HIGH SCHOOL EDITION
Journal of Student Research
Volume 11 Issue 4 (2022)

Quantum Computing for Self-Driving Cars and Pedestrian Detection

Shuhul Mujoo[1] and Usha Bhatnagar[#]

[1] Evergreen Valley High School, San José, California, USA
[#]Advisor

ABSTRACT

Quantum computing and self-driving cars are at the forefront of modern-day technological progress. One of the most crucial tasks for the safety of self-driving cars is pedestrian detection, which must be quick and accurate. Due to the enormous amounts of data involved, quantum computers are ideal as they provide exponential speedup. The benefits of quantum are twofold. First, the speed and accuracy of object detection is significantly improved. Second, groups of self-driving cars can communicate effectively with a centralized quantum computer, which leads to path selection efficiency. This paper presents a novel implementation of a k nearest neighbors classification algorithm, in both classical and quantum versions. The experimental procedure is outlined in four steps: data collection, classical nearest neighbor model, quantum nearest neighbors, and a wireless networking framework. The data consists of square grayscale images from the PennFudan (Shi, 2007) and DaimlerMono (Gavrila., 2012) datasets. Both models were tested by running them on testing data independent of training data. The classical model achieved fifty percent accuracy rate, and the quantum model ninety five percent. Due to the unavailability of real quantum computers, runtime was not tested, but theoretical constructs estimate the speedup to be on the order of millions for moderate input sizes. The networking framework was qualitatively analyzed to be feasible and sufficient. The successful implementation of pedestrian detection serves as a proof of concept and can be extended to a broader range of detections. The results demonstrate quantum computers as a possible solution to safe self-driving car technology.

Quantum Computing for Self-Driving Cars and Pedestrian Detection

Car technology has gone through numerous changes throughout the past century. From the gas-powered automobile to electric cars, advancements are made annually. Vehicles are the backbone of modern society, allowing for large scale cities and global trade of goods. Such an important technology continues to be improved in an ever-changing world (Gavrila., 2012). Automation is a large part of this improvement. The next step is self-driving capabilities.

Advancements are already being made in this field. Some cars can do rudimentary parallel parking if started in the correct location, and others lane drive. However, when the life of individuals is in the control of a computer, the number one priority is safety. Ethical implications aside, for self-driving cars to truly drive safe they would need to recognize situations not explicitly programmed (Harrison, 2018). This requires object detection and a clear vision of what a human driver would see. One of the most crucial parts of object detection is pedestrian detection.

If self-driving cars can reliably detect passersbys, then their overall security goes up. Additionally, machine learning methods can be used to translate this detection to other forms of objects such as sidewalks, signs, and bikes. Human error is by far the largest factor in accidents (Global, 2020), and the benefits of mitigating this variable outweigh the potential costs. Although no algorithm will be perfect, automated cars can potentially be safer and cheaper than human drivers.

Moreover, there are various other benefits of self-driving cars. Delivery services wouldn't need human drivers, and taxi services could have sets of preprogrammed cars. Not only would this save time, but the efficiency of this process is no longer dependent on the operators. If stable technology was created, nearly all vehicles could be automated. This makes traffic almost nonexistent and makes going somewhere easy as entering the destination (Global, 2020). Ideas of stop sign-less signals have been proposed, leading to unparalleled efficacy.

Why isn't this technology on the road yet? There are two reasons. First, the reliability of fully independent systems is far lower than even the worst human drivers. Second, the processing power of modern-day computers is limited, which leads to slow computation times, rendering the entire system unfeasible. Quantum computers are not limited by these constraints (Locef, 2015), having exponentially faster capabilities.

This purpose of this paper is to propose and detail one procedure for quantum pedestrian detection for self-driving cars. Additionality, a framework for wireless networks of fleets is shown qualitatively. The methods section gives a step-by-step procedure for this approach. The results show the capabilities when compared to state-of-the-art systems today. The discussion section analyzes the implications and potential impact. The conclusion gives a summary of the paper and a rationale for future study.

Methods

The code for this project was created using visual studio and the quantum development kit (QDK) by Microsoft. The procedure is described in detail below. The procedure consists of four main steps: data collection, classical model, quantum model, and wireless networking.

Data Collection

The data sources found about pedestrians were in the form of image sequences, which are the PennFudan and Daimler Mono datasets. PennFudan had ~200 images and Daimler Mono ~4000 images.

Image Sequences

The image sequences were 256 square grayscale images tagged with the number of pedestrians in each, with a separate category for zero pedestrians.

Cleaning Data

Some images were discarded due to either the similarity to other images in the sequence or lack of tagging.

