물체 탐지 알고리즘을 활용한 블랙박스 영상 내 사고 위험 감지 시스템

홍진석 · 한명우 · 김정선 · 김경섭*

충남대학교

The Accident Risk Detection System in Dashcam Video using Object Detection Algorithm

Jin-seok Hong · Myeong-woo Han · Jeong-seon Kim · Kyung-sup Kim*

Chungnam National University

E-mail : home0085@naver.com / auddn9876@naver.com / yui1541@naver.com / sclkim@cnu.ac.kr

요 약

본 논문에서는 물체 탐지 알고리즘 중 하나인 Faster R-CNN과 컴퓨터 비전을 목적으로 한 프로그래밍 라이브러리인 OpenCV를 사용하여 차선 변경이 가능한 고속도로나 국도, 일반 도로 등의 블랙박스 영상에서 다른 차량이 자신의 차선으로 차선 변경을 시도할 때 위험을 감지 할 수 있는 시스템을 구현하였다. 또한, 구현한 시스템의 성능을 평가하여 성능이 나쁘지 않음을 증명하였다.

ABSTRACT

In this paper, we use Faster R-CNN that is one of object detection algorithm and OpenCV that purposes computer vision, to implement the system that can detect danger when a vehicle attempts to change lanes into its own lane in videos of highway, national road, general road and etc. Also, the performance of implemented system is evaluated to prove that the performance is not bad.

키워드

Faster R-CNN, OpenCV, Resnet, 경계 박스

Ⅰ. 서 론

2010년 이후 딥 러닝이 많은 관심을 받으면서 다양한 분야에서 이용하려는 연구가 활발히 진행중에 있다. 딥 러닝은 여러 비선형 변환기법의 조합을 통해 높은 수준의 추상화를 시도하는 기계학습 알고리즘의 집합이다. 컴퓨터가 학습할 수 있게 하도록 많은 연구가 진행되고 있으며, 이러한 노력의 결과로 Deep Neural Network, Convolutional Deep Neural Network, Recurrent Neural Network와 같은 다양한 딥 러닝 기법들이 컴퓨터 비전, 음성인식, 자연어 처리, 음성/신호처리 등의 분야에서 적용되어 최첨단 결과들을 보여주고 있다.

자동차 분야에서도 딥 러닝 기법을 활용하여 많은 연구 및 개발이 이루어지고 있다. 딥 러닝 기법을 적용한 대표적인 사례로 자율주행 자동차[1]를 들 수 있는데, Google, Tesla, NVIDIA 등 많은 기업에서 자율주행 자동차 개발을 진행하고 있으며, 실제 일반 도로에서 완전 자율주행에 성공하고 있다. 또, 자동차 충돌 사고를 탐지[2]하거나, 사고를 예측[3]하는 등 많은 연구가 진행 중이다.

* corresponding author

본 논문에서는 주행을 하는 차량에 대해 학습을 진행하고 이를 토대로 블랙박스 영상을 통하여 측면 충돌이 발생할 수 있는 다른 차량이 자신의 차선으로 끼어드는 상황에 대해 위험을 감지하는 시스템을 만드는 것을 목적으로 한다.

2장에서는 자동차 위험 감지 모델을 구성하는 훈련 모듈인 Faster R-CNN[5]에 대한 설명과 신경망을 구성하는 ResNet[4]에 대한 설명을 서술했으며, 3장은 시스템 설계에 대한 내용이 서술되어 있다. 시스템 설계는 데이터 수집 및 전 처리 과정, 학습 모듈, 차선 탐지, 시각화 순으로 이루어져 있다. 4장은 개발한 시스템에 대한 성능 평가에 관하여 서술되어 있다. 마지막으로 5장에서는 시스템의 성능을 향상시키기 위한 향후 계획과 발전 가능성에 대해 논의한다.

II. 연구 배경

2.1 ResNet(Residual Network)[4]

2015년에 제안된 모델로 이미지 인식분야에서 사용되고 있는 딥 네트워크 모델이다. 레이어의 깊이를 늘렸을 때 생기는 부작용을 해결한 모델로 기존의 평범한 CNN[11]에서 레이어의 입력을 레이어의 출력에 바로 연결시키는 Skip Connection을 사용하여 학습을 진행한다. 그림1은 ResNet의 구조를 나타낸 것이며, ResNet은 F(x)가 0이 되는 방향으로 학습을 진행한다. 이러한 구조로 ResNet은 깊은 레이어도 쉽게 최적화가 가능하고, 늘어난 깊이로 인해 정확도를 개선할 수 있다.

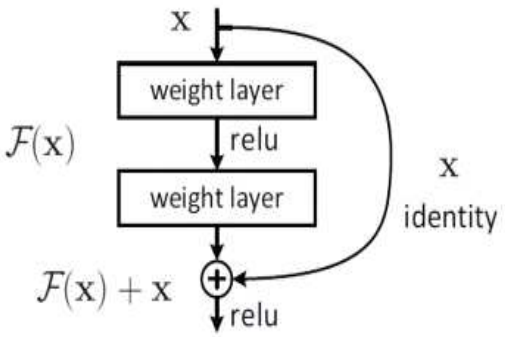

그림 1. ResNet 구조

2.2 Faster R-CNN(Faster Region-based Convolutional Neural Networks)[5]

이미지 안에 어떤 물체들이 들어있는지 구분하기 위해 이미지 분류기 앞에 이미지 영역을 찾아내는 탐지 영역을 결합한 모델이다. 그림2는 Faster R-CNN의 구조를 나타낸 것이다.

