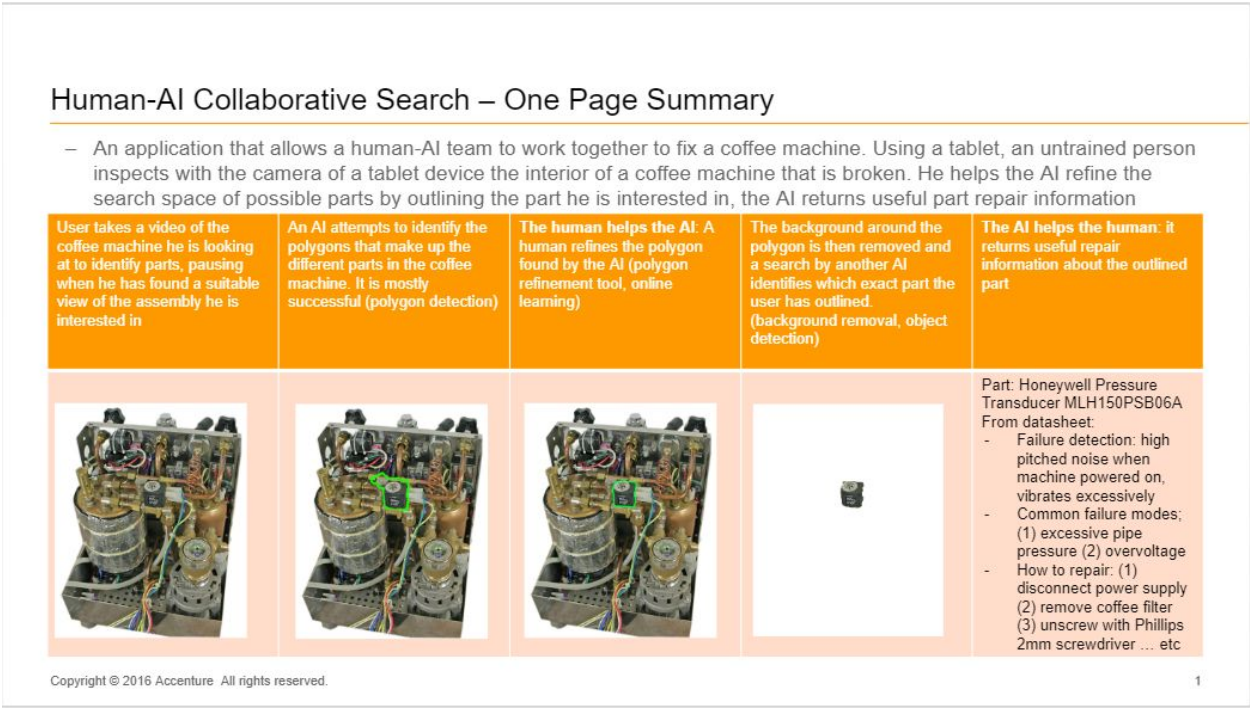
Naveen Ram and Tyler Lian

Repair Assistant (2018)

Accenture Tech Labs: Digital Experience

Introduction

This is an application that allows a human-AI team to work together to fix a machine. Using a tablet, an untrained person inspects the machine with a camera of a tablet device at the different parts of machine. The AI segments an object based on the database of images of the parts within the machine. They edit the AI's segmenting if the AI is not properly segmenting the part. The AI crops this segmented part and returns with part identification and part repair information located in its TensorFlow database.



Human-AI Collaborative Search – One Page Summary

– An application that allows a human-AI team to work together to fix a coffee machine. Using a tablet, an untrained person inspects with the camera of a tablet device the interior of a coffee machine that is broken. He helps the AI refine the search space of possible parts by outlining the part he is interested in, the AI returns useful part repair information

User takes a video of the coffee machine he is looking at to identify parts, pausing when he has found a suitable view of the assembly he is interested in	An AI attempts to identify the polygons that make up the different parts in the coffee machine. It is mostly successful (polygon detection)	**The human helps the AI**: A human refines the polygon found by the AI (polygon refinement tool, online learning)	The background around the polygon is then removed and a search by another AI identifies which exact part the user has outlined. (background removal, object detection)	**The AI helps the human**: it returns useful repair information about the outlined part
				Part: Honeywell Pressure Transducer MLH150PSB06A From datasheet: - Failure detection: high pitched noise when machine powered on, vibrates excessively - Common failure modes; (1) excessive pipe pressure (2) overvoltage - How to repair: (1) disconnect power supply (2) remove coffee filter (3) unscrew with Phillips 2mm screwdriver … etc

