The Shelf Detector System For Retail Stores Using Object Detection

Introduction

In today's highly competitive retail landscape, being able to control and optimize retail execution at the point of sale has never been more critical.

Audit solutions today are manual and time-consuming. Auditing takes approximately 15 minutes per product category, involving physical measurements that are prone to human errors and inaccuracies. It is an expensive solution — companies can spend as much as $12 million annually in a single market to employ a sizeable salesforce to undertake these audits.However, only a handful of the traditionally conservative FMCG players have embraced progressive technology to achieve this.

Much of this growth can be attributed to the Internet of Things, whereby objects are brought online as digital assets.While purchase patterns can be obtained from the cash registers in relative ease, measuring the shelving execution standards is a much more complicated task. While the retail market is huge, the industry's ability to grow and compete has been challenged by infrastructure, increased competition, and most importantly, the absence of effective tracking and analysis tools of retail execution in the stores.

When we ask questions about size, flavor, geographies, demographics, temperature, and so on, we understand how much intelligence resides in the supply chain of retail goods. What makes a brand more or less successful comes from its ability to turn all that information into actionable knowledge. So how do we do this?

Machine Learning & Big Data in Retail

It stands to reason that if you desire more knowledge, one would follow the larger sources of market information. These enormous pools are known as Big Data and they a can be obtained in various methods. Measuring the shelving execution standards is a much more complicated task.

With image recognition technology, manufacturers and retailers can now understand the marketplace and react in real-time. Using image recognition can result in up to 60 percent of audit time saved in stores and can present a 98 percent auditing accuracy — this more often than not a 20 percent accuracy improvement from manual auditing accuracy level.

This means you will have accurate and reliable data on your distribution, know which items are out of stock and the share of your products within the category as well as a wealth of other actionable insights at your fingertips.

Automatic Stock Replenishment

Shelf-out-of-stock is one of the leading motivations of technology innovation in the shelf of the future. Traditionally, for Stock Replenishment visual check of shelves is performed by human employees, a task known as planogram compliance monitoring. Using image recognition technologies, in-store cameras can automatically recognize items on shelves, and assist humans in this tedious process.

Automatic item recognition also helps for omnichannel retail strategies: with real-time shelf monitoring, the store can be used as a warehouse for online orders.

This innovation matters because it takes human error and processing scalability out of the equation. When you are selling billions of products around the world, even a small percentage of variance has a significant impact.

Image Recognition Identify void spaces on the shelf and accurately detect and count front-facing products at category, range, brand or SKU level.

It has the immense potential to solve the problem of shelf optimisation, reduce data capture time, measure in-store execution quality of merchandisers in real-time and more importantly, help the same merchandisers correct their mistakes while they are still at the store.

Value: inventory distortion costs globally nearly $1.1 Trillion. On average, each store could increase sales by 7.5 percent, if it could completely fix this problem.

Current Offerings

Amazon Go

Amazon Go uses cameras disseminated all around the store, with real-time video analysis by artificial intelligence. Learning from Amazon Go Retailers can eliminate checkout by progressively increasing their use of AI in video. Moreover, each step improves their revenue and customer experience.

Stores can use AI cameras for merchandising placement, shelf monitoring, and shoplifting detection. This will prepare them for a smooth checkout-free transition.

RELEX Solutions

With their solution Stock levels are optimized based on the priority or role of each SKU to maximize sales and shelf availability, while minimizing inventory and waste.

Integration with planograms ensures attractive displays by keeping shelves well-stocked and sends alerts when space/demand mismatches occur.

