

Lecture 14: Error Correcting Codes

ENGR 76 lecture notes — May 14, 2026

David Tse¹, Stanford University

1 Introduction

In the previous lecture we studied how digital information is transmitted over communication systems, focusing on how binary data is converted into physical signals or waveforms that propagate through a communication channel. Ideally, the receiver would observe exactly the same waveform that was transmitted and recover the original sequence of bits without error. In practice, however, communication channels are imperfect. As signals travel through the channel they may be affected by noise, interference, attenuation, or distortion, which can cause the receiver to interpret some bits incorrectly. Such discrepancies are known as **transmission errors**. Since reliable communication is essential in modern systems, mechanisms are needed to detect when these errors occur and, where possible, to recover the correct data.

This motivates introducing **redundancy** into transmitted data to improve reliability. Instead of sending only the original information bits, the transmitter adds additional bits derived from the data. These extra bits allow the receiver to check whether errors occurred during transmission and, in some cases, to determine the correct data. We next describe these two problems in more detail.

1.1 Error Detection

The first problem is **error detection**, which involves determining whether one or more errors have occurred during transmission. In error detection schemes, additional check information is transmitted together with the data so that the receiver can verify the integrity of the received message. If the received sequence does not satisfy the required consistency checks, the receiver can conclude that an error has occurred.

Error detection is widely used in practice. For example, internet packets often include a checksum or cyclic redundancy check (CRC) that allows the receiver to verify whether the packet was corrupted during transmission. Similarly, many barcode systems include a check digit computed from the other digits; when the barcode is scanned, the system recomputes this value to detect scanning errors.

1.2 Error Correction

The second problem is **error correction**, which involves determining the original transmitted data despite the presence of errors. Error correction schemes introduce structured redundancy into the transmitted data so that the receiver can not only detect that an error has occurred but also infer the most likely original message and correct the corrupted bits.

Error correction is particularly important when retransmission is difficult or impossible. For example, in satellite or deep-space communication, the long delay between transmitter and receiver makes retransmission impractical. Error correction is also used in compact discs (CDs), where error-correcting codes allow the player to recover the original data even if scratches or imperfections corrupt some bits during reading.

2 Simplest Scheme: Repetition Codes

We begin with one of the simplest ways to introduce redundancy: **repetition**. Suppose we wish to transmit a message that consists of only two possible symbols, represented by bits 0 and 1. Instead of sending the bit

¹These notes are adapted from the 2026 Winter offering by Siddarth Chandak

once, we transmit it multiple times. More precisely, if the message bit is $b \in \{0, 1\}$, the transmitter sends a sequence consisting of n copies of that bit:

$$0 \mapsto 00 \dots 0 \quad (n \text{ times}), \quad 1 \mapsto 11 \dots 1 \quad (n \text{ times}).$$

At the receiver, the repeated bits are examined to infer the original transmitted value. Since the transmitter sends either a sequence of all 0s or all 1s, the receiver looks at the received bits and decides which value appears most often. This is known as a *majority rule*. For example, if three bits are sent and the receiver observes 011, then two of the bits are 1, so the receiver concludes that the original bit was 1. The redundancy introduced by repeating the bit therefore allows the receiver to recover the correct message even if a small number of bits are flipped during transmission.

Example 1. Consider the case $n = 3$. The encoding rule is

$$0 \mapsto 000, \quad 1 \mapsto 111.$$

Suppose the transmitter sends the bit 1, so the transmitted sequence is 111. If a single bit is flipped during transmission, the receiver might observe one of the following: 011, 101, or 110. In each case, two of the three bits are still equal to 1. Using a *majority rule*, the receiver concludes that the original bit was 1. Similarly, if 000 is transmitted and a single bit is flipped, the receiver might observe 100, 010, or 001, and the majority rule correctly recovers 0. Thus, when $n = 3$, the repetition scheme can **correct one bit error**.

