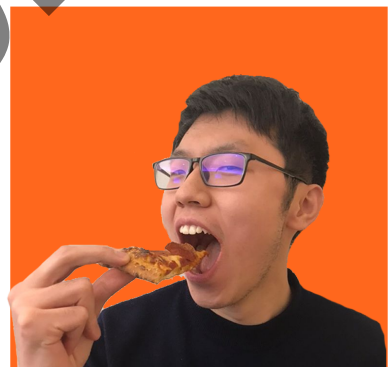
Meet Our Team

Kevin Li

Amanda Yao

Ran Yoo

Susie Alptekin



Ziyang Ji

Batuhan Akcay

Sponsors and Faculty

Sponsors:

Little Caesars Enterprises Inc.
Sponsor Company

Tim Somero
Lead Software Development

Kevin Hinks
Lead Software Engineer

Faculty:

Brent Griffin
Assistant Research Scientist, Electrical Engineering and Computer Science

MULTIDISCIPLINARY DESIGN PROGRAM

Agenda

1. Introduction & Project Review

- Project Background
- Expected Final Deliverable
- Demo
- Current Status

2. Critical Requirement

- Critical Requirement 1: Classifier Accuracy
- Critical Requirement 5: User Testing

3. Risk & Contingency

- Current Risk 2: Poor Model Performance

4. Q&A Section

1. Introduction & Project Review

- Project Background
- Expected Final Deliverable
- Demo
- Current Status

Motivation

COMPUTER VISION
+
MACHINE LEARNING

PIZZA ANALYSIS **=** **HIGHEST QUALITY PIZZA**

The best pizza

Every time

Expected Final Deliverable

1) Pizza Quality Analysis:

- Measure and evaluate pizza features that affect quality
- Determine pizza quality passes the LC standard

2) Feedback Generation:

- Create feedback depending on the pizza features evaluated

3) Web Application:

- Display the feedback to the employee on User Interface system
- Alert the employee if pizza does not pass LC standards

Approach

Model Team

- Neural Network Models:
  - PizzaClassifier
  - PizzaDetector
  - BurntClassifier
  - PizzaTypeClassifier
  - PepperoniDetector
- VM with GPU on Azure

Data Team

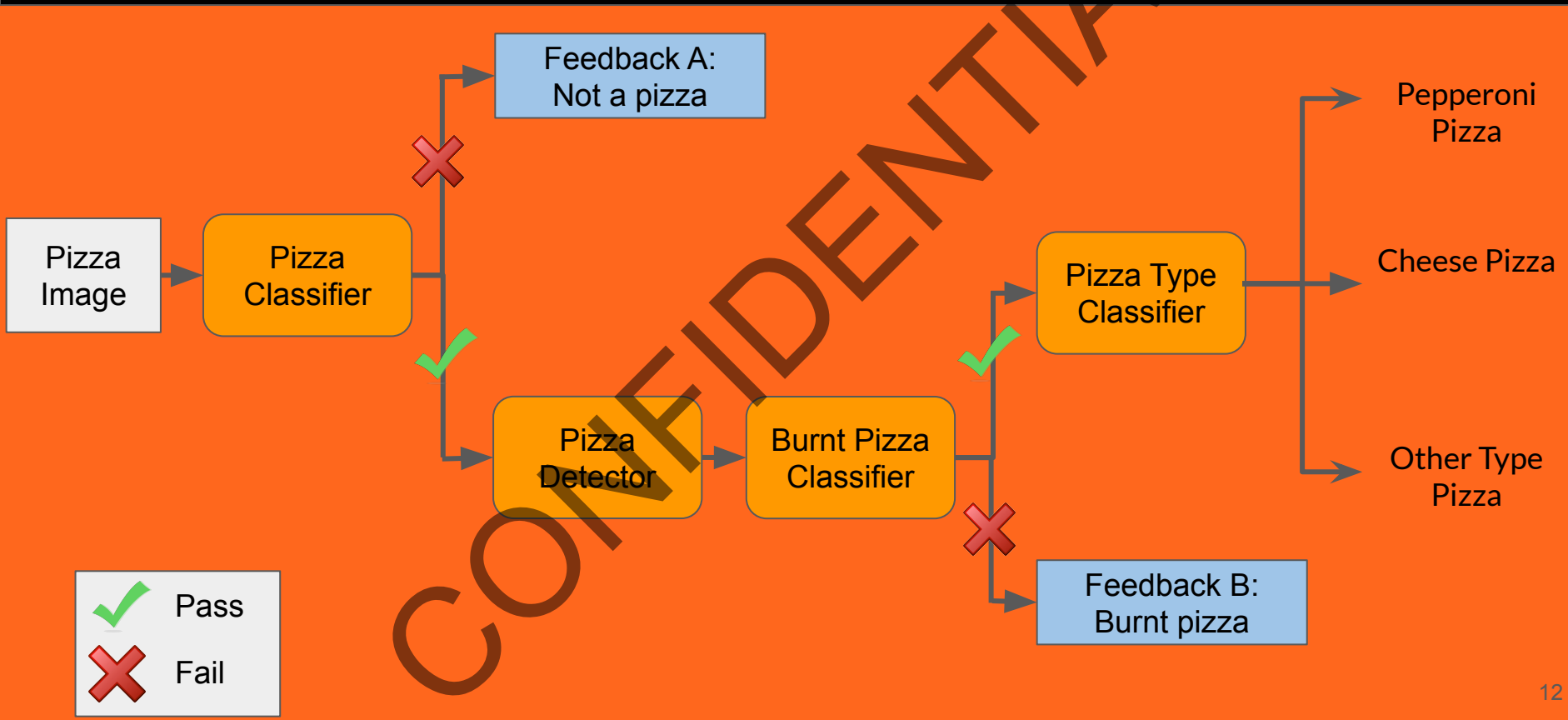
- Image collection for training, validation, and testing sets
  - Internet
  - Sponsor-Provided
  - LC pizza purchases funded by MDP

