Honors Final Report

Organization: MDP, Little Caesars Enterprises Inc.

Sponsors: Tim Somero, Kevin Hinks

Faculty Mentor: Brent Griffin

Team: Batuhan Akcay, Susie Alptekin, Ziyang Ji, Kevin Li, Yue Yao, Ran Yoo

Date: Dec 11, 2020

Introduction

Little Caesar Enterprises Inc. is the third-largest pizza chain in the United States. It operates and franchises pizza restaurants internationally. Little Caesars strives to always deliver the best pizzas to its customer. However, it is not easy to maintain consistency in quality at every place and every time.

One way to tackle this problem is to establish an automated pizza quality checking process. Our team uses Image Analysis and Computer Vision to help Little Caesars accomplish this goal. In the past year, we created a system that performs quality checking after pizzas come out of the oven and detects bad quality pizzas before being served to customers. The system is also connected to an online web application that offers real-time specific feedback to Little Caesars employees regarding the pizza quality. Employees are able to make changes as per the advice.

Project Review

This project focuses on building a software system to evaluate pizza quality. Our system will share feedback of pizza quality with Little Caesars employees given a pizza image. In the real use case, the images will be taken from cameras above the ovens in store kitchens. However, implementing the camera system is out of our project scope, so we only focus on working with images collected by ourselves. Deploying the Pizza Quality Evaluator in Little Caesars stores will help employees to prepare high quality pizzas with better consistency. Furthermore, in the future, the system could play an important role if Little Caesars decides to automate the process of preparing pizza.

The final deliverable of this project is a web application with built-in neural network models. The web application evaluates the quality of pizza given its image and generates feedback to the employees via the user interface (UI). For example, given a burnt pizza image, the final deliverable will inform the users that the pizza is not of high quality because it is burnt. Given a pizza image with incorrectly placed toppings (bad topping distribution), the users will be notified that the pizza is not of high quality because the toppings are not equally distributed on the pizza.

Team Approach

We divided our team members into 3 subteams: Model, Data, and User Interface (UI). The Model team was responsible for developing a fully functional model that analyzes pizza quality given pizza images. Neural networks along with image analysis techniques were used to accomplish this task. The Model team concluded that breaking up the quality analysis into smaller parts would improve the accuracy and the quality of the feedback. To execute this approach, neural networks were trained for each part of the quality analysis, totaling the 5 following models: Pizza Classifier, Pizza Detector, Burnt Classifier, Pizza Type Classifier, Pepperoni Detector. In addition to neural networks, the image analysis process was responsible for cropping images, centering the image on the pizza if it has one, and analyzing pepperoni distribution.

By pipelining the 5 neural network models together, the Model team were able to build an end to end system. Given an image, it will first go through Pizza Classifier to determine if a pizza is in the image. If so, the image will go through Pizza Detector to detect the location of the pizza. Then, the image is cropped and centered on the pizza. Then we will run Burnt Classifier on the image to classify if the pizza in the image is burnt. If all criteria passes so far, Pizza Type Classifier will be run on the image to classify the type of pizza. If the pizza type is pepperoni, we will run Pepperoni Detector to locate the pepperonis and check if the pepperoni numbers are within a specified range, and if they are evenly distributed. In order to improve the training speed, the Model team utilized a Microsoft Azure Virtual Machine equipped with a GPU.

