Non Iterative Decoding of Low Density Parity Check Codes Using Artificial Neural Networks

D. Blackmer[1], F. Schwaner[1], and A. Abedi[1]
[1]Department of Electrical Engineering, University of Maine, Orono, ME, 04473 USA

Abstract[1]— *The decoding of Low Density Parity Check (LDPC) codes through the iterative process of belief propagation presents practical challenges for designers looking for real time performance in communication systems. Due to the geometric and finite pattern nature endemic in the construction of LDPC codes, this paper proposes the use of Multi Layer Perception (MLP) feed forward artificial neural networks to replace belief propagation to achieve constant decoding times while retaining performance levels comparable to more traditional decoding methods. Due to the back propagation training method used for neural networks, and the requirement of showing the network every possible input output sequence it will ever see, this paper also presents a novel method of approaching long block length codes far larger than is otherwise possible to train neural networks for with modern computer hardware.*

Keywords – Neural Networks, Low Density Parity Check, Decoding algorithms, channel coding, communication theory

1. INTRODUCTION

IN the field of designing error correcting codes, two types of codes stand out as being able to achieve performance rates near the theoretical upper channel capacity limit laid out in the Shannon theorem [1]. In the field of convolution codes, turbo codes with proper iterative decoding based on belief propagation can come within fractions of a dB of the limit. [2] And in the field of linear block codes Low Density Parity Check (LDPC) codes with sufficient block length also can achieve comparable performance arbitrarily close to this theoretical upper limit on binary symmetric channels [3]. The limitation of both of these types of codes is that the decoding is an iterative process to which performance is frequently tied to the number of iterations in the decoding process. With sufficient iterations, the decoding algorithm can converge with maximum likelihood to what the original transmitted data was. The general rule tends to be, the more iterations, the closer to the Shannon limit the performance of each type of code can achieve. For real time communications this poses a problem, because for many applications users will not tolerate the latency required to iterate through a long code, and from a engineering point of view a powerful floating point processor with sufficient memory must be dedicated to the decoding process of the Sum Product Algorithm (SPA), where the computational complexity is a linear function of the number of 1's in the parity check matrix H [4]. Some research has been done with limited precision SPA, but the precision drops off for greater and greater quantization error [5]. For low signal powers associated with mobile devices which must contend with battery storage capacity and maximum efficiency requires that every clock cycle must be used to complete as much work as possible. A new method of efficiently decoding these signals at high, constant speeds has been proposed. The idea of using Artificial Neural Networks (ANN) for the purpose of decoding LDPC codes by itself is not a new idea, however one of the inherent limitations has always been the length of the codes being constrained by the length that is reasonable to train [6]. Using the inherent pattern recognition and generalization abilities of a properly trained neural network can enable constant time very high speed, non iterative LDPC decoding, with error performance levels on short codes approaching or even surpassing more traditional iterative belief propagation decoding methods.

2. LDPC A BRIEF REVIEW

To understand how errors are corrected with LDPC, first look at the highlighted row in *Figure 1*. As with all other rows in this particular parity check matrix there are exactly 4 bits. These four bits for every valid codeword will have known parity.

$$\begin{bmatrix}
1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0
\end{bmatrix}$$

Figure 1: H Matrix for (12,6) LDPC code

$$mod_2\{X_2 + X_7 + X_8 + X_{12}\} = 0 \qquad (1)$$

As demonstrated in (1) the parity check matrix

[1] This work was sponsored by NSF IGERT Grant Number DGE-0504494 and NASA Grant Number MSGC-EP-11-05.

guarantees that the indexed values from the rows of the received codeword specified by the ones will have known parity. This is at the core of what the LDPC does to provide the decoder with a-priori information to allow error correction. To further this process and to allow for the exchange of information between rows, the highlighted column of *Figure 1* demonstrates that the two rows share one common check value. Using a tanner graph one can verify that the smallest cycle in this *H* matrix is of length 3. The objective is that each row has no more than one bit in common with any other row to reduce short cycles in the connectedness graph. The example in *Figure 1* only has two ones per column; however there is no specific number of 1's per row or column which makes a good code. Irregular LDPC codes have been shown to have better performance in certain situations [7]. The neural network approach proposed by this research allows us to deal with regular or irregular codes; however a new network must be trained for each new row with a different number of ones.

