Pedestrian Detection Project Paper

Zhirong Jian, jianzhirong@gmail.com

1. Introduction

The goal of this project is to explore how better feature representation and various visual cues can be utilized to improve detection quality.

Specifically, this project targets the fascinating and meaningful real world problem "pedestrian detection" as a test case. Using current state of the art pedestrian detector SquaresChnFtrs [1] as a baseline, I leverage two approaches to increase detection accuracy. Expand 10 HOG+LUV channels into 20 channels by using DCT (discrete cosine transform); Encode the optical flow using SDt features (image difference between current frame T and coarsely aligned T-4 and T-8).

Note that this project is largely to reproduce observations/discovery in “Benenson etc., 2014” paper [2]. On the basis of the baseline detector, the DCT method is expected to yield 3.53% miss rate improvement and the optical flow method is expected to yield 4.47% improvement.

This is a work in progress, and the current project paper will only focus on DCT approach. It will be updated and cover optical flow approach as soon as optical flow as an additional visual cue is incorporated in the baseline detector.

2. Foundational Theories

This section will briefly mention two relevant theories upon which our experiments are founded.

2.1. Decorrelated representation Advantage

For the baseline detector SquaresChnFtrs, the underlying learning algorithm is Adboost with level 2 decision tree, and the decision tree uses orthogonal (single feature) split. Recent research [3] shows that, for Adboost learning with orthogonal split, removing correlations in the input feature representation can significantly improve learning result. Therefore it’s natural to speculate that having “decorrelated represention” might help with the detection rate of SquaresChnFtrs. This has been confirmed in [2]. It transformed the original 10 input channels in SquaresChnFtrs by DCT, and reported 3.53% miss rate improvement.

2.2. DCT

A discrete cosine transform (DCT) expresses a finite sequence of data points in terms of a sum of

cosine functions oscillating at different frequencies. Discrete cosine transform (DCT) can linearly transform data into the frequency domain, where the data can be represented by a set of coefficients. The advantage of DCT is that the energy of the original data may be concentrated in only a few low frequency components of DCT depending on the correlation in the data [4,5].

3. Experiment result

Based on SquaresChnFtrs, our new detector expands 10 HOG+LUV channels into 20 channels by using DCT basis function, with 8 x 8 pixel block. To speed up transformation computations, DCT algorithm is implemented in GPU parallel programing fashion by means of CUDA toolkit. Let's begin with baseline detector.

3.1. Baseline detector

To measure the effect of adding DCT channels, the first step is to establish the baseline detector performance statistics. Our model is re-trained with a speedy version of SquaresChnFtrs training configurations, with HOG-like feature, 3 rounds of training looking for hard negatives, and 2048 weak classifiers are finally being combined to produce a strong classifier. In my machine, I observed that the miss rate on Inria dataset is 22% , as shown in Fig 1.

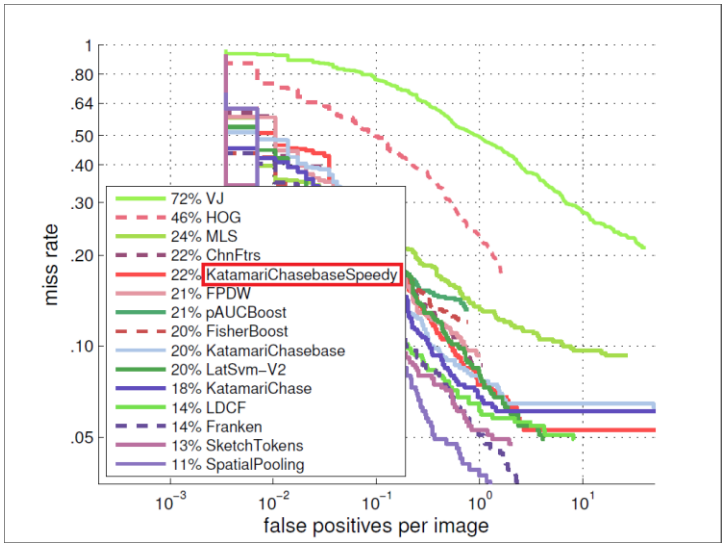

Fig 1. Baseline detector miss rate on Inria dataset (KatamariChasebaseSpeedy)

3.2. Baseline + DCT detector

With the same training configuration as above, the new detector's miss rate on Inria dataset is 23%, as shown in Fig 2.

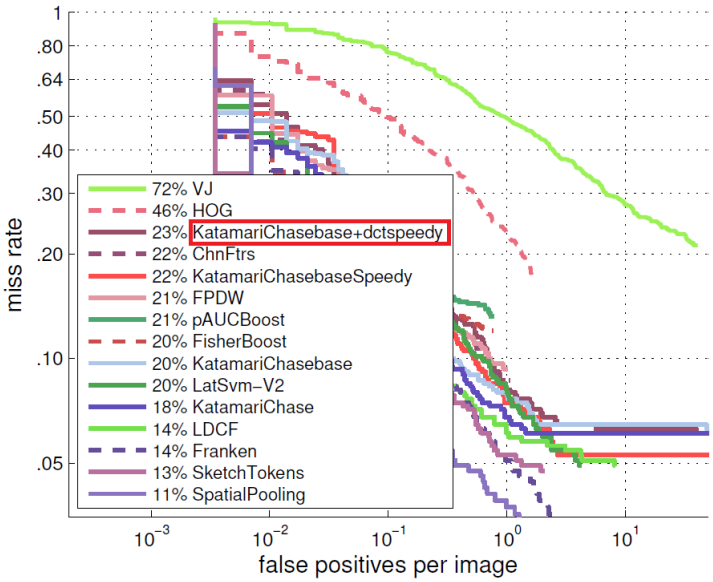

Fig 2. Baseline + DCT miss rate on Inria dataset (KatamariChasebase+dctspeedy)

Fig4 and Fig5 are the "features per channel" and "top features", they can shed some light on the final classifier being generated.

