

Decoding of Block Turbo Codes with RBF Networks

Xingcheng Liu^{1,2} and Zhuofeng Chen¹

¹Dept of Electronics and Communications Engineering
Sun Yat-Sen University, 510275 Guangzhou, China

²State Key Laboratory of Mobile Communications
Southeast University, 210096 Nanjing, China

isslxc@mail.sysu.edu.cn

Zhongfeng Wang³ and Paul Cull³

³School of Electrical Engineering and Computer Science
Oregon State University
Corvallis, Oregon 97331, USA

pc@eecs.oregonstate.edu

Abstract - A novel iterative decoding scheme for block Turbo codes (BTC) with RBF neural networks was proposed for the first time. The BTC codes considered here were composed of linear block codes, BCH codes. The algorithm benefited from the parallel processing capability of artificial neural networks and the non-linear classification capability of RBFs. Simulation experiments show that the proposed algorithm decreases decoding complexity, speeds up the decoding process and improves the decoding delay while achieving better performance in bit error rate (BER) and code word error rate (WER). Furthermore, the proposed decoding algorithm could be easily implemented with hardware.

I. INTRODUCTION

In 1993 C. Berrou and his research group proposed a powerful error-correction code, Turbo code, which showed that it was possible to transmit data with an exceptionally low Bit Error Rate (BER) with a signal-to-noise ratio (SNR) per information bit (E_b/N_0), which is close to Shannon's theoretical limit on a Gaussian channel [1][2]. In 1994 R. Pyndiah proposed Block Turbo Codes (BTC) [3][4], which also have low BERs on a Gaussian channel at a high code rate. Later this study was expanded to Reed-Solomon (RS) codes [5] and again low BERs were achieved. All the above results were obtained on condition that long Turbo codes and decoding algorithms such as the Maximum A Posterior (MAP) or its modified version were employed. The cost of these encoding and decoding schemes is rather high in computation complexity and time delay.

Many researchers have contributed much in the field of channel decoding with artificial neural networks (ANN). L. G. Tallini and P. Cull proposed a scheme of using ANN technology to decode Hamming codes and Reed-Muller codes [6]. S.E. El-Khamy used ANN to decode block codes and compared the performance between soft-decision decoding and hard-decision decoding [7]. Due to their parallel processing capability, ANNs are a promising technology for error correction to meet the needs of high data-rate transmission. R. Annauth et al proposed a scheme of using error back propagation (EBP) algorithm to decode Turbo codes [8]. However, they only got rather poor performance results although the decoding complexity was reduced.

The hidden layers of radial basis function (RBF) networks [9][10][11] are capable of transforming input vectors in a low-dimensional vector space to those in a high-dimensional space, which makes it possible to classify them in such a high-dimensional space while it is unable to do so in the low-dimensional one. Since RBF is appropriate for non-linear classification, it is also suitable for Turbo decoding, which was

justified later in our experiments. In the field of equalization with RBF technology, L. Hanzo and M. S. Yee proposed a scheme, in which the information between SISO-RBF equalizer and SISO (Soft-Input Soft-Output) channel decoder was exchanged to accomplish iterative processing [9][10]. As far as we know, there are very few publications in BTC decoding with RBF technology [12].

ANN is a type of large-scale, self-adaptive and non-linear systems. Due to its collective computation capability, ANN is capable of carrying out computing and processing based on codewords or on bits. So it could be employed for BTC decoding more conveniently. Furthermore, ANN is characterized with its capabilities of large-scale analog computing, parallel processing, and self-learning, which could be used to improve decoding performance, reduce decoding complexity and shorten decoding delay. The decoding algorithm proposed here could be easily implemented and employed in wireless 3G and beyond communication systems.

