Multiple Object Instance Recognition

Shubham Kumar Bharti (15807702)
Sunny Kant(15817740)

5th February 2019

Abstract

Given a dataset of images of several objects, the job is to learn a model that could retrieve all the instances of a given test object. More specifically the given dataset consists of images of several objects and multiple instances of each object. When tested on an image/object our model's task would be to retrieve all instances of the image in the dataset. In other words our model would give a similarity ranking on all images in the dataset such that instances of the object being tested have the highest rankings. In training data, each image contains only one object but in test data images will contain multiple objects and possibly with different backgrounds. The model would be able to retrieve an image if any one of the object present in the query image matches with the object present in that image.

1 Introduction

The problem of image recognition is a very well known one where the task is to recognize the objects of a particular class in a given image. In pre Deep Learning era, researchers relied mostly on feature extraction based methods like SIFT, SURF, HOGS etc. which followed a similar paradigm to generate a set of representative feature vectors of image instances and then use methods like Clustering, Nearest Neighbour, K-Means etc to find the similarity between feature vectors of the images. Though these methods worked quite better when compared to simple feature based methods that relied on manual feature constructions, they were not able to perform beyond a particular accuracy measure. Also, the feature based methods struggled with variations in lightening conditions of input images and were slow at inference time. With the advent of high computing resources and availability of large data corpus, Deep Learning took a major leap in improving several vision related tasks.

2 Object Detection Methods

We have tried several models both of pre-deep learning era models and deep learning models and checked the accuracy on all the models. At last, we got better results in case of Faster R-CNN. The models we tried are as follows

2.1 Feature extraction based models

Classically, in feature extraction based methods, an image is represented by the key points extracted from the image. The key points are usually extracted using edge, corner or other blob detection techniques. By comparing the key points of an query image with the key points of dataset image, we can deduce if common information are present in both images and then use some metrics to represent the similarity of images based on extracted key points. Some of the popular feature based models like SIFT/HOGS have shown promising results in object detection task and are robust to the orientation, scaling and rotation of images.

- **SIFT (Scale-Invariant Feature Transform)**

  SIFT is a feature detection algorithm to detect and describe local features in images. The SIFT algorithm has 4 basic steps. First is to estimate a scale space extrema using the Difference of Gaussian (DoG). Secondly, a key point localization where the key point candidates are localized and refined by eliminating the low contrast points. Thirdly, a key point orientation assignment based on local image gradient and lastly a descriptor generator to compute the local image descriptor for each key point based on image gradient magnitude and orientation.

- **SURF (Speeded-Up Robust Features)** The main advantage of SURF is that it made the extraction of interest points much faster. Instead of approximating Laplacian of Gaussian(LoG) using Difference of Gaussian(DoG), SURF approximated LoG with Box filters. Since the calculation with box filters are easy to perform on integral images and this task could be parallelized for different scales, SURF enabled to extract the interest points in a faster way when compared to SIFT and thus improved the performance.

  Initially, we tried SIFT to extract features on the given dataset and employed soft clustering based methods to represent features vectors of the images. This method could not yield good results for multiple instance/hard test images and we decided to shift to Deep Learning based object detection models to improve the task.

2.2 Deep Learning models

With the recent surge in availability of large corpus of data and high computing resources like GPUs, Convolutional Neural Networks (CNNs) based Deep Learning models managed to outperform all previous feature based techniques on different Computer Vision problems like images recognition, images classifications, objects detections tasks etc. In early versions of Deep learning based Object Detection models people considered selecting every possible bounding box in some size range and classifying each of them for the presence of different class of objects. As it may seem true, these methods were very computationally expensive as they needed to perform operations over a large number of bounding boxes which blows up exponentially. A breakthrough in this direction came with the introduction of R-CNN model, by Girshick et al, which dramatically reduced the computation time by bounding the number of bounding boxes that are to be further investigated for presence of object instances.

- **R-CNN:** R-CNN introduced a Region Proposal Network that could suggest a set of just 2000 regions called region proposals, produced by a selective search algorithm, that are probable candidates for the presence of object instances. These 2000 candidate region proposals are warped into a square and fed into a Deep CNN network that produces a dense feature vector as output. The CNN acts as a feature extractor, yielding a 4096-dimensional vector that represents an image. Next, the extracted features are fed into an SVM to classify the presence of the object within that candidate region proposal. RCNN's were still very slow as one need to process 2000 region proposal for every single image. Also the selective search algorithm used in R-CNN was a fixed algorithm and no learning was involved to improve its accuracy and efficiency.

- **Fast R-CNN:** Fast R-CNN provided a performance improvement over R-CNN as we don't have to feed 2000 region proposals to the CNN per image, instead the convolution operation is done only once per image and a convolution feature map is generated from it. The region of proposals are identified using this feature map and then they are warped into squares and using a RoI pooling layer we reshape them into a fixed size so that it can be fed into a fully connected layer. The Fast R-CNN method performed much better than R-CNN as it did convolution operation only once over the entire image and further processing was done on convoluted feature map.

