Upper-body detection using R-CNN

Nguyen Phan Manh Hung
Honours Program
Faculty of Information Technology
University of Science, VNU-HCM
Email: nguyenphanmanhhung@gmail.com

Nguyen Hoang Khanh Duy
Honours Program
Faculty of Information Technology
University of Science, VNU-HCM
Email: nguyenhoangkhanhduy@gmail.com

Luc Kien Nghiep
Honours Program
Faculty of Information Technology
University of Science, VNU-HCM
Email: luckiennghiep@gmail.com

***Abstract*—Human dectection is an important technique used in many other problems such as traffic control, movies' features extracting, etc. However, it's hard to detect the whole body because the lower part is usually obscured by other objects like bikes, cars, tables, etc. This motivates us to use R-CNN to detect upper-body of the target. The experimental result using this technique achieves the accuracy of 95.5 %. The dataset consists of 300 images. People in these images are from different ages and have different angles. This method can be improved by using better dataset.**

I. Introduction

The increasing development of machine learning provide us more tools for different kind of detection problems. One of the problems is human detection in images. Human detection is an important step to solves other important problems such as the pedestrian detection system in self-driving cars or surveillance camera systems. However, human bodies are mostly obscured by other objects like bikes, cars, tables, trees, etc. This is an obstacle that can dramatically decrease the accuracy of full-body detection algorithms. That's why the authors recommend detecting upper body only using Region-based Convolutional Neural Networks.

There are many methods used for human detection, including SVM with discrete Wavelet transform [1] and HOG[2]. However, these methods have some disadvantages. For example, if images have low contrast and unclear boundaries due to some factors such as light or low-quality cameras, the images will provide incomplete gradient information, the fact that dramatically decreases the accuracy of detectors. To get rid of these disadvantages, we can apply some image preprocessing techniques. However, each image has different characteristics, therefore requires different techniques to preserve its properties.

The core idea of our method is using R-CNN trained with the dataset of humans' upper-body. To improve the accuracy, we pre-train the model using a large auxiliary dataset, ILSVRC, as recommended by [3]. After trained with 300 images, the origin model achieves the accuracy of 90% [need to fix]. The modified model give a better result with the accuracy of 95.5% [need to fix].

The rest of this paper is organized as follows. In section II, the authors describe dataset and methodology use to train and validate the model. In section III, we discuss about our model used in this paper. Section IV presents the experimental results and evaluations. The conclusions and future work are presented in section V.

II. Background and Related Work

Given a set of objects, the main goal of detection problem is to localize these objects in images, videos, or pieces of musics. This problem is difficult than the classification problem, in which we have to build a model that, given a object, can map it into a particular set specified by the problem. Formally, given $x \in D^n$ and a discrete set $R = \{y_0, y_1, ...y_m\}$, we have to build a transformation $F : D^n \rightarrow R$ that satisfies some constraints. However, there are many paradigms allowing us to utilize well known classifiers as detectors in our problems.

These paradigms usually have 2 main stages, whose structure will be discussed in the next section.

A. The objects proposal stage

The main goal of this stage is to propose a set of image's portions that is likely to contain the target objects. For this reason, it's important to choose an appropriate algorithm to produce a good set of candidate objects because no matter how well the classifier performs, if the number of candidates is insufficient or the objects extracted from the images are badly affected, the overall accuracy of the detector system will drastically decrease. Fortunately, there are many methods, divided into 2 types which base on grouping super-pixels and sliding windows [9], both simple and complex to solve the problem.

- **Sliding Window.** [4][5] We use rectangles of different sizes sliding over images. The set of images' portions, called sub-images, covered by those rectangles will be used as inputs for the classifier next stage.
  For better results, before fed to the classifier, those sub-images should be evaluated by some function F_{score} and then removed if their scores are below a specific

threshold. In this way, we can speed up the whole system. For example, we want to detect people in images. Suppose that we have a sub-image where there isn't any change of image's brightness or color, which shows that the sub-image contains purely background information. Instead of feeding it to the classifier, which eventually classify it as the background, we can get rid of it from the beginning, using simple evaluation.
Additionally, sizes of windows, or the ratio between edges in particular, should be chosen carefully. To be clear, let examine a human's face detector. It's likely that humans' faces fit into squares or rectangles with approximately equal sides. Therefore, if we choose a window with ratio between sides approximate 5:1, the classifier may fail to recognize the face.
Another way to use this technique in detection problem is to decompose the interest object into many parts [7]. Each part is then detected by a specific type of window and combined to get the original object.
In conclusion, this technique is simple and efficient but has high computational complexity in practice due to the redundancy of sub-images.