II. RBF DECODER FOR LINEAR BLOCK CODES

RBF model is a multi-layer feed-forward NN in structure, as shown in Fig. 1. For (n, k) linear block codes there are 2^k centers, denoted as t_i ($i=1, 2, \dots, 2^k$). The connection weights between the hidden layer and the output layer are denoted as ω_{ij} ($i=1, 2, \dots, 2^k, j=1, 2, \dots, k$). The input layer in RBF model has n neurons, each of which is indexed with m . The hidden layer has 2^k neurons, where each neuron is numbered with i , and the "Basis Function" of the i -th hidden neuron is denoted as $\varphi(X, t_i)$. Assume the "Basis Function" to be Gaussian function, which is represented as follows:

$$\begin{aligned} \varphi(\mathbf{X}, t_i) &= G(\|\mathbf{X} - t_i\|) = \exp\left(-\|\mathbf{X} - t_i\|^2 / 2\sigma_i^2\right) \\ &= \exp\left(-\sum_{m=1}^n (x_m - t_{im})^2 / 2\sigma_i^2\right) \end{aligned} \quad (1)$$

where $t_i = [t_{i1}, t_{i2}, \dots, t_{i2^k}]$ is the center of the Gaussian function, and σ_i is the variance of this function. The computation of the variance could be carried out as follow:

$$\sigma_1 = \sigma_2 = \dots = \sigma_I = d_{\max} / \sqrt{2I} \quad (2)$$

where $I=2^k$ is the number of hidden neurons, and d_{\max} is the maximum Euclidean distance among the centers.

The RBF decoder for BCH(n, k) codes is shown in Fig. 1. Such an RBF decoder could be viewed as a Multi-Input-Multi-Output (MIMO) system in view of bits, where each input or each output corresponds to 1-bit information. Also, the RBF

decoder could be treated as a Single-Input-Single-Output (SISO) model from the viewpoint of code-words, where its input or output is corresponding to 1-codeword information. In this way, interleavers used in Turbo codes could be configured based on bit interleaving or codeword interleaving. However, it is generally accepted that the permutation performance based on bit-interleaving is better than that based on codeword-interleaving. This is the reason that we will choose the bit-interleaver in our simulation.

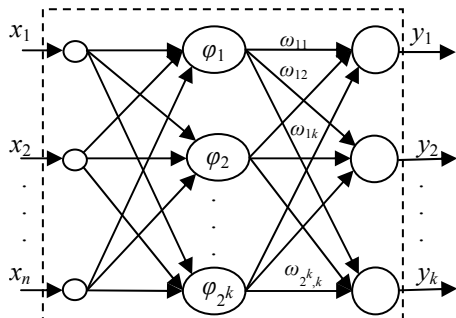


Fig. 1. RBF decoder for BCH(n, k) codes

III. RBF DECODING OF BLOCK TURBO CODES

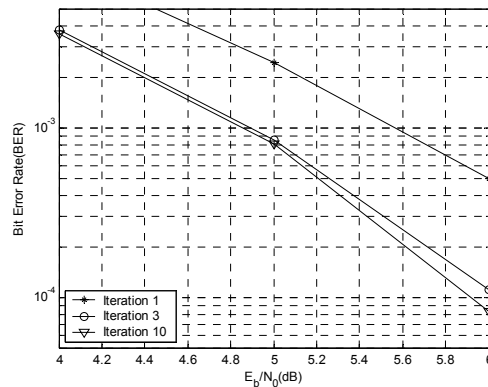
For (n, k) linear block codes the 2^k permissible codes could be treated as the centers in the n -dimension space, and the decoding of the linear block codes could be modeled as non-linear classification in the n -dimension space. That is to say, in the n -dimension space these 2^k centers could be used to partition the space optimally so that any point in the space can be correctly classified as the center to which it belongs.

There are two stages of decoding for BTC codes with RBF technology, i.e., the first stage of training the RBF network and the second stage of decoding the BTC codes with the RBF network. The following gives the details.

