High-Order Perceptrons for Decoding Error-Correcting Codes

Yuen-Hsien Tseng, student member, IEEE, and Ja-Ling Wu, Member, IEEE
Department of Computer Science and Information Engineering
National Taiwan University, Taipei, Taiwan, R.O.C.

Abstract

In this paper , we prove that the single-error correcting (2^n-1, 2^n-1-n) Hamming code and its extended single-error correcting/double-error detecting (2^n, 2^n-1-n) code can be decoded by low-complexity single-layer perceptrons which use high-order polynomials as their discriminant functions, and that multiple-error correcting codes can be decoded by two-layer networks with high-order perceptrons in the first layer and linear perceptrons in the second layer.

I. Introduction

Previous works on the application of neural networks for decoding error-correcting codes include Hopfield nets for decoding graph-theoretic codes [1], and Counter-propagation networks (CPN) and backpropagation networks (BP) for Hamming codes [2]. The decoding of a graph-theoretic code is formulated as a problem of searching for the global minimum of a corresponding energy function. This formulation allows the Hopfield net to solve the decoding problems as it solves the optimization problems [3]. However, decoding a received codeword needs a recalculation of the weights of the network. Besides, Hopfield network does not guarantee to find a global minimum solution. Although without the overhead of training costs, using Hopfield nets for decoding is ineffective at the current stage [4]. The CPN and BP for (7, 4) Hamming code solve the decoding problem in a direct way; they both require the number of hidden units equal to the number of legal codewords for reliable decoding.

Our approach to the decoding problem makes use of high-order perceptrons, which have polynomials rather than conventional linear functions as their discriminants. The output function of a high-order perceptron can be described as [5]

$$z = \mathrm{sgn}(\, g(X)\,) \tag{1}$$

where $X = [x_1, x_2, ..., x_n] \in \{1, -1\}^n$ is an input pattern, sgn is the sign function: sgn(a)=1 if a>0, –1 if a<0, and undefined if a = 0, and g is a rth-order polynomial function, i.e.,

$$g(X) = w_1 f_1(X) + w_2 f_2(X) + ... + w_N f_N(X) + w_0 \tag{2}$$

where each product term $f_i(X)$ is of the form:

$$x_{k_1}^{n_1} x_{k_2}^{n_2} ... x_{k_r}^{n_r} \tag{3}$$

0-7803-0559-0 /92 $3.00 © 1992 IEEE

$k_1, k_2, ..., k_r \in \{1, ..., n\}$ and $n_1, n_2, ..., n_r \in \{1, 0\}$. The high-order perceptrons are able to decode the entire class of single-error correcting $(2^n-1, 2^n-1-n)$ Hamming code and single-error correcting/double-error detecting $(2^n, 2^n-1-n)$ extended Hamming code. The complexity of the one-layer network decoder is relatively low. As it will be shown, only n+1 weights for each perceptron are needed for the Hamming code and n+2 for the extended one. It is also shown that having a two-layer structure with high-order perceptrons in the first layer and linear perceptrons in the second layer, decoding of multiple-error correcting codes is also possible.

The general idea of decoding an error-correcting code could be described as follows [6]. A systematic binary (M, p) code, in which there are p information bits and n = (M–p) parity bits for each codeword, can be described by a *parity-check matrix* $H=[h_{ij}]_{n*M}$, $h_{ij} \in \{1, 0\}$. Let $A = [a_1, a_2, ..., a_p]$ be an information word. After A being encoded and transmitted, the receiving end receives the codeword $V = [v_1, v_2, ..., v_M]$. The codeword V is then multiplied (modulo 2) by the parity check matrix H to result in a syndrome vector $S = [s_1, s_2, ..., s_n]$, where

$$s_i = \sum_{k=1}^{M} v_k h_{ik} \mod 2 \qquad (4)$$

The syndrome S provides the information to decode the codeword V: If S is a zero vector, then there is no error, the parity bits v_i, i = p+1, ..., M, are discarded and the information bits are directly accessed, i.e., $a_i = v_i$, i = 1, ..., p. If a single error occurs, in which case S matches the *jth* column vector of H, then the *jth* bit v_j is dirty and has to be complemented. If more than one errors occur in V, then S matches the sum (modulo 2) of the corresponding columns of H. In the case that S is nonzero and matches no combinations of the columns of H, we say an erroneous codeword is detected, but the incorrect positions in V can not be located.