As the proposed method discussed here breaks apart the H space into individual rows, where each row has $\boldsymbol{n}$ ones**.** Now since each row must have even parity that means there are $\mathbf{2^{n-1}}$ possible permutations of these values. Thus in the case of a row with $\boldsymbol{n = 4}$ there are $\mathbf{2^3 = 8}$ valid codewords. Each sequence of which is separated by at least $d_{min} = 2$ values. Since the neural network only has to be trained with $\mathbf{2^{n-1}}$ sequences it simplifies and expedites the training process. But more than simplifying it, this approach makes the training possible. It can be easily seen that with modern computer memory limitations it would be impossible to train a network with a data of binary length 100 bits, since it would have $\mathbf{2^{100}}$ permutations. Even in binary form this would require $\mathbf{5.07 * 10^{21}}$ GB of ram to store all the permutations of the training vector. And due to the so called curse of dimensionality, require such a large network as to be completely impossible to train or operate with any modern systems.

3. Multi Layer Perceptions

Artificial Neural Networks (ANN) of the Multi Layer Perception (MLP) are a class of feed forward neural networks, meaning they have no recursive or feedback connections. As shown in Figure 2, they are constructed by interconnecting multiple summing blocks which each sum

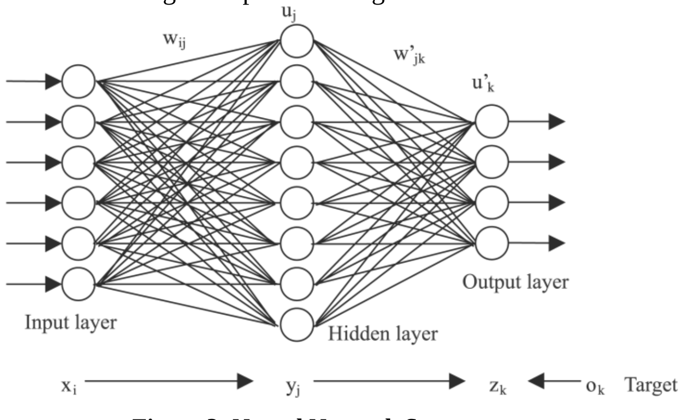

Figure 2: Neural Network Structure

together some scaled version of the input. For the application of pattern recognition, the results of each summation gets multiplied by a nonlinear sigmoid activation function then feed to the next layer of perceptions where the same set of operations gets repeated. After this has propagated through several layers as shown in Figure 2, the signal reaches a probability based competitive layer which makes a decision about the likelihood that the input vector belongs to each possible class of valid outputs.

These feedforward networks are ideally suited for pattern recognition [8], and have several major benefits which make them a great choice decoding signals. Their ability to properly classify inputs when presented with novel signal data means that even when corrupted with random noise, the neural networks can be trained to look past the signal errors and noise and find the underlying geometric relationship that defines the coded signal. The method of training the network, rather than having a static design provides another benefit for the purposes of codeword recognition. Being shown perfect versions of the signal, then having a gradient descent algorithm update each interconnecting weight in an attempt to iteratively find global function minima in the output space. Then being shown corrupted versions to allow the network to generalize itself to novel inputs better. This method allows the training to be done offline, iteratively approaching successively better and better network weight and bias configurations for network performance. Then when the network is online, the performance will be tied to the performance of the trained network. However MLP networks, unlike other algorithms used for pattern recognition, require no recursion or iteration to achieve similar performance levels online as other algorithms achieve. [9]

4. FPGA Neural Network Pattern Recognition

One major benefit of this approach can be seen in the lack of precision required by neural networks to achieve good results. The exact mathematical precision for the sigmoid activation functions is not nearly as important as its shape, so fixed point lookup tables can be used to perform what could otherwise be a computationally complex nonlinear transformation [10]. It has been shown that when using 16-bit fixed point math with VHDL synthesis tools for FPGA's using the *uog_fixed_arith* library that performance of neural networks can be increased by 12x, and exhibit quantization error of only $2.411x10^{-4}$ for the bipolar range of [-8 8]. [11] Also due to the manner in which the network is trained instead of programmed, there are many possible convergent points which will produce good results. This means that the network can be trained using fixed point weights which consist only of powers of two and sums of powers of two. This means that all the multiplications can be done with shift registers and additions rather than requiring dedicated multiplication hardware [12]. This results in a network which can be implemented in a FPGA in a massively parallel fashion taking up no extra clock cycles for a CPU to accomplish near real-time decoding. This highlights the true benefit of the neural network approach, the ability to do the training offline, then to implement the trained network as dedicated logic registers. The referenced papers and texts provide proof for the reader that neural networks can be implemented effectively in this way. It is outside of the scope of this paper to implement and demonstrate the specific performance of this method, however the reader should be aware that through the work of others achieving this extreme performance is possible using the neural network approach.