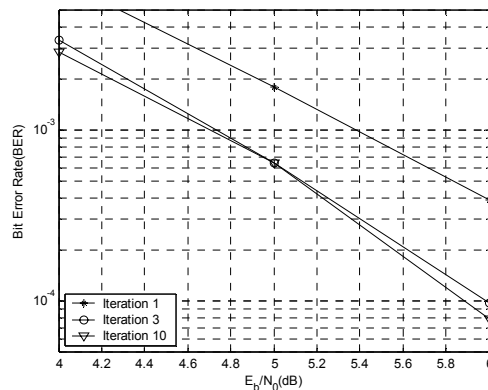
A. Training Stage

Based on the principle of Minimum Mean Square Error(MMSE), the known information is used to train the RBF network, and the weights of the NN are accordingly changed during training to obtain the optimal output. The two component decoders in Turbo codes have to be trained separately. The number of samples for training influences the performance of RBF decoder a lot. There should be neither too few nor too many training samples. In the first case of too few samples, the weights of the RBF network would not be correct, which would lead to poor decoding performance. On the other hand, if too many samples were applied, the performance would not be improved substantially while it would prolong the training. Based on our experiments the training samples should not be less than 20 times the number of centers in the RBF network. Fig. 2 shows the relationship between the performance for BCH(7, 4) code and the number of the training samples under 3 different number of iterations. It shows that the decoding performance with RBF networks improves as the training samples increase. For a BER of 10^{-3} , there is only 0.2dB additional coding gain at iteration 3 when comparing Fig. 2(b) to Fig. 2(a). This result indicates that excessively increasing training samples would not lead to

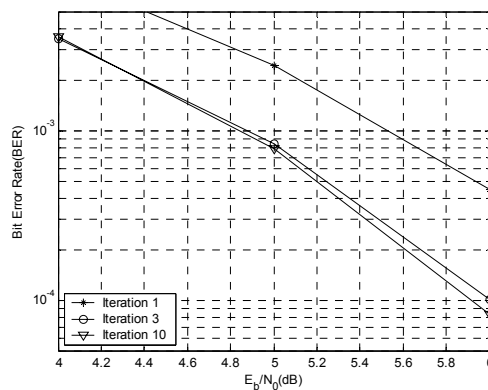
noticeable improvement of the BER performance. When 10 iterations are carried out, the performance of the two scenarios is similar to the case of 3 iterations. Many experiments show, further increasing training samples will not improve the decoding performance greatly after the number of the training samples is up to 50 times the number of centers in RBF networks, which is shown in Fig. 2(c). In other words, the gap of performance resulted from the number of training samples narrows after the training of RBF networks is sufficient.



(a) 30 times the number of centers in RBF nets



(b) 130 times the number of centers in RBF nets



(c) 50 times the number of centers in RBF nets

Fig. 2. Performance of BCH (7, 4) Turbo code with different number of training samples.

B. Decoding Stage

This stage includes two steps, initial decoding and iterative decoding. The information bits are encoded with a Turbo encoder shown in Fig. 3 (BC: Block Codes). It outputs information bits and corresponding check bits, both of which are fed to channels after modulation. During the stage of initialization, information bits and check bits 1 are input to RBF component decoder 1, as shown in Fig. 4. After the first decoding in RBF decoder 1, the decoded information is output and then treated as extrinsic information to the next component decoder after interleaving. This extrinsic information together with check bits 2 is fed to RBF decoder 2. Then the next half-iteration continues until new probabilistic information is obtained. At this point the initial decoding finishes. That is to say one full iteration ends. Next, the probabilistic information output by RBF decoder 2 is fed back to RBF decoder 1 after de-interleaving, and a new cycle of iterative decoding begins. After a pre-defined number of iterations, RBF decoder 2 gives soft-information output and the final decoding output is obtained after de-interleaving and hard decision. It is necessary to mention that RBF decoders 1 and 2 are initialized to zero at the beginning of decoding. Obviously, the RBF decoding model is similar to the original one of the Turbo decoders. And the major difference between them is that RBF decoders for block codes are used instead of the original one with MAP decoding or Soft-Output Viterbi-Algorithm (SOVA) decoding.