Example 2. Now consider the case $n = 2$. The encoding rule is

$$0 \mapsto 00, \quad 1 \mapsto 11.$$

Suppose the transmitter sends 1, so the transmitted sequence is 11. If one bit is flipped during transmission, the receiver might observe 10 or 01. In this situation, the receiver knows that an error must have occurred, since neither 10 nor 01 matches a valid encoded message. However, the receiver cannot determine whether the original message was 00 or 11. Therefore, the scheme can **detect** an error but cannot **correct** it.

These examples illustrate that the ability to detect or correct errors depends on how different the valid transmitted sequences are from each other. We formalize this in Section 4. We next study general coding schemes, and define required notation to describe a code.

3 General Schemes and Figures of Merit

To study error detection and correction more systematically, we introduce a general framework for coding schemes. A coding scheme specifies a collection of valid sequences that may be sent through the channel. Any sequence that is not in the code is considered invalid.

Definition 1. A **code** is a set of binary sequences of fixed length that may be transmitted over a communication channel. The elements of this set are called **codewords**.

3.1 Encoding and Decoding Process

The transmitter begins with an **information sequence**, which is the sequence of bits that we wish to communicate. An encoder maps this sequence to a longer binary sequence called the **encoded sequence**. This encoded sequence is a codeword in the code. The encoded sequence is transmitted over the communication channel. Because of noise or other impairments, the receiver may observe a different sequence of bits, called the **received sequence**. The receiver applies a decoder to the received sequence. The decoder outputs a **decoded sequence**, which is its estimate of the transmitted codeword. From this, the receiver obtains a **decoded information sequence**, which is the estimate of the original information.

When decoding, the receiver compares the received sequence with the valid codewords and chooses the one that appears most likely to have been transmitted. A natural rule is to choose the codeword that differs from the received sequence in the fewest positions. This is called **minimum distance decoding**.

Example 3 (Full Communication Process (Repetition Code)). Consider the repetition code

$$0 \mapsto 000, \quad 1 \mapsto 111.$$

- Information sequence: 1011
- Encoded sequence: 111000111111

Suppose the received sequence is 101100111100. Then, to decode, divide the received sequence into blocks of 3 bits and map each block to the closest codeword.

- Received sequence: 101100111100
- Decoded sequence: 111000111000
- Decoded information sequence: 1010

3.2 Hamming Distance and Minimum Distance

To analyze decoding more formally, we introduce a notion of distance between binary strings.

Definition 2. The *Hamming distance* between two binary strings of the same length is the number of positions in which the strings differ.

Definition 3. The *minimum distance* of a code is the smallest Hamming distance between any two distinct codewords in the code.

Example 4. Consider the repetition code with $n = 3$:

$$C = \{000, 111\}.$$

The Hamming distance between the two codewords is 3. Thus the minimum distance of this code is $d_{\min} = 3$.

Example 5. Consider the code

$$C = \{00000, 00111, 11100, 11011\}.$$

Computing pairwise Hamming distances:

- $d_H(00000, 00111) = 3$
- $d_H(00000, 11100) = 3$
- $d_H(00000, 11011) = 4$
- $d_H(00111, 11100) = 4$
- $d_H(00111, 11011) = 3$
- $d_H(11100, 11011) = 3$

The smallest of these distances is 3, so the minimum distance of the code is $d_{\min} = 3$.

Alternate Perspective

So far, we have viewed coding as the process of adding redundancy to messages before transmission. An equivalent viewpoint is that a code is simply a subset of all binary strings of a fixed length, where the codewords are chosen to be far apart from each other in Hamming distance. From this perspective, designing a good code amounts to selecting binary strings that are well separated; a larger minimum distance between codewords directly leads to better error-detection and error-correction capabilities.

3.3 Figures of Merit

Several quantities are used to evaluate the performance of a coding scheme.

- n : The number of bits in each codeword (block length).
- k : The number of information bits in each message.
- $M = 2^k$: The number of possible messages (equivalently, the number of codewords in the code).
- $d_{\min}$: The minimum Hamming distance between any two distinct codewords.
- Rate:

$$R = \frac{k}{n}.$$

The rate measures the fraction of transmitted bits that carry information.

