॥ सा विद्या या विमुक्तये ॥

भारतीय प्रौद्योगिकी संस्थान धारवाड

Indian Institute of Technology Dharwad

B.Tech. Project Presentation

Fixed-Point Implementation of Deep Learning based NPRACH Receiver for NB-IoT

Rishabh Tripathi (180030036)
Supervisor: Prof. Naveen M B

Introduction

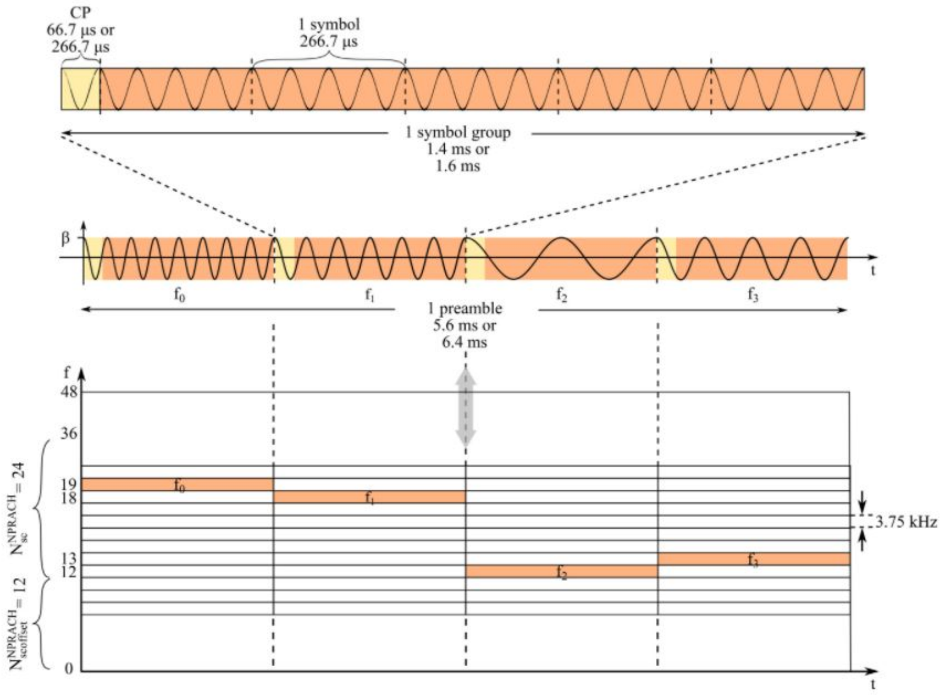
Narrowband Internet of Things (NB-IoT) is a low-power wide-area (LPWA) technology developed by 3GPP to operate on top of existing LTE systems.

NB-IoT devices do not maintain a constant connection with the base station since they transmit very small packets of data sporadically. Whenever an NB-IoT user equipment (UE) requires to transmit data in the uplink, it first acquires the downlink timing and frequency synchronization using the NPSS and the NSSS. Then the NB-IoT UE requests uplink access by randomly selecting a preamble from the available set of preambles and transmitting it on the NB-IoT physical random access channel (NPRACH). The NB-IoT base station (referred to as the eNB) detects the presence of the preamble, estimates time of-arrival (ToA) and residual carrier frequency offset (RCFO) of the UE and acknowledges the reception through the random access response (RAR).

To improve power efficiency 3GPP proposed a new random access (RA) waveform for NB-IoT based on a **single-tone frequency-hopping scheme.** RA is the first connection between the UE and the base station and ensures that the UE can be identified and synchronized with the BS.

NPRACH

Narrowband Physical Random Access Channel (NPRACH) transmits the NB-IoT preamble which is the first transmitted signal by the UE towards the eNB in order to request access to the network. Each preamble is composed of four groups of symbols. Each group of symbols is a set of five symbols with one CP. The subcarrier spacing used for the transmission of NPRACH is equal to 3.75 kHz. This leads to 48 possible subcarriers that can be used to transmit a preamble over the 180 kHz bandwidth.

