Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

School of Electronic Engineering

Detection Algorithms for Packet Based Wireless Networks Using Deep Learning

Project Portfolio

Niall Lyons
ID Number: 13493628

August 2020

MEng in Electronic and Computer Engineering

Supervised by Dr. Kevin McGuinness

ACKNOWLEDGEMENTS

My greatest gratitude is extended to everyone who supported this research. For their enthusiastic support and belief, my family and friends. For their academic guidance, my DCU supervisor Dr. Kevin Mc Guinness and secondary assessor Dr. Dushyantha A Basnayaka. My Cypress Semiconductor colleagues and industrial supervisor Kiran Uln for his day today knowledge and excellent mentorship. For the invaluable scholarship opportunity to pursue such research, the Dublin – San Jose Sister cities.

TABLE OF CONTENTS

ACKNOWLEDGMENTS **iii**

PAPER **1**

INTRODUCTION 1
PRIOR WORK 1
TRADITIONAL WIRELESS PACKET DETECTION 1
DEEP LEARNING APPROACHES 2
TECHNICAL DESCRIPTION 2
DEEP NEURAL NETWORK OVERVIEW 2
DATA ACQUISITION 2
RECURRENT NEURAL NETWORK 3
LONG SHORT-TERM MEMORY 3
BIDIRECTIONAL LONG SHORT-TERM MEMORY 3
MODEL TRAINING 4
EVALUATION METRICS 4
EXPERIMENTAL RESULTS AND ANALYSIS 4
BINARY SYMMETRIC CHANNEL DATA 4
ADDITIVE WHITE GAUSSIAN NOISE DATA 5
MULTIPATH DATA 5
CONCLUSION AND FUTURE WORK 5
ACKNOWLEDGMENT 6
REFERENCES 6

APPENDIX A - LITERATURE REVIEW

APPENDIX B - PROJECT PLAN

APPENDIX C - RESEARCH LOG

APPENDIX D - PROJECT DESIGN AND IMPLEMENTATION

APPENDIX E - TESTING AND RESULTS

APPENDIX F - SOURCE CODE LISTING

Detection Algorithms for Packet Based Wireless Networks Using Deep Learning

Niall Lyons

***Abstract*—This paper discusses packet detection algorithms for 802.11b wireless preambles using different deep learning architectures. Two different approaches are examined, namely traditional methods and deep learning and compared by detection performance. From a traditional standpoint, preamble packet detection using correlation and its limitations are investigated. The role of deep learning is examined from three different deep learning architectures with memory, recurrent neural network (RNN), Long-short-term memory (LSTM) and Bi-directional LSTM (BLSTM). Each model's performance is evaluated against traditional detectors and each is found to outperform traditional methods considerably. BLSTM while minimizing complexity has slight superiority of evaluation metrics over RNN and LSTM. The value of using deep learning in wireless packet detection can reduce cost of hardware and increase efficiency across the industry.**

***Index Terms*—deep learning, preamble detection, wireless networks, RNN, LSTM, BLSTM**

I. Introduction

THE central research problem discussed and investigated in this paper and preceding literature review asks:

"Is there a tradeoff between traditional preamble detection techniques, against deep learning architectures in 802.11b wireless packet detection?"

For the purpose of this paper, the preamble of a wireless packet is specifically looked at for detection. Prior research has extensively explored traditional preamble detection and frame synchronization [1-3]. The preamble is a bit sequence that allows the receiver to obtain time frequency synchronization and channel estimation for each payload transmission [4]. Prior research explores deep learning being applied to wireless communication topics including channel estimation and the communication with encoders. This paper investigates, recurrent neural networks (RNN), long short-term memory (LSTM) and bi-directional LSTM (BLSTM) are evaluates performance metrics against traditional methods ensuring low complexity. The analysis assumes additive white gaussian noise (AWGN) with models robust to multipath channels. I describe in the literature review how a multivocal approach was adapted during this research, which continued throughout. Using this approach, a wide variety of viewpoints and various approaches could be analyzed and accessed. Examining detection algorithms for packet based wireless networks required setting specific search terms and alias, to gain a perspective on the ever-changing workings over time.

To determine a tradeoff between traditional and deep learning approaches to detection past analogies and methods were looked at. Specifically showing, how deep learning (DL) approaches significantly reduces the complexity and improves the performance and accuracies in comparison to traditional simple detectors. Thus, answering the question referred to in the literature review that through the use of deep learning architectures, hardware firmware can be re-configured without the need for redesign, consequently reducing costs.

Appendix A represents the literature review that looks at the initial research outlining the review and analysis of work that has been done on this topic in the past along with the intended approaches for this research paper. As the core of this research portfolio, this paper looks at prior cutting-edge methodologies in Section II, while in Section II I thoroughly go through the technical description of deep learning architectures used and their benefits to the problem. Section IV investigates the results and performs a relevant analysis. From these results, in Section V, future research proposals are introduced, and conclusions drawn.

II. Prior Work

To evaluate the impact that deep learning can have on the detection of packet based wireless networks, traditional methods of detection were evaluated against different DL architectures. The models that were implemented throughout this project were RNN, LSTM and BLSTM.

Furthermore, the trade-off between each architecture and how they outperform traditional simple detectors was evaluated with each architecture performing extensively superior to traditional methods, minimizing model complexity [5].

A. Traditional Wireless Packet Detection

Traditionally, wireless packet detection consists of calculating the correlation of possible packet preamble sequences and the received signal. A threshold value modified essentially given the noise floor is used to detect the signal efficiently. The improved composite pseudo noise correlation (ICMP) method of detection correlates preconceived preamble sequences against received signals [6]. The correlation of both, shows peaks at specific positions where preamble sequences are detected, while if no peaks occur the correlation value is commensurate to the length of the preamble sequence.

Traditional method detection algorithms are used in many communication systems [7-9]. However, with these approaches to achieve robust,scalable results the algorithm requires some feature engineering. By taking a deep learning approach, this project will evaluate how these problems can be overcome to produce more robust, low complexity algorithms outlined in [10-18].

B. Deep Learning Approaches

DL techniques are rapidly becoming widely used in the wireless communications domain in various areas [18-19]. Many applications have engineered architectures for specific classification, [20] and regression, [21] problems to significantly improve robustness, scalability and design. Thus, resulting in a better performing application.

Several DL approaches have been investigated to determine sequence classifications. Research in [10], outlines DL architectures using the Poisson channel model applied to optical and molecular systems. A Sliding bidirectional recurrent neural network (SBRNN) technique is trained to estimate the data in real-time as the signal stream arrives at the receiver. Similarities between molecular and wireless communication systems is shown in Fig. 1. The approach taken here is different from prior works since it is assumed that the mathematical models for the communication are Poisson channels.

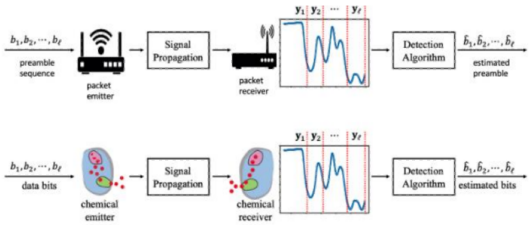


Fig. 1. Similarity between molecular communication using a Poisson channel model and wireless communication using an additive white gaussian noise channjel

In further research, [22] investigates the use of convolutional neural networks (CNN) for preamble detection and Time of Arrival (TOA) estimation without the need of knowing the transmit waveform. The specific waveforms given, are classified against received waveforms with the addition of AWGN, due to the multipath propagation effect. Through the use of a CNN, trained on both synthetic and real measured data shows the prediction accuracy in comparison to traditional approaches improved by three times, while carrying the same computational complexity.

III. Technical Description

This paper builds upon the traditional correlation methods of detection, finding the tradeoff between DL approaches and traditional simple detectors. The theory implemented throughout this paper and architectures researched are outlined here.

A. Deep Neural Network Overview

With emerging DL technologies, and their ability to classify and predict specific outcomes, models are being produced in industry at scale. As outlined above, an 802.11b wireless packet preamble is a pseudorandom sequence, which will be used for detection. Thus, different artificial NN architectures were implemented to utilize their memorizing capacities for long range preamble sequences, namely RNN, LSTM and BLSTM. Fig. 2, outlines the general architecture of a neural network, where the input vectors are $x_1, x_2, ..., x_n$. W represents the weights connection to the next layer, bias is represented by b_k and y_k is the final output after being passed through an activation function.

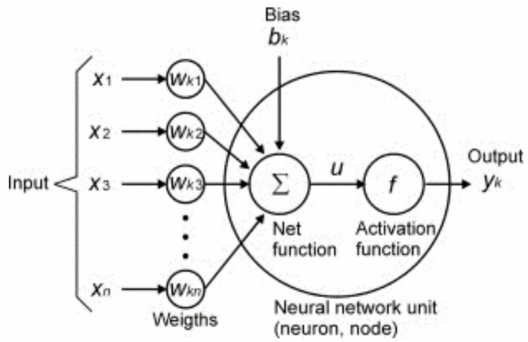


Fig. 2. General structure of a Neurel Network , showing weights conecting the input vectors to the next layer then being passed to an activation function to produce an output [23]

B. Data Acquisition

Prior research substantiates the belief that an 802.11b wireless packet uses direct sequence spread spectrum (DSSS), to reduce overall signal interference. Packet bits are modulated using pseudorandom spreading sequences, to provide signal privacy against multiple packets using the same resource. For the purpose of this paper maximum length sequences are used as the spreading sequence due to excellent auto correlation abilities for large sequences.

M-sequences are generated using linear feedback shift registers (LFSR) structures. Galois field architecture is applied in the generation of sequences. (1), denotes an L_{th} order polynomial in GF(2) taking only binary values. L represents delay elements, where $g_0, g_1, ..., g_L$ represent the coefficients of the generator polynomial [27].

$$g(x) = g_0 + g_1 x + g_2 x^2 + \cdots + g_{L-1} x^{L-1} + g_L x^L \quad (\mathrm{mod}\, 2) \quad (1)$$

Comprehensive simulations of three datasets generated using contrasting elements of noise and channels are evaluated.

1) Binary Symmetric Channel (BSC): Both channel input and output are binary, with error probability based on receiving an incorrect bit. Each data point, takes an m-sequence and is passed through a BSC model for a given error probability, resulting in a randomly flipped sequence.

2) Additive White Gaussian Noise (AWGN) Channel: This dataset consists of m-sequence datapoints over an AWGN channel, whose amplitude is gaussian distributed. Various signal to noise ratios (SNR) values in the range of -15 to 10 in steps of 5 are simulated.

3) Multipath Channels: Multipath channels arrive at the receiver with multiple different strengths and phase angles of the same signal causing many variations. Each datapoint generated consists of 100 random normalized channels convolved with the current LFSR sequence of each packet, with added AWGN with a SNR of -5 dB.

C. Recurrent Neural Network

As referred to in the literature review an RNN is a generalization of a feed forward neural network that has an internal memory [10]. Thus, when making a decision the model considers the input and output that it has learned previously. [11], takes the approach defined by

$$h_t = j_h\left(x_t, h_{t-1}\right) = \phi_h\left(W^{\tau} h_{t-1} + U^T x_t\right) \tag{2}$$

$$y_t = j_0\left(h_t, x_t\right) = \phi_o\left(V^T h_t\right) \tag{3}$$

Where W, U and V are the transition, input and out matrices ϕ_h and ϕ_o are element wise nonlinear functions. While having an input x_t, an output y_t , and hidden state h_t.

The received signal was first reshaped to a three-dimensional array, with X number of samples, one time step and 38 features. The three-dimensional input sequence then goes through the model with 38 neurons. To reduce the overall complexity of the model, a recurrent dropout rate of 0.2 is introduced on the simple RNN layer, with the addition of two dense layers gradually reducing the kernel size. Table one, shows the final model summary. The sigmoid activation function was also used [24]. ReLU activation was considered here, however as outlined in the literature review RNNs are unable to process very long sequences if used. The resulting model yielded a resounding performance accuracy evaluated against traditional detectors, seen in Section IV.

However, with RNN's the vanishing gradient and exploding gradient problem must be addressed, resulting in a non-robust model using multipath data.

TABLE I
RNN MODEL SUMMARY

Type	Output Shape	Parameters
Simple RNN Layer	(None,1,38)	2926
Global Max Pooling	(None,38)	0
Dense Layer	(None,16)	624
Dropout	(None,16)	0
Dense Layer	(None,1)	17

D. Long Short-Term Memory

RNNs come with a degree of complications. LSTM networks [12], a modified version of RNN, resolves the above vanishing gradient problem resulting in a more robust, low complexity model for the purpose of this project. Fig. 3, illustrates a general LSTM architecture an added output gate, forget gate, input gate and memory cell which in turn stores information over long periods of time. The added gates of LSTM are used to take the model offline, delete sequences and neglect incoming activations [15]. Through this, working with long preamble sequences illustrates a straightforward process.

The input gate, forget gate, output gate and cell state vector are outlined as $i(t)$, $f(t)$, $o(t)$, $c(t)$ while computed by standard means highlighted in the aforementioned literature review.

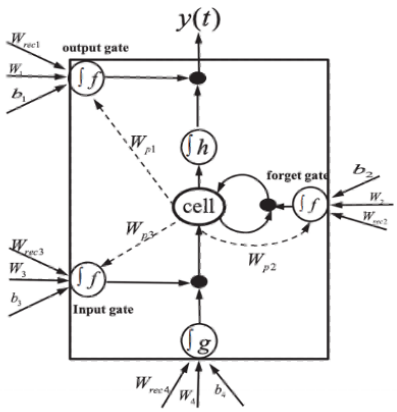


Fig. 3. General Structure of LSTM model. The LSTM cell shows the nput gate, forget gate, output gate. The LSTM use both sigmoid and tanh activation functions to produce outputs

The semantic representation of the preamble is shown by x_t when the input preamble sequence is the last preamble sequence y_t [16].

As illustrated in the RNN approach, a low complexity solution is paramount. The received signal is reshaped into a three-dimensional array the same as the previous approach and passed to a two-layer LSTM model using 38 neurons, with one dense layer using the sigmoid activation function. Table two shows the final model summary.

TABLE II
LSTM MODEL SUMMARY

Type	Output Shape	Parameters
LSTM Layer	(None,1,38)	11704
LSTM Layer	(None,38)	11704
Dense Layer	(None,1)	39

However, the LSTM architecture can be re-engineered to understand the context of the preamble and preserve information from both the past and future by running input two ways. Using a BLSTM this can be achieved.

E. Bidirectional Long Short-Term Memory

BLSTM models hold a significant advantage over previous architectures, as preamble sequences can be preserved. A BLSTM runs inputs in two ways, one from the future and one from the past. In this model, input is presented in forward and backward states to two separate LSTM networks, that are both connected to the same output layer [17]. By preserving information from the future and using two hidden states $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ respectively, we can combine both layers and successfully

preserve preamble information from both the past and future to compute the output sequence y_t shown in Fig. 4

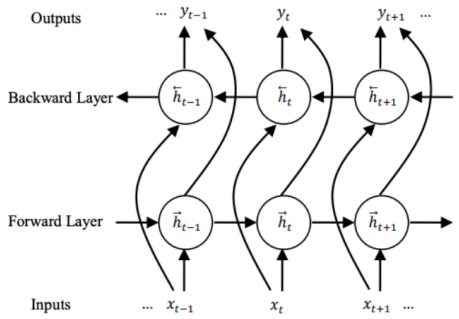


Fig. 4. General Structure of BLSTM model. The bottom LSTM layers shows the forward feature, while the top LSTM feature is used for backward feature. Both LSTM layers connect to an activation function to produce outputs

Likewise, in RNN and LSTM model data preprocessing, the received signal is reshaped and fed to the model. In this case, the final models consist of two initial Bidirectional LSTM layers with 38 neurons followed by a global pooling layer, reducing dimensionality and complexity. As used in the LSTM model, a dense layer is used to reduce the kernel size to 16, followed by a dropout rate of 0.2. The features are then extracted with a final dense layer using the sigmoid activation function. The final model summary can be seen in table 3.

By using this model, the robustness and resilience of the model was maximized to ensure significantly improved results over the LSTM and RNN models shown in Section IV.

TABLE III
BLSTM MODEL SUMMARY

Type	Output Shape	Parameters
Bidirectional LSTM Layer	(None,1,62)	15624
Bidirectional LSTM Layer	(None,62)	23312
Global Max Pooling	(None,62)	0
Dense Layer	(None,16)	1008
Dropout 0.2	(None,16)	0
Dense Layer	(None,1)	17

F. Model Training

A training and test dataset were derived from each dataset before training commenced. The datasets generated in part B, contain an input x and the corresponding label y. The input x represents the received signal sequence. The label y represents the binary representation of signal detection. The loss function used to evaluate each model was binary cross-entropy.

Where y is the label, with 1 representing a dected preamble sequence, otherwise 0 and $p(y)$ evaluating to the predicted probability of the sequence being a detected preamble for all data points. The optimizer used across all final models is the efficient Adam optimizer, however the SGB optimizer was tested yielding poorly results. The model is fit over multiple epochs, with a reduced learning rate callback, with a factor of 0.1. Resulting in a model learning rate being reduced based on a patience value of 2.

The testing stage consists of the unseen testing data being evaluated with the trained model. It is ensured that the model uses data it has not encountered previously. The output of each models represents a detected preamble or otherwise.

G. Evaluation Metrics

The academic community has extensively explored evaluation metrics to determine the effectiveness and performance of DL models. In this paper, Neyman Pearson Lemma curevs are used to dissect the performance and tradeoff between traditional and DL approaches of detection of the final models. False alarm rates are plotted against missed detection rates to outline the performance of the different methods used [24]. Receiver Operating Characteristic (ROC) curves are used to compare models for each dataset by plotting the true positive rate against false positive rate for varying thresholds [25]. Area Under Curve (AUC) curves are also evaluated to aggregate the measurement of performance across all possible thresholds. Precision and recall of models are also considered for comparison of model performance.

IV. Experimental Results and Analysis

A. Binary Symmetric Channel Data

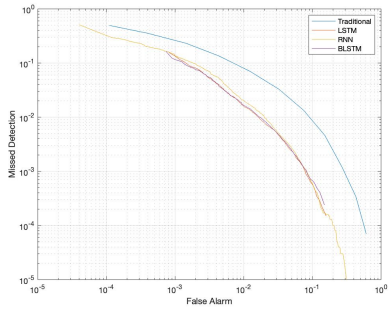
Each DL architecture's detection performance is evaluated using the aforementioned evaluation metrics. Fig. 5 (a) compares traditional methods and DL methods of detection using Neyman-Pearson curves to determine the highest performance. If a packet is detected, where the prediction is wrong the error represents a false alam, since we mistakenly detected the wrong packet received. This is shown on the x axis. In contrast to not detecting a packet, when it is a packet represents missed detection shown on the y axis. With a probability threshold of, λ as it decreases or increases the curve becomes smaller or larger.

TABLE IV
Performance Evaluation of Models of BSC Data

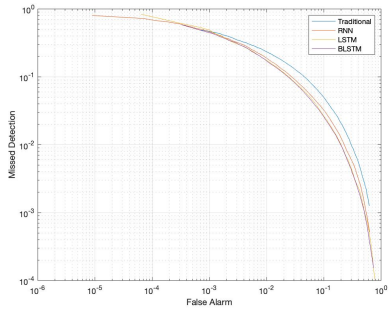
	RNN	LSTM	BLSTM
Parameters	2,482	16,153	39,961
Accuracy	0.98	0.98	0.98
Precision	0.97	0.98	0.98
Recall	0.98	0.98	0.97
AUC	0.99	0.99	0.99

In (a) the trained model curves are significantly inside the traditional method resulting in the level of detection being higher. The closer the models NP curve comes to the

traditional curve shows that the level of detection is reducing. It can be seen in Table. IV that the RNN model has the highest performance with a lower rate of trainable parameters ensuring low complexity. It specifies when a packet is correctly detected, the model is correct 96% of the time, while also indicating a correct detection of 98% of packets.



