Neural network-based prediction of the secret-key rate of quantum key distribution

Hua-Lei Yin (hlyin@nju.edu.cn)
Nanjing University
Min-Gang Zhou
Nanjing University
Zhi-Ping Liu
Nanjing University
Wen-Bo Liu
Nanjing University
Chen-Long Li
Nanjing University
Jun-Lin Bai
Nanjing University
Yi-Ran Xue
Nanjing University
Yao Fu
MatricTime Digital Technology Co. Ltd
Zeng-Bing Chen
Nanjing University

Article

Keywords:

Posted Date: March 23rd, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1462416/v1

Neural network-based prediction of the secret-key rate of quantum key distribution

Min-Gang Zhou,[1] Zhi-Ping Liu,[1] Wen-Bo Liu,[1] Chen-Long Li,[1,2] Jun-Lin Bai,[1,2] Yi-Ran Xue,[1,2] Yao Fu,[2] Hua-Lei Yin,[1,*] and Zeng-Bing Chen[1,2,†]

[1] *National Laboratory of Solid State Microstructures, School of Physics, Collaborative Innovation Center of Advanced Microstructures, Nanjing University, Nanjing, China*

[2] *MatricTime Digital Technology Co. Ltd., Nanjing, China*

(Dated: March 21, 2022)

Numerical methods are widely used to calculate the secure key rate of many quantum key distribution protocols in practice, but they consumes many computing resources and are too time-consuming. In this work, we take the homodyne detection discrete-modulated continuous-variable quantum key distribution (CV-QKD) as an example, and construct a neural network that can quickly predict the secure key rate based on the experimental parameters and experimental results. Compared to traditional numerical methods, the speed of the neural network is improved by several orders of magnitude. Importantly, the predicted key rates are not only highly accurate but also highly likely to be secure. This allows the secure key rate of discrete-modulated CV-QKD to be extracted in real time on a low-power platform. Furthermore, our method is versatile and can be extended to quickly calculate the complex secure key rates of various other unstructured quantum key distribution protocols.

With the concurrent rise of artificial intelligence and quantum information science, these two fields are merging in a synergistic manner. In this growing trend, some works try to design new theoretical models based on quantum algorithms to improve classical machine learning for desired quantum speed-up [1–10]. At the same time, with the ever-increasing complexity of quantum systems, advanced quantum information technologies also require powerful tools for data processing and data analysis. We therefore urgently need to leverage existing classical machine learning techniques to solve practical, but difficult, problems in quantum information science, such as tomography [11–13], classifying quantum states [14–16], quantum metrology [17–19], quantum control [20, 21] and quantum cryptography[22].

Quantum key distribution (QKD) [23, 24] is by far the most practical technology in quantum information. It allows two distant parties (Alice and Bob) to establish secure keys against any eavesdropper. Various QKD protocols have been proposed one after another in recent decades [25]. Calculating the secure key rates of these QKD protocols is typically done by analytical methods [26], but these analytical methods are usually inseparable from certain symmetry assumptions. These assumptions are often broken by experimental imperfections in practice. Therefore, to analyze the security of QKD protocols that are more suitable for practical implementations, some numerical methods based on convex optimization [27–30] have been developed.

For instance, continuous-variable (CV) QKD has its own distinct advantages at a metropolitan distance [31] due to the use of common components of coherent optical communication technology. In addition, the homodyne [32] or heterodyne [33] measurements used by CV-QKD have inherent extraordinary spectral filtering capabilities, which allows the crosstalk in wavelength division multiplexing (WDM) channels to be effectively suppressed. Therefore, hundreds of QKD channels may be integrated into a single optical fiber and can be cotransmitted with classic data channels. This allows QKD channels to be more effectively integrated into existing communication networks. In CV-QKD, discrete modulation technology has attracted much attention [26, 34–44] because of its ability to reduce the requirements for modulation devices. However, due to the lack of symmetry, the security proof of discrete modulation CV-QKD also mainly relies on numerical methods [37–42, 45].

Unfortunately, calculating a secure key rate by numerical methods requires minimizing a convex function over all eavesdropping attacks related with the experimental data [46, 47]. The efficiency of this optimization depends on the number of parameters of the QKD protocol. For example, in discrete modulation CV-QKD, the number of parameters is generally $1000 - 3000$ depending on the different choices of cutoff photon numbers [38]. This leads to the corresponding optimization possibly taking minutes or even hours [45]. Therefore, it is especially important to develop tools for calculating the key rate that are more efficient than numerical methods.

In this work, we take the homodyne detection discrete-modulated CV-QKD [38] as an example to construct a neural network capable of predicting the secure key rate for the purpose of saving time and resource consumption. We apply our neural network to a test set obtained at different excess noises and distances. Excellent accuracy and time savings are observed after adjusting the hyperparameters. Importantly, the predicted key rates are highly likely to be secure. Note that our method is versatile and can be extended to quickly calculate the complex secure key rates of various other unstructured quantum key distribution protocols. Through some open source deep learning frameworks for

on-device inference, such as TensorFlow Lite [48], our model can also be easily deployed on devices at the *edge* of the network, such as mobile devices, embedded Linux or microcontrollers.

RESULTS

Discrete-modulated CV-QKD. To clearly show the problem we try to solve, we briefly introduce the main ideas of discrete-modulated CV-QKD and give the convex optimization problem of finding its key rates in this section. See Ref. [38] and Appendix A for a detailed description of discrete-modulated CV-QKD.

The protocol involves two parties, Alice and Bob. Alice randomly prepares one of the four coherent states and sends it to Bob by an untrusted quantum channel. Bob measures the received coherent state using homodyne detection. After repeating N rounds, Alice and Bob perform sifting, parameter estimation, error correction and privacy amplification over the classical authentication channel to obtain the final secure key rates. The key rate formula in the asymptotic limit can be expressed according to Refs. [27, 28] as

$$R^{\infty}=\min_{\rho_{AB}\in\mathbf{S}} D\left(\mathcal{G}\left(\rho_{AB}\right)\|\mathcal{Z}\left[\mathcal{G}\left(\rho_{AB}\right)\right]\right)-p_{\mathrm{pass}}\delta_{\mathrm{EC}}, \tag{1}$$

where $D(\rho\|\sigma)=\mathrm{Tr}\left(\rho\log_2\rho\right)-\mathrm{Tr}\left(\rho\log_2\sigma\right)$ is the quantum relative entropy; ρ_{AB} is the bipartite state of Alice and Bob; $\mathcal{G}$ is the mapping to describe the postprocessing of the bipartite state ρ_{AB}; $\mathcal{Z}$ is a pinching quantum channel for reading out the results of the key rate mapping; $\mathbf{S}$ is the set of all density operators that match the experimental observations; p_{pass} is a sifting factor that determines how many rounds of data are used for generating keys; δ_{EC} represents the amount of information leakage per bit in the error-correction process.