Polygon Segmentation

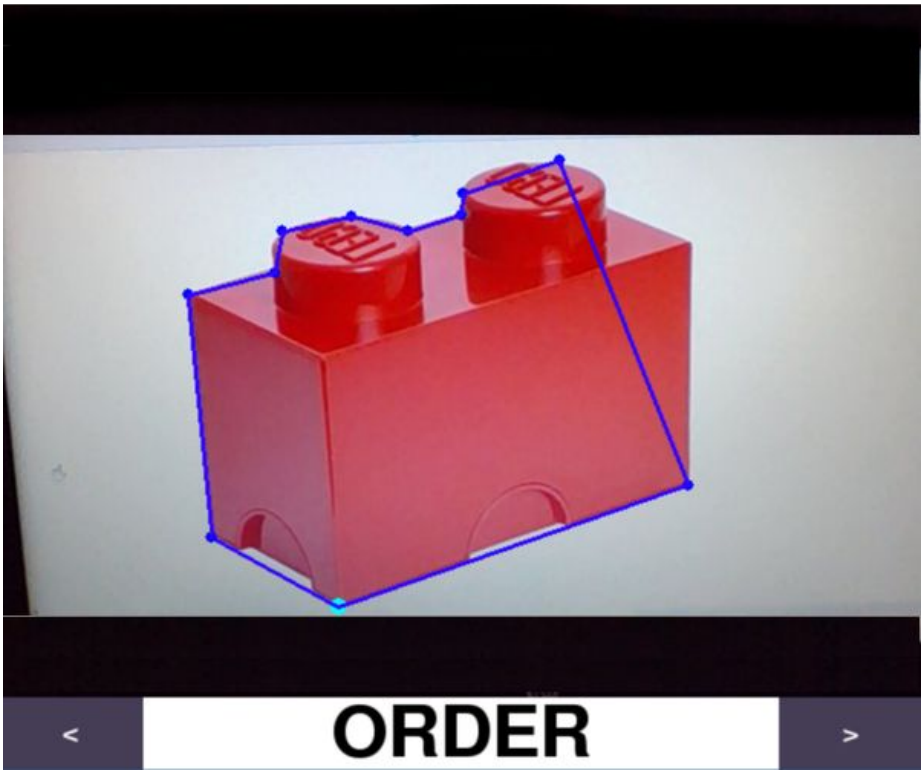
The project is split into two separate parts with the first being ‘Polygon Segmentation’. In this section the AI and the user collaborate to segment the desired part of the machine. Prior to the segmentation the user would capture the camera feed to center the desired part for the AI to segment.

AI Segmentation

After capturing the video feed, the user runs a program, called ‘PolyRNN’, on a Docker container, containing a TensorFlow environment. This program was trained on a TensorFlow database that included the parts located in the machine so that is how the AI knows how to correctly segment the parts

Example:

In this diagram you can see that the AI has just finished segmenting the part incorrectly which leaves the user able to fix the segmentation by navigating through the program

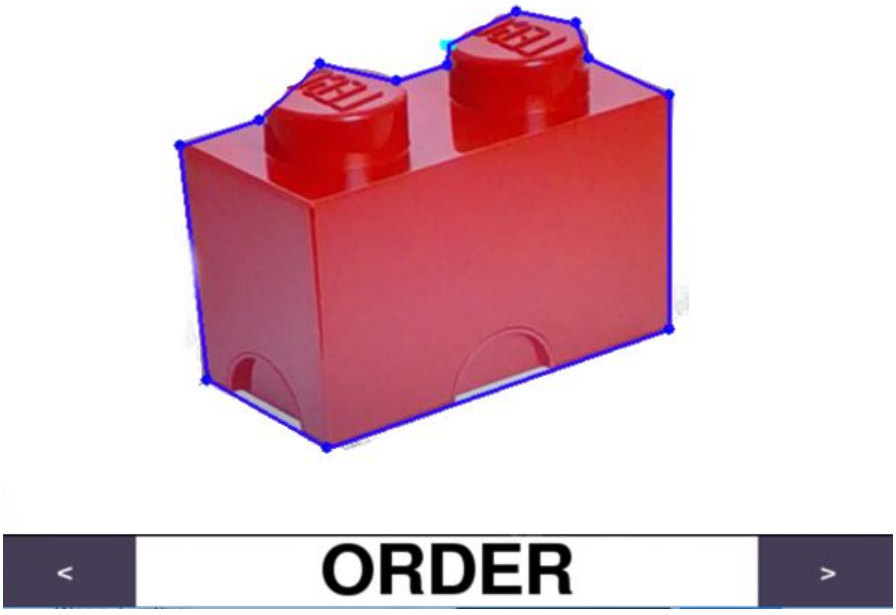


Human Segmentation

After the AI segments the image the results will be displayed on the screen in a pygame window. This window will allow for the user to either confirm that the AI has successfully segmented the part properly, or to fix the segmentation that the AI did incorrectly. If it comes to the user having to change the segmentation there is a menu help screen that displays how the user can do so.

