"Calcul formel avancé et application". Brief lecture notes.

28.09.2023. Lecture 3.

3.1 Hamming distance

Definition 1. For any alphabet Σ, the *Hamming distance* between words $\bar{x} = x_1 \dots x_n, \bar{y} = y_1 \dots y_n \in \Sigma^n$ is defined as the number of indices i such that $x_i \neq y_i$ (the number of positions between 1 and n where the words $\bar{x}$ and $\bar{y}$ differ from each other).

Let us denote the Hamming distance between $\bar{x}$ and $\bar{y}$ as $\mathrm{dist_H}(\bar{x}, \bar{y})$. Observe that the Hamming distance satisfies the basic properties of distance familiar to us from Euclidean space:

- $\mathrm{dist_H}(x, x) = 0$,
- $\mathrm{dist_H}(x, y) = \mathrm{dist_H}(y, x)$,
- $\mathrm{dist_H}(x, y) \leq \mathrm{dist_H}(x, z) + \mathrm{dist_H}(y, z)$.

In the space Σ^n with the Hamming distance we can re-use the standard notion of a sphere and a ball. For example, a *ball* of radius r with the center at x is the set

$$B_r(x) = \{y \ : \ \mathrm{dist_H}(x, y) \leq r\}.$$

Let us mention that for the binary alphabet $\Sigma = \{0, 1\}^n$, the Hamming distance between $x_1 \dots x_n$ and the word that consists of n zeros $\underbrace{00\dots0}_{n}$ is equal to the number of *ones* in the words $x_1 \dots x_n$. This number is called the *Hamming weight* of $x_1 \dots x_n$.

3.2 Binary error correcting codes

We say that a function $C : \{0, 1\}^k \to \{0, 1\}^n$ is a binary code correcting e errors, of for all $x, y \in \{0, 1\}^k$, if $x \neq y$

$$\mathrm{dist_H}(C(x), C(y)) > 2e.$$

The images of C are called *codewords* of the code. The definition can be reformulated as follows: the mapping C is a code correcting e errors, if the balls of radius e centered at the codewords are pairwise disjoint.

This definition has a clear combinatorial meaning. If $\mathrm{dist_H}(C(x), C(y)) > 2e$, then there is no $z \in \{0, 1\}^n$ such that

$$\mathrm{dist_H}(C(x), z) \leq e \text{ and at the same time } \mathrm{dist_H}(C(y), z) \leq e.$$

In other words, if we take a string $x' = C(x)$ and flip in it at most e bits, the "corrupted" version of the codeword x' contains enough information to reconstruct uniquely x' and, respectively, the initial message x. Thus, if we send through a noisy channel the value $C(x)$, the initial k-bits message x can be reconstructed even of the noise corrupts e letters of the transmitted codeword.

Proposition 1. *If $C : \{0, 1\}^k \to \{0, 1\}^n$ is a code correcting e errors, then*

$$2^k \leq \frac{2^n}{1 + C_n^1 + C_n^2 + \ldots + C_n^e}.$$

Proof. In the space $\{0, 1\}^n$, every ball of radius e contains

$$1 + C_n^1 + C_n^2 + \ldots + C_n^e$$

points (the center of the ball, the $C_n^1 = n$ points at the distance exactly 1 from the center, the points at the distance 2 from the center, ..., the points at the distance exactly e from the center). The balls of radius e centered at the codewords must be disjoint, and they all belong to the set $\{0, 1\}^n$ of cardinality 2^n. $\square$

Corollary 1. *If we need to correct* one *error, then we cannot have more than* $2^n/(n+1)$ *codewords of length* n*. For example, we cannot have more than* $2^7/(7+1) = 16$ *binary codewords of length* 7 *in a code correcting one error.*

3.3 Linear codes

We can understand $\{0,1\}^n$ as a the linear space (= vector space) $(\mathbb{Z}/2\mathbb{Z})^n$, i.e., the n-dimensional linear space over the field of two elements. In this space, the operation of the *sum* of two vectors is the XOR of the vectors' coordinates. In the field $\mathbb{Z}/2\mathbb{Z}$ there are only two elements (zero and one), so the operations of multiplication of a vector by a constant are trivial (multiplication by 1 does not change the vector, multiplication by 0 gives the vector whose all coordinates are equal to zero).

