intelligent and actually identify the signal that they are presented with before deciding on a course of action. While this is clearly important in the case of military and intelligence applications, it is beginning to become fairly important for commercial and consumer-based communications, given the recent confluence of 5G and Open-Radio Access Network (O-RAN). For example, cognitive radios that can differentiate between signal waveforms/modulations based on classification of the transmitted signals, can be more effective in maneuvering through the spectrum white-spaces, thereby increasing spectral efficiency, than those that can only perform simple energy detection. However, in crowded environments without a central coordinator, classifying known signals is not enough and the CRs may need to understand even unknown sources of interference before they decide on a course of action.

In previous works, many signal classification models were feature-based classifiers, which are usually limited to using bandwidth, center frequency, synchronization signals, or cyclostationary signal properties for classification. Additionally, most of them use standard supervised learning algorithms. The features that they classify are specifically defined by people, so they may not always produce the highest accuracy. While it is convenient for humans to distinguish signal modulations based on self-defined characteristics, it is often not close to optimal.

Machine learning (ML) based RF signal modulation classifiers are a relatively new technology. Unlike feature-based classifiers, ML classifiers are feature-learning, that is they can determine the best features of the data to train on. Recent studies of ML based classifiers are mostly closed-set, i,e., they only classify within the training dataset. A recent example of closed set classifier for signals can be found in [3]. Also, the authors in [4] created many closed-set CNN variants, such as Residual Networks (ResNet) and LSTM networks. One drawback associated with these classifiers is that they are unable to classify outside of the training dataset. That is, if an unknown signal is provided, they can only classify that into one of the trained classes, so the closed-set classifiers do not have the ability to actually classify an unknown signal as not belonging to any of the trained classes. In the real world, it is often possible that the classifier may encounter signals that it was not trained on. The solution to this problem is to create open-set models that can classify outside of the training dataset.

This paper explores three such open-set versions of existing closed-set frameworks, namely, CNN, LSTM network, and ResNet. All three of these networks have been compared to the closed-set version of themselves and the differences analyzed. The rest of the paper is organized as follows: In

Section II, we describe the datasets that we used to perform our analysis. Section III provides details on open set classification and Section IV describes the model architecture. Section V describes the simulation environment and presents accuracy results for the closed as well as the corresponding open set models. Finally, in Section VI, we provide concluding remarks.

II. Dataset

In this paper, we used 2016 RADIOML [5], which is a signal dataset that is provided by DeepSig, for conducting our analysis. The dataset contains 1320000 total signals with 11 modulations, 3 of which are analog and the remaining 8 are digitally modulated signals types. There are 120000 signals per class. The modulations included are 8-Phase Shift Keying (8PSK), Double-Sideband Amplitude Modulation (AM-DSB), Single-Sideband Amplitude Modulation (AM-SSB), Binary Phase Shift Keying (BPSK), Continuous Phase Frequency Shift Keying (CPFSK), Gaussian Frequency Shift-Keying (GFSK), Pulse Amplitude Modulation 4-level (PAM4), 16 Quadrature Amplitude Modulation (QAM16), 64 Quadrature Amplitude Modulation (QAM64), Offset Quadrature Phase Shift Keying (QPSK), Wide Band Frequency Modulation (WBFM).

Every signal in the dataset underwent multipath fading, I/Q imbalances, frequency offsets, and AWGN impairments to mimic real-life signals. This will improve the robustness of the models during the training phase. Specifically, I/Q imbalances included randomly scaling the signal from -3 to 3 dB. The frequency offset is randomly chosen from -2.5 KHz to +2.5 KHz. The SNR values of the signals in the dataset range from -20 to 18 dB in increments of 2 dB. The signals are also transformed through the Short-Time Fourier Transform (STFT).

The closed-set models were trained on every modulation except for AM-SSB. This is because, in the case of open-set models, we chose the unknown signal to be AM-SSB. While we chose only one unknown signal in this paper, the methodology developed can be easily extended to address multiple unknown signals as well.