The key to finding the secure key rates is to solve the minimum value of $D\left(\mathcal{G}\left(\rho_{AB}\right)\|\mathcal{Z}\left[\mathcal{G}\left(\rho_{AB}\right)\right]\right)$, since $p_{\mathrm{pass}}\delta_{\mathrm{EC}}$ is a fixed quantity. The associated optimization problem is [38]

$$\begin{aligned}
&\mathrm{minimize} D\left(\mathcal{G}\left(\rho_{AB}\right)\|\mathcal{Z}\left[\mathcal{G}\left(\rho_{AB}\right)\right]\right)\\
&\text{subject to}\\
&\qquad \mathrm{Tr}\left[\rho_{AB}\left(|x\rangle\langle x|_A\otimes\hat{q}\right)\right]=p_x\langle\hat{q}\rangle_x\ ,\\
&\qquad \mathrm{Tr}\left[\rho_{AB}\left(|x\rangle\langle x|_A\otimes\hat{p}\right)\right]=p_x\langle\hat{p}\rangle_x\ ,\\
&\qquad \mathrm{Tr}\left[\rho_{AB}\left(|x\rangle\langle x|_A\otimes\hat{n}\right)\right]=p_x\langle\hat{n}\rangle_x\ ,\\
&\qquad \mathrm{Tr}[\rho_{AB}(|x\rangle\langle x|_A\otimes\hat{d})]=p_x\langle\hat{d}\rangle_x,\\
&\qquad \mathrm{Tr}\left[\rho_{AB}\right]=1,\\
&\qquad \rho_{AB}\geq 0,\\
&\qquad \mathrm{Tr}_B\left[\rho_{AB}\right]=\sum_{i,j=0}^{3}\sqrt{p_ip_j}\left\langle\varphi_j\mid\varphi_i\right\rangle|i\rangle\langle j|_A\ ,
\end{aligned} \tag{2}$$

where $|x\rangle\langle x|_A$ is a local projective measurement operator of Alice's side, where $x\in\{0,1,2,3\}$; $\hat{q}=\frac{1}{\sqrt{2}}\left(\hat{a}^{\dagger}+\hat{a}\right)$, where $\hat{a}$ and $\hat{a}^{\dagger}$ are the annihilation and creation operators of a single-mode state, respectively; $\hat{p}=\frac{i}{\sqrt{2}}\left(\hat{a}^{\dagger}-\hat{a}\right)$; $\hat{n}=\frac{1}{2}\left(\hat{q}^2+\hat{p}^2-1\right)=\hat{a}^{\dagger}\hat{a}$; $\hat{d}=\hat{q}^2-\hat{p}^2=\hat{a}^2+\left(\hat{a}^{\dagger}\right)^2$; $\langle\hat{q}\rangle_x$, $\langle\hat{p}\rangle_x$, $\langle\hat{n}\rangle_x$ and $\langle\hat{d}\rangle_x$ represent the corresponding expectation values of the operators $\hat{q}$, $\hat{p}$, $\hat{n}$ and $\hat{d}$ acting on ρ_B^x, respectively; $\rho_B^x=\frac{1}{p_x}\mathrm{Tr}_A\left[\rho_{AB}\left(|x\rangle\langle x|_A\otimes\mathrm{id}_B\right)\right]$ is the state of Bob after Alice has performed measurement $|x\rangle\langle x|$ on ρ_{AB}, and p_x is the corresponding probability; id_B is the identity transformation acting on system B.

The first four constraints in Eq. (2) are derived from experimental observations. The fifth and sixth constraints are conditions that the density matrix must satisfy. The seventh constraint comes from the fact that Alice's states do not change because they do not go through insecure quantum channels.

The optimization problem in Eq. (2) is to find the optimal ρ_{AB} in $\mathbf{S}$ such that R^{∞} is minimized. ρ_{AB} is infinite-dimensional because the attacker has the ability to arbitrarily perturb the optical mode sent by Alice into an infinite-dimensional state to send to Bob. To solve this optimization problem using numerical methods, we need to apply the photon-number cutoff assumption to ρ_{AB} to ensure that the number of variables is in a reasonable range. A detailed description of this method can be found in Ref. [38].

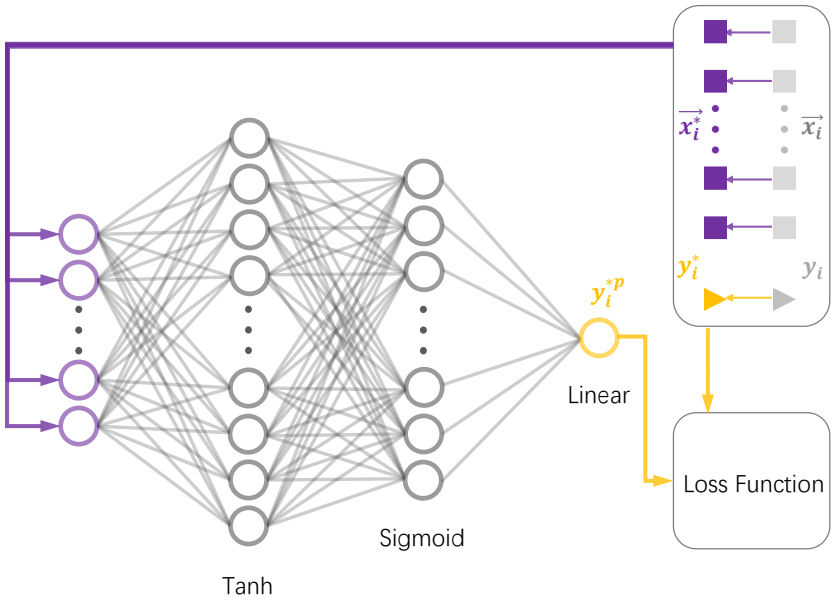

FIG. 1. Schematic diagram of our neural network model. We preprocess each training input $\vec{x}_i$ and its corresponding label y_i to obtain $\vec{x}_i^*$ and y_i^*. The neural network receives $\vec{x}_i^*$ and outputs the corresponding y_i^{*p}. The numbers of neurons in the first hidden layer and the second hidden layer of the neural network are 400 and 200, respectively. y_i^{*p} and y_i^* are used to compute the loss function designed by us. Minimization of the loss function completes the training process.

After applying the photon-number cutoff assumption, the optimization problem in Eq. (2) can be solved by applying the numerical method in Refs. [28, 38], but this is very time consuming. In this work, to reduce the time to predict secure key rates, we use the key rates obtained by the numerical method in Refs. [28, 38] as labels to train our neural network.

Neural networks for predicting the key rates. We use an artificial neural network to predict the key rates of discrete-modulated CV-QKD. The general spirit of the work is to encode the optimization problem in Eq. (2) on the loss function of a feedforward neural network and train the neural network by minimizing this loss function. The trained neural network can be seen as a mapping, which has learned the structure of the training set. For new instances, the neural network outputs the results directly via mapping, unlike traditional numerical methods that perform complex searches. As a result, the trained neural network saves a great deal of time, while ensuring a good level of accuracy. A more detailed description of neural networks can be found in Ref. [49].

A four-layer neural network model is designed to predict the key rates of discrete-modulated CV-QKD (Fig. 1). The input layer of the network has 29 neurons, which are used to receive the training inputs. The first hidden layer and the second hidden layer of the network have 400 and 200 neurons respectively, and their activation functions are the tanh function and sigmoid function, respectively. The output layer has only one neuron, which is used to predict secure key rates.

To train our neural network, we generate the data set containing $552,000$ training inputs $\{\vec{x}_i\}$ and $552,000$ corresponding labels $\{y_i\}$ using the numerical method in Refs. [28, 38]. Each $\vec{x}_i \in \{\vec{x}_i\}$ represents a vector of 29 variables, and label y_i represents the corresponding key rate. There are 16 variables in each $\vec{x}_i$ that are the right parts of the first four restrictions of Eq. (2), 12 variables in each $\vec{x}_i$ are nondiagonal elements of the right side matrix of the last restriction of Eq. (2), and the remaining variable is excess noise ξ. The 29 variables in each $\vec{x}_i$ can be calculated in the experiment by using experimental parameters and experimental observations. In our simulation, these random training inputs $\{\vec{x}_i\}$ are generated directly from seven experimental parameters (transmission distance L, light intensity μ, excess noise ξ, and probability $p0$, $p1$, $p2$ and $p3$) and the following method.