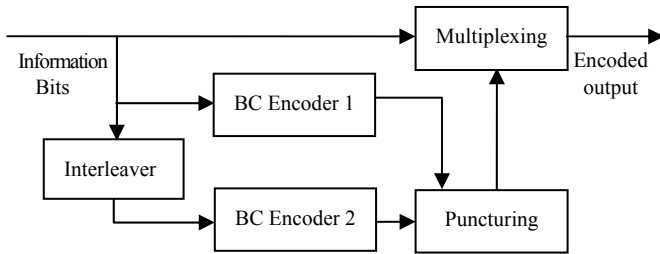


Fig. 3. Block Turbo codes (BTC) encoder

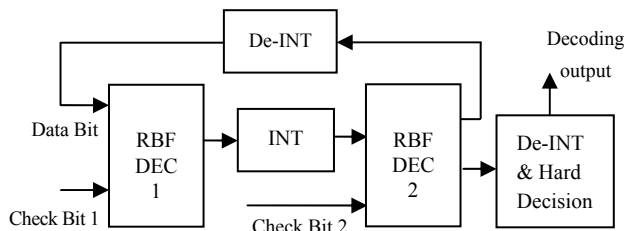


Fig. 4. Iterative decoding of block Turbo codes with RBF NNs

IV. SIMULATION RESULTS AND ANALYSIS

In our experiments the block Turbo codes constructed with BCH(7, 4) codes, BCH(15, 5) codes and BCH(15, 7) codes respectively are used in simulation as shown in Fig. 3. The information bits are grouped into blocks. Each block has 256 units and each unit contains k bits, where k is the length of information bits to be encoded in (n, k) block codes. Therefore, the size of pseudo-random bit interleavers is accordingly chosen to be $256 \times k$ bits. The encoded sequence is sent through

AWGN channel after BPSK modulation. The decoding scheme shown in Fig. 4 is adopted. The RBF network is trained with samples up to 50 times as many as the RBF's centers. The performance of the BCH(7, 4) Turbo code is shown in Fig. 5 under different conditions. It is shown that the performance improves as the iterative number increases. At the BER of 10^{-4} , 0.5dB additional coding gain is achieved at iteration 3 when compared with that at iteration 1. However, the improvement is negligible when further increasing the number of iterations after the third iterative decoding is finished.

After studying the characteristics of RBF network weights, it was found that the absolute values of the weights tend to be the same after being trained sufficiently. Therefore, we could use the weights-fixed scheme to reduce the decoding complexity. The performance of BCH(7, 4) Turbo codes with weights-fixed decoding scheme is shown in Fig. 5. It is shown that the performance with this decoding scheme also improves with increasing iterations. Furthermore, the performance improvement still develops after the third iteration, which hints that this scheme is better than that with the weights-trained mechanism. At the BER of 10^{-4} , 3 iterations bring in 0.6dB coding gain comparing with the first iteration while 10 iterations further lead to 0.7dB additional coding gain. Comparing the two weights-trained and weights-fixed schemes, the former leads to 0.3dB coding gain more than the latter does at the BER of 10^{-2} with 10 iterations. However, a surprising result emerges when decoding at a channel with an increasing SNR. The weights-fixed scheme has 0.3dB coding gain more than the trained one at the BER of 10^{-5} with 10 iterations. Experiments show that the performance with the weights-fixed scheme is degraded at an SNR of less than 4.5dB. However, this performance degradation is not great. When the SNR is larger than 4.5dB, the performance with the weights-fixed scheme is better than that with weights-trained one. Furthermore, the operation complexity with the first scheme is lower than that with the trained scheme and hence it is easier to be implemented with simple hardware.