II. Decoding of Single-error Correcting Codes

The class of $(2^n-1, 2^n-1-n)$ Hamming code is a single-error correcting code with each codeword having 2^n-1-n information bits and n parity bits [7].The decoding rule for this code is simply to compute S and see if S matches any of the column of H. If S is zero, then $a_i=(v_i \oplus 0)$, i =1, ..., 2^n-1-n, where $\oplus$ denotes XOR operation. If S matches column j of H, then $a_j = (v_j \oplus 1)$ and $a_i = v_i$, i =1, ...,2^n-1-n, $i \neq j$. This rule can be expressed in a concise Boolean formula as

$$a_j = v_j \oplus \prod_{i=1}^{n} (s_i h_{ij} + \neg s_i \neg h_{ij}) \qquad (5)$$

for j = 1, ..., 2^n-1-n. The expression enclosed by the parentheses is an equivalence test and the AND operation over n elements is to see if S matches column j of H. With a direct transform from the above expression to a polynomial, we have the following theorem.

Theorem 1: The $(2^n-1, 2^n-1-n)$ Hamming code can be decoded by one-layer high-order perceptrons with only n+1 product terms for each polynomial discriminant function.

Before proving the theorem, let us first define two notations. Suppose a is a Boolean expression that results in a binary value and x and y are two expressions that result in a bipolar value and a real number, respectively. The notation 'a ‹–› x' denotes the relation between a and x as a = 1 iff x = –1, a = 0 iff x = 1, while the notation 'a ‹=› y' denotes a = 1 iff y < 0 and a = 0 iff y > 0. The following four lemmas can be proved by considering all possible cases and are thus given without proof.

Lemma 1: If a ‹–› x and b ‹=› y, then $a \oplus b$ ‹=› xy.

Lemma 2: If a_i ‹–› x_i for i = 1, ..., n, then

$$\prod_{i=1}^{n} a_i \;\langle\!=\!\rangle\; \sum_{i=1}^{n} x_i + (n-1)$$

or

$$\prod_{i=1}^{n} a_i \;\langle\!-\!\rangle\; \mathrm{sgn}\left(\sum_{i=1}^{n} x_i + (n-1) \right)$$

Lemma 3: If a ‹–› x and $h \in \{1, 0\}$ is a binary value, then $(ah + \neg a \neg h) \langle\!-\!\rangle (xh - x(1-h))$.

Lemma 4: If a_i ‹–› x_i and $h_i \in \{1, 0\}$ for i = 1, ..., n, then

$$\sum_{i=1}^{n} a_i h_i \mod 2 \;\langle\!-\!\rangle\; \prod_{i=1}^{n} x_i^{h_i}$$

where the expression on the left hand side denotes the XOR operation on $a_i h_i$, i = 1, ..., n.

By assuming that the binary variable v_i and the bipolar variable x_i satisfy the relation $v_i \langle\!-\!\rangle x_i$, we now give the proof of the theorem.

Proof: Let M be the codeword length. To decode a Hamming code, we only have to find a set of polynomial functions g_j, j = 1, ..., M–n, that satisfy a_j ‹=› g_j, where a_j is defined in (5). By applying Lemma 1 through Lemma 4, these polynomial functions g_j satisfying the relation are

$$g_j = x_j \left\{ \sum_{i=1}^{n} \left[(t_i h_{ij} - t_i(1 - h_{ij}) \right] + (n-1) \right\} \tag{6}$$

where

$$t_i = \prod_{k=1}^{M} x_k^{h_{ik}} \tag{7}$$

Q.E.D.

