Designing efficient fault-tolerant circuits for converting between two quantum codes is an effective way to realize universal fault-tolerant quantum computing. In this work, we adopt the round-robin circuit construction to design a conversion circuit between CSS 7-qubit error-correcting code and Reed-Muller 15-qubit code. We take the pieceable fault-tolerant protocol to guarantee the fault tolerance of our conversion circuit. A neural network decoding scheme is proposed to suppress the propagation of quantum errors. Our decoding scheme can effectively reduce the damaging effects of imperfect syndrome extraction operations and predict possible logical errors that occurred after the error correction process. We provide a detailed description of our syndrome data model and analyze the depolarizing noise threshold of our conversion circuit through numerical simulations of the stabilizer circuit.

I. INTRODUCTION

The practical quantum computation may be coming soon [1, 2]. Consequently, the prospect of large-scale quantum computers has generated great interest, as they can solve certain problems exponentially faster than computers that use the best-known classic algorithms [3–5]. Intermediate-scale quantum computers (NISQ) based on various technologies have been invented [6–9]. These experimental platforms provide the opportunity to implement some simple quantum algorithms [10–12] with the number of qubits ranging from 50 to a few hundred. However, given the limitations of current NISQ, high-fidelity quantum circuits for the compiling of certain quantum algorithms require a lot of quantum resources [12–14].

To protect quantum information from environmental noise and carry out large-scale computational tasks with high fidelity, it is necessary to carefully choose an appropriate method to prevent the possible propagation of quantum errors. Specifically, by combining physical qubits through the use of quantum error correction codes [15–17], we can use logical qubit to simulate the corresponding physical one. Consequently, we can use logical circuit to compile a quantum algorithm in a fault-tolerant manner [18].

Since the cost of implementing logical circuits is usually considered to be a major figure of merit of fault-tolerant quantum computation, many studies have focused on designing efficient fault-tolerant protocols. One of the most widely used and simplest such schemes is transversal. A logical quantum operation with transversal construction is automatically fault-tolerant, as each qubit acted upon by this logical circuit can interact with no more than one qubit per each code block. Besides, as there always exists a set of finite quantum gates such that any quantum circuit can be decomposed by a sequence of these gates with arbitrary precision, it would be very desirable to focus on the fault-tolerant design for such universal gate library.

However, for most universal fault-tolerant protocols, additional resources are always required due to the limitation of no-go theorem [19, 20]. This theorem has shown that any nontrivial quantum code doesn’t exist a universal set of transversal logical circuits. This fact encourages the development of other methods to circumvent this limitation. By fault-tolerantly preparing the magic states, and combining with the existing transversal Clifford gates, a logical T or Toffoli gate can be constructed fault-tolerantly. However, an additional distillation procedure needs to be applied to guarantee the quality of magic states, which consumes a large amount of auxiliary resources [21, 22]. By combining two different codes with complementary transversal gate sets into a single larger code, a universal gate library can be completed but with low efficiency [23, 24].

Other approaches involve code conversion techniques to bypass this limitation. By fault-tolerantly exchanging encoded data between two different codes $C_a$ and $C_b$, where each non-transversal logical gate in $C_a$ has a transversal implementation on $C_b$ and vice versa, a given universal gate library can be transversally implemented without magic states. For example, Steane’s 7-qubit code has transversal Hadamard gate $H$, controlled-NOT gate (CNOT) and phase gate $P$. Reed-Muller 15-qubit (RM-15) code has transversal $T$ gate, where $T = diag(1, exp(\frac{i\pi}{3})))$. The gate set $\{H, CNOT, P, T\}$ is a universal gate library. Therefore, we can adopt code conversion technology to realize a fault-tolerant universal logical gate library on these two codes. Anderson [20] designed a fault-tolerant conversion method for the fam-
ily of quantum Reed-Muller codes, their method can be used to design a fault-tolerant conversion circuit between Steane’s 7-qubit code and RM-15 code [27]. By exploiting measurement-based quantum computation, Muller [28] developed an algorithm for converting between any two different stabilizer codes and pointed out that the initial code distance will not be reduced during the application of their conversion circuit. Nevertheless, a conversion circuit usually requires considerable ancillary resources due to multiple rounds of error correction. Besides, it only acts on the single logical qubit, which makes physical qubits in a code block interact with each other too many times. Therefore, a single error can easily propagate and cause a logical fault, which reduces the anti-noise ability of the conversion circuit.

Better decoders result in higher error thresholds and can thus tolerate a larger noise rate. Some researchers have designed appropriate neural network decoders for the error correction of a logical qubit to effectively improve its fault tolerance [29,31]. But their methods are practical only when they store encoded information. In addition, these decoders also do not consider the errors caused by the imperfect error correction procedure. Different from the transversal structure, it has been proposed that a round-robin logical circuit can still be executed fault-tolerantly if it can be partitioned into several pieces with some carefully designed intermediate error correction processes. This protocol is also called pieceable fault-tolerant [32]. Meanwhile, as the syndrome data between different circuit pieces have specific correlations, we think that the event of error propagation can be effectively predicted with some machine-learning-based decoders. To summarize, we plan to propose a pieceable fault-tolerant conversion circuit between Steane’s 7-qubit code and RM-15 code and design a neural-network decoder to reduce its logical error rate.

