# Linear Complementary Dual Codes Constructed from Reinforcement Learning\*

WU Yansheng · MA Jin · YANG Shangdong

DOI:

Received: 04 Jul 2024 / Revised: 03 Sep 2024

© The Editorial Office of JSSC & Springer-Verlag GmbH Germany 2024

Abstract Recently, Linear Complementary Dual (LCD) codes have garnered substantial interest within coding theory research due to their diverse applications and favorable attributes. This paper directs its attention to the construction of binary and ternary LCD codes leveraging curiosity-driven reinforcement learning (RL). By establishing reward and devising well-reasoned mappings from actions to states, it aims to facilitate the successful synthesis of binary or ternary LCD codes. Experimental results indicate that LCD codes constructed using RL exhibit slightly superior error-correction performance compared to those conventionally constructed LCD codes and those developed via standard RL methodologies. The paper introduces novel binary and ternary LCD codes with enhanced minimum distance bounds. Finally, it showcases how Random Network Distillation aids agents in exploring beyond local optima, enhancing the overall performance of the models without compromising convergence.

Keywords artificial intelligence, LCD code, reinforcement learning.

## 1 Introduction

The rapid development of data communications has increasingly underscored the necessity for reliable data transmission. In this context, Claude Shannon published his groundbreaking work "A Mathematical Theory of Communication" [1], which laid the foundational groundwork for modern information theory and coding theory. Shannon posited that every communication channel possesses a parameter known as channel capacity; if the transmission rate demanded by a communication system is less than this capacity, there exists a coding scheme such that when applied with an appropriate code length and maximum likelihood decoding, the probability of error in the system can approach zero arbitrarily.

WU Yansheng

School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing, 210023, China; Email:yanshengwu@njupt.edu.cn

MA Jin · YANG Shangdong

School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing, 210023, China. Emai: 1222046136@njupt.edu.cn; sdyang@njupt.edu.cn.

<sup>\*</sup>This work was supported by the National Natural Science Foundation of China (Nos. 62372247, 12101326, 62206133) and the Open Project of Guangxi Provincial Key Laboratory (No. MIMS22-01).

<sup>♦</sup> This paper was recommended for publication by Editor .

However, while Shannon established the existence theorem for such codes, he did not provide explicit instructions on how to construct them. This challenge spurred further research in the field of information theory, particularly in the design of error-correcting codes. Researchers, inspired by Shannon's theorem, diligently sought innovative coding strategies to enhance the reliability of data transmission.

Over the subsequent decades, scholars have devised numerous classes of error-correcting codes with strong performance metrics. Yet, these high-performance codes typically embody specific structures or adhere to certain properties. Without such structures, the utility or error-correcting capability of a code may be significantly limited.

Linear Complementary Dual (LCD) codes, as a special class of linear codes, have garnered substantial interest due to their unique property that their intersection with their dual codes contains only the zero element. This inherent characteristic makes LCD codes a focal point of ongoing research, as they offer potential advantages in terms of resistance to errors and security enhancements in communication systems.

Currently, the construction of LCD codes predominantly relies on principles within coding theory, where the design challenge is reframed as an optimization problem of code parameters to control the code's performance. For LCD codes, three essential parameters are typically considered: the code length n, dimension k, and minimum distance d. If  $\mathcal{C}$  is a nonempty subset of an n-dimensional linear space over  $\mathbb{F}_q$ ,  $\mathcal{C}$  is said to be a q-ary error-correcting code. Each element in the code  $\mathcal{C}$  is called a codeword of  $\mathcal{C}$ . Let k represent the information bits of the code, and k/n represent the code rate, which is the basic parameter to measure the effectiveness of the code. The minimum distance d refers to the smallest Hamming distance between any two distinct codewords, indicating the code's error-correcting prowess. These three parameters interact. For example, under a fixed code length n, increasing the dimension k might result in a smaller minimum distance. Their mutual constraints can be mathematically represented through inequalities, thereby establishing bounds for the parameters. Codes that closely approach or reach these bounds are generally considered to have superior performance.

Based on coding theory, different codes and construction methods require specific expertise, and without considering decoders or channel conditions during construction, accurate assessment of code performance is not possible. Moreover, certain coding theory approaches involve constraint conditions; for instance, constructing truncated LCD codes may necessitate the use of distinct theorems depending on whether the codeword weights are odd or even, or even the parity of the dimension, which significantly impacts the flexibility in constructing LCD codes [2]. Finally, construction methods rooted in coding theory often concentrate exclusively on a singular performance metric, thereby encountering challenges in simultaneously optimizing multiple criteria, rendering comprehensive consideration intricate. And many methods for computer implementation are more difficult, can not use computer resources.