- **Local interest point** [8] Instead of considering all the windows, we only need to concern the areas around interested points extracted from images. In this way, we can decrease the number of candidate object dramatically.

B. The classifier stage

After being proposed by the previous stage, in this stage, proposed regions are examined more carefully using other algorithms. To classify these object, we don't usually use the pixels directly. The two popular methods are appearance-based method and feature-based method. Appearance based method uses example images to perform this stage, but changes in lighting or color, changes in view direction, and changes in size or shape can fail our system. Edge matching, greysacle matching and gradient matching are some of appearance-based methods.

The other method is feature-based method. In this method, we usually extract some features from objects to use instead of using the pixels directly.[**?**]. The reason they explained in [**?**] is because feature an act to encode ad-hoc domain knowledge that is difficult to learn using a finite quantity of data and the feature based system operates much faster than a pixel based system. There are a lot of feature we can use.

In face detection[**?**], Paul and Michael use Rectangle feature. ——Chen cai hinh vo day——— Them sum of pixels which lie within the white rectangles are substracted from the sum of pixels in the grey rectangles. Using intergral image [**?**], we can compute any rectangle feature rapidly. Other popular feature we can use are SIFT, SURF, HOG, etc.

Traning:

III. Method

Our system is divided into following part:

1) Object localization
2) Pre-training
3) Supervised fine-tuning

A. Region proposal network

In this paper, we use Region Proposal Network (RPN)[9] to suggest regions for the classifier. RPN takes images of multiple sizes as input and outputs pairs of (proposed region[1], objectness score[2]).

1) Generate Region: We treat the feature maps generated by the last shared convolutional layer [3] as input images and slide the RPN over the feature maps. Each sliding window is then mapped into a lower dimensional feature and fed into 2 fully connected layers, called box regression layer and box classification layer correspondingly. The box regression layer predicts coordinates of the bounding box associated to each anchor. The box classification layer computes the corresponding objectness score of each proposed region.

Each sliding window is associated to an abstract object called anchor. The anchor is placed at the center of the sliding window and includes scale and aspect ratio of the proposal region. By using anchors, we can predict regions of any shape without changing scale and aspect ratio of the sliding window.

This approach, using shared RPN and the sliding window, has 2 important properties.

1) *Translation invariance* By using the shared RPN and the sliding window, the detector can tolerate the translation of objects. Furthermore, using shared RPN allows reducing size of the model, therefore requires fewer traning data to achieve an equivalent accuracy.
2) *Multi-scale object* There are many ways to deal with multi-scale objects such as image-pyramid in HOG and multi-scale sliding windows. The former is efficient but time-consuming. The latter is also efficient and sometimes is used together with the first one. In this paper, we use multi-scale anchors instead, which allows us to use fixed-size images/feature maps and sliding windows. This is more efficient than methods mentioned above in term of time complexity.

[1]"proposed region" is an abstract term. The shape of a proposed region depends on algorithms, objects, problems, etc. However, in this problem, we use rectangle regions because it's simple and efficient enough to solve the problem.

[2]The objectness score measures the probability if the proposed region is a object or not.

[3]The RPN and the classifier share a convolutional feature maps to reduce the computational cost.