In this paper, we first provide the details on how to construct a supervised multi-classification model on the error syndrome data of a pieceable fault-tolerant logical circuit; and explain how to use this model in the error correction process improve its noise immunity further. Then we give the logical CNOT gate between two logical qubits, where each of them is protected by a unique code, such as Steane’s 7-qubit code or RM-15 code. Based on the construction of these CNOT gates, we adopt pieceable fault-tolerant protocol to realize fault-tolerance and apply our syndrome data prediction model to the error correction process. Finally, we use these logical gates to design a code conversion circuit between these two codes. Our conversion circuit guarantees that all possible error propagation caused by a single-qubit error can be signaled without a complicated error-syndrome lookup table. To estimate the fault tolerance of our conversion circuit, we follow the simulation scheme in [33] and give an estimation of the depolarizing noise pseudo-threshold of our conversion circuit at the first level of encoding.

The paper is organized as follows. In Sec. III we describe the details of constructing a classification model for the syndrome data of a pieceable fault-tolerant circuit. In Sec. IV we express our pieceable fault-tolerant conversion circuit design and explain how to use deep neural network technology to improve its fault tolerance. In Sec. V we design several numerical simulation experiments to present the pseudo-threshold results and discuss the performance of our conversion circuit.

II. STABILIZER CODE AND FAULT-TOLERANT COMPUTATION

This section reviews definitions and preliminary results about quantum error correction code (QECC) and pieceable fault tolerant protocol. The stabilizer code is defined by a set of particular Pauli operators, also referred to as the stabilizer formalism developed by Gottesman [10].

Definition 1 (n-qubit Pauli group) A n-qubit Pauli group $G_n$ consists of the tensor products of single-qubit Pauli operators that can be described by the following set:

$$G_n = \{\lambda \sigma_1 \otimes \sigma_2 \otimes \cdots \otimes \sigma_n : \sigma_i \in \{I, X, Y, Z\}\} \cdots$$

, where $\lambda \in \{\pm 1, \pm \sqrt{-1}\}$.

Given an operator $g \in G_n$, its support is defined as a subset of $[n] := \{1, 2, \ldots, n\}$, where we denote by the symbol $\text{supp}(g)$ the set of all $i \in [n]$ such that $g$ acting on the $i$-th qubit is not identity. The weight of a Pauli operator $\text{wt}(g)$ equals the size of its support. For example, $\text{wt}(Z \otimes Z \otimes I) = \text{wt}(Z_0 Z_1) = 2$.

Next, we describe the stabilizer formalism. A stabilizer $S$ is an abelian subgroup belonging to $G_n$ which does not contain $-I$. Because the operators in this group are Hermitian and mutually commuting, they can be diagonalized simultaneously. Therefore, the stabilizer code can be defined by the following statement:

Definition 2 (n-qubit Stabilizer Code) Given a stabilizer subgroup $S < G_n$, a n-qubit stabilizer code $C_n$ is the joint $+1$ eigenspace of operators in $S$ belonging to the Hilbert space $(\mathbb{C}^2)^\otimes n$ and can be described by the following set:

$$C_n = \{|\psi\rangle : g|\psi\rangle = |\psi\rangle, \forall g \in S\}.$$
commute with all elements in $S$. We can first choose operators $Z_1, \ldots, Z_k$ and $X_1, \ldots, X_k$ in $C(S)$ that are independent of the generators of $S$ and satisfy the communication conditions $\hat{X}_i \hat{Z}_j = (-1)^{\delta_{ij}} \hat{Z}_j \hat{X}_i$. For convenience, we also refer to these operators as logical Pauli operators. Therefore, it is not difficult to say that the encoded $|0\rangle^\otimes k$ can be uniquely represented by the joint $+1$ eigenspace of the following set of $n$ operators: $\{g_1, \ldots, g_m, \hat{Z}_1, \ldots, \hat{Z}_k\}$. Similarly, we can give other basic encoded states by applying the corresponding logical Pauli $X$ operators to $|0\rangle^\otimes k$ such that $|x_1 \ldots x_k\rangle = \prod_{i=1}^{k} X_i^x |0\rangle^\otimes k$, where $x_i \in \{0, 1\}$.

### TABLE I. The generators of the stabilizer $S$ and logical Pauli operators of Steane’s $[[7;1;3]]$ code

| Stabilizer generators | Logical Pauli Operator |
|------------------------|------------------------|
| $X_0X_2X_4X_6$        | $X_1 = X^\otimes 7$    |
| $X_1X_3X_5X_7$        | $\hat{Z}_1 = Z^\otimes 7$ |
| $X_0Z_2Z_4Z_6$        |                         |
| $Z_1Z_3Z_5Z_7$        |                         |
| $Z_0Z_2Z_4Z_6$        |                         |

### TABLE II. The generators of the stabilizer $S$ and logical Pauli operators of Reed-Muller $[[15;1;3]]$ code

| Stabilizer generators | Logical Pauli Operator |
|------------------------|------------------------|
| $X_7X_8X_9X_{10}X_{11}X_{12}X_{13}X_{14}$ | $X_1 = X^\otimes 15$ |
| $X_3X_4X_5X_6X_{11}X_{12}X_{13}X_{14}$    | $\hat{Z}_1 = Z^\otimes 15$ |
| $X_1X_2X_3X_4X_5X_9X_{10}X_{13}X_{14}$    |                         |
| $X_0X_2X_4X_6X_{10}X_{12}X_{13}X_{14}$    |                         |
| $Z_7Z_9Z_{10}Z_{11}Z_{12}Z_{13}Z_{14}$    |                         |
| $Z_3Z_4Z_5Z_6Z_7Z_8Z_{11}Z_{12}Z_{13}Z_{14}$ |                         |
| $Z_1Z_3Z_5Z_7Z_{10}Z_{11}Z_{13}Z_{14}$    |                         |
| $Z_0Z_2Z_4Z_{10}Z_{12}Z_{13}Z_{14}$       |                         |
| $Z_{10}Z_{12}Z_{13}Z_{14}$                |                         |
| $Z_{11}Z_{12}Z_{13}Z_{14}$                |                         |
| $Z_{12}Z_{13}Z_{14}$                      |                         |

We show the stabilizer group of Steane’s 7-qubit code and RM-15 code in Table I and Table II. On the other hand, how to perform operations on an encoded state without losing the code’s protection is a highly important topic. We next give the definition of $t$-fault-tolerant; more details can be found in [16, 17].

