Analyzing feature space via Pauli decomposition for quantum classifier

Yudai Suzuki · Hiroshi Yano · Qi Gao · Shumpei Uno · Tomoki Tanaka · Manato Akiyama · Naoki Yamamoto

Received: date / Accepted: date

Abstract A method for analyzing the feature space used in the quantum classifier on the basis of Pauli decomposition is developed. In particular, for 2-dimensional input datasets, the method boils down to a general formula that easily computes a lower bound of the exact training accuracy, which eventually helps us to see whether the feature space constructed with the selected feature map is suitable for linearly separating the dataset. The effectiveness of this formula is demonstrated, with the special type of five feature maps and four 2-dimensional non-linear datasets.

Y. Suzuki, H. Yano: Equally contributing authors.

Yudai Suzuki
Department of Mechanical Engineering, Keio University, Hiyoshi 3-14-1, Kohoku, Yokohama 223-8522, Japan

Hiroshi Yano
Department of Information and Computer Science, Keio University, Hiyoshi 3-14-1, Kohoku, Yokohama 223-8522, Japan

Qi Gao · Shumpei Uno · Tomoki Tanaka · Naoki Yamamoto
Quantum Computing Center, Keio University, Hiyoshi 3-14-1, Kohoku, Yokohama 223-8522, Japan

Manato Akiyama
Department of Biosciences and Informatics, Keio University, Hiyoshi 3-14-1, Kohoku, Yokohama 223-8522, Japan

Qi Gao
Mitsubishi Chemical Corporation Science & Innovation Center, 1000, Kamoshida-cho, Aoba-ku, Yokohama 227-8502, Japan

Shumpei Uno
Mizuho Information & Research Institute, Inc., 2-3 Kanda-Nishikicho, Chiyoda-ku, Tokyo 101-8443, Japan

Tomoki Tanaka
Mitsubishi UFJ Financial Group, Inc. and MUFG Bank, Ltd., 2-7-1 Marunouchi, Chiyoda-ku, Tokyo 100-8388, Japan

Naoki Yamamoto
Department of Applied Physics and Physico-Informatics, Keio University, Hiyoshi 3-14-1, Kohoku, Yokohama 223-8522, Japan, E-mail: yamamoto@appi.keio.ac.jp
1 Introduction

Over the last 20 years, the unprecedented improvements in the cost-effectiveness ration of computer, together with improved computational techniques, make machining learning widely applicable in every aspect of our lives such as education, healthcare, games, finance, transportation, energy, business, science and engineering [12]. Among numerous developed machining learning methods, Support Vector Machine (SVM) is a very established one which has become an overwhelmingly popular choice for data analysis [3–7]. In SVM method, a non-linear dataset is transformed via a feature map to another dataset and is separated by a hyperplane in the feature space, which can be effectively performed using the kernel trick. However, choosing a suitable feature map is not easy. Furthermore, the computational expense of SVM significantly increases for a very large dataset.

Quantum computing is expected to speed-up the performance of machine learning through exploiting quantum mechanical properties including superposition, interference, and entanglement. Particularly as for the quantum classifier, a renaissance began in the past few years, with the quantum least-squares SVM method developed by Rebentrost et al. [8], the variational quantum circuit learning method developed by Mitarai et al. [9], and some parallel development by other research groups [10, 11]. Furthermore, recently Havlicek et al. proposed two classifiers and demonstrated some classification tasks on a real device [12]. These papers show the possibility that machine learning will get a further boost by using quantum computers in the near future.

For all the developed quantum classification methods, a function called the feature map is needed for encoding the dataset taken from its original low dimensional real space onto a high dimensional quantum state space (i.e., Hilbert space). In principle, if we can choose a suitable feature map, the quantum classifier may presumably achieve a high training accuracy due to the fact that the non-linearly separable dataset can become linearly separable dataset in the quantum space. Finding such a suitable feature map for a specific dataset is not trivial as mentioned above in the first paragraph, and there are two heuristic ways to do this task, as follows. 1) One prepares many feature map candidates and tries to find a suitable one by comparing the results of training accuracy attained with all those feature maps [13]; this means that for determining a suitable feature map, numerous times of classification or regression analysis may be necessary. 2) To reduce the number of feature map candidates, one exploits a method for roughly and systematically estimating the training accuracy of every feature map candidate, before really performing the classification or regression analysis; after that one carries out the same procedure of 1) for remaining feature map candidates. The latter is a much more efficient way, hence our goal is to establish such a convenient method that reliably and effectively determines a suitable feature map used in the quantum classifier.