2) Training: To train RPN, we use stochastic gradient descent and back-propagation. The loss function is:

$$L(p, l, \hat{p}_g, \hat{l}_g) = \frac{1}{N_{cls}} \sum_i L_{cls}(l^i, \hat{l}^i) + \frac{1}{N_{reg}} \sum_i L_{reg}(p^i - \hat{p}^i) \tag{1}$$

In 1, p^i is vector representing the coordinate of the bounding box, which includes x, y, w, h[4], predicted by regression layer associated with anchor i of mini-batch samples and l^i is the probability that region i contains object or not. Similarly, $\hat{p}^i$ is the coordinate of the ground-truth bounding box, and $\hat{l}^i$ is the ground-truth label of region i.

$$l_g^i = \begin{cases} 1 & \text{if region i contains object} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

L_{cls} is the cross-entropy loss function:

$$L_{cls} = \hat{l}^i \log l^i + (1 - \hat{l}^i) \log(1 - l^i) \tag{3}$$

L_{reg} is a robust L_1[**?**]:

$$L_{reg}(p^i, \hat{p}^i) = \sum_{u \in \{x,y,w,h\}} smooth_{L_1}(u_{p^i} - u_{\hat{p}^i}) \tag{4}$$

in which

$$smooth_{L_1} = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases} \tag{5}$$

We train the model with 100000 iterations. Learning rate, weight decay, and momentum is 0.001, 0.0005, and 0.9 correspondingly.

B. Pre-training

Due to the enormous number of parameters, the model we use, Zeiler and Fergus model, requires a huge amount of training examples, which takes much time a to collect and label. To address the issue, we first initialize the model by training it with images from ILSVRC 2012 and then fine-tune the domain of our model by Uper-Body dataset. In this way, the model can learn a higher level of abstract features before trying to learn from the specific dataset of the problem.

C. Supervised fine-tuning

Many algorithms are proposed to address human detection problem. Most of them achieve high accuracy when whole body is captured or only obscured a little. We use upper-body dataset for training model, concerning the fact that humans in "wild environment" are usually obscured by other object such as tables, cars, motorbikes, etc. To be clear, take a look at 1. In this image, the lower parts of these people are hidden by the car. Therefore, should any detectors try find humans in this image by searching whole bodies, they will fail.

[4]x, y are the coordinates of the box's center, and w, h are its width and height correspondingly.

Fig. 1

We divide the dataset into 4 classes: face wide [5], face side, upper body wide [6], upper body side. Dividing upper-body dataset into 4 sub-classes allows the model to learn common features of each class better than to learn features of a super class, upper-body, only. Additionally, by using symmetric transformation, model can learn both left/right side of face/upper-body simultaneously.

IV. Experiment results

The authors retrain the model with a different dataset to solve the Upper Body Detection problem. The dataset contains 98 images of people or groups of people. The authors concern 4 types of upper body, which is used as 4 classes for the ground truth.
The 4 types of upper body are: face_wide, face_side, upper_body_wide (ub_wide), upper_body_side (ub_side) The number of images that each class appears in is as the table below:

face_wide	face_side	upper_body_wide	upper_body_side
51	43	31	62

To rate the result, we recommend a new scale:
Let R = $(C - W)/A$

Where:
R (rating) is the rating of the detection algorithm on a specific test. We want to maximize this value. And we can use this value to tell if one algorithm is better than another.
C (correct) is the number of correct objects (upper body or face) detected.
W (wrong) is the number of objects that is mistakenly detected.
A (all) is the total number of faces and upper bodies appeared

[5]front face
[6]the part above belly

in the images used for testing.

Let see why this formula works. We want to maximize C to reach A (the total number of objects in the test). Thus, C/A is pretty much the right formula for this detection problem. It represents the posibility that an object will be detected. The higher this value is, the better our algorithm is. But we don't want to forget that sometimes our detection system fails to detect an object and claims one object to be another. So we add the W value to the formula. In that way even the algorithm manages to detect all the objects in the test, but meanwhile mistake other objects to be faces or upper bodies, it still gets a bad rating.

The authors run the model on 5 tests. Each test contains 10 images. After that, we calculate the R values for each test and calculate the average value avg(R).

The results can be desmonstrated by the following images:

And here we see in figure 2a, most men's upper bodies are detected, except for 2 men on the left and 1 man on the right. This is because these upper bodies are not regular and the angles of their body are not included in the training set.

In this case, R = 4/9 (< 50 %), 4 upper bodies are detected among 9 upper bodies. this is still an encouraging result considering the complexity of this test case.

In figure 2b, 5/6 upper bodies are detected despite some bodies are obscured and the image's quality is quite low. The undetected upper body is obscured and the color of the T-shirts also make it harder to detect in this test.