A code $C : \{0,1\}^k \to \{0,1\}^n$ is called *linear*, if it is linear mapping from $(\mathbb{Z}/2\mathbb{Z})^k$ to $(\mathbb{Z}/2\mathbb{Z})^n$. The set of codewords of a linear code is a linear subspace in $(\mathbb{Z}/2\mathbb{Z})^n$. Observe that the zero vector (the vector whose all coordinates are equal to zero) is a always a codeword of a linear code.

A k-dimensional linear subspace in a vector space V can be defined in two different ways:

- by k vectors that provide a basis in this subspace,
- by $(n-k)$ linear equations such that this subspace is the set of solutions of this system of linear equations.

Equivalently, these two ways can be explained as follows. A linear code can be specified

- by a *generating matrix* G with k columns and n rows, such that the codewords are all vectors of the form
$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ \vdots \\ x_n \end{pmatrix} = G \cdot \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_k \end{pmatrix}$$
- by a *checksum matrix* H with n columns and k rows, such that the codewords are all vectors satisfying
$$H \cdot \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Proposition 2. *A binary linear binary code corrects* e *errors if and only if every non-zero codeword contains at least* $2e+1$ *ones.*

Proof. First of all, we observe that a code corrects e errors if and only if the distance between every two codewords is at least $2e+1$. Thus, we need to characterise the linear codes where the distance between every two codewords is greater or equal to $2e+1$.

If x and y are codewords in a linear code, the the bitwise XOR of x and y is also a codeword z. Therefore, the Hamming distance between x and y is equal to the Hamming weight of some codeword z. Thus, we can guarantee that the distance between every two codewords is greater or equal to $2e+1$ if the Hamming weight of every codeword z is greater or equal to $2e+1$ (besides the only one exception – the zero codewords).

The *only if* part follows from the observation that the Hamming weight of a codeword z is equal to the Hamming distance between z and the zero codeword $\underbrace{000\ldots0}_{n}$. ☐

Proposition 3. *A checksum matrix H with n columns and s rows defines a linear code that corrects* 1 *errors, if and only if (i) H contains no column of all zeros, and (ii) all columns in H are pairwise different.*

Proof. This follows from the fact that multiplication by H should not annulate any non-zero vectors with Hamming weight *strictly less* than $2e+1=3$. In other words, the matrix should not annulate vectors that contain *exactly one* or *exactly two* 1s. ☐

For a fixed s, a matrix H with s rows and n columns such that *there is no column of all zeros and all columns in H are pairwise different* can contain at most $n=2^s-1$ columns (all non-zero binary vectors of length s, without repetition). Here is an example of such a matrix for $s=4$ and $n=2^s-1=15$:

$$H=\begin{pmatrix} 0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\\ 0&0&0&1&1&1&1&0&0&0&0&1&1&1&1\\ 0&1&1&0&0&1&1&0&0&1&1&0&0&1&1\\ 1&0&1&0&1&0&1&0&1&0&1&0&1&0&1 \end{pmatrix}$$

Such a checksum matrix defines a vector space of dimension

$$n-s=2^s-1-s$$

(the space of solution of a system of s linear equations with $n=2^s-1$ variables). Therefore, we obtain a linear code with

$$2^{n-s}=2^{2^s-s-1}$$

codewords. This code is optimal among all codes correcting one error: we cannot obtain any code with the same length of codewords n and greater number of codewords. Indeed, the Hamming balls of radius 1 contain $n+1$ points; since the balls of radius 1 constructed around all codewords must be disjoint, we have

$$(n+1)\cdot[\text{number of codewords}]\le 2^n.$$

Thus, the number of codewords is bounded by

$$\frac{2^n}{n+1}=\frac{2^n}{2^s}=2^{n-s}=2^{2^s-s-1},$$

which is exactly the number of codewords in the code constructed above. Such a code is called *Hamming code.*