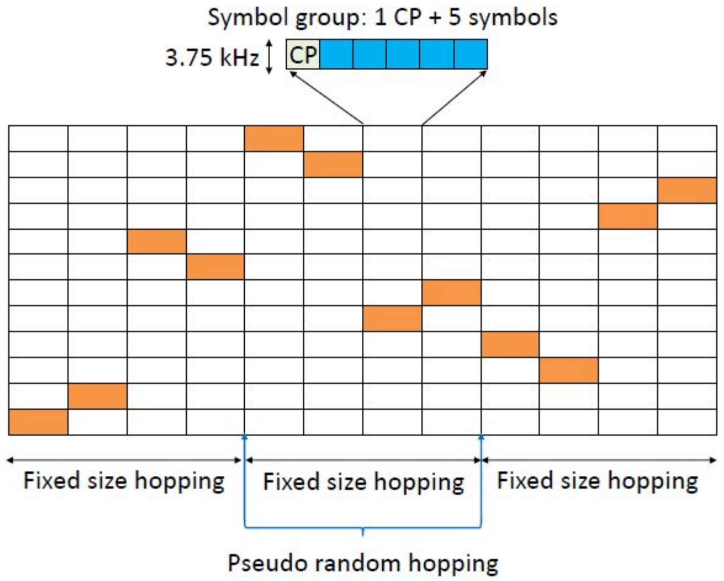


Frequency Hopping

The frequency hopping algorithm is designed in a way that different selections of the first subcarrier lead to hopping schemes which never overlap.

$$f_i = (n_{sc}^{RA}(i) + Kk_0 + \frac{1}{2}) \times 3.75, \tag{4}$$

where $n_{sc}^{RA}(i) \in \{0, 1, .., 47\}$ is the pseudo-random subcarrier index, K and k_0 are two parameters defined in [65] and $K \times k_0$ always equals -24 which leads to $f_i = (n_{sc}^{RA}(i) - 23.5) \times 3.75$ kHz.



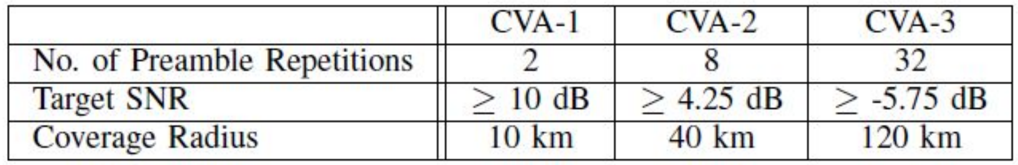
NPRACH Detection

The eNB receiving the NPRACH preamble has to do the following:

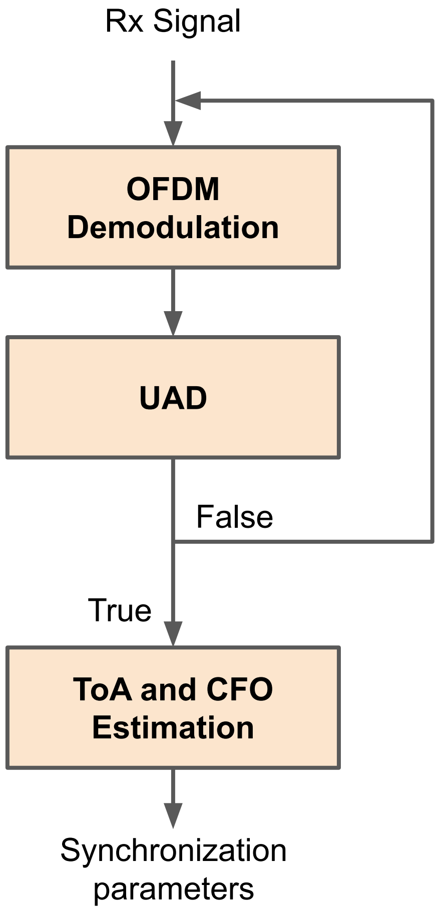
1. **User Activity Detection (UAD):** Detecting the presence of a UE sending preamble to the base station.
2. **Estimate the synchronization parameters:** Estimate the time of arrival (ToA) and carrier frequency offset (CFO), of each active user.