When the excess noise ξ is within $0.002 - 0.014$, we first generate a two-dimensional grid with excess noise and distance in the horizontal and vertical coordinates, respectively. Specifically, the value of the distance is between 0 and 100 km in a step of 5 km. The value of the excess noise is between 0.002 and 0.014 in a step of 0.001. Then, each grid point is sampled 80 times. With each sampling, the excess noise fluctuates around the exact value, and the float range is 0.0005 up and down. Once the excess noise for this sampling is determined, the light intensity will take a

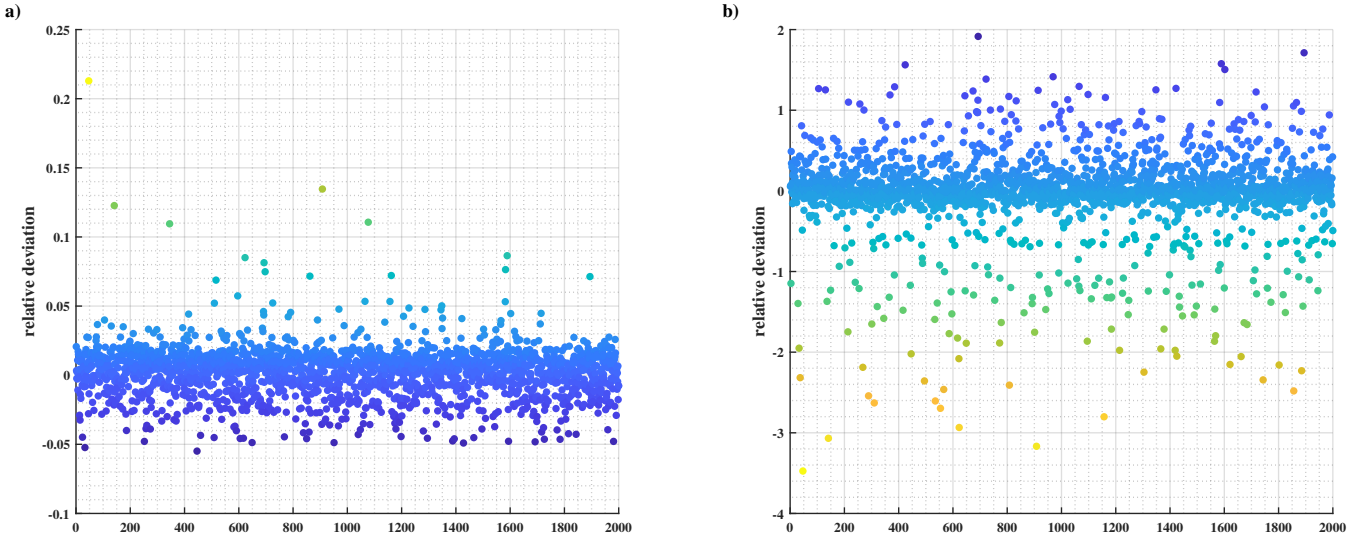

FIG. 2. Relative deviations before and after data preprocessing. We use the network structure shown in Fig. 1 with the mean square error as the loss function to compare the results of data preprocessing (a) and without data preprocessing (b). The data set is generated under the excess noise of $0.002 - 0.005$, and is split into a training set containing 158000 samples and a test set containing 2000 samples. The horizontal coordinate represents the different samples in the test set. The vertical coordinate represents the relative deviations between the key rate predicted by our neural network and the key rate obtained by the numerical method at each sample.

value every 0.01 between 0.35 and 0.60. Each sampling needs to generate 25 training inputs with a positive key rate; otherwise, the current round of sampling is discarded and restarted. In this way, 2000 training inputs are generated on each grid point. Correspondingly, a total of $520,000$ training inputs are generated on this two-dimensional grid. When the excess noise ξ is 0.015, a similar two-dimensional grid is generated. However, we only sample to 80 km, so only $32,000$ training inputs are generated. In this way, we collect a total of $552,000$ samples to train a neural network with excess noise ξ between 0.002 and 0.015. Using the numerical approach in Refs. [28, 38], we calculate the corresponding key rate for each sample as the label of the training set on the blade cluster system of the High Performance Computing Center of Nanjing University. We consume over $40,000$ core hours, and the node we used contains 4 Intel Xeon Gold 6248 CPUs, which involves immense computational power.

To improve the convergence speed and accuracy of our neural network, we preprocess the training inputs $\{\vec{x}_i\}$ and the corresponding labels $\{y_i\}$ before training the neural network. To demonstrate the necessity of the data preprocessing, we use the network structure shown in Fig. 1 to perform a controlled experiment with the mean square error as the loss function. With the excess noise of $0.002-0.005$, the absolute values of the relative deviations between the key rates predicted by our neural network and the corresponding key rates obtained by the numerical method do not exceed 25% after the data preprocessing (Fig. 2), whereas the absolute values of the relative deviations exceed 400% without the data preprocessing. Here, the relative deviation is the absolute deviation between the predicted value and true value divided by the true value. A detailed description of the data preprocessing can be found in Appendix B.

A new loss function is specifically designed to make key rates predicted by our neural network as information-theoretically secure as possible, rather than using the traditional mean squared error as a loss function. The expression of the loss function is as follows:

$$C = \frac{1}{n}\sum_{i=1}^{n} \gamma\left(e_i^{*2} + \max\left(e_i^*, -\log_{10}(\varepsilon)\right)\right) - (1-\gamma)\left(\min\left(e_i^*, 0\right)\right) \tag{3}$$

, where n is the number of training inputs. $e_i^* = y_i^{*p} - y_i^*$ is the residual error between the preprocessed label y_i^* and the corresponding output y_i^{*p} of the neural network.

The minimum function part in Eq. (3) is the penalty term and is used to make the key rates predicted by the neural network as information-theoretically secure as possible. On the other hand, the part consisting of the maximum function and the squared term in Eq. (3) is used to bound the upper limit of e_i^* to obtain higher key rates. The

parameter γ is used to balance the effects of the two parts. With the help of this loss function, we expect that the predicted value and true value can be bound in $(\varepsilon - 1, 0)$ after choosing the proper ε and γ.

