NUS CS4243: Computer Vision and Pattern Recognition November 7, 2019

Where's Waldo Detector using Computer Vision

Lecturer: Angela Yao Student(s): Maximilian Fruehauf, David Drews, Choo Wen Xin

Abstract

This report describes our group's implementation of a computer vision algorithm to detect Waldo, Wenda, and the Wizard from a series of "Where's Waldo" books. The goal of this project is to detect the three characters from the provided high-resolution images, which can be very complex with a lot of detail and many other characters. The three characters also may or may not appear in any given image.

Due to the complex nature of the given images, and variation of the characters' appearances, detecting the characters accurately proved to be a challenge. In some cases, we could not identify where the characters are, and a lot of false positives were present as well.

Our proposed solution is to use a histogram over gradients (HoG) feature descriptor, then training a linear support vector machine (SVM) to create our classifier. We were able to detect some instances of Waldo, espcially in the postcard in the top left hand corner of the page.

1 Introduction

"Where's Waldo" is a series of books containing detailed, high-resolution illustrations. For this project, we are given a set of scanned images from the book, and are tasked to design a computer vision algorithm that can detect the three characters from the book: Waldo, Wenda, and the Wizard. Each image may contain one or more of these characters. We are provided with a set of training images with annotations of the bounding box locations of Waldo, Wenda, or the Wizard in each image.

Throughout the course of the project, we have attempted several different methods with varying degrees of success. While there are several existing computer vision algorithms for face and object detection, such as the "You Only Look Once" object detection [1], these do not seem to work too well with illustrated characters like Waldo, in a complex illustrated iamge. The characters' appearance and size also seems to vary across images, sometimes only a smaller part of the character (i.e a part of the face) is visible. This makes identifying the characters more challenging as well.

Across all approaches, we've only been working with the data that was handed to us as part of the assignment. Although our code alters the given data wherever needed, we did neither add any entirely new data from sources like the internet nor did we manually improve the existing data as could be achieved by adding or improving annotations.

The data we utilised can be summarised as follows:

- 80 high resolution images from the *Where's Waldo* book series
- 137 images of Waldo cropped to the patch of the image as describe in the annotation files

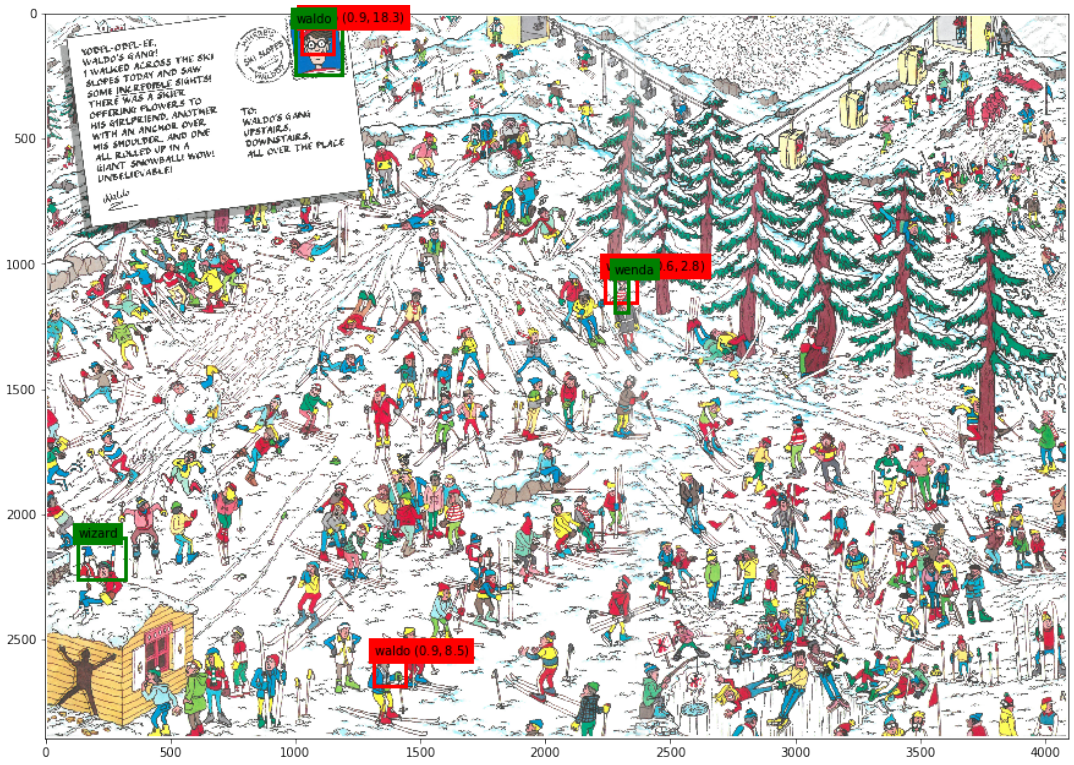

Figure 1: Exmaple of an image showing detections with $(name, p, p_{ratio})$ in red, ground truth in green

