Debbie Cohen
Collab with Talia Leitner

Final Project Machine Learning
Computer Vision

The company: Hashuk Inventory Ltd
About: Hashuk makes small retail businesses supply chains come alive through accessible software combining procurement, inventory, and machine vision.

What do they do?

- Easy Inventory:
  - Increase employee efficiency
  - See your inventory the way you want to
  - Track expiration dates
- Supplier Wizardry:
  - Know when to raise prices by tracking supplier costs
  - One click shelf restock
  - Join group orders to seriously cut costs
- Analytics Magic:
  - Know which products are selling, and which ones aren't
  - Know where to place products in-store to maximize revenue
  - Know what products are popular in your area

What are we working on? Track expiration dates

Problem:
Printed expiration dates are everywhere and there is to communicate to retailers which goods are nearing the printed dates, but they consistently fail to do so.

The main problem is the lack of uniformity. Expiration dates have inconsistent, and sometimes multiple titles; they're chronically printed in hard to spot locations, with almost every manufacturer finding a new and different location to place the crucial information. Barcodes and invoices don't contain the information either, and suppliers as a whole don't provide it separately to their clients. The result is that stores with tens of thousands of individual products can't and don't track when products go bad or where they're located in the store. Hundreds of millions of metric tons of food go to waste in the United States each year in consequence, with improper spoilage tracking representing a significant portion of that figure.

Most retailers are stuck playing defense and damage control, manually combing through their stores and wasting billions of dollars annually and countless man hours every year to remove expiring goods. However, it does not need to be this way.

Advanced technology like machine vision is changing the way we look at product tracking. Businesses who seek to overcome these challenges need to leverage these technical solutions and ensure they come in organic and easy-to-use packages.

Task/Project Purpose:
Given pictures of expiration dates in products, read them and detect if the product is expired. This is to reduce employee workload and product spoilage.

Tools we are using:

- OpenCV in python. CV2
- Tesseract for python. Pytesseract. For character recognition
- Front end: html, JavaScript and tornado in python

Subtasks:
One of the issues with expiration dates that prevent AI/ML from solving the problem entirely is that machine vision simply does not work when there is nothing to read. Expiration dates can be rubbed out, printed on extremely reflective surfaces, or simply not well printed on the item. To improve this situation, we had to do some things:

- Image Preprocessing:
  - Images provided can be rotated, since the rotation angle vary, we wanted to create a model that predicts the rotation angle and rotates the image back. By doing this we would allow the object detection algorithm to detect the regions that contain the text.
  - Most of the expiration dates are written with a dot matrix font and were misrecognized by the object detector(east detection algorithm) as well as the text recognition. To fix this problem we tried to blur the image to connect the dots, change the image to grey scale and then sharpen it using image thresholding.
- Generate more data: the company didn't give us enough data to work with. Therefore we decided to expand our dataset by using the same images just rotated by different angles.
- Object Detection: for this part we used the OpenCV EAST text detector. The function is in the google collab, called east_detect and it takes an image as an input and returns a tuple with the image and the boxes surrounding the detected text.
- Text recognition: for this we used pytesseract and played with the different arguments to be able to turn the image into string. Ex: lang, config(--psm 1 --oem 1 --dpi 330…)

Some challenges we had while working on the project:

- Lack of data: the company didn't provide us with good data. They gave us only 60 images that were not labeled. From that very small dataset only a few of them had readable text(Ex: some of them had black font with black background), or were rotated. Without enough data it is impossible to train a model to predict rotation angle, how to we create a classification model with 180 classes(180 degrees) and at most 40 good images. Even if we generate data(rotating each image 10 times), we would only have 400 images which is not enough.
- How to preprocess the images
- The east detector detects many boxes and we want to put them together. If we loop through each box separately and display the text, they wont be in order. We must sort the boxed based on their coordinates in the image.

- After reading the text we need to figure out the meaning of it. Month-Day-Year or Year-Month-Day or Day-Month-Year, etc.
- cv2.imread(image) only reads png, jpg and JPEG. The same happens with the front end. Html doesn't display .HEIC uploads. HEIC is a file format commonly used to store photos on iOS devices.

Because of the lack of time and many roadblocks we had we decided to simplify the problem. We decided to assume that the input images are not rotated.

Machine Learning Models:

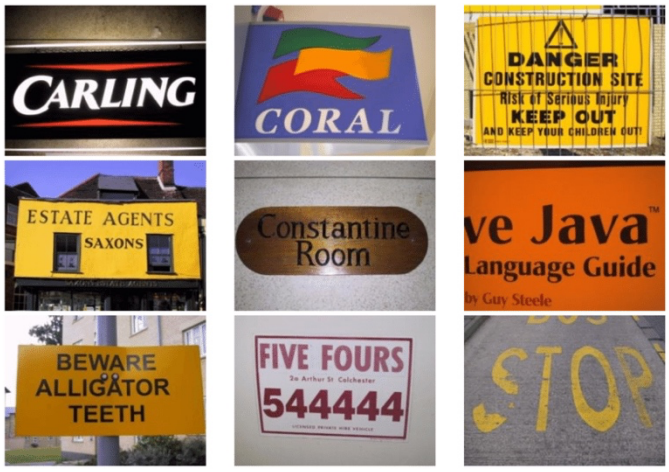
Text Detection: OpenCV EAST model

The EAST model is a deep learning model that detects the text in the image and creates and bounding box around the portion of the image having text. Since it is only a text detection model, it is usually used in combination with any text recognition method.