(a) BSC Data



(b) AWGN Data

Fig. 5. Neyman Pearson Curves of Traditional Detection Methods Vs DL models using BSC and AWGN Data

B. Additive White Gaussian Noise Data

TABLE V
PERFORMANCE EVALUATION OF MODELS OF AWGN DATA

	RNN	LSTM	BLSTM
Parameters	2,482	15,656	39,961
Accuracy	0.94	0.94	0.94
Precision	0.95	0.94	0.95
Recall	0.92	0.94	0.94
AUC	0.98	0.98	0.98

The findings here, surmise the effectiveness of performance metrics of models. Initially, substandard performance results initiated that traditional approaches outperformed all DL models substantially. However, the generation of additional data rapidly increased performance metrics to outperform traditional approaches. As mentioned in Section III, varying SNR dB values ranging from -15 dB to 10 dB were evaluated. As data encoding SNR is 0 dB, -5 dB SNR is analyzed. Fig. 5 (b), as outlined above outlines NP curves to compare different methods. In contrast to Fig. 5 (a) the margin of detection has reduced, however this is expected as the level of noise has increased. Table. V shows that the BLSTM model detection performance is high, however relatively close to the RNN with 10x lower training parameters.

However, as the value of SNR dB increases the less noise is in the received signal, resulting in higher performing detection models. In contrast, as the value of SNR dB decreases additional noise has a relatively small effect on the performance of each model. Table four, strengthens this observation. Evidently, through each value of SNR dB examined the DL models substantially outperform traditional methods while ensuring low complexities discussed in part (d) of this section.

TABLE VI
LSTM MODELS FOR VARYING SNR DB SUMMARY

	Precision	Recall
10 SNR dB	0.97	0.99
-10 SNR dB	0.69	0.71
0 SNR dB	0.99	0.99

C. Multipath Data

TABLE VII
PERFORMANCE EVALUATION OF MODELS OF MULTIPATH DATA

	RNN	LSTM	BLSTM
Parameters	3,567	23,447	59,617
Accuracy	0.95	0.95	0.94
Precision	0.96	0.97	0.97
Recall	0.94	0.94	0.92
AUC	0.94	0.95	0.94

Fig. 6 looks at the NP curves to compare traditional methods and DL methods on multipath data. Interestingly, it can be seen that the margin between traditional and DL approaches is large, showing that all DL models increase the level of detection even through multipath data with added AWGN noise. These results show how far DL models can bring the detection of packets, thus, reducing costs as hardware will not have to be redesigned with the updating of firmware serving sufficient. The scalability and robustness of models outweighs traditional methods while keeping the same complexity.

V. CONCLUSION AND FUTURE WORK

This paper illustrates the power of DL architectures and when applied correctly can have a major impact on packet detection and the wireless communications industry. Three neural network architectures with memory were applied to

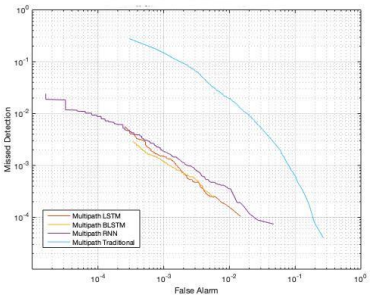


Fig. 6. Neyman Pearson Curves of Traditional Detection Methods Vs DL models using Multipath Channel Data

three separate datasets in this paper to find the tradeoff between DL approaches and traditional simple packet detectors. The models implemented outperform traditional methods immensely in accuracy, precision and recall while ensuring low complexity.

The BLSTM architecture is found to perform significantly outweigh traditional approaches for low values of SNR dB and multipath channels.

Future work includes running the model on sequential input and performing anomaly detection within each packet to determine its relevance.

ACKNOWLEDGMENT

My greatest gratitude is extended to everyone who supported this research. For their enthusiastic support and belief, my family and friends. For their academic guidance, my DCU supervisor Dr. Kevin Mc Guinness and secondary assessor Dr. Dushyantha A Basnayaka. My Cypress Semiconductor colleagues and industrial supervisor Kiran Uln for his day to day knowledge and excellent mentorship. For the invaluable scholarship opportunity to pursue such research, the Dublin – San Jose Sister cities.

REFERENCES

[1] Y. Mostofi and D. C. Cox, "Robust timing synchronization design in OFDM systems - Part II: high-mobility cases," in IEEE Transactions on Wireless Communications, vol. 6, no. 12, pp. 4340-4348, December 2007, doi: 10.1109/TWC.2007.05901.

[2] M. Chiani and M. G. Martini, "Practical frame synchronization for data with unknown distribution on AWGN channels," in IEEE Communications Letters, vol. 9, no. 5, pp. 456-458, May 2005, doi: 10.1109/LCOMM.2005.1431170.

[3] Y. Mostofi and D. C. Cox, "A robust timing synchronization design in OFDM systems-part I: low-mobility cases," in IEEE Transactions on Wireless Communications, vol. 6, no. 12, pp. 4329-4339, December 2007, doi: 10.1109/TWC.2007.05900.

[4] IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Further Higher Data Rate Extension in the 2.4 GHz Band," in IEEE Std 802.11g-2003 (Amendment to IEEE Std 802.11, 1999 Edn. (Reaff 2003) as amended by IEEE Stds 802.11a-1999, 802.11b-1999, 802.11b-1999/Cor 1-2001, and 802.11d-2001) , vol., no., pp.1-104, 27 June 2003, doi: 10.1109/IEEESTD.2003.94282.

[5] "Model Complexity - an overview | ScienceDirect Topics", Sciencedirect.com, 2020. [Online]. Available: https://www.sciencedirect.com/topics/computer-science/model-complexity. [Accessed: 27- Aug- 2020].

[6] J. Park, M. Kim, D. Hwang and J. Han, "Analysis on Preamble Detection and False Alarm Probabilities of ICPC Method," 2015 IEEE 82nd Vehicular Technology Conference (VTC2015-Fall), Boston, MA, 2015, pp. 1-5, doi: 10.1109/VTCFall.2015.7390844.

[7] S. Kim, K. Joo and Y. Lim, "A delay-robust random access preamble detection algorithm for LTE system," 2012 IEEE Radio and Wireless Symposium, Santa Clara, CA, 2012, pp. 75-78, doi: 10.1109/RWS.2012.6175341.

[8] W. Chen, X. Wen, L. Yang, W. Li and T. A. Gulliver, "Dynamic preamble scheduling and detection for cooperative relay networks," Proceedings of 2011 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, Victoria, BC, 2011, pp. 131-134, doi: 10.1109/PACRIM.2011.6032880.

[9] S. Nagaraj, S. Khan, C. Schlegel, and M. Burnashev,"On preamble detection in packet-based wireless networks," in IEEE International Symposium onSpread Spectrum Techniques and Applications, 2006,pp. 476–480, doi: 10.1109/ISSSTA.2006.311817.

[10] N. Farsad and A. Goldsmith, "Neural Network Detection of Data Sequences in Communication Systems."

[11] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio,"How to Construct Deep Recurrent Neural Networks."

[12] S. Squartini, A. Hussain, and F. Piazza,"Preprocessing based solution for the vanishing gradient problem in recurrent neural networks," in Proceedings - IEEE International Symposium on Circuits and Systems, 2003, vol. 5, doi: 10.1109/iscas.2003.1206412

[13] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Comput., vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735

[14] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Comput., vol. 9, no. 8, pp. 1735– 1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[15] A. Graves, N. Beringer, and J. Schmidhuber, "Rapid Retraining on Speech Data with LSTM Recurrent Networks Instituto Dalle Molle di studi sull' intelligenza artificiale Galleria 2 CH-6900 Manno, Switzerland Rapid Retraining on Speech Data with LSTM Recurrent Networks," 2005

[16] Y. Lu, Y. Shi, and J. Yang, "A new method for semantic consistency verification of aviation radiotelephony communication based on LSTM-RNN," in International Conference on Digital Signal Processing, DSP, Jul. 2016, vol. 0, pp. 422–426, doi:10.1109/ICDSP.2016.7868592.

[17] A. Graves and J. Schmidhuber, "Framewise Phoneme Classification with Bidirectional LSTM Networks."

[18] F. Belabed and R. Bouallegue, "A comparative Analysis of Machine Learning Classification Approaches for Fountain Data Estimation in Wireless Sensor Networks," 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC), Tangier, Morocco, 2019, pp. 1251-1254, doi: 10.1109/IWCMC.2019.8766690.

[19] H. Huang et al., "Deep Learning for Physical-Layer 5G Wireless Techniques: Opportunities, Challenges and Solutions," in IEEE Wireless Communications, vol. 27, no. 1, pp. 214-222, February 2020, doi: 10.1109/MWC.2019.1900027.

[20] F. Belabed and R. Bouallegue, "A comparative Analysis of Machine Learning Classification Approaches for Fountain Data Estimation in Wireless Sensor Networks," 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC), Tangier, Morocco, 2019, pp. 1251-1254, doi: 10.1109/IWCMC.2019.8766690.

[21] H. Huang et al., "Deep Learning for Physical-Layer 5G Wireless Techniques: Opportunities, Challenges and Solutions," in IEEE Wireless Communications, vol. 27, no. 1, pp. 214-222, February 2020, doi: 10.1109/MWC.2019.1900027.

[22] H. Sun, A. O. Kaya, M. Macdonald, H. Viswanathan and M. Hong, "Deep Learning Based Preamble Detection and TOA Estimation," 2019 IEEE Global Communications Conference (GLOBECOM), Waikoloa, HI, USA, 2019, pp. 1-6, doi: 10.1109/GLOBECOM38437.2019.9013265.

[23] H. Apaydin, H. Feizi, M. T. Sattari, M. S. Colak, S. Shamshirband, and K.-W. Chau, "Comparative Analysis of Recurrent Neural Network Architectures for Reservoir Inflow Forecasting," Water, vol. 12, no. 5, p. 1500, May 2020.

[24] [3]Web.stanford.edu, 2020. [Online]. Available: https://web.stanford.edu/ lmackey/stats300a/doc/stats300a-fall15-lecture13.pdf. [Accessed: 27- Aug- 2020].

[25] "Receiver-Operating Characteristic Analysis for Evaluating Diagnostic Tests and Predictive Models," Circulation, 06-Feb-2007. [Online]. Available: https://www.ahajournals.org/doi/10.1161/CIRCULATIONAHA.105.594929. [Accessed: 27-Aug-2020].

APPENDIX A

LITERATURE REVIEW

Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

School of Electronic Engineering

Detection Algorithms for Packet based Wireless Networks using Deep Learning

Literature Survey

Niall Lyons
ID Number: 13493628

March 2020

MEng in Electronic and Computer Engineering

Supervised by

Dr. Kevin McGuinness

Declaration

I hereby declare that, except where otherwise indicated, this document is entirely my own work and has not been submitted in whole or in part to any other university.

Signed: Niall Lyons Date: 09/04/20

Detection Algorithms for Packet Based Wireless Networks Using Deep Learning

Niall Lyons

***Abstract*—In this paper, detection algorithms for packet based wireless networks using deep learning will be discussed. This will be examined using two different approaches, namely traditional methods of packet detection and the application of deep learning models to detect wireless packets. From a traditional standpoint, it is first looked at preamble packet detection using correlation and the limitations that it has. The role of deep learning is then examined from three different deep learning architectures, recurrent neural network (RNN), Long short-term memory (LSTM) and Bi-directional LSTM (BLSTM). Each of the architectures are then assessed in terms of performance and precision and evaluated against traditional methods of detection. It is found that the BLSTM neural network architecture will perform at a higher level to the LSTM and RNN while being compared to traditional methods. The value of using deep learning in wireless packet detection is found to reduce cost of hardware and efficiency across industry.et**

***Index Terms*— deep learning, preamble detection, wireless networks, communication systems**

I. Introduction

PACKETIZED communication is quickly becoming the standard throughout wireless systems used in industry today. As stated by IEEE standards for local, metropolitan and personal area networks and several other communication standards. In this form of communication, packets are random and unknown to the receiver. For the purpose of this paper, the preamble of the packet will specifically be looked at. The preamble is a section of data prefixed for each payload transmission at the head of the wireless packet to facilitate the process of synchronization. Figure 1 shows the typical packet format showing the exact starting time of the payload. Therefore, throughput this paper preamble detection in wireless based networks is addressed.

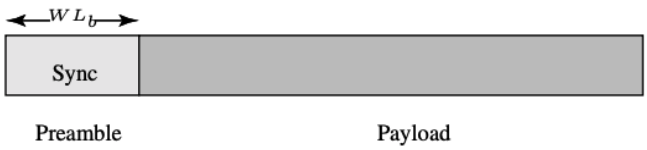


Fig. 1. Packet Format

When researching this topic, a multivocal approach was adopted, meaning that both peer reviewed and non-peer-reviewed artefacts were included. With this approach, a wide range of perspectives and different methods could be addressed and evaluated. Examining detection algorithms for packet based wireless networks using deep learning is an expensive undertaking, meaning specific search terms and alias were considered. Through a preliminary examination of literature, it was found that at the present time there are few published documents relating to this topic that consider deep learning. Therefore, workings over time were considered genuine to gain a perspective on methods introduced over time.

To thoroughly evaluate how deep learning methodologies can benefit the detection of wireless networks based on the packet's preamble, past analogies must be looked at. Specifically, how deep learning can change the problems, accuracies and dramatically improve the preconceived state of the art methodologies of the past. In particular, with the use of deep learning, hardware can be re-configured without the need for redesign, thus reducing cost. Along with this, how the robustness and low complexity detection of deep learning models outweighs previous traditional detection algorithms. However, new techniques must be extensively tested to ensure detection algorithms can perform under changing noisy channel conditions without underlying knowledge of channel models.

Traditional methods of detection have changed overtime; however, the deep learning approach has branched out to different types of deep learning methodologies. In this paper, recurrent neural networks (RNN), long short-term memory (LSTM) and bi-directional LSTM (BLSTM) will be evaluated against traditional methods. Identifying the discrepancies between traditional and deep learning architectures will eliminate the need for feature engineering and unnecessary costs and utilize the best deep learning approach to deliver high quality results.

Three different types of architectures with memory will be discussed and tested to determine the best approach to detect wireless packets, and show it is possible to train detectors that perform well, without any knowledge of underlying channel models. The accuracy of the BLSTM model will be evaluated against an RNN model and traditional digital signal processing methods, for comparison to determine the best possible method. These models will be evaluated using IEEE 802.11 [10] standard compliant data where the model and channels are unknown. The aim is to find the trade-off between BLSTM and RNN architectures and evaluate how they outperform traditional simple detectors.

The rest of the paper is organized as follows. In section II, traditional models of preamble detection are described, while the above deep learning architectures are analyzed in relation to prior work. In section III, prior work to the project problem is reviewed and how these workings can assist this particular topic. The conclusion follows in section V.

II. Review and Analysis of Prior Work

To evaluate the impact that deep learning can have on the detection of packet based wireless networks, traditional methods of detection must be evaluated against different deep learning architectures. The models that will be looked at throughout this project will be RNN, LSTM and BLSTM.

Furthermore, the trade-off between BLSTM and RNN architectures and how they outperform traditional simple detectors will be evaluated.

A. Traditional Methods

In [1], the performance bounds on detecting a preamble embedded at the beginning of every packet for communication over an additive white Gaussian noise channel are derived. Correlation based detection is used to detect the preamble with an extensive performance analysis undertaken. Here the probability of successful preamble detection is given by

$$P(success) = 1 - P(wrong) - P(miss) \qquad (1)$$

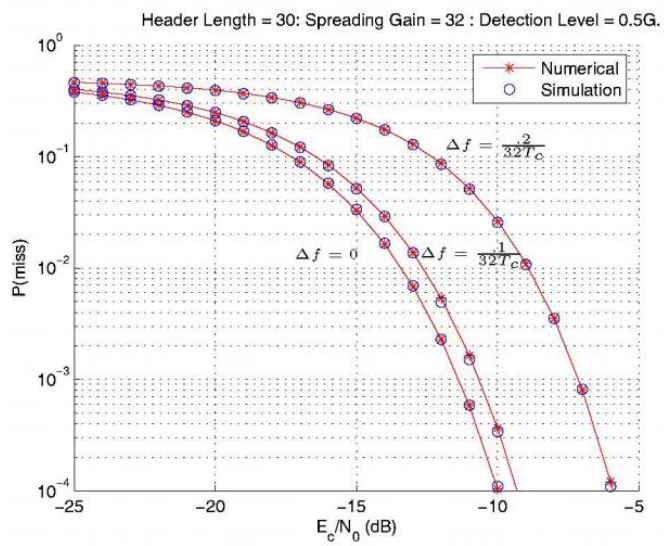
With the probability of miss given by

$$P(miss) = P(Re\,\eta_{i0} < G) = P\left(\sum_{i=0}^{W-1} \lambda_i\,|Z_i|^2 < 2G\right) \quad (2)$$

And the probability of wrong given by

$$\begin{aligned} P(wrong) &= \sum_{i=1}^{S} P(wrong|i)P(i_0 = i) \quad (3) \\ &\leq \sum_{i=1}^{S}\sum_{k=1}^{i-1} P(Re\,\eta_k \geq G|i)P(i_0 = i) \quad (4) \\ &\cong \left(\frac{1}{\lambda} - 1\right) P(Re\,\eta_1 \geq G) \quad (5) \end{aligned}$$

where $\frac{1}{\lambda} - 1$ represents the wireless packet communication traffic intensity. The overall performance of this method is outlined in figure 2 and figure 3.



Fig, 2. Probability of missing a packet using cross correlation for different SNR and offset values

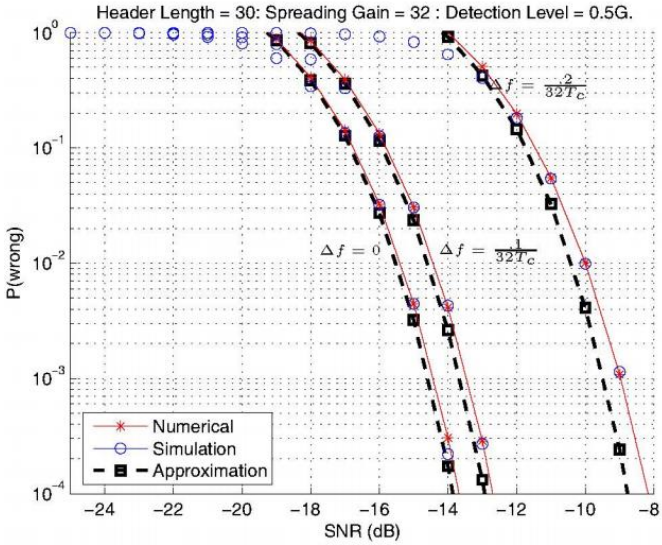


Fig. 3. Probability of erroneous packet detection using cross correlation for different SNR and offset values

Figure 2 shows the probability of packet loss for different signal noise ratio (SNR) and frequency offset values and reduces the probability of intercarrier inference. Thus, reducing the signal-to-noise ratio (SNR). Frequency offset values are calculated to reduce the interference between transmitter and receiver. Figure 3 shows the probability of erroneous packet detection.

However, with this approach to achieve these results the algorithm requires some feature engineering as the presence of frequency offset rotates the signal and degrades the performance. By taking a deep learning approach, this project will evaluate how these problems can be overcome to produce more robust, low complexity algorithms outlined in [2-9].

B. Recurrent Neural Network

To utilize the power of deep learning and overcome the problems above, as outlined in [3], an RNN is a generalization of a feed forward neural network that has an internal memory. Neural network (NN) architectures with memory are used over alternative NN architectures as preamble sequences are stored in memory to detect wireless packets. Therefore, when making a decision, it considers the current input and the output that it has learned previously. Taking the approach of an RNN in [4], it is defined by

$$h_t = j_h(x_{t,}h_{t-1}) = \phi_h(W^T h_{t-1} + U^T x_t) \quad (6)$$

$$y_t = j_o(h_{t,}x_t) = \phi_o(V^T h_t) \quad (7)$$

Where W, U and V are the transition, input and out matrices and ϕ_h and ϕ_o are element wise nonlinear functions. While having an input $x_{t,}$, an output $y_{t,}$ and hidden state $h_{t,}$.

However, the RNN methodology applied to the preamble detection has its disadvantages. RNNs are unable to process very long sequences if they use ReLU as an activation function. Resulting in a non-efficient and costing model. Moreover, the vanishing gradient and exploding gradient problem will have a sufficient impact on the performance of the model. [5] Thus, bringing its overall value and robustness of low quality.