**Definition 3** ($t$-fault-tolerant) For a $[[n;k;d]]$ code $C_n$, let $t = \lfloor \frac{d-1}{2} \rfloor$, a quantum operation which is protected by $C_n$ is $t$-fault-tolerant, if the following two conditions are satisfied:

(i) For an input codeword with an error of weight $w_1$, if $w_2$ single-qubit faults occur during the operation with $w_1 + w_2 \leq t$, ideally decoding the output state gives the same codeword as ideally decoding the input state.

(ii) For $w$ single-qubit faults during the implementation of a fault-tolerant operation with $w \leq t$, no matter how many errors are present in the input state, the output state differs from a codeword by an error with its weight no more than $w$.

Here, we say that ideally decoding is equivalent to a round of noise-free error correction. Both conditions are required to ensure that correctable errors do not propagate through the entire operation and prevent errors from accumulating during multiple rounds of error correction.

Next we introduce the pieceable fault tolerant protocol. It would be convenient to intuitively imagine that a specific encoded operation $C$ on $C_n$ can be decomposed into $r$ pieces:

$$C = C_rC_{r-1} \ldots C_1,$$

where parameter $r$ refers to the minimum number of circuit pieces that an encoding operation can be divided without the loss of fault-tolerance. More specially, we can obtain a fault-tolerant variant of $C$ if each $C_i$ is carefully designed such that certain uncorrectable errors can be signaled by some adapted error correction process $E_i$. So a modified $t$-fault-tolerant variant of $C$, denoted as $\tilde{C}$, can be described as follows:

$$\tilde{C} = E_rC_rE_{r-1} \ldots E_1C_1.$$

By performing adapted error correction after each $C_i$ on the encoded state, we obtain several fault-tolerant gadgets $E_i \cdot C_i$ ($i = 1, \ldots, r$).

To specifically explain these adjustment error correction processes, we would like to introduce some useful concepts. The first is contagious error; it is a type of Pauli error operators that can be described as:

$$E_C = \{ E \in G_n : \exists i \ s.t. [E, C_i] = EC_i - C_iE \neq 0 \}.$$

Only contagious error occurred in $C_i$ may propagate. For example, we assume that one of a circuit piece $C_m$ only contains a single physical control-Pauli $Z$ ($CZ$) gate with control qubit $a$ and target qubit $b$, and the input state at the $a$-th qubit has a Pauli $X$ error; then after the application of $C_m$, the input error will become a 2-qubit error $X \otimes Z$, i.e., $CZ(X \otimes I)CZ^\dagger = X \otimes Z$. So the single Pauli $X$ is a contagious error related to $C_i$, and the $Z$-type Pauli errors ($I \otimes Z, Z \otimes I, Z \otimes Z$) are non-contagious.

In each $E_i$ ($i = 1, \ldots, r-1$) we correct contagious errors immediately and left the non-contagious error until $E_r$. The syndrome information of $E_C$ will be recorded and sent to $E_r$ for the correction of non-contagious error. Here we also introduce the tool for correcting contagious error:

$$S_C = \{ g \in S : \forall i, [g, C_i] = 0 \}.$$

For a $r$-pieceable fault-tolerant circuit $C$, the elements of the stabilizer group satisfying the above conditions are called constant stabilizers.
III. MULTI-CLASSIFICATION MODEL FOR ERROR SYNDROME OF THE PIECEABLE FAULT-TOLERANT LOGICAL CIRCUIT

Next, we design a decoding scheme for the final error correction of a pieceable fault-tolerant logical circuit. For convenience, we first introduce the definition of error syndrome. The syndrome is defined as a classical bit-string for tracking error events of a fault-tolerant circuit, and it can be obtained by fault-tolerantly measuring stabilizer generators. Since the effect of most noise channel on the quantum state can be decomposed as a linear combination of Pauli operators [17], we only consider this kind of error throughout the paper. Assuming an error \( E \) has occurred on a \([n; k; d]\) encoded state, we then fault-tolerantly measure the stabilizer generators and obtain a syndrome vector as follows:

\[
s_i(E) = \begin{cases} 
1 & [E, g_i] \neq 0 \\
0 & [E, g_i] = 0 
\end{cases}, \quad i = 1, \ldots, n - k. \tag{7}
\]

Then the syndrome vector of \( E \) can be defined as a bit-string \( s(E) = (s_1(E), \ldots, s_{n-k}(E)) \in \mathbb{Z}_2^{n-k} \).

A typical decoding process can be described by matching the measured syndrome \( s \) with its most likely recovery operator \( R_s \) such that \( R_s E \in S \). More specifically, the recovery operator can be written as:

\[
R_s = \mathcal{L}(s) \mathcal{T}(s) \mathcal{G}(s), \tag{8}
\]

where operator \( \mathcal{G}(s) \in S \) and \( \mathcal{T}(s) \) belongs to an Abelian group composed of Pauli operators called pure error. This group has \( n - k \) generators and satisfies \([g_i, T_j] = \delta_{ij}, i, j = 1, \ldots, n - k\). The part of the recovery operator \( \mathcal{L}(s) \in N(S)/S \) belongs to the logical Pauli group acting on \( k \) logical qubits.

