YOLO 네트워크와 추적 기법을 이용한 보행자 검출

*이상훈 **조남익

서울대학교

*tkd1088@ispl.snu.ac.kr **nicho@snu.ac.kr

Pedestrian Detection using YOLO and Tracking

*Lee, Sang-Hoon **Cho, Nam Ik

Seoul National University

요약

최근 딥 러닝의 발전과 함께 보행자 검출 기술의 성능이 발전하면서 다양한 분야에서 응용되고 있다. 영상 내 보행자의 위치나 움직임을 파악함으로써 위험 지역이나 보안 지역에 접근하는 보행자를 찾아낼 수 있다. 일반적인 딥 러닝 기반의 물체 검출기는 멀리 있는 보행자와 같은 작은 물체를 검출 하는 데에 적합하지 않다. 또, 검출을 수행하기 위해서 큰 계산량을 필요로 하기 때문에, 동영상의 매 프레임 마다 수행하기 부적합 하다는 단점이 있다. 본 논문에서는 작은 물체도 잘 검출할 수 있도록 기존 YOLO 네트워크의 구조를 변경하고, 보행자 데이터를 이용하여 추가로 학습함으로써 보행자를 검출하는 성능을 증가시켰다. 그리고 검출한 보행자들에 대해 추적 기법을 이용함으로써, 동영상의 매 프레임 마다 검출을 수행하는 것을 피할 수 있도록 하였다. 실제로 DukeMTMC Dataset을 이용하여 실험을 해본 결과, YOLO 네트워크의 구조를 변경하고 추가 학습을 함으로써 검출 정확도가 개선되는 것을 확인할 수 있었다. 또, 추적 기법을 이용했을 때, 성능이 크게 떨어지지 않으면서 검출 속도를 개선할 수 있는 것을 확인할 수 있었다.

1. 서론

보행자 검출 기술은 그림 1과 같이 영상으로부터 서 있거나 걷고 있는 사람의 영역을 검출하는 기술이다. 이 기술은 다양한 분야에서 사용된다. 위험 지역이나 보안 지역에는 감시카메라가 설치되어 있는 경우가 많은데, 이 감시카메라에 의해 촬영된 영상에서 보행자를 검출할 수 있다. 검출한 보행자의 위치를 계산함으로써 해당 지역에 접근하려는 보행자를 검출해 낼 수 있고, 보행자의 움직임을 분석함으로써 경계해야할 보행자를 구분해 낼 수 있다.

그림 1 보행자 검출 기술

초기의 보행자 검출 기술은 특징벡터를 추출하고, 추출된 특징벡터들을 보행자인지 아닌지 분류하는 방식으로 수행되었다. [1]에서는 HOG 특징벡터를 추출하였고, SVM을 이용하여 특징벡터들을 분류하였다. HOG란 Histogram of Oriented Gradients의 약자로, 일정 영역 내에서 여러 가지 방향에 대한 Gradient를 구하고, 그 분포를 특징벡터로 이용하는 것이다. [2]에서는 YUV채널의 밝기 값, 여러 가지 방향에 대한 gradient의 크기 등 여러 가지 정보를 포함한 총 10차원의 특징벡터를 이용하였다.

최근에는 딥 러닝의 발전과 함께 딥 러닝을 이용하는 방법들이 많이 등장하고 있다. 더 잘 설계된 네트워크를 학습함으로써, 더 좋은 특징벡터를 추출하고 더 정확한 분류가 가능해지고 있다. [3]은 특징벡터들을 분류할 때, 보행자 인지 아닌지만 분류하는 것이 아니라 보행자라면 어떤 자세인지 보행자가 아니라면 어떤 물체인지 등 좀 더 세부적인 분류를 통해 보행자 검출 성능을 높이는 방법을 사용했다. [4]는 보행자를 한 번에 검출하는 것이 아니라, 보행자의 다양한 부분을 검출한 후 결과들을 조합하여 최종적으로 검출하는 방법을 이용하였다. 이 방법은 보행자의 특정 부분이 가려져 있더라도 잘 검출할 수 있다.