III. Open Set Classification

Open-set classification is an ML-based classification of not only known data but unknown data as well. Here, data that the model has been trained on is defined as “known” data. Data the model has not been trained on is defined as “unknown” data. When the open-set model’s prediction accuracy is tested, the model is exposed to both known and unknown data.

There are two approaches to open-set classification: generative and discriminative classification. Generative open-set classification is the process of identifying all unknown signals as unknown signals. If the model is trained on N classes, or in this case, modulations, any signal the model identifies as not pertaining to any of the N classes will be treated as an unknown signal. Discriminative open-set classification goes beyond generative classification in that it also discriminates between the unknown signals. Discriminative open-set classification methods are far more useful than Generative ones as discriminative classification would provide more information regarding the unknown signals it classifies, rather than lumping all unknown signals into one class.

The open-set classification method that we have employed in the open-set models explored in this paper is called OpenMax, which uses an alternative activation function to the commonly used SoftMax function. OpenMax [6], created by Bendale and Boult, acts as an unknown class detector for unknown data. Previously, it has been employed in other works such as for classifying open-set images [7] and adversarially perturbed images [8]. OpenMax, being a discriminative open-set classification method, allows for the open-set model to automatically classify unknown signals into multiple classes as appropriate.

A. OpenMax

The OpenMax function is based on the SoftMax function (see Equation 1) which, in a closed set network, classifies each data point as the class the network most confidently assesses it as. For every class j, the confidence that one data point belongs to a certain class is determined as a value between 0 and 1 and the sum of the confidence scores add up to 1. Let $\mathbf{v(x)} = v_1(x), ..., v_N(x)$ be the activation vector for sample $\mathbf{x}$ across classes $y = 1, ..., N$. The numerator, $e^{v_j}x$, is the activation level for class j. The denominator, $\sum_{i=1}^{N} e^{v_i x}$, is the sum of the vectors of all of the classes to ensure that the probabilities of sample x belonging to any one of the classes add up to 1. The SoftMax function is shown below:

$$P(y = j|x) = \left(\frac{e^{v_j x}}{\sum_{i=1}^{N} e^{v_i x}} \right) \tag{1}$$

Since the SoftMax function relies on the premise that the probabilities of the sample belonging to each of the output classes add to 1, it can not be used in an open-set scenario, where the model is exposed to unknown classes as well. The OpenMax function extends the SoftMax function by enabling it to predict unknown classes. The final classification is made by OpenMax by analyzing the penultimate layer in the neural network and estimating if and how far the network is from classifying the data point confidently. The OpenMax function utilizes two main algorithms. The first is Algorithm 1, which first computes the mean activation vector (μ_i) for each class. The mean activation vector is the mean of the activation vectors of correctly computed training data. Earlier works on core meta-recognition, using Extreme Value Theory (EVT) [9] found that the distribution of the final model scores followed the Weibull distribution. As a result, the algorithm uses the FitHigh function from the libMR library to do Weibull fitting on the largest distances between all correctly classified data and the mean activation vector associated with their class. The resulting parameter, ρ_j, is used to estimate the probability of an input not belonging to class j. The function returns the mean activation vector μ_i, and the Weibull parameter ρ_j.

Algorithm 1 EVT Meta-Recognition Calibration for Open Set Deep Networks, with per class Weibull fit to η largest distance to mean activation vector. Returns libMR models ρj which includes parameters τi for shifting the data as well as the Weibull shape and scale parameters:κi, λi.

```
Require: FitHigh function from libMR
Require: Activation levels in the penultimate network layer
    v(x) = $v_1(x)...v_N(x)$
Require: For each class $j$ let $S_{i,j} = v_j(x_{i,j})$ for each
    correctly classified training example $x_{i,j}$
1: for $j = 1...N$ do
2:     Compute mean AV, $\mu_j = \text{mean}_i(S_{i,j})$
3:     EVT Fit $\rho_j = (\tau_j, \kappa_j, \lambda_j) = FitHigh(||\hat{S}_j$ - $\mu_j||, \eta)$
4: end for
5:     Return means $\mu_j$ and libMR models $\rho_j$
```