There can be dynamic detection of the coverage area for the active UE.

NB-IoT COVERAGE AREAS [4]



	CVA-1	CVA-2	CVA-3
No. of Preamble Repetitions	2	8	32
Target SNR	$\geq$ 10 dB	$\geq$ 4.25 dB	$\geq$ -5.75 dB
Coverage Radius	10 km	40 km	120 km



2D FFT Based Receiver [2]

UE Transmit Signal

$$s[n;m] = \frac{\sqrt{E}}{N}\sum_{k} S[k;m]e^{j2\pi\frac{k}{N}n}, n = -N_{cp}, \ldots, N-1,$$

Spectrum of Received Signal

$$\tilde{y}[i;m] = B(\Delta f, D)a[m]u[m]e^{j2\pi\Delta f(m(N_{cp}+\xi N)+iN))} \times e^{-j2\pi\frac{\Omega(m)}{N}D} + \tilde{v}[i;m].$$

PREAMBLE FORMAT 0: $N_{cp} = N/4$

PREAMBLE FORMAT 1: $N_{cp} = N$

Detection using 2D Fast Fourier Transform

$$W_g[p,q] = \sum_{n=0}^{M_1-1}\sum_{k=0}^{M_2-1} w_g[n,k]e^{-j2\pi\frac{n}{M_1}p}e^{-j2\pi\frac{k}{M_2}q}$$

$$z[i;m] = \tilde{y}[i;m]u^*[m].$$

$$w_g[n,k] = \begin{cases} z[i;m] & \text{if } n = (m-gQ)(\xi+1)+i,\ k = \Omega(m); \\ 0 & \text{otherwise.} \end{cases}$$

2D FFT Based Receiver [2]

Adding Correlation over Multiple Repetitions

$$\tilde{J}[p,q] = \sum_{g=0}^{L/Q-1} |W_g[p,q]|^2$$

$(p^\star, q^\star)$ Point of Maximum Correlation

Estimating CFO and ToA

$$\Delta f^\star = \begin{cases} \frac{1}{NM_1} p^\star & \text{if } p^\star < \frac{M_1}{2}; \\ \frac{1}{NM_1}(p^\star - M_1) & \text{otherwise.} \end{cases}$$

$$D^\star = \begin{cases} -\frac{N}{M_2} q^\star & \text{if } q^\star < \frac{M_2}{2}; \\ -\frac{N}{M_2}(q^\star - M_2) & \text{otherwise.} \end{cases}$$

The problem with 2D FFT based receiver is that it is basically a brute force search for ToA and CFO estimates which maximize the metric *J[p, q]*.

The 2D FFT operation is very computationally expensive and consumes the majority of the time in the receiver logic.

Deep Learning Based Receiver [3]

To avoid the computational expense of the 2D FFT based receiver an alternative receiver design is used which is based on deep learning and convolutional neural networks [3]. Different DNNs are designed and trained for the following cases:

1. UAD for CVA1 and CVA2 with dynamic CVA detection.
2. UAD for CVA3
3. ToA and RFCO estimation for CVA1 and CVA2
4. ToA and RFCO estimation for CVA3