With the development of artificial intelligence technology, the combination of error-correcting code and artificial intelligence has become an effective method to build error-correcting code. This kind of method can not only automate coding, access channel conditions and decoder modules, accurately evaluate the performance of the code, but also access various modules, such

as curiosity module, with high scalability. Kim et al. proposed "deepcode" based on AWGN feedback channel and conducted unified training for encoders and decoders based on recurrent neural networks [3]. They demonstrate that the conjunction of deep learning methodologies with channel-specific insights has the potential to excel over coding systems meticulously refined through decades of rigorous mathematical inquiry. Gruber [4] also found that structured codes are easier to learn, and neural networks can generalize structured codes, which means that structured codes have more advantages in learning. Recently, some scholars have also applied artificial intelligence techniques to coding theory, which is called AI coding [5]. However, the code constructed in [5] is limited to ordinary linear code, does not contain some structure, and only focuses on the case of binary field, which makes its usability limited.

Based on the properties of LCD codes, this paper presents a method of constructing binary and ternary LCD codes based on reinforcement learning (RL) from the perspective of reward function and action mapping. This makes it possible for RL to construct any kind of code, and is no longer limited to the binary field, further improves the usability of RL in error-correcting coding, and accelerates the cross-application of RL and error-correcting coding. RL shares remarkable similarities with the challenges encountered in constructing error-correcting codes, particularly in confronting the necessity to navigate through immense state spaces. Employing the principles of RL, in order to deal with the challenge of excessive code space, the random network distillation (RND) algorithm [6] is added to the evaluation module to generate internal incentives, and curiosity is added to the agent to assist exploration. By using the method proposed in this paper, the existence of some almost optimal LCD codes is found, and the range of the minimum distance bound of some LCD codes is reduced.

#### 1.1 Related work: LCD codes

LCD codes, first proposed by Massey [7] in 1992, have been the focus of extensive research because of their unique theoretical properties and practical significance. From a theoretical point of view, Massey proved that LCD codes can provide an optimal encoding of a two-user binary addition channel [7]. Sendrier [8] proved that LCD codes satisfy the asymptotic Gilbert-Varshamov bound by using Hull dimension spectrum of linear codes and their dual codes. Carlet [9] et al. characterized LCD codes and studied the structure of LCD codes and their subsets, and the results showed that almost all binary LCD codes belong to odd-like codes with odd-duals. These studies have extended the discussion on the overall structure of LCD codes. In addition, Carlet et al. [10] also revealed that for q > 3, any q-ary linear code can be represented equivalently as an LCD code on  $\mathbb{F}_q$ . In other words, in the study of LCD codes, it is enough to study only binary and ternary cases, simplifying the scope of the study. LCD cyclic codes are also known as reversible codes. Li [11] further analyzes the code by studying its reversibility and can improve its minimum distance lower bound.

In practical applications, LCD codes are not only used for communication error correction, but also widely used in the field of cryptography. Based on LCD codes, a new paradigm for side channel defense and resistance to fault injection attacks ([12, 13]), orthogonal direct and mask algorithm, is proposed. The algorithm uses the generator matrix of LCD code to encode

the original ciphertext, and uses its check matrix to encode a random vector to generate mask. In other words, the algorithm takes advantage of the properties of LCD codes to divide the linear space into two complementary subspaces: one for function operations and the other for generating random masks. Because of the characteristic of the error correcting code, the algorithm can also detect whether there is an attack and produce an error during calculation. According to the provable security, if the LCD code parameter is [n, k, d], the algorithm can resist the unary attack of order less than d. The orthogonal straight sum mask algorithm based on LCD code also has the properties of Boolean addition mask scheme, lookup table mask scheme and low entropy mask scheme, so it has high availability.

In the stage of key distribution and key recovery, Ghosh et al.[17] proposed an excellent secret sharing scheme based on LCD code by using the characteristics of LCD code, which could outperform other secret sharing schemes based on other linear codes while the information rate was close to 1. In addition, the orthogonality of LCD codes makes them of great value in constructing quantum error-correcting codes ([14, 15]) that can protect quantum information from noise, thus greatly advancing the development of quantum computers. Rajput et al.[16] were also able to provide a solution for single node errors in distributed data storage by establishing a link between local recoverable codes and LCD codes.

## 1.2 Related work: reinforcement learning

RL is a learning mechanism whose goal is to learn how to map states to actions in order to get the maximum reward. However, RL in the initial stage faces the challenge of large state space in solving complex problems. With the rise of neural networks, the combination of RL and deep learning can better solve complex problems, thus making RL widely used [18]. Vlad Mnih et al.[19] first proposed the Deep Q Network (DQN) algorithm, which has pioneering significance in the field of deep reinforcement learning.

