Project Stage 2

Poojan Patel, Sourav Choudhury, Rinita Nair, Benjamin Powell, Rahul Patel

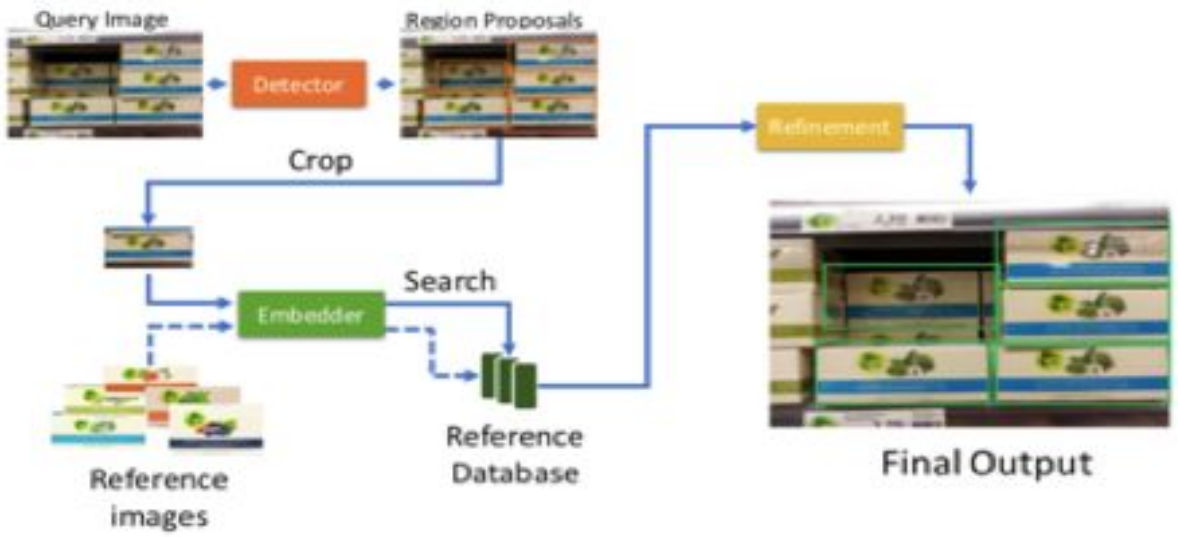
Relevant Academic Work

Recognition of products in store shelves have unusual challenges. First would be the number of products that need detection which might range from hundreds to thousands or even more depending on the size of the retail unit. Second would be the database which generally has very few detailed images of the products, also at the time of detection the products are needed to be detected from images where only a specific portion of the product is at display in shelves. Thirdly, the shelves and the products kept in them keep on changing at a relatively fast basis.

The product recognition is more of a hard instance recognition problem, rather than a classification one as it primarily deals with a lot of products looking similar in most ways but different in just a few specific ways (eg: different flavors of the same brand of cereals).

Thus, to tackle the above mentioned issues, product recognition is proposed by a pipeline of three different stages. Firstly, for a shelf image, we perform a class agnostic object detection to extract region proposals enclosing the individual product items. This stage relies on a deep learning object detector trained to localize product items within images taken in the store; we will refer to this network as to the **Detector**. In the second stage, we perform product recognition separately on each of the regions provided by the detector, then a K-NN similarity search is carried out. A convolutional neural network is trained using reference images to learn an image embedding function that maps the RGC inputs to the n-dimensional global descriptors -- this is referred to as the embedder. The next challenge is to prune out false detections and refine the disambiguation between similar looking products we refine the recognition output by re-arranging the first K-proposals delivered by similarity search.

Another thing worthy of mention is that this approach requires samples of annotated in-store aisle images only to train the model, which doesn't require product specific labels but just bounding boxes drawn around items. This system while being computationally heavy during training is light once the model is trained, thus it is fast at deployment easing real time operations easily.