UI Team

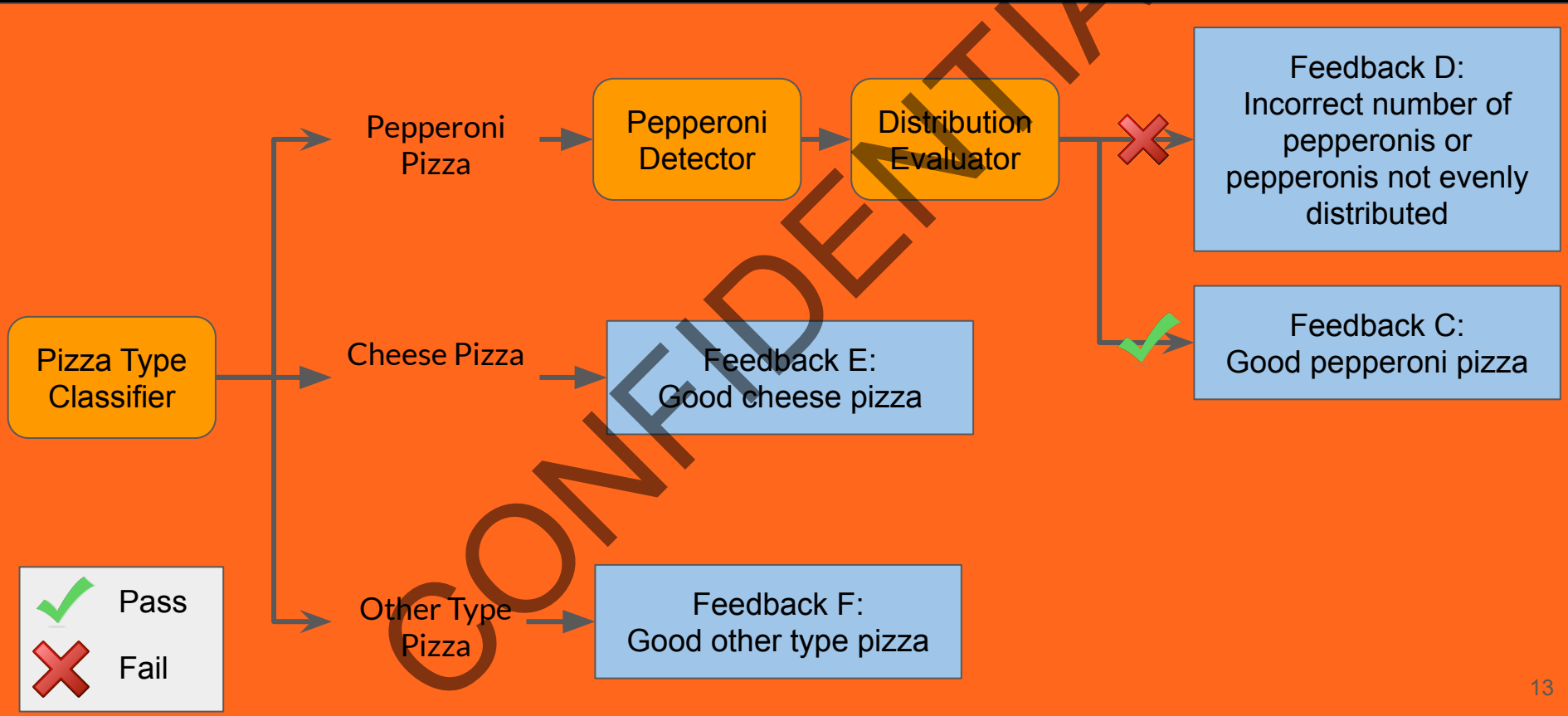
- Web application to connect frontend and backend
  - Flask
  - Nodejs
  - React

Current Status

Current Status: End-to-End System Diagram

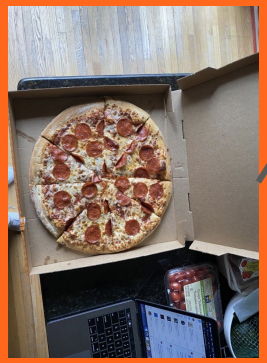


Current Status: End-to-End System Diagram



Current Status: End-to-End System Example

Original Image



Pizza Classifier

Pizza Detector

There's a pizza in this image.

