International Journal of Electronics
Publication details, including instructions for authors and subscription information:
http://www.informaworld.com/smpp/title~content=t713599654

Decoding Reed-Muller codes by multi-layer perceptrons

Yuen-Hsien Tseng [a]; Ja-Ling Wu [a]
[a] Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, Republic of China

Online Publication Date: 01 October 1993

To cite this Article Tseng, Yuen-Hsien and Wu, Ja-Ling(1993)'Decoding Reed-Muller codes by multi-layer perceptrons',International Journal of Electronics,75:4,589 — 594

To link to this Article: DOI: 10.1080/00207219308907134

URL: http://dx.doi.org/10.1080/00207219308907134

INT. J. ELECTRONICS, 1993, VOL. 75, NO. 4, 589–594

Decoding Reed–Muller codes by multi-layer perceptrons

YUEN-HSIEN TSENG† and JA-LING WU†

The decoding of rth-order Reed–Muller codes of blocklength 2^m, capable of correcting up to $(1/2)2^{m-r}-1$ errors, by use of multi-layer perceptrons is illustrated The multi-layer perceptrons used have all their weights either $+1$ or -1 and all thresholds integers, thus making them very suitable for hardware implementation.

1. Introduction

Error-correcting codes, having wide applications in data transmission, data storage, and fault-tolerance computing, is to protect information from accidental errors. In this letter, we consider the decoding of a class of error-correcting codes called Reed–Muller codes. The Reed–Muller codes, used to transmit the Mariner photographs of Mars in 1972 (Blahut 1983), cover a wide range of rate and minimum distance. For each integer m and $r<m$, there is a Reed–Muller code of blocklength $n=2^m$ and information length, k,

$$k=\binom{m}{0}+\binom{m}{1}+\cdots+\binom{m}{r}$$

capable of correcting up to $(1/2)2^{m-r}-1$ errors. The decoding rule of Reed–Muller codes involves only parity and majority operations, both of which can be implemented by multi-layer perceptrons, as is shown below.

A basic element in a multi-layer perceptron can be described as

$$z=\operatorname{sgn}(g(X))$$

In the above equation, bipolar vector $X=[x_1, x_2, \ldots, x_n]\in\{+1, -1\}^n$ comes from the network input (or from the output of a previous layer); sgn is a sign function: $\operatorname{sgn}(a)=1$ if $a>0$, -1 if $a\le 0$; and g is a linear discriminant function:

$$g(X)=w_0+w_1x_1+w_2x_2+\cdots+w_nw_n$$

where w_i are called weights and w_0 is a threshold. The majority operations is to determine which one dominates the other in a sequence of $+1$ and -1. If $B=\{x_1, x_2, \ldots, x_n\}$ denotes this sequence, then a single perceptron is able to perform the majority operation (Widrow and Winter 1988):

$$\operatorname{maj}(x_1, x_2, \ldots, x_n)=\operatorname{sgn}\left(\sum_{i=1}^{n} x_i\right)$$

Received 2 March 1993; accepted 10 May 1993.

†Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, Republic of China.

The parity problem (or equivalent modulo-2 addition) is to determine if there is an odd number of 1 in a sequence of 0 and 1. A two-layer perceptron for the parity problem has been presented by Rumelhart *et al.* (1986). Here we reformulate the network such that it operates in bipolar domain and its weights are $+1$ or -1, thresholds are integers, and output unit is linear. Specifically, for a bipolar sequence B, define

$$S=n-\sum_{i=1}^{n} x_i$$

$S\in\{0, 2, 4, \ldots, 2n\}$. Let the parity function $P(x_1, x_2, \ldots, x_n)$ output -1 if there is an odd number of -1, and $+1$ otherwise. Then we have

$$P(x_1, x_2, \ldots, x_n)=1-\sum_{k=1}^{\lfloor \frac{n+1}{2} \rfloor}[\operatorname{sgn}(4k-1-S)+\operatorname{sgn}(S-(4k-3))]$$

where the value in the square bracket is 2 if $S=2(2k-1)$, and 0 otherwise. Note this is a two-layer network with one hidden layer of approximately n units and one output unit which needs no sign function.