The second algorithm that is used by OpenMax predicts whether the data point belongs to any of those classes (see Algorithm 2). First, the algorithm defines the unknown class to be at index 0. In line 3, the algorithm uses the Weibull CDF probability on input sample x and μ_j for the core of the rejection estimation. The model then generates weights for the α largest activation classes and uses them to scale the Weibull CDF probability. The algorithm then computes a revised activation vector for the input sample and computes a pseudo-activation for the unknown class in order to make sure the activation level of each class is the same. The algorithm then computes the probabilities of the input belonging to all of the classes, including the unknown class. The algorithm then rejects unknown and uncertain inputs.

Algorithm 2 OpenMax probability estimation with rejection of unknown or uncertain inputs

```
Require: Activation vector for $\mathbf{v(x)} = v_1(x)...v_N(x)$
Require: means $\mu_j$ and libMR models $\rho_j$
Require: $\alpha$, the number of "top" classes to revise
1: Let $s(i) = argsort(v_j(x))$; Let $\omega_j = 1$
2: for $i = 1...\alpha$ do
       $$\omega_{s(i)} = 1 = (\alpha - 1)/(\alpha)e^{-((||x - \tau_{s(i)}||)/(\lambda_{s(i)}))^{\kappa_{s(i)}}} \quad (2)$$
3: end for
4:     Revise activation vector $\hat{v}(x) = \mathbf{v(x)} \circ \omega\mathbf{(x)}$
5:     Define $\hat{v}_0(x) = \sum_i v_i(x)(1 - \omega_i(x))$
       $$\hat{P}(y = j|\mathbf{x}) = \frac{e^{v_j(x)}}{\sum_{i=0}^{N} e^{v_j(x)}} \quad (3)$$
6:     Let $y^* = argmax_j P(y = j|\mathbf{x})$
7:     Reject input if $y^* == 0$ or $P(y = y^*|\mathbf{x}) < \epsilon$
```

When implementing the OpenMax function in code, the network is first trained as a closed-set network without the unknown class. After the network is trained, the function calculates the mean activation vector for each class and the distances between each sample and the mean activation vector alongside the weibull CDF probability for each class. These three calculations are then used to calculate the OpenMax probabilities.

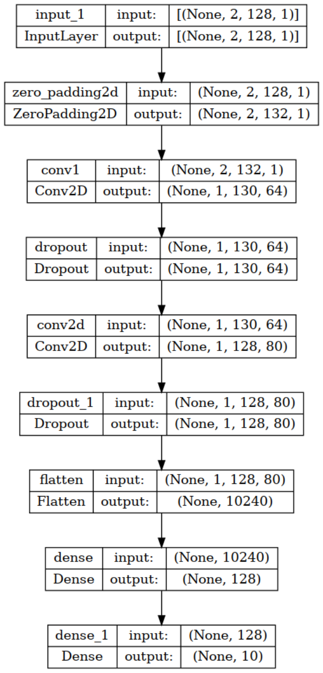

Fig. 1. CNN Architecture

IV. Model Architecture

In this paper, we compared the performance of three different classifier models, namely, 2D CNN, a 2D LSTM network, and a 1D ResNet model. These three architectures were chosen because of the specific differences between their classification methods. The CNN is built to classify data based on local and global patterns within the sample. While the LSTM network also classifies data based on local and global patterns within the sample, it also takes into consideration the long-term dependencies between different samples as well. The ResNet uses the skip connection to avoid the gradient-vanishing problem. Differences in performance can be attributed to these specific structural differences. We first trained and tested the closed-set models, then implemented the OpenMax function to generate openmax probabilities for a new set of test data. All models were built using Python 3.9 with TensorFlow and Keras. The OpenMax function was used at the end of the open-set functions after the models had been trained.