Pizza detected and cropped!

Detecting pizza...

Model successful loaded!

Welcome to pizza quality checker!

Current Status: End-to-End System Example

Pizza detected and cropped!

Pizza detected and cropped!

Checking if it is burnt...
Ops... Pizza's burnt...

Checking if it is burnt...
Perfect timing!

Burnt Classifier

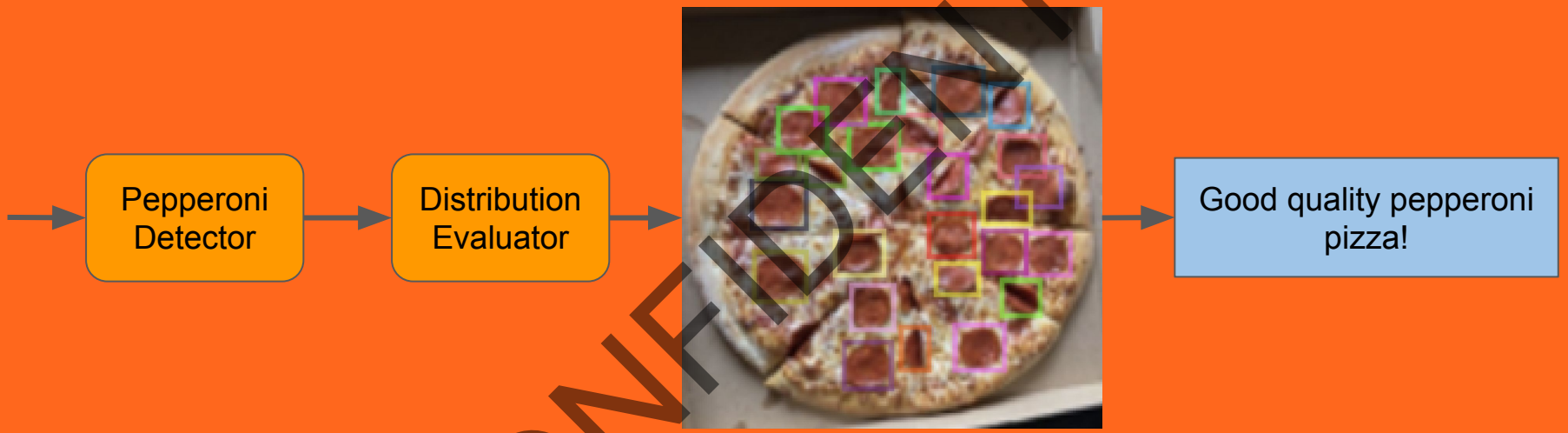
Pizza burnt!

Pizza Type Classifier

This is a pepperoni pizza.

Pepperoni Detector

Current Status: End-to-End System Example




```
This pepperoni pizza has 25 pepperonis.
Pepperoni count in 4 quadrants:  [7. 7. 4. 7.]
Passed pepperoni tests!
```

Current Status: End-to-End System Timing

Mean Time with Testing Set on GPU:

Model Loading Time:	
Mean over 100 times	5.986s
Prediction Time:	
PizzaClassifier	0.036s
PizzaDetector + BurntClassifier	0.317s
PizzaTypeClassifier	0.039s
PepperoniDetector	0.294s
DistributionAnalysis	0.005s
Overall (prediction)	0.691s

Current Status: Data

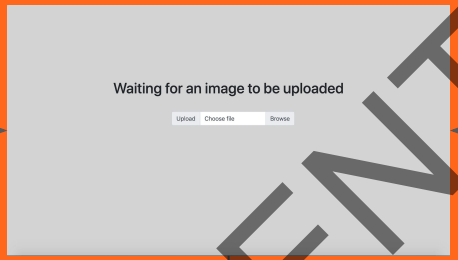
- Training and validation set collected from internet - 4000+
- Purchased sets of pizzas from Little Caesars with MDP funds ($87)
- Testing 1 - 352 images


- Testing 2 - 64 images

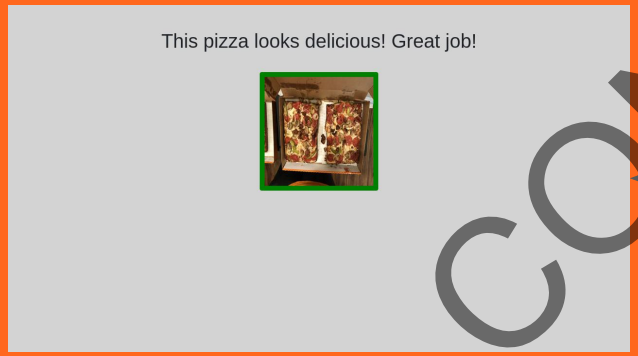

```
Label 0: Pizza
        0 = Not Pizza
        1 = Pizza
Label 1: Pizza Type
        1 = Pepperoni
        2 = Cheese
        3 = Other
Label 2: Pizza Shape
        0 = Round
        1 = Square
Label 3: Burnt?
        0 = Not Burnt
        1 = Burnt
Label 4: Topping Distribution
        0 = Good Topping Distribution
        1 = Bad Topping Dist
```