Grey Level Co-Occurrence Matrix

This was used to create a feature representation of the image. For each image, the image is quantized, and the pixels are classified into eight bins. The gradients, averages, and variances are calculated. These are combined into four variables representing contrast, correlation, energy, and entropy. Figure 1 shows the algorithmic implementation.

```
# Normalize the quantized image P
normalize(P)
# Calculate the average and variance
mu = calcMu(P)
var = calcVariance(P)
# Set default values to 0
contrast, correlation, energy, entropy = (0, 0, 0, 0)
# Iterate through the
for i in range(P.shape[0]):
    for j in range(P.shape[1]):
        # Get the value of the pixel at point (i,j)
        p_value = P[i][j]
        # Calculate the contrast
        contrast += p_value*pow((i-j), 2)
        # Calculate the correlation
        correlation += p_value*(i-mu)*(j-mu)/var
        # Calculate the energy
        energy += pow(p_value, 2)
        # Calculate the entropy
        entropy += -math.log(p_value)*p_value
        # Append the values to the arrays
        contrasts.append(contrast)
        correlations.append(correlation)
        energys.append(energy)
        entropys.append(entropy)
```

Figure 1. GLCM Code - Python code to use a gray scale co-occurrence matrix to create a feature representation of classical images. These are then saved as vector arrays.

Classical Model

A classical K nearest neighbor model was used to classify the images by number of pedestrians.

K Nearest Neighbors

This model was created with k equal to five, and eight input features. (Harrison, 2018)

Prediction Features

The features from the grey-level co-occurrence matrix were used to train and predict the model output.

Hyperparameters

The number of epochs, clusters, and learning rate were slightly increased for optimal performance.

Quantum Model

A paper on quantum image classification was used for the quantum model. (Dang, 2018)

Q-Sharp

The quantum programming language chosen was q sharp (Q#) due to accesses to qubit memory allocation. (Locef, 2015)

Quantum KNN

The quantum classifier consists of five subroutines that are combined to create the quantum KNN model (Li, 2020).

Quantum Comparator (QC): The quantum comparator determines which quantum state represents a larger number in big endian format. (Vudadha, 2014)

State Preparation (SP): This is used to convert the classical features into quantum registers. Figure 2 shows the preparation of the input states.

$$|\alpha\rangle = \frac{1}{\sqrt{N}} \sum_{i=1}^{N} |i\rangle \left(\sqrt{1 - v_{0i}^2}\,|0\rangle + v_{0i}\,|1\rangle\right) |0\rangle \tag{3}$$

$$|\beta\rangle = \frac{1}{\sqrt{M}} \sum_{j=1}^{M} |j\rangle \frac{1}{\sqrt{N}} \sum_{i=1}^{N} |i\rangle\, |0\rangle \left(\sqrt{1 - v_{ji}^2}\,|0\rangle + v_{ji}\,|1\rangle\right) \tag{4}$$

Figure 2. State Preparation - The preparation of the alpha and beta input states respectively. The amplitude of each state is encoded using the novel enhanced quantum representation algorithm.

Novel Enhanced Quantum Representation (NEQR): NEQR encodes the images as superpositions of qubits. (Zhang, 2013)

Durrs Algorithm (DA): This is the main step of the quantum model. Durrs algorithm finds the image with the minimum distance and uses it to classify the image. (Dürr, 1996)

Amplitude Estimation (AE): The final quantum state is translated back into a classical probability through the amplitude estimation algorithm This (Hoyer, 2000)

Wireless Networking Framework

The wireless networking framework is for communicating, processing the images in real-time and uploading the features to the quantum computer

Bell State Generator

This is done through the preparation of bell states which can be transferred through the quantum teleportation algorithm.

Quantum Teleportation (QT)

The quantum information can be transferred by passing classical bits that encode the qubit states. See Table 1 for comparison with the other subroutines.

Results

The method described in this paper was analyzed both qualitatively and quantitatively. All the original goals of the project were met, and the criteria satisfied. The quantum model was clearly showed to be better than the classical model; the quantum model had a lower than five percent error rate. The most significant result was the less than ten percent false-negative rate on both datasets. The fact that the method rarely, if at all, missed the presence of pedestrians attests to its viability. The rest of the results detail the outcomes of the three main parts in depth.

Testing

Each part was tested independently, and the testing procedure was different for each one. The testing procedure and results are described in the following sections.

Classical Model

The classical model was analyzed with default machine learning metrics such as a confusion matrix and accuracy score. The classical model ran on both datasets PennFudan and DaimlerMono. Figure 3 shows the confusion matrix on DaimlerMono. Although the classical model had a slightly higher than fifty percent accuracy rate, it also identified the number of pedestrians, not only their presence.