CribMaster

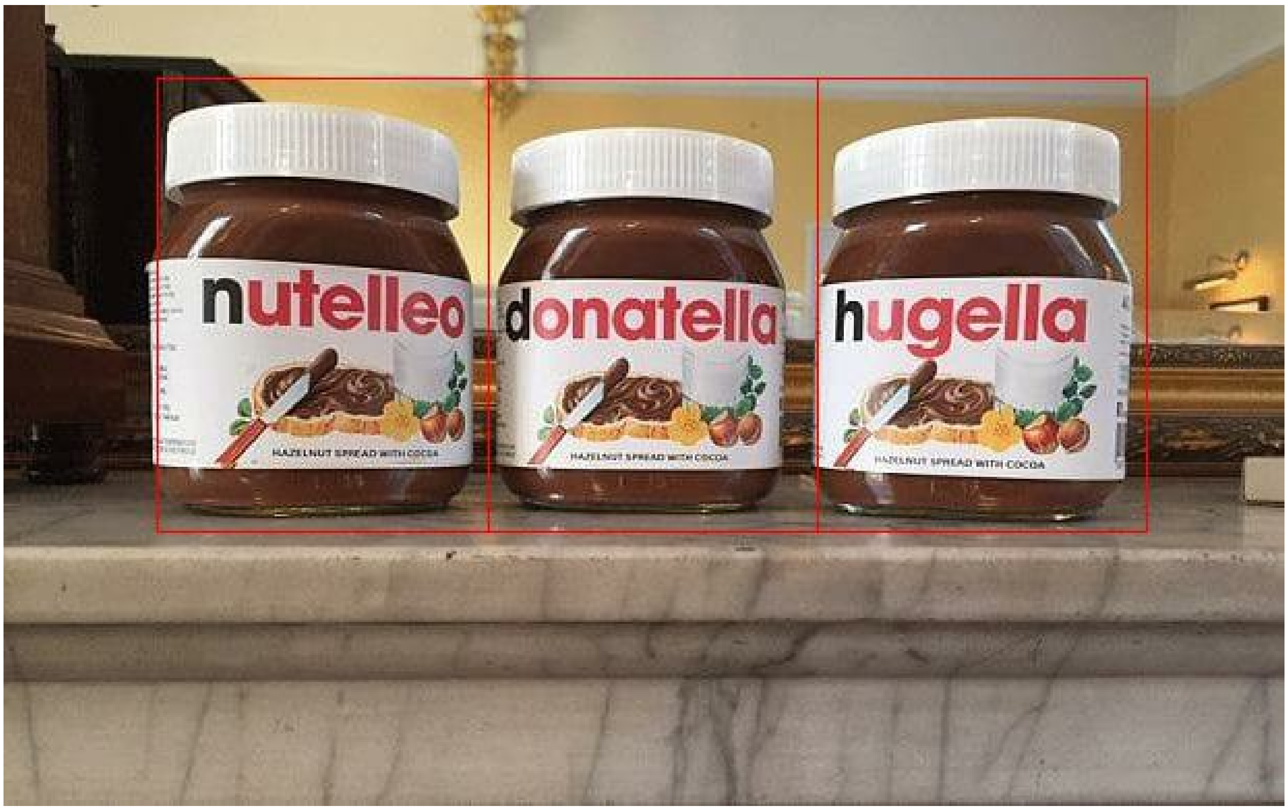
CribMaster's Automated Replenishment offering automates replenishment and purchasing as soon as stock reaches its defined minimum level. It also helps in Preventing stock-outs, overstock and dead stock & Increasing inventory turns

The Setup

Image of the shelf can be captured via Surviliance camera and real-time reports can be pushed into the POS (Point of Sale) of merchandiser within few minutes of the data capture. These reports can cover critical areas like shelf share, market operating price of self and competition, scores of stores, out of stock incidence of core SKUs as well as POSMs.

