Object Detection using SIFT and Bag of Words

Akhil Nalamala
CS 682
George Mason University

***Abstract*—This is the writeup for the final project for the CS 682 Fall 2020 class 'Computer Vision'. I have implemented Object Detection using a Scale Invariant Feature Transform (SIFT) feature extractor and the Bag of Visual Words method. I have trained an Support Vector Machine (SVM) classifier on pictures containing my mountain bike ('positive' images) and pictures that don't contain my mountain bike ('negative' images), and then tested it on a set of new images.**

I. Introduction

In this project, I aimed to implement Object Detection, or to more accurately describe it, Image Classification, i.e. classify an image as 'positive' or 'negative' based on whether the image contained my mountain bike or not. This might not seem very complicated at first, considering how good the human brain is at instinctively performing this action, but making a computer capable of this (and building on top of it) has been at the forefront of Computer Vision research for a long time.

A. Related Work

Over the years, researchers have come up with a variety of ingenious ways to perform Object Detection and Image Classification. The general idea of an object detection algorithm usually follows the form of extracting important features from images and using those to learn how to detect other similar images. The following are some of the most notable works regarding the same:

- **R-CNN (Region-based Convolutional Neural Network)**: This algorithm utilizes, as the title suggests, a convolutional neural network in order to extract features from the image. The proposed idea here is that instead of performing feature extraction on every single region of the image, around 2000 regions are selected using a 'selective search', and only those regions are sent to the CNN for feature extraction. A classifier (such as an SVM classifier) is then trained on those features. The disadvantage here is that R-CNN takes a really long time to train, hence it has since been improved (Fast R-CNN and Faster R-CNN) [3]

- **SSD (Single Shot Detector)**: This algorithm uses a single neural network and combines 2 steps (region proposal and image classification) and performs them both in one forward pass of the network. [4]

- **SPP (Spatial Pyramid Pooling)**: This algorithm allows us to perform object detection on different sized images as it adds an additional layer to the CNN, which outputs a fixed-length image. [5]

- **YOLO (You Only Look Once) v3**: YOLO is a blisteringly fast take on Object Detection that only 'looks once' at an image, meaning it forward propagates it through the single neural network once and classifies it. This is my comparison algorithm. [6]

- **Bag of Visual Words**: Bag of Visual Words is an adaptation of the original Bag of Words method used for classifying text, that involves replacing the text words with image features as 'words'. This contains two parts, feature extraction, and classification of the image. Features (keypoints and descriptors) can be extracted from an image using an image descriptor such as a Histogram of Gradient (HoG) descriptor, SIFT, or SURF. The features are then used to compute a visual 'vocabulary' of sorts, and a classifier is trained on it. This is my implementation algorithm. [2]

B. Applications and Use-Cases of Object Detection

Object Detection is one of the fundamental abilities of the Computer Vision field, and is important for both researchers, and also industrial implementation. Object Detection acts as a building block for many parts of Computer Vision research, the most common of them being Robotics. It is also an essential part of Optical Flow tracking. A few industrial applications of Object Detection are:

- **Face recognition:** Object detection can be used to locate human faces in images.

- **Self-driving cars:** Self-driving cars use a number of different ways to detect their surroundings, object detection being one of them.

- **Security and surveillance:** Security cameras can detect if specific suspicious object/persons enter the view.

- **Classifying a huge set of images:** Object detection can help classify and make huge image sets easier to handle.

II. Theory

- **How SIFT works:** SIFT is a keypoint detector that is mostly invariant to scale, rotation, illumination and different viewpoints. Firstly, for scale invariance, a 'scale space' is constructed. A scale space holds different 'octaves', which are a set of increasingly Gaussian blurred images at that size. The number of octaves depends on the size of the original input image. The scale space is then used to find 'keypoints' in the image, which can be found by calculating the Difference of Gaussians (DoG) on the scale space. The DoG can be used to approximate the Laplacian of Gaussians (as calculating LoG individually turns out to be quite expensive). Using the LoG and checking for local extrema, keypoints can be estimated. So far, scale invariance has been achieved and keypoints have been computed. Unnecessary keypoints are then removed, and the next goal is rotation invariance. This is achieved by constructing a histogram with 36 bins representing 10 degrees each (for a total of 360 degrees). The orientation for each pixel is checked and assigned to one of the bins. The highest bin is used to calculate the orientation of image. Now that we have the orientation of each image, we make sure that the next calculations are done with respect to the orientation, which makes the algorithm is rotation invariant. Finally, a 16 by 16 window is taken around each keypoint, and broken into 16 4 by 4 windows. For each of the 16 windows, an 8 bin orientation histogram is calculated (45 degrees per bin). This gives us a total of 128 bin values, which is the 'descriptor'. The keypoints and descriptors are returned. [1]

- **How Bag of Words is implemented:** For Bag of Visual Words, a feature descriptor like SIFT is first used to extract features from different images. The image features are then 'clustered' using some clustering algorithm, and the result is then used to construct a frequency histogram with the features as different bins. The histogram acts as the the bag of visual words. A classifier can be trained on the bag of visual words and then used to predict the class of new images based on the features. [2]

- **How YOLO works:** You Only Look Once (YOLO) works by applying a single neural network to each image. The network divides the image into a 13 by 13 grid. For each region, the network then predicts the bounding boxes and probabilities. It then uses these predicted probabilities as a way to assign weights to the bounding boxes. YOLO also gives a 'confidence score' as an output for each object it classifies. [6]

For my implementation, I evaluated the performance by checking the number of correctly classified training images, and the number of correctly classified test images.