Example 1: The simplest nontrivial Hamming code is a (7, 4) Hamming code, which can be described by the following parity-check matrix

$$H = \begin{bmatrix} 1\,1\,0\,1\,1\,0\,0 \\ 1\,0\,1\,1\,0\,1\,0 \\ 0\,1\,1\,1\,0\,0\,1 \end{bmatrix}$$

So the polynomial functions for the neural network decoder are

$$g_1(X) = x_1(2 + x_1x_2x_4x_5 + x_1x_3x_4x_6 - x_2x_3x_4x_7) \quad (8)$$
$$= 2x_1 + x_2x_4x_5 + x_3x_4x_6 - x_1x_2x_3x_4x_7$$
$$g_2(X) = x_2(2 + x_1x_2x_4x_5 - x_1x_3x_4x_6 + x_2x_3x_4x_7) \quad (9)$$
$$= 2x_2 + x_1x_4x_5 - x_1x_2x_3x_4x_6 + x_3x_4x_7$$
$$g_3(X) = x_3(2 - x_1x_2x_4x_5 + x_1x_3x_4x_6 + x_2x_3x_4x_7) \quad (10)$$
$$= 2x_3 - x_1x_2x_3x_4x_5 + x_1x_4x_6 + x_2x_4x_7$$
$$g_4(X) = x_4(2 + x_1x_2x_4x_5 + x_1x_3x_4x_6 + x_2x_3x_4x_7) \quad (11)$$
$$= 2x_4 + x_1x_2x_5 + x_1x_3x_6 + x_2x_3x_7$$

As (8) through (11) show, one can alternatively implement the decoder by using n = 3 common product terms and thus obtains a network of much lower complexity.

III. Decoding of Single-error Correcting/Double-error Detecting Codes

The parity-check matrix H' for the single-error correcting/double-error detecting (2^n, $2^n - 1 - n$) extended Hamming code can be derived by appending one more row and column to the parity-check matrix H of the Hamming code as follows [7]

$$H' = \begin{bmatrix} H & 0 \\ 1 & 1 \end{bmatrix}_{(n+1)x2^n} \quad (12)$$

where $\mathbf{1} = [1, ..., 1]$ is a row vector of 1 by n, $\mathbf{0} = [0, ..., 0]^T$ a column vector of $2^n - 1$ by 1. Let S be the original syndrome vector and s_{n+1} be a parity check over the whole messages, i.e., the sum of all v_i (modulo 2). Then any single error will produce the right S and set s_{n+1} to 1; while a double error causes a nonzero S but makes $s_{n+1} = 0$.

Similar to (5), the Boolean functions for the information bits are

$$a_j = v_j \oplus \left[s_{n+1} \prod_{i=1}^{n} (s_i h_{ij} + \neg s_i \neg h_{ij}) \right] \quad (13)$$

$j = 1, ..., 2^n - 1 - n$, and for the bit a_0 indicating a double error is

$$a_0 = \neg s_{n+1} \sum_{i=1}^{n} s_i \quad (14)$$

The polynomial discriminant functions corresponding to a_j in (13) can be derived by applying Lemma 1 through Lemma 4:

$$g_j = x_j \left\{ \sum_{i=1}^{n} [(t_i h_{ij} - t_i(1 - h_{ij})] + t_{n+1} + n \right\} \quad (15)$$

where t_i is defined similarly in (7), while the one for indicating a double error is

$$g_0 = t_{n+1} \left[\sum_{i=1}^{n} t_i - (n-1) \right] + \sum_{i=1}^{n} t_i - (n-1) - t_{n+1}$$
$$= t_{n+1} \sum_{i=1}^{n} t_i + \sum_{i=1}^{n} t_i - n\, t_{n+1} - (n-1) \quad (16)$$

by the following two lemmas:

Lemma 5: If $a_i \leftrightarrow x_i$ for i = 1, ..., n, then

$$\sum_{i=1}^{n} a_i \Leftrightarrow \sum_{i=1}^{n} x_i - (n-1)$$

Lemma 6: If $a \leftrightarrow x$, $b \Leftrightarrow y$, and $|y| > 0.5$, then $ab \Leftrightarrow y(1-x) + x = -xy + y + x$.