In addition to value-based methods, policy-based methods are also an important direction of reinforcement learning. The policy-based approach directly uses approximators to approximate and optimize the strategy, and finally obtains the optimal policy. This method is more suitable for continuous high dimensional action space and random policy. In 1992, Williams et al.[20] proposed the REINFORCE algorithm, unifying the form of the policy gradient (PG). Subsequently, AC algorithm [21] combined the value-based method and the policy-based method, and used Q value to reduce the variance of the policy, which helped to learn the policy function better. Huang et al.[5] proposed a general constructor-evaluator framework based on previous studies and constructed binary linear block codes using REINFORCE algorithm and polarization codes using A2C algorithm. The data show that the performance of these learned codes is comparable to that of existing classical codes.

#### 1.3 Related work: Curiosity module

Exploration and utilization are the core problems of RL. In the face of huge state space, an efficient exploration method is essential. In order to better explore, a common approach is to encourage additional exploration by means of an additional reward signal, namely an update

strategy with a reward signal  $r = r_e + \beta * r_i$  consisting of two parts, where  $r_e$  is the external reward for environmental feedback,  $r_i$  is the exploration reward, and  $\beta$  is the parameter used to balance the exploration. The intrinsic reward model can be divided into two main approaches: count-based and curiosity-based. Count-based exploration measures state novelty by counting or estimating the number of states that have already been visited, encouraging the agent to visit the new state. However, for extremely large state spaces such as code space, count-based exploration is more laborious than curiosity-based exploration. The curiose-based intrinsic reward is designed to simulate environmental dynamics, which can be expressed as an error in predicting the consequences of the agent's actions, and gives a higher intrinsic reward to areas that are poorly simulated.

In part, this intrinsic reward is inspired by psychology's intrinsic motivation, just as curiosity-driven exploration may be an important way for children to grow and learn. Curiosity is essentially an internal reward, the error of the agent in predicting the consequences of its own actions in the current state. The RND (Random Network Distillation) algorithm stands out among curiosity-driven exploration methods, particularly well-suited for scenarios with vast state spaces. It leverages the discrepancy between the outputs of two neural networks to gauge the novelty of a state. Specifically, two neural networks are randomly generated, one is the target network, the parameters do not participate in the update, and the other is the prediction network, which will be constantly updated in the iteration, that is, the prediction network will fit the target network in the continuous training, and the resulting loss or the gap in the output vector can be used as an assessment of novelty. The greater the loss, the less the number of states encountered. The RND algorithm is shown in Figure 1.

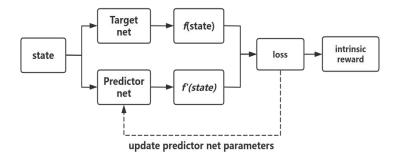


Figure 1 RND algorithm

# 2 LCD codes construct based on PG algorithm

#### 2.1 Constructor-Evaluator

In this paper, the constructor-evaluator framework is used to construct LCD codes. In this framework, the construction module uses RL algorithm to construct the generator matrix of LCD code. The evaluation module uses the decoder (or other performance estimation) to evaluate the performance of the LCD code, and combines with the intrinsic incentive for reprocessing to form a reward. The evaluation module feeds the reward back to the construction module, which updates itself based on the reward and continues to construct the LCD code. Repeat this process until the performance indicator converges. The specific frame is shown in Figure 2.

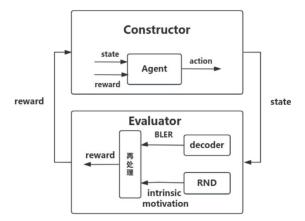


Figure 2 Constructor-Evaluator Frame

#### 2.2 Constructor

The construction process of LCD code is regarded as a decision process. This process is similar to the Markov decision process (MDP) and is defined as a quadruple (S, A, P, r). Where, the state s represents the standard form generator matrix of the code; Action a indicates the modification of the code; p stands for state transition function; r represents the reward returned by the evaluation module, and the reward is positively correlated with performance. The essence of RL is to determine an optimal policy to maximize the expected return. As the construction module continuously optimizes its decision, the performance of the constructed code is improved in continuous iteration. Finally, the desired code structure can be obtained from the final state of the MDP.