III. Data and Implementation

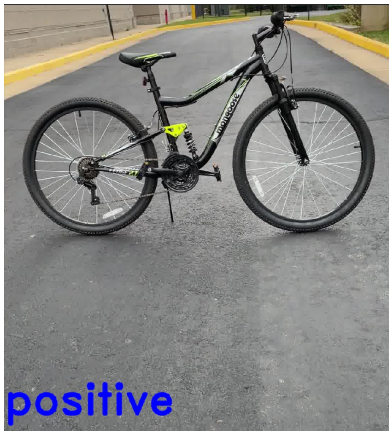
Regarding data, I have collected both the training and testing data by myself. I took a few short videos of my mountain bike around my dorm, split the videos into positive and negative videos. I then converted the videos into around 80 frames, and saved those as images. I also did the same for testing data.

In order to implement Object Detection using SIFT and Bag of Words, I originally intended (and attempted) to implement SIFT by myself, but unfortunately I was unable to get it working within the time constraint. I had to use OpenCV's implementation of SIFT and perform the Bag of Visual Words method on the descriptors returned by that function. For clustering, I used the scipy implementation of the K-means algorithm and vector quantization. I also used sklearn's LinearSVC as the SVM Classifier for this project.

IV. Results

- **My Implementation:** As expected, my implementation was not perfect. The implementation was trained on around 70 images (35 positive and 35 negative), and then tested on the same training images, and then on a set of held-out testing images. The results were:

  The classifier correctly classified all of the 70 training images. Two of the correctly classified training images are:



For the testing images, all the positive images were correctly classified as positive (no false negatives). The performance on negative images, however, was not that great. There were a total of 28 false positives out of 58 negative images, with the program seeming to find the mountain bike where it wasn't actually present.

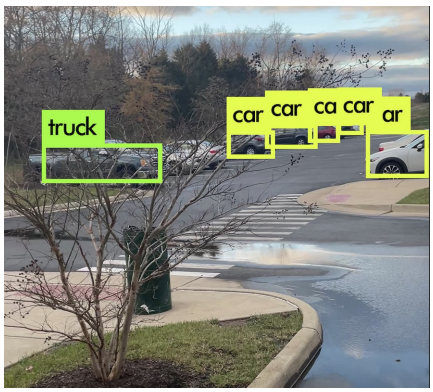
The following are examples of testing images that were correctly classified:

But the classifier also classified a set of images as positive, even when they did not have the bike in them:

In total, the classifier had an accuracy of 75 out of 103 test images, or **72%**.

- **Comparison Component:** For the comparison component, I originally intended to train YOLOv3 on the COCO dataset, but then read that the pretrained weights file for YOLO v3 has a 'bicycle' class. Hence I chose to use the pretrained weights file.

  YOLOv3 had a 100% accuracy rate, as it did not misclassify any of the images (with respect to whether or not it detected a bicycle class in the image). The following are two of the images that YOLO correctly classified:



REFERENCES

[1] David G. Lowe. *Distinctive Image Features from Scale-Invariant Keypoints*. URL: https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf

[2] Gabriella Csurka, Christopher R. Dance, Lixin Fan, Jutta Willamowski, Cédric Bray *Visual Categorization with Bags of Keypoints*. URL: https://europe.naverlabs.com/wp-content/uploads/ultimatemember/temp/2004_010.pdf

[3] Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik *Rich feature hierarchies for accurate object detection and semantic segmentation*. URL: https://openaccess.thecvf.com/content_cvpr_2014/papers/Girshick_Rich_Feature_Hierarchies_2014_CVPR_paper.pdf

[4] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg *SSD: Single Shot MultiBox Detector*. URL: https://arxiv.org/pdf/1512.02325.pdf

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun *Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition*. URL: https://arxiv.org/pdf/1406.4729.pdf)\unhbox\voidb@x\bgroup\let\unhbox\voidb@x\setbox\@tempboxa\hbox{\OT1\i\global\mathchardef\accent@spacefactor\spacefactor}\let\begingroup\endgroup\relax\let\ignorespaces\relax\accent18\OT1\i\egroup\spacefactor\accent@spacefactor%20~

[6] Joseph Redmon, Ali Farhadi *YOLOv3: An Incremental Improvement*. URL: https://pjreddie.com/media/files/papers/YOLOv3.pdf