- 43 images of Wenda cropped to the patch of the image as describe in the annotation files
- 27 images of Wizard cropped to the patch of the image as describe in the annotation files
- Approximately 14,000 to 15,000 images (128x128px) subsampled from the original images, which did neither contain Waldo, Wenda or Wizard, used as negative training samples

In order to assess the performance of our algorithms we split the given set of images in a training and a testing set. This reduced the amount of positive samples of the three characters to 124 training samples for Waldo, 36 for Wenda and 24 for Wizard. As we will discuss in the upcoming sections, this uneaven split of training samples influenced the performance of our solution for the three different classes.

We proposed using a few methods:

- histogram over gradients (HoG) feature descriptor
- training a linear support vector machine (SVM) to create a multiclass-classifier

We will discuss our proposed solution in the next section.

2 Proposed Solution

Give an overview of your solution, put it in a framework. Then, detail each part in the framework.

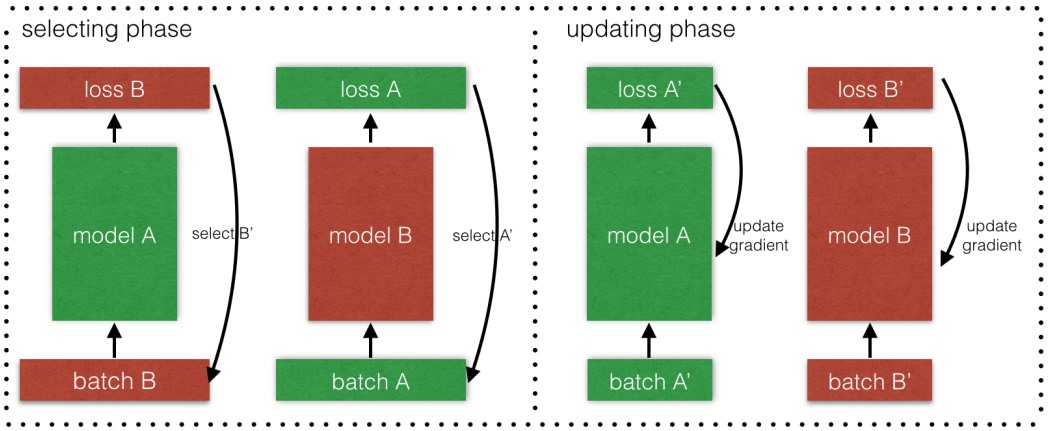

Figure 2: Our proposed solution.

3 Experiments

3.1 Data Preparation and Configuration

The images in the dataset come as annotated bounding boxes of varying aspect ratios and sizes in pixels. Also the size of the respective images changes as well. Therefore we simplified the detection process by generating (128×128) px training images from the provided bounding boxes. Doing so resulted in some croping to the actual characters appearances, which cannot be avoided in some corner cases. However in general we were able to regularize the bounding boxes without loosing any pixels belong to the positive annotations. See Figure 3 for examples of the cropped and scaled input images.

To map the color images as robustly as possible to gray scale values we used a K-Means detector and assigned each of the modes an equally gray spaced color. K-Means works by intially choosing random starting points in the image and then projecting an area around them, calculating the "center of gravity" of those inlier points and then shifting the new centroid to this "center of gravity". When repeating this process until convergence k such different means are found. This process results in a non-uniform quantization of the input image.

We use this property to train the K-Means classifier on a sample image of the `wenda` class, which only shows the colors red, blue, black white. However we use $k = 5$ to classify any color not lying in these four as a separate class, and then merge its results with the white color class. Then these four resulting classes colored centroids are mapped to uniformly chosen values $v \in [0, 255]$. This drastically reduces the complexity of the input images as can be seen in the right image in Figure 3, reducing input noise when processing the images with our feature descriptor in subsection 3.2.