In this study, we propose a method that, via the Pauli decomposition of the density matrix, transforms the quantum space generated with a selected feature map, to the set of real spaces associated with the input data. The constructed set of real spaces gives us a visualization of the original input data distribution and, further, enables us to easily calculate the minimum accuracy, i.e., a lower bound of the exact training
Analyzing feature space via Pauli decomposition for quantum classifier accuracy, without performing the actual classification procedure. The minimum accuracies calculated for the feature map candidates are then used to do the task 2), which eventually screens a library of suitable feature maps. In this work, we demonstrate the effectiveness of our method, with the special type of five feature maps and four 2-dimensional non-linear datasets, showing that our approach introduces a new dimension to design a suitable feature map.

2 Methods

2.1 Transformation of the feature space via Pauli decomposition

In the SVM method with quantum kernel estimator, proposed by Havlicek et al. [12], the feature map transforms an input dataset to multi-qubit states, which form the feature space (i.e., Hilbert space); then the kernel matrix is constructed by calculating all the inner products of quantum states, and it is finally used in the (standard) SVM for classifying the dataset. Surely different feature maps lead to different kernels and accordingly influence a lot on the classification accuracy, meaning that a careful analysis of the feature space is necessary. However, due to the complicated structure of the feature space, such analysis is in general not straightforward. Here, we propose the Pauli-decomposition-based method for constructing another feature space, which is easier to analyze and might be used as a guide to select a suitable feature map.

Here we give a brief summary of the SVM method with quantum kernel estimator, which we call simply the “quantum SVM method” in what follows. First, a classical data $x \in \mathbb{R}^n$ is encoded into the unitary operator $U_{\Phi}(x)$ through an encoding function $\Phi(x)$, and it is applied to the initial state $|0\rangle^\otimes n$ with $|0\rangle$ the qubit ground state; note that we here assume that the number of qubits is equal to the dimension of the classical data (or more precisely the number of feature variables of the data). Thus, the feature map is a transformation from the classical data $x$ to the quantum state $|\Phi(x)\rangle$, and the feature space is $(\mathbb{C}^2)^\otimes n = \mathbb{C}^{2^n}$. The kernel is then naturally defined as $K(x,z) = |\langle \Phi(x)|\Phi(z)\rangle|^2$; this quantity can be practically calculated as the ratio of zero strings $0^n$ in the Z-basis measurement result, for the final state $U_{\Phi(x)}^\dagger U_{\Phi(z)} |0\rangle^\otimes n$.

Finally, the constructed kernel is used in the standard manner in SVM; that is, a test data $x$ is classified into two categories depending on the sign of

$$
\sum_{i=1}^N \alpha_i y_i K(x_i,x) + b,
$$

where $(x_i, y_i) (i = 1, \ldots, N)$ is the pair of training data, and $(\alpha, b)$ are the optimized parameters. A possible advantage in using the quantum system for the classification problem would lie in the fact that even an intermediate-scale quantum computer with e.g. $n = 20$ has a huge dimensional feature space.

Now we introduce another feature map and space. The key idea is simply to use the fact that the kernel can be expressed in terms of the density operator $\rho(x) = |\Phi(x)\rangle \langle \Phi(x)|$ as $K(x,z) = \text{tr}[\rho(x)\rho(z)]$, where
and the density operator can be always expanded by the set of Pauli operators as
\[ \rho(x) = \sum_{i=1}^{4^n} a_i(x) \sigma_i \]  
with \( a_i(x) \in \mathbb{R} \) and \( \sigma_i \in P_n = \{I, X, Y, Z\} \otimes^n \) the multi-qubit Pauli operators. The followings are the example elements of \( P_2 \):

\[
\begin{align*}
XI &= \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, & ZY &= \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \otimes \begin{bmatrix} 0 & i \\ i & 0 \end{bmatrix}, & YZ &= \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.
\end{align*}
\]