The policy-based approach first needs to parameterize the policy. Therefore, the target policy  $\pi_{\theta}$  is assumed to be random and differentiable everywhere, where  $\theta$  represents the relevant parameters. Then the objective function of policy learning is defined as:

$$J(\theta) = E_{s_0}(V^{\pi_{\theta}}(s_0)) \tag{1}$$

Where  $s_0$  represents the initial state and  $V^{\pi_{\theta}}$  represents the state value function. Then take the derivative of the objective function to  $\theta$ , get the derivative, use the gradient ascending method to maximize the objective function, so as to get the optimal policy. After collecting

certain samples, update policy is adopted by (2):

$$\nabla_{\theta} J(\theta) = E_{\pi_{\theta}} \left[ \sum_{t=0}^{T} \left( \sum_{t'=t}^{T} \gamma^{(t'-t)} r_{t'} \right) \nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \right]$$
 (2)

Where T is the maximum number of steps interacting with the environment, and  $\gamma$  is the discount factor.

To solve MDP, the REINFORCE algorithm was used, which was a classic algorithm of the policy gradient method in RL. The REINFORCE algorithm theoretically ensured local optimization and used the Monte Carlo method to sample trajectories to estimate the action value. The advantage of this method was that an unbiased gradient could be obtained. In addition, using the powerful approximation ability of neural network, the policy function is represented by neural network. Then the neural network is updated by gradient descent to promote the optimization of the policy function to improve the performance.

# 2.3 Evaluator

The role of the evaluator includes evaluating the performance of the constructed code, generating intrinsic incentives and determining whether it is an LCD code, and converting the performance indicators into appropriate rewards. The reward from the evaluator is the key communication link between human intent and algorithm execution, communicating the desired outcome to the AI and guiding the direction in which the AI learns. By setting the right rewards and punishments, AI can effectively construct LCD codes while avoiding the construction of substandard codes.

Specifically, the evaluator will first determine whether the constructed code is LCD code, if it is LCD code, it will evaluate the block error rate (BLER) of the constructed LCD code at the fixed signal-to-noise ratio (SNR) point through the hierarchical ordered decoding (OSD) device, and then evaluate the performance of the LCD code. Certainly, other performance metrics for LCD codes are also viable for evaluation. However, at the beginning, AI does not understand coding theory and does not understand LCD codes, and may construct non-LCD codes, in order to avoid this situation, AI will be given corresponding penalties when constructing non-LCD codes. Then the generator matrix of the constructed code is input into the RND algorithm module to evaluate the novelty of the state, and output the novelty as an intrinsic incentive. The RND module updates itself based on this state. Finally, the evaluator combines the external reward and the internal incentive, and feeds back to the construction module in the form of reward. This reward mechanism helps guide the AI to learn the process of constructing LCD codes, so that it gradually understands and follows the coding theory, while maintaining the novelty of the construction code.

## 3 Some examples of binary LCD codes

Suppose that the generator matrix of a binary LCD code  $\mathcal{C}$  with length n and dimension k is G. Meanwhile  $G = [I_k, P]$  is a standard generator matrix, where  $I_k$  is an identity matrix of size  $k \times k$ , and P is an ordinary matrix of size  $k \times (n - k)$ . Let  $\mathbb{F}_q$  be a finite field of order q,

where q is an odd prime number. An  $[n,k]_q$  linear code  $\mathcal{C}$  in  $\mathbb{F}_q$  is a subspace of  $\mathbb{F}_q^n$ . We call an  $[n,k,d]_q$  code  $\mathcal{C}$  optimal if there is no  $[n,k,d+1]_q$  code. An  $[n,k,d]_q$  code is called almost optimal if there exists an optimal  $[n,k,d+1]_q$  code [22]. The dual code of a code  $\mathcal{C}$  of length n in  $\mathbb{F}_q$  is defined as:

$$\mathcal{C}^{\perp} = \{ \mathbf{x} \in \mathbb{F}_q^n \mid \langle \mathbf{x}, \mathbf{y} \rangle = 0 \ \forall \ \mathbf{y} \in \mathcal{C} \},$$

where  $\mathbf{x} = (x_0, \dots, x_{n-1})$ ,  $\mathbf{y} = (y_0, \dots, y_{n-1})$ , and  $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=0}^{n-1} x_i y_i$ . A linear code  $\mathcal{C}$  over  $\mathbb{F}_q$  is called a *Linear Complementary Dual (LCD) code* if  $\mathcal{C} \cap \mathcal{C}^{\perp} = \{0\}$ .

The following lemma is very important for LCD codes.

**Lemma 3.1** (see [23]) Let G denote the generator matrix of a code C. Then C is LCD if and only if  $GG^T$  is nonsingular, where  $G^T$  is the transpose of G.

We're going to restrict AI to constructing codes that satisfy Lemma 3.1. Mainly, soft constraints are enforced through the use of rewards.