Theorem 2: The $(2^n, 2^n - 1 - n)$ extended Hamming code can be decoded by high-order perceptrons with 2n+2 weights for one perceptron and n+2 for each of the remaining ones.

IV. Decoding of Multiple-error Correcting Codes

Decoding of a t-error correcting code needs to test whether the syndrome S matches any combination of up to t columns of H or not. For the case of t = 2, the Boolean expression for doing this can be written as

$$a_j = v_j \oplus \left[\prod_{i=1}^{n} (s_i h_{ij} + \neg s_i \neg h_{ij}) + \sum_{k=1}^{M} \prod_{i=1}^{n} (s_i h_{ijk} + \neg s_i \neg h_{ijk}) \right] \quad (17)$$

where $h_{ijk} = h_{ij} + h_{ik} \bmod 2$. The first AND term is to see if S matches column j of H and the second AND term tests if S matches the sum of column j and k of H, over all k.

From Lemma 2 through Lemma 5, we can transform the expression in the square bracket in (17) into the following function

$$G_j = \mathrm{sgn}\left(\sum_{i=1}^{n} [(t_i h_{ij} - t_i(1 - h_{ij})] + (n-1)\right) + \sum_{k=1}^{M} \mathrm{sgn}\left(\sum_{i=1}^{n} [(t_i h_{ijk} - t_i(1 - h_{ijk})] + (n-1)\right) - M$$

such that $a_j \Leftrightarrow x_j G_j$ by Lemma 1, or alternatively $a_i \leftrightarrow x_i \mathrm{sgn}(G_j)$.

The expression $x_i \mathrm{sgn}(G_j)$ can be implemented by a two-layer network with high-order perceptrons in the first layer followed by a linear perceptron in the second layer. The outputs of the first layer are summed and clamped with a threshold M by the linear perceptron. This output is then polarized by the bipolar variable x_j to produce a correct result.

Decoding of up to t errors can be derived similarly, in which case more high-order perceptrons are needed since the number of combinations of up to t columns of H increases as t grows.

Theorem 3: A multiple-error correcting code can be decoded by a two-layer network with high-order perceptrons in the first layer and linear perceptrons in the second layer.

V. Conclusion

We have shown that the Hamming code and its extended one can be decoded by single-layer high-order perceptrons with unexpectedly low network complexity, and illustrated how to decode multiple-error correcting codes by two-layer networks. Since the neural network decoder can operate in real domain as well, it is a kind of soft-decision decoder, which uses more information passed from the demodulator and thus in general gives better performance than its hard-decision counterpart.

References

[1] J. Bruck and M. Blaum, "Neural Networks, Error-Correcting Codes, and Polynomials over the Binary n-Cube", IEEE Trans. on Inform. Theory, vol.35, pp.976-987, 1989.

[2] A. D. Stefano and O. Mirabella, "On the Use of Neural Networks for Hamming Coding", IEEE, Int. Sym. on Circuits and Systems, pp.1601-1604, 1991.

[3] J. J. Hopfield and D. W. Tank, "Neural Computation of Decisions in Optimization Problems", Biol. Cybernet. 52, pp.141-152, 1985.

[4] Hsiu-Hui Lin, Ja-Ling Wu, and Li-Chen Fu, "Solving Problems of Maximum Likelihood Decoding of Graph Theoretic Codes via a Hopfield Neural Network", IEEE, Int. Sym. on Circuits and Systems, pp.1200-1203, 1991.

[5] Nils J.Nilsson, *Learning Machines: Foundations of Trainable Pattern-Classifying Systems*, McGraw-Hill, 1965.

[6] R. E. Blahut, *Theory and Practice of Error Control Codes*, Addison-Wesley, 1983.

[7] R. W. Hamming, *Coding and Information Theory*, Prentice-Hall, 1986.