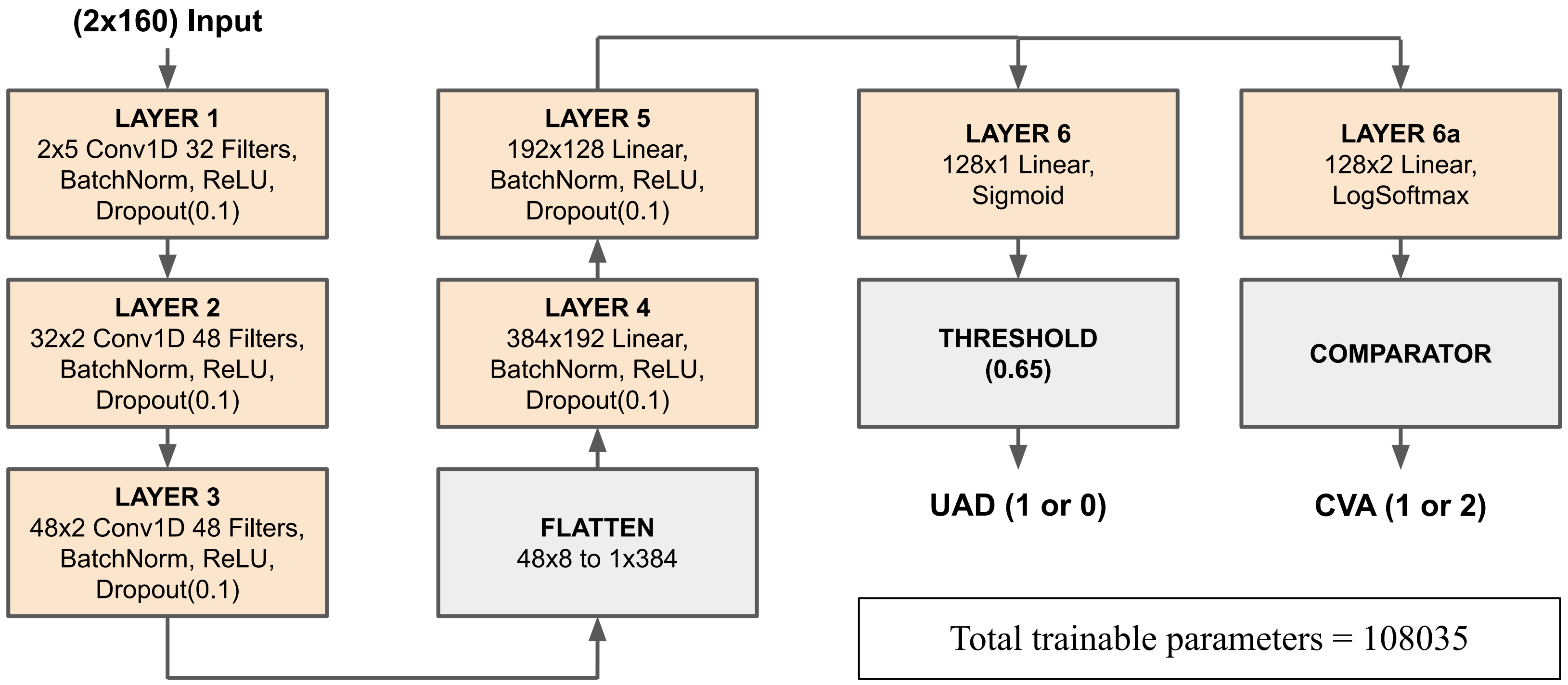
Here we will be looking at the implementation of the first DNN mentioned above (using preamble format 0). The implementation aims at verifying the feasibility and reliability of the receiver in a fixed-point processing environment with limited computational and memory resources.

NPRACH Rx Signal for UAD

Since CVA-2 has larger number of repetitions and the eNB has to dynamically detect whether the UE belong to CVA-1 or CVA-2, we begin with the received grid in frequency domain whose size corresponds to CVA-2, i.e, 8 repetitions. We separate the in-phase and quadrature components of the received complex samples, obtaining a matrix of size **(2x160)** for each user. The channel model used is EPA 1 Low and CP length of 128 (PRM0). Antenna configuration is: 1 Tx and 2 Rx.