#### 3.1 Binary LCD codes

We denote the finite field of order 2 by  $\mathbb{F}_2 = \{0, 1\}$ , an [n, k, d] LCD code over  $\mathbb{F}_q$  by  $\mathcal{C}$ , and its generator matrix by  $[I_k | \mathcal{P}_q^{n,k}]$ . We use a fully connected neural network with one input layer, two hidden layers and one output layer. Details of each layer of the network are as follows:

- The input to the neural network is the generator matrix of binary LCD codes. Since the size of the generator matrix is  $k \times n$ , the number of input nodes of the neural network is kn. Set the initial status to  $s_0 = [I_k|0]$ .
- The output node number of the neural network is k(n-k). Action a is a real matrix of size  $k \times (n-k)$  sampled from the policy  $\pi_{\theta}$ . In order to construct a binary LCD code, sigmoid activation function is used to normalize the size of the action to the range of 0-1. If the value is greater than 0.5, element 1 in the binary field is selected; otherwise, element 0 in the binary field is selected.
- The number of hidden layers is 2k(n-k), and there are two hidden layers. Two hidden layers with appropriate activation functions can represent any decision boundary with any precision and can fit any smooth map with any precision.
- The reward is set at  $1 \lambda * BLER + \beta * r_i$ . Note that here  $\lambda$  is a constant, BLER is the block error rate calculated by OSD decoder, and  $r_i$  is the internal incentive generated by RND module. However, AI does not understand coding theory, nor does it understand LCD codes, and may construct non-LCD codes during the construction process. In this case, very small rewards are used to guide the agent. If the obtained code is non-LCD, which means that  $GG^T$  is a singular matrix, then the reward will be equal to  $-1 + \beta * r_i$ .

The PG algorithm for LCD code construction is also described in Algorithm 1.

In this paper, the Tianshou RL framework [24] is adopted, and the Adam optimizer and the update policy based on small-batch random gradient descent are used in the algorithm. The

## Algorithm 1 Policy gradient based LCD codes design

```
while 1 do

Set initial state s_0 = [I_k|0], \ \sigma^2 = 0.1

for step = 1, \cdots, maxstep do

\pi_{\theta} \leftarrow NeuralNet(\theta)

for i = 1, \cdots, k do

for j = 1, \cdots, n - k do

a_{i,j} \sim N(\pi_{\theta i,j}, \sigma^2)

P_{i,j} = (a_{i,j} \geqslant 0.5)?1:0

end for

end for

s_1 \leftarrow [I_k|P]

r \leftarrow Evaluator(s_1)

\pi_{\theta} \leftarrow f_N(a|\mu, \sigma^2)

end for

\theta \leftarrow agent.learn
end while
```

performance comparison between the constructed LCD code and the code constructed based on coding theory and the common linear code constructed based on RL is shown in Figure 3.

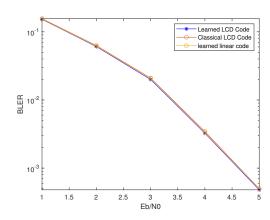


Figure 3 BLER comparison

Figure 3 shows the comparison of BLER calculated by OSD decoder for LCD codes constructed based on RL, almost optimal LCD codes constructed based on coding theory and ordinary linear codes constructed based on RL under AWGN channel condition when the code length is n=24 and the dimension is k=12. Among them, the almost optimal LCD codes constructed based on coding theory are constructed on the basis of [23,11] LCD code [25]. The BLER of LCD codes constructed based on RL is slightly lower than that of almost optimal LCD codes constructed based on coding theory, which indicates that the performance of LCD codes constructed based on AI technology can not only reach, but even surpass the performance

of LCD codes constructed based on classical construction algorithms based on coding theory under some channel conditions. In addition, compared with the method based on coding theory, the construction method based on RL does not need to rely on researchers' manual calculation and derivation, can realize automatic coding, and does not need to pay attention to the constraint between code length and dimension, so it is very flexible. The BLER of LCD code constructed based on RL in AWGN channel is lower than that of common linear code constructed by RL, that is, when using RL to construct linear code, adding structure to the code can further improve the performance of the constructed code. Compared with ordinary linear codes, LCD codes constructed by RL can be used not only in data error detection and error correction, but also in the construction of codes with other structures. For example, in coding theory, [n+1,k+1] LCD codes, quantum error correction codes and linear complementary pair of codes (LCP) can be constructed on the basis of [n,k] LCD codes. The usability of the constructed code is improved.

In coding theory, in addition to length n and dimension k, minimum distance d is also an important parameter of linear code, so [n,k,d] is often used to represent its parameters. There is a constraint relationship between the three parameters. However, when n and k are fixed, the larger d is, the better the performance of the code. Theoretically, the error correction code can check d-1 errors and correct  $\lfloor \frac{d-1}{2} \rfloor$  errors.