Then, by substituting Eq. (3) into Eq. (2) and using the trace relation \( \text{tr}(\sigma_i \sigma_j) = 2^n \delta_{i,j} \), the kernel can be written as
\[ K(x, z) = 2^n \sum_{i=1}^{4^n} a_i(x) a_i(z). \]  

This expression means that the vector of coefficients \( a(x) = [a_1(x), \ldots, a_{4^n}(x)]^\top \) is also interpreted as a feature map corresponding to the kernel \( K(x, z) \). That is, the input dataset \( \{x_i\} \) are encoded into the set of vectors \( \{a(x_i)\} \) in a much-bigger real feature space \( \mathbb{R}^{4^n} \) and will be classified by the SVM with the kernel (4). Note that \( a(x) \) is a generalization of the Bloch vector, and thus the corresponding feature space is the generalized Bloch sphere.

### 2.2 Feature map for 2-dimensional input data

We now focus on the case of two-dimensional input data \( x \in \mathbb{R}^2 \) and particularly follow the procedure of [12]; in the case \( n = 2 \), an input data \( x \) is mapped to the unitary operator \( U_{\Phi(x)} \) through the set of encoding functions \( \Phi(x) = \{\phi_1(x), \phi_2(x), \phi_{1,2}(x)\} \), which is composed of two layers of Hadamard gate \( H^\otimes 2 \) and the unitary \( U_{\Phi(x)} \) as follows:
\[
U_{\Phi(x)} = U_{\Phi_{1}(x)} H^\otimes 2 U_{\Phi_{2}(x)} H^\otimes 2,
\]
where
\[
U_{\Phi(x)} = \exp(i\phi_1(x)ZI + i\phi_2(x)IZ + i\phi_{1,2}(x)ZZ) .
\]

The quantum circuit representation realizing this unitary gate is shown in Fig. 1. The three user-defined encoding functions \( \phi_1(x), \phi_2(x), \) and \( \phi_{1,2}(x) \) nonlinearly transform
the input data $\mathbf{x}$ into the qubit $|\Phi(\mathbf{x})\rangle = \mathcal{H}_{\Phi(\mathbf{x})} |0\rangle^\otimes 2$. Now, through a lengthy calculation, we have the explicit Pauli decomposed form (3) of the density operator $\rho(\mathbf{x}) = |\Phi(\mathbf{x})\rangle \langle \Phi(\mathbf{x})|$; the coefficients $\{a_{ij}(\mathbf{x})\}$ with $i, j = I, X, Y, Z$ are listed in Table 1. The coefficients are composed of bunch of trigonometric functions, which make the kernel complicated enough to transform the input data highly nonlinearly.

Table 1 Pauli decomposition coefficients for 2 dimensional input data, $\{a_{ij}(\mathbf{x})\}$ with $i, j = I, X, Y, Z$.

| Index $ij$ | Pauli decomposition coefficients $a_{ij}$ |
|-----------|------------------------------------------|
| II        | $\frac{1}{4}$                            |
| IX        | $\frac{1}{4} \sin \phi_1 \sin \phi_2 \cos \phi_{12} \cos \phi_{12} + \cos \phi_1 \cos \phi_2 \sin \phi_{12}$ |
| IY        | $\frac{1}{4} \cos \phi_2 \cos \phi_{12} \sin \phi_{12} + \cos \phi_1 \sin \phi_{12}$ |
| IZ        | $\frac{1}{4} \sin \phi_1 \sin \phi_2 \cos \phi_{12} \sin \phi_{12} + \cos \phi_1 \cos \phi_2 \cos \phi_{12}$ |
| XI        | $\frac{1}{4} \cos \phi_2 \cos \phi_{12} \sin \phi_{12} + \cos \phi_1 \sin \phi_{12}$ |
| YI        | $\frac{1}{4} \cos \phi_2 \cos \phi_{12} \sin \phi_{12} + \cos \phi_1 \sin \phi_{12}$ |
| ZZ        | $\frac{1}{4} \cos \phi_1 \cos \phi_{12}$ |