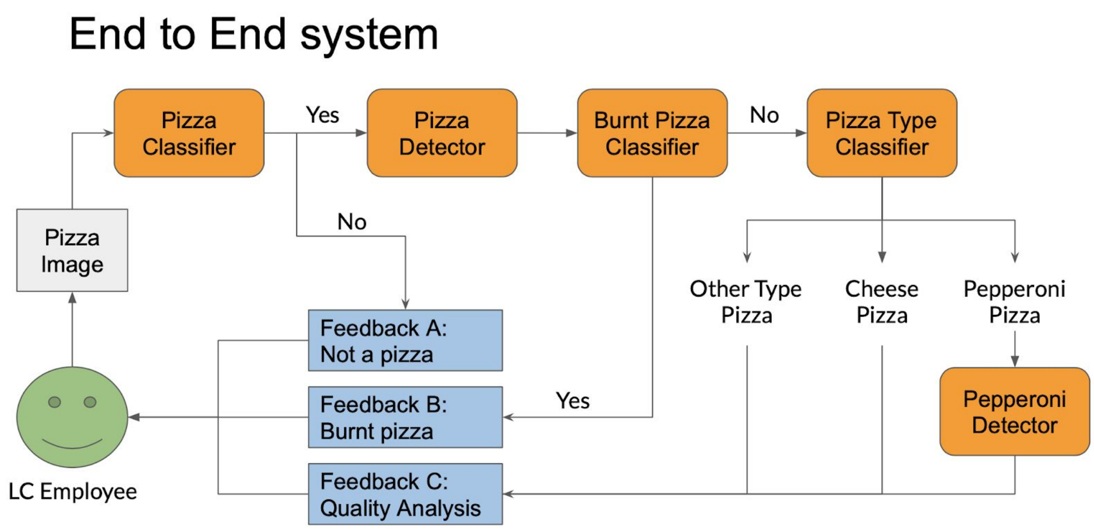

Figure 1: End-to-End system diagram

The Data team was responsible for collecting, storing, augmenting, and annotating a diverse set of images for the neural network models. Images were collected mainly from 3 sources. Most of them were from the internet. A few of them were provided by the Little Caesars corporate sponsors. The Data team also purchased some pizzas from Little Caesars store directly and took pizza images themselves with different conditions in a controlled environment.

Collected images were divided into 3 categories: training set, validation set, and testing set. The training set was to help the Model team train separate models. The validation set's images were similar to the training set. It was used to test the accuracy of individual models. The testing set images were never utilized during the training and validation of the models, but would be used to verify whether the real-life performance of the models passed the threshold values or not. More specifically, the testing set was composed of 90% good quality pizza, and 10% bad quality pizza, because the data team thought this ratio could best mimic the real-life scenario happening in Little Caesars store. The Data team used Microsoft Azure's Blob Storage to store the images, Microsoft's Visual Object Tagging Tool to annotate them, and Python scripts to create datasets from images and their corresponding annotations.