Figure 3: Preprocessed sample images for the waldo class. The left images are cropped and scaled to 128×128 pixels. To the right the same images are shown after kemans preprocessing (originally gray-cale, but color mapped for display purposes)

Class name	#samples	percentage
waldo	137	1.01%
wenda	43	0.32%
wizard	27	0.20%
negative	13315	98.47%

Table 1: Dataset statistics for the training

Generating negative samples is done by randomly sampling regions of differing scales and positions from the provided training images. In this process we additionally make sure, none of the proposed negative samples overlapping with any on the ground truth annotations present in the images. Then we scale these proposed negative samples to (128×128) px and apply the filtering described in the above paragraph to each sample. Since the negative samples are randomly generated, we can create arbitrary amounts of them. For our testing ≈ 15000 worked well. Additonal information in the exact size of the dataset is shown in Table 1

3.2 Implementation

The pipeline for detecting the classes of a given bounding box are displayed in Figure 4, the first 3 steps of which are covered in subsection 3.1. In the following paragraps we will focus on the steps 4-7 of any detection and provide some details of interest in the implementation.

HOG & HOV feauture descritor To capture shape and color information from the pre-processed sample images, we adopt a twofold feature descriptor, which consists of the HOG descriptor and a histogramed version of the SIFT detector, we call HOV.

HOG works by first calculating the horizontal and vertical gradients in the image, which are then separated into quadratic cells. From both a horizontal and vertical window, the magnitude and direction (radians or degrees) of the gradient get computed. Then the magnitude is binned according to the corresponding direction. These bins then form a histogram for the cell. This

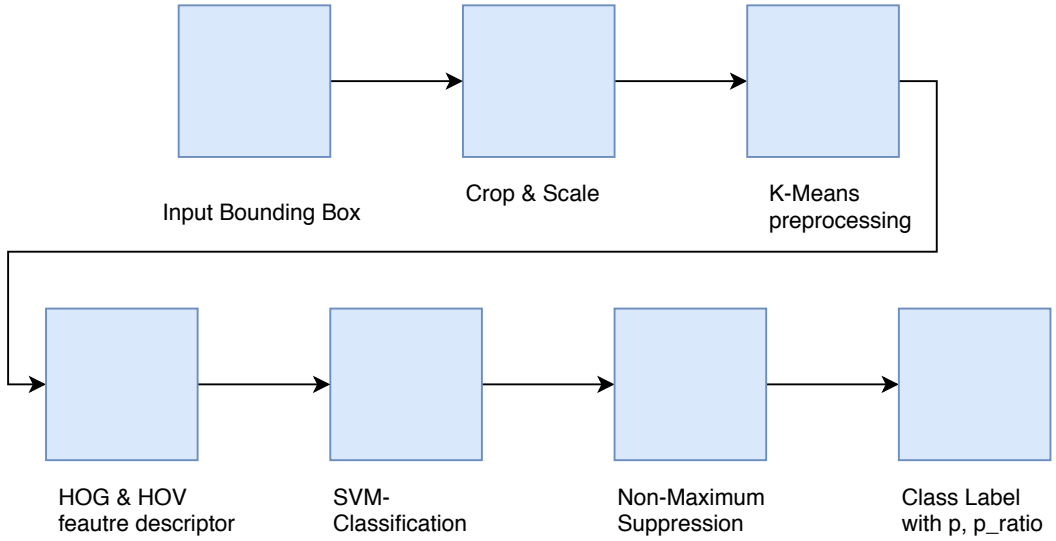

Figure 4: Schematic overview of the wlado / wenda / wizard detection pileline.

gives some invariance of the feature descriptor to slight pixel value changes. To also gives some spatial invariance, a set number of these histogramed cells are concatenated together. Which is repeated for each cell. In a final step, the resulting feature vector is normalized with one of a multitude of possible norms. We decided to use the L_2-Hys norm.

The HOV (Histograms over Values) descriptor works essentially the same (in fact most of the code is copied from the `scikit-learn` HOG implemenation), with the only difference being that the initial histogram on each cell is not computed from the gradient of the input, but rather the values of the input image inself.

Concatenating these two independently (with the same parameters) computed feature descriptors gives a good spatial and value representation of the input images. The effect of which we evaluate in subsection 3.3.

SVM Classification

Non-Maximum suppression

Class Label outputs

3.3 Results

Present the results, both qualitatively (visualize) and quantitatively (specific numbers).. Analyze the results

Evaluate hog only vs hog / hov

3.4 Discussion

Strengths and weakness in your method.

4 Conclusion

In this project, we ...

5 Group Information

Member	Student ID	Email	Contribution
Maximilian Fruehauf	Axx	e0445541@u.nus.edu	xxx
David Drews	Axx	e0454245@u.nus.edu	xxx
Choo Wen Xin	A0160465H	e0053347@u.nus.edu	xxx

Table 2: Group member information.

References

[1] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. University of Washington, Allen Institute for AI, Facebook AI Research, 2016.