레터논문 (Letter Paper)
방송공학회논문지 제24권 제2호, 2019년 3월 (JBE Vol. 24, No. 2, March 2019)
https://doi.org/10.5909/JBE.2019.24.2.357
ISSN 2287-9137 (Online) ISSN 1226-7953 (Print)

임베디드 GPU에서의 딥러닝 기반 실시간 보행자 탐지 기법

비엔 지아 안[a)], 이 철[a)‡]

Deep Learning-Based Real-Time Pedestrian Detection on Embedded GPUs

An Gia Vien[a)] and Chul Lee[a)‡]

요 약

본 논문은 임베디드 GPU에서 실시간 동작하는 딥 컨볼루션 뉴럴 네트워크(CNN) 기반의 보행자 탐지 기법을 제안한다. 제안하는 기법에서는 먼저 영상 내 보행자 크기에 대한 통계적 분석을 통해서 최적의 컨볼루션 층의 개수를 결정한다. 또한, 본 논문에서는 다중 스케일 CNN 학습 기법을 적용하여 영상 내의 보행자 크기 변화에 강인한 탐지 기법을 개발한다. 컴퓨터 모의실험을 통해 제안하는 알고리즘이 임베디드 GPU에서 실시간 동작하면서도 기존의 기법과 비교하여 평균적으로 높은 정확도를 보임을 확인한다.

Abstract

We propose an efficient single convolutional neural network (CNN) for pedestrian detection on embedded GPUs. We first determine the optimal number of the convolutional layers and hyper-parameters for a lightweight CNN. Then, we employ a multi-scale approach to make the network robust to the sizes of the pedestrians in images. Experimental results demonstrate that the proposed algorithm is capable of real-time operation, while providing higher detection performance than conventional algorithms.

Keyword : Pedestrian detection, convolutional neural network, embedded system

a) 부경대학교 컴퓨터공학과(Pukyong National University, Department of Computer Engineering)
‡ Corresponding Author : 이철(Chul Lee)
E-mail: chullee@pknu.ac.kr
Tel: +82-51-629-6228
ORCID: https://orcid.org/0000-0001-9329-7365
※ 이 논문은 부경대학교 자율창의학술연구비(2017년)에 의하여 연구되었음.
※ This work was supported by a Research Grant of Pukyong National University(2017).
・ Manuscript received February 20, 2019; Revised March 8, 2019; Accepted March 8, 2019.

Ⅰ. Introduction

Pedestrian detection is an essential component of automotive safety, robotics, and intelligent surveillance, which has recently attracted broad attention in both academia and industry [1-3]. One of the most important challenges in this task is the variation of pedestrians in the background and the size of objects in the image.

Many algorithms have been developed to address these challenges that use hand-crafted features [4, 5]. These algorithms extract the features, such as HOG-LBP [5], that capture the most discriminative information of pedestrians. Then, a classifier decides if a bounding box is detected as a pedestrian. Support vector machine (SVM) and random forests are often applied in these approaches.

Recently, deep learning has achieved high performance in general object detection problems [6-10]. However, most deep learning-based approaches focus on improving accuracy by applying more convolution layers or integrated region proposal networks as in Faster R-CNN [11]. Therefore, both the size and complexity of these algorithms are prohibited to be employed in practical applications. A few algorithms have been recently developed using an effective single-shot convolutional neural network (CNN) for general object detection, e.g., YOLO [9] and YOLOv2 [10]. YOLO consists of an end-to-end network to predict both object locations and classification probabilities. In [9], a simpler version is also proposed to process images for real-time applications, which is called tiny YOLO. YOLOv2 is an improved model of YOLO with various improvements. Despite their success, YOLO, tiny YOLO, and YOLOv2 are optimized for general object detection on Pascal VOC and COCO datasets. Furthermore, YOLO and YOLOv2 are deep CNNs that contain an enormous number of parameters.

To address the aforementioned issues of the deep learning-based pedestrian detection models, we develop a real-time pedestrian detector for embedded systems that is as accurate as YOLOv2 and more robust to pedestrian size variations. To this end, we simplify the network and optimize hyper-parameters to make the representation easier to learn. Specifically, we first analyze the scale variations of the pedestrians in images, and determine the optimal number of convolution layers in the network. Then, we employ multi-scale training techniques, making the detector robust to object sizes. Experimental results on the Caltech dataset demonstrate that the proposed model progresses at 300 fps on Nvidia Titan X GPU and 30 fps on Jetson TX2 embedded GPU, while providing higher performance than tiny YOLO and YOLOv2.