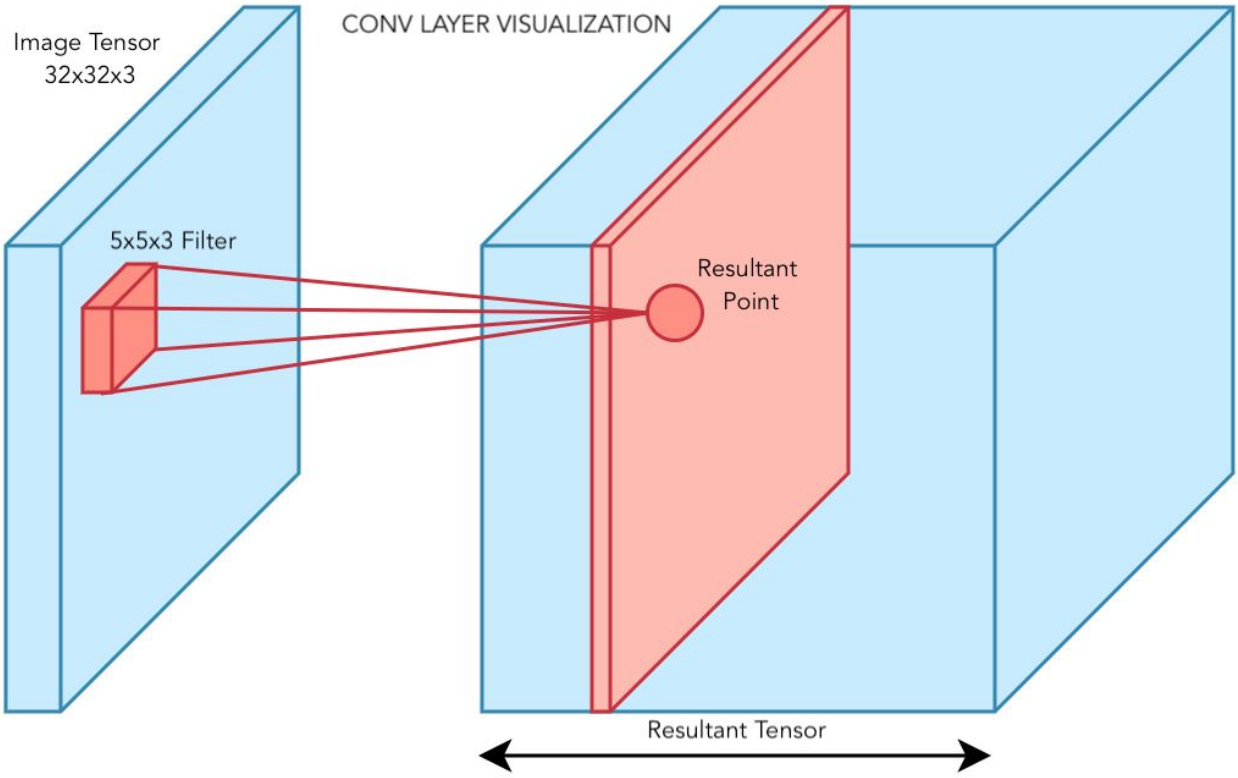
Example:

After the user has finished segmenting the part they can press the return key to confirm the segmentation to move onto the next step of part classification



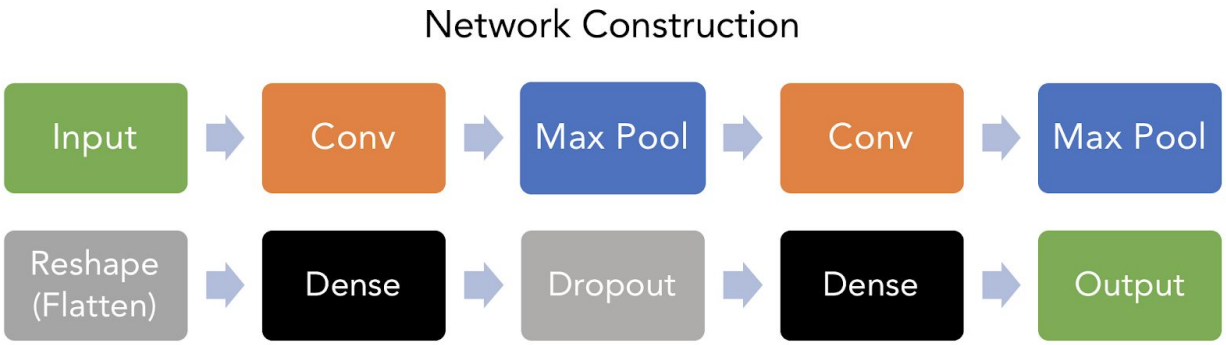
Part Classification

In order to classify each lego picture as one of the ten possible parts, we decided to design a convolutional neural network. This network would be comprised of convolutional (diagrammed below), max pooling, and dense (fully connected) layers.