Implementation Details

For image detection, YOLO has been used as the state of the art one stage object detector. YOLO was chosen as it provides real time object detection on a GPU. The backbone of the network is the VGG_16 embedder which is pre-trained on the ImageNet 1000 classification task. From this network we obtain the global image descriptors by computing MAC features on the convolutional layer and L2 normalization is applied to obtain unit norm embedding vectors.

To enrich the dataset and create the anchor images, augmentation functions are performed randomly A: blur by a Gaussian kernel with random σ, random crop, ran- dom brightness and saturation changes. This augmentations were engineered so to transform the images in a way that renders them similar to the proposals cropped from the *query* images. The hyper-parameters obtained by cross validation for the training process are as follows: α = 0.1 for the triplet loss, learning rate lr = 0.000001, ADAM op- timizer and fine-tuning by 10000 steps with batch size 24.

We propose a novel kind of local features for the *+lf* refinement: as *MAC* descriptors are obtained by applying a max-pool operation over all the activations of a convolutional layer, by changing the size and stride of the pool operation it is possible to obtain a set of local descriptor with the associated location being the center of the pooled area re-projected into the original image. By leveraging on this intuition, we can obtain in a single forward computation both a global descriptor for the initial K-NN search as well as a set of local features to be deployed in the refinement step (subsection 3.3). For our test we choose kernel size equal 16 and stride equals 2 as to have 64 features per image.

Other Solutions

This approach proposes a weakly supervised method using two algorithms to predict object bounding boxes given on an image dataset. The first algorithm is a full convolutional neural network trained to classify object instances. The FCN output mask is enhanced into final output by the Convolutional Encoder Decoder. To perform object detection tasks, two things are to be done at random a method which uses just object instances and not only learns to recognize the instances, but also localize them. The solution proposes to do object localization using the same network using the same network which is trained for the task of classification on object instances. A fully convolutional Network classifier is used, there are no fully connected dense layers so a higher resolution input can be passed to the network to get a corresponding output mask. Another optional network is used to enhance the outputs from FCN, for this an encoder-decoder framework is used which is called ConvAE. The output of the ConvAE is converted to obtain bounding boxes.

Method

Weakly supervised learning is a method of learning in which the supervision can be indirect, inexact and incomplete. In this setting, one doesn't have access to supervised data but some other form of labeled data which is useful.

During training task, we perform classification on instances of the objects. The network architecture of the FCN is as follows. We take the pretrained layers of VGG11 from torch-vision and remove all the FC Layers. We then add a single convolutional layer at the end with appropriate kernel size so as to get a Nx1x1 output, where N is number of classes. One of the core advantages of our method is that FCNs are independent of input size. We take the valid assumption that the object instances (training images) are smaller in size than testing images (images containing multiple objects). Hence, the FCN which is designed to do a classification task will give a corresponding output mask when a higher resolution input is passed.In the testing task, we pass an image pyramid (consisting of various downscales of the image) to the same network. We use downscale factor of 1.5 to generate the pyramid. We resize the corresponding outputs from each level of pyramid to the original image size. The mean of the pyramid outputs are then taken as the final output.

In the datasets of product recognition, it often happens that there are very similar products which belong to different classes. This tends to make the classifier's (FCN) output probability of the correct class a little weaker. Here, the confidence score of the correct class isn't as high as we'd like it to be. In such cases, one can find reasonable confidence score in other classes even though the FCN predicted the correct class. Hence, this induces a lot of false-positives de- pending on the threshold of converting the segmentation into a binary mask.

The ConvAE works great in eliminating these false positives. It is able to learn the prominent class and remove the false positives from other classes easily. Hence, we don't have to worry about the threshold. We note that it's not possible to train the ConvAE if our classifier isn't strong. The false positives make it very difficult for the network to learn the refinement task.