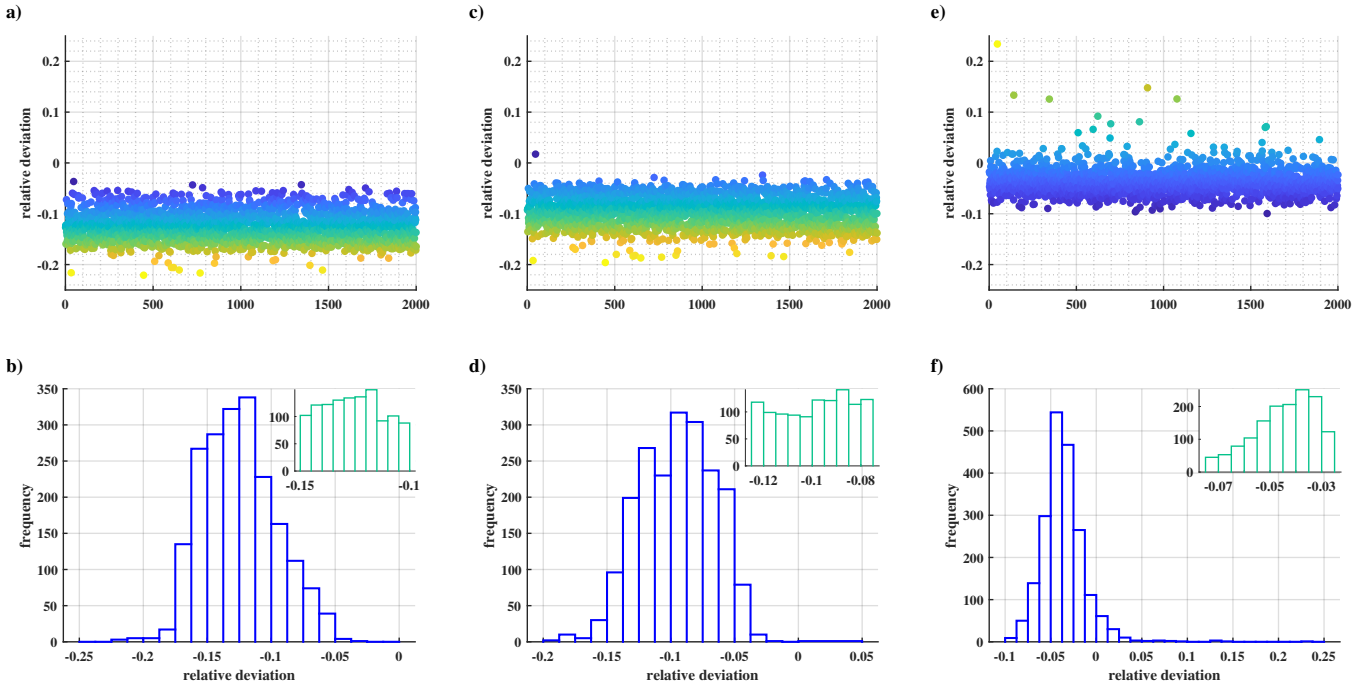

FIG. 3. Performance comparison of neural networks with different hyperparameters. (a) shows the results of the neural network with the hyperparameters $\gamma = 0.20$ and $\epsilon = 0.80$ in predicting $2,000$ samples with excess noise between 0.002 and 0.005 in the test set. The predicted key rates are strictly below the key rates obtained by the numerical method in Refs. [28, 38]. (b) plots the histogram of the relative deviation distribution in (a). The absolute value of the relative deviations remains roughly in the region of $5\% - 20\%$. (c-f) plot the corresponding results for the hyperparameters $\gamma = 0.20$, $\epsilon = 0.90$ and $\gamma = 0.80$, $\epsilon = 0.80$, respectively.

The performance of the neural networks is related to hyperparameters γ and ϵ. Without loss of generality, we take the examples of neural networks with excess noise ξ between 0.002 and 0.005 (Fig. 3). When $\gamma = 0.20$ and $\epsilon = 0.80$, the key rates predicted by the neural network are strictly lower than those obtained by the numerical method in Refs. [28, 38], which means that the key rates predicted by the neural network are information-theoretically secure. Meanwhile, the absolute values of the relative deviations are mainly distributed between 0.05 and 0.20 (Fig. 3a-b). Fig. 3c-f plot the corresponding results for the hyperparameters $\gamma = 0.20$, $\epsilon = 0.90$ and $\gamma = 0.80$, $\epsilon = 0.80$, respectively. Note that the partial key rates predicted by the neural networks under $\gamma = 0.20$, $\epsilon = 0.90$ and $\gamma = 0.80$, $\epsilon = 0.80$ are higher than the key rates obtained by the numerical method. This indicates that the performance of neural networks trained with hyperparameters $\gamma = 0.20$, $\epsilon = 0.90$ and $\gamma = 0.80$, $\epsilon = 0.80$ is not as good as that of neural network trained with hyperparameters $\gamma = 0.20$ and $\epsilon = 0.80$. Therefore, we need to carefully tune hyperparameters of the neural networks to ensure their stable performance.

The $552,000$ data generated by the numerical method are split into a training set containing $524,400$ data and a test set containing $27,600$ data. The test set is sampled from the original data set and covers instances generated under all combinations of excess noise and distance. The data preprocessing procedure follows data splitting. The Adam optimization algorithm [50] is used to train our neural network. The initial learning rate is set to 0.001. For each training, we set 200 epochs and 256 batch sizes. In addition, techniques such as early stopping and dropout [51] are used to prevent overfitting. The relative deviations of the trained network on the test set and the training set have similar distributions, which indicates that the model has good generalization performance.

Key rate comparison. We use our neural network to predict, given the optimal light intensity, key rates of discrete-modulated CV-QKD at different distances and different excess noises after training the neural network under $\gamma = 0.20$ and $\epsilon = 0.80$ according to the method described in Section III above. As shown in Fig. 4, we compare the key rates with the corresponding key rates obtained by the numerical method in Refs. [28, 38]. The results show that all key rates predicted by the neural network are strictly lower than those obtained by the numerical method. It is worth

noting that the relative deviations between them are basically within 20% (relevant data can be found in Appendix C).

To illustrate the more general case, we test the test set containing $27,600$ samples mentioned at the end of Section III. The results show that the number of samples, for which the key rates predicted by the neural network are lower than the corresponding results calculated by the numerical method, is 27379. Namely, the probability that the key rate predicted by the neural network on the test set is secure is as high as 99.2%.

Our neural network shows greater advantages over the numerical method in terms of time and resource consumption. We compare the time required to predict the key rates with our neural network and the time required to calculate the key rates with the numerical method on a high-performance personal computer with a 3.3 GHz AMD Ryzen 9 4900H and 16 GB of RAM (Fig. 5). The neural network is $6-8$ orders of magnitude of the numerical method for predicting the key rates of the discrete-modulated CV-QKD within $0-100$ km for excess noise $\xi = 0.008-0.012$. In addition, as the excess noise increases, the speed of the neural network increases even more. Refer to Appendix C for more detailed data.

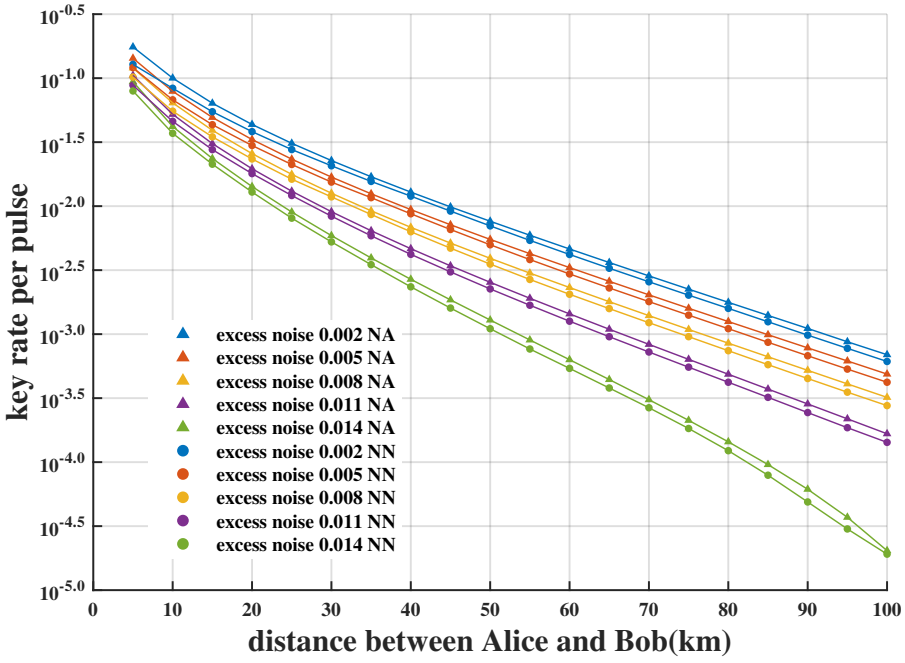

FIG. 4. Secure key rate versus the transmission distance for homodyne detection discrete-modulated CV-QKD with excess noise ξ of 0.002, 0.004, 0.008, 0.011 and 0.014 using our neural network (circles) and the numerical method in Refs. [28, 38] (triangles). The light intensity is chosen to be optimal in the interval $[0.35, 0.6]$. Tht transmission efficiency $\eta = 10^{-0.02L}$. The reconciliation efficiency $\beta = 0.95$. The neural network used for comparison is trained by setting the hyperparameters $\gamma = 0.20$ and $\epsilon = 0.80$. The cutoff photon number in the numerical method is set as 10.

DISCUSSION

We have constructed neural networks and shown that these neural networks can predict the information-theoretically secure key rates of homodyne detection discrete-modulated CV-QKD with a great probability (up to 99.2%) at a distance of $0-100$ km and an excess noise of no more than 0.015. In particular, with excess noise up to 0.008 or more, the speed of our method is at least improved by six orders of magnitude compared to that of the numerical method in Refs. [28, 38]. For example, it takes an average of 190 seconds to numerically calculate the point with the excess noise ξ around 0.008, which greatly affects the efficiency of QKD systems to calculate the secure key rate. In contrast, a neural network can calculate tens of thousands of key rates in one second. Considering that it takes a certain amount of time for the QKD system to collect data, the speed of predicting the key rates by the neural network completely meets practical applications. This advantage brings us one step closer to achieving low latency for discrete modulated CV-QKD on a low-power platform.