TensorFlow Architecture

In order to build the network quickly, we decided to base it heavily on one of TensorFlow's public example networks (https://goo.gl/aQJadk). This network was built with TensorFlow's Estimator tool, which provides a clear and understandable way to construct deep learning networks. Our network follows the input function diagrammed below, and is trained using a TensorFlow gradient descent algorithm.



Training and Performance

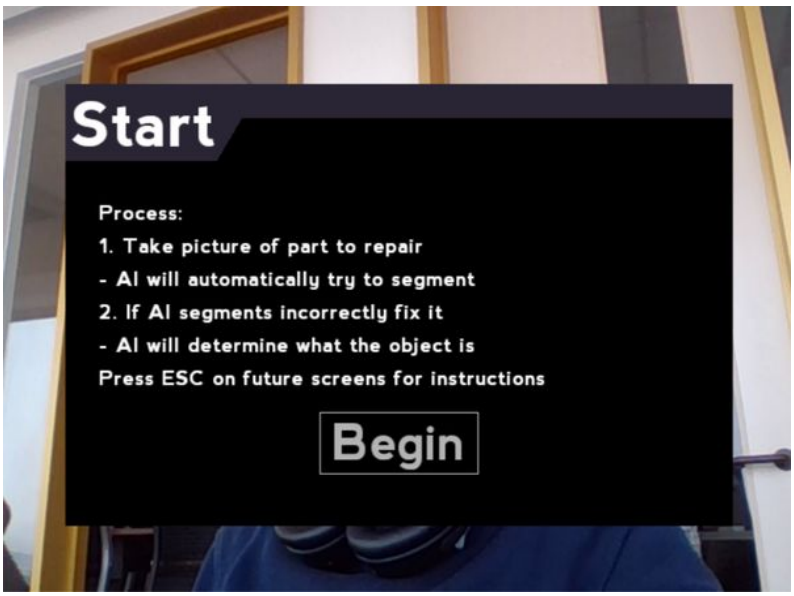
Training was a very straightforward process. We ran our training function on shuffled batches of 100 images. After running the function (which stops after 20000 steps) a few times we were able to get the network classifying out test data at around 92% accuracy.

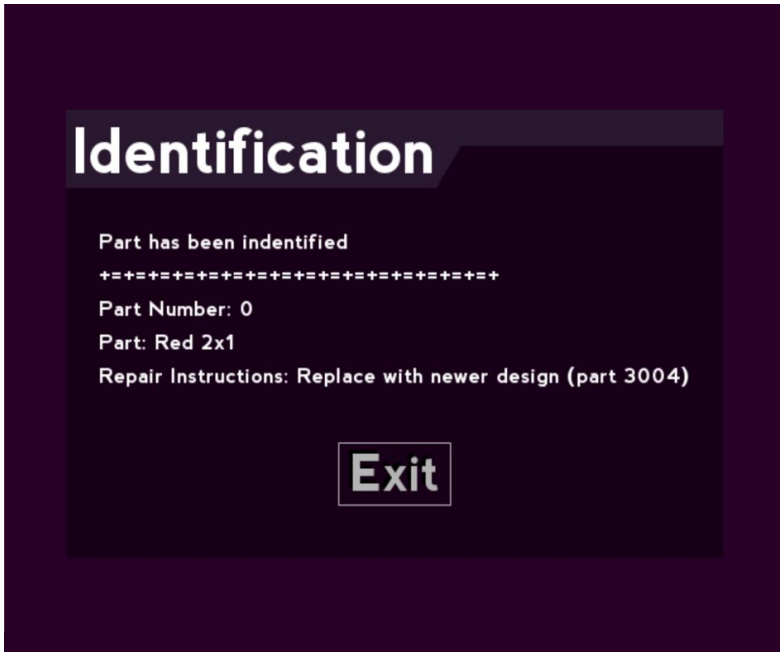
User Interface

This interface was created by using mostly a python library known commonly as pygame menu. The interface is easily able to be toggled on and off with the press of the “ESC” key which gives instructions to the user on how to use the program. Besides these instructions it is also used to show outputs and how the program works overall. Other from pygame menu the application also uses python to make usable arrows to switch between modes in the program.

Beginning

When the program begins it will automatically have a popup with the pygame-menu that states how the program will work as the user progresses through it.





Outputs

After the user confirms the cropped part then the AI will identify and output the part information onto an ending screen.

Help Menus

These menus are both toggleable and allow the user to check on how to use to program if they forget, while also not getting in the way of the efficiency of the application.