Ⅱ. Method

We develop a unified approach to predict multiple bounding boxes and class probabilities for pedestrian detection by a single CNN. The proposed network is extended from tiny YOLO to optimize end-to-end for pedestrian detection.

1. Network Architecture

To design an effective CNN architecture, we first analyze the characteristics of the pedestrian dataset. The sizes of the pedestrians in the dataset can be categorized into three groups according to the height in pixels: small scale (30 pixels or less), medium scale (between 30 and 140 pixels), and large scale (140 or more pixels) [1]. Also, we notice that most pedestrians are observed at small and medium scales [1].

Based on the analysis, to increase the detection performance for pedestrians in small and medium scales, the proposed network consists of two stages, i.e., feature extraction and prediction. The feature extraction stage includes convolution layers with max pooling, batch norm [12], and leaky ReLU [13] after each convolution as an activation function. We determine four (conv1~4) as the optimal number of convolution layers in the feature extraction stage to increase the size of the grid cell without enlarging the number of parameters and the size of the network.

Hence, we achieve a larger grid cell and less computation time. Table 1 summarizes the full details of the proposed network.

표 1. 제안하는 네트워크의 파라미터 요약
Table 1. Summary of the proposed network layers

Layer name	Layer size	Filter size/stride	Number of parameters
Input	$416\times416\times3$		
Conv1	$416\times416\times16$	$3\times3/1$	448
Pool1	$208\times208\times16$	$2\times2/2$	
Conv2	$208\times208\times32$	$3\times3/1$	4,640
Pool2	$104\times104\times32$	$2\times2/2$	
Conv3	$104\times104\times64$	$3\times3/1$	18,496
Pool3	$52\times52\times64$	$2\times2/2$	
Conv4	$52\times52\times128$	$3\times3/1$	73,856
Pool4	$52\times52\times128$	$2\times2/1$	
Conv5	$52\times52\times256$	$3\times3/1$	295,168
Conv6	$52\times52\times256$	$3\times3/1$	590,080
Conv7	$52\times52\times18$	$1\times1/1$	4,626

2. Training Settings

To make the proposed network robust to the size variations of pedestrians, we change the sizes of images in the training data after every few iterations. Specifically, after every ten batches in training, the network randomly selects a new image size. Following [10], the smallest size is 320×320, and the largest size is 608×608. We used the Darknet deep learning library [14] to implement the proposed network. The model is trained on the Caltech dataset [1] with stochastic gradient descent (SGD) optimizer on Nvidia Titan X GPU. We trained the network with the loss function in [9].

III. Experimental Results

We evaluate the performance of the proposed algorithm on the Caltech dataset, and then compare our results with those of tiny YOLO [9] and YOLOv2 [10]. We also evaluate the computational complexity of the proposed algorithm on two different GPUs: Nvidia Titan X and Nvidia Jetson TX2.

We evaluate the accuracy of pedestrian detection using the recall metric, which is the positive predictive value, given by

$$Recall = \frac{tp}{tp+fn}\times 100 \tag{1}$$

where tp and fn denote a true positive value and false negative value, respectively. To compare the performance of bounding box prediction, we employ the intersection over union (IoU) metric that computes the ratio between intersection region and predicted bounding boxes and references, defined as

$$IoU = \frac{Areaof Overlap}{Areaof Union}\times 100. \tag{2}$$

표 2. Caltech 데이터셋을 이용한 recall 및 IoU 성능 비교.
Table 2. Comparison of the detection performance using recall and IoU on the Caltech test dataset.

Model	Small scale		Medium scale		Large scale		Average	
	Recall	IoU	Recall	IoU	Recall	IoU	Recall	IoU
tiny YOLO	18.31	28.85	35.71	40.62	82.76	63.41	32.16	38.13
YOLOv2	13.80	25.19	29.28	36.42	72.41	57.38	26.51	34.05
Proposed	37.75	44.96	62.44	53.92	65.52	54.40	57.73	52.08

Table 2 compares the detection accuracy in terms of the recall and IoU performance for different sizes of pedestrians. If the predicted bounding boxes overlap with the references, IoU values are close to 100. Otherwise, the values are close to 0. Table 2 shows that, while tiny YOLO and YOLOv2 provide higher detection rates for large scale, the proposed algorithm outperforms tiny YOLO and YOLOv2 for medium and small scales. This is because the proposed network is designed to be shallow to improve the computational efficiency. Therefore, it learns low- and medium-level features to detect the pedestrians at small and medium scales effectively, while providing lower performance for large-scale pedestrians.