The rest of this section describes each of the classifiers in more detail.

A. CNN

Convolutional Neural Network (CNN) models are traditionally used to classify high dimension data, such as images, or in this case, signals. Figure 1 shows the sequence of layers and each layer's input and output shapes within the network. The model uses two convolutional layers with filter shapes of 64 and 80 and kernel sizes (2, 3) and (1, 3) resepctively, in order

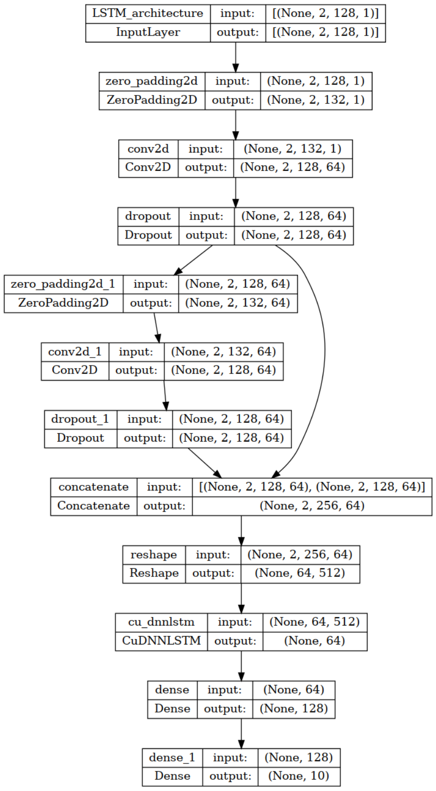
LSTM_architecture | input: | [(None, 2, 128, 1)]
InputLayer | output: | [(None, 2, 128, 1)]
zero_padding2d | input: | (None, 2, 128, 1)
ZeroPadding2D | output: | (None, 2, 132, 1)
conv2d | input: | (None, 2, 132, 1)
Conv2D | output: | (None, 2, 128, 64)
dropout | input: | (None, 2, 128, 64)
Dropout | output: | (None, 2, 128, 64)
zero_padding2d_1 | input: | (None, 2, 128, 64)
ZeroPadding2D | output: | (None, 2, 132, 64)
conv2d_1 | input: | (None, 2, 132, 64)
Conv2D | output: | (None, 2, 128, 64)
dropout_1 | input: | (None, 2, 128, 64)
Dropout | output: | (None, 2, 128, 64)
concatenate | input: | [(None, 2, 128, 64), (None, 2, 128, 64)]
Concatenate | output: | (None, 2, 256, 64)
reshape | input: | (None, 2, 256, 64)
Reshape | output: | (None, 64, 512)
cu_dnnlstm | input: | (None, 64, 512)
CuDNNLSTM | output: | (None, 64)
dense | input: | (None, 64)
Dense | output: | (None, 128)
dense_1 | input: | (None, 128)
Dense | output: | (None, 10)

Fig. 2. LSTM Network Architecture

to train on local data patterns. The model also uses dropout layers with dropout rate 0.5 to mitigate possible sources of overfitting. Addtionally, the model uses dense layers to train on global data patterns. Eventually, the model uses the SoftMax activation function to classify each data point.

B. LSTM Network

LSTM networks, also referred to as recurrent networks, differ from CNNs as they are able to analyze patterns in entire sequences of data through the use of feedback loops. In addition to this, LSTM networks also analyze local and global patterns, just like CNNs. The LSTM network used in this paper is also a 2D network, as shown in Figure 2, which describes the sequence of layers and each layer's input and output shapes within the network. The model also uses two convolutional layers just as in the case of CNNs described earlier. However, the filter shapes used are 64 and 64 and kernel shapes that are used are (1,5) and (1,5). The model also used dropout layers with dropout rate 0.5 to train on patterns restricted to individual images and then used a CuDNNLSTM layer with filter shape 64 to train on long term sequential patterns. The model then used Dense layers and the SoftMax activation function at the end.