Faster R-CNN의 훈련은 4단계의 교대훈련으로 이루어져 있다. 1단계는 미리 훈련된 데이터 셋을 통해 RPN을 훈련시킨다. 2단계는 1단계에서 제안된 박스를 가지고 RoI Layer를 훈련시킨다. 3단계는 2단계를 바탕으로 RPN을 훈련시키며 이때 특징맵을 추출하는 레이어는 고정된다. 마지막으로 3단계 결과를 가지고 RoI Pooling Layer를 훈련시킨다.

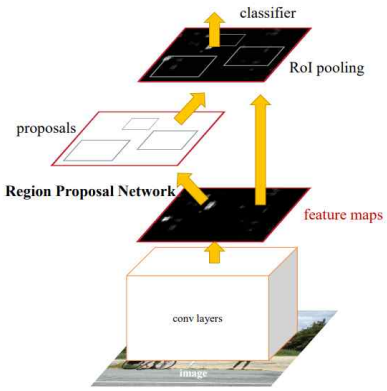

그림 2. Faster R-CNN 구조

III. 시스템 설계

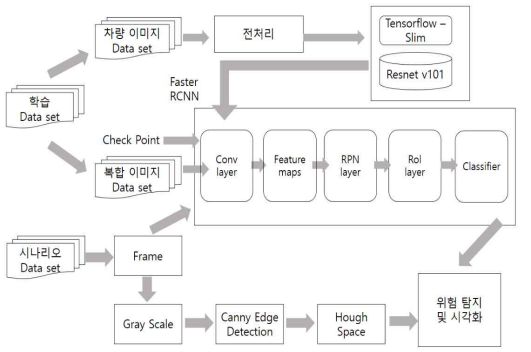

그림 3. 시스템 구성도

그림3은 전체적인 시스템의 구성도를 나타낸 것이다. 본 위험 감지 시스템을 위해 차량, 사람 등이 존재하는 복합 이미지에 라벨링 작업을 진행하여 데이터 셋을 만드는 전처리 과정을 거쳤다. 전처리된 데이터 셋을 훈련시키기 위해 Faster R-CNN(Faster Region-based Convolutional Nueral Network)[5] 모델을 사용하여 학습을 진행하였으며, 이미지 분류 모델로 Resnet101[4]을 사용해 훈련 횟수를 감소시켰다. 훈련된 모델을 바탕으로 블랙박스 영상을 통하여 차량과 사람을 탐지하고, OpenCV

라이브러리를 사용하여 차선을 탐지한 후 측면 충돌이 일어날 수 있는 끼어들기 상황에 대해 위험 상황이라고 보여주는 시각화 모델로 이루어져 있다.

3.1 Data Set

3.1.1 학습 Data Set

차량과 사람을 학습데이터로 정의하고, 그중 차량의 종류가 여러 가지 있기 때문에 차량 6가지와 사람 2가지로 분류하여 총 8가지의 이미지를 학습 데이터로 정의하였다. 단일 이미지는 차량 데이터 셋[6]를 사용하였다. 데이터 셋은 다양한 크기로 구성된 차량 이미지이며, 훈련데이터 8000개로 훈련을 진행하였고, 시험 데이터 4000개로 테스트를 진행하여 훈련의 성과를 평가하였다.

여러 차량과 사람이 존재하는 복합이미지는 KITTI[7] 데이터 셋을 사용하였다. 데이터 셋은 1242x375의 고정된 크기의 이미지로 훈련 데이터 5000개, 시험 데이터 2500개로 데이터 셋을 구성하였다.

3.1.2 시나리오 Data Set

끼어들기 상황에 대한 위험 시나리오에 적합한 영상을 웹상에서 총 30개를 찾아 위험 시나리오가 있는 부분만 잘라서 영상을 확보하였으며, 훈련 데이터 15개, 시험 데이터 15개로 데이터 셋을 구성하였다.

3.2 데이터 전 처리

사전학습을 위한 데이터 셋은 해당 사이트에서 각 이미지별 라벨 데이터를 제공하기 때문에 라벨 데이터를 가져와 훈련을 진행하였다.

복합이미지의 경우는 해당 사이트에서 각 이미지별 라벨을 제공은 하지만, 라벨 데이터의 내용 중 쓰지 않는 값들이 존재하여 그 부분을 제거한 새로운 라벨 데이터를 만들어 훈련을 진행하였다.

영상의 경우는 각 영상에 프레임을 추출하여 위험 상황인지 아닌지를 사람의 눈으로 직접 판단하여 위험상황을 1로, 아닌 상황을 0으로 설정하여 진행하였다.

