Jerry Wang

jerrryw.github.io | github.com/jerrryw
(732) 853 – 5281 | jerryhhw23@gmail.com | linkedin.com/in/jerryhhw

Education

University of Maryland, College Park Expected May 2026
Master of Science in Applied Machine Learning
Rutgers University – New Brunswick May 2023
Bachelor of Science in Computer Science

Skills

Languages: C++, C, Python, Java
Frameworks: PyTorch, TensorFlow, Keras, Scikit-learn, OpenCV

Experience

Software QA Engineer Intern Oct 2020 – Aug 2021
Eslite Corporation – Taipei, Taiwan

- Improved search bar performance using AWS CloudSearch to customize text relevance ranking and indexing document data, resulting in a 15% increase in search performance
- Created and ran automated test scripts using Python, debugged multiple errors regarding shipment information, payment method, and delivery instructions

Software Engineer Intern June 2019 – Aug 2019
Acer – Taipei, Taiwan

- Engaged in a group project to build an automated guided vehicle with vision guidance using TensorFlow, OpenCV, and Microsoft Azure for Acer's workshop
- Trained a classifier model using TensorFlow and Keras to classify traffic signals and control the automated guided vehicle movements

Projects

Optimal Autoencoder

- Designed an optimal autoencoder with linear encoder and decoder with linear activation function to minimize the mean square error
- Trained the autoencoder with the rectified linear unit activation function and evaluated performance by varying the size of code dimensions

Neural Network Classification using MNIST Dataset

- Trained a feedforward neural network with fully-connected layers, including three hidden layers and using the rectified linear unit activation function
- Trained a convolutional neural network with three convolutional layers and two fully-connected layers, using the rectified linear unit activation function
- Achieved average testing accuracy of 0.985 over 20 iterations on the feedforward neural network and 0.991 on the convolutional neural network

Binary Classification with Dimensionality Reduction

- Trained binary classifiers using decision tree, k-nearest neighbor, linear discriminant analysis, and support vector machine algorithm
- Implemented the principal component analysis to approximate the training samples by reducing the data dimensionality into 5, 10, and 15 principal components
- Evaluated classifiers performance before and after applying principal component analysis by calculating the false positive and false negative error rate