A decoder based on the minimum weight scheme can quickly infer the pure error \( \mathcal{T}(s) \) from the syndrome \( s \). However, this kind of decoders are not scalable for large quantum code and cannot infer potential logical errors. Actually, an optimal decoder should be able to infer the most likely logical fault \( \mathcal{L}(\mathcal{T}) \) from the known syndrome, such that:

\[
\mathcal{L}(\mathcal{T}) = \operatorname{argmax}_\mathcal{L} \{ P(\mathcal{L}|\mathcal{T}) \}. \tag{9}
\]

The pieceable fault-tolerant protocol shown in Eq. 4 makes the syndrome data of different intermediate error corrections have potential correlation. For example, the round-robin logical controlled-Z gate on \([5; 1; 3]\) code has been carefully designed to be fault-tolerant. The constant stabilizer syndromes of this gate can be further analyzed to infer some weight-2 \( Z \)-type Pauli errors [22]. This observation inspired us to design an extended decoding scheme to infer the possible error propagation event among several circuit pieces as much as possible. More specially, for the final error correction process \( \mathcal{E}_r \), we explain its decoding process as a pattern matching model so as to jointly analyze measurement information of \( r \) segmented error correction processes. Then this process would infer the most likely logical error if we effectively training the model functions. Therefore, we conclude that a optimal decoder of \( \mathcal{E}_r \) should be able to infer the logical error information \( \mathcal{L}(\mathcal{T}^{(1)}, \ldots, \mathcal{T}^{(r)}) \) such that:

\[
\mathcal{L}(\mathcal{T}^{(1)}, \ldots, \mathcal{T}^{(r)}) = \operatorname{argmax}_\mathcal{L} \{ P(\mathcal{L}|\mathcal{T}^{(1)}, \ldots, \mathcal{T}^{(r)}) \}
\]

where \( \mathcal{T}^{(i)} \) is the syndrome information of pure errors obtained from \( \mathcal{E}_i \).

Since all codes considered in this paper encode a single logical qubit, we first rewrite the recovery operator of \( \mathcal{E}_r \) as follows:

\[
R_{s_r} = \mathcal{L}(s) \mathcal{T}(s_r) \mathcal{G}(s_r), \tag{11}
\]

where \( s_i \) is the syndrome vector obtained by the error correction process of \( \mathcal{E}_i \) and \( s = s_1 \times \cdots \times s_r \). Single logical qubit Pauli group can be expressed as \( \langle i, \bar{X}_1, \bar{Z}_1 \rangle \), so the logical component can be decomposed as \( \mathcal{L}(s) = \mathcal{X}_1^a \mathcal{Z}_1^b \) regardless of the global phase factor, where \( a, b \in \mathbb{Z}_2 \).

Therefore, we introduce two functions \( g_X \) and \( g_Z \) such that:

\[
R_{s_r} = \mathcal{X}_1^{g_X(s)} \mathcal{Z}_1^{g_Z(s)} \mathcal{T}(s_r) \mathcal{G}(s_r). \tag{12}
\]

According to Eq. (12), we model the decoding process as a labeled classification task of discrete data. Then we define the data set as \( D \subseteq \{ s \} \times L \), and any element of \( D \) can be represented with the form \((s, l)\), where \( l \) is the label of class. Here we use the one-hot encoded label for \( k \) classes, i.e., \( l \in L = \{ l : l \in \{0, 1\}^k, 1^T l = 1 \} \). From Eq. (12), the output of model functions need to give the predicted value \( (g_X(s), g_Z(s)) \) based on all intermediate syndromes. So we take different predicted values as classification labels to corresponding to logical recovery operators, and denote these labels as \( \{ l_1, l_3, l_5, l_7 \} \).

For a \([n; k; d]\) code, we note that the logical operator of this code belongs to the following subgroup \( \mathcal{L} \subseteq \langle i, \bar{X}_1, \ldots, \bar{X}_k, \bar{Z}_1, \ldots, \bar{Z}_k \rangle \). Similarly, we can construct a multi-classification model with the number of \( 2^n \) categories for a \( r \)-pieceable fault-tolerant logical circuit, and the recovery operator of \( \mathcal{E}_r \) is given as follows:

\[
R_{s_r} = \mathcal{X}_1^{g_X^{(1)}(s)} \cdots \mathcal{X}_k^{g_X^{(k)}(s)} \mathcal{Z}_1^{g_Z^{(1)}(s)} \cdots \mathcal{Z}_k^{g_Z^{(k)}(s)} \mathcal{T}(s_r) \mathcal{G}(s_r). \tag{13}
\]

IV. PIECEABLE FAULT-TOLERANT CODE CONVERSION CIRCUIT

A. The construction of conversion circuit

Following the pieceable fault-tolerant protocol, we propose a conversion circuit between the Steane’s 7-qubit code \( C_7 \) and RM-15 code \( C_{15} \). First, we observe that a SWAP gate [17] between two entangled qubits can be
equivalently implemented by the circuit shown in Fig. 1.

For the information encoded in a code block, we can similarly construct a logical SWAP gate so that the encoded data can be exchanged between different logical qubits.

More specially, for an encoded qubit $|\tilde{\psi}\rangle = \alpha|0\rangle + \beta|1\rangle$ protected by $C_7$, when we want to transmit the encoded information to the logical qubit protected by $C_{15}$, we can first prepare a logical basic state $|0\rangle$ in $C_{15}$ and add it to the original system. Then we fault-tolerantly apply three logical CNOT gates to the combined system, as shown in Fig. 2. Finally, we finished this conversion procedure by tracing out the subsystem of $C_7$. Therefore, realizing fault-tolerant logical CNOT gates between $C_7$ and $C_{15}$ is the key for constructing this conversion circuit.