C. Long Short-Term Memory

As outlined above, RNNs come with a degree of complications. LSTM networks [6], a modified version of RNN, resolves the above vanishing gradient problem developing a more robust, low complexity model for the purpose of this project. The architecture of an LSTM is shown in figure 5. This figure shows an added output gate, forget gate, input gate and memory cell which in turn stores information over long periods of time. The added gates of LSTM are used to take the model offline, delete sequences and neglect incoming activations. [8] An advantage of using this type of architecture in this project, as working with lengthened CDMA preamble sequences is straightforward.

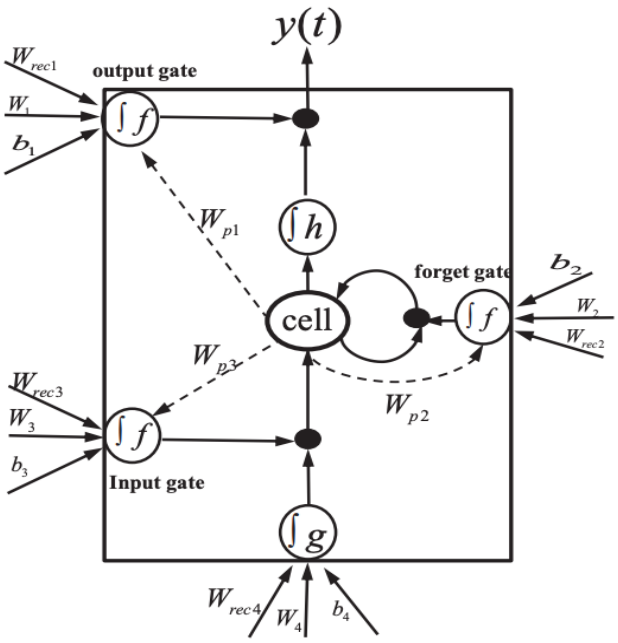


Fig. 5. LSTM Architecture

The output gate, forget gate, input gate and cell state vector are outlined as $o(t), f(t), i(t)$ and $c(t)$ are computed as follows

$$i(t) = \sigma\left(W_3 x(t) + W_{rec3} y(t-1) + W_{p3} c(t-1)\right) \quad (8)$$

$$f(t) = \sigma\left(W_2 x(t) + W_{rec2} y(t-1) + W_{p2} c(t-1)\right) \quad (9)$$

$$o(t) = \sigma\left(W_1 x(t) + W_{rec1} y(t-1) + W_{p1} c(t-1)\right) \quad (10)$$

$$l(t) = tanh\left(W_4 x(t) + W_{rec4} y(t-1)\right) \quad (11)$$

$$c(t) = f(t)c(t-1) + i(t)l(t) \quad (12)$$

$$y(t) = \tanh\left(c(t)\right) o(t) \quad (13)$$

where W_i and W_{reci} (i=1,2,3,4) are input connections and recurrent connections of output gate, forget gate, input gate and cell. W_{pi} (i=1,2,3) represents peephole connections. Activation functions $tanh(.)$ and $\sigma(.)$ are used. $\sigma(.)$ represents the sigmoid function. The semantic representation of the preamble is shown by x_t, when the input preamble sequence is the last preamble sequence y_t. [7]

The performance and robustness of LSTM over RNN will be looked at to test for tradeoff between detected probability and false alarm probability using a receiver operating characteristic (ROC) curve to display the relationship.

However, the LSTM architecture can be re-engineered to understand the context of the preamble and preserve information from both the past and future by running input two ways. Using a Bi-directional LSTM (BLSTM) this can be achieved.

D. Bi-directional Long Short-Term Memory

As outlined above we can preserve preamble data using a BLSTM. The BLSTM runs inputs in two ways, one from the future and one from the past. In this model, input is presented in forward and backward states to two separate LSTM networks, that are both connected to the same output layer. [9] By preserving information from the future and using two hidden states $\overleftarrow{h}$, $\overrightarrow{h}$ respectively, we can combine $\overleftarrow{h}$ and $\overrightarrow{h}$ to successfully preserve preamble information from both the past and future and compute the output sequence y shown in figure 6.

$$\overrightarrow{h}_t = \mathcal{H}\left(W_{x\overrightarrow{h}} x_t + W_{\overrightarrow{h}\overrightarrow{h}} \overrightarrow{h}_{t-1} + b_{\overrightarrow{h}}\right) \quad (14)$$

$$\overleftarrow{h}_t = \mathcal{H}\left(W_{x\overleftarrow{h}} x_t + W_{\overleftarrow{h}\overleftarrow{h}} \overleftarrow{h}_{t+1} + b_{\overleftarrow{h}}\right) \quad (15)$$

$$y_t = W_{\overrightarrow{h}y} \overrightarrow{h}_t + W_{\overleftarrow{h}y} \overleftarrow{h}_t + b_y \quad (16)$$

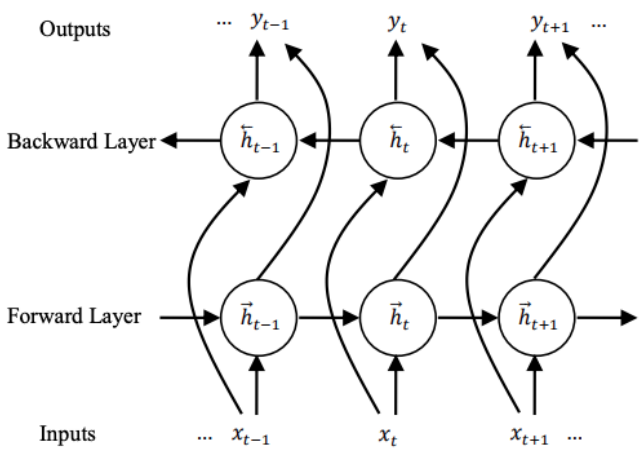


Fig. 6. BLSTM Architecture

By using this model, the robustness and resilience of the system will be maximized to ensure significantly improved results over the LSTM and RNN models. The tradeoff between each model in relation to traditional methods outlined in II part A, will be examined to determine the most robust model and by how much it outperforms traditional methods of wireless packet detection.

III. Review of Prior Work to the Project Problem

Part of the challenge when faced with a problem is to determine whether or not deep learning needs to be applied, and what advantages it will have over a traditional method. In this project, the aim is to determine what advantages different

deep learning models will have to traditional approaches in preamble by preamble detection in wireless based networks. The main problems being addressed by this project is by developing deep learning models, hardware can be re-configured without need for re-design, thus reducing cost. The robustness of these models is key to show it is possible to train detectors that perform well without any knowledge of underlying channel models.

In [3], many different NN detectors are directly trained using measurement data from experiments. These NN architectures use a Poisson channel model to be used in optical and molecular communication systems. Sliding bidirectional recurrent neural networks (SBRNN) are trained. After training, the detector estimates the data in real-time as the signal stream arrives at the receiver. Similarities between molecular and wireless communication systems is shown in figure 7. The approach taken here is different from prior works since it is assumed that the mathematical models for the communication are Poisson channels.

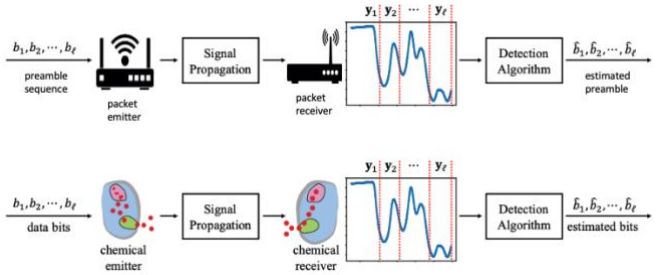


Fig. 7 Similarity between molecular and wireless communication systems

This project is a state-of-the-art deep learning approach to the detection problem in wireless communication systems. Different NN detection architectures will be trained using IEEE 802.11 standard generated preamble data. [10] The models outlined in section II, use preamble data with a gaussian channel model used in wireless communication systems. This model is used to compare the performance against traditional methods of detection. Through this comparison, the best model can be evaluated and tested thoroughly to ensure it is robust, accurate, resilient and fit for purpose.

The proposed solution aims to be show how deep learning can replace and outperform traditional wireless packet detectors. Each architecture's will visually show the progression from traditional methods through RNN and different types of RNN, LSTM and BLSTM.

IV. CONCLUSIONS

Packet based wireless detection algorithms in communication systems are complex and in earlier work rely on known underlying channel models. The challenges involved in developing an algorithm that can detect wireless packets without any known underlying channel models involves deciding what specific deep learning models we want to use and how their performance and architectures will benefit the problem we are trying to solve.

We have therefore examined three different models in depth, to evaluate the best possible method to solve the given problem. This research suggests that a BLSTM at the present time, will outweigh the RNN and LSTM models in terms of precision and accuracy. However, it can be said that computationally this could be expensive with the LSTM model the better option.

Researchers and engineers are emerging with new and improved deep learning models at a vast pace. This research indicates that deep learning is the future of detection algorithms in packet based wireless networks, although traditional methods work and work efficiently. The reduction of costs and re-configuration of hardware without redesign will evidently be used in industry going forward.

REFERENCES

[1] S. Nagaraj, S. Khan, C. Schlegel, and M. Burnashev, "On preamble detection in packet-based wireless networks," in *IEEE International Symposium on Spread Spectrum Techniques and Applications*, 2006, pp. 476–480, doi: 10.1109/ISSSTA.2006.311817.

[2] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553. Nature Publishing Group, pp. 436–444, May 27, 2015, doi: 10.1038/nature14539.

[3] N. Farsad and A. Goldsmith, "Neural Network Detection of Data Sequences in Communication Systems."

[4] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, "How to Construct Deep Recurrent Neural Networks."

[5] S. Squartini, A. Hussain, and F. Piazza, "Preprocessing based solution for the vanishing gradient problem in recurrent neural networks," in *Proceedings - IEEE International Symposium on Circuits and Systems*, 2003, vol. 5, doi: 10.1109/iscas.2003.1206412.

[6] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[7] Y. Lu, Y. Shi, G. Jia, and J. Yang, "A new method for semantic consistency verification of aviation radiotelephony communication based on LSTM-RNN," in *International Conference on Digital Signal Processing, DSP*, Jul. 2016, vol. 0, pp. 422–426, doi: 10.1109/ICDSP.2016.7868592.

[8] A. Graves, N. Beringer, and J. Schmidhuber, "Rapid Retraining on Speech Data with LSTM Recurrent Networks Instituto Dalle Molle di studi sull' intelligenza artificiale Galleria 2 CH-6900 Manno, Switzerland Rapid Retraining on Speech Data with LSTM Recurrent Networks," 2005.

[9] A. Graves and J. Schmidhuber, "Framewise Phoneme Classification with Bidirectional LSTM Networks."

[10] The Institute of Electrical and Electronics Engineers, "IEEE Standard for Information technology--Telecommunications and information exchange between systems LAN and MAN--Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Std 802.11-2016," *IEEE Std 802.11-2016 (Revision IEEE Std 802.11-2012)*, vol. 2016, pp. 1–3534, 2016, doi: 10.1109/IEEESTD.2016.7786995.

APPENDIX B

DESIGN PLAN

TABLE OF CONTENTS

INTRODUCTION **2**

RESEARCH QUESTION **2**
- PROBLEMS TO BE SOLVED 2

PROJECT SCOPE **3**
- DEEP LEARNING ARCHITECTURES 3
- TECHNOLOGIES TO BE USED 3

DESIGN APPROACH **3**
- DATA GENERATION 3
- MODEL GENERATION 3
- TESTING 4

REMOTE WORK CONFIRMATION **4**
- DCU 4
- CYPRESS SEMICONDUCTOR, SAN JOSE, CALIFORNIA 4

TIMELINE **5**

SUCCESS CRITERIA **6**

INTRODUCTION

My name is Niall Lyons and I am the San Jose Dublin sister cities Pat Mc Mahon Scholar for 2020, undertaking my thesis in San Jose, California with Cypress Semiconductor through DCU. For the duration of this project I am working alongside my industrial supervisor Kiran Uln and my Thesis supervisors at DCU Dr. Kevin McGuinness and Dr. Dushyantha Basnayaka.

After exploring potential projects with Cypress it was agreed to pursue this project titled Detection Algorithms for Packet Based Wireless Networks using Deep Learning.

The goal of this project is to focus on an important element of communication systems, the detection algorithms to show that deep learning algorithms are significantly more robust than traditional simple detectors. With the application of deep learning methods with memory, applying an RNN, Long Short-Term Memory (LSTM) and Bidirectional Long Short-Term Memory (BLSTM) network to detect wireless packets, and show it is possible to train detectors that perform well, without any knowledge of underlying channel models. The accuracy of the LSTM and BLSTM models will be evaluated against a Recurrent Neural Network (RNN) model and traditional digital signal processing methods, for comparison to determine the best possible method. These models will be evaluated using IEEE standard compliant data where the model and channels are unknown. The aim is to find the trade-off between LSTM, BLSTM and RNN architectures and evaluate how they outperform traditional simple detectors.

RESEARCH QUESTION

"Detection Algorithms for Packet Based Wireless Networks Using Deep Learning"

PROBLEMS TO BE SOLVED

- Robust and low complexity detection
- Hardware can be reconfigured without need for re-design, thus reducing costs
- Packets can be addressed for anomalies

PROJECT SCOPE

The scope of this project is to evaluate the trade off between deep learning architectures and technologies in relation to wireless packet detection in comparison to traditional simple detection methods.

DEEP LEARNING ARCHITECTURES

In this project I will be specifically looking at three separate deep learning architectures,

- Recurrent Neural Network (RNN)
- Long Short-Term Memory (LSTM)
- Bidirectional LSTM (BLSTM)

TECHNOLOGIES TO BE USED

- Python
- Keras
- Multiple deep learning python libraries
- Matlab

DESIGN APPROACH

DATA GENERATION

- In order to train the NN models I will be using Cypress IEEE standard test data. This data is developed using maximum length sequences based on the galois field theory to represent wireless packet preambles.

MODEL GENERATION

- I will first utilize the power of deep learning by developing an RNN model to determine the tradeoff between it, and the traditional approaches.
- LSTM models will then be developed to further enhance the power of deep learning by resolving the vanishing gradient problem within RNN architectures to produce a more robust, low complexity model.
- As a stretch goal of the project, I will also try to enhance the robustness of my final model by using a BLSTM to understand the context of the wireless packet preamble and preserve information from both the past and future by running input two ways. Using this model, input is presented in forward and backward

states to two separate LSTM networks, that are both connected to the same output layer. By preserving information from the future and using two hidden states we can successfully preserve preamble information from both the past and future and compute the output sequence.

- The trade off between each model in relation to traditional methods, will be examined to determine the most robust NN model and by how much it outperforms traditional methods of wireless packet detection.

TESTING

- Neyman-Pearson testing criterion will be used to represent the fundamental trade off in hypothesis testing and detection theory
- A receiver operating characteristic (ROC) curve will be used to display the relationship between detection probability and false-alarm probability respectively
- An array of deep learning model evaluation metrics will be used to ensure the most robust, low complexity model is developed. Some metrics that will be used include:
    - Logarithmic Loss
    - Confusion Matrix
    - ROC Curve
    - Area under Curve
    - F1 Score
    - Mean Absolute Error
    - Mean Squared Error

REMOTE WORK CONFIRMATION

DCU

- It has been agreed that all meetings will be done over zoom calls. Both Kevin and Dushyantha have made it clear that they are happy to answer any questions I have and are open to zoom meetings to discuss any technical topics.

CYPRESS SEMICONDUCTOR, SAN JOSE, CALIFORNIA

- Arrangements have been made to conduct regular meetings with my supervisor at Cypress to ensure the project is going smoothly.

TIMELINE

To ensure this project runs smoothly I generated the following tables for time management, separated into two sections, spring semester and summer internship respectively.

The project plan began on my first day at Cypress Semiconductor, San Jose. As the Spring semester in San Jose State University is over, I have successfully completed the first element of my time management framework. As each deliverable was reached, the project plan was and will continue to be updated.

Here we can see the spring semester consists of the core understanding of the project and the generation of data.

Task	Allocated Time	Target Completion	Status
SPRING SEMESTER 2020 – Beginning January 21st			
Begin Cypress Placement SJ	1 day	Week 1	Complete
Meet Supervisor and Team	1 day	Week 1	Complete
Explore potential projects	1 week	Week 2	Complete
Agree project with both academic and industrial advisor	1 day	Week 2	Complete
Research project field and latest related technologies	1 week	Week 3	Complete
Read related research papers	3 weeks	Week 6	Complete
Write Literature Review	2 weeks	April 9th	Complete
Test chosen NN architectures	2 weeks	Week 9	Complete
Oral Presentation	1 Week	May 9th	Complete
Generate test dataset using LFSR	1 week	Week 10	Complete
Apply noise and test on traditional methods	1 week	Week 11	Complete
Observe traditional method results	1 week	Week 12	Complete
END OF SEMESTER – EXAM PERIOD			

While the summer internship aspect of the project focuses on implementation and testing of the NNs.

Task	Allocated Time	Target Completion	Status
SUMMER INTERNSHIP – Beginning June 1st			
RNN Implementation	4 weeks	Early June	TBD
LSTM Implementation	3 weeks	Mid-June	TBD
BLSTM Implementation	3 weeks	Mid July	TBD
Model Enhancements	1 week	Early August	TBD
Testing, Assessment and Writing of Thesis	3 weeks	Late August	TBD
Thesis Submission	1 day	September	TBD
Interview and Assessment	1 day	Mid-September	TBD

SUCCESS CRITERIA

The success criteria of this project is to show the trade off between traditional wireless packet detection algorithms with cutting edge deep learning approaches, to enhance wireless networks. By doing so I will:

- Identify what deep learning architecture best suits this type of problem, by producing a low complexity and robust model
- Show how new technologies outperform existing methodologies
- Produce a model that will be integrated into the firmware of Cypress Semiconductor chips and be used on millions of devices across the world

APPENDIX C

RESEARCH LOG

TABLE OF CONTENTS

INTRODUCTION 2

WHY I SELECTED THESE PAPERS 2

HOW I USED THIS LITERATURE 2

COMPARISON OF IMPLEMENTATIONS 2

THE VALUE OF CONTINUED READING OF LITERATURE 3

IMPACTS CONTINUED READING HAD ON THE PROJECT 3

INTRODUCTION

This document outlines the importance of reading literature and understanding different approaches in great detail.

WHY I SELECTED THESE PAPERS

I selected these papers as they cover all aspects of the research I wanted to explore. From one standpoint I looked at lots of different wireless communication and wireless network papers to understand how detection occurs using correlation and how it could be improved. I also then looked at things from a deep learning perspective and this gave me a great foundation in what architectures I wanted to use, and I thoroughly understood them before impending.

Selecting such papers and reading about deep learning and its applications to everyday use, I became more and more in line with how things worked.

HOW I USED THIS LITERATURE

I used Mendely throughout the course of this project to store and track all of the research papers I have read. I used this literature to my gain, as I was able to understand thoroughly new and different aspects I wasn't exposed to before. Each new paper brought something new to the project and how I could adapt it to ensure the project became more of a success.

COMPARISON OF IMPLEMENTATIONS

As the approach of comparing deep learning to detection algorithms isn't widely researched yet, I built upon other ideas including traditional methods and from there was able to formulate a plan on how to first generate my datasets but then also see an improved performance.

Essentially, the main element of this paper and why it has been successful is the use of neural networks with memory which is explained throughout the project aids in the detection of packets using the preamble sequence.

THE VALUE OF CONTINUED READING OF LITERATURE

I love to read, and find reading the only way to thoroughly understand something. I take great pleasure in reading something and implementing it into everyday life. In this project the initial literature review was the propeller to kickstart the project as, thankfully I read and put a lot of research into different topics which paid significant dividends in the end.

IMPACTS CONTINUED READING HAD ON THE PROJECT

The value of the literature review, and the reading that came after it were invaluable and without taking the time to do the literature review I would not have been able to finish this project to any decent standard.

However, as I continue this project continued reading will become more and more influential and determine the outcome of the entire project. Applying the means of what you learn to everyday life I feel makes you a better person. As the saying goes, “the day you stop learning is the day you die”

Research Log

Niall Lyons

August 31, 2020

1 Refernces In Final Paper

References

[1] Y. Mostofi and D. C. Cox, "Robust timing synchronization design in OFDM systems - Part II: high-mobility cases," in IEEE Transactions on Wireless Communications, vol. 6, no. 12, pp. 4340-4348, December 2007, doi: 10.1109/TWC.2007.05901.