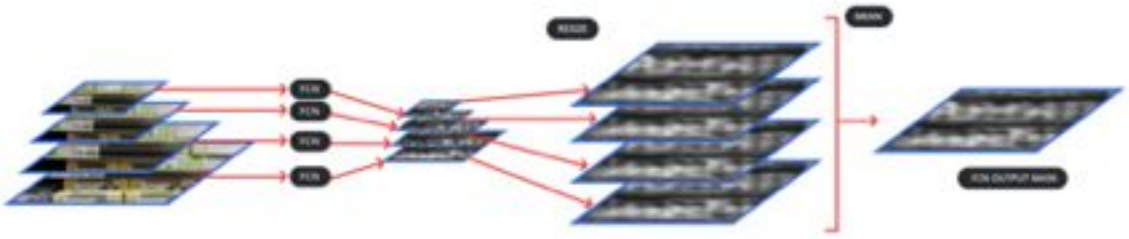


Figure 1: Object detection pipeline on shelf images

References:

[1] Recognizing Groceries *in situ* Using *in vitro* Training Data
http://www.michelemerler.com/papers/grozi_cvprw07.pdf

[2]A deep learning pipeline for product recognition on store shelveshttps://arxiv.org/abs/1810.01733

[3] Retail Shelf Analytics Through Image Processing and Deep Learning
http://tesi.cab.unipd.it/62145/1/DeBiasioAlviseTesiMagistrale.pdf

[4] Product recognition on the shelves using neural networks using Keras and Tensorflow Object Detectionhttps://super-geek-news.github.io/articles/416123/index.html

Business/Customers

In massive retail stores like Lowe's, it becomes difficult for a salesperson to keep track of all the products in the aisle and re-stock those products every time a particular product goes out of stock an even to improve the way the shelves can be organized. Such stores require large manpower to handle such issue. With the help of our Computer vision project with the help of the security camera

or using the mobile phone's camera one can keep track of the products in the aisle. It helps to identify which products are placed in a particular aisle and segment the image by adding labels to individual product. This project can be enhanced further to incorporate trigger events which will help the store managers to know that a particular item is out of stock in an aisle and need to restock those items in the aisle. Thus increasing the Key Performance Indicator of that store. This project can be useful to other retail stores like Walmart, Target, Tesco and many other retail businesses.

The goal of our project is to correctly identify Lowe's products from an image of a shelf in-store. This project will help understand the difference between the way the products are stocked and the way they actually are in-store. This can help with decisions in the future to replace missing items, get a better idea of productivity by shelf space, correct improper stocking methods, and ultimately create a better experience for our customers.

The main idea behind this project is to get relevant context from the given image. With the help of Open Source tools based on Object Detection and Optical Character Recognition we need to form bounding boxes and obtain product information like the product name, price, where exactly is the item place in the aisle, number of facings and many others.

Open Source

For this project, we will be identifying products based on their visual characteristics. Visual characteristics will differ depending on the type of product that we are classifying, therefore, traditional object detection will not always work-- and neither will methods for identifying text within an image. To approach this problem, Lowes has suggested that we utilize a two-pronged approach with a combination of object detection and optical character recognition in tandem.

Lowes has provided us with a dataset that contains labeled, clean images of products that they will want us to train our models with. They have provided us with an abundance of data, however, for testing our algorithm we are likely going to have to acquire image data in person from a Lowes store after ensuring that we have permission from our industry partners and the local store that we are acquiring our testing data from.

Object Detection:

YOLO (You Only Look Once) was suggested as a starting point for classifying products using object detection alone. There are multiple open source repositories containing code for training and applying this model, however, we will be focusing on Python repositories since that is what we use in class and what Lowes uses in house for data science. For implementing this in Python, there is an open source repository located here under the project name "Darkflow" that is an adaptation of Darknet (written in C) to TensforFlow for use in Python.

