# Application of Convoluted Neural Networks for Pedestrian Detection

Anil Variyar
Stanford University
anilvar@stanford.edu

## Abstract

*This paper focuses on the application of Convoluted Neural Networks for pedestrian detection. This is critical for self driving cars. Work in this area was based on feature based methods for a while before convoluted neural networks were applied to it. In this work, a convoluted neural network based pedestrian detection pipeline was set up using the AlexNet and ResNet algorithms. These were trained on a subset of the Caltech Pedestrian Database and then tested on another subset of the database. It was observed that both the methodologies performed fairly well in detecting pedestrians from the specified images. However the results seemed a little too optimistic because of the small dataset used to test the results. Tests on the complete dataset is required to verify whether the methodology works well.*

## 1. Introduction

Object detection has been an important and challenging part of computer vision problems. It is a critical component of many areas including robotics, autonomous systems, security systems, and unmanned aerial vehicles. Pedestrian detection, is a sub-part of object detection where the goal is to take either a frame (an image) or video and detect the presence of people (humans) in the image. This has been a subject of research over the past few years as it is an essential component of autonomous/self-driving vehicles. It can also be used as a safety feature for existing vehicles to detect pedestrians and warn the driver, as well as in security systems like traffic cameras. For this work we focus on detection in frames/images. Moreover while realistic systems require real time detection, we do not try to speed up the test/detection mechanism. Thus the problem input reduces to a set of images containing or not containing pedestrians, which are fed into the pedestrian detection pipeline (described below) containing a convoluted neural network. The neural network is required to provide a bounding box indicating the presence of a pedestrian. As a sliding window methodology (described below) is used for image subsampling, the CNN gets a set of cropped images and the output is a classification stating whether the cropped image contains a pedestrian or not. The post-processing methods use this output to compute the bounding boxes.

## 2. Related Work

Initially feature based methods were applied to this problem. With the resurgence of convoluted neural networks, these were applied to the problem as well.

### 2.1. Feature-Based Methods

Supervised and unsupervised learning techniques have been applied to this problem. The work of Rowley[6] which applied neural networks for facial detection was a first step in this direction with the focus on feature based detection. The work by Viola and Jones[7] between 2001 and 2003 was a major milestone in pedestrian/object detection. They used an integral image representation and a new feature detection/learning algorithm based on AdaBoost. This was followed by the work of Dalal and Triggs[5] on Histograms of Oriented Gradients (HOG) for human detection which resulted in further improvements to feature based prediction and pedestrian detection systems. These were followed by improvements to the feature based methods in order to improve feature-based detection. However most of these methods used feature-based detection coupled with SVM regression methods to perform pedestrian detection. The review papers by Dollar et al. in 2009[2] and 2012[4] summarize the different feature based methods in detail and compare their performance on the Caltech pedestrian database[3].

2.2. Convoluted Neural Networks

However since 2012, convoluted neural networks have revolutionized the field of computer vision and image detection. Architectures like AlexNet[12] and GoogleNet[13] reduced the errors in the Image net challenge by record levels. This has prompted a significant shift in the approach of the community with convoluted neural networks being applied successfully to a large number of vision and image classification problems. Because convoluted neural nets are a new concept, their application to pedestrian detection systems is still in its infancy. The work by Tome et al.[1] in 2015 applied CNNs to this problem. The work looked at the performance of different region proposal and feature extraction methods coupled with a CNN. Both the AlexNet and the GoogleNet architectures were studied. Other work in the same time frame includes the work by Fukui et al.[10], Sermanet et al.[8] and Angelova et al.[9] . All these approaches perform a feature extraction from the image and then use CNNs for classification. A different approach was observed in the thesis of Molin, D [11] where the network takes in the entire image as an input (instead of a feature) and generates the probability of a pedestrian on the image pixels (at a location) as the output. The approach while giving results similar to existing methods was a shift from existing methods observed. R-CNN based pipeline was recently successfully applied to this problem[17].

## 3. Methods

3.1. Convoluted Neural Networks

Solving the pedestrian detection problem requires the setting up of a pedestrian detection pipeline as shown in Figure 1. The pipeline contains a region proposal method, a feature extraction pipeline and a region classifier. In the case when an R-CNN method is used (not here), the region proposal and the feature extraction methods are part of the R-CNN.