표 3. YOLOv2, tiny YOLO 및 제안하는 기법의 속도 비교.
Table 3. The computation speed in fps of YOLOv2, tiny YOLO, and the proposed algorithm

Model	Frames per second (fps)	
	Titan X	Jetson TX2
YOLOv2	67	8
tiny YOLO	207	27
Proposed	290	29.5

표 4. 모델 크기 및 네트워크 파라미터 수 비교
Table 4. Comparison of the model size and network parameters.

Model	Parameters	Size (MB)
YOLOv2	70,000,000	280
tiny YOLO	12,000,000	48
Proposed	987,314	3.95

Finally, we compare the computational and storage efficiency in terms of computation time and model size in Tables 3 and 4, respectively. The computation time and storage of the proposed model are faster and smaller than tiny YOLO and YOLOv2. Specifically, the proposed algorithms is 1.4 times faster than tiny YOLO and 4.5 times faster than YOLOv2. For model size, the proposed network is 12.2 times smaller than tiny YOLO and 70.9 times smaller than YOLOv2. To summarize, the proposed detector is capable of real-time operation on resource-constrained systems, while providing accurate detection and robust to pedestrian size variations.

IV. Conclusions

We proposed a fast single CNN for pedestrian detection on embedded GPUs in this work. First, we determined the optimal number of convolution layers in the network based on the statistical analysis of pedestrians. Then, to process images of different sizes, we employ a multi-scale approach to train the network with different sizes of images. Experimental results demonstrated that the proposed algorithm outperforms tiny YOLO and YOLOv2 in terms of the average recall and IoU scores. We also showed that the proposed algorithm is capable of real-time operation on embedded GPUs.

참 고 문 헌 (References)

[1] P. Dollar, C. Wojek, B. Schiele, and P. Perona, “Pedestrian detection: An Evaluation of The State of The Art,” *IEEE Transaction Pattern Analysis and Machine Intelligence*, Vol. 34, No. 4, pp. 743-761, April 2012.

[2] P. Sermanet, K. Kavukcuoglu, S. Chintala, and Y. Lecun, “Pedestrian Detection with Unsupervised Multi-Stage Feature Learning,” *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3626-3633, June 2013.

[3] P. Dollar, R. Appel, S. Belongie, and P. Perona, “Fast Feature Pyramids for Object Detection,” *IEEE Transaction Pattern Analysis and Machine Intelligence*, Vol. 36, No. 8, pp. 1532-1545, August 2014.

[4] X. Wang, T. X. Han, and S. Yan, “An HOG-LBP Human Detector with Partila Occlusion Handling,” *Proceeding of IEEE Conference on Computer Vision*, pp. 32-39, September 2009.

[5] S. Zhang, C. Bauckhage, and A. B. Cremers, “Informed Haar-Like Features Improve Pedestrian Detection,” *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 947-954, June 2014.

[6] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” *Proceedings of International Conference on Learning and Representations*, May 2015.

[7] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going Deeper with Convolutions,” *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-9, June 2015.

[8] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770-778, June 2016.

[9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779-788, June 2016.

[10] J. Redmon and A. Farhadi, “YOLO9000: Better, Faster, Stronger,” *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7263-7271, July 2017.

[11] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” *Proceeding International Conference on Neural Information Processing Systems*, pp. 91-99, December 2015.

[12] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” *Proceeding of International Conference on Machine Learning*, pp. 448-456, July 2015.

[13] A. L. Maas, A. Y. Hannun, and A. Y. Ng, “Rectifier Nonlinearities Improve Neural Network Acoustic Models,” *International Conference on Machine Learning Workshop on Deep Learning for Audio, Speech, and Language*, 2013.

[14] J. Redmon, “Darknet: Open Source Neural Network in C,” http://pjreddie.com/darknet/, 2013-2016.