- **Faster R-CNN:** The region proposal networks on extracted feature map still proved to be a bottle neck in Fast R-CNN method. Instead of using selective search algorithm on the feature map to identify the region proposals, Faster R-CNN model proposed a separate network to predict the region proposals as selective search is slow and time consuming. Then predicted region proposals are reshaped using a RoI pooling layer which is used to classify the image within the proposed region. With the new region proposal network well embeded into the Neural Network architecture, Faster R-CNN could perform very fast on object detection task.

- **YOLO:** YOLO (You Only Look Once) is an object detection algorithm much different from the above region based algorithms as all the above algorithms use regions to localize the object within the image considering only parts of the image which have high probabilities of containing the object not the complete image, while in YOLO a single convolutional network predicts the bounding boxes and the class probabilities for these boxes and then bounding boxes having the class probability above a threshold value is selected and used to locate the object within the image. YOLO is faster and works better in many cases but struggles with small objects within the image. And in our case, many objects are small and so we got better results in case of Faster R-CNN than YOLO.

- **SSD:** SSD (Single Shot MultiBox Detector) detects objects in images using a single deep neural network. SSD, discretizes the output space of bounding boxes into a set of default boxes over different aspect ratios and scales per feature map location. At prediction time, the network generates scores for the presence of each object category in each default box and produces adjustments to the box to better match the object shape. Basically, SSD uses shallow layers to predict small objects and deeper layers to predict big objects, as small objects don't need bigger receptive fields and bigger receptive fields can be confusing for small objects.

We first implemented feature based methods like SIFT/SURF (using open-cv library) to extract key-point features of the images and then used clustering based comparisons to find the similarity between the images to report the occurrence of test object in train dataset. This method worked well for single object instances but performed very badly in instances where we had multiple objects in same images and hard images with cluttered backgrounds. We next moved to the deep learning methods and implemented fine-tuned Fast R-CNN (with resnet-50 as base architecture), Faster R-CNN model (with resnet-101 as base architecture), Single Shot Multibox(SSD) model (with Inception v2 as base architecture) pre-trained on Microsoft COCO dataset. We finally report file retrieval ranking list obtained using SSD Inception V2 model.

- **MD5 Hashed of SSD Inception V2 Model**

  Trained Model Graph: `4e4f79a5f957707c142643a9736d00c9`
  Model Checkpoint: `a87d7266237e1bf8ee7d0dfd217b1f33`

3 Data Processing

3.1 Basic Data Processing

We first, manually annotated the bounding boxes of a subset of the given dataset using labelImg toolkit and then bootstrapped the annotations to the entire corpus using object detection models trained on this subset of data. We also augmented the annotated dataset using a set of image transformations like rotation, scaling and translations.
The model trained using bootstrapped dataset could perform very well on single instance examples but it failed miserably in case of hard images where the objects were cluttered with background features. We applied following techniques to given dataset-

Data Annotation

For each given data, we annotate them by selecting a "zone" of the data, and adding a label to this specific zone. We have done this by "labelImg" image annotation tool available in python.

Data Augmentation

Rotation Using Scikit-Image, we rotated each given figure in angles of -30, -15, 15, 45 degrees and found the corresponding annotation by considering the annotation box of given figure and applying rotation for each vertex of box.

Scaling We used PIL library available in python to scale the image to half(both in width and length) and then fill the space with white colour to keep the length and width same across images of non-scaled dataset.

Translation Objects can be located at almost anywhere in the image. So, we translated given images to create new images with corresponding annotation box as translated images will force convolutional neural network to look everywhere.

Flip We flip images horizontally to created new images for each given image with new corresponding bounding box.

3.2 Advanced Data Processing

The given set of training samples are taken from BigBIRD dataset. To deal with the hard examples, we planned to generate our own set of hard examples and train our Faster R-CNN model on on a collective of bootstrapped and generated image dataset. We first collected object images and their mask of 16 given classes from BigBIRD dataset. Using the image mask, we generated over 10000 hard image samples of objects with random background images collected from internet. We finally trained our model on the basic and advanced datasets and got good improvements in results.

4 Major Libraries and Tools

We used the following libraries for the purpose mentioned by their side.

- `tensorflow` : for transfer learning of Fast(er) R-CNN method for our dataset
- `open-cv` : for image processing and SIFT/SURF feature extractions
- `skimage/pillow` : for image processing and data augmentaiton
- `labelImg` : for manual data annotations

5 References

- Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks
- Fast R-CNN
- You Only Look Once: Unified, Real-Time Object Detection
- SSD: Single Shot MultiBox Detector
- Cut, Paste and Learn: Surprisingly Easy Synthesis for Instance Detection