2.3 Minimum accuracy

Here we give a convenient method for calculating the minimum accuracy. First, for a fixed index $i$ and the given training dataset $\{\mathbf{x}_k, y_k\}_{k=1, \ldots, N}$, we calculate the classification rate $R_i$ achieved by the function $a_i(\mathbf{x})$, through the following procedure: (i) for a fixed threshold between two neighboring data points, indicated by an arrow shown in Fig. 2, calculate the error rate of the classification task via this threshold on the 1-dimensional dataset $\{a_i(\mathbf{x}_k)\}_{k=1, \ldots, N}$, (ii) calculate the error rates for all the thresholds, and finally (iii) set the minimum error rate to $R_i$. Then, the minimum accuracy is defined as $\max_i R_i$, where the index now takes $i = 1, \ldots, 16$. Importantly, this procedure for obtaining the minimum accuracy can be readily done, because the data points are directly produced from the functions $\{a_i(\mathbf{x})\}$ listed in Table 1 and each classification task needs neither a kernel matrix nor the SVM optimization pro-
Fig. 3 Dataset called (a) “circle”, (b) “exp”, (c) “moon” and (d) “exp”.

Procedure but it is merely a line search. Also note that $R_i$ represents the success rate of classification via the hyperplane orthogonal to the $i$th axis; in view of the fact that the optimal hyperplane in the feature space, which best classifies the dataset, is in general orthogonal to none of the basis axis, the minimum accuracy $\max_i R_i$ gives a lower bound of the exact training accuracy achieved by the SVM. Hence, the minimum accuracy can be used as a convenient measure for estimating the worst-case accuracy and eventually a guide of selecting a suitable encoding function (i.e., feature map).

3 Results and Discussions

3.1 Classification accuracy with different encoding functions

Here we apply the quantum SVM method, with several encoding functions, to the benchmark classification problems. We consider the non-linear 2-dimensional datasets, “circle”, “exp”, “moon”, and “xor” shown in Fig. 3 in each case, 100 data-points are generated and categorized to two groups (orange or blue). The datasets are encoded
Table 2  Training accuracy achieved by the quantum SVM method, for 4 non-linear datasets and five different encoding functions.

| encoding function | circle | exp | moon | xor   |
|-------------------|--------|-----|------|-------|
| (7)               | 0.998  | 0.880| 0.863| 0.963 |
| (8)               | 0.985  | 0.955| 0.978| 0.965 |
| (9)               | 1.000  | 0.980| 0.845| 0.990 |
| (10)              | 1.000  | 0.910| 0.928| 0.988 |
| (11)              | 1.000  | 0.980| 0.968| 0.928 |

The reason of fixing $\phi_1(x) = x_1$ and $\phi_2(x) = x_2$ for all the encoding functions is that our aim is to investigate the dependence of the classification accuracy on $\phi_{1,2}(x)$.

3.2 Analysis of the feature map with Pauli decomposition coefficients

We now apply our method to the classification problem described above, and see how the new feature map $a(x)$ would benefit us to figure out the distribution of dataset in
Table 3 Test accuracy achieved by the quantum SVM method, for 4 non-linear datasets and five different encoding functions.

| encoding function | circle | exp | moon | xor |
|------------------|--------|-----|------|-----|
| (7)              | 0.99   | 0.89| 0.84 | 0.96|
| (8)              | 0.95   | 0.93| 0.91 | 0.93|
| (9)              | 1.00   | 0.97| 0.80 | 0.97|
| (10)             | 1.00   | 0.90| 0.93 | 0.99|
| (11)             | 1.00   | 0.96| 0.91 | 0.90|

the feature space, and moreover, enable us to predict a suitable encoding function using the minimum accuracy.

First, Figs. 5-9 show $a_{ij}(x)$ with $i, j = I, X, Y, Z$ for the encoding functions (7)-(11), respectively. Note that each $a_{ij}(x)$ is a 2-dimensional real space which itself does not depend on the input dataset. Nevertheless, very interestingly, some of those 2-dimensional spaces have a clear relationship to the shape of distribution of the input dataset, which will thus affect on the resulting classification accuracy. For example, in all cases of Figs. 5-9 the ZZ element $a_{ZZ}(x)$ has a circle shape, meaning that the "circle" dataset can be classified only by the feature map $a_{ZZ}(x)$; this observation is consistent with the fact that the "circle" dataset can be indeed classified with high training/test accuracies as shown in Tables 2 and 3. Similar results can also be clearly found in Fig. 5 (the case of encoding function (7)) and in Fig. 7 (the case of encoding function (9)); the shape of $a_{YX}(x)$ in Fig. 5 has a similar distribution to the "xor" dataset, and actually the encoding function (7) achieves the high training accuracy 0.963 for the "xor" dataset; the shape of $a_{YY}(x)$ in Fig. 7 has a similar distribution to the "exp" dataset, and actually the encoding function (9) achieves the high training accuracy 0.98 for the "exp" dataset.

