Developing Songs Reflecting the Predicted Future Stock Price of IonQ

Jeffrey Tejada Peralta, Kevin Rapp, Mojtaba Khavaninzadeh, Mariah Smith-Jones, Allen Baranov

Introduction

There is no denying classical computing has provided humanity with an unprecedented level of creative liberty. With classical computers, researchers have been able to combine multiple fields of study to develop new technologies and solve complex problems. However, the limitations of classical computing have become increasingly evident, leading to the rise of quantum computing as a potential solution for tackling even more complex and challenging tasks.

Just as how classical computers combine fields that were seemingly unrelated, so too can quantum computers. We were inspired by the prompt to combine interests of ours in a new way. WIth a music major, avid investors, and quantum enthusiasts, we decided to combine finance and music in a way that has never been seen before. The Qrabs are excited to present the first quantum machine learning generated song inspired directly by predicted stock movements.

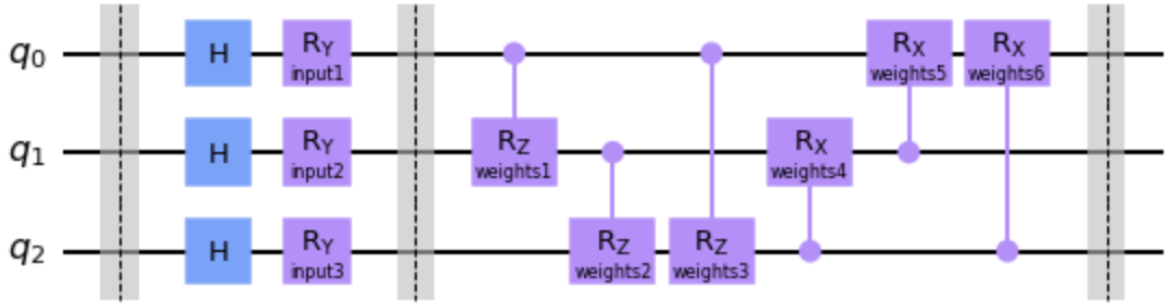
Without lack of generality, we use $IONQ as our stock of interest. Out of many possible parameters to choose from to develop our stock prediction model, we have selected a competitor's stock price, specifically $IBM. We have considered many variables which can be used such as using Twitter follow counts or quantum papers published each day, however, as a proof of concept, we will explore a very simple parameter with only one competitor's stock price.

After generating twenty business days forward worth of stock price, we used that data to generate music based on how we predicted the stock was going to perform. A positive stock movement would imply a happy note was generated, and a negative stock movement would imply a sad note. The results are shown, and we invite the reader to play around and generate their own songs!

Quantum Neural Network (QNN)

We created a quantum neural network to predict the $IONQ closing price every day for the next twenty business days. Our first plan of action was to find a training set for our neural network. We did this by using the Qiskit finance module on Python. We scraped the daily close of $IONQ and $IBM from 2021 to Friday, Jan 27, 2023. Then, we normalized these values from 0 to 2pi so that we could fit them into the phase of the qubits in the QNN we were going to make. After preparing the data, we split it into a train and test set so that we could rate the accuracy of the model. Only after this did we move onto the neural network.

The neural network was provided in large part by Qiskit's EstimatorQNN. It worked by applying H gates to three qubits, each of which held the same input value. The reason we used three qubits despite only having one input is because it allows for a more complicated QNN and more nonlinearity. Then, we encoded the input value as the phase of each qubit by applying a Ry gate to each qubit. Finally, to ensure entanglement and make the structure sophisticated enough to properly optimize, we added controlled Rx and Rz gates. This works for optimization purposes because all of these gates are linear maps and therefore do not affect optimization. After adding a classical wrapper around this QNN, we had a finalized product ready to provide predictions for the task of composing.



Quantum Composing

For the next portion of the project we took the data points from the output of the quantum neural network, and used them to create music. The music we composed consisted of a single line with varied pitch, rhythm, mood, and a consistent tempo.

The number of notes in the piece directly corresponds to the number of data points received as input; the number of data points we received was twenty, so our composition consists of twenty notes.

For the pitch of the piece we used a form of serialism, similar to that Schoenberg used in composing his tonal rows. We took all twelve tonal steps of a harmonic scale, and placed them inside of a probability matrix, where each note would only be able to move to specific other notes. However, unlike Schoenberg's rows, each note in our probability matrix has an equal chance to move to multiple notes that it harmonically makes sense with.

```
data_set_E = [0, 0.5, 0, 0, 0, 0.5, 0, 0, 0, 0, 0, 0]
data_set_D_sharp = [0.25, 0, 0, 0.25, 0.25, 0, 0.25, 0, 0, 0, 0, 0]
data_set_C_sharp = [0, 0, 0.33, 0, 0.33, 0.33, 0, 0, 0, 0, 0, 0]
data_set_G = [0, 0.33, 0, 0.33, 0, 0, 0, 0.33, 0, 0, 0, 0]
```

```
data_set_D = [0, 0.25, 0.25, 0, 0, 0, 0.25, 0, 0.25, 0, 0, 0]
data_set_F = [0.25, 0, 0.25, 0, 0, 0, 0, 0.25, 0, 0.25, 0, 0]

data_set_C = [0, 0.25, 0, 0, 0.25, 0, 0, 0, 0.25, 0, 0.25, 0]
data_set_F_sharp = [0, 0, 0, 0.25, 0, 0.25, 0, 0, 0, 0.25, 0.25, 0]
data_set_A = [0, 0, 0, 0, 0.25, 0, 0.25, 0.25, 0, 0.25, 0, 0.25]
data_set_G_sharp = [0, 0, 0, 0, 0, 0.25, 0, 0.25, 0.25, 0, 0.25, 0]
data_set_B = [0, 0, 0, 0, 0, 0, 0.25, 0.25, 0, 0.25, 0, 0.25]
data_set_A_sharp = [0, 0, 0, 0, 0, 0, 0, 0, 0.5, 0, 0.5, 0]
```