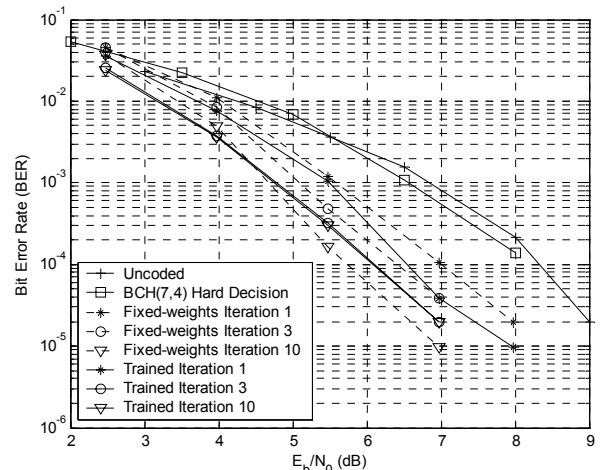


Fig. 5. BER performance of BCH(7, 4) Turbo code in weights-fixed and weights-trained schemes with RBF iterative decoding

The values of the RBF fixed-weights can be obtained through sufficient training of the neural network constructed

for a specific code. Also there is another way to get the weights, where the weights-matrix can be immediately constructed with each center of the RBF network. It is demonstrated in the experiment that the absolute values of RBF weights converge to a certain value for short BCH codes at a certain SNR when the network gets more and more sufficiently trained. In low SNR zone, the absolute values of the mean and variance of the weights are relatively large, while these values decrease as the SNR increases, the absolute value of the mean tends to 1, and the absolute value of the variance tends to 0. These results can explain why there is a loss in BER performance in the low SNR zone, while in the high SNR zone the performance is not degraded and can be improved further after quite a few iterations. On the contrary, there is only slight improvement in error correction performance as the number of iterations grows large in the case of the trained scheme.

Fig. 6 shows the BER performance of BCH(15, 5) Turbo code and BCH(15, 7) Turbo code with the weights-fixed decoding scheme. It is shown that the performance improves with increasing decoding iterations. For the BCH(15, 5) Turbo code, 1.4dB additional gain is achieved at the BER of 10^{-4} with 10 iterations comparing with 3 iterations. The BCH(15, 7) Turbo code can obtain 0.5dB relative gain at the BER of 10^{-4} with 10 iterations in contrast with 3 iterations. Among the block Turbo codes discussed here, the BER performance of BCH(15, 5) Turbo code improves obviously with increasing the number of iterations or the value of SNR, and its computation complexity is acceptable, although its code rate is rather low. For BCH(7, 4) Turbo code, the BER performance is not satisfactory, but its complexity is low. Therefore, it is safe to say that BCH(15, 7) Turbo code lies in the middle among the block Turbo codes considered here in view of both performance and complexity.

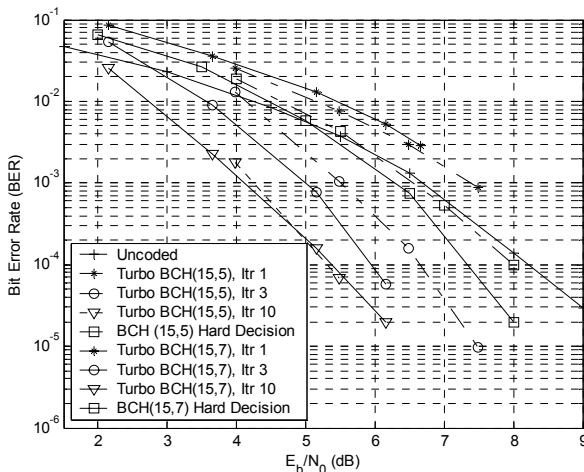


Fig. 6. BER performance of BCH(15, 5) Turbo code and BCH(15, 7) Turbo code in weights-fixed schemes with RBF iterative decoding

The advantages of the BTC decoder with RBF-assisted decoding could be summarized as following:

(1) Low decoding complexity. The complexity for (n, k) block Turbo codes with traditional decoding algorithms like MAP and Log-MAP is rather high, which is shown in Table I. For (n, k) block Turbo codes there are 2^k hidden neurons in the

corresponding RBF-assisted decoding network. There are n additions (or subtractions) and $n+2$ multiplications (or divisions) at each hidden neuron. There are 2^k additions and 2^k multiplications in each of the n outputs. So there are a total of $2n \times 2^k$ additions and $2n \times 2^k$ multiplications in the RBF decoder. The comparison of the complexities among the three decoding algorithms is shown in Table I. For BCH(7, 4) Turbo code, the complexity with RBF decoding is 1/3 as much as that with MAP. For BCH(15, 7) Turbo code, the complexity with RBF decoding is even lower, 1/5 of that with MAP. For BCH(15, 5) Turbo code, the complexity with RBF decoding is much lower, only 1/300 as low as that with MAP. For the codes with low code rates (high ratio of n and k), the complexity with RBF decoding is much lower than that with MAP or Log-MAP.

TABLE I
COMPLEXITY COMPARISON BETWEEN THE DECODING ALGORITHM WITH RBF NETWORKS AND THE TRADITIONAL TURBO DECODING ALGORITHMS (UNITS: NUMBER OF OPERATIONS)

	MAP	Log-MAP	RBF Decoding
Finding Maximum	0	$5 \times 2^n - 2$	0
Additions / Subtractions	4×2^n	$15 \times 2^n + 9$	$2n \times 2^k$
Multiplications / Divisions	$6 \times 2^n + 1$	8	$(2n+2) \times 2^k$
Searching	0	$5 \times 2^n - 2$	0

(2) Low decoding delay. The decoding operations in RBF networks can be parallelized while those with the MAP or Log-MAP algorithms are performed in cascade or serially, which is difficult to be made run in parallel. Obviously, parallel processing is faster than cascaded processing. For (n, k) block code, the decoding delay with RBF is shorter than that with MAP or Log-MAP algorithms. This is because the decoding complexity with RBF is 1/3 and 1/5 times that of MAP for BCH(7, 4) Turbo code and BCH(15, 7) Turbo code respectively. Therefore, the decoding delay with RBF is definitely shorter than that with MAP decoding and even shorter when considering the parallel processing capability in RBF networks.

(3) Strong error correction performance. The WER performance with the weights-fixed scheme is shown in Fig. 7. It is shown that the WER is less than k times the value of the BER in low SNR zone, or it is approximately equal to the BER value in the high SNR zone. For BCH(7, 4) Turbo code, $WER \leq 4 \times BER$ while $WER \leq 5 \times BER$ and $WER \leq 7 \times BER$ for BCH(15, 5) and BCH(15, 7) Turbo codes respectively. Therefore, the WER performance will not degrade noticeably. For the weights-fixed decoding scheme and 10 iterations, the WERs are as low as 10^{-3} at $E_b/N_0 = 5.3$ dB for BCH(7, 4) Turbo code, at $E_b/N_0 = 5.1$ dB for BCH(15, 7) Turbo code, and at $E_b/N_0 = 4.9$ dB for BCH(15, 5) Turbo code.

(4) Simple hardware implementation. Since the RBF model is simple and symmetrical, and its decoding complexity is low, it is convenient to be implemented with DSP and FPGA devices.

V. CONCLUSION

In this paper a novel iterative decoding scheme with RBF networks was proposed. This algorithm was applied in decoding of block Turbo codes. Simulation results showed that

it is the parallel processing capability of the RBF networks and the powerful non-linear classification capability that lead to the lower decoding complexity and shorter decoding delay for the block Turbo codes. Meanwhile, it is also shown that the BER and WER performance is promising with this decoding algorithm. Moreover, the simple model and parallel processing capability of the RBF make the proposed decoding algorithm easy to be implemented with hardware.