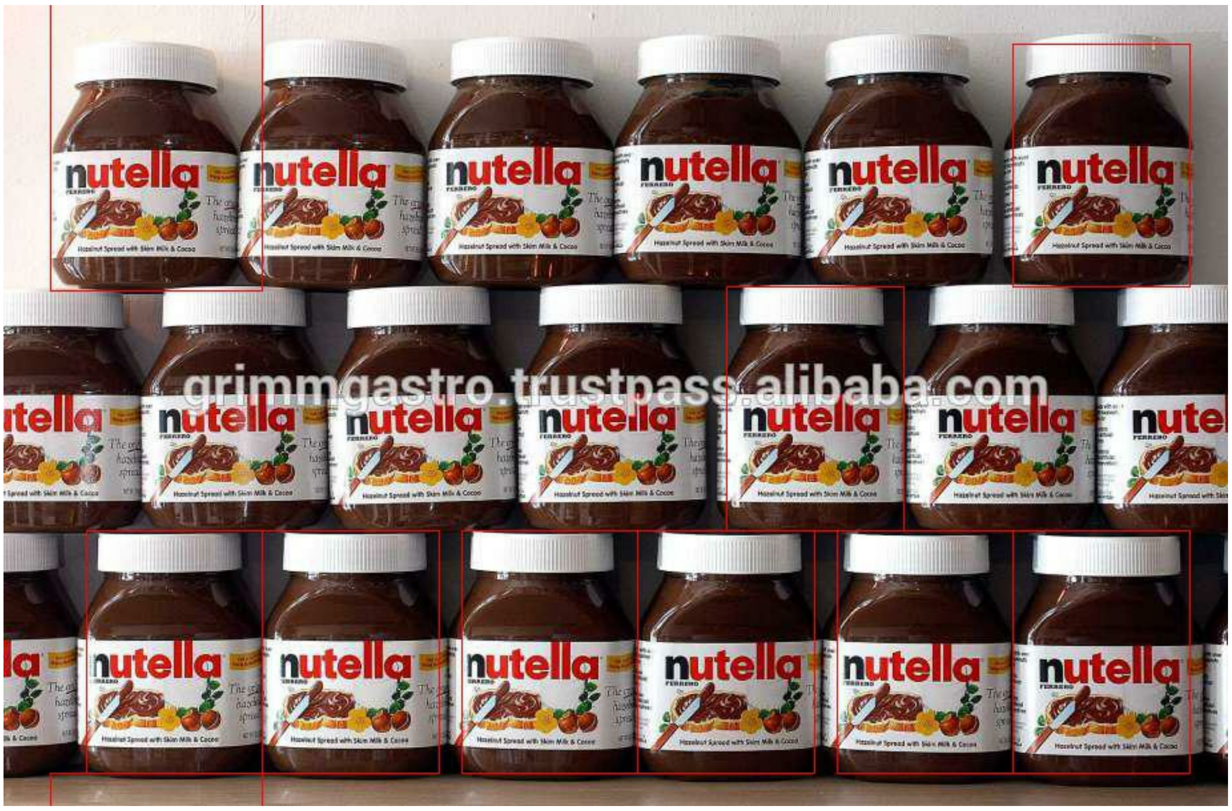
It can help automatically recognizes products from standard images taken in-store at scale, with high precision and fast detection results.

It can help find logos and identify individual product and their types for even greater business value.


nutelleo
donatella
hugella
HAZELNUT SPREAD WITH COCOA
HAZELNUT SPREAD WITH COCOA
HAZELNUT SPREAD WITH COCOA

The Proof Of Concept

For the POC we took image examples of Nutella Jars kept on shelf from google and trained out program with 10 images. The example shows how to dlib to perform object detector for semi-rigid object. In particular, we go though the steps to train the kind of sliding window object detector first published by Dalal and Triggs in 2005 in the paper Histograms of Oriented Gradients for Human Detection.



The code

First we initialize the libraries

```
import os
import sys
import glob

import dlib
from skimage import io
from skimage.draw import polygon_perimeter
import traceback
```

In this example we are going to train a object detector based on the small objects dataset in the train directory. The train_simple_object_detector() function has a bunch of options, all of which come with reasonable default values. The next few lines goes over some of these options.

```
options =
dlib.simple_object_detector_training_options()
```

Since objects are left/right symmetric we can tell the trainer to train a symmetric detector. This helps it get the most value out of the training data. options.add_left_right_image_flips = True

The trainer is a kind of support vector machine and therefore has the usual SVM C parameter.

```
options.C = 5
```

We tell the code how many CPU cores your computer has for the fastest training.

```
options.num_threads = 4
options.be_verbose = True
```

This function does the actual training. It will save the final detector to detector.svm. The input is an XML file that lists the images in the training dataset and also contains the positions of the object boxes.

To create your own XML files you can use the imglab tool which can be found in the tools/imglab folder. It is a simple graphical tool for labeling objects in images with boxes. For this example, we just use the training.xml file in the test folder.

```
if not os.path.exists(detector_svm):

dlib.train_simple_object_detector(training_xml_path,
detector_svm, options)
```

Now that we have a object detector we can test it. The first statement tests it on the training data. It will print(the precision, recall, and then)average precision.

```
print("")  # Print blank line to create gap from
previous output
print("Training accuracy: {}".format(

dlib.test_simple_object_detector(training_xml_path,
detector_svm)))
```