[2] M. Chiani and M. G. Martini, "Practical frame synchronization for data with unknown distribution on AWGN channels," in IEEE Communications Letters, vol. 9, no. 5, pp. 456-458, May 2005, doi: 10.1109/LCOMM.2005.1431170.

[3] Y. Mostofi and D. C. Cox, "A robust timing synchronization design in OFDM systems-part I: low-mobility cases," in IEEE Transactions on Wireless Communications, vol. 6, no. 12, pp. 4329-4339, December 2007, doi: 10.1109/TWC.2007.05900.

[4] IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Further Higher Data Rate Extension in the 2.4 GHz Band," in IEEE Std 802.11g-2003 (Amendment to IEEE Std 802.11, 1999 Edn. (Reaff 2003) as amended by IEEE Stds 802.11a-1999, 802.11b-1999, 802.11b-1999/Cor 1-2001, and 802.11d-2001) , vol., no., pp.1-104, 27 June 2003, doi: 10.1109/IEEESTD.2003.94282.

[5] "Model Complexity - an overview — ScienceDirect Topics", Sciencedirect.com, 2020. [Online]. Available: https://www.sciencedirect.com/topics/computer-science/model-complexity. [Accessed: 27- Aug- 2020].

[6] J. Park, M. Kim, D. Hwang and J. Han, "Analysis on Preamble Detection and False Alarm Probabilities of ICPC Method," 2015 IEEE 82nd Vehicular Technology Conference (VTC2015-Fall), Boston, MA, 2015, pp. 1-5, doi: 10.1109/VTCFall.2015.7390844.

[7] S. Kim, K. Joo and Y. Lim, "A delay-robust random access preamble detection algorithm for LTE system," 2012 IEEE Radio and Wireless Symposium, Santa Clara, CA, 2012, pp. 75-78, doi: 10.1109/RWS.2012.6175341.

[8] W. Chen, X. Wen, L. Yang, W. Li and T. A. Gulliver, "Dynamic preamble scheduling and detection for cooperative relay networks," Proceedings of 2011 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, Victoria, BC, 2011, pp. 131-134, doi: 10.1109/PACRIM.2011.6032880.

[9] S. Nagaraj, S. Khan, C. Schlegel, and M. Burnashev,"On preamble detection in packet-based wireless networks," in IEEE International Symposium onSpread Spectrum Techniques and Applications, 2006,pp. 476–480, doi: 10.1109/ISSSTA.2006.311817.

[10] N. Farsad and A. Goldsmith, "Neural Network Detection of Data Sequences in Communication Systems."

[11] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio,"How to Construct Deep Recurrent Neural Networks."

[12] S. Squartini, A. Hussain, and F. Piazza,"Preprocessing based solution for the vanishing gradient problem in recurrent neural networks," in Proceedings - IEEE International Symposium on Circuits and Systems, 2003, vol. 5, doi: 10.1109/iscas.2003.1206412

[13] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Comput., vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735

[14] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Comput., vol. 9, no. 8, pp. 1735– 1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[15] A. Graves, N. Beringer, and J. Schmidhuber, "Rapid Retraining on Speech Data with LSTM Recurrent Networks Instituto Dalle Molle di studi sull' intelligenza artificiale Galleria 2 CH-6900 Manno, Switzerland Rapid Retraining on Speech Data with LSTM Recurrent Networks," 2005

[16] Y. Lu, Y. Shi, and J. Yang, "A new method for semantic consistency verification of aviation radiotelephony communication based on LSTMRNN," in International Conference on Digital Signal Processing, DSP, Jul. 2016, vol. 0, pp. 422–426, doi:10.1109/ICDSP.2016.7868592.

[17] A. Graves and J. Schmidhuber, "Framewise Phoneme Classification with Bidirectional LSTM Networks."

[18] F. Belabed and R. Bouallegue, "A comparative Analysis of Machine Learning Classification Approaches for Fountain Data Estimation in Wireless Sensor Networks," 2019 15th International Wireless Communications & Mobile

Computing Conference (IWCMC), Tangier, Morocco, 2019, pp. 1251-1254, doi: 10.1109/IWCMC.2019.8766690.

[19] H. Huang et al., "Deep Learning for Physical-Layer 5G Wireless Techniques: Opportunities, Challenges and Solutions," in IEEE Wireless Communications, vol. 27, no. 1, pp. 214-222, February 2020, doi: 10.1109/MWC.2019.1900027.

[20] F. Belabed and R. Bouallegue, "A comparative Analysis of Machine Learning Classification Approaches for Fountain Data Estimation in Wireless Sensor Networks," 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC), Tangier, Morocco, 2019, pp. 1251-1254, doi: 10.1109/IWCMC.2019.8766690.

[21] H. Huang et al., "Deep Learning for Physical-Layer 5G Wireless Techniques: Opportunities, Challenges and Solutions," in IEEE Wireless Communications, vol. 27, no. 1, pp. 214-222, February 2020, doi: 10.1109/MWC.2019.1900027.

[22] H. Sun, A. O. Kaya, M. Macdonald, H. Viswanathan and M. Hong, "Deep Learning Based Preamble Detection and TOA Estimation," 2019 IEEE Global Communications Conference (GLOBECOM), Waikoloa, HI, USA, 2019, pp. 1-6, doi: 10.1109/GLOBECOM38437.2019.9013265.

[23] H. Apaydin, H. Feizi, M. T. Sattari, M. S. Colak, S. Shamshirband, and K.-W. Chau, "Comparative Analysis of Recurrent Neural Network Architectures for Reservoir Inflow Forecasting," Water, vol. 12, no. 5, p. 1500, May 2020.

[24] [3]Web.stanford.edu, 2020. [Online]. Available: https://web.stanford.edu/ lmackey/stats300a/doc/stats300a-fall15-lecture13.pdf. [Accessed: 27- Aug- 2020].

[25] "Receiver-Operating Characteristic Analysis for Evaluating Diagnostic Tests and Predictive Models," Circulation, 06-Feb-2007. [Online]. Available: https://www.ahajournals.org/doi/10.1161/CIRCULATIONAHA.105.594929. [Accessed: 27-Aug-2020].

2 References Not In Final Paper

References

[1] S. Hassan, A. Irfan, A. Mirza and I. Siddiqi, "Cursive Handwritten Text Recognition using Bi-Directional LSTMs: A Case Study on Urdu Handwriting," 2019 International Conference on Deep Learning and Machine Learning in Emerging Applications (Deep-ML), Istanbul, Turkey, 2019, pp. 67-72, doi: 10.1109/Deep-ML.2019.00021.

[2] X. Li, Y. He, Y. Yang, Y. Hong and X. Jing, "LSTM based Human Activity Classification on Radar Range Profile," 2019 IEEE International Conference on Computational Electromagnetics (ICCEM), Shanghai, China, 2019, pp. 1-2, doi: 10.1109/COMPEM.2019.8779144.

[3] M. Roopak, G. Yun Tian and J. Chambers, "Deep Learning Models for Cyber Security in IoT Networks," 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 2019, pp. 0452-0457, doi: 10.1109/CCWC.2019.8666588.

[4] W. Chen, J. Sun and C. Gao, "Improving Residue-Residue Contacts Prediction from Protein Sequences Using RNN-Based LSTM Network," 2019 International Conference on Machine Learning and Cybernetics (ICMLC), Kobe, Japan, 2019, pp. 1-7, doi: 10.1109/ICMLC48188.2019.8949207.

[5] X. Liu, C. Gong and Z. Xu, "Sequential detection for optical wireless scattering communication," in IEEE/OSA Journal of Optical Communications and Networking, vol. 9, no. 9, pp. D86-D95, Sept. 2017, doi: 10.1364/JOCN.9.000D86.

[6] Y. Li, H. Yin, X. Ji and B. Wu, "Design And Implementation Of Underwater Wireless Optical Communication System With High-Speed And Full-Duplex Using Blue/Green Light," 2018 10th International Conference on Communication Software and Networks (ICCSN), Chengdu, 2018, pp. 99-103, doi: 10.1109/ICCSN.2018.8488232.

[7] H. Zheng, F. Lin, X. Feng and Y. Chen, "A Hybrid Deep Learning Model With Attention-Based Conv-LSTM Networks for Short-Term Traffic Flow Prediction," in IEEE Transactions on Intelligent Transportation Systems, doi: 10.1109/TITS.2020.2997352.

Appendix D

PROJECT DESIGN AND IMPLEMENTATION

TABLE OF CONTENTS

INTRODUCTION **2**

PROBLEM DEFINITION **2**
- PROBLEM STATEMENT 2
- IDEAL PROBLEM SOLUTION 2

RESEARCH **3**
- RESEARCH METHODOLOGIES 3
- RESEARCH FINDINGS 3

DATA **4**
- DATASET GENERATION 6
  - BINARY SYMMETRIC CHANNEL (BSC) 6
  - ADDITIVE WHITE GAUSSIAN NOISE (AWGN) CHANNEL 6
  - MULTIPATH CHANNELS 6

PROJECT ARCHITECTURE **7**

MODEL DESIGN AND ARCHITECTURE 7
- DEEP LEARNING TECHNIQUES 8
  - GLOBAL MAX POOLING 8
  - GLOBAL AVERAGE POOLING 8
- RNN ARCHITECTURES 8
- LSTM ARCHITECTURES 9
- BLSTM ARCHITECTURES 12

MODEL TRAINING AND EVALUATION METRICS **12**
- MODEL COMPILATION 12
- MODEL EVALUATION METRICS 13
  - NEYMAN PEARSON (NP) CURVES 13
  - PRECISION AND ACCURACY 14
  - RECALL 15
  - RECEIVER OPERATING CHARACTERISTIC (ROC) CURVES & AREA UNDER CURVE (AUC) 15

CONCLUSION **16**

INTRODUCTION

This document, outlines the processes involved in the successful outcome of this project and the approaches made throughout.

PROBLEM DEFINITION

PROBLEM STATEMENT

In 802.11b wireless packets the preamble of the packet is a bit sequence that allows the receiver to obtain time frequency synchronization and channel estimation for each payload transmission. A preamble has various different attributes, however in this case it will be used for packet detection. In a wireless network if packets are missed, the performance of the network will decrease, therefore detecting when a preamble is present is key. On the contrary, falsely declaring that a packet is present will also reduce the performance of a network causing unnecessary delay increasing unneeded signal processing using extra computation.

The application of deep learning in industry has increased rapidly in the last number of years showing that it can have a vast impact on the performance of companies by reducing errors, identifying new trends and looking at new methodologies in certain domains. It is widely used in speech recognition, image processing, natural language processing and machine translation. It has also become an increasingly popular research topic in communications.

By applying deep learning to the detection of 802.11b wireless packets can it be proven that these methods are more robust than traditional packet detectors. In theory, what is the trade off between deep learning approaches and traditional approaches of detection? The central question to be investigated is:

> *"Is there a tradeoff between traditional preamble detection techniques, against deep learning architectures in 802.11b wireless packet detection?"*

IDEAL PROBLEM SOLUTION

An ideal solution to this problem, is to find an appropriate deep learning architecture

that has a high percentage accuracy of the detection of wireless packets. The model must be robust, scalable and must perform well without any prior knowledge of underlying channel models.

RESEARCH

RESEARCH METHODOLOGIES

As defined in the literature review in appendix A, when in the research section of this project a multi vocal approach was taken. To this front, peer reviewed and non peer reviewed papers and articles were considered. The reason behind this approach ensured that a wide range of topics and elements of wireless communications were examined. Wireless communications is a widely researched area, however during the initial literature review it was found that there were small numbers of peer-reviewed and non peer reviewed papers on the specific topic of "Detection algorithms for packet based wireless networks using deep learning". However, elements of the traditional methods of wireless packet detection gave a very in-depth and good understanding of the area. With this, the application of neural networks (NN) with memory were looked at. WIth the lengths of preambles varying, NN's with memory was deemed an interesting topic to be looked at further. Deep research was carried out on the architectures and complexities of three specific NN architectures namely, recurrent neural networks (RNN), long short term memory (LSTM) and bi-directional LSTM (BLSTM).

RESEARCH FINDINGS

The literature review findings proved to be a major factor in the final results of the project, outlined in Appendix E. RNN's were initially looked at as the most efficient architecture for this problem. A feed forward NN with internal memory, therefore considering both the current input and output previously learned. However, it was found that RNN's struggle processing long sequences, when using the ReLU activation function. With this it was ensured when building the architecture activation functions were tested to prove a best fit for this specific problem as premale sequences vary. It was also found that a disadvantage of this architecture covers a vanishing gradient and exploding gradient problem, when gradient computation involving recurrent multiplication on each cell results in a number less than 1 (it will vanish) and greater than 1 (it will explode).

It was found however, that a modified version of the RNN architecture, an LSTM resolves the gradient problem resulting in a more robust and efficient model. An LSTM utilizes

four specific gates that control the flow of the model. Namely, an input gate, input modulation gate, forget gate and output gate. The element of difference in this architecture and advantage over RNN's is that the recurrent multiplication of each cell is multiplied by the same value over and over resulting in the gradient problems. In this architecture the forget gate varies ths value each time which is being recurrently multiplied, thus avoiding this problem. However, with this architecture preamble information will only be preserved from the past. Through this research it was found that a re-engineered architecture, namely a BLSTM could be used to understand the context of the preamble and preserve the information from both the past and the future by running input both ways using two LSTM layers.

BLSTMs were found in most peer reviewed papers to improve LSTM model performance in varying classification problems. Using this model, input is presented in forward and backward states to two separate LSTM networks, that are both connected to the same output layer. By preserving information from the future and using two hidden states we can successfully preserve preamble information from both the past and future and compute the output sequence. This will serve as an advantage as the length of 802.11b wireless packets can be extensively long.

The assumptions taken from the literature review were that the BLSTM model would perform well in the detection of wireless packets. However, all architectures were looked at in the process of this project. It was found that this assumption was true using both multipath and additive gaussian white noise data. However, using a binary symmetric channel dataset the RNN model proved to outperform both LSTM and BLSTM data. Although, this dataset consisted of randomly flipped sequences based on an error probability. In industry and real world applications this would not be the case of how a preamble sequence is properly received, however is used extensively in research of such areas.

DATA

As outlined in the final paper, 802.11b wireless packets use direct sequence spread spectrum (DSSS), to reduce overall signal interference. This method spreads a signal across a large bandwidth, thus creating a low power density across various spectrums. Wireless packet preambles, then represent a string of bits represented by logic 1 and 0 data bits. DSSS are represented by code sequences, creating pseudo random sequences that aid in the protection of noise and interference as well as different packets using the same preamble sequence. There are many different code sequences used, namely

- Maximum length sequences
- Gold codes
- Orthogonal codes

This paper utilises maximum length sequences (m-sequences) due to its autocorrelation properties which ensures avoiding a false synchronisation problem. For large sequences, m-codes are the go-to and the main form of code sequences employed in industry and research. Therefore, this approach was taken.

The construction of maximum length sequences is explained in the final paper. Linear feedback time register (LFSR) structures are utilised here to create the sequence codes. In industry, there are two main types of LFSR structures used, namely Galois LFSR and Fibonacci LFSR, although the Galois approach is preferred as its structures run on a faster frequency. The fibonacci approach uses XOR between several bits using multiple XOR gates. In contrast to Galois structures using only two XOR gates with the delay path minimised.

As outlined in equation (1) of the final paper, the sequence is generated outlines that a sequence is made using a generator polynomial and its coefficients. This generator polynomial must be a primitive polynomial. s also outlined in the paper the sequence length is calculated based on the degree (L) of the polynomial, with the sequence length being in the form,

$$N = 2^L - 1$$

For the simulations in this project the primitive polynomial,

$$X^5 + x^2 + 1$$

Was used resulting in a sequence length of 31. As computationally this is effective for the purpose of this research.

As outlined in the final paper, to test a wide range of situations, three datasets were generated to formulate a comprehensive set of results gauging contrasting elements of noise and channel capacities. Each dataset generated contains an input x and the corresponding label y. The input x represents the received signal sequence. The label y represents the binary representation of signal detection. Where 1 represents signal detected and is computed by the reference polynomial and reference initial state. When y is equal to 0, two different states are represented. Namely, a reference polynomial with

a random initial state and also, a random polynomial and a random initial state.

DATASET GENERATION

1) BINARY SYMMETRIC CHANNEL (BSC)

BSC channels accept an input sequence, in this case the m-sequence generated by the galois LFSR structure and randomly flips the sequence based on a given error probability. The error probability represents the probability of receiving an incorrect bit, and a range of error probabilities were looked at that can be seen in Appendix E.

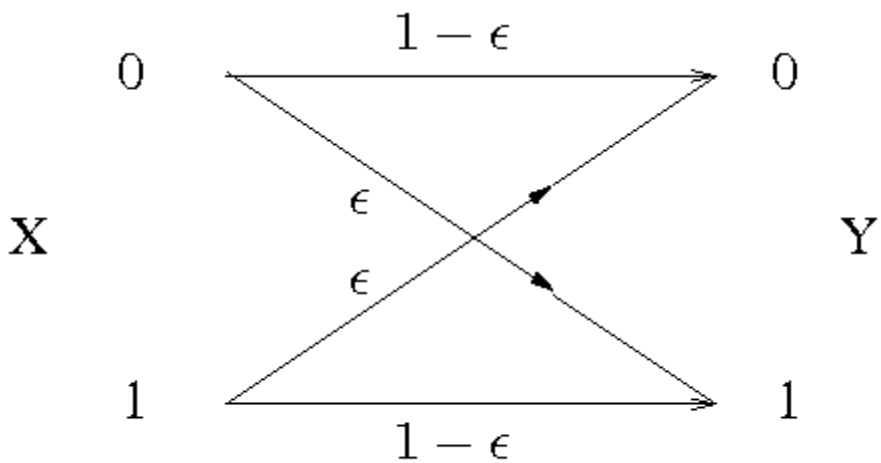


Fig. 1 The binary symmetric channel

In Fig.1 ε represents the error probability rate that bits will be flipped at.

2) ADDITIVE WHITE GAUSSIAN NOISE (AWGN) CHANNEL

The second dataset generated adds AWGN to the m-sequences. The power of noise to the sequences were generated by varying values of signal to noise ratio. As shown in Appendix E, -15 SNR db to 10 SNR dB were addressed. -15 SNR dB represents a large amount of noise in the signal, where 10 SNR dB represents little noise in the signal. Looking at these varying values gave a scope on what was the correct amount of SNR dB to use, and it was concluded that -5 SNR dB is the most relatively common power of noise used in industrial research and a decision was made to use this when generating the final dataset.

3) MULTIPATH CHANNELS

In wireless channels parameters randomly change over time. In a wireless signal

the same packet is sent multiple times. Each one with different arrival times and different paths. Resulting in the receiver, receiving multiple packets at multiple different strengths and phase angles. From this, it can be said that a multipath channel changes randomly in respect to time. The output of the channel is expressed as the convolution of the channel input and the normalized channel. How this is generated can be seen in the code snippet below:

```
current_channel = [0 1 (rand(1,6)-1/2)/6];
channel_normalised = current_channel/sqrt(sum(abs(current_channel).^2));
channel_out = conv(channel_normalised, channel_in);
```

AWGN is then added to the output channel to generate the whole dataset. Again -5 SNR dB is set.

PROJECT ARCHITECTURE

Fig. 2 below, outlines the high level architecture of the developed system based on the dataset that is given. It outlines, when a wireless packet preamble is received, and based on the specific dataset what channel the preamble is in. The resulting channel is then passed through the traditional and deep learning classification model, ranging from RNN, LSTM and BLSTM.