Recently, there have been two main types of situations in which machine learning is used in QKD. One is used for experimental parameter optimization [52, 53] and the other is used to assist experimental control [54–56]. They all use machine learning to replace traditional optimization or feedback control algorithms, which are significantly different from our work. To the best of our knowledge, this is the first time we have tried to apply machine learning methods to predict key rates of QKD. This poses a greater challenge than parameter optimization with machine learning methods. This is because the parameters predicted by the neural networks are substituted into numerical or analytical methods to find the corresponding key rates, which naturally ensures that the key rates are information-theoretically secure. However, the key rates obtained by neural networks do not guarantee this naturally, which forces us to redesign the loss function and seek better data preprocessing methods to guarantee the acquired key rate with information-theoretic security.

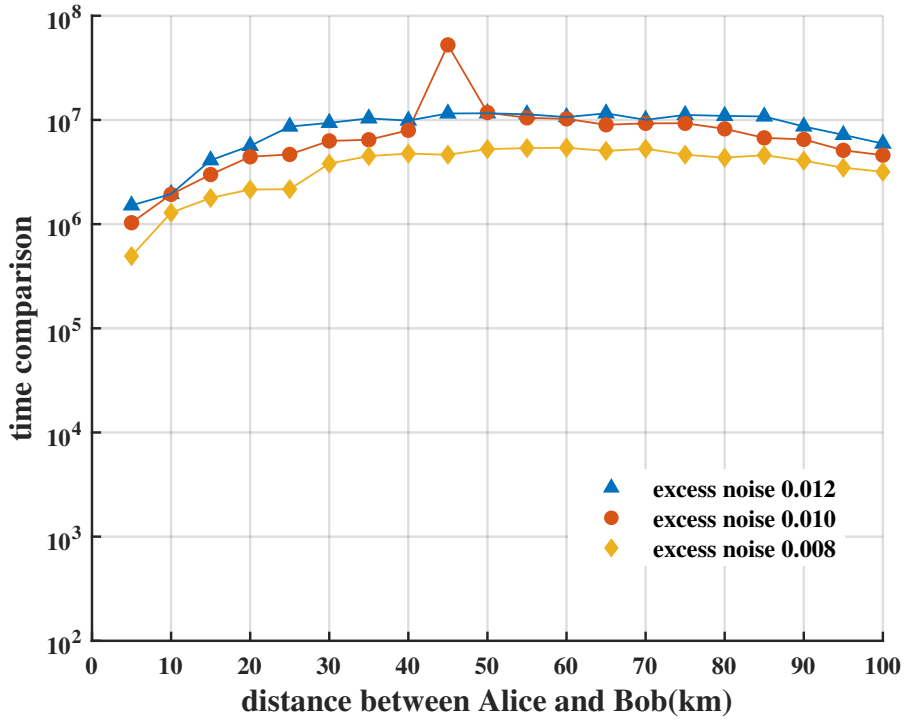

FIG. 5. Time consumption comparison between the neural network method and numerical method. The comparison results with excess noise of 0.008, 0.010 and 0.012 are shown as diamonds, circles and triangles, respectively. Each point represents the logarithm of the ratio of the running time of the numerical method divided by the running time of the neural network method. The neural network used for comparison is trained by setting the hyperparameters $\gamma = 0.20$ and $\epsilon = 0.80$. The cutoff photon number in the numerical method is set as 10.

We expect that larger excess noises and longer distances will require a deeper network, more sophisticated loss functions, and more detailed data preprocessing methods to improve the performance of neural networks on the training set. More training data are also necessary to improve the generalization ability of the neural networks. For deep neural networks, the rapid growth or rapid disappearance of the transmitted gradient hinders the optimization process; therefore, the debugging process is highly technical. The debugging process can be guided by monitoring the activation function values of the neurons and histograms 1 of those gradients [49].

Our machine learning approach is at least six orders of magnitude of the numerical method at predicting the secure key rates of homodyne detection discrete-modulated CV-QKD with excess noise up to 0.008 or more. However, training our neural network is still time consuming. This is because we need to use traditional numerical methods to obtain a number of key rates as the training set of the neural networks. In particular, the performance of our neural network is dependent on the choice of hyperparameters γ, ϵ and initial learning rate. This means that we may need to train several times to obtain a suitable neural network. To make our machine learning method more intelligent, further work is necessary to design another neural network to automatically find the most suitable hyperparameters. We have also tried other machine learning methods, such as boosting decision trees. These methods have smaller relative deviations, but have greater variances. We have left the fusion of these methods to future research.

The important contribution of our work is that it opens the door to using classical machine learning to predict QKD key rates. In particular, our ideas and methods are very easy to generalize to other QKD protocols. We expect that our work will stimulate further research to help most QKD systems run on low-power chips in mobile devices.

METHODS

Discrete-modulated CV-QKD. According to Ref. [38], homodyne detection discrete-modulated CV-QKD is described below:

(1) State preparation.—Alice prepares a coherent state $|\psi_k\rangle$ from the set $\{|\alpha\rangle, |-\alpha\rangle, |i\alpha\rangle, |-i\alpha\rangle\}$ according to the probability of $[p_A/2, p_A/2, (1-p_A)/2, (1-p_A)/2]$, where $\alpha \in R$ is a predetermined amplitude and k is the number of rounds. Then Alice sends the state $|\psi_k\rangle$ to Bob.

(2) Measurement.—Bob performs a homodyne measurement on the received state. He chooses to measure a certain orthogonal component (q or p) according to the probability of $[p_B, 1-p_B]$. If q is chosen, Bob notes $b_k = 0$, otherwise he notes $b_k = 1$. Then, Bob records his measurement outcome $y_k \in R$.

(3) Announcement and sifting.—After repeating the first two steps N times, Alice and Bob communicate via the classical authentication channel and divide the obtained data into the following four subsets:

$$\begin{aligned}
\mathcal{I}_{qq} &= \{k \in [N] : |\psi_k\rangle \in \{|\alpha\rangle, |-\alpha\rangle\}, b_k = 0\}, \\
\mathcal{I}_{qp} &= \{k \in [N] : |\psi_k\rangle \in \{|\alpha\rangle, |-\alpha\rangle\}, b_k = 1\}, \\
\mathcal{I}_{pq} &= \{k \in [N] : |\psi_k\rangle \in \{|i\alpha\rangle, |-i\alpha\rangle\}, b_k = 0\}, \\
\mathcal{I}_{pp} &= \{k \in [N] : |\psi_k\rangle \in \{|i\alpha\rangle, |-i\alpha\rangle\}, b_k = 1\},
\end{aligned} \tag{4}$$

where $[N]$ denotes the set of all integers from 1 to N. Then Alice and Bob randomly select a subset $\mathcal{I}_{\text{key}}$ of size m from $\mathcal{I}_{qq}$ for generating keys. The key string $\mathbf{X} = (x_1, x_2, \ldots, x_m)$ at Alice is also determined according to the following rules:

$$\forall j \in [m], \quad x_j = \begin{cases} 0 & \text{if } \left|\psi_{f(j)}\right\rangle = |\alpha\rangle, \\ 1 & \text{if } \left|\psi_{f(j)}\right\rangle = |-\alpha\rangle, \end{cases} \tag{5}$$

where $f(j)$ is a function that maps from $\mathcal{I}_{\text{key}}$ to $\mathcal{I}_{qq}$. The remaining data in $\mathcal{I}_{qq}$, $\mathcal{I}_{qp}$, $\mathcal{I}_{pq}$ and $\mathcal{I}_{pp}$ are integrated into the set $\mathcal{I}_{\text{test}}$ and used for parameter estimation.

