Toward A Quantum Neural Network

Erick Serrano
Department Of Physics & Astronomy, University of Nevada Las Vegas

Abstract

In the 1950s, Alan Turing argued that modern digital computing technology holds the promise for artificial intelligence. With recent breakthroughs in quantum computing, such as the Shor and Grover algorithm in the 1990s, the future of quantum computing seems filled with time-efficient algorithms. This project attempts to take the next step in machine learning and quantum computing. Using **neural networks (NN)**, our goal is to:

- Implement a practical example of a NN model
- Explore the Time Complexity of our NN model
- Discuss key differences between classical and quantum computations
- Study ways to reduce the time resources needed for a NN using quantum methods

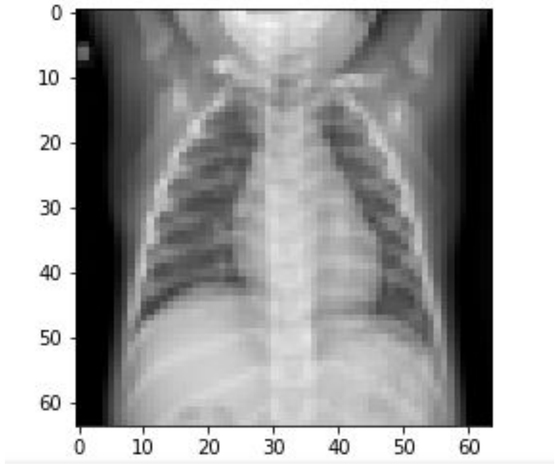

Figure 1. X-ray of a patient's lungs with 12,288 *input features.* These images were used to identify if a patient presented signs of pneumonia. The NN repeatedly computes the values of the input features until it produces a final prediction: yes or no.

NN Structure and Example

We created 4 NNs with 3 layers: an input, hidden, and output layer. Each NN used the ReLu function to predict whether a patient's x-rays presented symptoms of pneumonia. These NNs helped us:

1. Understand how to train a classical neural network using Gradient Descent
2. Understand the time it takes to train a NN using Gradient Descent

```
p_test_list = []
accuracy_test_list = []

for x in parameters:
        p_test, accuracy_test = predict(x_test, y_test, x, hidden_activation)

        p_test_list.append(p_test)

        accuracy_test_list.append(accuracy_test)

#Print accuracy of Neural Networks on test set
print("Accuracy of Neural Networks on test_list: ",accuracy_test_list)
```

```
Accuracy of Neural Networks on test_list:  [0.7932692307692306, 0.721153846153846, 0.685897435897435
9, 0.7516025641025641]
```

Figure 2. Code snippet that reveals the success rate of each NN. The best NN was approximately 80% accurate, and the average accuracy was approximately 74%

Results of the Shallow NN model

Of the four NNs, the most effective model was 79.3% successful using batch gradient descent. Meanwhile, the average success rate was approximately 74%, indicating that our NN model is precise and therefore a reliable model. Using Batch Gradient Descent, every NN trained approximately 3800 images in 1 hour.

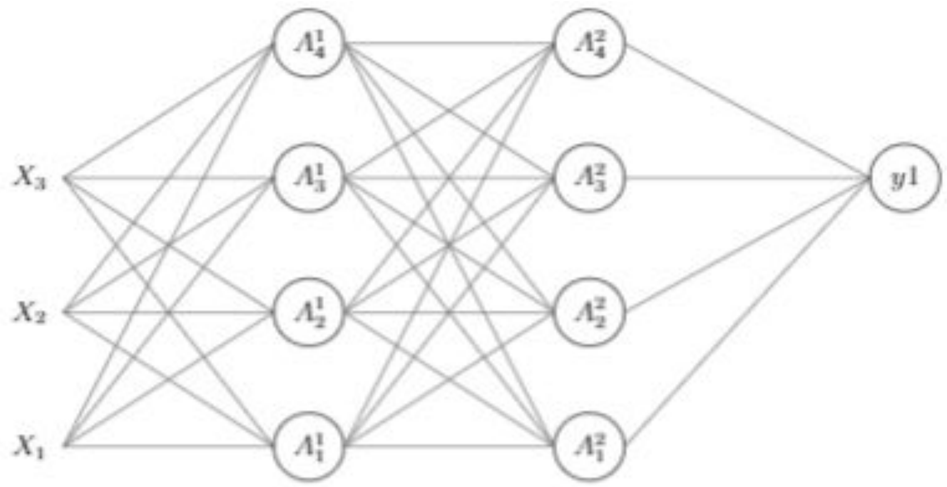

Figure 3. NN Structure with 1 input, hidden, and output layer. There are 4 nodes per input and hidden layer, and there is 1 node per output layer. In the figure, the NN is processing 3 input features X_1,X_2,X_3. In our implementation, each NN processes 12,288 input features (X_1,…,$X_{12,288}$)

Note that the number of output, hidden, and input nodes *can vary independently of each other*. The number of hidden layers and input features also change

Classical Vs. Quantum

Our NN implementation is entirely classical, and we demonstrated that it takes a prodigious amount of time resources before the NN can make successful predictions. A NN requires an hour to process thousand of images, and it can take days to process millions, even with the help of supercomputers. **This time constraint is a shortcoming of NNs that suggests Gradient Descent is time inefficient.**

This is where quantum comes in. Certain quantum algorithms, such as the Shor and Grover algorithms, were proven to be faster than classical algorithms in search and numerical factoring problems . As a result, quantum has since been encouraged as a contender in optimization problems, including the *optimization of neural networks*.

Proposed quantum improvements

Many scientists now recognize the optimization potential of quantum computing because of its superposition property. Our proposal is to reduce the time it takes a NN to learn. This proposed algorithm would *ideally serve as a replacement for gradient descent, compute the same X-ray images in a smaller time frame*, and underscore the advantages of quantum-based methods.

Acknowledgments

Thank you to Dr. Bernard Zygleman, an expert in quantum computation and my mentor for this project. Thank you also to the SRI for sponsoring this project.

References

1. Afshine Amidi and Shervine Amidi. 2020. Deep Learning cheat sheet. (2020). Retrieved October 10th 2020 from https://stanford.edu/~shervine/teaching/cs-229/cheatsheet-deep-learning
2. Hugo Mayo, Hashan Punchihewa, Julie Emile, and Jack Morrison. 2018. History of Machine Learning. Imperial College London. Retrieved September 30th 2020 from https://www.doc.ic.ac.uk/~jce317/history-machine-learning.html
3. Paul Mooney. 2017. Chest X-ray Images (Pneumonia). (2017). https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia
4. Andrew Ng. 2017. Gradient Descent – Neural Networks and Deep Learning. DeepLearning.Ai. https://www.coursera.org/lecture/neural-networks-deep-learning/gradient-descent-A0tBd
5. Bernard Zygelman. 2018. A First Introduction to Quantum Computing and Information. Springer International Publishing, Gewerbestrasse