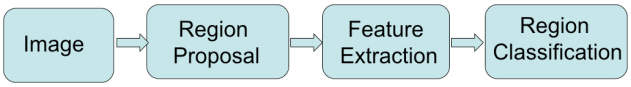


Figure 1: Pedestrian Detection Pipeline

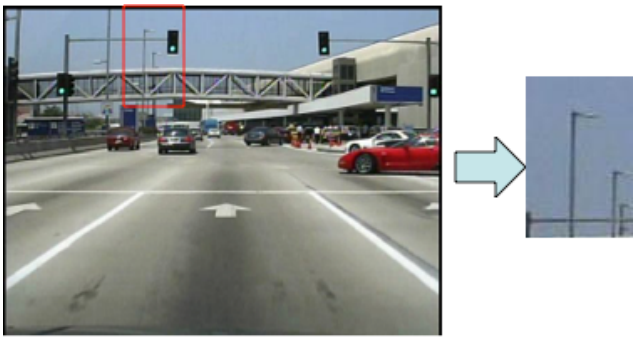

Figure 2: Sliding window approach

For region proposal, a sliding window approach is adopted here. During this method, sections of the images (sub images) , which can have different sizes/aspect ratios are extracted from the image. As shown in Figure 2 where the red section/window is extracted as a feature. These are then fed into the feature extractor. The sliding window approach is used both during training and test runs in the same fashion.

The feature extractor for earlier implementations was a simple HOG like feature detector. For this approach the convoluted neural network (CNN) is used as a feature extractor. The cropped image from the sliding window detector is passed on to convoluted neural network. In the training runs, the image is preprocessed by first scaling the image (which is smaller ). For most of the cases, the sliding window detector is set to return images of size 96x48. These are scaled to required size of the CNN (224 x 224) inputs. The images are then normalized (subtracting a mean and scaling by a standard deviation). Training is then performed on these images in batches of 32. Two CNN architectures are tried as feature extractors, AlexNet[12] and and a 18 layer ResNet[14]. The AlexNet architecture has been applied to the problem in existing literature but no cases of the ResNet being applied to the problem were found by the author. Thus studying how well the ResNet performs on the problem compared to the other architectures was seen to be an interesting problem.

The AlexNet model is a 8 layer CNN containing 5 convolutional layers and 3 fully connected layers. It is a relatively simple but fairly powerful model that was tried for its simplicity and its effectiveness. The details of the model are described in Reference [12].

The ResNet is a fairly complicated CNN with many layers. An 18 layer network is used for this work though much larger layers have been used for other applications. The basic idea of the ResNet (Residual Network) is that unlike previous architectures, the network also has connections between neurons 2 layers apart. Thus the alternate layers receive information from neurons that are not just in the previous layer allowing the features in that layer to be affected by more neurons and layer with feature sets. Details of the method are described in Reference [14].

The last step of the pipeline is a classification pipeline is the classifier. For the current study, the classification problem is kept simple with two classes, person detected (class 0), person not detected (class 1). However, a more complicated problem with classes including vehicles detected, etc. can be solved using the same approach by changing the last fully connected of the CNNs. The softmax classifier is used here. The formula for the loss is shown in Equation 1 (taken form the CS231N notes)[18] where $L_i$ is the loss associated with sample i, and $f_{yi}$ is the class score of the class y for sample i.

$$L_i = -\log\left(\frac{e^{f_{yi}}}{\Sigma_j e^{f_j}}\right) \quad \text{- Equation 1}$$

In training mode, once the set of the scaled and normalized images are passed to the CNN (both architectures), it is trained in batch mode (32 images/batch) using a Nesterov momentum based SGD method. The presence of the bounding boxes in the scaled image for the training images is passed on to the CNN for training. During the test evaluations, the images are passed to the CNN which classifies it as class 0 or class 1 and these class values are extracted out of the pipeline. These can be used to check the test accuracy of the CNN. Moreover, these can be used for further post processing. The sub images can be regrouped into their parent images and based on the class detected, the bounding box can be created in the image. Moreover, these can be used to generate other evaluation/testing metrics, such as those specified in the paper by Dollar et al[2,4].