After experiments, using the minimum distance d as a reward can also construct an LCD code, and the minimum distance of the constructed LCD code will be as large as possible, and sometimes even reach the minimum distance limit.

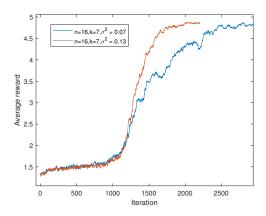


Figure 4 Comparison of different variances

In Figure 4, we compare the influence of  $\sigma^2$  value on the experiment. The figure shows that reducing the value of  $\sigma^2$  will slow down the learning process; On the contrary, appropriately increasing the value of  $\sigma^2$  will have a certain effect on the whole process, but too large  $\sigma^2$  will have the opposite effect. This is because  $\sigma^2$  controls the degree of exploration of RL in the construction process, and appropriate setting of parameters can accelerate the learning process.

When the minimum distance is rewarded, the construction method based on RL can construct the LCD code with the minimum distance as large as possible, and even reach the

minimum distance boundary. At present, the minimum distance bound of some LCD codes is not clear, only the range of existence of the bound is clear ([2, 25]). Through the proposed method, the existence of some almost optimal LCD codes is found to further shorten the range of minimum distance bounds. The specific results are shown in Table 1 and several examples below. Table 1 records the minimum distances and upper bounds of some constructed binary LCD codes:

n	k	bound	learned distance
19	13	3	3
23	10	7	6
24	11	7	6
29	11	8-9	8
33	12	10	9
37	9	12-14	13
39	9	12-16	14
40	9	12-16	13
40	10	12-16	13

 ${\bf Table~1} \ \ {\bf Minimum~distances~of~learned~binary~LCD~codes}$ 

**Example 3.2** Let  $\mathcal{C}$  be an [37,9] binary LCD code. As we know, optimal minimum distance of  $\mathcal{C}$  should be between 12 and 14. We find the existence of an LCD code with minimum distance of 13. The [37,9,13] binary LCD code  $\mathcal{C}$  has a generator matrix  $(I_9|\mathcal{P}_2^{37,9})$ 

**Example 3.3** Let  $\mathcal{C}$  be an [39,9] binary LCD code. As we know, optimal minimum distance of  $\mathcal{C}$  should be between 12 and 16. We find the existence of an LCD code with minimum distance of 14. The [39,9,14] binary LCD code  $\mathcal{C}$  has a generator matrix  $(I_9|\mathcal{P}_2^{39,9})$ 

**Example 3.4** Let  $\mathcal{C}$  be an [40,9] binary LCD code. As we know, optimal minimum distance of  $\mathcal{C}$  should be between 12 and 16. We find the existence of an LCD code with minimum distance of 13. The [40,9,13] binary LCD code  $\mathcal{C}$  has a generator matrix  $(I_9|\mathcal{P}_2^{40,9})$ 

#### 3.2 Some examples of ternary LCD codes

For LCD codes, the examination of both binary and ternary systems is found to be adequately enlightening, thereby motivating our endeavor to explore the realm of ternary LCD code construction. In the binary case, sigmoid activation functions are commonly used for binary classification. Therefore, we assumed that by establishing an appropriate mapping between actions and states, we could construct ternary LCD codes in a similar manner. And we denote the finite field of order 3 by  $\mathbb{F}_3 = \{0, 1, 2\}$ .

The algorithm for constructing ternary LCD codes is similar to that for constructing binary LCD codes, with one key difference. Specifically, we set the activation function of the output layer to softplus and apply a linear transformation to amplify its output values. This approach is necessary because we will be using modulo 3 to generate three elements in the finite field.

Similarly, the minimum range bounds of some ternary LCD code are uncertain. Through **Algorithm 2**, some ternary LCD codes are constructed to shorten the range of the current bounds, and even reach the bounds. Detailed data are shown in Table 2.