In this test, the detector also found 5 faces, which are presented in 2c. The undetected upper body's face is also undetected. This can be traced down to the limitation of the training stage, which only uses 98 images.

Still, this result is quit good, which gives R = 5/6 (about 83 %)

In figure 2d, all the upper bodies (and all faces) are detected with high certanties (> 0.95). This proves that in common cases, the detector can work well and give trusted result. In this case, R = 1.

In figure 2e, all upper bodies are detected correctly, which gives R = 1.

Only 2 out of 3 faces are detected. The last one is mostly obscured and too hard for the detector.

V. Conclusion

In this paper, we suggest using Faster R-CNN to address human detection problem. However, by observing hundreds of images from surveillance cameras and the internet, we conclude that lower-bodies, below bellies, contribute little useful information for detecting humans, and under some circumstances they become serious obstacles for detectors because in most cases, lower-bodies are obscured by other objects or by camera perspective. That's why we propose training the model with the upper-body dataset instead. The model trained with this dataset perform well on images randomly collected from the internet even though the dataset contains only 98 images.

References

[1] M. M. Deshpande, J. G. Rana, and M. M. Pawar. *"Human detection based on discrete Wavelet transform"*. Sustainable Energy and Intelligent Systems. IET Chennai International Conference. 3rd 2012.

[2] N. Dalal and B. Triggs. *"Histograms of Oriented Gradients for Human Detection"*. Sustainable Energy and Intelligent Systems, IET Chennai International Conference, 3rd 2012.

[3] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. *"Region-based Convolutional Networks for Accurate Object Detection and Segmentation"*. IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 38.

[4] H. A. Rowley, S. Baluja, and T. Kanade. *"Neural network-based face detection"*. TPAMI, 1998.

[5] R. Vaillant, C. Monrocq, and Y. LeCun. *"Original approach for the localisation of objects in images"*. IEE Proc on Vision, Image, and Signal Processing, 1994.

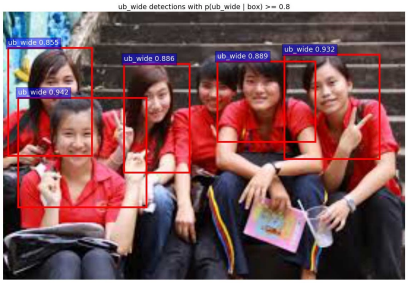
[6] Kevin Murphy, Antonio Torralba, Daniel Eaton, and William Freeman. *"Object detection and localization using local and global features"*. Volume 4170 of the series Lecture Notes in Computer Science pp 382-400.

[7] Anuj Mohan, Constantine Papageorgiou, and Tomaso Poggio. *"Example-based object detection in images by components"*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 23(4):349361, 2001.

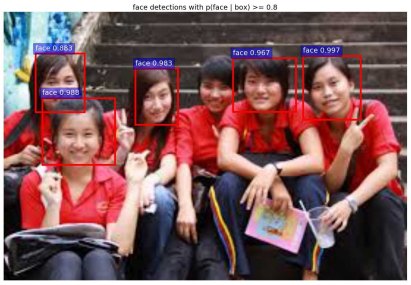
[8] G. Bouchard and B. Triggs. *"A hierarchical part-based model for visual object categorization"*. In CVPR, 2005.

[9] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. Neural Information Processing Systems 2015.

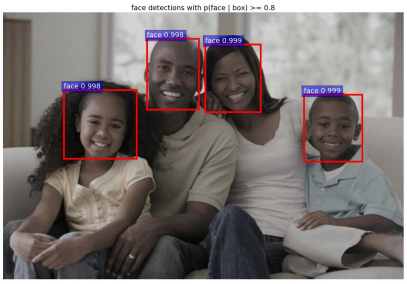
(a)



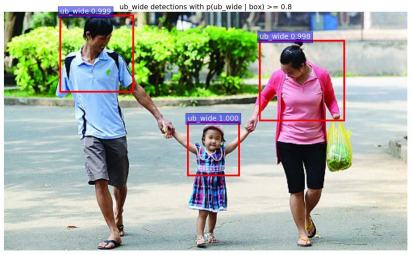
(b)



(c)



(d)



(e)

(f)

Fig. 2: Result pictures