The pedestrian detection pipeline is setup in Torch7[19] using existing implementation in torch which were modified to fit into the pipeline. Python scripts were used for pre and post processing. The general CNN testing and training framework was obtained by modifying the ResNet implementation on github by facebook [15,16]. This implementation contains the resent along with and interface to load/preprocess images, train using sgd and run test evaluations. This framework was adapted to permit other networks (AlexNet obtained from the work of Soumith[20,who set up a set of ConvNet benchmarks on github) to be plugged in and integrated with the pedestrian detection pipeline. Post-processing modules were added to permit extraction of the test/evaluation results for further post processing.

**4. Dataset and Features**

The Caltech Pedestrian Database[3] containing about 250,000 frames (60x480) with 350,000 bounding boxes is used to obtain training and test data. This is a standard test for the pedestrian detection problem. The dataset is described in detail in the paper by Dollar et al[2,4]. Samples from the database are shown in Figure 3. As shown some of the image contain a single pedestrian, some contain multiple pedestrians and some do not contain any pedestrians. Moreover in some of the images the pedestrians are clearly visible and are larger in size(closer to the camera) while in some of the images, the pedestrians are occluded by objects or very small(far away from the camera). The dataset consists of 11 sessions of data collected from video taken driving around busy streets. Sessions 0 to 5 are assigned to the training set and the remaining are assigned to the test set. The images are obtained as .seq files which were preprocessed using matlab code obtained along side the database at the Caltech Pedestrian Database webpage[3] and python code. The bounding boxes around the pedestrian is also

provided as a collection of text files (again these need to be extracted from a compressed file using matlab code provided). Each file corresponds to a frame/image in the database and is empty if no pedestrian is present or contains the coordinates od the bounding boxes if a person/multiple people are present.

The Dollar et al. paper suggested that every third frame in the training set is taken as a training sample and every 30$^{th}$ image in the test sets are used as test samples. However the sliding approach resulted in the images being further divided into about 50 sub images. This made the training and test sample sizes intractable for this work as the training times were becoming extremely long and even testing a challenge. Thus the datasets were further sampled from. The training dataset was sub sampled (randomly) to 2000 images (around 100000 images for the CNN) and the test set was also reduced to 1500 images by random sampling. The goal is to re-run the solution on larger datasets once basic pipeline is completed validated.

The standard evaluation metrics for the accuracy of the detection system is described in the work by Dollar et al and also in Tome et al. For each algorithm/architecture, the miss rate (MR) and the number of false positives per image (FPPI) are measured. For each bounding box, if the bounding box specified by the algorithm and the actual bounding box (truth value) do not overlap by more than 50% of their area, then it is considered as miss or a False Negative. If the bounding box is specified by the algorithm in a region that does not contain a (truth )bounding box/pedestrian, then it is considered a False Positive (FP). This is used to compute the number of false positives per image (FPPI) and averaged. The miss rate (MR) is the ratio of the number of False Negatives to the number of correctly identified boxes (Positives). However as described in the section below, as the classification results seem unrealistic, the standard CNN metrics of test accuracy are used for the time being. Once more realistic results are obtained by training/testing on a larger dataset and ensuring that the pipeline itself is not flawed (in its implementation), these metrics will be used/described.