3.4 Correcting errors in the Hamming code

Let $\bar{x}=(x_1\ldots x_n)$ be a codeword of the Hamming code, i.e., a string of bits such that

$$H\cdot\begin{pmatrix} x_1\\ x_2\\ \vdots\\ \vdots\\ \vdots\\ x_n \end{pmatrix}=\begin{pmatrix} 0\\ 0\\ \vdots\\ 0 \end{pmatrix}$$

(where H is the checksum matrix of the hamming code, with s rows and $n=2^s-1$ columns). Let us flip one bits in the string and denote the result $\bar{x}'$. The Hamming code corrects one error, so we can reconstruct

$\bar{x}$ uniquely given the "corrupt" vector $\bar{x}'$. Moreover, we can do it quite efficiently. If we multiply $\bar{x}$ by H, we will get a string of of s bits (called *syndrome* of $\bar{x}'$) which shows which bit of $\bar{x}'$ is corrupt. In fact, it is not hard to see that the syndrome will give us the *binary expansion* of the index of the corrupt bit. For example, if $s = 3$ and

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix},$$

the string $\bar{x} = (1010101)$ is a codeword. Let us flip the 4-th bit, $\bar{x}' = (1011101)$. When we compute the syndrome of $\bar{x}'$ (the product of the checksum matrix H and the column-vector $\bar{x}'$), we get

$$\begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0\cdot 1 + 0\cdot 0 + 0\cdot 1 + 1\cdot 1 + 1\cdot 1 + 1\cdot 0 + 1\cdot 1 \\ 0\cdot 1 + 1\cdot 0 + 1\cdot 1 + 0\cdot 1 + 0\cdot 1 + 1\cdot 0 + 1\cdot 1 \\ 1\cdot 1 + 0\cdot 0 + 1\cdot 1 + 0\cdot 1 + 1\cdot 1 + 0\cdot 0 + 1\cdot 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

(the calculations done modulo 2). The resulted syndrome (100) is the binary expansion of the number $4 = 1 \cdot 2^2 + 0 \cdot 2^1 + 0$, which is equal to the position of the corrupt bit.

3.5 Encoding procedure for the Hamming code

A linear code $C : \{0,1\}^k \to \{0,1\}^n$ assigns to every binary vector of length k the corresopinding binary vector of length n. Even if we fix the set of codewords $\mathcal{C}$ (in our case, it is a linear subspace of dimension k in $(\mathbb{Z}/2\mathbb{Z})^n$), we can establish a bijection between $\{0,1\}^k$ and $\mathcal{C}$ in many different ways. Moreover, even a *linear* mapping between $(\mathbb{Z}/2\mathbb{Z})^k$ and $\mathcal{C}$ can be defined in many different ways. However, not all encoding procedures are equally useful in practice. In what follows we discuss a simple and natural encoding for the Hamming codes.

We consider a Hamming code with the length of the codewords $n = 2^s - 1$. Let us recall that the checksum matrix consists of all non-zero binary columns of size s, e.g., for $s = 3$ we have

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ * & * & & * & & & \end{pmatrix}$$

Observe that k of these columns contain exactly one bit 1 (and $k-1$ bits 0). Let us call these columns *special* (in our example the special are the columns *one*, *two*, and *four*). If we are looking for a solution a of linear system defined by the checksum matrix,

$$\left\{ \begin{array}{cccccccccc} & & & x_4 & +x_5 & +x_6 & +x_7 & = & 0 \\ & x_2 & +x_3 & & & +x_6 & +x_7 & = & 0 \\ x_1 & & +x_3 & & +x_5 & & +x_7 & = & 0 \\ * & * & \uparrow & * & \uparrow & \uparrow & \uparrow & & \\ & & w_1 & & w_2 & w_3 & w_4 & & \end{array} \right.$$

and we fix all the values x_i for *non-special* columns, we can then easily compute the corresponding values of the *special* x_i.