Nikahd et al. [34] has proposed a round-robin logical CNOT gate with $C_7$ encoded state as control and $C_{15}$ as target, but they didn’t give a specific solution to make this logical circuit fault-tolerant. Following their results, we divide this circuit into two pieces and insert a constant stabilizer error correction process between them to make the entire circuit fault-tolerant, as shown in Fig. 3. For the control code block, the stabilizer $Z_0Z_2Z_4Z_6Z_7Z_9$ and $Z_3Z_4Z_5Z_6$ are invariant under the conjugation of circuits $C_1$ and $C_2$, so these operators are constant stabilizers by definition. Similarly, for the target code block, those generators that only contain Pauli-X operator in Table I are constant stabilizers.

In contrast, we give a 2-pieceable fault-tolerant logical CNOT circuit with $C_{15}$ encoded state as control and $C_7$ encoded state as target, which is shown in Fig. 4. We have verified that the stabilizer $X_0X_2X_4X_6X_1X_2X_5X_3$ and $X_3X_4X_5X_6$ of the target code block are invariant under the conjugation of circuits $C_1$ and $C_2$. For the control code block, those generators that only contain Pauli-Z operator in Table I are constant stabilizers.

We have verified that these two logical CNOT circuits preserve tensor product of the stabilizer groups of two codes. Moreover, we verified that these circuits effect the appropriate transformation on the logical Pauli operators of two code blocks. For instance, the logical CNOT circuit shown in Fig. 3 has the following property:

$$\begin{align}
\bar{C}_A(\bar{X}_1 \otimes I)\bar{C}_A^4 & \rightarrow \bar{X}_0 \otimes \bar{Z}_4 , \\
\bar{C}_A(\bar{Z}_c \otimes I)\bar{C}_A^4 & \rightarrow \bar{Z}_c \otimes I .
\end{align}$$

(14)

$$\begin{align}
\bar{C}_A(\bar{I} \otimes \bar{X}_1)\bar{C}_A^4 & \rightarrow \bar{Z}_c \otimes \bar{X}_4 , \\
\bar{C}_A(\bar{I} \otimes \bar{Z}_2)\bar{C}_A^4 & \rightarrow \bar{I} \otimes \bar{Z}_4 .
\end{align}$$

(15)

B. The analysis of fault tolerance

By keeping the constant stabilizer syndrome data in $E_1$ and passing it to $E_2$, some uncorrectable errors caused by contagious error will have unique syndromes. Therefore, we can extend the error-syndrome lookup table to include more correspondences. This extended decoding strategy can ensure that $\bar{C}_A$ and $\bar{C}_B$ are at least 1-fault-tolerant circuits [22] [34]. A fault-tolerant logical circuit can be modeled as a leve-1 encoding extend rectangle (1-exRec) [15], as shown in Fig. 5. Based on this circuit model, an existing study [25] has designed a simulation scheme to calculate the pseudo-threshold value of a 1-exRec. Pseudo-threshold is defined as the intersection between the error rate line of unprotected gate and its corresponding logical variant, the higher the pseudo-threshold of the logical circuit, the better its anti-noise ability. Actually, from our numerical experiments we find that the above strategy cannot utilize the syndrome results of $E_i$ to infer possible logical error. Besides, the noisy LEC circuit may also introduce faults due to the unperfect quantum operations, but the minimal weight decoding procedure of $E_2$ can’t avoid the disturbance of these errors.

Our neural-network decoding scheme can be used to analyze the relationship between syndrome and error, and design an error diffusion early warning mechanism to correct possible logical errors. We construct a syndrome data classification model for these 2-pieceable fault-tolerant circuits. We use the deep neural network algorithm to train these models and apply them to the final error correction process. We then propose the following decoding process:

(i) During the implementation of $E_2$, we use the minimum weight decoding method to apply the pure error recovery operator $T(s_2)$;
(ii) Take the syndrome data $s = s_{LEC} \times s_1 \times s_2$ as model input to get the class value $g_X(s)$ and $g_Z(s)$, where we define $s_{LEC}$ as the syndrome of LEC circuit. Then we apply the logical recovery operator $L(s) = X_1^g_X(s)Z_1^{g_Z(s)}$.

1. Data set and labels

We take $\bar{C}_A$ 1-exRec as an example to explain how we generate the syndrome data needed to train the logical
error prediction model. Our training set is generated by the following steps:

(i) Prepare the initial encoded state and perform the LEC circuit in a depolarizing noise environment, and collect the syndrome vector $s_{LEC}$, where its dimension is 20.

(ii) After a round of ideal projection, we apply the noisy $\tilde{C}_A$ to the encoded state, and collect the syndrome vector $s_1$ and $s_2$ obtained during its application, where the dimension of $s_1$ is 7, $s_2$ is 20.

(iii) Perform a round of noise-free projection after the implementation of noisy TEC circuit $E_2$, which aims to make the uncorrectable error in the conversion circuit become a logical error, then we perform noise-free logical $Z$-based or $X$-based measurement to detect the logical error on the output state. Finally, we obtain the class label values of control and target code blocks.