A good code typically aims to have a large k (more information), small n (shorter transmissions), large rate R , and large $d_{\min}$ (better error-correction capability). These goals, however, are often in tension with each other.

Example 6 (Repetition Code). Consider the repetition code $C = \{000, 111\}$. Its parameters are $n = 3$, $k = 1$, $M = 2$, rate $= \frac{1}{3}$, and $d_{\min} = 3$.

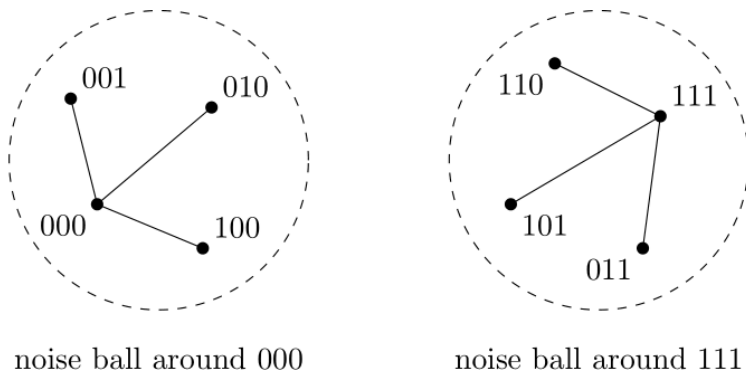
Example 7. Consider the code $C = \{00000, 00111, 11100, 11011\}$. Its parameters are $n = 5$, $k = 2$, $M = 4$, rate $= \frac{2}{5}$, and $d_{\min} = 3$.

4 Error Detection and Error Correction Capabilities

We first define Hamming balls, which will help us both build intuition and formally prove results. For a codeword c and integer $t \geq 0$, define the Hamming ball

$$B(c, t) = \{x : d_H(x, c) \leq t\}.$$

This consists of all binary strings that differ from c in at most t positions. The below figure the Hamming balls of radius 1 around codewords 000 and 111.



4.1 Error Detection

Let $\mathcal{C}$ be a code with minimum distance $d_{\min}$.

Theorem 1 (Error Detection). *A code can detect up to t bit flips if and only if*

$$d_{\min} \geq t + 1.$$

Equivalently, a code with minimum distance $d_{\min}$ can detect up to $d_{\min} - 1$ bit flips.

Proof. Suppose $d_{\min} \geq t + 1$. Let $c_1 \in \mathcal{C}$ be transmitted and suppose at most t bit flips occur. Then the received word r differs from c_1 in at most t positions. Since every other codeword is at distance at least $d_{\min} \geq t + 1$ from c_1 , it is impossible for r to equal another codeword. In other words, r cannot coincide with any valid codeword in $\mathcal{C}$. Therefore $r \notin \mathcal{C}$, and the receiver detects that an error has occurred.

Conversely, suppose $d_{\min} \leq t$. Then there exist two distinct codewords c_1 and c_2 whose distance is at most t . If c_1 is transmitted and the channel flips exactly those bits in which c_1 and c_2 differ, the received word becomes c_2 . Since c_2 is also a valid codeword, the receiver has no way to detect that an error occurred. Therefore, the code cannot guarantee detection of t bit flips.

Therefore, the code can detect up to t bit flips if and only if $d_{\min} \geq t + 1$. $\square$

Geometric Interpretation. The diagrams provide a useful geometric intuition. In the left figure, we draw the Hamming ball $B(c_1, t)$ around c_1 . As long as $t < d_{\min}$, this ball does not contain any other codeword such as c_2 . Hence any received word inside the ball cannot be another valid codeword, and an error is detected. In the right figure, if we allow $t = d_{\min}$, the ball around c_1 can reach another codeword c_2 . In this case, it is possible for c_1 to be transformed into c_2 by $d_{\min}$ bit flips. Since c_2 is itself a valid codeword, the receiver cannot detect that an error occurred. This explains why a code can detect at most $d_{\min} - 1$ bit flips.