Thus, in our example we can encode a string of four bits (w_1, w_2, w_3, w_4) into a codeword $(x_1, x_2, x_3, x_4, x_5, x_6, x_7)$

as follows:

$$\begin{array}{rcll} x_3 & := & w_1 & \\ x_5 & := & w_2 & \\ x_6 & := & w_3 & \\ x_7 & := & w_4 & \\ x_1 & := & w_1+w_2+w_4 & \bmod 2 \\ x_2 & := & w_1+w_3+w_4 & \bmod 2 \\ x_4 & := & w_2+w_3+w_4 & \bmod 2 \end{array}$$

Such a string of 7 bits will satisfy system of linear equations determined by the matrix H and, therefore, this string will be a codeword of the Hamming code.

Thus, we have a simple algorithm of encoding, which computes for a a bit string $(w_1 \dots w_k)$ the corresponding codeword $(x_1 \dots x_n)$. Observe that in our encoding procedure every bits w_i of the initial message is embedded directly in the codeword: in the example above, the bits $w_1, \dots, w_4$ appear in the corresponding codeword $(x_1 \dots x_7)$ at the positions $3, 5, 6, 7$ respectively. Such a code is called *systematic*. For a systematic code, if there is no error in the codeword, the procedure of decoding is trivial (in the example discussed above, it is enough to "erase" in the codeword the bits x_1, x_2, and x_4, and the remaining four bits will give the original message $(w_1w_2w_3w_4)$).

3.6 The game *guess a number* with one false response

In the class we discussed how to use the Hamming codes to play the game *guess a number*, where the first players chooses a natural number x between 1 and N, and the second players asks *yes/no* questions. The first player is allowed to give at most one false answer. The challenge is, as usually, to suggest a strategy with the minimum number of questions so that the second player can reveal the number x, despite one possible false answer of the first player.

Without loss of generality we may assume that the second player always asks the same number of questions, whatever x is given and whatever answers the first player gives, i.e., the *average* number of questions is the same as the *worst case* number of questions. Indeed, if some "branch" of the protocol is shorter than the others, we can add there dummy questions; this would not worsen the strategy performance since we only care of the number of questions in the worst case situation.

In what follows we discuss lower and upper bounds on the size of the optimal strategy for a specific example of N. Let us take, for instance, $N = 2048$. Assume that there is a strategy where the second player reveals a number $x \in \{1, \dots, N\}$ after asking n questions. We fix this strategy of the second player. We know that for every x, the first player may choose a question between 1 and n to give a false answer, or decide to give all true answers. Thus, for every possible x there are $n + 1$ possible sequences of answers. Since each sequence of answers must correspond to only one number x, we have

$$N \le 2^n/(n+1).$$

As $N = 2048$, we see that n must be *at least* 15 (for $n = 1, \dots, 14$ we would have $2^n/(n+1) < 2048$, which gives a contradiction with the inequality above).

Let us show now that 15 questions is enough to guess an integer number x between 1 and $n = 2048$ with one false answer. We will construct such a strategy using the Hamming code for $s = 4$ and $n = 2^4 - 1 = 15$. Indeed, in this code we have $2^{n-s} = 2048$ codewords of length 15, and every two codewords doffer in at least 3 positions. We can associate these codewords with the numbers $\{1, \dots, N\}$. The second player should ask questions

- Does the 1-st bits of the x-th codeword equal to 0 ?
- Does the 2-nd bits of the x-th codeword equal to 0 ?
- Does the 3-rd bits of the x-th codeword equal to 0 ?

$\vdots$

- Does the 15-th bits of your codeword equal to 0 ?

The answers to this question give us the bits of the x-th codeword, possibly with one error (if one answers was false). But even of one of the answers is false, the Hamming code allows to reconstruct the entire codeword. This is enough to reveal the number x chosen by the first player.

In the class we also discussed an optimal strategy in this game for $N = 100$.