We then took the probabilities and normalized them from a range of [0:1] to [-1:1].

```
def normalize(given_list):
    list_min = min(given_list)
    list_max = max(given_list)

    normalized_list = [(2*np.pi*(given_list_item - list_min))/list_max
                       for given_list_item in given_list]

    return normalized_list

def unnormalize(given_list, reference_list):
    list_min = min(reference_list)
    list_max = max(reference_list)

    unnormalized_list = [(list_max*given_list_item)/(2*np.pi) + list_min
                         for given_list_item in given_list]

    return unnormalized_list
```

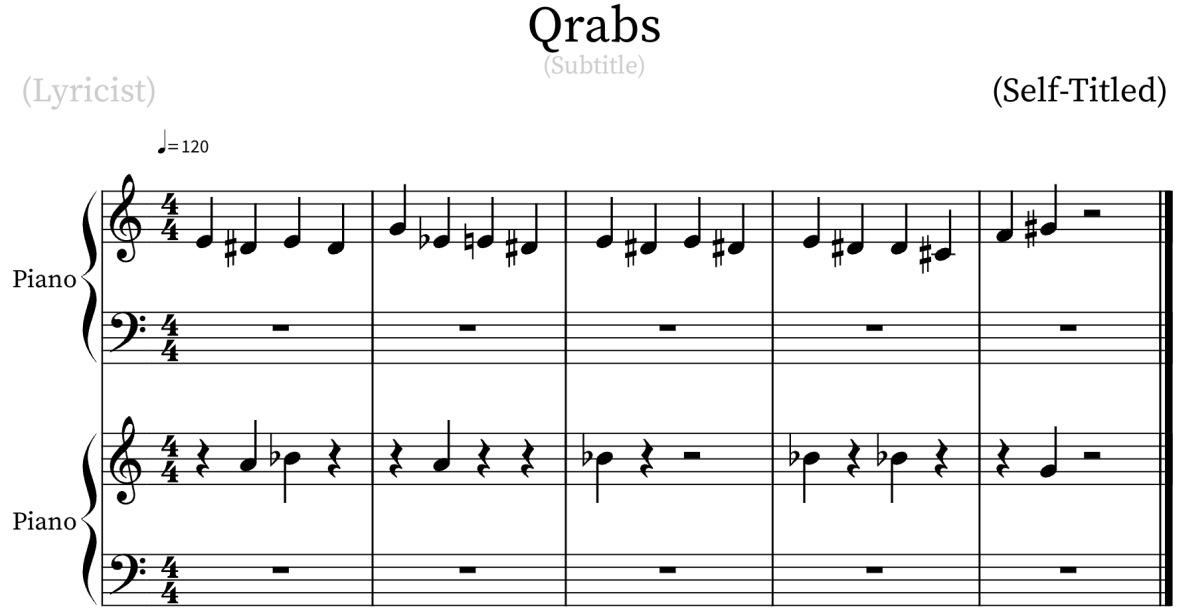
In order to determine the first note of the piece we take 4 qubits, apply Hadamard gates to all of them, read them out and measure them, remove the last four, and whichever note that remains with the highest count is the starting pitch of the piece. In our case the starting note of the piece was E.

The rhythm of the piece was determined by doing a Hadamard randomization. In order to keep the rhythm from becoming too complex, we only used eight different note length options. The

note lengths we decided to use were eighth notes, quarter notes, half notes, whole notes, and their dotted equivalents. Because we utilized eight different note length values, we only needed to use three qubits to represent them in the Hadamard randomization.

We struggled with determining how to convey a change in mood for the piece, and eventually concluded the best way to approach it was by playing a tritone when the slope goes down, and sticking with more harmonious iterations when the slope is going up.

The tempo of the piece was determined by the overall slope of the twenty data points we received. In order to have a consistent tempo throughout the whole piece, we decided to develop a rule for determining the tempo. If the overall slope was positive the tempo was 120 beats per minute, and if it was negative it was 80 beats per minute. In the rare occurrence the overall data slope was a horizontal line, the tempo of the piece would be 100 beats per minute. As the overall slope of the points was positive, our piece's temp was 120 beats per minute.



Due to time constraints, we were unable to incorporate the rhythm element of the music composition in this transcription

Next Steps

Time permitting, it would have been nice to add additional voices to the piece instead of only having one line. In order to prevent the piece from becoming too dissonant we would need to determine a more robust way to decide when to move from one note to the other in each voice.

Additionally, we had an idea to use ChatGPT to write lyrics to the song we wrote. Although this avenue could have been explored, we unfortunately ran out of time to add this feature.

We could have also trained the neural network more accurately. The neural network converges so it definitely has potential to perform extremely well, but it sometimes converges to an incorrect value. This makes the network feel far more fragile and infeasible, but with further research the problem could definitely be resolved.

Sources and Citations

1) https://pennylane.ai/qml/glossary/circuit_ansatz.html
2) https://qiskit.org/documentation/finance/tutorials/index.html

Miranda, E. R., & Basak, S. (2021). Quantum Computer Music: Foundations and initial experiments. *Quantum Computer Music*, 43–67. https://doi.org/10.1007/978-3-031-13909-3_3