(4) Parameter estimation.—Alice and Bob perform parameter estimation based on the data in $\mathcal{I}_{\text{test}}$. First, they calculate the first and second moments of q and p quadratures for each of the four coherent states sent by Alice. Then they calculate the secret key rate based on the convex optimization problem in Eq. (8).

If the result shows that the key rate is equal to 0, Alice and Bob abort the protocol and start over. Otherwise, they continue with the next step.

(5) Reverse reconciliation key map.—The key string $\mathbf{Z} = (z_1, z_2, \ldots, z_m)$ at Bob is determined according to Bob's measurement outcome y_k in step 2 and the following rules:

$$z_j = \begin{cases} 0 & \text{if } y_{f(j)} \in [\Delta_c, \infty), \\ 1 & \text{if } y_{f(j)} \in (-\infty, -\Delta_c], \\ \perp & \text{if } y_{f(j)} \in (-\Delta_c, \Delta_c), \end{cases} \tag{6}$$

where $\Delta_c \geq 0$ is determined by the postselection of data.

Alice and Bob then pick out the location of the symbol $\perp$ and remove the data at that location by classical communication. The set $\mathbf{X}$ and $\mathbf{Z}$ after removing $\perp$ is the raw key string.

(6) Error correction and privacy amplification.—Alice and Bob choose a suitable error-correction protocol and a suitable privacy-amplification protocol to generate secret key rates.

The key rate can be calculated using the well-known Devetak-Winter formula [57] in the asymptotic limit and under collective attacks. To apply this formula, we transform the prepare-and-measure protocol into the entanglement-based protocol.

Alice prepares the state according to the ensemble $\{|\varphi_x\rangle, p_x\}$ in the prepare-and-measure protocol. In the equivalent entanglement-based protocol, Alice prepares the bipartite state in the form of $|\Psi\rangle_{AA'} = \sum_x \sqrt{p_x}|x\rangle_A |\varphi_x\rangle_{A'}$. Here

Alice keeps $|x\rangle_A$ in register A and sends $|\varphi_x\rangle_{A'}$ to Bob. $|\varphi_x\rangle_{A'}$ changes as it passes through an insecure quantum channel. The process can be described by a completely positive and trace-preserving map $\mathcal{E}_{A'\to B}$. The bipartite state ρ_{AB} thus transforms into

$$\rho_{AB} = (\mathrm{id}_A \otimes \mathcal{E}_{A'\to B}) (|\Psi\rangle \langle \Psi|_{AA'}) , \tag{7}$$

where id_A is the identity transformation acting on A. Under reverse reconciliation [58], the key rate formula can be expressed according to Refs. [27, 28] as

$$R^{\infty} = \min_{\rho_{AB}\in\mathbf{S}} D\left(\mathcal{G}\left(\rho_{AB}\right) \| \mathcal{Z}\left[\mathcal{G}\left(\rho_{AB}\right)\right]\right) - p_{\mathrm{pass}}\delta_{\mathrm{EC}}. \tag{8}$$

Algorithm 1: Training stage

Input: $\{(\vec{x}_i, y_i)\}$ // Original training data set of discrete-modulated CV-QKD collected from the numerical method. $\vec{x_i}$ is feature vector containing 29 variables, and y_i is the corresponding key rate.
Input: γ, ϵ // Two hyperparameters in our self-designed loss function.
Output: $\{\theta_r\}$ // The final learned weights of the neural network $\aleph_r$

Preprocessing $\{\vec{x}_i^*\} \leftarrow \{\vec{x}_i\}$:
  Calculate the mean vector $\vec{x}$ of $\{\vec{x}_i\}$
  Calculate the variance vector $\vec{\sigma}$ of $\{\vec{x}_i\}$
if $\sigma_j == 0$ **then**
  $x_{ij}^* = x_{ij}$
else
  $x_{ij}^* = (x_{ij} - \mu_j)/\sigma_j$
Preprocessing $\{y_i^*\} \leftarrow \{y_i\}$:
$y_i^* = -\log_{10}(y_i)$
Train the neural network under$\{\gamma, \epsilon\}$ with $\{(\vec{x}_i^*, y_i^*)\}$
return $\{\theta_r\}$

Algorithm 2: Inference stage

Input: $\{\vec{x}_i\}$ // A set of original feature vectors containing 29 variables collected from the experiment.
Output: $\{y_i\}$ // A set of corresponding key rates predicted by Neural network $\aleph_r$.

Preprocessing $\{\vec{x}_i^*\} \leftarrow \{\vec{x}_i\}$
for $\vec{x}_i^* \in \{\vec{x}_i^*\}$ **do**
  $y_i^* \leftarrow \aleph_r(\vec{x_i^*})$
  $y_i = 10^{-y_i^*}$
end
return $\{y_i\}$

Details of data preprocessing. To improve the performance of our neural network, we preprocess the training inputs $\{\vec{x}_i\}$ before training the neural network. The process can be expressed as

$$x_{ij}^* = \frac{x_{ij} - \bar{x}_j}{\sigma_j}, \tag{9}$$

where x_{ij} represents the j-th component of the i-th sample; $\bar{x}_j$ and σ_j are the mean and variance of the j-th component in all samples, respectively; x_{ij}^* is the j-th component of the i-th sample after being preprocessed.

The preprocessed data $\{\vec{x}_i^*\}$ follow a standard normal distribution with a mean of 0 and a variance of 1. The process removes dimensional restrictions and facilitates the comparison of features of different dimensions. Since the

TABLE I. Relative deviations between key rates predicted by our neural network and the corresponding key rates obtained by the numerical method for the given optimal light intensity at different distances and different excess noises.

L	Relative deviations				
(km)	$\xi = 0.002$	$\xi = 0.005$	$\xi = 0.008$	$\xi = 0.011$	$\xi = 0.014$
5	0.27	0.16	0.15	0.16	0.15
10	0.17	0.14	0.14	0.12	0.12
15	0.14	0.12	0.11	0.10	0.10
20	0.12	0.10	0.09	0.08	0.09
25	0.11	0.09	0.08	0.07	0.10
30	0.09	0.09	0.06	0.08	0.11
35	0.08	0.07	0.06	0.09	0.11
40	0.07	0.07	0.08	0.10	0.13
45	0.07	0.08	0.09	0.10	0.14
50	0.08	0.09	0.10	0.11	0.14
55	0.09	0.10	0.11	0.12	0.15
60	0.09	0.11	0.11	0.12	0.14
65	0.10	0.11	0.12	0.13	0.14
70	0.10	0.11	0.12	0.13	0.13
75	0.10	0.12	0.12	0.13	0.14
80	0.10	0.12	0.13	0.13	0.15
85	0.11	0.13	0.13	0.14	0.17
90	0.11	0.13	0.14	0.14	0.20
95	0.11	0.14	0.14	0.15	0.19
100	0.11	0.14	0.14	0.14	0.06

maximum difference between different key rates in these samples is 4 orders of magnitude, we preprocess the labels as follows to speed up the training process of the neural networks:

$$y_i^* = -\log_{10}(y_i), \tag{10}$$

where y_i^* is the label corresponding to the i-th sample after being preprocessed. Note that the outputs predicted by the neural networks trained with preprocessed labels $\{y_i^*\}$ need to be inverse solved using the following equation:

$$y_i^p = 10^{-y_i^{*p}}, \tag{11}$$

where y_i^{*p} and y_i^p are the output value and the predicted key rate of the neural networks for the i-th sample, respectively.

Algorithms 1 and 2 show the detailed training process of the neural networks and the process of using trained neural networks to predict new samples, respectively.

Detailed data. Table I shows the relative deviations between the key rates predicted by our neural network and the corresponding key rates obtained by the numerical method for the given optimal light intensity at different distances and different excess noises. This table is a supplement to Fig. 4.