5. IMPLEMENTATION

Therefore this paper presents two different approaches to using neural networks for decoding. The first approach is for illustrative purposes and has been investigated before [13] for linear block codes. This technique is to train the network with all possible valid codewords. Being that this approach will only work for short codes, a second approach is proposed by this paper. The idea of training the network using the row based parity sequences endemic of valid LDPC codes, then allowing the belief about the state of certain bits to propagate through the network.

The procedure for the first process is:

1) *Offline: With a linear block code of M data bits and N parity bits. Train a neural network where the input is a* $(2^M \times (M+N))$ *channel output matrix, and the output is a* $(2^M \times M)$ *matrix of properly decoded values.*

2) *Online: Feed the network input with each of the received noise corrupted vectors from the channel, and the maximum likelihood decoded sequence will be produced on the output.*

There are two drawbacks to this design. The first being the training being prohibitive for long sequences as discussed before. The second is referred to as the curse of dimensionality. This basically means that the number of multiplications and additions performed by a feed forward network is given by:

$$(M + N + 2^M) * H$$

M = Data Bits
N = Parity Bits
H = Number of Hidden Nodes

As the number of data bits grows, the number of hidden nodes must grow proportionally for the decoding performance to stay constant. Accordingly the number of multiplications can grow to the point of inefficiency quite quickly.

As a hypothesized alternative which removes these limitations, this paper proposes the following alternative process:

1) *Offline: Determine the number of ones* X_i *in each of the i rows of the H matrix.*

2) *Train individual networks for each unique* X_i*.*
Ex: if a LDPC has 6 rows with 5 ones each and 2 rows with 4 ones each, two unique networks, one for 5 ones and one for 4 ones are necessary