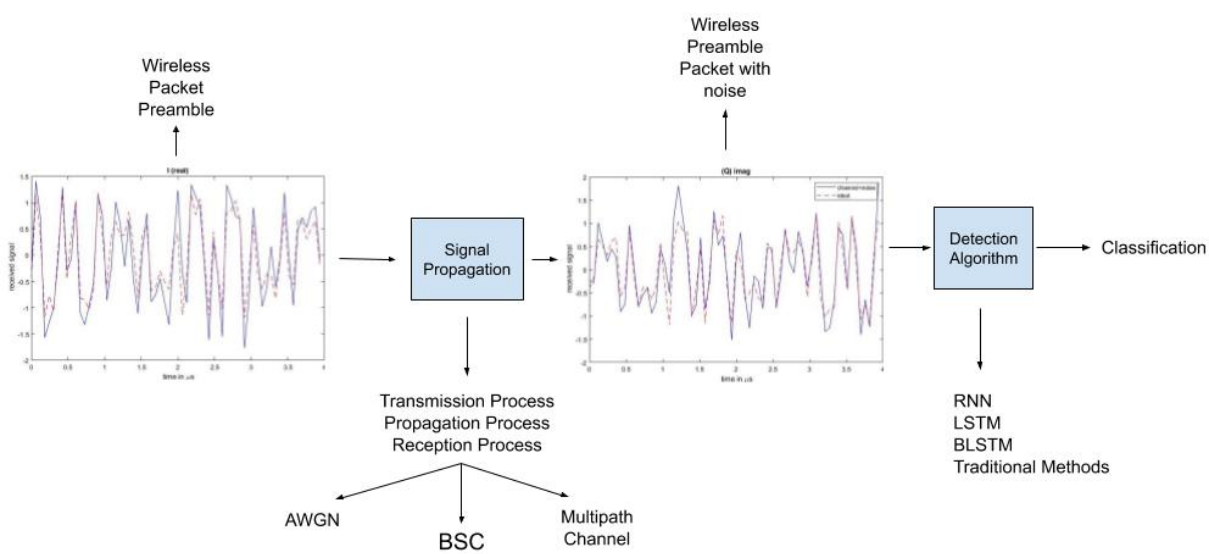


Fig. 2. High Level Architecture

MODEL DESIGN AND ARCHITECTURE

As outlined in the final paper, table I - III represent the final models and implemented architectures. However many more were also looked at. This section breaks down different summaries of models and how they were applied.

DEEP LEARNING TECHNIQUES

Initially, different pooling techniques were trialled. Pooling is essentially downscaling the given input that is obtained from the previous layer. This operation is done using hardcoded tensor operations such as max, global max, average and global average. In this project, global max and global average pooling were specifically looked at.

1) GLOBAL MAX POOLING

   In this case, the pool size within the pooling layer is equal to the input size and the max of the entire pool is the output value.

2) GLOBAL AVERAGE POOLING

   In this case, the same process applies as above, however the average of the entire pool is the output value.

In all the models tested dense layers are used with the sigmoid activation function. This layer reduces the dimensionality of the output space. The operation implemented by dense represents

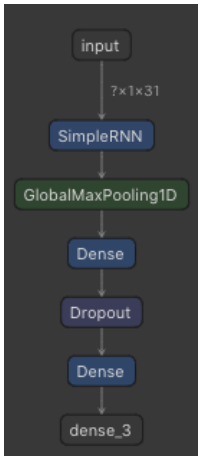
$$Output = activation(dot(input, kernel) + bias)$$

Where activation, in our case sigmoid is the activation function, the kernel represents the weights matrix calculated by the layer and bias represents the bias matrix calculated by the error. This layer is used throughout all models developed in the three architectures.

RNN ARCHITECTURES

A simple RNN model architecture was developed as shown in Fig. 3. This shows a simple RNN layer utilising global max pooling with a dense layer with an output of 16. A dropout rate of 0.2, with the total training parameters of 2,482.

The results of this model can be seen in Appendix E.



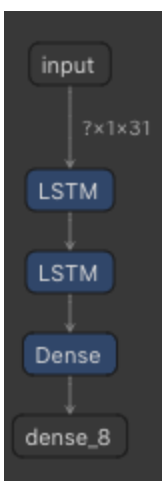

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
simple_rnn_1 (SimpleRNN)     (None, 1, 31)             1953
_________________________________________________________________
global_max_pooling1d_1 (Glob (None, 31)                0
_________________________________________________________________
dense_2 (Dense)              (None, 16)                512
_________________________________________________________________
dropout_1 (Dropout)          (None, 16)                0
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 17
=================================================================
Total params: 2,482
Trainable params: 2,482
Non-trainable params: 0
_________________________________________________________________
```

Fig. 3 RNN Architecture used across all datasets

LSTM ARCHITECTURES

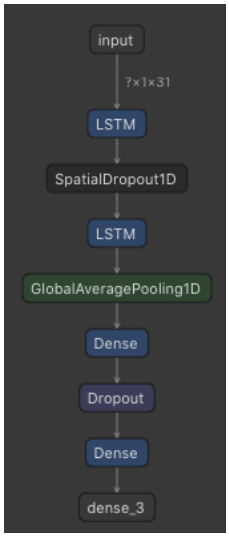
Five different LSTM architectures were tested as seen below in Fig. 4 - Fig. 9 each with varying layers using spatial dropout, dropout, average and max global pooling and dense layers using a sigmoid activation function.

The results of this model can be seen in Appendix E.



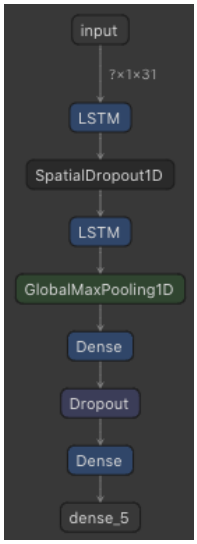

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_11 (LSTM)               (None, 1, 31)             7812
_________________________________________________________________
lstm_12 (LSTM)               (None, 31)                7812
_________________________________________________________________
dense_8 (Dense)              (None, 1)                 32
=================================================================
Total params: 15,656
Trainable params: 15,656
Non-trainable params: 0
```

Fig. 4. Two Layer LSTM



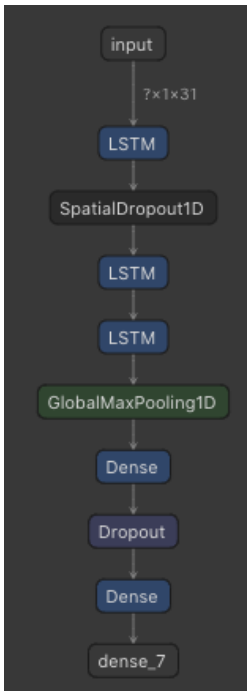

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_2 (LSTM)                (None, 1, 31)             7812
_________________________________________________________________
spatial_dropout1d_3 (Spatial (None, 1, 31)             0
_________________________________________________________________
lstm_3 (LSTM)                (None, 1, 31)             7812
_________________________________________________________________
global_average_pooling1d_1 ( (None, 31)                0
_________________________________________________________________
dense_2 (Dense)              (None, 16)                512
_________________________________________________________________
dropout_1 (Dropout)          (None, 16)                0
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 17
=================================================================
Total params: 16,153
Trainable params: 16,153
Non-trainable params: 0
```

Fig. 5. Two layer LSTM, using spatial dropout and global average pooling



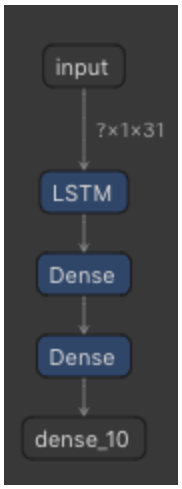

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_6 (LSTM)                (None, 1, 31)             7812
_________________________________________________________________
spatial_dropout1d_5 (Spatial (None, 1, 31)             0
_________________________________________________________________
lstm_7 (LSTM)                (None, 1, 31)             7812
_________________________________________________________________
global_max_pooling1d (Global (None, 31)                0
_________________________________________________________________
dense_4 (Dense)              (None, 16)                512
_________________________________________________________________
dropout_2 (Dropout)          (None, 16)                0
_________________________________________________________________
dense_5 (Dense)              (None, 1)                 17
=================================================================
Total params: 16,153
Trainable params: 16,153
Non-trainable params: 0
```

Fig. 6. Two layer LSTM, using spatial dropout and global max pooling




```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_8 (LSTM)                (None, 1, 31)             7812
_________________________________________________________________
spatial_dropout1d_6 (Spatial (None, 1, 31)             0
_________________________________________________________________
lstm_9 (LSTM)                (None, 1, 31)             7812
_________________________________________________________________
lstm_10 (LSTM)               (None, 1, 31)             7812
_________________________________________________________________
global_max_pooling1d_1 (Glob (None, 31)                0
_________________________________________________________________
dense_6 (Dense)              (None, 16)                512
_________________________________________________________________
dropout_3 (Dropout)          (None, 16)                0
_________________________________________________________________
dense_7 (Dense)              (None, 1)                 17
=================================================================
Total params: 23,965
Trainable params: 23,965
Non-trainable params: 0
```

Fig. 7. Stacked LSTM, using spatial dropout and global max pooling



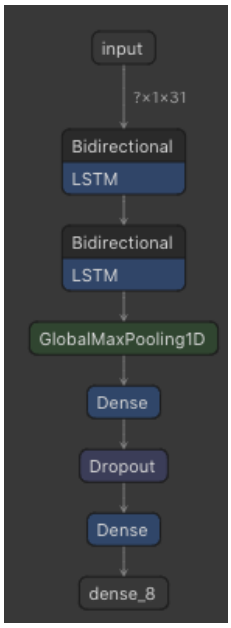

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_13 (LSTM)               (None, 1, 31)             7812
_________________________________________________________________
dense_9 (Dense)              (None, 1, 15)             480
_________________________________________________________________
dense_10 (Dense)             (None, 1, 1)              16
=================================================================
Total params: 8,308
Trainable params: 8,308
Non-trainable params: 0
```

Fig. 8. A low complexity LSTM with two dense layers

BLSTM ARCHITECTURES

Fig.9 outlines the BLSTM architecture built using two BLSTM layers and a global max pooling layer. Two dense layers are also used here reducing the dimensionality from 62 to 16, and 16 to 1 respectively. Total trainable parameters is 39,961.

The results of this model can be seen in Appendix E.