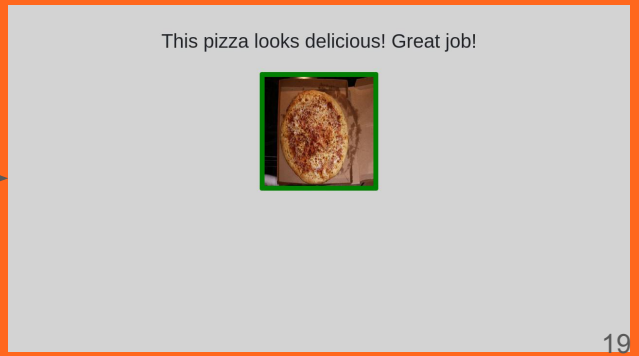
Current Image Annotation Method

Current Status: User Interface

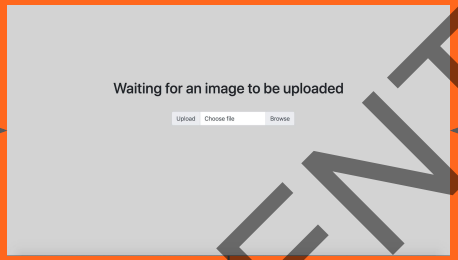


End to End Model System

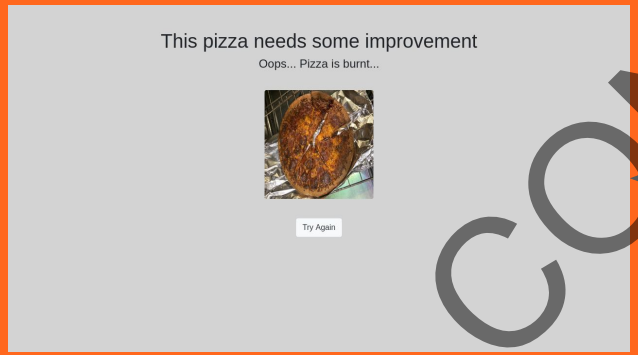




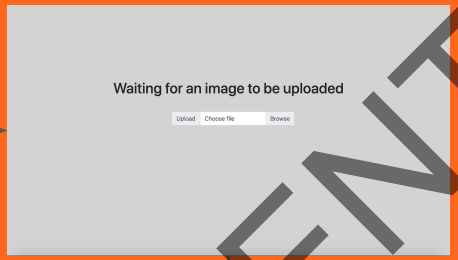
Current Status: User Interface



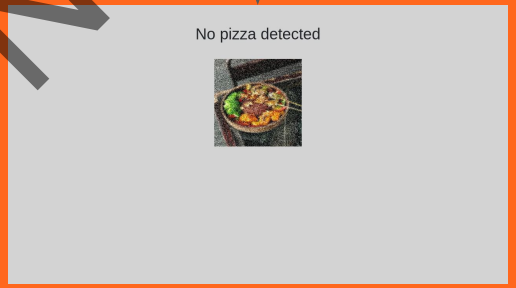
End to End Model System



Current Status: User Interface



End to End
Model System



2. Critical Requirements

- Critical Requirement 1: Classifier Accuracy
- Critical Requirement 5: User Testing

Critical Requirement 1: Model Development

Title: Classifier Accuracy Testing

Pass Criteria: accuracy on testing dataset:

	PizzaClassifier	BurntClassifier	PizzaTypeClassifier
Accuracy	95%	95%	90%

Yellow Status: but we are confident we can accomplish this goal

Model Development Validation Method

Running individual classifier model on validation set, which is 20% images we split from training set

Running end-to-end system on testing dataset, which comprises of images we took from Little Caesars pizza and largely represents real-world image distribution.

	Validation result	Initial testing result	Threshold
PizzaClassifier	99.6%	82.9%	95%
BurntClassifer	97.8%,	90.7%	95%
PizzaTypeClassifer	90.6%	84.0%	90%

Model Development Validation Method

Initial testing result failure cases analysis:

- Wrong Pizza Type (30% of failing cases)
- Not recognize square pizza (30% of failing cases)
- Pepperoni is folded when cutting pizza (10% of failing cases)
- Weird angle (10% of failing cases)

1. other pizza

2. not a pizza

3. not even

4. not a pizza

Critical Requirement 5: User Interface

Title: User Interface intuitive navigation testing

Pass Criteria: at least 12 of 15 users should navigate the application from image upload to receiving feedback with minimal guidance within 2 minutes

Red Status: COVID-19 pandemic and cybersecurity concerns with contacting 15 LC kitchen employees and providing them with demo