2. Decoding of Reed-Muller codes

A Reed-Muller code is best defined by a procedure for constructing a non-systematic generator matrix that will prove convenient for decoding. To construct the generator matrix, first define the vector product of two vectors by a component-wise multiplication. That is, let $A=[a_1, a_2, \ldots, a_n]$ and $B=[b_1, b_2, \ldots, b_n]$, then $AB=[a_1b_1, a_2b_2, \ldots, a_nb_n]$. The generator matrix for the rth-order (n, k) Reed-Muller code is defined as an array of blocks

$$G=\begin{bmatrix} G_0 \\ G_1 \\ \vdots \\ G_r \end{bmatrix}$$

where G_0 is the vector of length 2^m containing all ones; G_1, an m by 2^m matrix, has each binary representation of j ranging from 0 to 2^m-1 appearing once as a column, with zero in the leftmost column and low-order bits in the bottom row; and G_r is constructed from G_1 by taking its rows to be all possible vector products of any r rows of G_1.

An example is the zeroth-order $(n, 1)$ Reed-Muller code. It is a simple repetition code and is decoded trivially by a majority vote. As another example, let $m=4$, $n=16$, and $r=3$. Then

$$G_0=\begin{bmatrix}1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1\end{bmatrix}=\begin{bmatrix}B_{-1}\end{bmatrix}$$

$$G_1=\begin{bmatrix}0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\\0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1\\0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1\\0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1\end{bmatrix}=\begin{bmatrix}B_3\\B_2\\B_1\\B_0\end{bmatrix}$$

G_2 has $\binom{4}{2}$ rows since G_1 has four rows,

$$G_2 = \begin{bmatrix} 0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1 \\ 0&0&0&0&0&0&0&0&0&0&1&1&0&0&1&1 \\ 0&0&0&0&0&0&0&0&0&1&0&1&0&1&0&1 \\ 0&0&0&0&0&0&1&1&0&0&0&0&0&0&1&1 \\ 0&0&0&0&0&1&0&1&0&0&0&0&0&1&0&1 \\ 0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1 \end{bmatrix} = \begin{bmatrix} B_3B_2 \\ B_3B_1 \\ B_3B_0 \\ B_2B_1 \\ B_2B_0 \\ B_1B_0 \end{bmatrix}$$

and G_3 has $\binom{4}{3}$ rows

$$G_3 = \begin{bmatrix} 0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1 \\ 0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1 \\ 0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1 \end{bmatrix} = \begin{bmatrix} B_3B_2B_1 \\ B_3B_2B_0 \\ B_3B_1B_0 \\ B_2B_1B_0 \end{bmatrix}$$

The generator matrix for this third-order (16, 15) Reed-Muller code is

$$G = \begin{bmatrix} G_0 \\ G_1 \\ G_2 \\ G_3 \end{bmatrix}$$

In fact, it is simply a parity-check code. Note we have given each row in the above a label to indicate its relation with the rows of G_1 for later convenience.

The Reed algorithm is designed specially for decoding Reed-Muller codes (Blahut 1985). The Reed algorithm is unusual as compared to most decoding algorithms for most codes in that it recovers the information bits directly from the received word without computing the syndrome vectors. The basic idea is to construct a decoder for an rth-order Reed-Muller code by reducing it to a decoder for an $(r-1)$th-order Reed-Muller code. Because the 0th-order Reed-Muller code can be decoded by majoritty vote, the complete decoder is then established by induction. A more detail description of this idea is given below.

Let the information vector be broken into $r+1$ segments, written $A=[A_0, A_1, \ldots, A_r]$, where segment A_i contains $\binom{m}{i}$ information bits. Each segment multiplies one block of G. The encoding can be represented as a block vector-matrix product

$$C = AG \bmod 2 = [A_0, \ldots, A_r] \begin{bmatrix} G_0 \\ \vdots \\ G_r \end{bmatrix} \bmod 2$$

The information bits are distributed into the codeword C. The received word is $V=[v_0, v_1, \ldots, v_{n-1}]=C+E$, where E is an error pattern. If we can recover A_r from V, then we can compute their contribution to the received word and subtract this contribution. That is

$$V'=V-A_rG_r \bmod 2=[A_0, \ldots, A_{r-1}]\begin{bmatrix} G_0 \\ \vdots \\ G_{r1} \end{bmatrix}+E \bmod 2$$

This reduces the problem to that of decoding an $(r-1)$th-order Reed-Muller code.

As to recovering A_r from V, consider decoding the information bit a_j which multiplies one row of G_r. This is decoded by setting up 2^{m-r} linear check sums in the 2^m received bits; each such check sum involves 2^r bits of the received word, and each received bit is used in only one check sum. The check sums will be formed so that a_j contributes to only one term of each check sum, and every other information bit contributes to an even number of terms in each check sum. Hence, each check sum is equal to a_j in the absence of errors. But if there are at most $(1/2)2^{m-r}-1$ errors, the majority of the check sums will still equal a_j.