```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
bidirectional_10 (Bidirectio (None, 1, 62)             15624
_________________________________________________________________
bidirectional_11 (Bidirectio (None, 1, 62)             23312
_________________________________________________________________
global_max_pooling1d_6 (Glob (None, 62)                0
_________________________________________________________________
dense_7 (Dense)              (None, 16)                1008
_________________________________________________________________
dropout_2 (Dropout)          (None, 16)                0
_________________________________________________________________
dense_8 (Dense)              (None, 1)                 17
=================================================================
Total params: 39,961
Trainable params: 39,961
Non-trainable params: 0
```

Fig. 9. Two layer BLSTM using global max pooling

MODEL TRAINING AND EVALUATION METRICS

MODEL COMPILATION

As outlined in the as this is a binary classification problem, binary cross entropy is used as the loss function. This loss function evaluates the log probability of a packet being detected or not. This loss function effectively measures the effectiveness of the model learning.

An optimizer increases the learning speed of the model. The efficient adam optimizer is also used throughout, however the use of SGB optimizer was also looked at however this was deemed to have a lower performance rate. The benefits of using the adam optimizer for this type of preamble detection, are that it is computationally efficient, and has little memory requirements and is also very well suited to large data and sequences with

noise.

MODEL EVALUATION METRICS

NEYMAN PEARSON (NP) CURVES

NP curves are used to evaluate the trade off between traditional detection techniques and the trained deep learning models. False alarm rates are plotted against missed detection rates to outline the performance of the different methods used. If a packet is detected, where the prediction is wrong the error represents a false alam, since we mistakenly detected the wrong packet received. This is shown on the x axis. In contrast to not detecting a packet, when it is a packet represents missed detection shown on the y axis. With a probability threshold of, λ as it decreases or increases the curve becomes smaller or larger. As can be seen in Appendix E, the trained model curves are significantly inside the traditional method resulting in the level of detection being higher. The closer the models NP curve comes to the traditional curve shows that the level of detection is reducing. Fig. 10, outlines the miss detection and false alarm probabilities.

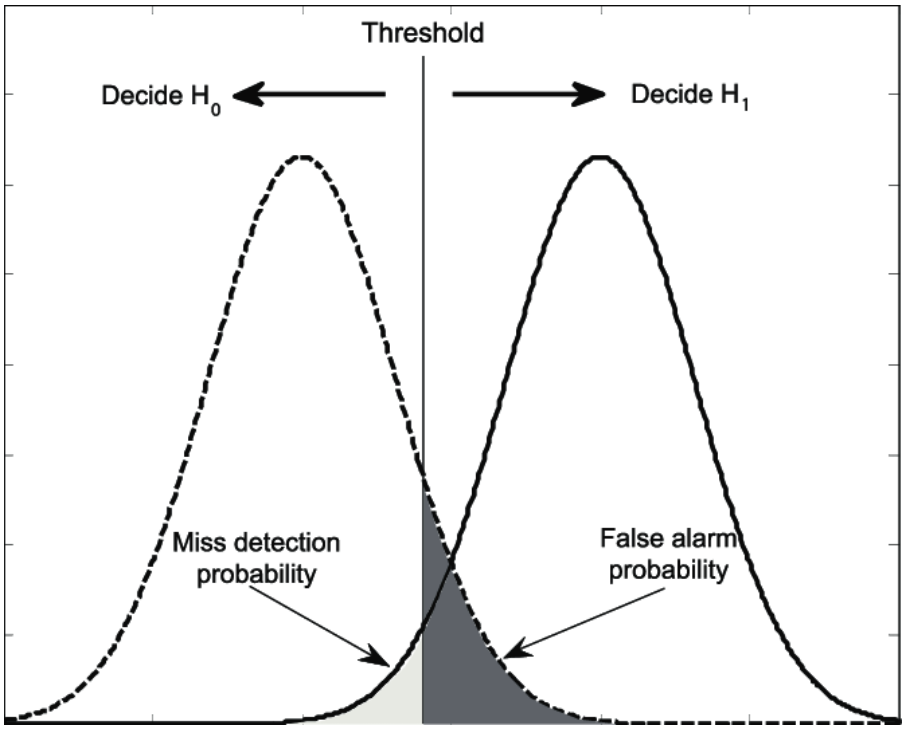


Fig. 10 Missed detection and False alarm probabilities based on a threshold

PRECISION AND ACCURACY

In binary classification problems, precision and accuracy are widely used in evaluating the performance of a model. They are calculated using four prior metrics that are calculated using a confusion matrix. Namely,

- True Positives (TP): Predicted positive and it's true. Here, the model predicted a packet is detected and that is true.
- False Positives (FP): Predicted positive and it's false. Here, the model predicted a packet is detected but it is actually wasnt.
- True negatives (TN): Predicted negative and it's true. Here, the model predicted a packet is not detected and it is actually wasnt.
- False Negatives (FN): Predicted negative and it's false. Here, the model predicted a packet is not detected but it actually was.

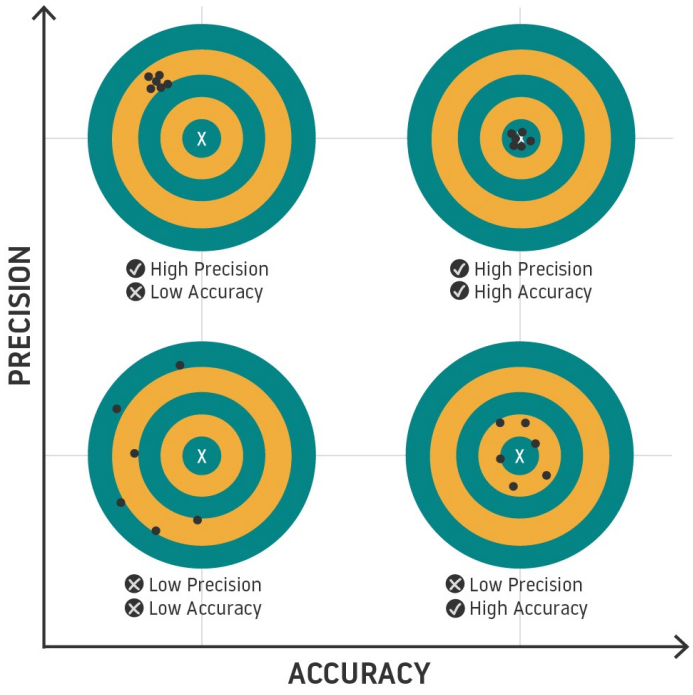


Fig. 11. Precision Vs Accuracy

As can be seen in Fig. 11, it is of upmost importance that the model has high values of precision and accuracy. Precision, evaluates the accuracy of positive predictions the

classifier makes. For example, as seen in Table IV in the final paper, the RNN model has a precision score of 0.97 and accuracy of 0.98. This specifies when a packet is correctly detected, the model is correct 96% of the time, while being 98% accurate. while also indicating a correct detection of 98% of packets. Precision is defined as,

$$Precision = \frac{TP}{TP + FP}$$

Although the accuracy metric is used, having a high accuracy does not ensure a high performing model. Precision and recall metrics along with ROC and AUC metrics give a wider overview of performance evaluation. Accuracy is defined as,

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

RECALL

Recall is another evaluation metric used along side precision and accuracy. This evaluates the ration of packets correctly identified by the classifier. In table IV of the final paper, recall is 0.98 indicating a correct detection of 98% of packets. Recall is defined by,

$$Recall = \frac{TP}{TP + FN}$$

RECEIVER OPERATING CHARACTERISTIC (ROC) CURVES & AREA UNDER CURVE (AUC)

A ROC curve represents the true positive rate plotted against the false positive rate, the amount of negative observations that are incorrectly classified. ROC curves are widely used today in real world machine learning projects and are used to great effect with AUC scores. An AUC score of 1, is the optimal score. Fig. 12, shows a range of ROC curves with specific AUC scores for clarity.

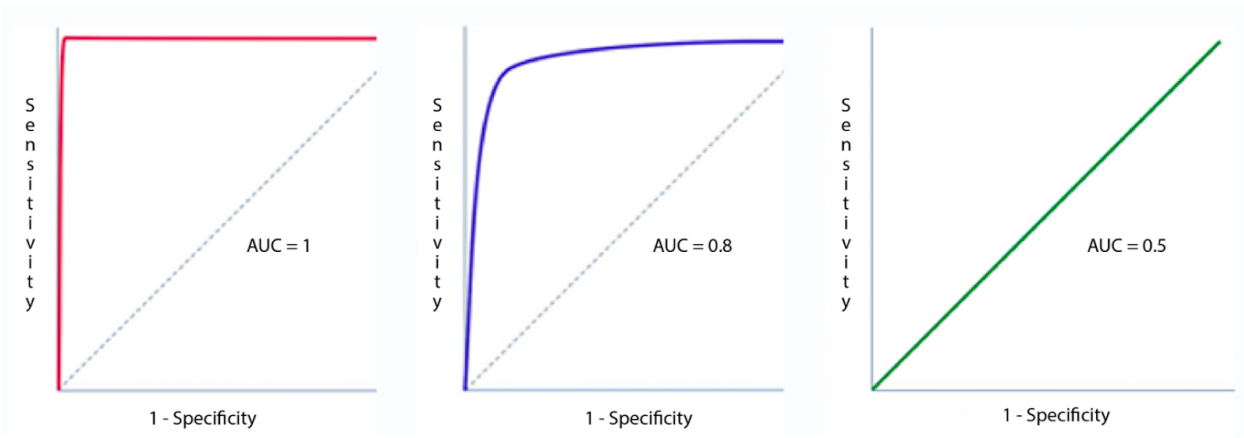


Fig. 12. Ranging ROC curves with AUC

CONCLUSION

To conclude, the extensive research in the initial literature review paved the way for the successful outcome of this project. The decision to generate data using m sequences using different datasets worked very well under all conditions. The assumptions made in most cases were correct and the findings of this project will now be brought forward to be used in industry. Which, was the goal of this project from the beginning.

APPENDIX E

TESTING AND RESULTS

TABLE OF CONTENTS

TABLE OF CONTENTS 1

INTRODUCTION 2

SETUP 2
- TRADITIONAL METHODS 2
- DEEP LEARNING 2

DATA 2

TRADITIONAL PACKET DETECTION METHODS 3
- BINARY SYMMETRIC CHANNEL (BSC) 3
- ADDITIVE WHITE GAUSSIAN NOISE (AWGN) CHANNEL 6
- MULTIPATH CHANNELS 9

DEEP LEARNING PACKET DETECTION MODELS 12
- RNN 12
  - BINARY SYMMETRIC CHANNEL (BSC) 12
  - ADDITIVE WHITE GAUSSIAN NOISE (AWGN) CHANNEL 12
  - MULTIPATH CHANNELS 13
  - RNN PERFORMANCE BREAKDOWN 13
- LSTM 14
  - BINARY SYMMETRIC CHANNEL (BSC) 14
  - ADDITIVE WHITE GAUSSIAN NOISE (AWGN) CHANNEL 15
  - MULTIPATH CHANNELS 15
  - LSTM PERFORMANCE BREAKDOWN 16
- BLSTM 17
  - BINARY SYMMETRIC CHANNEL (BSC) 17
  - ADDITIVE WHITE GAUSSIAN NOISE (AWGN) CHANNEL 17
  - MULTIPATH CHANNELS 18
  - BLSTM PERFORMANCE BREAKDOWN 19

EVALUATION AND ANALYSIS 19

CONCLUSION AND FUTURE PROSPECTS 23

INTRODUCTION

This document outlines setup, testing and results in the three generated datasets.

SETUP

TRADITIONAL METHODS

To implement the traditional approach to deep learning, MATLAB scripts were developed. These scripts can be found in Appendix F.

DEEP LEARNING

To implement the deep learning approaches, much more computational power was need in comparison to traditional methods in order to train the models. Models were trained on the Cypress Semiconductor GPU cluster. The specs of the cluster node used are,