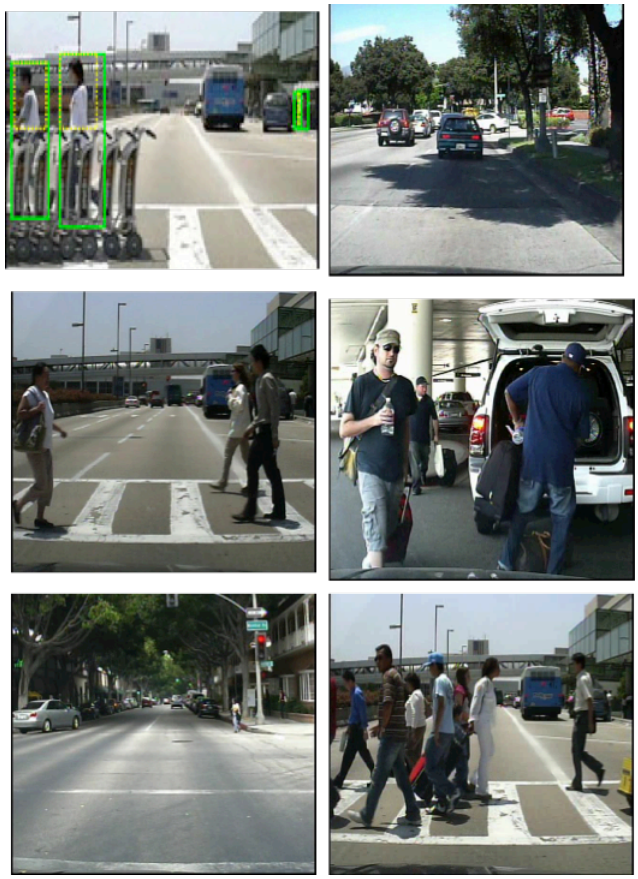

Figure 2: Samples from the Caltech Pedestrian Database

## 5. Experiments/ Results/ Discussion

To train the network, the images are first passed through the sliding window detector which generates a set of smaller sub images for training. These are then post-processed and fed into the CNN for training. The Nesterov momentum method is used for training. Different learning rates are tried for both the architectures and the best performing values are selected (0.1 for the Resnet and 0.01 for the AlexNet). The momentum value is set at 0.9 and the

weight decay is 1e-4. For both the networks, the training set is run for multiple epochs (6 for AlexNet, 3 for ResNet) with a batch size of 32 images an iteration.

Figure 4 shows the evolution of the loss function during training for both the CNN architectures. While training was performed for multiple epochs, we see that the loss is significantly reduced in the first 100 or so iterations and then doesn't change significantly. The loss for the ResNet comes down much faster than the AlexNet. The training was performed for 2000 training images that resulted in a total of about a 100000 images. The test accuracies for the trained models were shown to be extremely high (around **5%** for both the architectures). This seemed to be an extremely optimistic measure of performance. In order to determine the reasons, the algorithm and the training and test set was looked at in more detail. It was observed that the training images generated by the sliding window (as shown in Figure 5 contains multiple instances of clear people (full line person in the image) and of sections of people. Similarly there were also samples containing just the road, just the vehicle, lampposts etc (Figure 6). The sliding window generator generated only a few images during training that contained people far away/mixed with the background/trees etc. Thus the network overfitted for these cases.

During test because of the relatively small number of images, most of the images were dominated no person present (Figure 5) (cars, empty road) or a person clearly present similar to cases shown in Figure 5.

The cases that the algorithm had trouble with (shown in Figure 6) were smaller in number. Thus the overfitted model performed well on the test set as well. To overcome this issue, using a larger sample set containing more test images is required. Other causes are also being investigated.

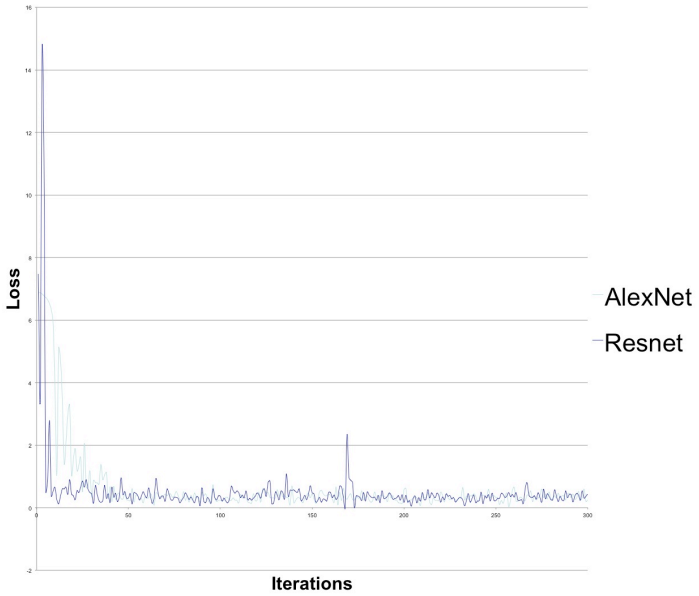


Figure 3: Training results of the CNNs

Figure 5: Images with people clearly defined (easily classified)

Figure 6: Images without people (only road etc) easily classified

Figure 4: Images with/without people that are not classified correctly

It was also observed that using a sliding window approach for feature detection resulted in the problem becoming extremely large (in terms of data size) requiring down sampling. Moreover the sliding window implemented was not smart enough to pull out people correctly(without cutting) especially in cases where the people were a small portion of the whole frame (far away from the camera). This probably resulted in the overfitting and incorrect test results. Thus for future work, moving to an R-CNN based approach where the CNN acts as both the region proposal and the feature detection methods is recommended.