User Interface Validation Method

1. Set up interface running locally (cannot deploy due to security reasons)

```
## SETUP

#### Virtual Environment
Move to web_app direction
Run: python3 -m venv env
Run: source env/bin/activate
     cd src
     export FLASK_APP=uploading.py
     export FLASK_ENV=development
     export FLASK_DEBUG=1

#### Installations
web_app dependencies (npm install)
flask_cors (pip install flask-cors)
torch (pip install torch torchvision)
detectron2 (python -m pip install 'git+https://github.com/facebookresearch/detectron2.git')
pandas (pip install pandas)
cv2 (pip install opencv-python)
imageio (pip install imageio)

#### Spin Up Web App
In terminal window in web_app directory with virtual environment activated, run (npm start)
In separate terminal window in src directory with virtual environment activated, run (flask run)
```

User Interface Validation Method

2. For each of the 15 users, follow the script and ensure to record all data specified

"Hello and thank you for taking the time to assist in our testing suite. Using the MacOS software, we will conduct screen sharing so that you can interact with our interface"
"Please click the 'Accept' button as it appears on your Mac device"
Wait for connection to be established
"Alright, please verify that you can see my screen and have the ability to interact with the browser up"
Wait for confirmation
"Great, now I will direct you to the browser interface we will be testing"
Once on the correct browser for testing, proceed.
"Please follow the steps on the browser. Use the image we have provided. It is the only file located on this computer's desktop"
Start stopwatch
"Please read us back the feedback given once you have properly followed the progression of the interface"
Wait until they read the feedback back to you
"Now that you have identified the feedback, should the pizza be served to the customer or should the pizza be thrown away and the order remade"
Wait until they respond and record if they successfully identify proper next steps with the pizza
Stop stopwatch and record time for this test
If time is over 2 minutes then record status as unsuccessful otherwise record status as successful
"Thank you for your time, your responses have been recorded. I hope you have a good day. Goodbye."

3. Risk & Contingency

- Current Risk 2: Poor Model Performance

Risk & Contingency

Risk: Poor performance of neural network models on testing dataset

Impact: 8 (out of 10)

Trigger: Neural network models' performance scores on the testing set is less than the threshold values described in user requirements 1.1b, 1.2b, and 1.3 in Appendix A: Project Requirements

Risk & Contingency

Probability: 40% (as of Oct 1., DR3 Report)

Model	Performance Type	Target	Validation Data	Testing Data
Pizza Classifier	Accuracy	95%	99.6%	
Burnt Classifier	Accuracy	95%	97.8%	
Pizza Type Classifier	Accuracy	90%	90.6%	
Pizza Detector	AP50	95%		
Pepperoni Detector	AP50	90%		

Table: Performance of our neural network models (Oct 1.)

Risk & Contingency

Probability: %70 (as of Oct. 8, DR3 Presentation)

Model	Performance Type	Target	Validation Data	Testing Data
Pizza Classifier	Accuracy	95%	99.6%	82.9%
Burnt Classifier	Accuracy	95%	97.8%	90.7%
Pizza Type Classifier	Accuracy	90%	90.6%	84.0%
Pizza Detector	AP50	95%		
Pepperoni Detector	AP50	90%		

Table: Current performance of our neural network models (Oct. 8)

Risk & Contingency

Contingency Plan I: Retrain neural network models with the current testing set for neural network to train on more data that represents real world image distribution.

Cost & Benefits: For accurate real life performance measurements, testing data needs to be hold out from the data used for training and fine tuning the neural network models. Therefore we will need to gather a new testing set of pizza images by purchasing more LC pizzas with MDP funds.

Status: Activated

Responsible: Data Team and Model Team (Ran, Batuhan, Kevin, Amanda, Ziyang)

Risk & Contingency

Contingency Plan II: By November 13th, if accuracy is still lower than threshold values, we will try to use a single neural network model, PizzaEvaluationClassifier, instead of using the current end-to-end model system to aim for a higher performance score.

Cost & Benefits: In the short time between November 13th and the project deadline it might be easier to improve the performance of a single neural network instead of an end-to-end model sytem with multiple models however this approach would not allow us to give pizza-specific feedback like the end-to-end model system.

Status: Not Activated

Responsible: Data Team and Model Team (Ran, Batuhan, Kevin, Amanda, Ziyang)

THANK YOU!

Q&A

ADDITIONAL SLIDES:

List of All Critical Requirements

	Requirement Group	Requirement Target & Units	Origin of Validation Method	Responsible for Completion	Status
1	Model Development	PizzaClassifier detects pizza with at least 95% accuracy on testing dataset (i.e. images we took from Little Caesars pizza, which represents the image distribution of the problem we are tackling); BurntClassifier detects burnt pizza with at least 95% accuracy on testing dataset; PizzaTypeClassifier detects pizza with at least 90% accuracy on testing set	Student Developed	Model Team, Data Team	Yellow
2	Model Development	PizzaDetector detects pizza with at least 95% AP50 accuracy on testing set; PepperoniDetector detects the pepperonis with at least 90% AP50 accuracy on validation set	Student Developed	Model Team, Data Team	Yellow
3	Model Development	Integrate existing models into an End-to-End system and return 90% correct feedbacks on testing set of 50 images	Student Developed	Model Team	Yellow
4	System Integration	End-to-End system can process 5 datasets of 10 images and give feedback in a total runtime of 1 minute per dataset	Student Developed	Model Team	Yellow
5	UI Development	At least 80% of 15 users can navigate the application from image upload to receiving feedback with minimal guidance within 2 minutes	Student Developed	UI/UX Team	Red