- Nvidia GeForce RTX 2080
- 512GB RAM
- 1 TB SSD

Python was used for the development of the models. In particular, keras with a tensorflow backend. Many different machine learning libraries were also used, including numpy, pandas etc. All libraries used, and code produced can be seen in Appendix E.

DATA

As outlined in Appendix D, the data is generated using m-sequences. The following abbreviations used in the diagrams throughout this document illustrate the following,

- Ref Ref
    - Reference polynomial and reference initial state, label y = 1
- Ref Diff:
    - Reference polynomial and random initial state, label y = 0
- Diff Diff
    - Random polynomial and Random initial state, Label y = 0

TRADITIONAL PACKET DETECTION METHODS

BINARY SYMMETRIC CHANNEL (BSC)

Fig.1 - Fig. 7 illustrate the probability of detection under a range of thresholds based on the given error probability. Fig.1 shows an error probability of 0.2, which is equivalent and close to what is used in industry and therefore used as the error probability to build the training dataset for the deep learning models below. It shows that for ref, ref values at a threshold value of 0 to 0.25, there is 100% probability of detection. With a threshold of 0.95 threshold yields a 0% probability of detection. However, for ref, diff and diff, diff sequences at a threshold value of 0 there is 0.45% and 0.5% chance of detection. WIth a threshold value of 0.4 having 0% detection. This illustration is correct as with random polynomials and random initial states the detection level will decrease quicker.

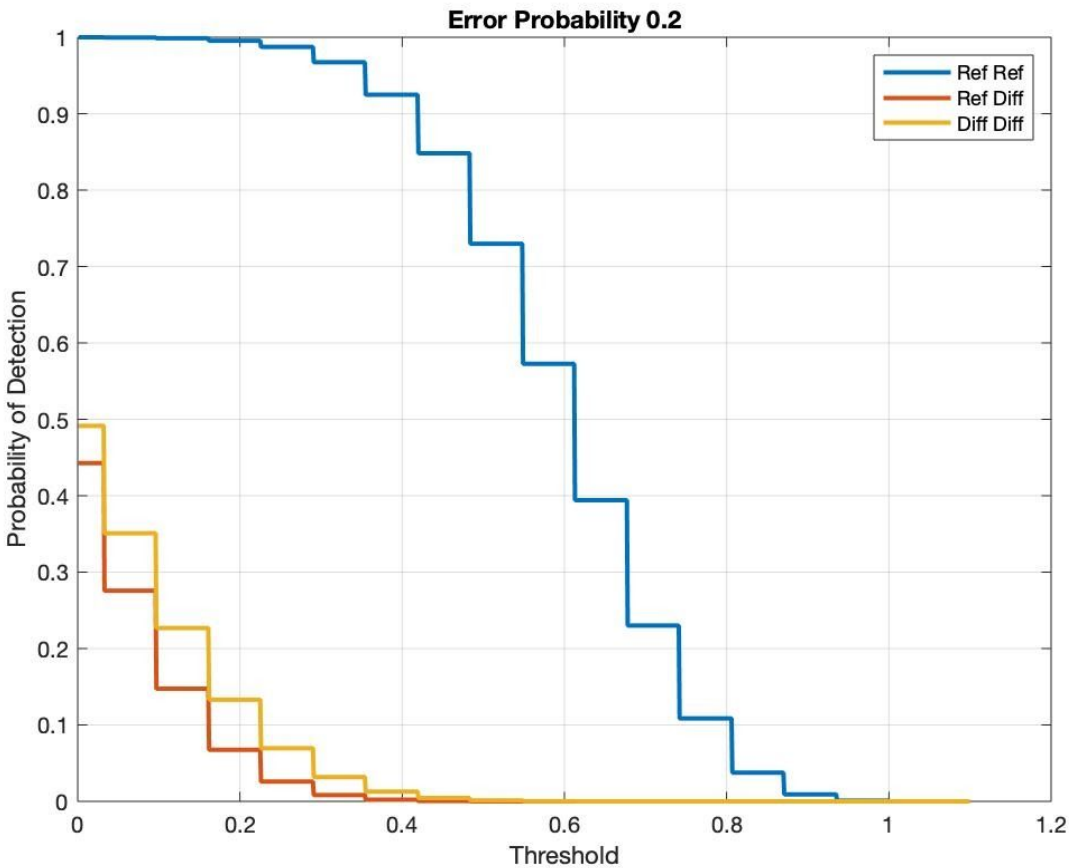


Fig. 1. Error Probability 0.2

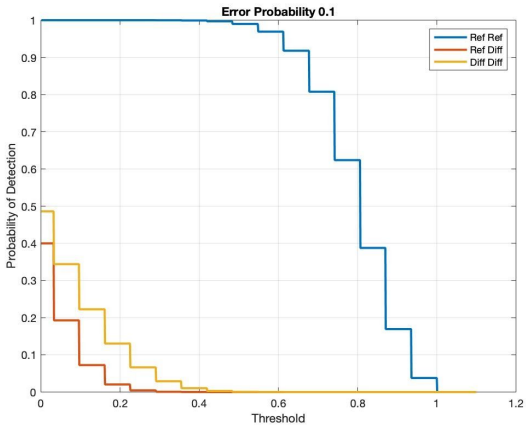


Fig. 2. Error Probability 0.1

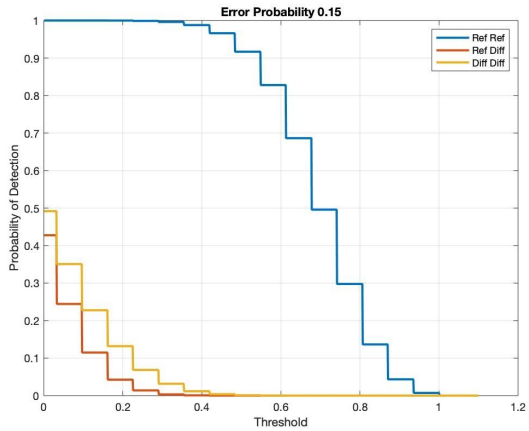


Fig. 3. Error Probability 0.15

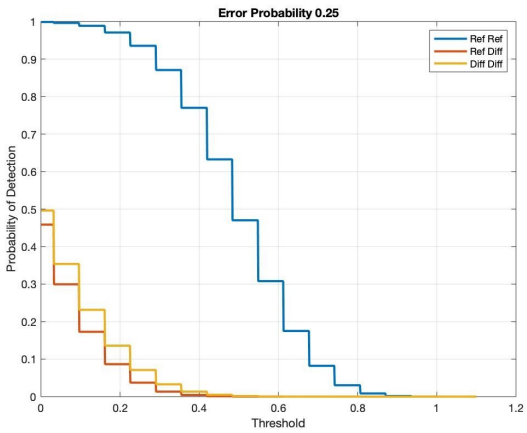


Fig. 4. Error Probability 0.25

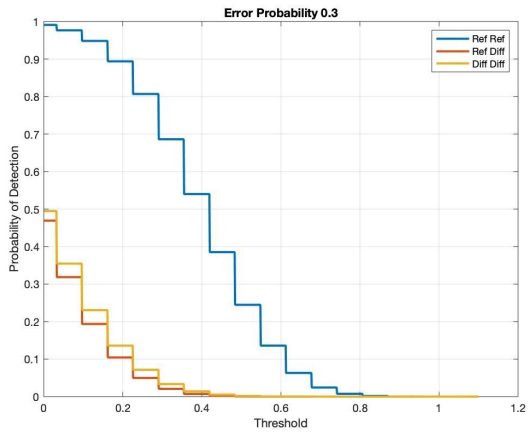


Fig. 5. Error Probability 0.3

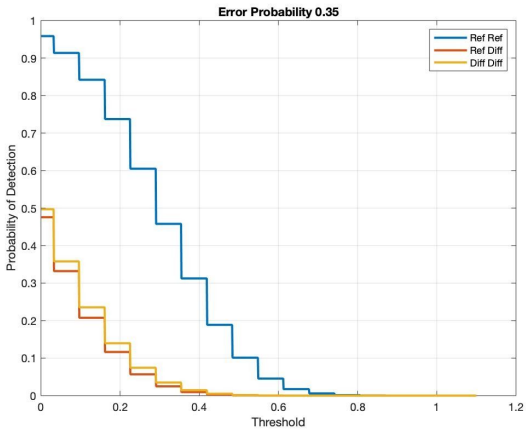


Fig. 6. Error Probability 0.35

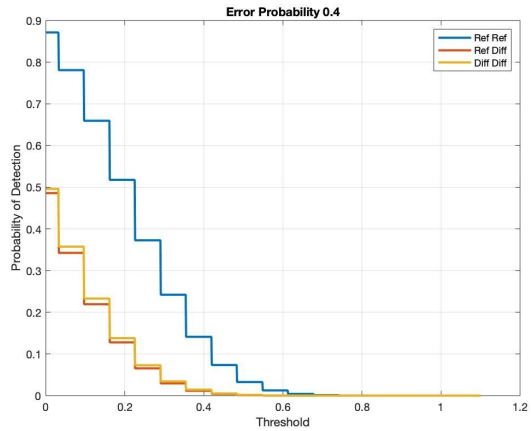


Fig. 7. Error Probability 0.4

Fig.2 - Fig.7 as expected illustrates that as the error probability increases, the probability of detection over varying thresholds for the ref, ref case decreases. However, the ref, diff and diff, diff case stay mostly the same throughout.

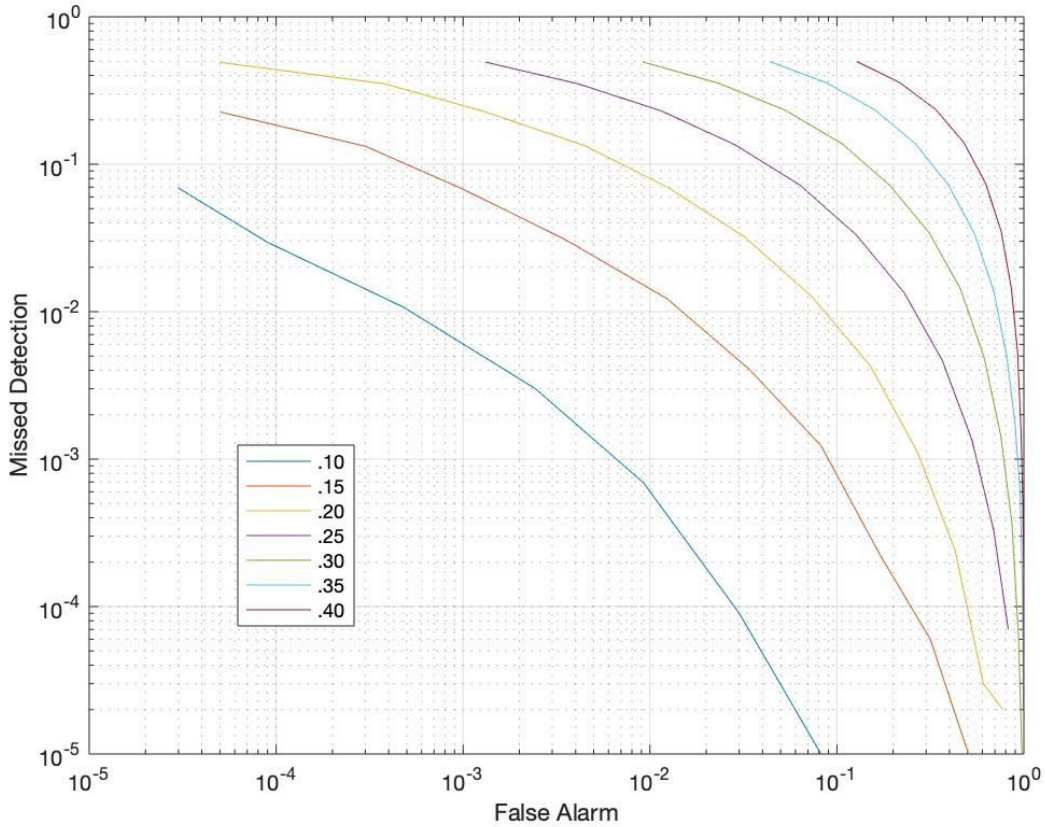


Fig. 8. NP Curve with varying error probabilities

Fig. 8 illustrates the NP curve at different values of error probability. It is seen that as the value of error probability is increasing the area of detection is decreasing, complementing the above figures.

ADDITIVE WHITE GAUSSIAN NOISE (AWGN) CHANNEL

Fig.9 - Fig. 13 illustrate the probability of detection under a range of thresholds based on the given SNR dB value. Fig. 9 shows an SNR dB value of -5, which is equivalent and close to what is used in industry and therefore used as the SNR dB to build the training dataset for the deep learning models below. It shows that for ref, ref values at a threshold value of 0 to 0.2, there is 100% probability of detection. With a threshold of 0.4 the probability of detection begins to decrease. Decreasing to 40% detection probability at a threshold of 1.2. However, for ref, diff and diff, diff sequences at a threshold value of 0 there is 0.45% and 0.5% chance of detection. With a threshold value of 0.75 having 0% detection.

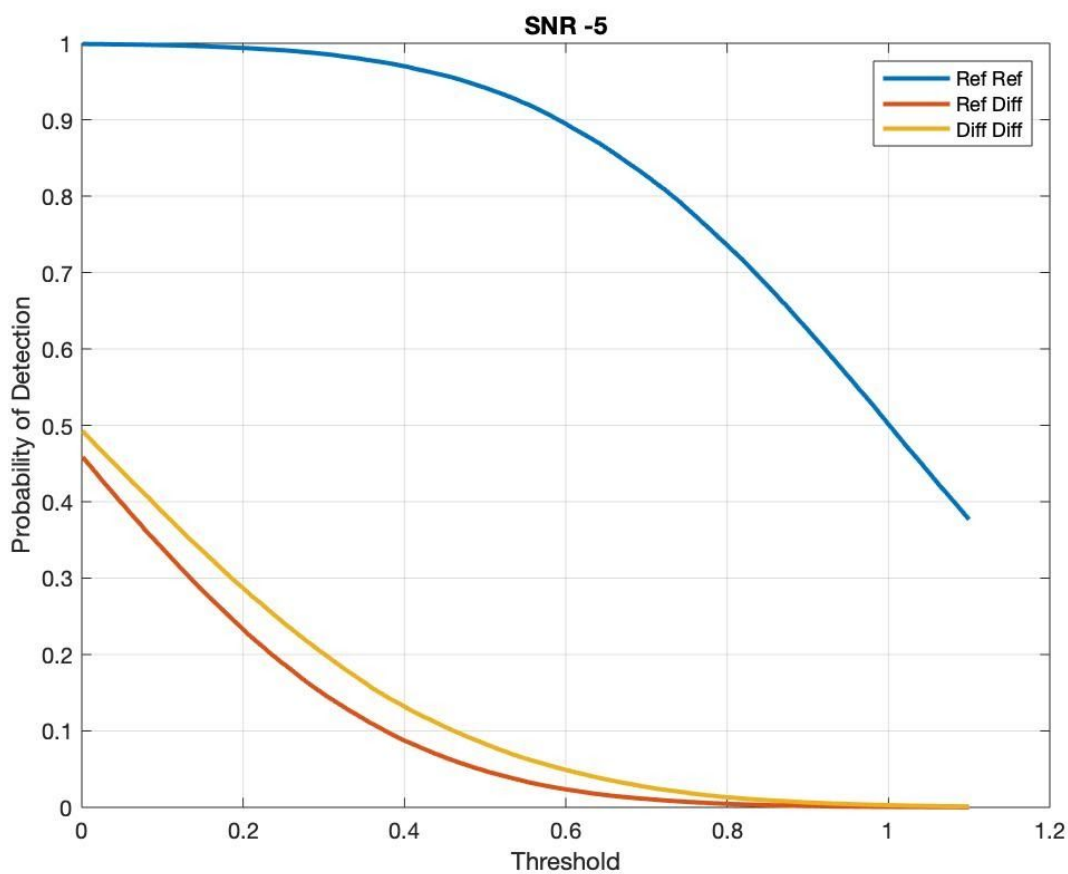


Fig. 9. SNR dB -5

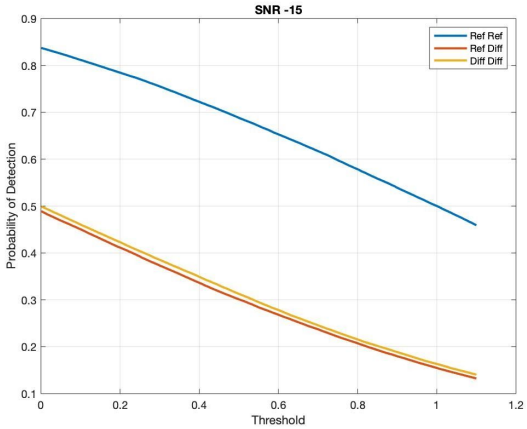


Fig. 10. SNR dB -1

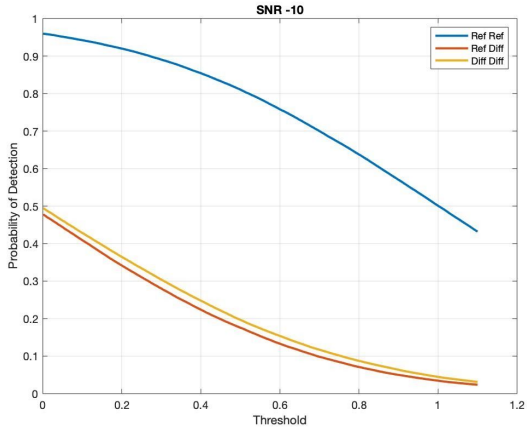


Fig. 11. SNR dB -10

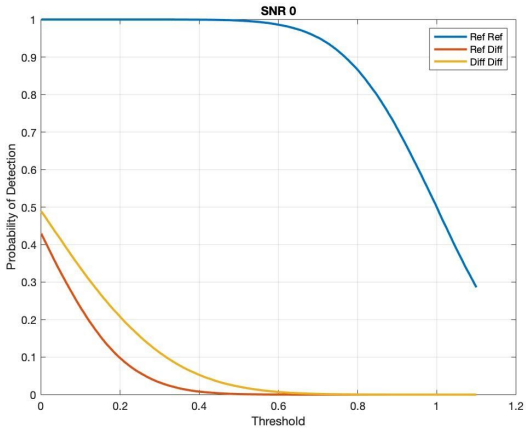


Fig. 12. SNR dB 0

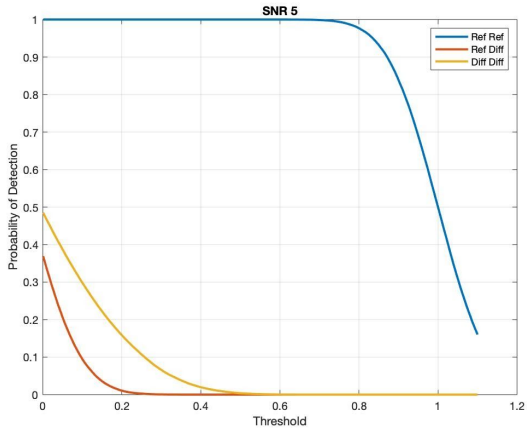


Fig. 13. SNR dB 5

As expected, as the SNR dB values increases toward one and over, the probability of detection increases for larger thresholds. As the value of SNR dB increases the less noise is in the signal. Interestingly, as the value of SNR db increases, the ref, diff and diff, diff cases decrease probability in lower thresholds. As the received signal, has multiple changes and interferences with a higher SNR dB.

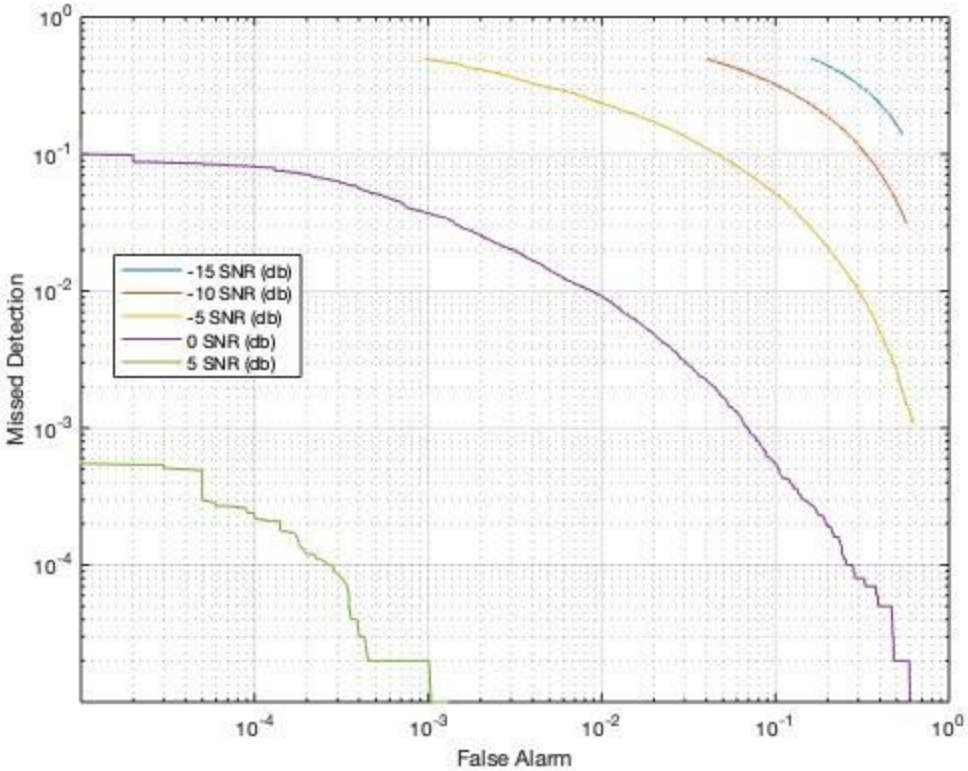


Fig. 14. NP Curve ranging from -15 SNR dB to 5 SNR -dB

Fig. 14 illustrates the NP curve at different values of SNR dB. It is seen that as the value of SNR dB increases the area of detection is increasing also, complementing the above figures. As -15 SNR dB is very high, the resulting area of detection reduces dramatically in relation to values -5 and 0 SNR dB.

MULTIPATH CHANNELS

Fig.15 - Fig. 19 illustrate the probability of detection under a range of thresholds based on multipath channels given different SNR dB values. Fig. 15 shows an SNR dB value of -5, which is equivalent and close to what is used in industry and therefore used as the SNR dB to build the training multipath dataset for the deep learning models below. It shows that for ref, ref and ref,diff values at a threshold value of 0 to 0.4, there is 100% probability of detection. However, ref, diff begins to dramatically fall in performance with ref, ref falling gradually however keeping a higher probability of detection over thresholds 0.4 to 1.2.

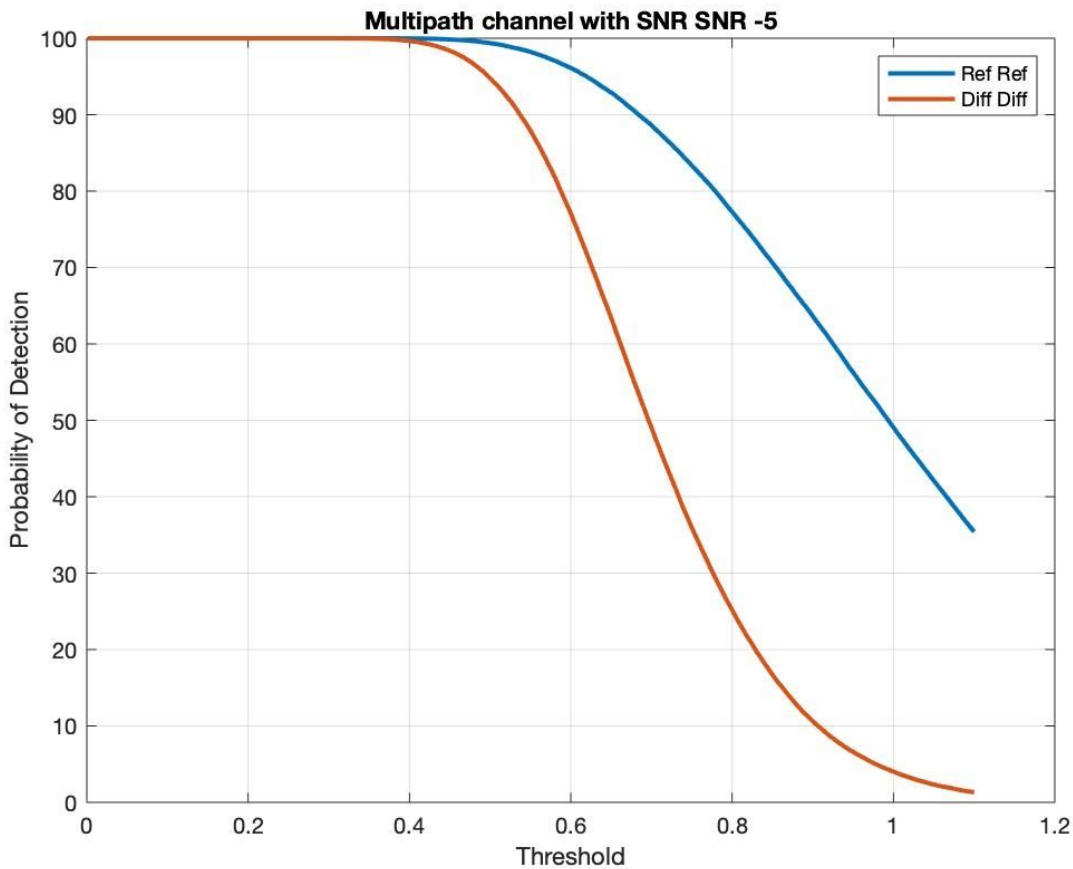


Fig. 15. Multipath Channel with SNR dB -5

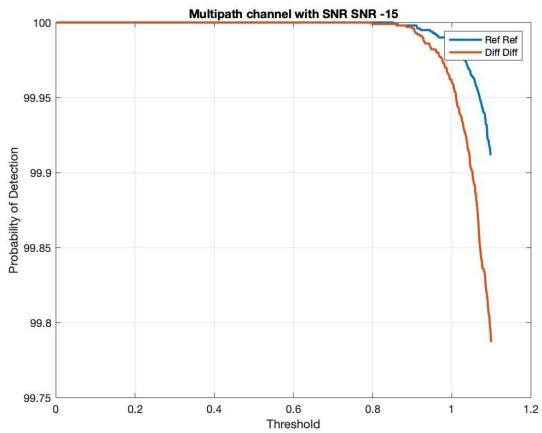


Fig. 16. Multipath SNR dB -15

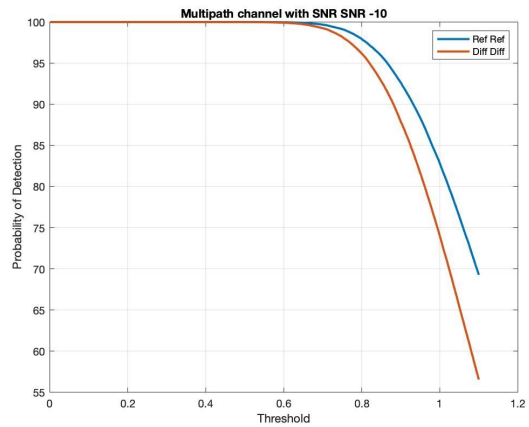


Fig. 17. Multipath SNR dB -10

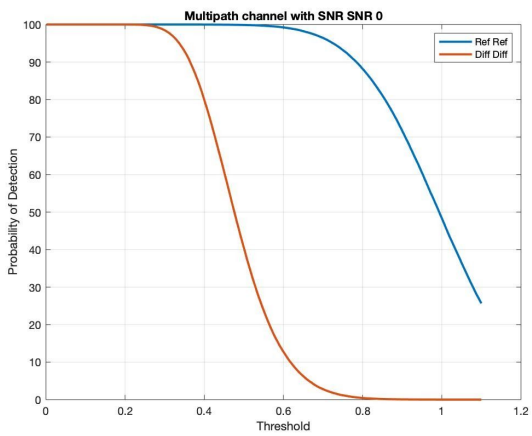


Fig. 18. Multipath SNR dB 0

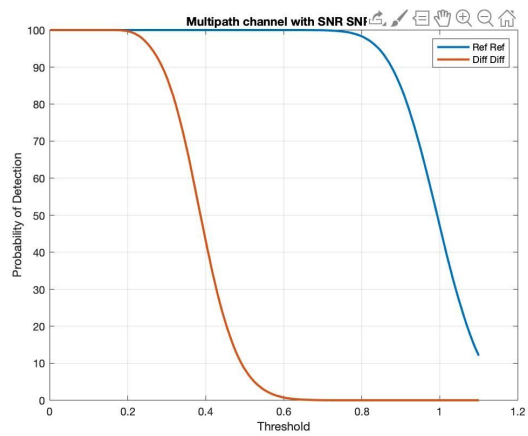


Fig. 19. Multipath SNR dB 5

As expected, as the SNR dB values increases toward one and over, the probability of detection increases for larger thresholds. As the value of SNR dB increases the less noise is in the signal. Interestingly, as the value of SNR db increases, the ref, diff cases decrease probability in lower thresholds. As the received signal, has multiple changes and interferences with a higher SNR dB.

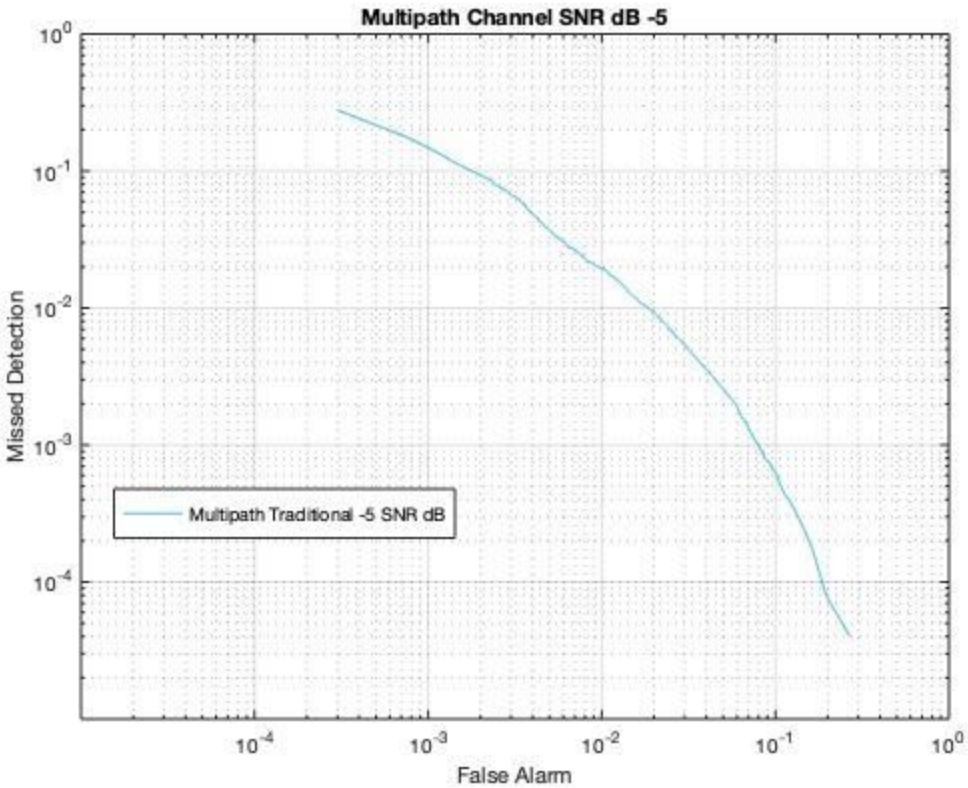


Fig. 20. NP curve for multipath channel with SNR dB -5

DEEP LEARNING PACKET DETECTION MODELS

In this section all ROC plots are shown for each architecture and each dataset, while metrics for each dataset are evaluated in the model breakdown section.

RNN

BINARY SYMMETRIC CHANNEL (BSC)

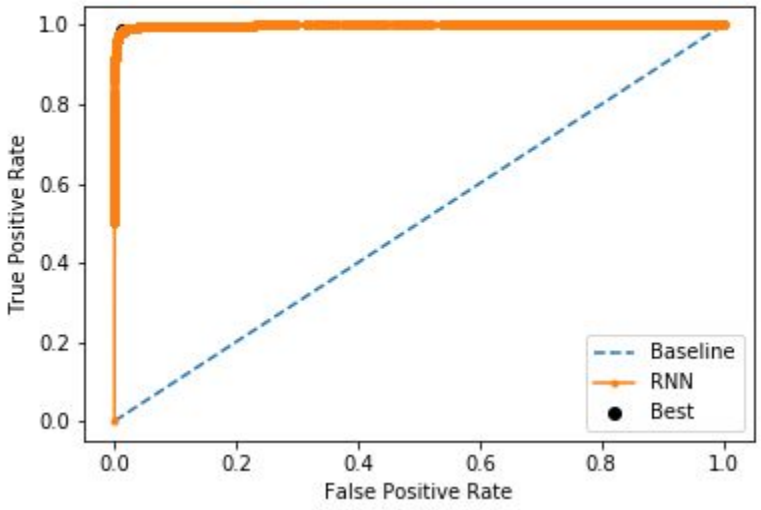


Fig. 21. BSC data RNN ROC

ADDITIVE WHITE GAUSSIAN NOISE (AWGN) CHANNEL

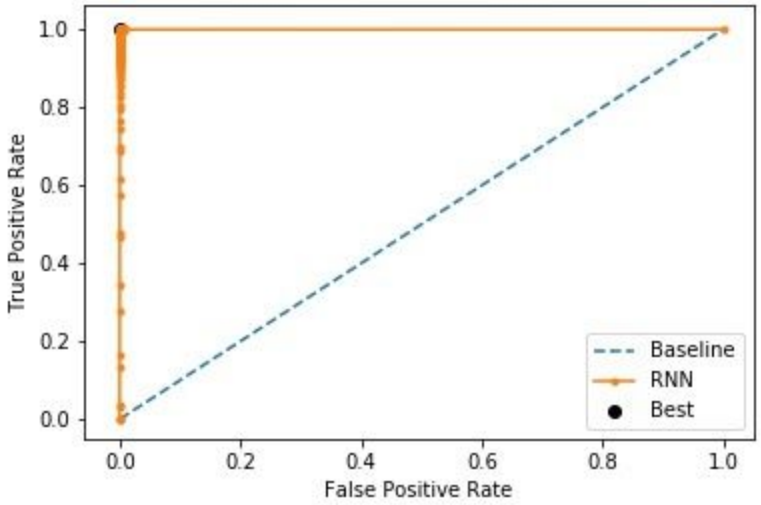


Fig. 22. AWGN data RNN ROC

MULTIPATH CHANNELS

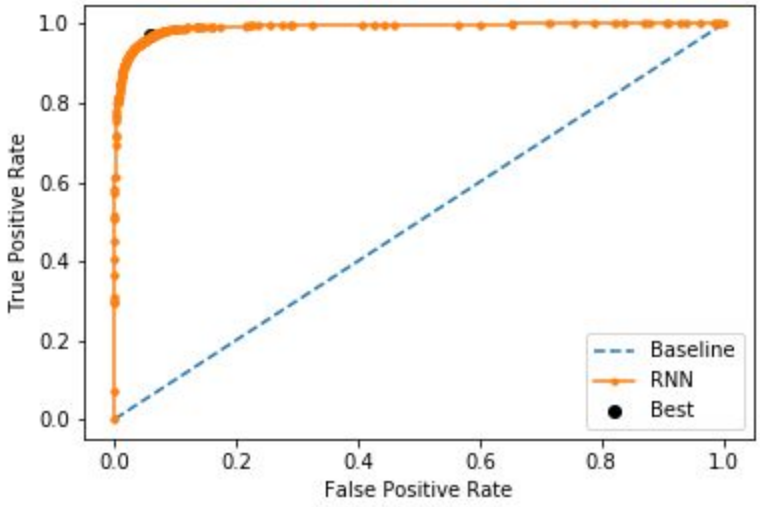


Fig. 23. Multipath data RNN ROC

RNN PERFORMANCE BREAKDOWN

The RNN architecture was trained and evaluated over three datasets as can be seen in table 1. Over all datasets the RNN performed particularly well, especially using BSC data. However this is expected because it has the least noise of the three datasets. With the addition of AWGN data however the RNN did not decrease performance too heavily with a precision of 0.95 and recall 0.92 which is a good performance over -5 SNR dB. However, interestingly the RNN model trained on multipath data outperformed the AWGN model with a precision score of 0.96 and recall of 0.94. As the multipath data is applied to real world data and applications this is a good result and has potential to be brought to a production level model.