YOLO Documentation:

[1] https://pjreddie.com/darknet/yolo/

[2] https://pjreddie.com/media/files/papers/YOLOv3.pdf

YOLO Tutorials:

[1] https://www.youtube.com/watch?v=h56M5iUVgGs

[2] https://www.youtube.com/watch?v=4eIBisqx9_g

[3] https://www.youtube.com/watch?v=Y73SWT79Rck

Tutorials for Darkflow:

[1] https://www.youtube.com/watch?v=PyjBd7IDYZs

[2]https://medium.com/coinmonks/detecting-custom-objects-in-images-video-using-yolo-with-darkflow-1ff119fa002f

[3]https://towardsdatascience.com/yolov2-object-detection-using-darkflow-83db6aa5cf5f

GitHub for Darkflow:

[1] https://github.com/thtrieu/darkflow

Optical Character Recognition (OCR):

To detect text in product labels and barcodes, we can use PyTesseract which is an open-source implementation of Optical Character Recognition. This step will be necessary when we will be dealing with products that are technically the same, but differ in their product labeling. It is also useful in other circumstances when it is not completely feasible to differentiate a product's class on first glance visual inspection alone.

Documentation for PyTesseract:

[1] https://pypi.org/project/pytesseract/

Tutorials for PyTesseract:

[1] https://stackabuse.com/pytesseract-simple-python-optical-character-recognition/

[2] https://realpython.com/setting-up-a-simple-ocr-server/

[3] https://www.youtube.com/watch?v=kxHp5ng6Rgw

GitHub for PyTesseract:

[1] https://pypi.org/project/pytesseract/

Notes on Open Source Code:

It is possible that these resources will not be enough depending on the depth of the project. We also might want to switch to YOLO v3, which it does not look like Darkflow currently supports. There are other repositories that contain Python wrappers for Darknet YOLO v3 and implementations using PyTorch, etc.

Lowes has suggested that in order to get the best performance, using a “canned” model will not be sufficient. This means that we will likely have to find clever ways to combine these approaches or utilize individual components that each one uses to create our own model.

Industry Solutions

The best example for this kind of project was developed by Amazon called Amazon Go. With the help of deep learning they are able to detect which products are are taken or returned and are able to add these product details in the virtual shopping cart. The Amazon GO app uses QR code that the user has to scan before entering. With the help of this Amazon can keep track of the items that the user is buying. They would take photos when people enter the store, when they removed items from a shelf, and when they left with items in their hands. There is also a mention of "facial recognition" and user information, which may include images of the user, details about the user like height and weight, user biometrics, a username and password, even user purchase history.Amazon has successfully implemented deep learning to retail stores. They have retail stores in Seattle, San Francisco and Chicago.

Even small companies like Planorama now under Trax developed an application called PlanoCheck which is using similar image recognition algorithms with deep learning that help retail stores to increase their sales. The sales representative click an image of the store aisle and then upload this image to their cloud then PlanoCheck performs analysis and recognizes the products with 98% accuracy and sends a feedback in the form of KPIs(Key Performance Indicator). This feedback contains information which products are out of stock and other information. The retail businesses can keep track of all this information from every store and help improve their sales. Other companies like Vince and Infrrd are also providing solutions related to Image recognition.

Reference:

[1] https://planorama.com/
[2] https://www.youtube.com/watch?v=xd23ivMz9w8
[3] Amazon go Patent :
http://appft.uspto.gov/netacgi/nph-Parser?Sect1=PTO2&Sect2=HITOFF&u=%2Fnetahtml%2FPTO%2Fsearch-adv.html&r=1&p=1&f=G&l=50&d=PG01&S1=(Steven.IN.+AND+Kessel.IN.)&OS=IN/Steven+and+IN/Kessel&RS=(IN/Steven+AND+IN/Kessel)
[4]https://www.pocket-lint.com/phones/news/amazon/139650-what-is-amazon-go-where-is-it-and-how-does-it-work
[5] https://www.youtube.com/watch?v=KyK-2u6X4tM&feature=youtu.be