Table II shows the specific data of the time consumption of the neural network and the numerical method with excess noise ξ of 0.008, 0.010 and 0.012. In the numerical method, each point with excess noise ξ of approximately 0.01 takes 200 seconds on average, which greatly affects the efficiency of the QKD system to calculate the secure key rate. In contrast, the neural network can calculate tens of thousands of key rates in one second. Considering that it takes a certain amount of time for the QKD system to collect data, the speed of predicting the key rates by the neural network completely meets practical applications.

Acknowledgements
We gratefully acknowledge the support from the Natural Science Foundation of Jiangsu Province (No. BK20211145), the Fundamental Research Funds for the Central Universities (No. 020414380182), the Key Research and Development Program of Nanjing Jiangbei New Aera (No. ZDYD20210101), the Key-Area Research and Development Program of Guangdong Province (No. 2020B0303040001). We are grateful to the High Performance Computing Center of Nanjing University for performing the numerical calculations in this paper on its blade cluster system.

TABLE II. Time consumption of the neural network versus the numerical method with excess noise ξ of 0.008, 0.010 and 0.012. NM and NN are the abbreviations of the numerical method and neural network, respectively. L is the distance between Alice and Bob.

$\xi = 0.008$			$\xi = 0.010$			$\xi = 0.012$		
L(km)	NM(s)	NN(s)	L(km)	NM(s)	NN(s)	L(km)	NM(s)	NN(s)
5	1.42×10^2	1.98×10^{-4}	5	1.54×10^2	3.28×10^{-4}	5	2.16×10^2	1.31×10^{-4}
10	7.86×10^1	7.25×10^{-5}	10	9.94×10^1	5.85×10^{-5}	10	1.27×10^2	4.70×10^{-5}
15	1.04×10^2	6.60×10^{-5}	15	1.72×10^2	5.70×10^{-5}	15	2.24×10^2	4.15×10^{-5}
20	1.09×10^2	6.50×10^{-5}	20	2.37×10^2	5.40×10^{-5}	20	3.07×10^2	4.30×10^{-5}
25	1.20×10^2	6.65×10^{-5}	25	2.45×10^2	6.30×10^{-5}	25	4.40×10^2	4.25×10^{-5}
30	1.98×10^2	5.65×10^{-5}	30	3.30×10^2	4.75×10^{-5}	30	4.92×10^2	4.20×10^{-5}
35	2.34×10^2	5.90×10^{-5}	35	3.71×10^2	5.90×10^{-5}	35	5.33×10^2	4.65×10^{-5}
40	2.47×10^2	5.70×10^{-5}	40	4.18×10^2	5.85×10^{-5}	40	5.72×10^2	4.60×10^{-5}
45	2.50×10^2	6.10×10^{-5}	45	2.73×10^2	5.70×10^{-5}	45	5.94×10^2	4.35×10^{-5}
50	2.62×10^2	6.35×10^{-5}	50	6.24×10^2	5.60×10^{-5}	50	5.79×10^2	4.55×10^{-5}
55	2.74×10^2	6.50×10^{-5}	55	5.55×10^2	5.10×10^{-5}	55	5.83×10^2	4.30×10^{-5}
60	2.68×10^2	6.65×10^{-5}	60	5.28×10^2	5.85×10^{-5}	60	5.96×10^2	4.30×10^{-5}
65	2.55×10^2	6.70×10^{-5}	65	5.48×10^2	5.10×10^{-5}	65	5.96×10^2	4.20×10^{-5}
70	2.72×10^2	6.55×10^{-5}	70	4.82×10^2	5.65×10^{-5}	70	5.91×10^2	5.30×10^{-5}
75	2.60×10^2	6.70×10^{-5}	75	4.78×10^2	6.70×10^{-5}	75	5.87×10^2	4.10×10^{-5}
80	2.30×10^2	6.00×10^{-5}	80	4.19×10^2	5.20×10^{-5}	80	5.57×10^2	4.35×10^{-5}
85	2.34×10^2	5.70×10^{-5}	85	3.63×10^2	5.95×10^{-5}	85	5.45×10^2	4.35×10^{-5}
90	1.99×10^2	5.75×10^{-5}	90	3.48×10^2	5.35×10^{-5}	90	4.37×10^2	4.10×10^{-5}
95	1.72×10^2	5.75×10^{-5}	95	2.92×10^2	5.10×10^{-5}	95	3.81×10^2	4.35×10^{-5}
100	1.54×10^2	5.85×10^{-5}	100	2.43×10^2	6.60×10^{-5}	100	3.47×10^2	4.65×10^{-5}

Author contributions

H.-L.Y. and Z.-B.C. conceived the research. M.-G.Z., Z.-P.L. and H.-L.Y. devised the neural network architecture and carried out the numerical simulations. M.-G.Z., Z.-P.L., W.-B.L., C.-L.L., J.-L.B., Y.-R.X, Y.F. and H.-L.Y. developed the theory and calculated the secure key rate. All authors discussed the results and prepared the manuscript. M.-G.Z. and Z.-P.L. contributed equally to this work

Competing interests

The authors declare no competing interests.

Correspondence and requests for materials should be addressed to H.-L.Y. and Z.-B.C.

* hlyin@nju.edu.cn
† zbchen@nju.edu.cn

[1] Lloyd, S., Mohseni, M. & Rebentrost, P. Quantum principal component analysis. *Nat. Phys* **10**, 631–633 (2014).
[2] Ciliberto, C. *et al.* Quantum machine learning: a classical perspective. *Proc. R. Soc. A* **474**, 20170551 (2018).
[3] Beer, K. *et al.* Training deep quantum neural networks. *Nat. Commun.* **11**, 808 (2020).
[4] Bondarenko, D. & Feldmann, P. Quantum autoencoders to denoise quantum data. *Phys. Rev. Lett.* **124**, 130502 (2020).
[5] Farhi, E. & Neven, H. Classification with quantum neural networks on near term processors. *arXiv preprint arXiv:1802.06002* (2018).
[6] Mitarai, K., Negoro, M., Kitagawa, M. & Fujii, K. Quantum circuit learning. *Phys. Rev. A* **98**, 032309 (2018).
[7] Wan, K. H., Dahlsten, O., Kristjánsson, H., Gardner, R. & Kim, M. Quantum generalisation of feedforward neural networks. *npj Quantum Inf.* **3**, 36 (2017).
[8] Chen, Z.-B. Quantum neural network and soft quantum computing. *arXiv preprint arXiv:1810.05025* (2018).
[9] Jerbi, S., Trenkwalder, L. M., Nautrup, H. P., Briegel, H. J. & Dunjko, V. Quantum enhancements for deep reinforcement learning in large spaces. *PRX Quantum* **2**, 010328 (2021).
[10] Abbas, A. *et al.* The power of quantum neural networks. *Nat Comput Sci* **1**, 403–409 (2021).
[11] Torlai, G. *et al.* Neural-network quantum state tomography. *Nat. Phys.* **14**, 447–450 (2018).
[12] Smith, A. W., Gray, J. & Kim, M. Efficient quantum state sample tomography with basis-dependent neural networks. *PRX Quantum* **2**, 020348 (2021).
[13] Quek, Y., Fort, S. & Ng, H. K. Adaptive quantum state tomography with neural networks. *npj Quantum Inf.* **7**, 105 (2021).

[14] Gao, J. *et al.* Experimental machine learning of quantum states. *Phys. Rev. Lett.* **120**, 240501 (2018).

[15] Ma, Y.-C. & Yung, M.-H. Transforming Bell's inequalities into state classifiers with machine learning. *npj Quantum Inf.* **4**, 34 (2018).

[16] Yang, M. *et al.* Experimental simultaneous learning of multiple nonclassical correlations. *Phys. Rev. Lett.* **123**, 190401 (2019).