# Algorithm 2 PG based ternary LCD codes design

```
while 1 do
   Set initial state s_0 = [I_k|0], \, \sigma^2 = 0.3
   for step = 1, \cdots, maxstep do
      \mu \leftarrow NeuralNet(\theta)
      for i=1,\cdots,k do
          for j = 1, \dots, n - k do
             a_{i,j} \sim N(\mu_{i,j}, \sigma^2)
             \mathcal{P}_{i,j} = \lceil a_{i,j} \rceil \mod 3
          end for
      end for
      s_1 \leftarrow [I_k | \mathcal{P}]
      r \leftarrow Evaluator(s_1)
      \pi_{\theta} \leftarrow f_N(a|\mu,\sigma^2)
   end for
   \theta \leftarrow Update\ by\ Equation\ (2)
end while
```

Table 2 Minimum distances of learned ternary LCD codes

n	k	bound	learned distance
20	5	11	10
22	4	12	11
22	4	12-13	13
22	5	11-12	12
23	4	13-14	14
23	5	11-13	12
24	6	11-13	12
25	4	15-16	15
25	5	13-15	14

**Example 3.5** Let  $\mathcal{C}$  be an [22,4] ternary LCD code. As we know, optimal minimum distance of  $\mathcal{C}$  should be between 12 and 13. We find the existence of an LCD code with minimum distance of 13. The [22,4,13] binary LCD code  $\mathcal{C}$  has a generator matrix  $(I_4|\mathcal{P}_3^{22,4})$ 

**Example 3.6** Let  $\mathcal{C}$  be an [23,5] ternary LCD code. As we know, optimal minimum

distance of  $\mathcal{C}$  should be between 11 and 12. We find the existence of an LCD code with minimum distance of 12. The [23, 5, 12] binary LCD code  $\mathcal{C}$  has a generator matrix  $(I_5|\mathcal{P}_3^{23,5})$ 

**Example 3.7** Let  $\mathcal{C}$  be an [24,6] ternary LCD code. As we know, optimal minimum distance of  $\mathcal{C}$  should be between 11 and 13. We find the existence of an LCD code with minimum distance of 12. The [24,6,12] binary LCD code  $\mathcal{C}$  has a generator matrix  $(I_4|\mathcal{P}_3^{24,6})$ 

## 3.3 RND module

This section verifies that RND module, as an exploration method of internal stimulation, can add curiosity to the agent and drive it to explore in the environment. In the experiment, only internal incentives were used as reward signals, and Figure 5 shows the changes of internal incentives with continuous exploration.

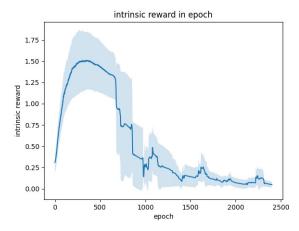


Figure 5 Intrinsic excitation variation

The first half of the curve describes that by making the agent curious about the unknown in the environment, the RND module encourages the agent to actively choose those actions that can maximize the prediction error, showing strong exploration behavior. This curiosity-driven approach to exploration makes agents more likely to discover new areas, new states, and more responsive to unknown environments. The second half of the curve shows that the agent can actively explore new behavior strategies even when facing a state similar to the known state, which helps to jump out of the local optimum.

In order to verify the influence of RND module on the construction process, we compared the construction effects of adding RND module and not adding RND module in the construction of [20,8] LCD codes through ablation experiments. The contrast reward changes are shown in Figure 6.

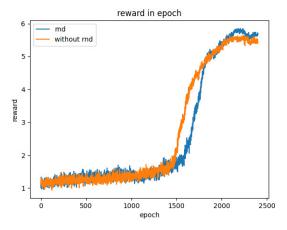


Figure 6 Performance comparison between with and without RND

The experimental results show that the number of convergent iterations with RND module is similar to that without RND module, indicating that RND module does not significantly increase the number of iteration rounds. However, with the addition of the RND module, the model achieved a higher average reward in the same number of iterations. By calculating the average reward of the last 100 rounds, it is found that the reward with RND module is 3.23% higher than that without RND module. This result shows that the RND module improves the performance of the model during training by guiding the agent to more efficient exploration. More importantly, the introduction of the RND module seems to help avoid falling into local optimal solutions.

#### 4 Conclusions

In this paper, by integrating coding theory into the reward setting of reinforcement learning, we successfully add structure to the codes constructed by reinforcement learning, and construct binary and ternary LCD codes. By setting appropriate rewards, AI can construct code with

a specified structure, further improving the performance and usability of the code; The multi-component LCD code is successfully constructed by reasonably setting the mapping of action to state, which means that the AI-based method is not limited to the binary field. Through the integration of the RND module within the evaluation mechanism, we have successfully augmented the model's exploratory capacity, leading to a subsequent enhancement in model performance.

Furthermore, conventional coding theory is based on optimizing parameters for code creation, whereas AI-driven coding relies on refining algorithms to enhance code performance. This entails employing advanced RL algorithms to further optimize code structure, and the progression of RL, along with its advanced ideas, also contributes to the advancement of ECC.

# Conflict of Interest

The authors declare no conflict of interest.

# Acknowledgements

This work was supported by the National Natural Science Foundation of China (Nos. 62372247, 12101326, 62206133) and the Open Project of Guangxi Provincial Key Laboratory (No. MIMS22-01).