Rx signal length (CVA2)	86016
No. of symbols per group	5
No. of symbol groups in a repetition	4
No. of repetitions used (CVA2)	8
Total no. of symbol groups	32
Total no. of symbols	160

DNN for UAD at CVA1 and CVA2 [3]


(2x160) Input
LAYER 1
2x5 Conv1D 32 Filters, BatchNorm, ReLU, Dropout(0.1)
LAYER 2
32x2 Conv1D 48 Filters, BatchNorm, ReLU, Dropout(0.1)
LAYER 3
48x2 Conv1D 48 Filters, BatchNorm, ReLU, Dropout(0.1)
FLATTEN
48x8 to 1x384
LAYER 4
384x192 Linear, BatchNorm, ReLU, Dropout(0.1)
LAYER 5
192x128 Linear, BatchNorm, ReLU, Dropout(0.1)
LAYER 6
128x1 Linear, Sigmoid
THRESHOLD (0.65)
UAD (1 or 0)
LAYER 6a
128x2 Linear, LogSoftmax
COMPARATOR
CVA (1 or 2)

Total trainable parameters = 108035

MATLAB Implementation (Floating Point)

MATLAB implementation is made done by converting the evaluation flow of Pytorch DL model to MATLAB functions using only matrix manipulations. The following shows an example snippet of code from *conv1d()*:

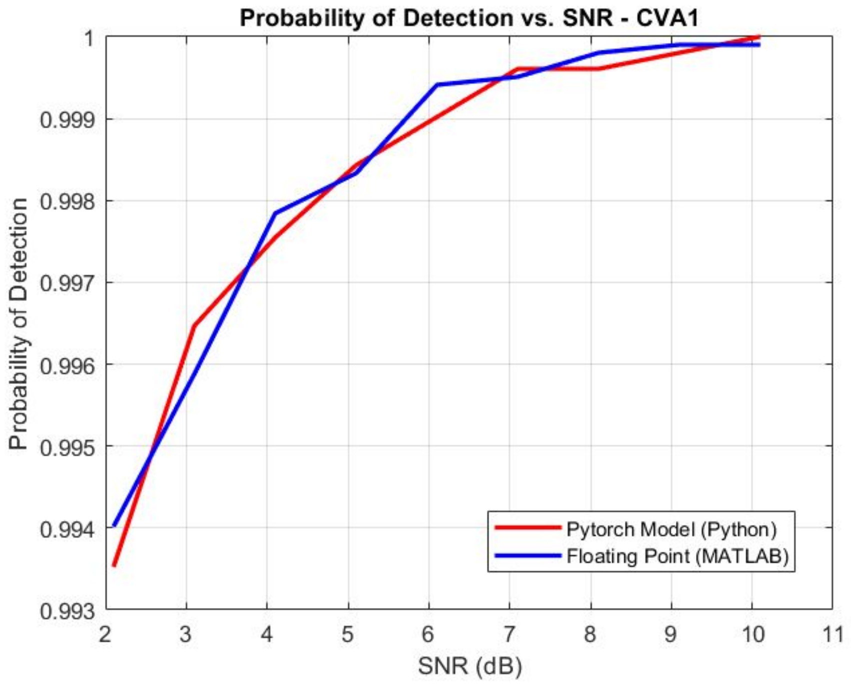
```
% Computing the output using the weights and bias
for ch = 1:y_channels
    w_ch = reshape(weights(ch, :, :), [w_shape(2), w_shape(3)]);

    for sample = 1:y_length
        temp = 0;
        for x_ch = 1:x_channels
            start_pt = (sample-1)*stride+1;
            temp = temp + sum(x(x_ch, start_pt:start_pt+kernel_size-1) .* w_ch(x_ch, :));
        end
        y(ch, sample) = temp + bias(ch);
    end
end
```