Specificially, if a_j corresponds to the row of G_r labelled $B_{i_1}B_{i_2}\ldots B_{i_r}$, one of the 2^{m-r} check sums starting with v_k, derived from the description given by Wosley Patterson and Weldom (1972), is expressed as follows.

$$\begin{aligned} \hat{a}_j(s)=v_k+ & \quad \} \binom{r}{0} \text{ term} \\ v_{k+2^{i_1}}+v_{k+2^{i_2}}+\cdots+v_{k+2^{i_r}}+ & \quad \} \binom{r}{1} \text{ terms} \\ v_{k+2^{i_1}+2^{i_2}}+\cdots+v_{k+2^{i_{r-1}}+2^{i_r}}+ & \quad \} \binom{r}{2} \text{ terms} \\ +\cdots+ & \quad \cdots \\ v_{k+2^{i_1}+2^{i_2}+\cdots+2^{i_r}} & \quad \} \binom{r}{r} \text{ term} \\ \bmod 2 & \end{aligned}$$

where s ranges from 1 to 2^{m-r}. The information bit a_j is obtained by taking the majority of these $\hat{a}_j(1)$. The received bit v_j without the contribution of the information bits in segment A_r can then be computed by

$$v'_j=v_j-\sum_{i=r'}^{r''} a_i g_{ij} \bmod 2$$

where $g_{ij}\in\{1,0\}$ is an entry of G, and

$$r' = \binom{m}{0} + \binom{m}{1} + \cdots + \binom{m}{r-1}$$

$$r'' = r' + \binom{m}{r} - 1$$

Because subtraction in modulo 2 is equivalent to addition, v'_j is rewritten as

$$v'_j = v_j + \sum_{i=r'}^{r''} a_i g_{ij} \bmod 2$$

Since the above decoding algorithm involves only addition and majority operations over binary elements, decoding of Reed-Muller codes by multi-layer perceptrons is now in order.

Theorem

An rth-order Reed-Muller code can be decoded by a $2(r+1)$-layer perceptron. The Reed-Muller decoder is implemented by the foregoing parity networks followed by the majority networks. Since the output of the parity network is linear, it can be immediately directed to the input of the majority networks. This makes it a two-layer perceptron for decoding each information segment.

Example

The first-order (8, 4) Reed-Muller code, or equivalently the single-error correcting/double error detecting Hamming code, can be decoded by a multi-layer perceptron with four outputs:

$$z_3 = \text{sgn}\,(P(x_0, x_1) + P(x_2, x_3) + P(x_4, x_5) + P(x_6, x_7))$$

$$z_2 = \text{sgn}\,(P(x_0, x_2) + P(x_1, x_3) + P(x_4, x_6) + P(x_5, x_7))$$

$$z_1 = \text{sgn}\,(P(x_0, x_4) + P(x_1, x_5) + P(x_2, x_6) + P(x_3, x_7))$$

$$z_0 = \text{sgn}\,(x_0 + P(x_1, z_3) + P(x_2, z_2) + P(x_3, z_2, z_3) + P(x_4, z_1) + P(x_5, z_1, z_3) + P(x_6, z_1, z_2) + P(x_7, z_2, z_3))$$

If there are two errors in the received word, there will be no majority in z_1, z_2, or z_3, in which case a double-error pattern is detected.

3. Conclusions

We have illustrated a direct implementation of the Reed algorithm to decode rth-order Reed-Muller codes by $2(r+1)$-layer perceptrons. The used multi-layer perceptrons have all their weights at $+1$ or -1 and all thresholds integers, thus making them simple for hardware implementation.

The multi-layer perceptrons in this paper are constructed by concatenating two operations that are implementable by perceptrons. The idea of regarding some well-known networks as building blocks could be an alternative approach in constructing multi-layer perceptrons for solving problems.

References

BLAHUT, R. E., 1983, *Theory and Practice of Error Control Codes* (Singapore: Addison-Wesley).

RUMELHART, D. E., HINTON, G. E., and WILLIAMS, R. J., 1986, Learning internal representations by error propagation. *Parallel Distributed Processing*, edited by D. E. Rummelhart and the PDP Research Group (Cambridge, Mass.: MIT Press).

WESLEY PETERSON, W., and WELDOM, JR., E. J., 1972, *Error-correcting Codes*, second edition (Cambridge, Mass.: MIT Press).

WIDROW, B., and WINTER, R., 1988, Neural nets for adaptive filtering and adaptive pattern recognition. *Computer Magazine*, pp. 25-39, March.