#### References

- [1] Shannon C E, A mathematical theory of communication, in *The Bell System Technical Journal*, 1948, **27**(3): 379–423.
- [2] Bouyuklieva S, Optimal binary LCD codes, Des. Codes Cryptogr., 2021, 89: 2445–2461.
- [3] Kim H, Jiang Y, Kannan S, Oh S and Viswanath P, Deepcode: Feedback Codes via Deep Learning, in IEEE Journal on Selected Areas in Information Theory, 2020, 1(1): 194–206.
- [4] Gruber T, Cammerer S, Hoydis J, et al., On deep learning-based channel decoding, 2017 51st Annual Conference on Information Sciences and Systems (CISS), Baltimore, MD, USA, 2017.
- [5] Huang L, Zhang H, Li R, et al., AI Coding: Learning to Construct Error Correction Codes, in IEEE Transactions on Communications, 2020, 68(1): 26–39.
- [6] Burda Y, Edwards H, Storkey A and Klimov O, Exploration by random network distillation, in arXiv preprint arXiv:1810.12894, 2018.
- [7] Massey J, Linear codes with complementary duals, Discret Math, 1992, 106-107: 337-342.
- [8] Sendrier N, Linear codes with complementary duals meet the Gilbert–Varshamov bound, *Discrete Math*, 2004, **285**(1): 345–347.
- [9] Carlet C, Mesnager S, Tang C, et al., New characterization and parametrization of LCD codes, IEEE Trans. Inf. Theory, 2019, 65(1): 39–49.
- [10] Carlet C, Mesnager S, Tang C, Qi Y and Pellikaan R, Linear codes over  $\mathbb{F}_q$  are equivalent to LCD codes for q > 3, *IEEE Trans. Inf. Theory*, 2018, **64**(4): 3010–3017.

- [11] Li C, Ding C and Li S, LCD Cyclic Codes Over Finite Fields, in *IEEE Transactions on Information Theory*, 2017, 63(7): 4344–4356.
- [12] Bringer J, Carlet C, Chabanne H et al., Orthogonal direct sum masking, a smartcard friendly computation paradigm in a code, with builtin protection against side-channel and fault attacks, in Proc. WISTP, Heraklion, Crete, Greece, 2014.
- [13] Carlet C and Guilley S, Complementary dual codes for counter-measures to side-channel attacks, Springer-Verlag, Berlin, 2014.
- [14] Calderbank A R, Rains E M, Shor P W, et al., Quantum error correction and orthogonal geometry, Phys. Rev. Lett., 1997, 78(3): 405–408.
- [15] Calderbank A R, Rains E M, Shor P W and Sloane N J A, Quantum error correction via codes over GF(4), *Proceedings of IEEE International Symposium on Information Theory*, Germany, 1997.
- [16] Rajput C, Bhaintwal M and Bandi R, On cyclic LRC codes that are also LCD codes, 2020 5th International Conference on Computing, Communication and Security (ICCCS), Patna, 2020.
- [17] Ghosh H, Maurya P K and Bagchi S, Secret Sharing Scheme: Based on LCD Code, 2022 International Interdisciplinary Conference on Mathematics, Durgapur, 2022.
- [18] Franc ois-Lavet V, Henderson P, Islam R, Bellemare M G, and Pineau J, An introduction to deep reinforcement learning, *Now Foundations and Trends*, United States, 2018.
- [19] Mnih V, Kavukcuoglu K, Silver D, et al., Playing atari with deep reinforcement learning, in arXiv preprint arXiv:1312.5602, 2013.
- [20] Williams R J, Simple statistical gradient-following algorithms for connectionist reinforcement learning, *Mach. Learn.*, 1992, **8**(3): 229–256.
- [21] Konda V R and Tsitsiklis J N, Actor-critic algorithms, in *Proc. Adv. Neural Inf. Process. Syst.*, 2000.
- [22] Huffman W and Pless V, Fundamentals of Error-Correcting Codes. Cambridge University Press, Cambridge, 2003.
- [23] Wu Y and Lee Y, Binary LCD Codes and Self-Orthogonal Codes via Simplicial Complexes, in *IEEE Communications Letters*, 2020, **24**(6): 1159–1162.
- [24] Weng J Y, Chen H Y, Yan D, et al., Tianshou: A highly modularized deep reinforcement learning library, J. Mach. Learn. Res., 2022, 23(267): 1–6.
- [25] Li S, Shi M, Liu H, Several constructions of optimal LCD codes over small finite fields, Cryptogr. Commun., 2024, 16: 779–800.