추적 기법은 연속적인 동영상 프레임 내에서 시간의 경과에 따라 움직이는 물체를 찾는 것이다. 이전 프레임에서 물체의 위치를 알고 있을 때, 다음 프레임에서 같은 물체의 위치를 찾아내는 것이다. [5]와 같이 점 기반의 추적을 할 수도 있고, [6]과 같이 커널 기반의 추적을 할 수 도 있다. 추가로, 물체의 윤곽선을 추적하는 방법도 존재한다.

본 논문에서는 보행자 검출의 검출 정확도를 개선하고, 속도를 향상시킬 수 있는 방법을 제안하고자 한다. 먼저, 검출 정확도를 개선하기 위해 YOLO 네트워크의 구조를 변경하고 추가 학습을 실시하였다. YOLO 네트워크의 구조를 변경함으로써 보행자와 같이 세로로 긴 물체를 잘 검출할 수 있도록 하였고, 멀리 있는 보행자처럼 작은 물체도

잘 검출할 수 있도록 하였다. 또, 보행자 영상 데이터를 이용하여 추가 학습을 함으로써 다른 물체들 보다 보행자를 좀 더 집중적으로 검출해 낼 수 있도록 하였다. 두 번째로, 속도를 향상시키기 위해 추적 기법을 이용하였다. 동영상의 매 프레임마다 검출을 수행하는 경우 긴 수행시간을 필요로 한다. 그래서 일정 주기의 프레임에 대해서만 검출을 수행하고, 그 사이에 존재하는 프레임에 대해서는 추적을 통해 보행자의 위치를 파악하고자 하였다. DukeMTMC Dataset에 대해 실험해본 결과 YOLO 네트워크의 구조 변경과 추가학습을 통해 보행자 검출 정확도가 증가하는 것을 알 수 있었고, 추적 기법의 적용을 통해 검출 정확도가 크게 감소하지 않으면서 수행 속도를 대폭 향상시킬 수 있었다.

2. YOLO를 이용한 보행자 검출

[7]의 YOLO 네트워크는 그림 2와 같이 입력된 영상을 19*19개의 큰 그리드로 분할한 후, 각각의 그리드에 해당하는 물체의 위치와 범위 그리고 종류를 출력하는 네트워크이다. [8]의 R-CNN과 함께 영상 내

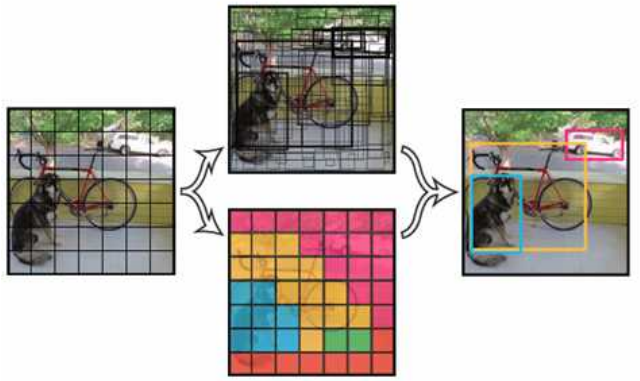

그림 2 YOLO 네트워크를 이용한 물체 검출

에서 물체를 검출하는데 있어서 최고의 성능을 가지고 있다.