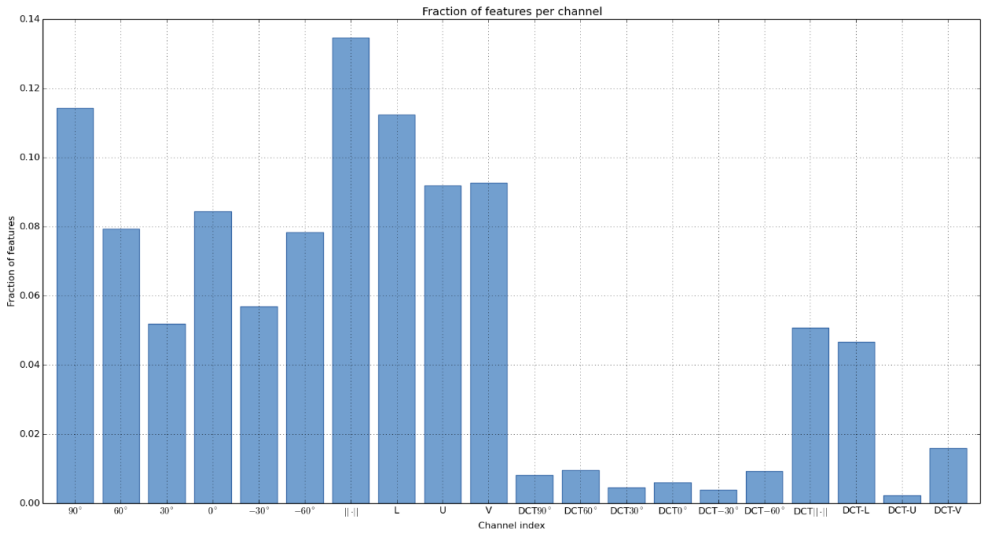

Fig 4. Features per channel. This chart indicates that, among the DCT transformed channels, gradient channel and L channel contribute most to the final classifier result.

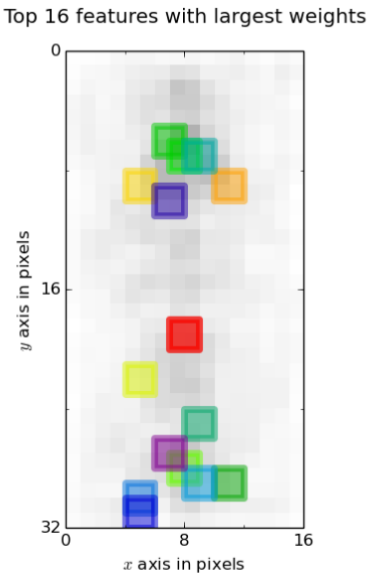

Fig5. Top features

3.3. Discussions

Two interesting points are observed during the implementation and merit brief discussions here.

1. What to transform

One way to transform the original 10 channels is to perform DCT on them as a whole. Suppose one channel has size width x height, then DCT can be performed on the whole channel whose size is width x (height *10). Another way is to transform these 10 channels individually, and add them into the final feature representation. Experiment results show that the latter one performs better. For our final baseline+dct detector, this design yields 23% miss rate, as opposed to 25% of the former one.

The explanation could be that, with images whose size are not multiples of DCT block size (8x8), some DCT computations will be done on the combined top/bottom margin of two adjacent channels, and consequently cause confusions to the detector.

2. When to transform

The feature computation workflow of baseline detector is as follows:

1) Compute 10 channels based on a single input image

2) Shrink the 10 channels by a factor of 4

3) Compute the integral image of the shrinked channels

So there are two timings when we can perform DCT, one is to perform DCT before the shrinking, and the other is after the shrinking. For the final baseline+dct detector, this former design yield miss rate of 23%, and the latter 42%.

The big gap between these two designs might be an indication that, in our current implementation, DCT transformation is a negative influencer to the final detection quality.

Intuitively speaking, the later we perform DCT, the bigger impact it will have on the final feature representation.

4. Improvement/Additional work

There are a variety of experiments/improvements that could be done with respect to DCT approach in the future. This includes:

1. Get to know how DCT approach is implemented in original paper and figure out the difference with our implementations.
2. Try adding 20 more channels by applying two more DCT variant transformations (DCT I, DCT III, or DCT IV) to the original 10 channels.
3. Try other feature decorrelation methods, like PCA, or using a single covariance matrix for each channel.

5. Conclusion

Based on open source pedestrian detector, I explored how better feature representation might improve detection quality by means of performing DCT on original feature channels. Though my implementation of this approach did not yield the performance boosting as expected, it's overall a very good learning experience.

Next I will move on to explore how to incorporate optical flow into current baseline detector as an additional visual cues to improve detection quality.

6. References

1. Doppia project repository: https://bitbucket.org/rodrigob/doppia
2. Rodrigo Benenson Mohamed Omran Jan Hosang Bernt Schiele(2014)."Ten Years of Pedestrian Detection, What Have We Learned?", EECV
3. Woonhyun Nam, Piotr Doll´ar, Joon Hee Han(2014). "Local decorrelation for improved detection". arXiv.
4. DCT tutorial: http://www.haberdar.org/Discrete-Cosine-Transform-Tutorial.htm
5. DCT wiki: http://en.wikipedia.org/wiki/Discrete_cosine_transform