We apply the depolarizing channel to $\tilde{C}_A$ 1-exRec with a sequence of physical error rates. All zero data are excluded from the training set.
Here we take the rectified linear unit as the active function \( f \), i.e., \( f(x) = \max\{0, x\} \). Then, we pass the output of the last hidden layer to the SoftMax layer to calculate the probability that \( s \) belongs to the \( j \)-th category. We denote \( \tilde{P} = (\tilde{P}_1, \ldots, \tilde{P}_k) \), and adapt the following equation to obtain this

\[
\tilde{P}_j = \frac{e^{V_j}}{\sum_{j=1}^k e^{V_j}}, \quad j = 1, \ldots, k,
\]

where \( V = (V_1, \ldots, V_k) = \omega^{(M)} H_M + b^{(M)} \). Finally, we predict the class label vector \( \hat{I} = (\hat{I}^{(1)}, \ldots, \hat{I}^{(k)}) \) of \( s \) by:

\[
\hat{s}^{(i)} = \begin{cases} 
1 & \text{for } i = \arg\max \{ \tilde{P}_j \} \\
0 & \text{otherwise}.
\end{cases}
\]

In contrast, reducing the empirical error usually makes the model over-fit the training data and gives a more complicated model. Such a case will greatly damage the generalization of the model and cause additional time consumption. Therefore, it is usually necessary to add an extra term to the objective function to reduce the structural error of the model; this term is also called regularization. Commonly used regularization methods are \( L1 \) regularization and \( L2 \) regularization. Here we use \( L2 \) regularization, as shown in Eq. (17), which will limit the coefficient \( \omega \) in the model function to reduce the complexity of the model. A hyper-parameter \( \lambda \) is also introduced to balance loss function and regularized weight.

For the training details of \( \widetilde{C}_A \), we construct two binary classification models for the two types of data labeled by \( g_X(s) \) and \( g_Z(s) \). The dimension of input data is 47, and each model has 4 hidden layers and the number of hidden layer nodes are 256, 512, 1024 and 256. We set the batch size is 30, the learning rate is \( 10^{-4} \). We train our models on pytorch.

V. THRESHOLD ANALYSIS FOR THE CONVERSION CIRCUIT

We next analyze the performance of our conversion circuit through the stabilizer circuit simulation algorithms [37]. Our simulation experiments are executed on the platform called LIQUI\[38\]. We show the pseudo-threshold results of \( \widetilde{C}_A \) and \( \widetilde{C}_B \) equipped with our neural-network decoding procedure. The fault-tolerant performance of the code conversion circuit composed of these two logical circuits is further analyzed through simulation experiments. We first introduce the concept of malignant error event [39], which can be defined as follows: Let \( |\psi⟩ \) be the input encoded state. As can be seen from Fig. 4, the LEC circuit is first applied to the input state, and then we ideally project the output state of the LEC circuit into the original code space. Let \( |\psi_{LEC}⟩ \) be the state after this projection, and |LM⟩ be the
I curs, where \( E \) be defined as the logical failure rate of the SWAP circuit as follows:

Thus, we give an estimate about SWAP circuit. In fact, due to the error propagation, any \( P \) where \( \epsilon \) is the physical error probability that satisfies \( P \) represents all the possible malignant error events that a two-qubit Pauli gate will occur, i.e., with a probability of \( \frac{1}{16} \), a two-qubit Pauli gate drawn uniformly and independently from \( \{I, X, Y, Z\} \otimes \{I, X, Y, Z\} \{I \otimes I\} \), and \( P(E) \) is the probability of logical error \( E \) with a given physical error rate. Therefore, the pseudo-threshold of a 1-exRec is the physical error probability that satisfies \( \epsilon = \sum E P(E) \).

Following the above definition, we further expand it and define the failure rate of the logical fault-tolerant SWAP circuit. In fact, due to the error propagation, any logical fault in one of three CNOT 1-exRec will cause the entire process to be fail. Thus, we give an estimate about the logical failure rate of the SWAP circuit as follows:

\[
P_{\text{swap}} = 2P_A(1 - P_B)(1 - P_A) + P_A(1 - P_B)^2, \tag{20}
\]

where \( P_A \) is the logical error rate of \( \tilde{C}_A \) 1-exRec and \( P_B \) is the logical error rate of \( \tilde{C}_B \) 1-exRec. We also give a method to calculate the pseudo-threshold of a logical SWAP gate. Similarly, we define its failure event as the failure of any CNOT gate it contains. Since the basic assumption of our error model is that the error probability of any physical operation is the same, then the physical failure probability of the SWAP gate is:

\[
\epsilon_{\text{swap}} = 3\epsilon(1 - \epsilon^2).
\]

So, the pseudo-threshold of a fault-tolerant logical SWAP circuit is the physical error probability \( \epsilon \) that satisfies \( \epsilon_{\text{swap}} = P_{\text{swap}} \). The logical error rate of a 1-exRec can be numerically computed by Monte Carlo method. We next describe our numerical calculation scheme. Our simulation design has the following two basic assumptions: First, since the error rate in classical computers is usually very low (per operation), we assume that classical information can be ideally protected and do not consider errors in classical computers in our simulation experiments. Second, to ensure the full use of resources, we assume that in every single logical qubit, the auxiliary state that passes the verification can be reused before it is measured to be a classical bit.

For our simulation scheme, we construct depolarization noise model for threshold calculation, that is, apply the following noise channel to each noise physical component in a 1-exREC:

\[
\epsilon_\rho = \left(1 - \frac{3\epsilon}{4}\right)\rho + \frac{\epsilon}{4}(X\rho X + Z\rho Z + Y\rho Y).
\]

The basic noise components in our simulation are listed in Table III. Here we make the assumption that the physical error rate of a two-qubit gate and one-qubit gate being the same. With the basic assumption described above, we now give the method of logical failure rate for \( \tilde{C}_A \) or \( \tilde{C}_B \) as follows:

(i) First, we choose a sequence of physical error rates ranged from \( 10^{-4} \) to \( 10^{-2} \). For each fixed physical error rate \( \epsilon \), we simulate the \( \tilde{C}_A \) or \( \tilde{C}_B \) under a depolarizing noise channel. The simulation is valid if all the ancillary blocks in a 1-exRec pass the verification.