그런데, 그림 3의 위쪽 그림을 보면 YOLO 네트워크가 작은 크기의 보행자를 검출하지 못하는 것을 알 수 있다. YOLO 네트워크를 이용하여 Full-HD 크기의 영상에서 보행자를 검출할 때, 두 가지 문제점이 발생한다. 첫 번째 문제는 보행자와 같이 가로길이가 짧은 물체를 검출하기 힘들다는 것 이고, 두 번째 문제는 멀리 있는 보행자와 같이 작은 물체를 검출하기 힘들다는 것이다. YOLO 네트워크는 입력 영상을 19*19 개의 그리드로 분할하게 되는데, 입력 영상의 가로 길이를 세로 길이와 같아지도록 축소시킨 후 분할을 수행한다. 이 과정에서 보행자의 가로 길이는 더 짧아지게 되고, 보행자를 검출하기 힘들어 진다. 또, 멀리 있는 보행자는 그 크기가 그리드의 크기에 비해 아주 작기 때문에 검출하기 힘들어 진다.

앞에서 설명한 문제를 해결하기 위해 YOLO 네트워크의 구조를 변경하였다. 먼저, 그리드의 수를 영상의 비율에 맞게 계산하여 설정하였다. 가로 길이가 세로 길이보다 긴 영상이라면 영상을 나눌 때, 세로보다 가로로 더 많이 나누는 것이다. 즉, 1920*1080 크기의 영상을 19*19개의 그리드로 나누는 것이 아니라 34*19개로 나누도록 하는 것이다. 이렇게 하면 영상의 가로 세로 비율을 조절하는 과정에서 정보가 손실되는 것을 막을 수 있다. 두 번째로, 그리드의 크기를 축소시켰다. 영상을 좀 더 많은 개수의 그리드로 나눔으로써, 작은 보행자도 검출할

그림 4 YOLO 네트워크의 보행자 검출 결과 비교
변경 전(위)과 변경 후(아래)

수 있도록 한 것이다. 예를 들어 1920*1080 크기의 영상은 60*34개의 그리드로 나누도록 하였다. 그림 3의 아래쪽 그림을 보면 구조가 변경된 YOLO 네트워크가 아주 작은 보행자를 검출해 내는 것을 확인할 수 있다.

기존의 YOLO 네트워크는 최대 80개의 물체를 검출 및 분류해낼 수 있도록 설계되었는데, 보행자 검출 정확도를 향상시키기 위해서 출력 단자를 수정하였다. 원래 물체의 종류가 어떤 것인지 출력하게 되어 있지만, 그 대신 보행자인지 아닌지의 여부만 출력하도록 수정하였다. 그 후, DukeMTMC Dataset의 총 8개 영상 중 1번부터 4번까지의 보행자 영상을 이용하여 추가로 학습을 진행하였다.

3. 추적 기법 적용

보행자 영상은 감시카메라와 같이 고정된 카메라에 의해 촬영된 영상이 많기 때문에, 영상의 배경이 변하지 않고 고정되어 있는 경우가 많다. 또, 대부분의 보행자는 멈추어 있거나 걷고 있기 때문에, 보행자의 모습이 단 시간 내에 크게 변하지 않는다. 따라서 보행자가 어느 순간 잘 검출된다면, 이후에는 검출 대신 추적을 이용하여 보행자의 위치를 파악할 수 있다. 그림 3을 보면 실제로 임의의 순간에 보행자 검출을 수행하고, 검출한 보행자들을 4초 동안 추적 해본 결과 각 보행자의 위치를 잘 추적하고 있음을 알 수 있다. 따라서 그림 4의 위와 같이 영상의 매 프레임에 대해 보행자 검출을 수행하는 대신 그림 4의 아래와

그림 3 보행자 검출 결과와 4초 동안 추적한 결과

그림 5 매 프레임 검출하는 방법(위)과
30프레임을 주기로 검출하는 방법(아래)

같이 일정한 주기마다 검출을 수행하고, 그 사이의 프레임에 대해서는 추적을 적용하더라도 보행자의 위치를 파악하는 것이 가능하다. 이렇게 하면 보행자 검출 성능은 크게 떨어지지 않으면서 속도를 크게 향상시킬 수 있다. 본 논문에서는 물체의 색상 정보를 이용하는 커널 기반의 추적 방법인 KCF[9]를 사용하여 추적을 수행하였다.