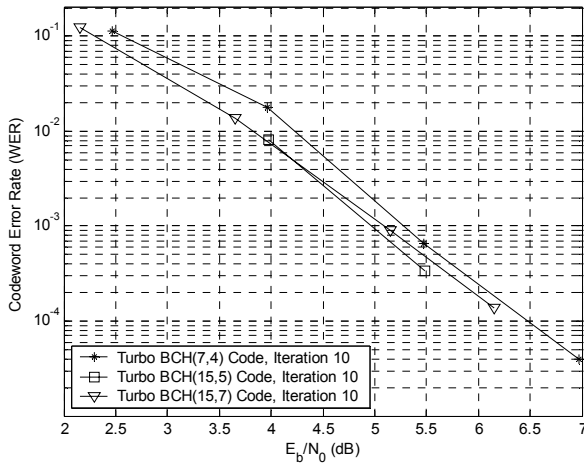


Fig. 7. WER performance with RBF decoding for weights-fixed schemes

ACKNOWLEDGMENT

We are grateful to Professor Bella Bose for his support and helpful suggestions on this paper. This work is supported by the Natural Science Foundation of Guangdong Province (No: 04009739) and the State Key Laboratory of Mobile

Communications (SLMC), Southeast University, Nanjing (No: A0403), China.

REFERENCES

- [1] C. Berrou, A. Clavier, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes," IEEE International Conference on Communications, vol. 2, pp. 1064-1070, May 1993.
- [2] C. Berrou, A. Glavieux, "Near optimum error correcting coding and decoding: Turbo-codes," IEEE Transactions on Communications, vol. 44, no. 10, pp. 1261-1271, October 1996.
- [3] R. Pyndiah, A. Glavieux, A. Picart, and S. Jacq, "Near optimum decoding of product codes," IEEE Global Telecommunications Conference (GLOBECOM' 94), vol. 1, pp. 339-343, November 28- December 2, 1994.
- [4] R. Pyndiah, "Near-optimum decoding of product codes: block Turbo codes," IEEE Transactions on Communications, vol. 46, no. 8, pp. 1003-1010, August 1998.
- [5] O. Aitsab, R. Pyndiah, "Performance of Reed-Solomon block Turbo code," IEEE Global Telecommunications Conference (GLOBECOM' 96), vol. 1, pp. 121-125, November 1996.
- [6] L. G. Tallini, P. Cull, "Neural Networks for Decoding Error-correcting Codes," Italian Journal of Pure and Applied Mathematics, vol. 10, pp. 91-106, 2001. A brief version of this paper also appeared in IEEE Technical Applications Conference and Workshops (Northcon 95), pp. 89-94, October 1995.
- [7] S. E. El-Khany, E. A. Youssef, H. M. Abdou, "Soft decision decoding of block codes using artificial neural network," Proc. IEEE Symposium on Computers and Communications, pp. 234-240, June 1995.
- [8] R. Annauth, H. C. S. Rughooputh, "Neural network decoding of Turbo codes," International Joint Conference on Neural Networks (IJCNN' 99), vol. 5, pp. 3336-3341, July 1999.
- [9] M. S. Yee, B. L. Yeap, L. Hanzo, "Radial basis function assisted Turbo equalization," IEEE 51st Vehicular Technology Conference (VTC 2000 Spring), vol. 1, pp. 640-644, May 2000.
- [10] M. S. Yee, S. X. Ng, L. Hanzo, "Iterative radial basis function assisted Turbo equalisation of various coded modulation schemes," IEEE 55th Vehicular Technology Conference (VTC Spring 2002), vol. 4, pp. 1705-1709, May 2002.
- [11] J. Gao, Principles of Artificial Neural Networks and Simulation Examples, Beijing: China Machine Press, pp. 56-60, 2003.
- [12] D. Liu, Principles of Turbo Codes and Application Technology, Beijing: Publishing House of Electronics Industry, pp. 94 & 242-243, 2004