Real-Time Pedestrian Detection With Deep Network Cascades

Anelia Angelova, Alex Krizhevsky, Vincent Vanhoucke
anelia,akrizhevsky,vanhoucke@google.com
Google Research
Mountain View, CA, USA

Abhijit Ogale, Dave Ferguson
ogale,daveferguson@google.com
Google X
Mountain View, CA, USA

Pedestrian detection has been an important problem for decades, given its relevance to a number of applications in robotics, including driver assistance systems, road scene understanding and surveillance systems. The two main practical requirements for fielding such systems are very high accuracy and real-time speed: we need pedestrian detectors that are accurate enough to be relied on and are fast enough to run on systems with limited compute power. This paper addresses both of these requirements by combining very accurate deep-learning-based classifiers within very efficient cascade classifier frameworks.

Deep neural networks (DNN) have been shown to excel at classification tasks [5], and their ability to operate on raw pixel input without the need to design special features is very appealing. However, deep nets are notoriously slow at inference time. In this paper, we propose an approach that cascades deep nets and fast features, that is both very fast and accurate. We apply it to the challenging task of pedestrian detection. Our algorithm runs in real-time at 15 frames per second (FPS). The resulting approach achieves a 26.2% average miss rate on the Caltech Pedestrian detection benchmark, which is the first work we are aware of that achieves high accuracy while running in real-time.

To achieve this, we combine a fast cascade [2] with a cascade of classifiers, which we propose to be DNNs. Our approach is unique, as it is the only one to produce a pedestrian detector at real-time speeds (15 FPS) that is also very accurate. Figure 1 visualizes existing methods as plotted on the accuracy - computational time axis, measured on the challenging Caltech pedestrian detection benchmark [4]. As can be seen in this figure, our approach is the only one to reside in the high accuracy, high speed region of space, which makes it particularly appealing for practical applications.

Fast Deep Network Cascade. Our main architecture is a cascade structure in which we take advantage of the fast features for elimination, VeryFast [2] as an initial stage and combine it with small and large deep networks [1, 5] for high accuracy. The VeryFast algorithm is a cascade itself, but of boosting classifiers. It reduces recall with each stage, producing a high average miss rate in the end. Since the goal is eliminate many non-pedestrian patches and at the same time keep the recall high, we used only 10% of the stages in that cascade. Namely, we use a cascade of only 200 stages, instead of the 2000 in the original work.

The first stage of our deep cascade processes all image patches that have high confidence values and pass through the VeryFast classifier. We here utilize the idea of a tiny convolutional network proposed by our prior work [1]. The tiny deep network has three layers only and features a 5x5 convolution, a 1x1 convolution and a very shallow fully-connected layer of 512 units. It reduces the massive computational time that is needed to evaluate a full DNN at all candidate locations filtered by the previous stage. The speedup produced by the tiny network, is a crucial component in achieving real-time performance in our fast cascade method.

The baseline deep neural network is based on the original deep network of Krizhevsky et al [5]. As mentioned, this network in general is extremely slow to be applied alone. To achieve real-time speeds, we first apply it to only the remaining filtered patches from the previous two stages. Another key difference is that we reduced the depths of some of the convolutional layers and the sizes of the receptive fields, which is specifically done to gain speed advantage.

Runtime. Our deep cascade works at 67ms on a standard NVIDIA K20 Tesla GPU per 640x480 image, which is a runtime of 15 FPS. The time breakdown is as follows. The soft-cascade takes about 7 milliseconds (ms). About 1400 patches are passed through per image from the fast cascade. The tiny DNN runs at 0.67 ms per batch of 128, so it can process the patches in 7.3 ms. The final stage of the cascade (which is the baseline classifier) takes about 53ms. This is an overall runtime of 67ms.

Experimental evaluation. We evaluate the performance of the Fast Deep Network Cascade using the training and test protocols established in the Caltech pedestrian benchmark [4]. We tested several scenarios by training on the Caltech data only, denoted as DeepCascade, on an independent dataset (completely disjoint from Caltech), DeepCascadeID, and on Caltech data plus some extra data from the publicly available Daimler and Eth datasets, DeepCascadeED.

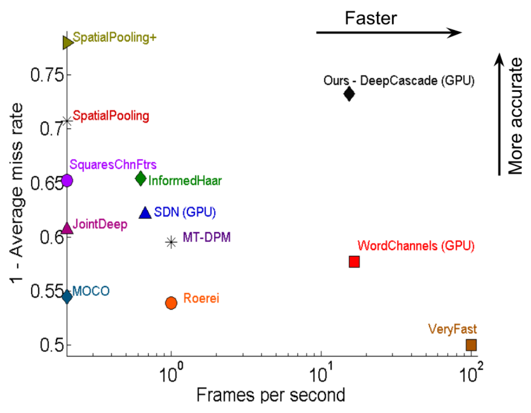

Figure 1: Performance of pedestrian detection methods on the accuracy vs speed axis. Our DeepCascade method achieves both smaller miss-rates and real-time speeds. Methods for which the runtime is more than 5 seconds per image, or is unknown, are plotted on the left hand side. The SpatialPooling+/Katamari methods use additional motion information.

Our method performs with average miss rates of 31.11%, 30.17%, and 26.21%, for DeepCascade, DeepCascadeID and DeepCascadeED respectively, and outperform most approaches. The best approaches, known on this dataset, perform at 36% [1], 29% [6] and at 22% for SpatialPooling+ and Katamari [3, 6] which use additional motion features. Our results also point to the strengths of DNNs, namely, achieving higher accuracy by simply incorporating more and higher quality data.

Apart from achieving very good accuracy, our method is much faster and runs at 15 FPS, which is real-time performance. Other real-time algorithms, we are aware of, VeryFast at 100 FPS and WordChannels at 16 FPS (on GPU) have high average miss rate of 50% and 42%, respectively. Previous methods, e.g. SDN, JointDeep have similarly HOG-based cascade and a deep network have runtime of at least 1-1.5 seconds on GPU. We further note that our algorithm is implemented using the publicly available 'cuda-convnet2' [5] and the VeryFast 'Doppia' code [2].

Conclusion. The main contribution of this work is a pedestrian detection system that is both accurate and runs in real-time. As such, it can be practically deployed within a real-life pedestrian detection system. No other prior work has demonstrated such capabilities. We expect our work to impact future methods by providing a simple to implement, accurate and effective real-time solution. Thus, future methods can continue to further push the boundaries in accuracy in pedestrian detection, while simultaneously keeping the methods fast and practically relevant.

[1] A. Angelova, A. Krizhevsky, and V. Vanhoucke. Pedestrian detection with a large-field-of-view deep network. *ICRA*, 2015.

[2] R. Benenson, M. Matthias, R. Tomofte, and L. Van Gool. Pedestrian detection at 100 frames per second. *CVPR*, 2012.

[3] R. Benenson, M. Omran, J. Hosang, and B. Schiele. Ten years of pedestrian detection, what have we learned? *2nd Workshop on Road scene understanding and Autonomous driving, ECCV*, 2014.

[4] P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. *CVPR*, 2009.

[5] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. *NIPS*, 2012.

[6] S. Paisitkriangkrai, C. Shen, and A. van den Hengel. Strengthening the effectiveness of pedestrian detection. *ECCV*, 2014.