C. ResNet

ResNets are specifically designed to combat the vanishing gradient problem, which impacts all high-density convolutional neural networks. As the network begins to search

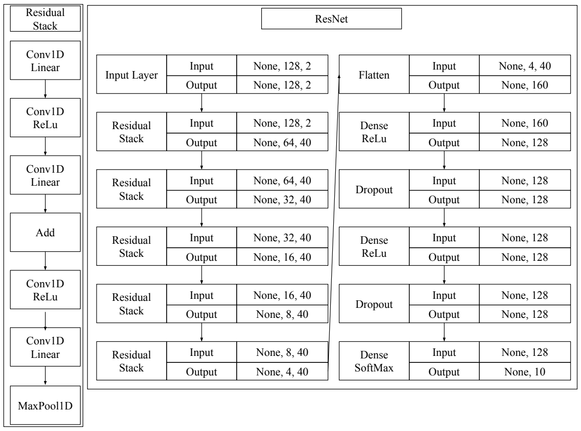
Residual Stack
Conv1D Linear
Conv1D ReLu
Conv1D Linear
Add
Conv1D ReLu
Conv1D Linear
MaxPool1D
ResNet
Input Layer | Input | None, 128, 2
Output | None, 128, 2
Residual Stack | Input | None, 128, 2
Output | None, 64, 40
Residual Stack | Input | None, 64, 40
Output | None, 32, 40
Residual Stack | Input | None, 32, 40
Output | None, 16, 40
Residual Stack | Input | None, 16, 40
Output | None, 8, 40
Residual Stack | Input | None, 8, 40
Output | None, 4, 40
Flatten | Input | None, 4, 40
Output | None, 160
Dense ReLu | Input | None, 160
Output | None, 128
Dropout | Input | None, 128
Output | None, 128
Dense ReLu | Input | None, 128
Output | None, 128
Dropout | Input | None, 128
Output | None, 128
Dense SoftMax | Input | None, 128
Output | None, 10

Fig. 3. ResNet Architecture

for local patterns with the use of convolutional layers, the gradients that the cost function produces sometimes become minute. At times, the gradient is so small that the CNN is unable to improve classification accuracy. ResNets combat this by relying on skip connections. ResNets consist of multiple residual stacks, or layers of a small number of convolutional layers. Before the convolutional layers in the residual stack are employed, a copy of the input tensor of the residual stack is put aside and is added to the tensor after the convolutional layers are employed. Even if the convolutional layers within the residual stack result in a vanishing gradient, the addition of the input tensor ensures that the gradient never stays small. In this paper, we use a 1D ResNet network. Figure 3 shows the sequence of layers and each layer's input and output shapes within the network. The residual stacks only consisted of convolutional layers with filter shape 40 and kernel size 3. Just as in the earlier cases, the model uses dense layers and dropout layers and the SoftMax activation function at the end.

V. Results

Each model was trained for 10 epochs using the Categorical Crossentropy loss function and Adam optimizer. All of the models were trained on a computer with an NVIDIA 2060 GPU with 8 CPU cores and 16 GB of RAM.

A. Closed-Set CNN

Figure 4 shows the average performance of the closed-set CNN across all SNRs provided in the dataset. The model's average accuracy ranges from 10% at -20 dB SNR to 82% at 18 dB SNR. The closed-set CNN, similar to the other closed-set models, classifies more accurately at high SNRs, and plateaus from 0 to 18 dB with accuracies ranging from 78% to 83%. At low SNRs, the model classifies far less accurately because the noise affects signal recognizability far more at low SNRs.