List of All User Requirements

Requirement Number	Requirement Group	Sponsor Priority	Requirement Target & Units	Origin of Validation Method	Responsible for Completion
1.1a	Model Development	1	PizzaClassifier classifies pizza with at least 98% accuracy on validation set (i.e. images from online, similar distribution as training set); BurntClassifier classifies burnt pizza with at least 98% accuracy on validation set; PizzaTypeClassifier classifies pizza with at least 90% accuracy on validation set	Student Developed	Model Team
1.1b	Model Development	1	PizzaClassifier detects pizza with at least 95% accuracy on testing dataset (i.e. images we took from Little Caesars pizza, which represents the image distribution of the problem we are tackling); BurntClassifier detects burnt pizza with at least 95% accuracy on testing dataset; PizzaTypeClassifier detects pizza with at least 90% accuracy on testing set	Student Developed	Model Team, Data Team
1.2a	Model Development	1	PizzaDetector detects pizza with at least 95% AP50 accuracy on validation set; PepperoniDetector detects the pepperonis with at least 90% AP50 accuracy on validation set	Student Developed	Model Team
1.2b	Model Development	1	PizzaDetector detects pizza with at least 95% AP50 accuracy on testing set; PepperoniDetector detects the pepperonis with at least 90% AP50 accuracy on testing set	Student Developed	Model Team, Data Team

List of All User Requirements

Requirement Number	Requirement Group	Sponsor Priority	Requirement Target & Units	Origin of Validation Method	Responsible for Completion
1.3	Model Development	2	Integrate existing models into an End-to-End system and return 90% correct feedbacks on testing set of 50 images	Student Developed	Model Team
1.4	Model Development & Data	3	The model should be 100% retrainable with 5 different sizes of dataset and larger dataset should have accuracy greater than or equal to the smaller dataset	Student Developed	Model Team
2.1	System Integration	2	For 5 different datasets composed of 10 images End-to-End system can process and give feedback for each dataset under a total runtime of 1 minute	Student Developed	Model Team
3.1	User Interface	1	At least 80% of 15 users can navigate the application from image upload to receiving feedback with minimal guidance within 2 minutes	Student Developed	UI/UX Team
3.2	User Interface	2	Conduct user testing on 15 users and at least 80% of users correctly identify the given feedback through survey form	Student Developed	UI/UX Team

Risk & Contingency

Number	1
Risk	Poor User Experience
Impact (1:best, 10:worst)	7
Probability	50%
Trigger	User research testing described in user requirements 3.1 and 3.2 in Appendix A: Project Requirements fail by October 30th
Contingency Plans	Redesign the UI according to feedback from the user research survey, which we estimate will take 2 weeks. Since 2 weeks will be allocated for redesigning, drop the low priority task to deploy the model and web interface to a virtual machine. Do a second round of user testing by November 20th, if possible and this time conduct user testing with both sponsors and other LC employees.
Responsible	UI/UX Team (Susie, Kevin)

Risk & Contingency

Number	2
Risk	Poor performance of neural network models on testing dataset
Impact (1:best, 10:worst)	8
Probability	70%
Trigger	Neural network models' accuracy and AP50 scores on the testing set is less than the threshold values described in user requirements 1.1b, 1.2b, and 1.3 in Appendix A: Project Requirements
Contingency Plans	Retrain neural network models with the current testing set that represents real world image distribution. Gather a new testing set of pizza images in different conditions taken in a controlled environment, by purchasing more LC pizzas with MDP funds. Measure the accuracy of models with the new testing set. If accuracy is still lower than threshold values by November 13th, use a simple PizzaEvaluationClassifier instead of using the current end-to-end model system to aim for a higher accuracy score.
Responsible	Data Team and Model Team (Ran, Batuhan, Kevin, Amanda, Ziyang)

Risk & Contingency

Number	3
Risk	Team member unavailable
Impact (1:best, 10:worst)	6
Probability	65%
Trigger	A team member becomes unavailable due to Covid-19 or unforeseen reasons
Contingency Plan	Drop the low priority task of evaluating multiple varieties of toppings. Restructure the subteams according to the remaining high priority tasks and redistribute tasks to responsible subteam members.
Responsible	Current Team Lead

Risk 2: Poor performance of neural network models on testing dataset

Impact: 8 (out of 10)

Models might:

- return “no pizza in image” feedback even if there is a pizza in image (Pizza Classifier)
- treat as if there is a pizza in image, even though there is none (Pizza Classifier)
- crop and rescale wrong part of the image (Pizza Detector)
- return “burnt pizza” feedback even if the pizza is not burnt (Burnt Classifier)
- execute quality evaluation for wrong type of pizza (Pizza Type Classifier)
- Detect pepperonis incorrectly and return wrong results for the distribution (Pepperoni Detector)

Our end-to-end model system follows a waterflow process and if any of the neural network models in the system does not exhibit high performance, errors in quality analysis will occur and employee will receive incorrect feedback.