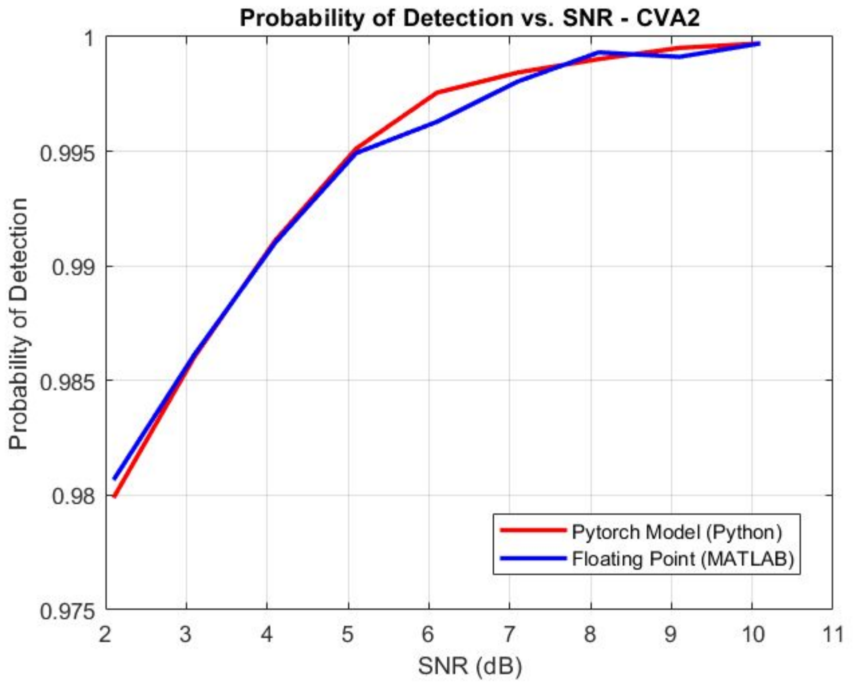
MATLAB Implementation Results

MATLAB implementation results match very closely with the Pytorch model results. The table below compares the probability of detection of users:

Model	Pytorch Model		MATLAB Model	
Coverage Area	1	2	1	2
SNR (dB)	10.1	6.1	10.1	6.1
Probability of detection	100%	99.76%	99.99%	99.63%
False Alarm Probability (6.1 dB)	0.0098%		0.0049%	

MATLAB Implementation Plots





C Code Implementation (Fixed Point)

The DL receiver is next implemented in fixed point system in C using a general register size of **16 bits.** The scaling factor for each floating point number is **2^8**, that is, **256.** The C code based system takes Rx Data Matrix of size **(Nx2x160),** where N is number of test cases, as input which is generated using OFDM demodulation from MATLAB functions. Below is shown a code snippet from *conv1d()* function:

```
for(int ch_i = 0; ch_i < out_ch; ch_i++) // iterate over output channels
{
    for(int size_i = 0; size_i < out_size; size_i++) // iterate over the size of output
    {
        temp = 0;
        // Calculate the convolution sum for 1 output sample
        for (int x_ch = 0; x_ch < in_ch; x_ch++) // iterate over input channels
        {
            int start_pt = size_i * stride;
            for(int i = start_pt, j = 0; i < (start_pt + f_size); i++, j++) // data slice to multiply with filter
            {
                temp = temp + mult_fixed(x[x_ch][i], conv_w[ch_i][x_ch][j]);
            }
        }
        y[ch_i][size_i] = temp + conv_b[ch_i];
    }
}
```

C Fixed Point Implementation Results

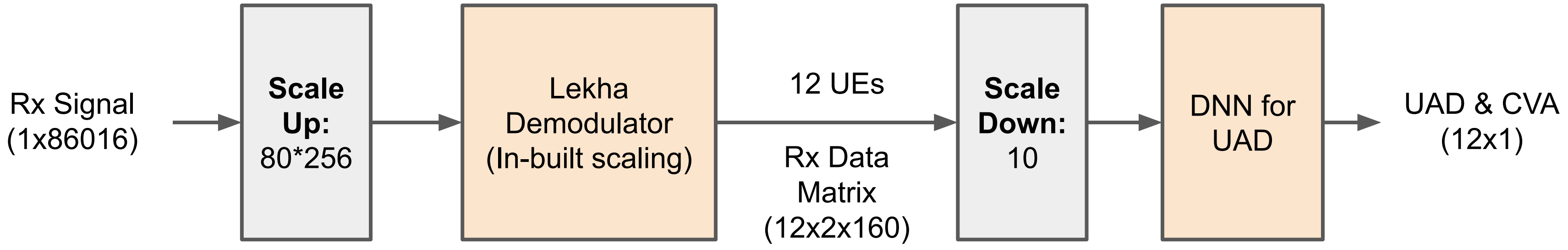
The table below compares the detection probabilities of fixed point C implementation with MATLAB and Pytorch models. It is observed from the table and the plots that threshold value which was **0.65** for floating point implementation does not work well with the fixed point implementation (too high threshold as false alarm never occurs). An optimal threshold value is searched for and used later.