Next, Table 4 gives the minimum accuracy for the encoding functions (7)-(11), which are calculated according to the procedure given in Section 2.3. Recall now that the minimum accuracy gives a lower bound of the exact training accuracy, or equivalently, guarantees a worst-case accuracy. Hence, our suggestion is to select the encoding function that has the largest minimum accuracy; that is, for the "exp" dataset, the encoding function (10) is recommended to use; similarly, for the "moon" and "xor" datasets, the encoding functions (10) and (7) are recommended, respectively. Also for the "circle" case, any encoding function can be used. Now let us compare the minimum accuracy with the exact training accuracies given in Table 2. Figure 4 gives the summary, where the minimum and exact accuracies are indicated with the orange and blue bars, respectively. Importantly, the encoding function selected according to the above-mentioned guide based on the minimum accuracy, does not necessarily produce the best training accuracies. Nonetheless, for all the cases of dataset, the selected encoding functions achieve the training accuracies bigger than 0.9. Therefore, in light of the fact that the minimum accuracy is obtained immediately without calculating the kernel matrix nor performing the SVM optimization procedure, our conclusion is that the minimum accuracy can be used as a convenient measure for determining a suitable encoding function and a result a good feature map. Finally note that, as seen in Fig. 4(c), (d), the encoding function (9) for the "moon" and the
Table 4 The minimum accuracy calculated using the Pauli decomposition coefficients shown in Figs. 5-9.

| encoding function | circle | exp  | moon | xor  |
|-------------------|--------|------|------|------|
| (7)               | 0.990  | 0.773| 0.830| 0.988|
| (8)               | 0.990  | 0.758| 0.803| 0.908|
| (9)               | 0.990  | 0.855| 0.873| 0.848|
| (10)              | 0.990  | 0.880| 0.880| 0.843|
| (11)              | 0.990  | 0.813| 0.848| 0.775|

Fig. 4 Comparison between the exact training accuracy in Table 2 (blue) and the minimum accuracy in Table 4 (orange).

encoding function (7) for the “xor” do not give a lower bound, due to the statistical errors; this gap can be decreased by taking more sample points.

4 Conclusion

In this paper, we developed a method that uses the Pauli decomposition to define a new real-valued feature map; in contrast to the quantum feature map given by a quantum state (ket vector), the proposed real-valued feature map can be used to easily calculate the minimum accuracy, which serves as a lower bound of the exact accuracy; moreover, the associate (real, low-dimensional) feature spaces allow us to qualita-
tively see the structure of the input dataset. The proposed method has been examined to analyze four kinds of different non-linear datasets, with five different encoding functions. We then found that the minimum accuracy provides a good prediction for the classification accuracy, particularly when only one element of the real-valued feature map makes an important role in separating the dataset. Hence our method may be used as a tool to effectively screen a library of feature maps suitable for classification. In the future work, we are attempting to further improve the minimum accuracy to predict the classification results. We also plan to develop a tool with the hope that the procedure of designing a suitable feature map can ultimately be made systematic.

References

1. Hastie, T., Tibshirani, R., Friedman, J.H.: The elements of statistical learning: data mining, inference, and prediction, 2nd Edition. Springer series in statistics. Springer (2009)
2. Alpaydin, E.: Machine Learning: The New AI. MIT Press Essential Knowledge. MIT Press, Cambridge, MA (2016)
3. Raikwal, J.S., Saxena, K.: Article: Performance evaluation of svm and k-nearest neighbor algorithm over medical data set. International Journal of Computer Applications 50 (14), 35–39 (2012)
4. Yu, L., Wang, S., Lai, K., Zhou, L.: Bio-Inspired Credit Risk Analysis: Computational Intelligence with Support Vector Machines. Springer Berlin Heidelberg (2010)
5. Li Y., Bontcheva K., Cunningham H.: SVM based learning system for information extraction. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 3635 LNAI, pp. 319–339 (2005)
6. Osuna, E., Freund, R., Girosit, F.: Training support vector machines: an application to face detection. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 130–136 (1997)
Fig. 6 Color map of all the 2-dimensional spaces $a_{i,j}(x)$ with the encoding function (8).