Risk 2: Poor performance of neural network models on testing dataset

Trigger: Neural network models' performance scores on the testing set is less than the threshold values described in user requirements 1.1b, 1.2b, and 1.3 in Appendix A: Project Requirements

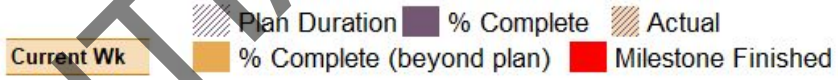
Req. Number	Req. Group	Sponsor Priority	Req. Target & Units	Origin of Validation Method	Responsible for Completion	Status
1.1b	Model Development	1	PizzaClassifier detects pizza with at least 95% accuracy on testing dataset (i.e. images we took from Little Caesars pizza, which represents the image distribution of the problem we are tackling); BurntClassifier detects burnt pizza with at least 95% accuracy on testing dataset; PizzaTypeClassifier detects pizza with at least 90% accuracy on testing set	Student Developed	Model Team, Data Team	Yellow
1.2b	Model Development	1	PizzaDetector detects pizza with at least 95% AP50 accuracy on testing set; PepperoniDetector detects the pepperonis with at least 90% AP50 accuracy on testing set	Student Developed	Model Team, Data Team	Yellow
1.3	Model Development	2	Integrate existing models into an End-to-End system and return 90% correct feedbacks on testing set of 50 images	Student Developed	Model Team	Yellow

Table: User requirements mentioned in the trigger of risk 2

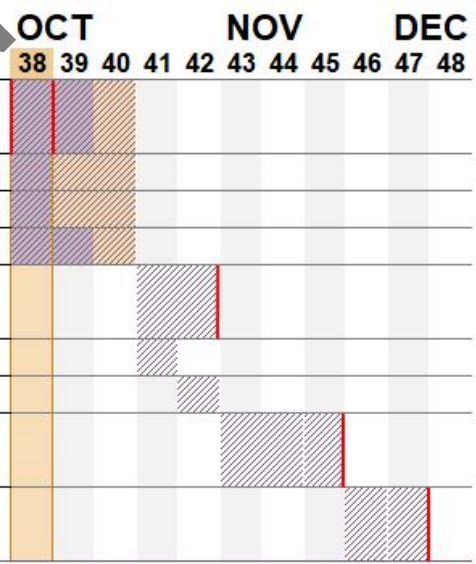
Future Work: User Interface

- Web application only running locally and not yet deployed for remote access
- Alert user that image has been successfully uploaded
- Conduct user testing