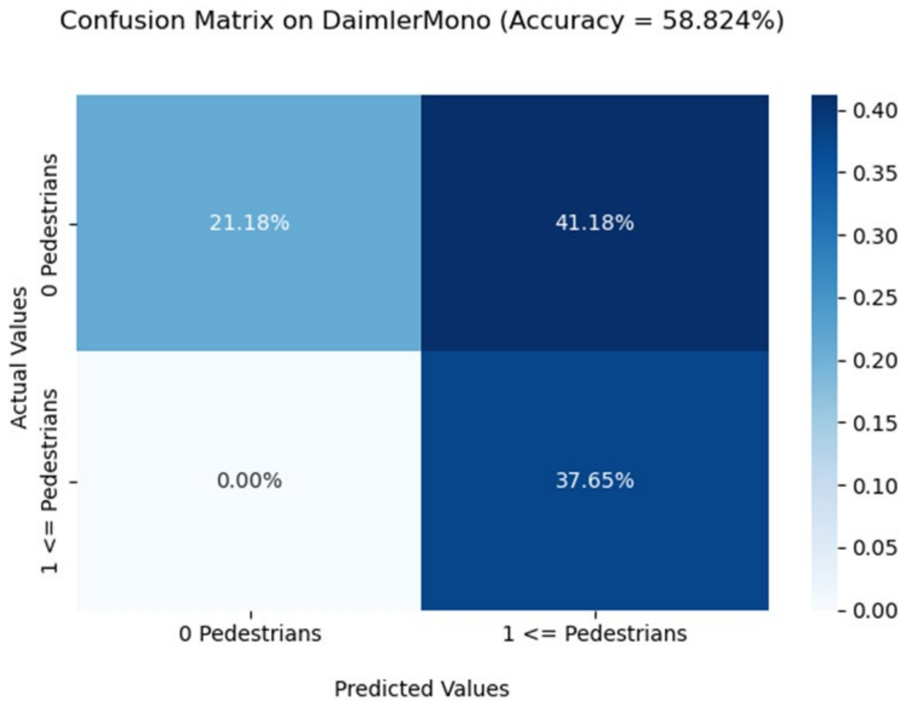

Figure 3. DaimlerMono Confusion Matrix - The confusion matrix of the classical model on the DaimlerMono dataset is shown above. Note the absence of false negatives and the distribution of other three categories.

Quantum Model

Due to the limitations of current quantum computers the complete quantum model could not be tested (explained in Unexpected Problems), but a cost analysis showed the number of resources used and possibilities of future testing. The individual subroutines were tested on the quantum simulator. This meant that the quantum model would be slower than the classical model, but this is only because the quantum k nearest neighbors required more qubits than currently possible to run on a real quantum computer today. Unit tests were created for each subroutine. Table 1 shows a comparison of each subroutine and their unit tests.

Table 1. Comparison of Quantum Subroutines

Subroutine	Qubits Required	Gates Required	Runtime (s)	Accuracy (%)
QC	35	74	28	100
SP	80	27	1	100
NEQR	40	36	14	100
DA	32	122	4	96
AE	78	62	1	99
QT	13	10	2	100

Note: Runtime reflects simulator performance. The real runtime is proportional to this value.

Wireless Networking

The wireless networking algorithm was tested through the quantum simulator and qualitative assessment will be done to see if it meets the criteria. The overall impact of the model was measured by its estimated speed up and reliability over the classical model, and the potential to be used through networks of quantum computers.

Discussion

The results prove the efficacy of the proposed method. From a purely technological standpoint, all the steps are in place for further development. However, there are many logistical and ethical challenges that remain. The next sections provide a detailed analysis of the method. The discussion section concludes with unexpected problems and future impact.

Classical Model

The classical model met the first criteria of above fifty percent accuracy. Most classical k nearest neighbor models have accuracies in the range of twenty to fifty percent. The classical model is far better than random trials and has a low false-negative rate. This is especially important for pedestrian detection. The model also predicted the presence of pedestrians when there weren't any which could signify overfitting to the data provided. Because the model is a simple k nearest neighbors classifier, the accuracy depends largely on the quality of the features extracted from the image (Harrison, 2018). Each image had eight features extracted, and by increasing this number the accuracy could be improved further. The accuracy of the model also shows that the features chosen are relatively good at identifying the image, but since these features were not created with pedestrian detection in mind, they are overgeneralized. Specific feature detection for pedestrians could drastically improve the model.