However, to get an idea if it really worked without overfitting we need to run it on images it wasn't trained on.

```
print("Testing accuracy: {}".format(

dlib.test_simple_object_detector(testing_xml_path,
detector_svm)))
```

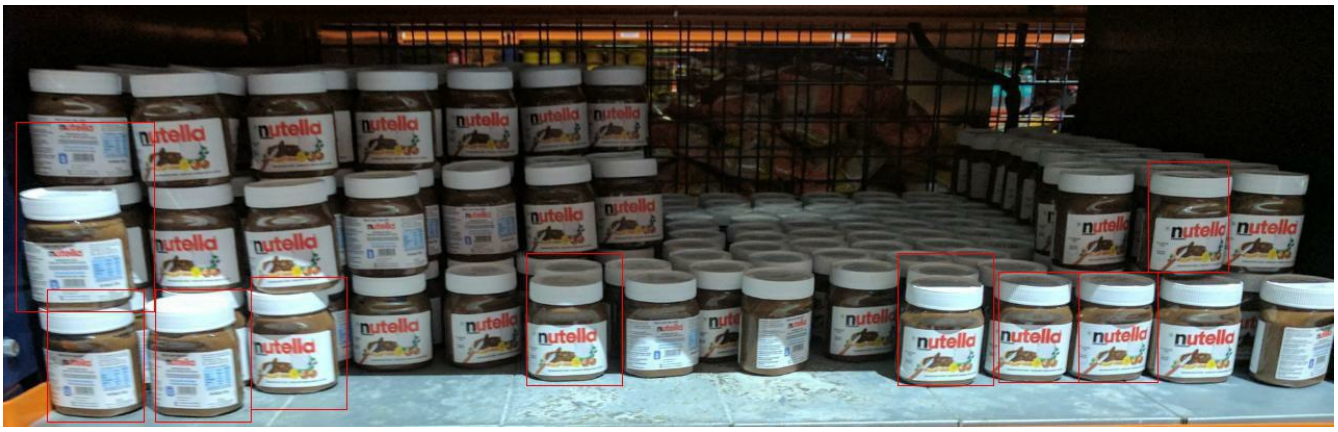
Now let's use the detector as you would in a normal application. First we will load it from disk.

```
detector = dlib.simple_object_detector(detector_svm)
```

Now let's run the detector over the images in the objects folder and display the results.print("Showing detections on the images in the test folder...")

```
win = dlib.image_window()
for f in glob.glob(os.path.join(detector_folder,
"test/*.jpg")):
    print("Processing file: {}".format(f))
    img = io.imread(f)
    dets = detector(img)
    print("Number of objects detected:
{}".format(len(dets)))
    bOverLays = False
    for k, d in enumerate(dets):
        print("Detection {}: Left: {} Top: {} Right:
{} Bottom: {}".format(
            k, d.left(), d.top(), d.right(),
d.bottom()))
        rr,cc = polygon_perimeter([d.top(), d.top(),
d.bottom(), d.bottom()],
                             [d.right(), d.left(),
d.left(), d.right()])
        try:
            img[rr, cc] = (255, 0, 0)
            if bOverLays == False:
                bOverLays = True
        except:
            traceback.print_exc()
    # Save the image detections to a file for future
review.
    if bOverLays == True:
        io.imsave(f.replace("test/","output/"), img)
    win.clear_overlay()
    win.set_image(img)
    win.add_overlay(dets)
    dlib.hit_enter_to_continue()
```

Conclusion



Unlike other object detectors in OpenCV or other libraries the number of training images used in dlib were less and still teh results were good, depending on the quality of image and number of objects and other considerations like blurring and object placement we got nutella jar detected in almost 88% of images and in almost 70% of the images where jars where detected the number of jars detected was good. And we got these results despite the fact that the images used for training and testing were diverse and did not represent the same environment. The good aprt of Dlib library is that it can even run on small devices like Android phones and even Rapsberry Pis which makes object detection solutions on dlib handy and usable.

Reference:

1. Automated Replenishment | CribMaster
2. Automatic Replenishment System | RELEX Solutions
3. LogoGrab are the leaders in logo recognition technology – LogoGrab
4. Why image recognition is a game changer in retail
5. dit & Image Recognition
6. How retailers can eliminate checkout, like Amazon Go. And increase profits in the process.
7. dlib C++ Library