(ii) When a noisy logical CNOT gate fails, it applies an ideal CNOT gate followed by one of the 15 nontrivial two-qubit logical Pauli operators. Thus, for each possible logical error \( E \), we prepare an appropriate initial encoded state and use the Monte Carlo method to estimate its conditional probability when the physical error rate is fixed. Here, we set the number of simulation \( N = 10^6 \) and denote the logical error rate as \( P(E|\text{cnot}, \epsilon) = \frac{m}{N} \), where \( m \) is the number of logical errors \( E \) occurring after \( N \) iterations.

We display our simulation results in Fig. 8 Fig. 9 and Fig. 10. Polynomial fitting algorithm is used to obtain two types of 1-exRec logical error curves, and it can be found that the best fitting effect can be achieved by using quadratic function fitting. Actually, for an effective fault-tolerant quantum operation, its logical error rate should be smaller than the error rate of unprotected one, there also has been proved that if error propagation through the entire fault-tolerant circuit is limited, and a good

| Physical noisy locations | Component type |
|-------------------------|----------------|
| 1 | Basic state (|0\rangle) preparation |
| 2 | Basic state (|+) preparation |
| 3 | Measurement of Pauli operator |
| 4 | Two-qubit quantum gate |
| 5 | Single-qubit non-identity quantum gate |

The logical circuit minimal weight neural-network decoding strategy and neural-network based decoding strategy.

| Logical circuit | Minimal weight neural-network |
|-----------------|------------------------------|
| \( \tilde{C}_A \) | \( 9.36 \times 10^{-4} \) | \( 1.06 \times 10^{-4} \) |
| \( \tilde{C}_B \) | \( 1.99 \times 10^{-6} \) | \( 1.98 \times 10^{-4} \) |
| Fault-tolerant SWAP circuit | \( 3.26 \times 10^{-6} \) | \( 1.07 \times 10^{-4} \) |
decoder is used, the logical error rate should exhibit the power law scaling [40], which means code concatenation techniques can be adapted to exponentially reduce the logical error rate.

From our numerical simulation results, we observe that the fitting curve of the logical error rate of $\tilde{C}_A$ and $\tilde{C}_B$ based on two decoding strategies show the characteristics of a quadratic curve. We think this is a reasonable numerical simulation result. Because the code distance of $C_7$ and $C_{15}$ is 3, decoders for these code are effective if they can correct most of the single-qubit error, such that the logical error can only be caused by two or more single-qubit errors that occur independently. In contrast, our decoding strategy can more effectively reduce the logical error rate of the pieceable fault-tolerant circuit, so it leads to a higher threshold result, as shown in Table IV.

VI. CONCLUSIONS

In summary, the main contribution of this work is to combine the pieceable fault-tolerant protocol to design an efficient code conversion circuit, which enables the exchange of information between two logical qubits protected by different error-correcting codes. Because the error that occurs on a pieceable fault-tolerant logical gate is prone to propagate, the error threshold of such gate is relatively lower than those who with transversal construction.

Therefore, we first extract the effective features of the syndrome data that is collected from different circuit pieces and introduce the classical pattern recognition algorithm in the decoding process. Such treatment actually transforms the prediction of error propagation into a supervised multi-classification task. Then, we extend the decoder of a pieceable fault-tolerant with a trained logical error discrimination model. Finally, we simulate this pieceable fault-tolerant code conversion circuit under the depolarization noise to obtain corresponding pseudo-threshold results.

These results have shown that our decoding process can effectively suppress the logical error rate compared to the minimum weight method. So we think that our fault-tolerant design of this conversion circuit has a certain practicability on the current NISQ. Effective pieceable fault-tolerant logical gate can be further introduced to universal computing scheme based on code concatenation, such as constructing a non-transversal but fault-tolerant non-Clifford gate on a 2-level concatenated code. Besides, with the consideration of the noise in the current
FIG. 10. The rate of the sum of logical errors for our fault-tolerant logical SWAP circuit between Steane’s 7-qubit code and RM 15-qubit code.

experimental platform, we will use the statistical learning methods to effectively analyze its impact on the output of circuit in our future work.

ACKNOWLEDGMENTS

This work is supported by the National Key R&D Program of China under Grant No. 2018YFA0306703 and the Natural Science Foundation of Guangxi under Grant No. 2019GXNSFAA185033.