[17] Hentschel, A. & Sanders, B. C. Efficient algorithm for optimizing adaptive quantum metrology processes. *Phys. Rev. Lett.* **107**, 233601 (2011).

[18] Fiderer, L. J., Schuff, J. & Braun, D. Neural-network heuristics for adaptive bayesian quantum estimation. *PRX Quantum* **2**, 020303 (2021).

[19] Cimini, V. *et al.* Calibration of multiparameter sensors via machine learning at the single-photon level. *Phys. Rev. Applied* **15**, 044003 (2021).

[20] Bukov, M. *et al.* Reinforcement learning in different phases of quantum control. *Phys. Rev. X* **8**, 031086 (2018).

[21] Wise, D. F., Morton, J. J. & Dhomkar, S. Using deep learning to understand and mitigate the qubit noise environment. *PRX Quantum* **2**, 010316 (2021).

[22] Coyle, B., Doosti, M., Kashefi, E. & Kumar, N. Variational quantum cloning: Improving practicality for quantum cryptanalysis. *arXiv preprint arXiv:2012.11424* (2020).

[23] Bennett, C. H. & Brassard, G. Quantum cryptography: public key distribution and coin tossing int. In *Conf. on Computers, Systems and Signal Processing (Bangalore, India*, vol. 175 (1984).

[24] Ekert, A. K. Quantum cryptography based on Bell's theorem. *Phys. Rev. Lett.* **67**, 661 (1991).

[25] Xu, F., Ma, X., Zhang, Q., Lo, H.-K. & Pan, J.-W. Secure quantum key distribution with realistic devices. *Rev. Mod. Phys.* **92**, 025002 (2020).

[26] Matsuura, T., Maeda, K., Sasaki, T. & Koashi, M. Finite-size security of continuous-variable quantum key distribution with digital signal processing. *Nat. Commun.* **12**, 1–13 (2021).

[27] Coles, P. J., Metodiev, E. M. & Lütkenhaus, N. Numerical approach for unstructured quantum key distribution. *Nat. Commun.* **7**, 11712 (2016).

[28] Winick, A., Lütkenhaus, N. & Coles, P. J. Reliable numerical key rates for quantum key distribution. *Quantum* **2**, 77 (2018).

[29] Primaatmaja, I. W., Lavie, E., Goh, K. T., Wang, C. & Lim, C. C. W. Versatile security analysis of measurement-device-independent quantum key distribution. *Phys. Rev. A* **99**, 062332 (2019).

[30] Tan, E. Y.-Z., Schwonnek, R., Goh, K. T., Primaatmaja, I. W. & Lim, C. C.-W. Computing secure key rates for quantum cryptography with untrusted devices. *npj Quantum Information* **7**, 1–6 (2021).

[31] Pirandola, S. *et al.* Advances in quantum cryptography. *Advances in Optics and Photonics* **12**, 1012–1236 (2020).

[32] Grosshans, F. & Grangier, P. Continuous variable quantum cryptography using coherent states. *Phys. Rev. Lett.* **88**, 057902 (2002).

[33] Weedbrook, C. *et al.* Quantum cryptography without switching. *Phys. Rev. Lett.* **93**, 170504 (2004).

[34] Zhao, Y.-B., Heid, M., Rigas, J. & Lütkenhaus, N. Asymptotic security of binary modulated continuous-variable quantum key distribution under collective attacks. *Phys. Rev. A* **79**, 012307 (2009).

[35] Leverrier, A. & Grangier, P. Unconditional security proof of long-distance continuous-variable quantum key distribution with discrete modulation. *Phys. Rev. Lett.* **102**, 180504 (2009).

[36] Hirano, T. *et al.* Implementation of continuous-variable quantum key distribution with discrete modulation. *Quantum Sci. Tech.* **2**, 024010 (2017).

[37] Ghorai, S., Grangier, P., Diamanti, E. & Leverrier, A. Asymptotic security of continuous-variable quantum key distribution with a discrete modulation. *Phys. Rev. X* **9**, 021059 (2019).

[38] Lin, J., Upadhyaya, T. & Lütkenhaus, N. Asymptotic security analysis of discrete-modulated continuous-variable quantum key distribution. *Phys. Rev. X* **9**, 041064 (2019).

[39] Lin, J. & Lütkenhaus, N. Trusted detector noise analysis for discrete modulation schemes of continuous-variable quantum key distribution. *Phys. Rev. Applied* **14**, 064030 (2020).

[40] Liu, W.-B. *et al.* Homodyne detection quadrature phase shift keying continuous-variable quantum key distribution with high excess noise tolerance. *PRX Quantum* **2**, 040334 (2021).

[41] Upadhyaya, T., van Himbeeck, T., Lin, J. & Lütkenhaus, N. Dimension reduction in quantum key distribution for continuous-and discrete-variable protocols. *PRX Quantum* **2**, 020325 (2021).

[42] Kanitschar, F. & Pacher, C. Tight secure key rates for cv-qkd with 8psk modulation. *arXiv preprint arXiv:2107.06110* (2021).

[43] Kaur, E., Guha, S. & Wilde, M. M. Asymptotic security of discrete-modulation protocols for continuous-variable quantum key distribution. *Phys. Rev. A* **103**, 012412 (2021).

[44] Denys, A., Brown, P. & Leverrier, A. Explicit asymptotic secret key rate of continuous-variable quantum key distribution with an arbitrary modulation. *Quantum* **5**, 540 (2021).

[45] Hu, H., Im, J., Lin, J., Lütkenhaus, N. & Wolkowicz, H. Robust interior point method for quantum key distribution rate computation. *arXiv preprint arXiv:2104.03847* (2021).

[46] Bunandar, D., Govia, L. C., Krovi, H. & Englund, D. Numerical finite-key analysis of quantum key distribution. *npj Quantum Inf.* **6**, 104 (2020).

[47] George, I., Lin, J. & Lütkenhaus, N. Numerical calculations of the finite key rate for general quantum key distribution protocols. *Phys. Rev. Research* **3**, 013274 (2021).

[48] Abadi, M. *et al.* TensorFlow: Large-scale machine learning on heterogeneous systems (2015). URL `https://www.tensorflow.org/`. Software available from tensorflow.org.
[49] Goodfellow, I., Bengio, Y. & Courville, A. *Deep learning* (MIT Press, Cambridge, Mass., 2016).
[50] Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
[51] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**, 1929–1958 (2014).
[52] Lu, F.-Y. *et al.* Parameter optimization and real-time calibration of a measurement-device-independent quantum key distribution network based on a back propagation artificial neural network. *J. Opt. Soc. Am. B* **36**, B92–B98 (2019).
[53] Wang, W. & Lo, H.-K. Machine learning for optimal parameter prediction in quantum key distribution. *Phys. Rev. A* **100**, 062334 (2019).
[54] Liu, W., Huang, P., Peng, J., Fan, J. & Zeng, G. Integrating machine learning to achieve an automatic parameter prediction for practical continuous-variable quantum key distribution. *Phys. Rev. A* **97**, 022316 (2018).
[55] Liu, J.-Y., Ding, H.-J., Zhang, C.-M., Xie, S.-P. & Wang, Q. Practical phase-modulation stabilization in quantum key distribution via machine learning. *Phys. Rev. Applied* **12**, 014059 (2019).
[56] Chin, H.-M., Jain, N., Zibar, D., Andersen, U. L. & Gehring, T. Machine learning aided carrier recovery in continuous-variable quantum key distribution. *npj Quantum Inf.* **7**, 20 (2021).
[57] Devetak, I. & Winter, A. Distillation of secret key and entanglement from quantum states. *Proc. R. Soc. A* **461**, 207–235 (2005).
[58] Grosshans, F., Cerf, N. J., Wenger, J., Tualle-Brouri, R. & Grangier, P. Virtual entanglement and reconciliation protocols for quantum cryptography with continuous variables. *arXiv preprint quant-ph/0306141* (2003).