4. 실험 결과

DukeMTMC Dataset의 보행자 영상들을 이용하여 총 세 가지 방법의 보행자 검출 실험을 해 보았다. 첫 번째는 기존의 YOLO 네트워크를 그대로 이용하여 보행자를 검출한 것이다. 기존의 YOLO 네트워크를 그대로 적용하여도 아주 높은 precision과 recall을 얻을 수 있었고 한 프레임을 처리하는데 평균 0.07초가 소요되는 것을 알 수 있었다. 두 번째 실험은 본 논문에서 제안한 방법으로 변경한 YOLO 네트워크를 이용하여 보행자를 검출한 것이다. 기존의 YOLO 네트워크가 매우 높은 precision과 recall을 얻을 수 있음에도 불구하고, 변경한 YOLO 네트워크는 더 높은 precision과 recall을 얻을 수 있었다. 하지만, 네트워크가 처리해야할 그리드 개수가 증가하는 바람에 평균 처리 속도가 0.25초로 증가하였다. 마지막 실험은 변경된 YOLO 네트워크를 이용하되 120프레임마다 검출을 수행하고, 그 사이의 프레임들은 추적을 이용하는 것이다. precision과 recall이 두 번째 실험에 비해 감소하는 것을 알 수 있다. precision의 경우 기존 YOLO 네트워크를 이용했을 때 보다 조금 낮게 측정되었다. 하지만, recall값은 여전히 높으며, 두 번째 실험의 8배, 첫 번째 실험의 2배 이상 빠른 수행속도가 측정 되었다.

	YOLO 기존 (매 프레임 검출)	YOLO 변경 (매 프레임 검출)	YOLO 변경 (추적 기법 이용)
precision	0.994	0.999	0.991
recall	0.950	0.967	0.963
time/frame	0.070	0.250	0.030

표 1 세 가지 방법을 이용한 보행자 검출 성능 측정 결과

5. 결론

본 논문에서는 YOLO 네트워크와 추적 기법을 이용하여 보행자를 검출하는 방법에 대해 제안하였다. 먼저, 기존의 YOLO 네트워크가 보행자를 검출 할 때 나타내는 문제점을 분석한 후, 네트워크의 구조를 변경하였다. 그 후, 보행자를 매 프레임 마다 검출하는 대신 일정 주기마다 검출하고, 그 사이에는 추적을 이용하여 보행자의 위치를 파악하는 방법을 이용 하였다. 첫 번째 방법을 이용하여 보행자를 검출하는 성능을 증가시킬 수 있었고, 두 번째 방법을 통해 성능을 크게 감소시키지 않은 채 검출 속도를 증가시킬 수 있었다.

감사의 글

본 연구는 과학기술정보통신부 및 정보통신기술진흥센터의 대학 ICT연구센터육성지원사업의 연구결과로 수행되었음
(IITP-2018-2016-0-00288)

참고문헌

[1] Dalal, Navneet, and Bill Triggs. "Histograms of oriented gradients for human detection." 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). Vol. 1. IEEE, 2005.

[2] Dollár, Piotr, et al. "Fast feature pyramids for object detection." IEEE Transactions on Pattern Analysis and Machine Intelligence 36.8 (2014): 1532-1545.

[3] Tian, Yonglong, et al. "Pedestrian detection aided by deep learning semantic tasks." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015.

[4] Girshick, Ross, et al. "Deformable part models are convolutional neural networks." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015.

[5] Arulampalam, M. Sanjeev, et al. "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking." IEEE Transactions on signal processing 50.2 (2002): 174-188.

[6] Comaniciu, Dorin, Visvanathan Ramesh, and Peter Meer. "Real-time tracking of non-rigid objects using mean shift." Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on. Vol. 2. IEEE, 2000.

[7] Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[8] Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." Advances in neural information processing systems. 2015.