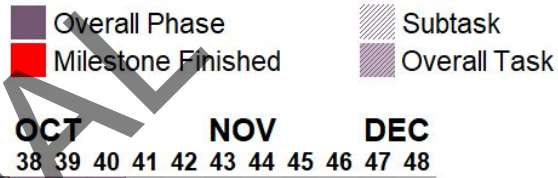
LCE Project Plan



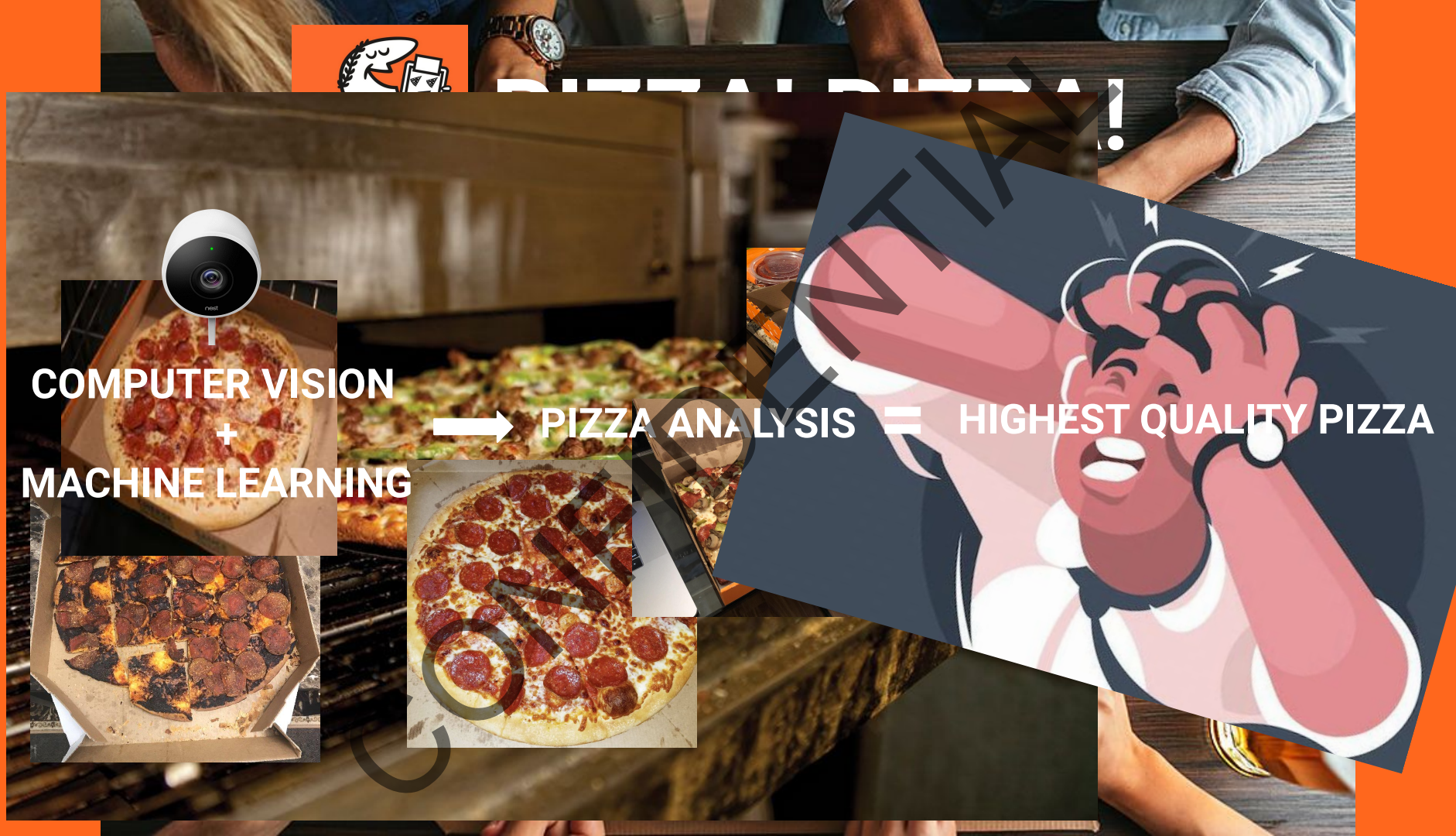
PHASE	MILESTONE	PERCENT COMPLETE
COMPLETE SOFTWARE DEVELOPMENT	6. Gather Feedback MDP: DR3	50%
Determine Quality of Pizza with Evaluator		50%
Develop User Interface and Browser		50%
System Integration		15%
FINAL TESTING AND ADDRESING ISSUES	7. Incorporate User and Sponsor Feedback	0%
Basic End to End Functionality Test		0%
User Testing		0%
CONTIGENCY TIME AND STRETCH GOALS	8. Finalize Product and Design	0%
FINAL DESIGN PRESENTATION AND DELIVERY	9. Final Product Delivery	0%



Detailed Project Plan



PHASE	MILESTONE
COMPLETE SOFTWARE DEVELOPMENT	1. Gather Feedback MDP: DR3
Determine Quality of Pizza with Evaluator - Model/Data Subteam	2. Distinguish Good vs Bad Pizza with Pizza Evaluator 3. Feedback Generator for Pizza
Collect Necessary Images Needed for Testing	
Train and Extract Evaluator Standards from Pizza Data (Pepperoni Distribution, Burnt)	
Evaluate the Extracted Quality with Binary Standards	
Provide Feedback Based on Evaluated Binary Standards	
Develop User Interface and Browser - UI/UX Subteam	4. Finalize Working User Interface
Develop Fully Functioning Local Interface and Backend	
System Integration - All Subteams	5. Connect Evaluator to Inteface
Connect Backend to End to End Model System	
FINAL TESTING AND ADDRESSING ISSUES	6. Incorporate User and Sponsor Feedback
End to End Integrated System Accuracy and Efficiency Testing	
User Interface User Testing	
CONTIGENCY TIME AND STRETCH GOALS	7. Finalize Product and Design
Deploy Interface on Web/Little Caesar Servers	
Develop Evaluator for Multiple and Non-pepperoni Toppings	
FINAL DESIGN PRESENTATION AND DELIVERY	8. Final Product Delivery


COMPUTER VISION
+
MACHINE LEARNING
PIZZA ANALYSIS
HIGHEST QUALITY PIZZA

What are we doing?

Burnt!

Final Deliverable:

A system to identify pizza quality: improve kitchen efficiency and help LC employees make perfect pizza consistently

Our System:

Classifier Models:
- Classify pizza types (i.e. cheese, pepperoni, others).
- Classify burnt vs not burnt pizzas.

Detector Models:
- Detect pizzas' relative locations in the image.
- Detect pizza toppings (i.e. pepperonis) and analyze topping distribution.

User Interface:
- Give feedback to employees about the pizza quality through UI.
- Simple and functional UI for employees.

With 90% accuracy on our system!

What does this accomplish?

Classifiers:

```
Ops... Pizza's burnt...
```

With the help of franchise infrastructure

- Identify pizza types: Hula Hawaiian, Supreme, …
- Identify pizza shapes: round, square, …
- Analyze for separate standards

Detectors:

```
This pepperoni pizza has 25 pepperonis.
Pepperoni count in 4 quadrants:  [7. 7. 4. 7.]
Passed pepperoni tests!
```

With the help of franchise infrastructure

- Detect topping types: black olives, ham…
- Analyze topping distributions
- Analyze for separate standards