Model	Pytorch Model		MATLAB Model		C (Fixed Point)	
Coverage Area	1	2	1	2	1	2
SNR (dB)	10.1	6.1	10.1	6.1	10.1	6.1
Probability of detection	100%	99.76%	99.99%	99.63%	99.92%	99.35%
False Alarm Probability (6.1 dB)	0.0098%		0.0049%		0%	

Fixed Point C Code with Lekha Demodulator

The demodulation block using the fixed point FFT function provided by *Lekha Wireless Solutions Pvt. Ltd.* is integrated with the fixed point C implementation of DL receiver. The Lekha Rx system takes the Rx signal of **86016** samples length as input (for CVA-2).

A new threshold was searched by fixing the SNR at 6.1 dB and iterating over threshold values. It is found that a threshold of **0.03** works well for both CVA-1 and CVA-2 cases.

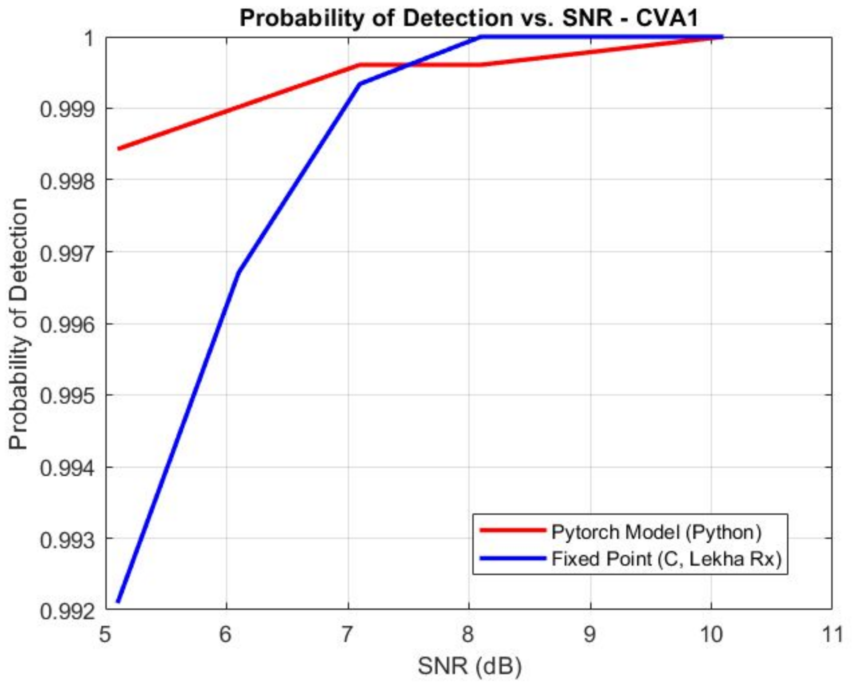


Final Results

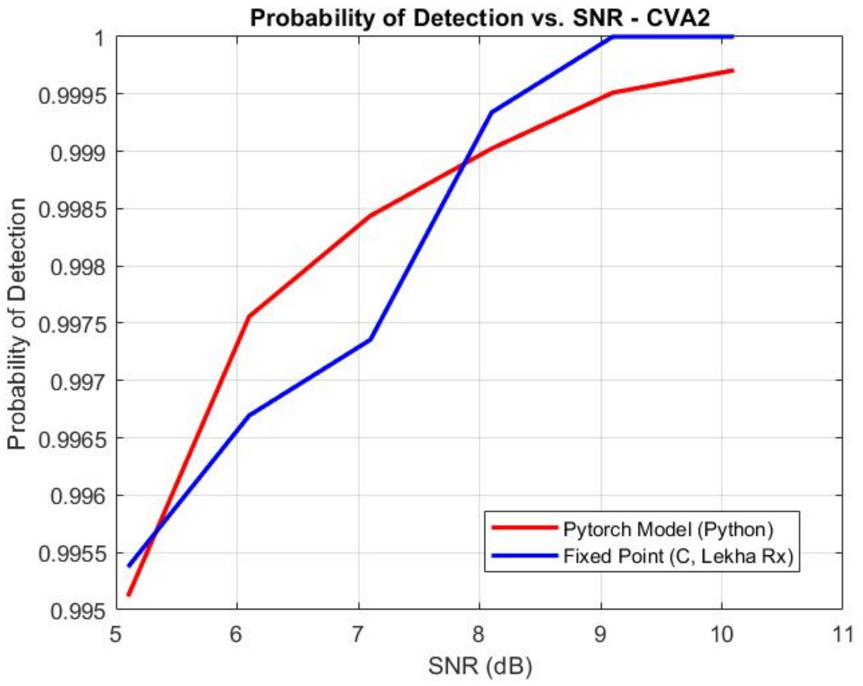
The final fixed point C implementation uses the Lekha Rx Demodulator and a threshold of **0.03** with a scaling of **256.** The table below compares its results against Pytorch and MATLAB floating point implementations.

Model	Pytorch Model		MATLAB Model		C (Fixed Point)	
Coverage Area	1	2	1	2	1	2
SNR (dB)	10.1	6.1	10.1	6.1	10.1	6.1
Probability of detection	100%	99.76%	99.99%	99.63%	100%	99.67%
False Alarm Probability (6.1 dB)	0.0098%		0.0049%		0%	

Final Plots



Probability of Detection vs. SNR - CVA1
Probability of Detection
SNR (dB)
1
0.999
0.998
0.997
0.996
0.995
0.994
0.993
0.992
5
6
7
8
9
10
11
Pytorch Model (Python)
Fixed Point (C, Lekha Rx)



Probability of Detection vs. SNR - CVA2
Probability of Detection
SNR (dB)
1
0.9995
0.999
0.9985
0.998
0.9975
0.997
0.9965
0.996
0.9955
0.995
5
6
7
8
9
10
11
Pytorch Model (Python)