[1] J. Preskill, Quantum computing in the nisq era and beyond, Quantum 2, 79 (2018).
[2] C.Neill, P. Roushan, K. Kechedzhi, S. Boixo, S. V. Isakov, V. Smelyanskiy, A. Meegrant, B. Chiaro, A. Dunsworth, K. Arya, et al., a blueprint for demonstrating quantum supremacy with superconducting qubits, Science 360, 195 (2018).
[3] R. P. Feynman, Simulating physics with computers, International journal of theoretical physics 21, 467 (1982).
[4] E. Farhi, J. Goldstone, and S. Gutmann, A quantum approximate optimization algorithm, arXiv preprint arXiv:1411.4028 (2014).
[5] P. W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM review 41, 303 (1999).
[6] H. Häffner, C. F. Roos, and R. Blatt, Quantum computing with trapped ions, Physics reports 469, 155 (2008).
[7] C. J. Ballance, T. P. Harty, N. M. Linke, M. A. Sepiol, and D. M. Lucas, High-fidelity quantum logic gates using trapped-ion hyperfine qubits, Phys. Rev. Lett. 117, 060504 (2016).
[8] S. Debnath, N. M. Linke, C. Figgatt, K. A. Landsman, K. Wright, and C. Monroe, Demonstration of a small programmable quantum computer with atomic qubits, Nature 536, 63 (2016).
[9] M. H. Devoret and R. J. Schoelkopf, Superconducting circuits for quantum information: an outlook, Science 339, 1169 (2013).
[10] A. M. Childs, R. Kothari, and R. D. Somma, Quantum algorithm for systems of linear equations with exponentially improved dependence on precision, SIAM Journal on Computing 46, 1920 (2017).
[11] S. Boixo, S. V. Isakov, V. N. Smelyanskiy, R. Babbush, N. Ding, Z. Jiang, M. J. Brenner, J. M. Martinis, and H. Neven, Characterizing quantum supremacy in near-term devices, Nature Physics 14, 595 (2018).
[12] E. T. Campbell, B. M. Terhal, and C. Vuillot, Roads towards fault-tolerant universal quantum computation, Nature 549, 172 (2017).
[13] D. Gottesman, Fault-tolerant quantum computation with constant overhead, Quantum Information & Computation 14, 1338 (2014).
[14] M. Raginsky, Scaling and renormalization in fault-tolerant quantum computers, Quantum Information Processing 2, 249 (2003).
[15] D. E. Gottesman, Stabilizer Codes and Quantum Error Correction, Ph.D. thesis, California Institute of Technology (1997).
[16] D. Gottesman, An introduction to quantum error correction and fault-tolerant quantum computation, in Quantum information science and its contributions to mathematics, Proceedings of Symposia in Applied Mathematics, Vol. 68 (2010) pp. 13–58.
[17] M. A. Nielsen and I. L. Chuang, Quantum computation and quantum information (2000).
[18] P. Aliferis, D. Gottesman, and J. Preskill, Quantum accuracy threshold for concatenated distance-3 codes, Quantum Information & Computation 6, 97 (2006).
[19] B. Eastin and E. Knill, Restrictions on transversal encoded quantum gate sets, Physical review letters 102, 110502 (2009).
[20] B. Zeng, A. Cross, and I. L. Chuang, Transversality versus universality for additive quantum codes, IEEE Transactions on Information Theory 57, 6272 (2011).
[21] S. Bravyi and A. Kitaev, Universal quantum computation with ideal clifford gates and noisy ancillas, Physical Review A 71, 022316 (2005).
[22] J. O’Gorman and E. T. Campbell, Quantum computation with realistic magic-state factories, Physical Review A 95, 032338 (2017).
[23] A. G. Fowler, S. J. Devitt, and C. Jones, Surface code implementation of block code state distillation, Scientific reports 3, 1939 (2013).
[24] T. Jochym-O’Connor and R. Laflamme, Using concate-
nated quantum codes for universal fault-tolerant quantum gates, Physical review letters 112, 010505 (2014).

[25] C. Chamberland, T. Jochym-O’Connor, and R. Laflamme, Overhead analysis of universal concatenated quantum codes, Physical Review A 95, 022313 (2017).

[26] J. T. Anderson, G. Duclos-Cianci, and D. Poulin, Fault-tolerant conversion between the steane and reed-muller quantum codes, Physical review letters 113, 080501 (2014).

[27] C. Lin and G. Yang, Concatenated pieceable fault-tolerant scheme for universal quantum computation, Physical Review A 102, 052415 (2020).

[28] K. R. Colladay and E. J. Mueller, Rewiring stabilizer codes, New Journal of Physics 20, 083030 (2018).

[29] G. Torlai and R. G. Melko, Neural decoder for topological codes, Physical review letters 119, 030501 (2017).

[30] A. S. Darmawan and D. Poulin, Tensor-network simulations of the surface code under realistic noise, Physical review letters 119, 040502 (2017).

[31] P. Baireuther, T. E. O’Brien, B. Tarasinski, and C. W. Beenakker, Machine-learning-assisted correction of correlated qubit errors in a topological code, Quantum 2, 48 (2018).

[32] T. J. Yoder, R. Takagi, and I. L. Chuang, Universal fault-tolerant gates on concatenated stabilizer codes, Physical Review X 6, 031039 (2016).

[33] C. Chamberland, T. Jochym-O’Connor, and R. Laflamme, Architectural analysis of universal concatenated quantum codes, (2016).

[34] E. Nikahd, M. S. Zamani, and M. Sedighi, Low-overhead code concatenation approaches for universal quantum computation, arXiv preprint arXiv:1707.00981 (2017).

[35] C. M. Bishop, Pattern Recognition and Machine Learning (Information Science and Statistics) (Springer-Verlag New York, Inc., 2006).

[36] Y. Lecun, Y. Bengio, and G. Hinton, Deep learning, Nature 521, 436 (2015).

[37] S. Aaronson and D. Gottesman, Improved simulation of stabilizer circuits, Physical Review A 70, 052328 (2004).

[38] D. Wecker and K. M. Svore, Liqui |: A software design architecture and domain-specific language for quantum computing, arXiv preprint arXiv:1402.4467 (2014).

[39] C. Chamberland, T. Jochym-O’Connor, and R. Laflamme, Thresholds for universal concatenated quantum codes, Physical review letters 117, 010501 (2016).

[40] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, Surface codes: Towards practical large-scale quantum computation, Physical Review A 86, 032324 (2012).