4.2 Error Correction

We now consider error correction under minimum-distance decoding.

Theorem 2 (Error Correction). *A code can correct up to t bit flips if and only if*

$$d_{\min} \geq 2t + 1.$$

Equivalently, a code with minimum distance $d_{\min}$ can correct up to

$$\left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor$$

bit flips.

Proof. Suppose first that $d_{\min} \geq 2t + 1$. Let c_1 be transmitted and suppose at most t bit flips occur, so that the received word r satisfies $d_H(r, c_1) \leq t$. To decode correctly, we must ensure that r is closer to c_1 than to any other codeword c_2 . Since every other codeword satisfies $d_H(c_1, c_2) \geq d_{\min}$, and since r differs from c_1 in at most t positions, the distance from r to c_2 must be at least

$$d_H(c_1, c_2) - d_H(r, c_1) \geq (2t + 1) - t = t + 1.$$

Thus r is strictly closer to c_1 than to any other codeword, and minimum-distance decoding correctly recovers c_1 .

Conversely, suppose $d_{\min} \leq 2t$. Then there exist two distinct codewords c_1 and c_2 whose distance is at most $2t$. In this case, the Hamming balls $B(c_1, t)$ and $B(c_2, t)$ overlap. Hence there exists a word r that lies within distance t of both c_1 and c_2 . If c_1 is transmitted and r is received, the decoder cannot uniquely determine which codeword was sent. Therefore the code cannot guarantee correction of t bit flips. $\square$

Geometric Interpretation. The figures illustrate the key idea. In the left diagram, the Hamming balls of radius t around distinct codewords do not intersect. Any received word within distance t of a transmitted codeword lies strictly inside its ball and cannot belong to another ball. Thus decoding is unambiguous. In the right diagram, the balls intersect. There exist received words that are within distance t of two different codewords. When this happens, minimum-distance decoding cannot uniquely recover the transmitted codeword. This overlap explains why error correction requires $d_{\min} \geq 2t + 1$.

5 The (7, 4) Hamming Code

Hamming codes form a family of binary linear codes with minimum distance $d_{\min} = 3$. Since a code with minimum distance 3 can correct one bit flip, Hamming codes provide single-error correction with very low decoding complexity. They were the first practical code and were developed in the 1940s by Richard Hamming and are still used in applications such as memory hardware, where fast and simple error correction is required.

In this section, we study the (7, 4) Hamming code, the smallest nontrivial member of this family. It is called a (7, 4) code because each codeword has length $n = 7$ bits, while each message consists of $k = 4$ information bits. The encoder therefore adds three redundancy bits to the four information bits.

5.1 Encoding

Let the information bits be b_1, b_2, b_3, b_4 . The encoder produces a 7-bit codeword of the form $b_1b_2b_3b_4p_1p_2p_3$, where the bits p_1, p_2, p_3 are chosen to enforce parity constraints. These parity bits are defined using the XOR operation, denoted by $\oplus$. For two bits, $a \oplus b$ equals 1 if the bits differ and 0 if they are the same. Equivalently, XOR outputs 1 if an odd number of input bits are 1, and 0 if the number of 1's is even. The parity bits are defined by

$$p_1 = b_1 \oplus b_2 \oplus b_3, \quad p_2 = b_1 \oplus b_2 \oplus b_4, \quad p_3 = b_2 \oplus b_3 \oplus b_4.$$

Each parity bit is chosen so that a specific group of bits contains an even number of 1's. For example, the definition of p_1 ensures that the bits b_1, b_2, b_3 , and p_1 together contain an even number of 1's. The other parity bits are defined similarly. In this way, every valid codeword satisfies three parity conditions, and each information bit participates in more than one such group. As a result, if a single bit is flipped during transmission, at least one of these conditions is disturbed, allowing the receiver to detect and locate the error.