Fig. 7 Color map of all the 2-dimensional spaces $a_{i,j}(x)$ with the encoding function (9).
Fig. 8 Color map of all the 2-dimensional spaces $a_{ij}(x)$ with the encoding function \( \mathbf{10} \).

Fig. 9 Color map of all the 2-dimensional spaces $a_{ij}(x)$ with the encoding function \( \mathbf{11} \).
7. Saha, I., Maulik, U., Bandyopadhyay, S., Plewczyński, D.: SVM ensemble fuzzy clustering for satellite image segmentation. IEEE Geoscience and Remote Sensing Letters 9(1), 52–55 (2012)
8. Rebentrost, P., Mohseni, M., Lloyd, S.: Quantum support vector machine for big data classification. Phys. Rev. Lett. 113, 130503 (2014)
9. Mitarai, K., Negoro, M., Kitagawa, M., Fuji, K.: Quantum circuit learning. Phys. Rev. A 98, 032309 (2018)
10. Bishwas, A.K., Mani, A., Palade, V.: An all-pair quantum SVM approach for big data multiclass classification. Quantum Information Processing 17(10), 282 (2018)
11. Li, T., Chakrabarti, S., Wu, X.: Sublinear quantum algorithms for training linear and kernel-based classifiers. In: K. Chaudhuri, R. Salakhutdinov (eds.) Proceedings of the 36th International Conference on Machine Learning, Proceedings of Machine Learning Research, vol. 97, pp. 3815–3824. PMLR, Long Beach, California, USA (2019)
12. Havlíček, V., Córdoles, A.D., Temme, K., Harrow, A.W., Kandala, A., Chow, J.M., Gambetta, J.M.: Supervised learning with quantum-enhanced feature spaces. Nature 567(7747), 209 (2019)
13. Schuld, M., Killoran, N.: Quantum machine learning in feature Hilbert spaces. Phys. Rev. Lett. 122, 040504 (2019)
14. Aleksandrowicz, G., Alexander, T., Barkoutsos, P., Bello, L., Ben-Haim, Y., Bucher, D., Cabrera-Hernández, F.J., Carballo-Franquis, J., Chen, A., Chen, C.F., Chow, J.M., Córdoles-Gonzales, A.D., Cross, A.J., Cross, A., Cruz-Benito, J., Culver, C., González, S.D.L.P., Torre, E.D.L., Ding, D., Dumitrescu, E., Duran, I., Esendebak, P., Everitt, M., Sertage, I.F., Frisch, A., Führer, A., Gambetta, J., Gago, B.G., Gomez-Mosquera, J., Greenberg, D., Hamamura, I., Havlicek, V., Hellmers, J., Herok, Ł., Horii, H., Hu, S., Imamichi, T., Itoko, T., Javadi-Abhari, A., Kanazawa, N., Karazeev, A., Krsulich, K., Liu, P., Liu, Y., Maeng, Y., Marques, M., Martin-Fernández, F.J., McClure, D.T., McKay, D., Meesala, S., Mezzacapo, A., Moll, N., Rodríguez, D.M., Nannicini, G., Nation, P., Ollitrault, P., O’Riordan, L.J., Paik, H., Pérez, J., Phan, A., Pistoia, M., Prutyanov, V., Reuter, M., Race, J., Davila, A.R., Rudy, R.H.P., Ryu, M., Sathyae, N., Schnabel, C., Schoute, E., Setia, K., Shi, Y., Silva, A., Siriaichi, Y., Sivarajah, S., Smolinski, J.A., Soeken, M., Takahashi, H., Tavernelli, I., Taylor, C., Taylour, P., Trabing, K., Treinish, M., Turner, W., Vogt-Lee, D., Vuillot, C., Wildstrom, J.A., Wilson, J., Winston, E., Wood, C., Wood, S., Wörner, S., Akhalwala, I.Y., Zoufal, C.: Qiskit: An open-source framework for quantum computing (2019)