Figure 5 shows the confusion matrix of the closed-set CNN, which provides the map of predicted labels to true labels for each modulation. In Figure 5 it can be noticed that the CNN tends to classify the GFSK, PAM4, and CPFSK modulations

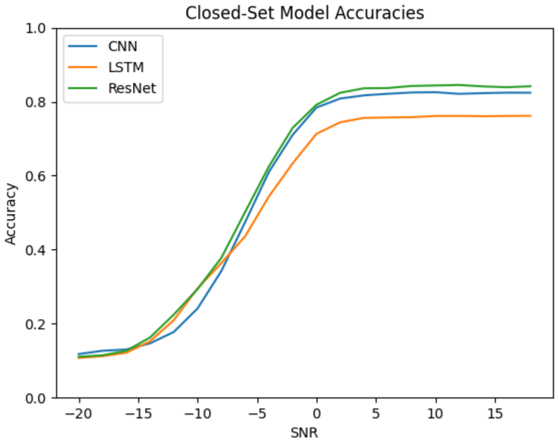

Fig. 4. Closed-Set Accuracy

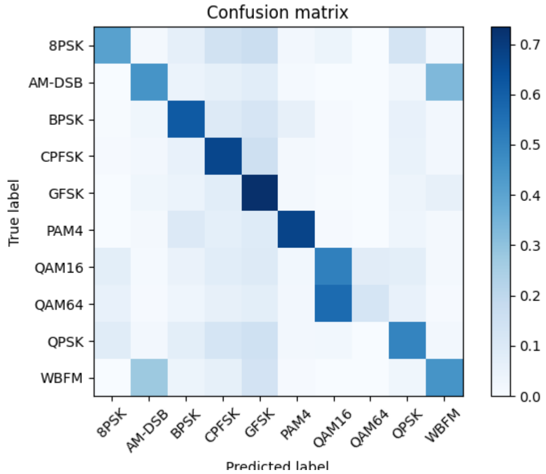

Fig. 5. CNN Confusion Matrix

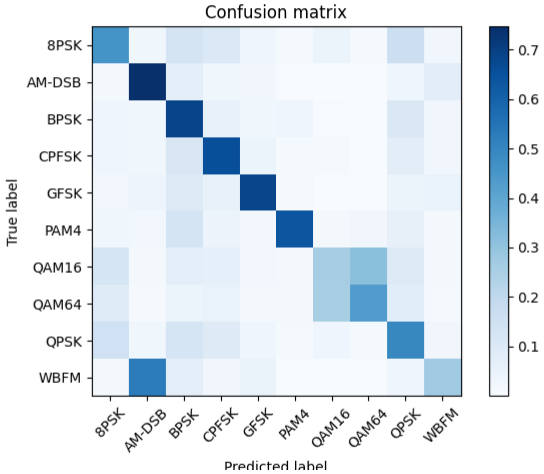

Fig. 6. LSTM Confusion Matrix

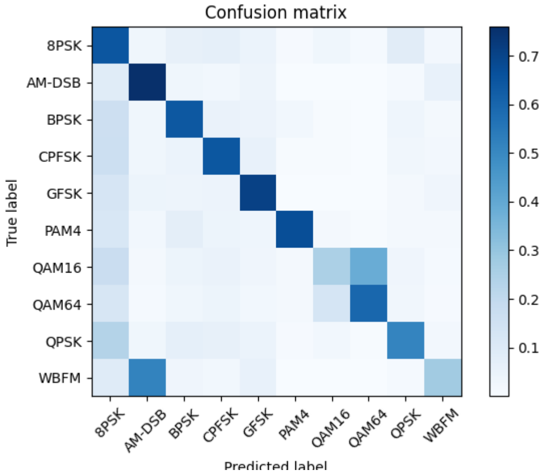

Fig. 7. ResNet Confusion Matrix

with relatively high accuracy. On the other hand, the model does not tend to do a good job of classifying the QAM64 modulation, and often mistakes it for QAM16. This most likely happens due to the similarities between symbol constellation in QAM16 and QAM64 modulations.