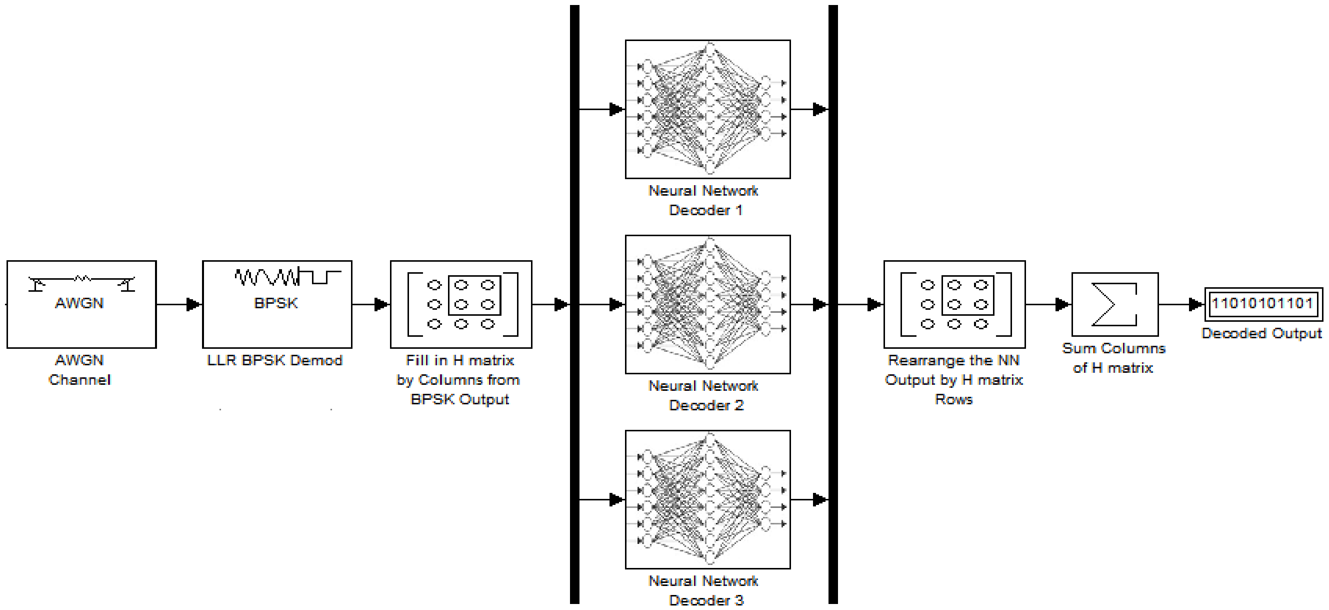

Figure 3: Full Proposed Decoder Arrangement

3) Online: Encode and transmit the LDPC vector through the channel as normal. On the receiving end rearrange the received vector from the channel into the shape of the encoded H space.

4) Feed each row of the received vector through the correctly sized neural network.

5) The output Y will now be the size $\mathbf{1}\ x\ \mathbf{2}^{X_i-1}$ *vector of likelihoods that the given input sequence belongs to each of the* $\mathbf{2}^{X_i-1}$ *possible sequences. Multiply the* $\mathbf{1}\ x\ \mathbf{2}^{X_i-1}$ *likelihood output sequence by the* $\mathbf{2}^{X_i-1}\ x\ \boldsymbol{X_i}$ *matrix of all possible valid sequences. This will generate the probabilistic* $\mathbf{1}\ x\ \boldsymbol{X_i}$ *values to fill back into the* X_i *positions from the current row of the H matrix.*

6) Sum each column to update the likelihood of each bit with the knowledge passed from each other common column bit.

The arrangement of these steps is shown in Figure *3*.

6. RESULTS

In *Figure 4* it is shown that several different neural network structures haven nearly indistinguishable performance to the Sum Product Algorithm. Once the network can be shown every possible input / output sequence, the decoding performance will achieve maximum likelihood. However because of the limitations discussed earlier this approach only works for very short codes.

Figure 5 shows the performance of the proposed alternative method. While it can be seen that the BER performance isn't up to par with the SPA approach, this demonstrates neural networks can decode the DVB-S2 standard (64,800, 32,400) LDPC code, which would as discussed earlier be completely impossible with modern hardware without using this approach.

7. CONCLUSION

Despite the lack of complimentary BER performance for the method this paper has proposed compared to SPA decoding, the concept that a neural network can be used to decode a much longer code by identifying substructures which can be individually and independently decoded has been verified. Performance levels could be improved if the identified substructures were uncorrelated, since it is assumed that it's the correlation between various sub elements which prevents the performance from improving any more with a greater number of ones in each column. Further research is needed to improve the performance levels of this method. This paper has presented a starting framework from which to build a better performing neural network decoder which can approach substantially longer LDPC codes than was previously possible.

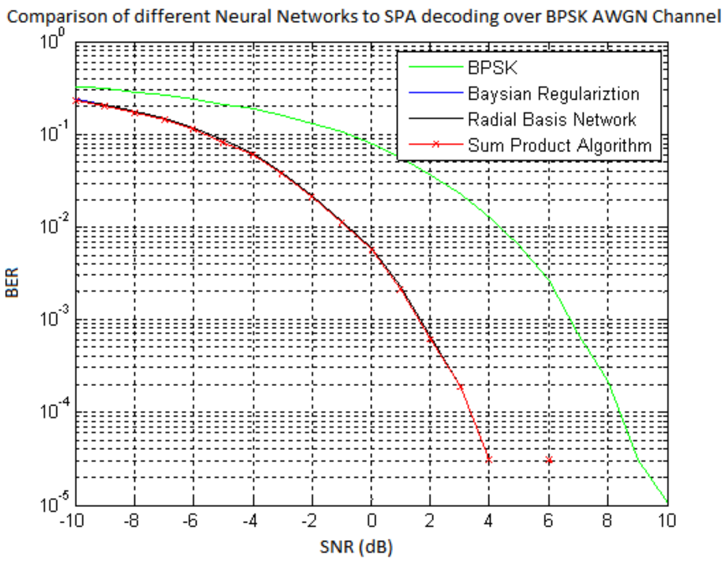

Figure 4: Comparison of Several Neural Network Types vs. SPA for a short (6,2) LDPC code