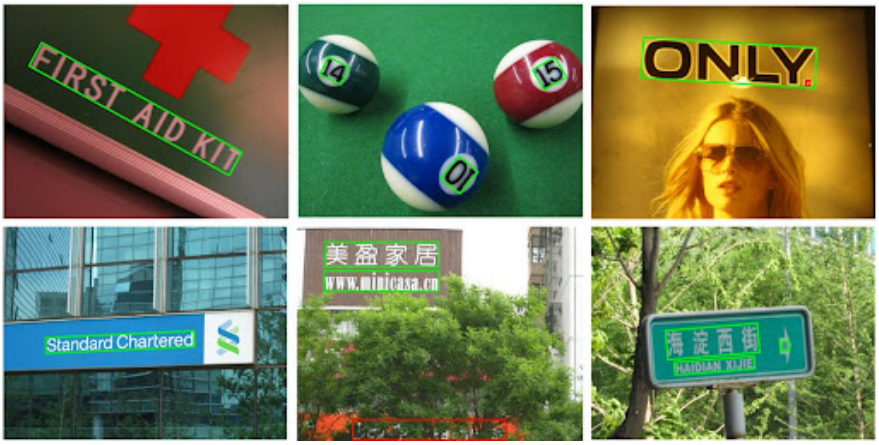
Text Detection Challenges: images are taken in a natural scene and these means that they have different lightning conditions, resolution, some are non-paper objects(some materials can be reflective), some are non-planar objects(text wrapped around a bottle)

Is called EAST because it is an Efficient and Accurate Scene Text detection pipeline. The pipeline was designed to directly predict words or text lines of arbitrary orientations in full images, eliminating unnecessary intermediate steps, with a single neural network

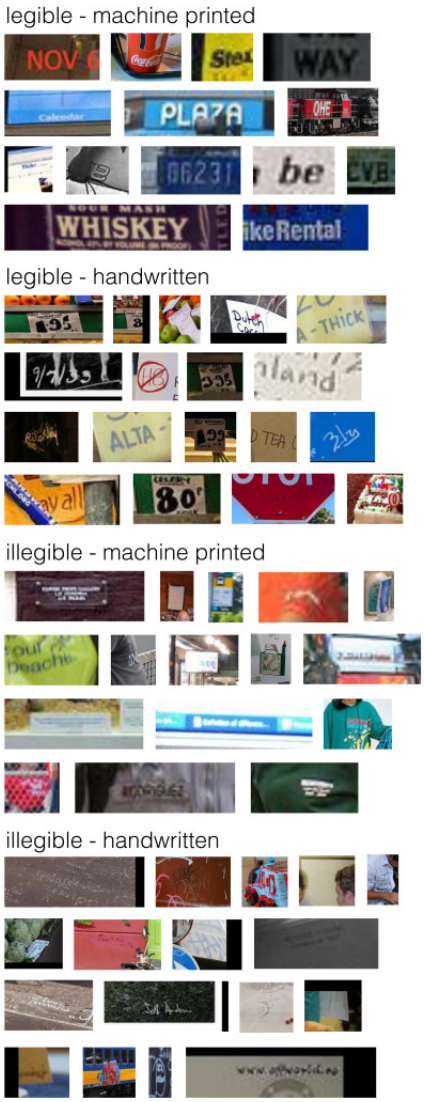
The model experiments on standard datasets including:
- ICDAR 2015: contains significant variabilities in terms of image quality of texts and writing styles. F-score of 0.7820



- MSRA-TD500: Text regions are arbitrarily. It contains both English and Chinese text. F-score of 0.7608



- COCO-Text: the dataset contains annotated each text region with an enclosing bounding box. F-score of 0.3945



Algorithm explained:
An image is fed into the Fully Convolutional Network(FCN) and then, the scores and geometry are generated.

The score stands for the confidence of the geometry shape predicted at the same location.

The function east_detect returns the image and an array with the boxes that contain text.

Loss Functions:
- For the scores: balanced cross-entropy loss function.
- For the geomery: IoU loss in the AABB part of RBOX regression, and a scale-normalized smoothed-L1 loss for QUAD regression.

Training:
The network is trained using an ADAM optimizer. To speed up learning, we uniformly sample 512x512 crops from images to form a minibatch of size 24. Learning rate of ADAM starts from 1e-3, decays to one tenth every 27300 minibatches, and stops at 1e-5. The network is trained until performance stops improving.

Text Recogniton: Tesseract

It is an Optical Character Recognition(OCR) engine that uses a neural network system. The tesseract package is used to recognize text in the bounding boxes detected for the text. It is important to note that Tesseract normally requires a clear image for working well. Therefore it is important to manage well the processing of the images. We used pytesseract that is a wrapper for Python.

Notebook with extra code snippets in this link:
Note: this notebook contains code for some things we tried but didn't work. For example: creation of a model to predict angle of rotation in images.
https://colab.research.google.com/drive/1UNQbLT0dGQISggLBQ56AfY41iCLhyWlK#scrollTo=9pQxuZz5DkOo

GitHub Repository:
https://github.com/debbiecohen/MLProject

References:

- EAST: An Efficient and Accurate Scene Text Detector: https://arxiv.org/pdf/1704.03155.pdf
- https://www.pyimagesearch.com/2018/08/20/opencv-text-detection-east-text-detector/