B. Closed-Set LSTM Network

In addition to closed-set CNN, Figure 4 also shows the average performance of the closed-set LSTM network across all tested SNRs. The model's average accuracy ranges from 13% at -20 dB SNR to 75% at 18 dB SNR. The closed-set LSTM network does not classify as well as the other closed-set networks at high SNRs. It can be seen in Figure 4 that the accuracy trajectory of the closed-set network visibly departs from those of the CNN and ResNet starting from -6 dB SNR.

The confusion matrix of the closed-set LSTM in Figure 6 shows that the LSTM network classifies the AM-DSB, GFSK, and BPSK modulations with relatively high accuracy. On the other hand, the model tends to do a poor job of classifying WBFM, QAM16 and QAM64 modulations. The LSTM network seems to confuse the QAM16 and QAM64 modulations for each other. This is different from the CNN case, which does not misclassify the QAM16 modulation. The LSTM network also tends to misclassify the WBFM modulation as AM-DSB.

C. Closed-Set ResNet

Finally, Figure 4 also shows the average performace of the closed-set ResNet across all tested SNRs. The model's average accuracy ranges from 12% at -20 dB SNR to 84% at 18 dB SNR. The ResNet performs the best out of all three of the networks at high SNRs.

Figure 7 shows the confusion matrix for the ResNet. The ResNet classifies the AM-DSB, GFSK and PAM4 modulations with relatively high accuracy and the QAM16 and WBFM modulations with relatively low accuracy. Whereas the CNN tends to misclassify the QAM64 modulation for QAM16, ResNet does the opposite. Additionally, similar to the LSTM network, ResNet also mistakes the WBFM modulation for AM-DSB.

D. Open-Set Networks

The open-set networks are trained on the same modulations as the closed-set networks, but are designed to classify one unknown class, that being AM-SSB.

Figure 8 shows the classification accuracy of open-set CNN alongside the open-set versions of the other models. Again, just as in the case of closed-set model, the open-set CNN classifies relatively accurately at high SNRs and poorly at low SNRs, with accuracies ranging from 5% at -20 dB SNR to 75% at 18 dB SNR. At every SNR, the performance of the open-set CNN is slightly less than its closed-set counterpart. When looking at the prediction scores of individual signals, we see that the open-set networks tend to be less confident than the closed-set networks, even for high SNRs. This lack of confidence is a byproduct of the presence of the unknown class.

The open-set LSTM network tends to do a relatively better job classifying signals at low SNRs as it was able to predict signals from -20 to -4 dB SNR with accuracies around 19%. While this may not be substantially much higher than in the other two open-set cases, it is relatively higher than the the prediction accuracies of the other open-set and closed-set networks, and the user might find this potentially beneficial for his application. This may be due to the LSTM network's ability to search for long term dependencies with the CuDNNLSTM layer, which was described in Section IV-B. However, at high SNRs, the LSTM network was unable to classify nearly as well as the ResNet or CNN, plateauing at around 70% between 8 to 18 dB SNR. This is less accurate than the closed-set LSTM network, which had an accuracy around 76% for 0 to 18 dB SNR. The LSTM network, when tasked with classifying the unknown class, does not classify as well as the CNN and ResNet. It classified approximately as well as the CNN and ResNet from -20 to -12 dB, but was unable to classify as accurate as the other models from 2 to 18 dB SNR.

When comparing open-set classifiers, the open-set ResNet does better than the open-set LSTM network and CNN at high SNRs with accuracy around 80 %. This most likely is the reslt of ResNet being complex and having far more convolutional layers than the other networks. However, this could also be the reason for its relatively poor performance at low SNRs. When classifying the unknown data, the ResNet was able to perform better than the other networks from 0 to 18 dB SNR.