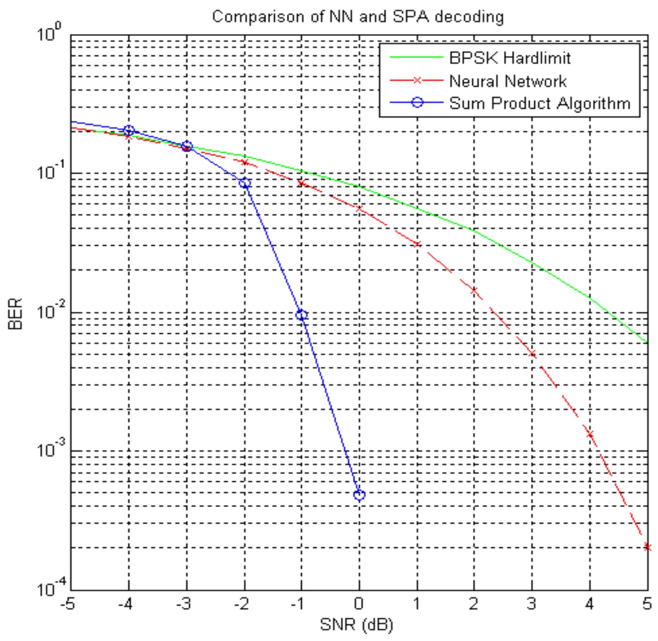

Figure 5: DVB-S2 (64,800, 32,400) LDPC Code with both methods

8. REFERENCES

[1] Sae-Young Chung, G. David Forney, Jr., Thomas J. Richardson, and Rüdiger Urbanke, "*On the Design of Low-Density Parity-Check Codes within 0.0045 dB of the Shannon Limit*", IEEE Communication Letters, vol. 5, pp.58-60, Feb. 2001. ISSN 1089-7798.

[2] Berrou, C "Near Shannon limit error-correcting coding and decoding: Turbo-codes" Communications, 1993. ICC 93. Geneva. Technical Program, Conference Record,

IEEE International Conference on Volume 2, 23-26 May 1993 Page(s):1064 - 1070 vol.2.

[3] Gallager, R. G., “ Low Density Parity Check Codes”, Monograph, M.I.T. Press, 1963.

[4] Lun, Shu.; Costello, Daniel J.; *"Error Control Coding 2nd edition"*, Prentice Hall, 2004.

[5] Moberly, Raymond; O'Sullivan, Michael; Waheed, Khurram; *"LDPC Decoder with a Limited-Precision FPGA-based Floating-Point Multiplication Coprocessor"*. Proceedings of SPIE, the International Society for Optical Engineering. ISSN 0277-786X

[6] Krose, Ben.; Smagt, Patrick van der; *"An Introduction to Neural Networks"*, University of Amsterdam: The Netherlands 1996

[7] MacKay, David J. C.; Wilson, Simon T.; Davey, Mathew C.; *"Comparison of Construction of Irregular Gallager Codes"*. IEEE Transaction on Communications, July 1998

[8] Campos, P.G, “MLP networks for classification and prediction with rule extraction mechanism”, Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on, Volume 2, 25-29 July 2004 Page(s):1387 - 1392 vol.2

[9] Benediktsson, Jon A.; Swain, Philip H.; Ersoy, Okan K.; *"Neural Network Approaches Versus Statistical Methods in Classification of Multisource Remote Sensing Data"*, IEEE Transactions on Geoscience and Remote Sensing, Vol 28. Nov 1991.

[10] Zhu, Jihan; Sutton, Peter; *"FPGA Implementation of Neural Networks - a Survey of a Decade of Progress"*, In Proc. 13th Ann. Conf. on Field Programmable Logic and Applications. pp. 1062-1066.

[11] Omondi, Amos R.; Rajapakse, Jagath C. *"FPGA Implementations of Neural Networks"*, The Netherlands: Springer, Page 46, 2006.

[12] Marchesi, M.; Orlandi, G.; Piazza, F.; Uncini, A. "*Fast Neural Networks Without Multipliers"*, Nerual Networks, IEEE Transactions on, 1993 , Page(s): 53 - 62

[13] Ja-LingWu.; Tseng, Yuen-Hsing.; Huang, Yuh-Ming; *"Neural Network Decoders for Linear Block Codes"*. International Journal of Computational Engineering, 2002.