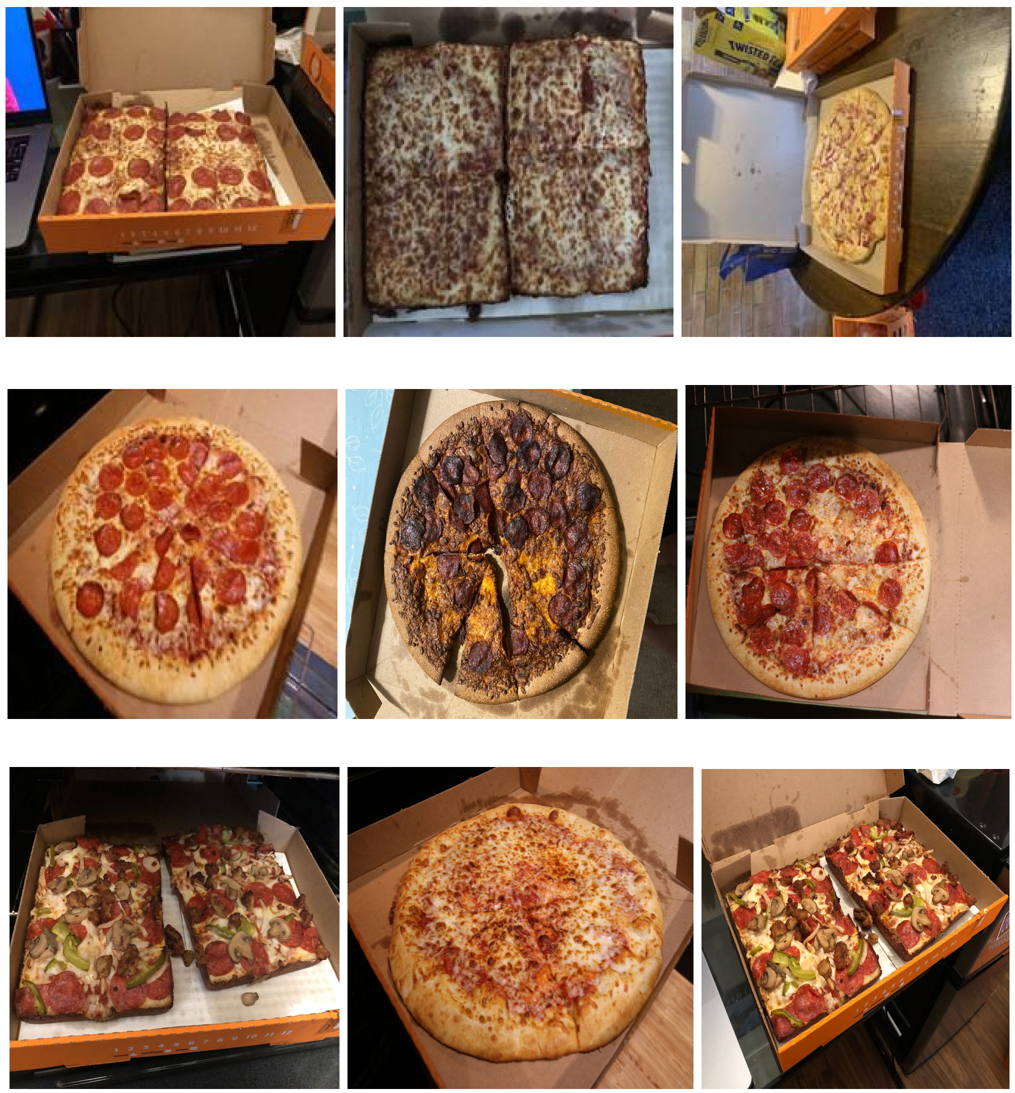

Figure 2: Sample Pizza Images from Testing and Validation Dataset

The User Interface team focused on building and deploying a user-friendly web application that runs the neural network models in the back-end. The web application would allow users to upload a pizza image, execute the end-to-end system to evaluate the quality of the pizza, and display appropriate feedback to users. The web application is hosted locally and is not accessible by users in the Little Caesars' internal network. This was approved by the Little Caesars sponsors due to security and confidentiality reasons. The User Interface team also conducted user testing with Little Caesars employees to improve the usability of the user interface and make sure our target users are satisfied with the functions of our project. Flask and Yarn were used for the back-end framework, while React was used for the front-end of the web application. The User Interface team used Microsoft Azure's App Services to deploy the web application.