Quantum Model

The quantum subroutines all worked exceedingly well, but due to the program running on a simulator instead of a quantum computer, took longer than a classical counterpart would. Even though the algorithms use qubits, and are therefore inherently probabilistic, some of the subroutines produced one hundred percent accuracy rates, which was surprising. This is probably due to quantum gates that reversed the superposition effect done earlier. The subroutines implemented are used in a variety of other quantum algorithms making them even more versatile. Amplitude Estimation and Durrs Algorithm both had less than completely correct accuracies, but this is because of the actual algorithm itself and quantum superposition. Amplitude estimation is based on Grover's algorithm, which can be used to find a specific quantum state (Hoyer, 2000). This amplifies the probability of the correct state, but there is always the chance an incorrect state is produced. In a real application, the wrong result could simply be discarded, and the algorithm rerun until the correct result is produced.

Wireless Networking

The ability to transfer quantum information is essential when large scale networks of connections exist (Locef, 2015). Bell state generation was successfully carried out on the quantum simulator and this specific program was even tested on a real quantum computer from IBM (due to the small number of qubits required). The quantum teleportation also worked, and the state of an arbitrary qubit was successfully teleported. Expanding on this, by entangling large amounts of qubits, and using the internet to send normal bits, quantum information can easily be transferred from one quantum computer to another.

Unexpected Problems

The most major unexpected problem was the dependence on quantum RAM. The algorithm presented in the paper does not mention this but instead relies on Durrs' maximum finding algorithm, which again references another sub-algorithm. This final paper mentions that it depends on quantum RAM which is only a theoretical idea. Therefore, the combined quantum algorithm could not be tested, but all the individual subroutine's work. By implementing a modified maximum finding algorithm, even this subroutine was completed, but it did not provide an exponential speed up and was incompatible with the rest of the procedures. Despite this, since the subroutines individually work, they can be used for other quantum algorithms that do not require the use of QRAM.

Impact

The project is an advancement over currently available solutions due to the potential to be extremely fast and scalable. The proposed Quantum KNN is exponentially faster than the classical version and combined with wireless networking and the quantum teleportation algorithm can provide near-instant feedback. The quantum subroutines implemented in this project are not available to be imported as modules in quantum programming languages currently but using this implementation resolves the issue. This means that potentially in the future, even more accurate methods like a quantum neural network could also be implemented.

Conclusion

Quantum computing is one of the frontiers of modern-day technological progress along with self-driving car technology. By combining these, the project provides a path toward rapid development. Due to a large amount of data processing to be able to accurately move around in the environment, classical computers are unfeasible. Autonomous vehicles must be able to quickly detect pedestrians. By leveraging the power of quantum computers, the project ensures a successful detection with close to zero false negatives. Although a fully integrated quantum k nearest neighbors model was ideal, the subroutine implementations alone expand on the abilities of quantum computing. Combined with wireless networking, eventually, a future where quantum computers power the world is possible.

Although a fully connected pedestrian detection system was not possible there are many development options. Improvements to this project include different feature detection, new algorithm implementations, and the ability to test programs with large amounts of qubits. Besides the expected application of this framework to pedestrian detection for self-driving cars, this project acts as a proof of concept. If pedestrian detection proves to be successful, this can be extended to a broader range of detections, and eventually might make a quantum computer the driver behind the wheels of a fleet of self-driving cars.

Acknowledgments

I would like to thank my advisor for the valuable insight provided to me on this topic.

References

Dang, Y. (2018). Image Classification Based on Quantum KNN. IEEE Computer Society.

Dürr, C. (1996). A Quantum Algorithm for Finding the Minimum. DBLP Quantum.

Gavrila., D. (2012, March 6). Daimler Pedestrian Classification Benchmark Dataset. Retrieved from gavrila.net: http://www.gavrila.net/Datasets/Daimler_Pedestrian_Benchmark_D/Daimler_Mono_Ped__Class__Bench/daimler_mono_ped__class__bench.html

Global, G. (2020). What are self-driving cars? Retrieved from edu.gcfglobal.org: https://edu.gcfglobal.org/en/thenow/what-are-selfdriving-cars/1/

Harrison, O. (2018, September 10). Machine Learning Basics with the K-Nearest Neighbors Algorithm. Retrieved from towardsdatascience.com: https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761

Hoyer, P. (2000). Quantum Amplitude Amplification and Estimation. Quantum Computation AMS .

Li, J. (2020). Quantum K-nearest neighbor classification algorithm based on Hamming distance. Quantum Inf. Process.

Locef, M. (2015). A Course in Quantum Computing. Los Altos: Foothill College Press.

Shi, J. (2007, January 2). Penn-Fudan Database for Pedestrian Detection and Segmentation. Retrieved from upenn.edu: https://www.cis.upenn.edu/~jshi/ped_html/

Vudadha, C. (2014). An Optimized Design of Reversible Quantum Comparator. IEEE.

Zhang, Y. (2013). NEQR: A novel enhanced quantum representation of digital images. Research Gate.