Fixed Point (C, Lekha Rx)

Conclusion

1. The fixed point implementation of DL based NPRACH receiver performs well in comparison to the Pytorch floating point model.
2. The implementation results show detection probabilities which fall in the required range for target SNRs in both CVA-1 and CVA-2 test cases.
3. The system can run on a scaling of 256 and with a register size of 16 bits which makes it suitable for typical DSP processors and deployment systems.
4. The results show that the fixed point implementation of the DL based NPRACH receiver is a potential alternative to 2D FFT based receiver.

Future Work

1. Performance comparison between the Fixed point DL based receiver and Fixed point 2D FFT receiver in terms of timing and memory requirements.
2. Further optimization to avoid time consuming arithmetic operations (e.g., divisions) in order to reduce the total processing time.
3. Memory optimizations by efficiently loading and clearing the required matrices of trained parameters.
4. Fixed point implementation of DNN used for UAD at CVA-3 as well as DNNs for ToA and RCFO estimation at all coverage areas.

References

1. Y.-P. E. Wang, X. Lin, A. Adhikary, A. Grovlen, Y. Sui, Y. Blankenship, J. Bergman, and H. S. Razaghi, “A primer on 3GPP narrowband Internet of things,” IEEE Commun. Mag., vol. 55, no. 3, pp. 117–123, 2017.
2. X. Lin, A. Adhikary, and Y.-P. E. Wang, “Random access preamble design and detection for 3GPP narrowband IoT systems,” IEEE Wireless Commun. Lett., vol. 5, no. 6, pp. 640–643, 2016.
3. Yashwanth Ramesh Kumar and Naveen Mysore Balasubramanya, “Deep Learning-based NPRACH Receiver for NB-IoT Systems.”

THANK YOU