	BSC DATA	AWGN DATA	MULTIPATH DATA
PARAMETERS	2,482	2,482	3,567
ACCURACY	0.98	0.94	0.95
PRECISION	0.97	0.95	0.96
RECALL	0.98	0.92	0.94
AUC	0.99	0.98	0.95

Table. 1. RNN Performance Evaluation

LSTM

BINARY SYMMETRIC CHANNEL (BSC)

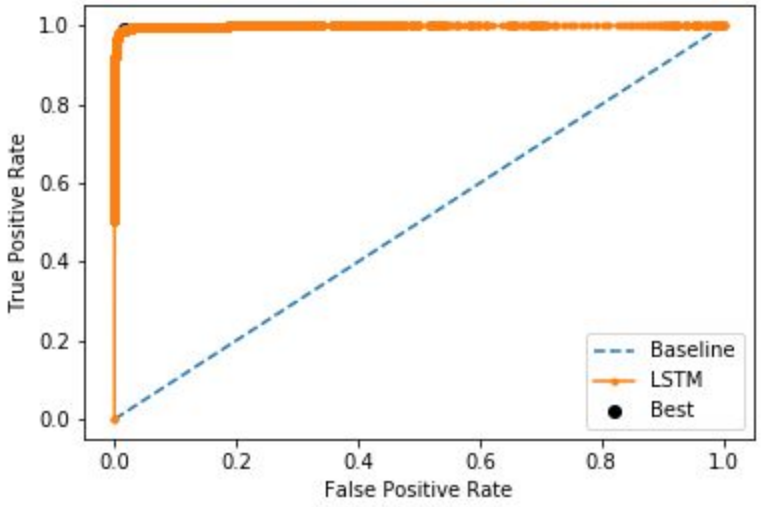


Fig. 24. BSC data LSTM ROC

ADDITIVE WHITE GAUSSIAN NOISE (AWGN) CHANNEL

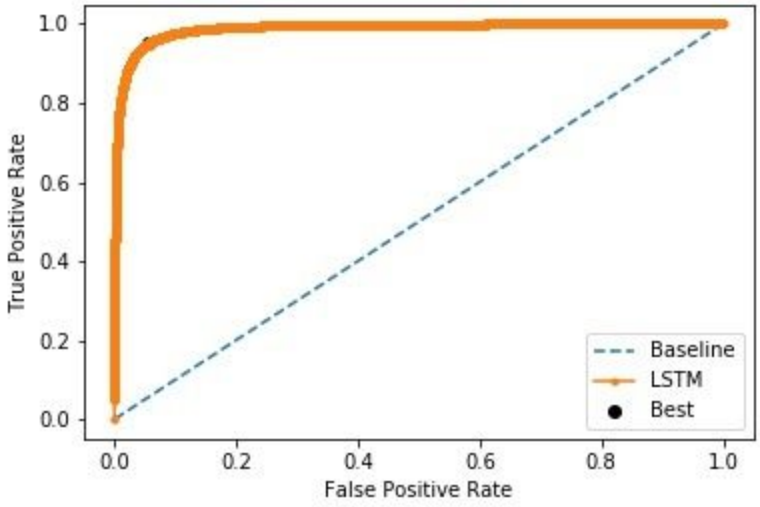


Fig. 25. AWGN data LSTM ROC

MULTIPATH CHANNELS

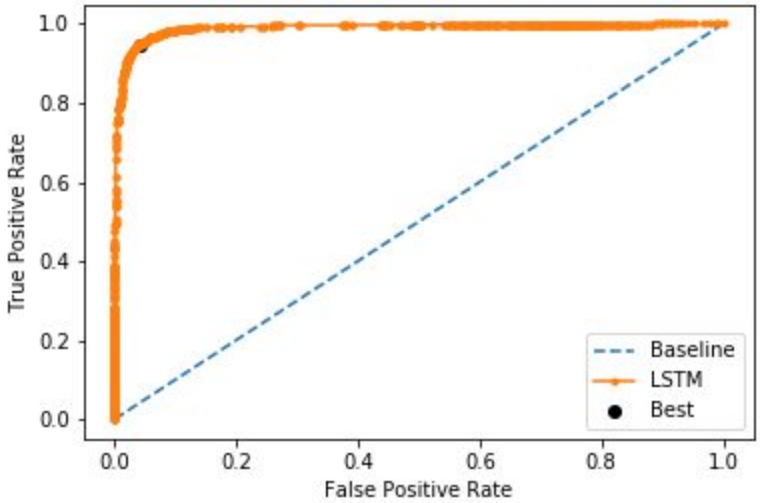


Fig. 26. Multipath data LSTM ROC

LSTM PERFORMANCE BREAKDOWN

In comparison to the RNN, the LSTM is more computationally expensive and depending on the chipset that the model will be integrated onto will determine its capabilities. Again, as expected the BSC model out performed all models comprehensively. Again, as in the RNN architecture the multipath model outperformed the AWGN model with a precision score of 0.97 in contrast to 0.94. Meaning that the multipath model correctly detected a packet 97% of the time. Again as above this model has the potential to be tested at a production level and integrated to different chipsets.

	BSC DATA	AWGN DATA	MULTIPATH DATA
PARAMETERS	2,482	15,656	23,447
ACCURACY	0.98	0.95	0.95
PRECISION	0.98	0.94	0.97
RECALL	0.98	0.94	0.94
AUC	0.99	0.98	0.95

Table. 2. LSTM Performance Evaluation

BLSTM

BINARY SYMMETRIC CHANNEL (BSC)

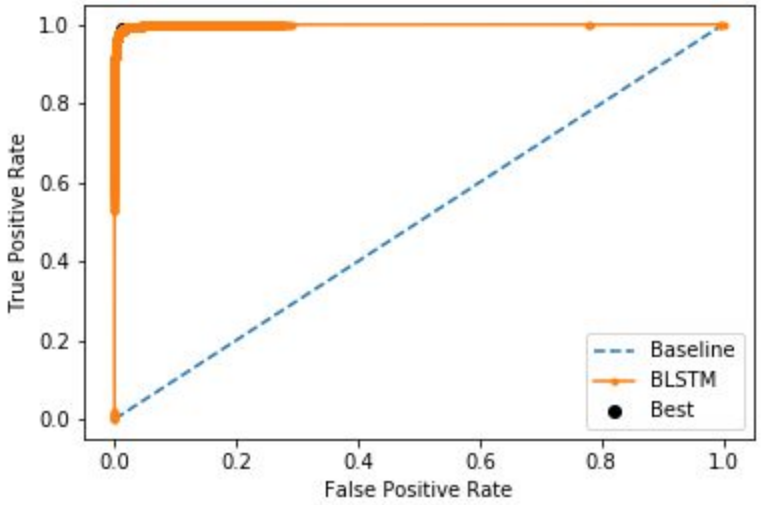


Fig. 27. BSC data BLSTM ROC

ADDITIVE WHITE GAUSSIAN NOISE (AWGN) CHANNEL

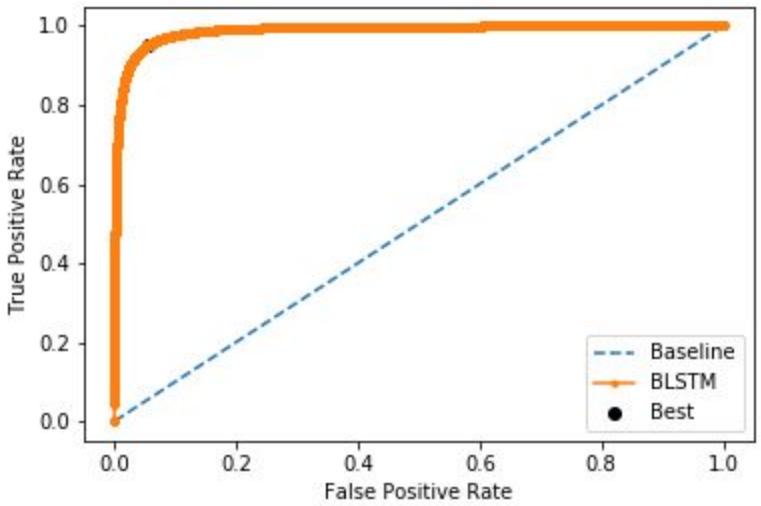


Fig. 28. AWGN data BLSTM ROC

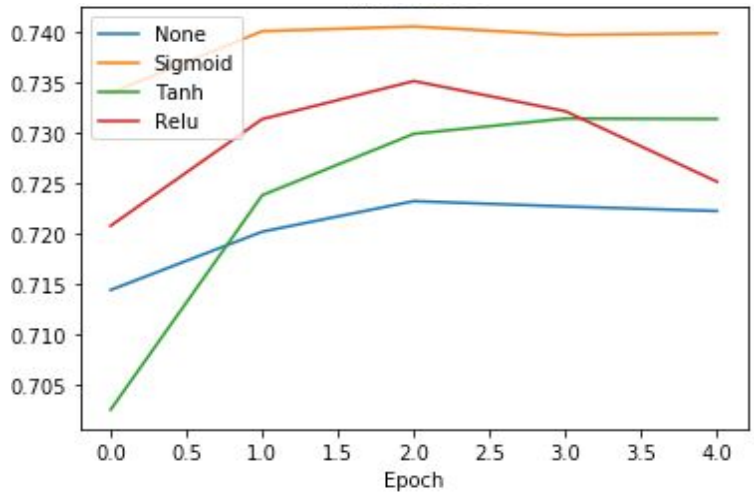


Fig. 29. Activation Function Testing

I wanted to ensure that using the sigmoid activation function was correct. I developed a never seen before test dataset separately and trained some models using the AWGN dataset over different activation functions as seen above. I found that the initial research conducted was correct and sigmoid was the correct choice to use in this case over ReLU activation.

MULTIPATH CHANNELS

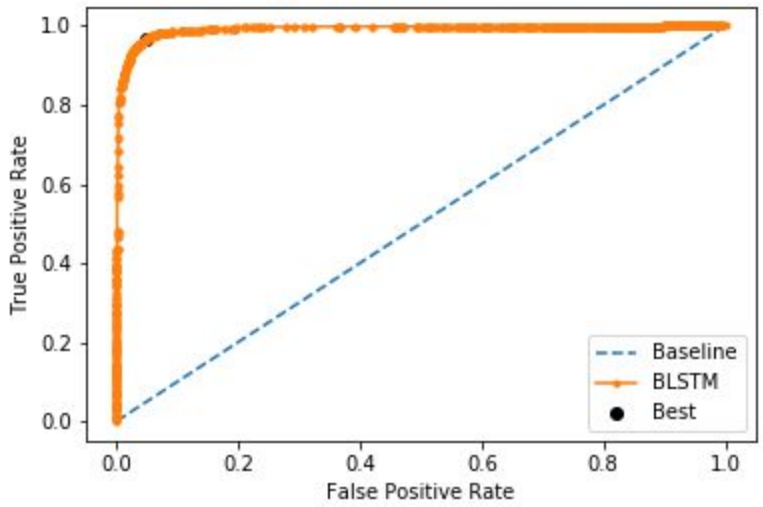


Fig. 30. Multipath data BLSTM ROC

BLSTM PERFORMANCE BREAKDOWN

As is the case with the LSTM architecture, the BLSTM architecture increases complexity as can be seen in the number of trainable parameters in each model. However, the application of BLSTM could be useful to specific chipsets in specific applications that vendors may be interested in. Again the top performing model is the BSC model as expected. Interestingly, again the multipath model outweighs the AWGN model and in precision, however has a lower recall score. However, they trade off one another resulting in largely the same detection performance.

	BSC DATA	AWGN DATA	MULTIPATH DATA
PARAMETERS	39,961	39,961	59,617
ACCURACY	0.98	0.94	0.94
PRECISION	0.98	0.95	0.97
RECALL	0.97	0.94	0.92
AUC	0.99	0.98	0.94

Table. 3. BLSTM Performance Evaluation

EVALUATION AND ANALYSIS

Fig. 31 - Fig. 33. Illustrate the NP curves between trained models and traditional methods. The objective of this project was to compare and determine the trade off between traditional packet detection methods and new state of the art deep learning approaches. Through the figures below, it can be said that deep learning approaches significantly outweigh and out perform their traditional counterparts for each dataset.

Fig.31. Sees a large increase in the performance of detection in particular the RNN and BLSTM models. However, as stated before BLSTM models carry more complexity and

varying on chipsets that these models can be integrated onto, the decision would be made on what to model to decide on.

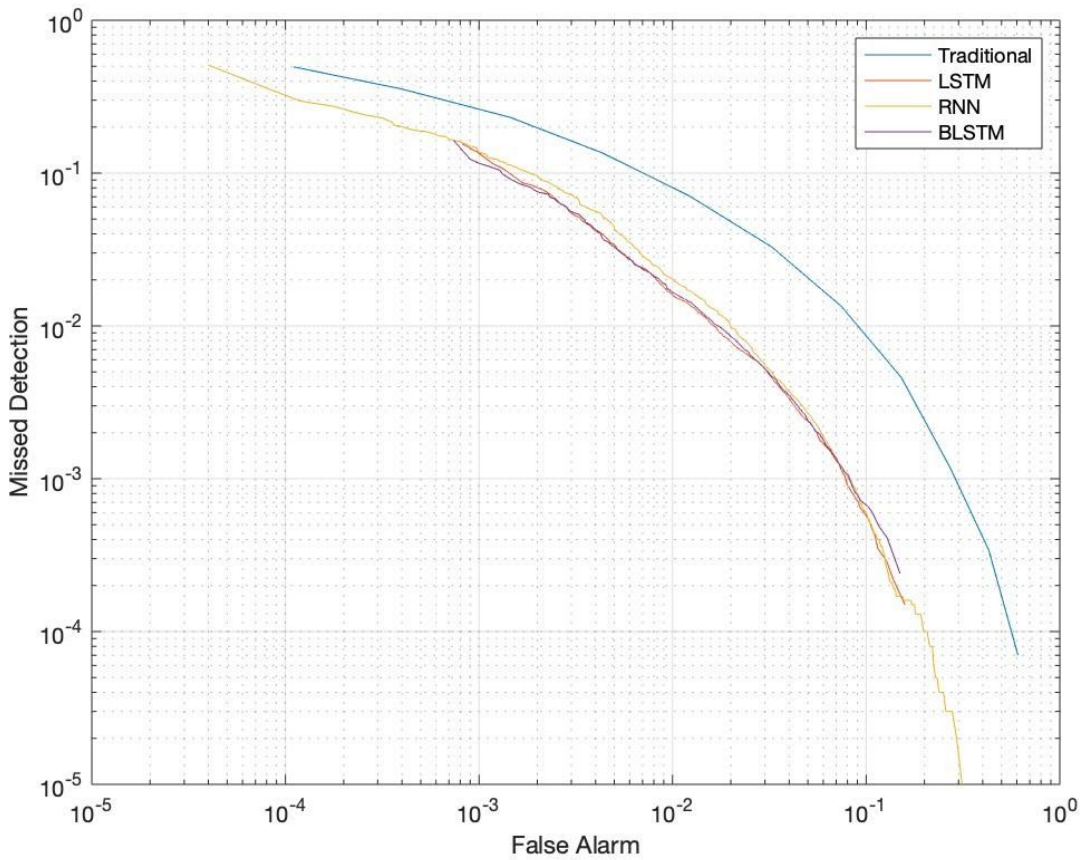


Fig. 31. NP Curves of BSC Models Vs Traditional Methods

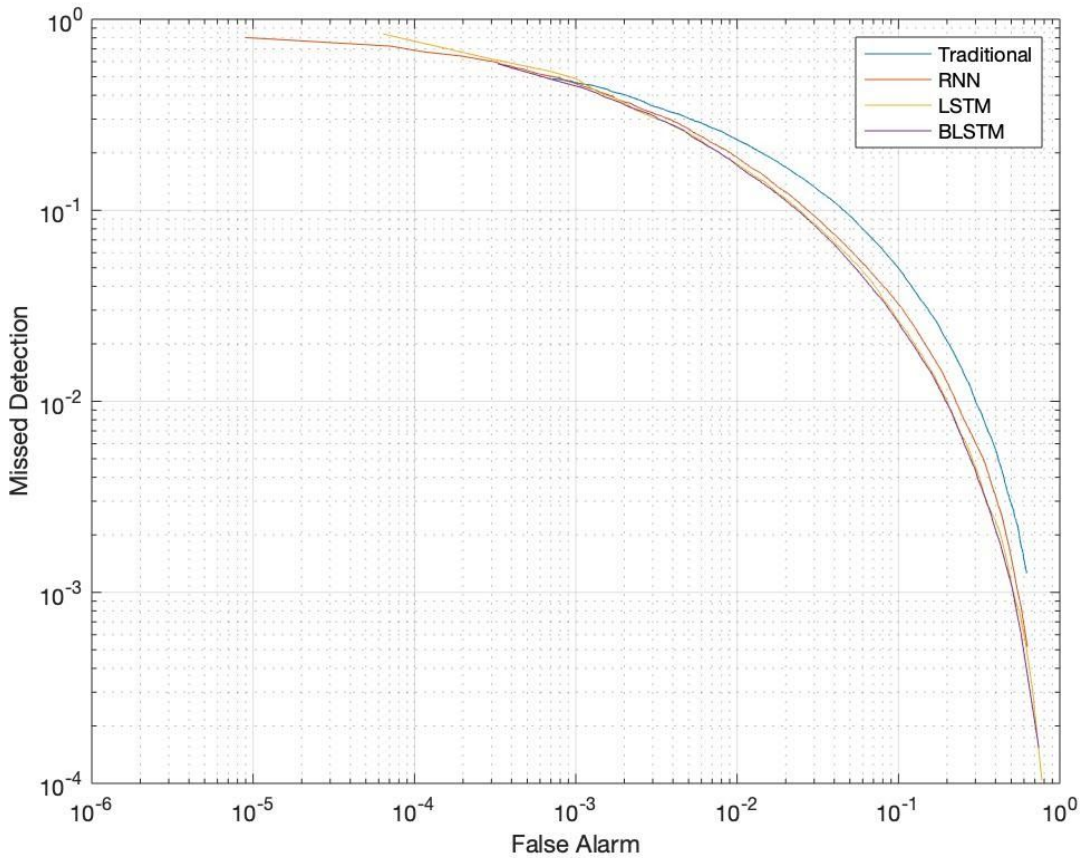


Fig. 32. NP Curves of AWGN Models Vs Traditional Methods

Fig. 32. sees a smaller increase in the performance of detection with each model performing equally illustrated by this curve. However, it can be said throughout that the AWGN models performed the worst overall.

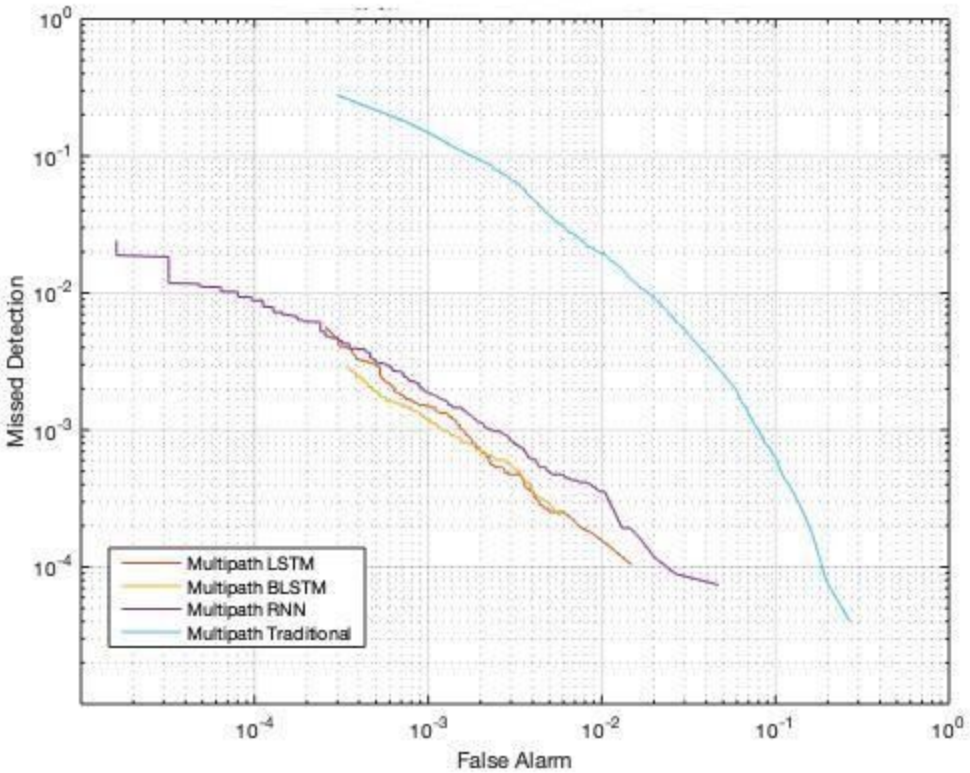


Fig. 33. NP Curves of Multipath Models Vs Traditional Methods

Fig. 33. Illustrates a large increase in the performance of detection using multipath models. This is central to real world data and the most probable models to begin integrating to chips to be tested. It can be seen here that the LSTM and BLSTM models are performing greater than the RNN however it can be seen from the metrics in table 3 that both models are quite similar. As outlined before, the computational complexity is paramount to producing the most robust and accurate model that works well under varying channel conditions.

CONCLUSION AND FUTURE PROSPECTS

In conclusion, it is evident that when comparing deep learning and traditional methods of detection that the deep learning approach significantly outperforms the traditional methods. The key aspect of this project was to find that comparison and my initial assumption that deep learning approaches would significantly increase the probability and overall packet detection in wireless networks is proven true.

The application of deep learning to this problem, will significantly reduce costs, as well as the time used redesigning chipsets. Deep learning models can be integrated through firmware, resulting in a faster, more efficient time to market.

This has proved to be encouraging research and I will continue the project with Cypress Semiconductor. Future prospects of the project include running the model on sequential data and also performing anomaly detection within packets.

APPENDIX F

SOURCE CODE LISTING

INTRODUCTION

This appendix contains all source code that is used in the project. To avoid a very long appendix all source code and documents are available on Github:

[Link to all source code and models]

The directory and subdirectory are in the order,

- Masters Project
    - Documentation
    - Matlab_scripts
    - Data*
    - Plots
        - Multipath
        - Flip_bits
        - AWGN
    - Predictions
        - LSTM
            - Multipath
            - Flip_bits
            - AWGN
        - RNN
            - Multipath
            - Flip_bits
            - AWGN
        - BLSTM
            - Multipath
            - Flip_bits
            - AWGN
    - Notebooks
        - LSTM
            - Multipath
            - Flip_bits
            - AWGN
        - RNN
            - Multipath

            - Flip_bits
            - AWGN
        - BLSTM
            - Multipath
            - Flip_bits
            - AWGN
    - Models
        - LSTM
            - Multipath
            - Flip_bits
            - AWGN
        - RNN
            - Multipath
            - Flip_bits
            - AWGN
        - BLSTM
            - Multipath
            - Flip_bits
            - AWGN

* Unable to include data used in the project on github as it is company data.