Finally, Figure 8 also shows the unknown class classification accuracy of the CNN and other models. We see that all three classifiers do a good job of classifying unknown signals at high SNRs. Additionally, the CNN seems to perform better than the others at low SNRs as well, starting around -12 dB. For SNR below -12 dB the accuracy of all three classifiers seem to be around 60%.

VI. Conclusions

In this paper, we compared three different open-set wireless standard classifiers. We not only analyzed their strengths and weaknesses, but also compared them against their corresponding closed-set models. Results show that at high SNRs, the open-set ResNet outperformed the CNN and the LSTM network while the LSTM network outperformed the CNN and ResNet at low SNRs. This may be because for high SNRs, models with more complex structures tend to classify better. Models with structures that tend to analyze long-term patterns tend to classify better at low SNRs.

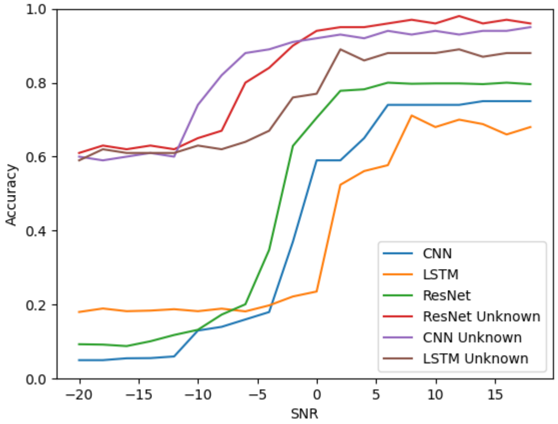

Fig. 8. Open-Set Accuracy

We have also observed that open-set networks generally classify less accurately than closed-set networks because they rely on the preexisting closed-set model to further predict whether the sample belongs to a known or unknown class. However, because the open-set models were able to classify unknown signals as well, they are far more applicable in the real world, where the input data can not be guaranteed to be within the data set that the closed set model is trained on.

VII. Acknowledgments

We would like to thank SPAR Labs for their support on this project.

References

[1] Ridhima and A. Singh Buttar, ”Fundamental Operations of Cognitive Radio: A Survey,” 2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT), 2019, pp. 1-5.

[2] Q. Zhao and B. M. Sadler, ”A Survey of Dynamic Spectrum Access,” in IEEE Signal Processing Magazine, vol. 24, no. 3, pp. 79-89, May 2007.

[3] S. R. Shebert, A. F. Martone, and R. M. Buehrer, “Open set wireless standard classification using Convolutional Neural Networks,” arXiv.org, 03-Aug-2021. [Online]. Available: https://arxiv.org/abs/2108.01656.

[4] T. J. O'Shea, T. Roy, and T. C. Clancy, “Over the air deep learning based radio signal classification,” arXiv.org, 13-Dec-2017. [Online]. Available: https://arxiv.org/abs/1712.04578.

[5] D. S. Inc., “RF datasets for Machine Learning,” DEEPSIG DATASET: RADIOML 2016.10A, 2016. [Online]. Available: https://www.deepsig.ai/datasets.

[6] A. Bendale and T. Boult, “Towards open set deep networks,” arXiv.org, 19-Nov-2015. [Online]. Available: https://arxiv.org/abs/1511.06233.

[7] Z. Y. Ge, S. Demyanov, Z. Chen, and R. Garnavi, “Generative OpenMax for Multi-class open set classification,” arXiv.org, 24-Jul-2017. [Online]. Available: https://arxiv.org/abs/1707.07418.

[8] A. Rozsa, M. Günther, and T. E. Boult, “Adversarial robustness: Softmax versus openmax,” arXiv.org, 05-Aug-2017. [Online]. Available: https://arxiv.org/abs/1708.01697.

[9] W. J. Scheirer, A. Rocha, R. J. Micheals and T. E. Boult, ”Meta-Recognition: The Theory and Practice of Recognition Score Analysis,” in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 33, no. 8, pp. 1689-1695, Aug. 2011.