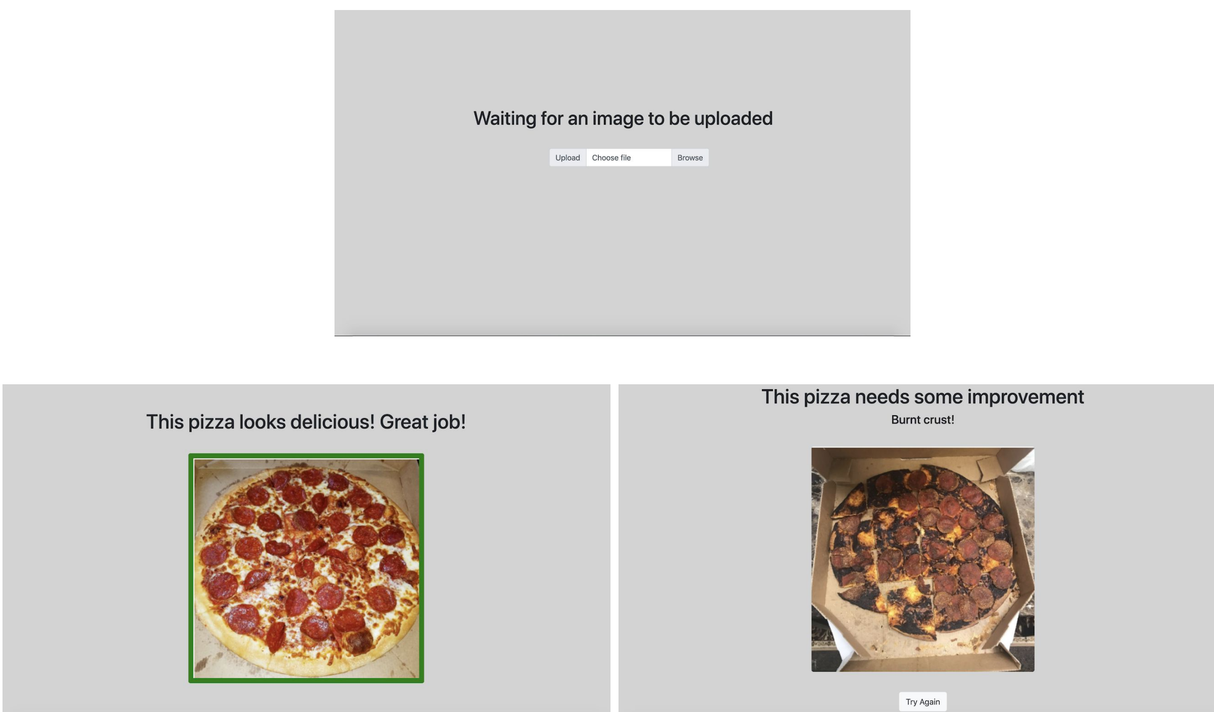

Figure 3: User Interface of the Prototype for the Web Application

Critical Requirements and Results

For the successful completion of the project, we defined our requirements as follows.

Number	Group	Requirement Target	Result
1.1	Model	Validation Set Accuracy: Pizza Classifier > 95% Burnt Classifier > 95% Pizza Type Classifier> 90% Pizza Detector AP 50 > 95% Pepperoni Detector AP 50 > 90%	Validation Set Accuracy: Pizza Classifier = 99.6% Burnt Classifier = 97.8% Pizza Type Classifier = 97.2% Pizza Detector AP 50 = 100% Pepperoni Detector AP 50 = 90.38%
1.2	Model	Testing Set Accuracy: Pizza Classifier > 95% Burnt Classifier > 95% Pizza Type Classifier> 90% Pizza Detector AP 50 > 95% Pepperoni Detector AP 50 > 90%	Testing Set Accuracy: Pizza Classifier = 98.52% Burnt Classifier = 99.22% Pizza Type Classifier = 97.48% Pizza Detector AP 50 = 100% Pepperoni Detector AP 50 = 90.93%
1.3	Model	End-to-End accuracy > 90%	End-to-End accuracy = 85.93%
2	System Integration	End-to-End system can process 10 images and give feedback under a total runtime of 1 minute	We didn't start testing because we prioritized satisfying model performance requirements over system runtime

3.1	User Interface	At least 80% of 15 users can navigate the application from image upload to receiving feedback with minimal guidance within 2 minutes	100% of 6 participating users can navigate the application from image upload to receiving feedback with minimal guidance within 2 minutes
3.2	User Interface	Conduct user testing on 15 users and at least 80% of users correctly identify the given feedback through survey form	Conduct user testing on 6 users and 100% of users correctly identify the given feedback through survey form

Table 1: Critical Requirements and Results

We chose these critical requirements since they represent our project requirements from frontend to backend. From users' side, we would like our interface to be easy to use and give clear feedback; from backend, in order for our whole system to work, we would like our model to give accurate predictions; and the connection between frontend and backend should be smooth and fast. The requirements above play an integral role in our minimum viable product, which is why they are high priority for our sponsors.

For requirement 2, currently, our backend model can process 1 image and make predictions under 1 second. However, loading all models and weight will take around 6 seconds. Due to time constraints and after discussing with our sponsors, we did not do further optimization because we prioritized satisfying model performance requirements over system runtime.

Model loading time mean over 100 images	5.986s
Pizza Classifier prediction time	0.036s
Pizza Detector +Burnt Classifier prediction time	0.317s
Pizza Type Classifier prediction time	0.039s
Pepperoni Detector prediction time	0.294s
Distribution Analysis prediction time	0.005s
Overall time	6.677s

Table 2: Model Loading and Processing Time

For critical requirement 3, due to Covid-19, we were unable to conduct user testing in person with kitchen employees. However, as an alternative, the team coordinated with our sponsors and performed the test with 6 corporate office employees who also had good knowledge on the kitchen operation.

Conclusion and Future Work

In the end to end system, the team found that feeding in a cropped image with pizza centered would give a much better result. Which meant that our pizza detection and pizza cropping process could still be improved. Considering this observation, below are a few suggestions for future work:

- Retrain Pizza Detector to detect more accurate pizza bounding boxes
- Conduct image analysis on uncropped images to make the whole system more robust

Despite a few unfilled requirements, our team learned and grew a lot through this project, both technically and professionally. One year ago when we started this project, we did not have any background knowledge in Machine Learning. But now we built a Computer Vision system from the ground up. We truly appreciate our sponsors and faculty mentor for providing us resources and guiding us through this journey.