## 6. Conclusions/Future Work

As part of the study, a pedestrian detection pipeline was set up using the Torch7 framework leveraging existing implementations [15,16,18,19,20]. A sliding window detector was used for region proposal and both the AlexNet and 18 layer ResNet were used for feature detection with the softmax function serving as a classifier. The problem was formulated as a binary classification problem (person detected or person not detected). Unfortunately the results obtained indicated that the model was probably overfit and because of the down sampling of the test set, test accuracies were overoptimistic. The model works fairly well for cases where people are clearly present in the frame or completely absent in the frame but doesn't really work well for cases of occluded pedestrians or for cases of small pedestrians (far from the camera).

The next steps to be performed involve training the image on a much larger training set and testing the performance on a larger test set, Moreover, in order to eliminate the problem of too many samples that occurs because of the sliding window approach, moving to a R-CNN approach, which eliminate the sliding window detector will be tried.

## 7. References


[1] Tome, D., Monti, F.,. Baroffio, L., Bondi, L., Tagliasacchi, M., Tubaro, S., "Deep convoluted neural netwoks for pedestrian detection.", Preprint. Elsevier Journal of Signal Processing: Image Communication, 2015.

[2] Dollar,. P, Wojek, C., Schiele, B., Perona, P., "Pedestrian Detection: An Evaluation of the State of the Art.", IEEE Transactions on Pattern Analysis and Machine Intelligence, ISSN:0162-8828, August 2011.

[3] http://www.vision.caltech.edu/Image_Datasets/CaltechPedestrians/

[4] Dollar, P., Wojek, C., Schiele, B., Perona, P., "Pedestrain Detection: A Benchmark", IEE

Conference on Computer Vision and Pattern Recognition, 2009
[5] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection”, IEEE Conference on Computer Vision and Pattern Recognition, 2005.
[6] Rowley, A. H., Baluja, S., Kanade.,T, ”Neural Network-Based face Detection”, IEEE Transactions on Pattern Analysis and Machine Intelligence, January 1998.
[7] Viola, P., Jones, M., “Robust Real-time Object Detection”,2nd International Workshop on Statistical and computational theories of vision, modeling , learning, computing and sampling, 2001.
[8] Sermanet, P., Kavukcuoglu, K., Chintala, S., LeCun, Y. “ Pedestrian Detection with unsupervised Multistage Feature Learning”, International Conference on Computer Vision and Pattern Recognition, 2013.
[9] Angelova, A., Krizhevsky, A., Vanhoucke, V., Ogale, A., Ferguson, D.,“Real-Time Pedestrian Detection with Deep Network Cascades”, White Paper
[10] Fukui, H., Yamashita, T., Yamauchi, Y., Fujiyoshi., H,Murase, H., “Pedestrian Detection based on Deep Convolutional Neural Network with ensemble Inference Network”, IEEE Intelligent Vehicles Symposium, 2015
[11] Molin,D., “Pedestrain Detection using convolutional neural networks”, Phd Thesis, Department of Electrical Engineering, Linkopings Universitet, Sweden, 2015
[12] Krizhevsky, A, Sutskever, I , Hinton, G.E. “ Image Net classification with deep convolutional neural networks”, Advances in neural information processing systems, 2012
[13] Szegedy, C., Liu, W, Jia, Y., Sermanet, P., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., “Going Deeper with Convolutions”,ILSVRC 2014
[14] He, K., Zhang, X., Ren, S., Sun, J., “Deep residual learning for image recognition”, ILSVRC 2015
[15] https://github.com/facebook/fb.resnet.torch
[16] http://torch.ch/blog/2016/02/04/resnets.html
[17] Li, J., Liang, X., Shen, S., Xu, T., Yan, S., ”Scale-aware Fast R-CNN for Pedestrian Detection”, arXiv preprint, arXiv:1510.08160
[18] http://cs231n.github.io/linear-classify/
[19] https://github.com/torch/torch7
[20] https://github.com/soumith/convnet-benchmarks