3.3 학습모듈

3.3.1 Tensorflow Slim

Tensorflow Slim은 저수준의 Tensorflow API를 간편하게 사용할 수 있는 고수준 경량 API로써, 이미지 분류에 사용되는 딥 러닝 CNN 모델을 제공한다. 지원되는 모델은VGG[8], ResNet[4] 등이 있으며 미리 훈련된 모델을 기반으로 파인 튜닝(fine-tuning)하는 과정이 단순화되어 있다. 본 시스템에서는 Tensorflow Slim에서 지원하는 ResNet_V1_101 모델을 사용하여 사전 학습을 진행하였으며, 총 100,000번의 반복학습을 진행하였다.

3.3.2 Faster R-CNN[5]

Faster R-CNN[5]은 Tensorflow로 구현이 되었으며, 전처리된 이미지와 반복학습을 줄이기 위한 사전 훈련 데이터를 입력으로 받아 복합이미지에서 차량과 사물을 인지한다. 훈련이 시작되면 Faster R-CNN[5]에 정의된 레이어들을 사전에 ResNet[4]으로 학습시킨 데이터를 가지고 초기화를 한다. 이후 데이터 셋 중 훈련 데이터 셋 부분을 랜덤하게 입력값으로 넣으며 각각의 이미지에 대하여 훈련을 진행하여 훈련이 완료되었을 시 시험 데이터를 이용해 모델을 평가하고 정확도를 측정한다. 사전 데이터를 사용하는 것은 처음 진행할 때만 사용하며 이후 추가로 훈련하고자 할 경우 훈련된 데이터를 바탕으로 진행된다. 훈련 횟수를 3가지로 나누어 400,000번, 800,000번, 1,600,000번 훈련을 진행하였으며, Ckpt 파일 형식으로 3가지 훈련 횟수에 대해 따로 저장하였다.

3.4 차선 탐지

차선을 탐지하기 위해 OpenCV 라이브러리를 사용하여 구현하였다. OpenCV는 주로 실시간 컴퓨터 비전을 목적으로 한 프로그래밍 라이브러리이다. 차선을 탐지하기 위해 처음에 기존의 이미지를 흑백화면으로 변환하는 Gray Scale 작업을 진행한다. 이 작업을 통해 백색 부분을 제외한 나머지 부분을 모두 까맣게 변환시킨다. 그 뒤 백색 부분의 채워진 부분을 제거하여 테두리 부분만 남긴 뒤, RoI(Region of Interest) 범위에 해당하는 영역만 남긴 뒤 모두 제거하는 Canny Edge Detection[9] 작업을 진행한다. 그 뒤 남은 부분에 Hough Space[10] 작업을 통해 직선을 그어 차선을 탐지한 결과를 보여준다.

3.5 위험탐지 및 시각화

Faster R-CNN[5]을 통해 훈련된 계층을 사용하여 얻은 물체 인식 정보(물체 이름, 물체 위치, 정확도)와 OpenCV라이브러리를 통해 나온 차선을 바탕으로 끼어들기 상황에 대한 위험 시나리오를 구성하였다. 인식된 물체의 클래스에 대해 물체의 경계 박스(bounding box)와 차선의 중첩된 영역 비율에 대한 조건을 다르게 하여 위험 탐지를 정의하였다. 그림4는 위험탐지에 대한 Decision Tree로서, 경계 박스의 크기가 커질수록 위험을 탐지하는 비율도 더 크게 적용해야 한다고 판단하여 최적의 비율 조건을 찾아 적용시켰다.

시각화는 위험 탐지가 된 프레임은 영상에서 배경색이 바뀌게 하여 진행하였고, 몇 번째 프레임에서 위험 상황을 탐지하였는지 로그를 남겨 확인할 수 있도록 하였다.

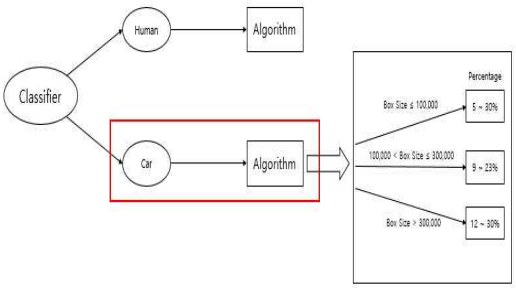

그림 4. 위험 탐지 Decision Tree

Ⅳ. 실험 결과

본 논문은 훈련 횟수에 따른 위험 상황 감지에 대한 성능을 분석할 것이다. 표1은 성능 측정을 위한 Confusion matrix로 실제 위험 상황을 위험 상황을 감지했을 경우를 TP(True Positive)한 상황으로 보고 성능 측정을 진행하였다. 본 논문에서는 정확도(Accuracy)와 정밀도(Precision), 민감도(Sensitivity) 3가지를 분석할 것이다.

		Prediction	
		위험감지	위험감지X
Actual	위험감지	TP	FN
	위험감지X	FP	TN

[표1] Confusion matrix

정확도는 (TP + TN) / (TP + TN + FP + FN)로 이루어져 있으며, 이 모델의 전체 정확도를 알려준다. 민감도는 TP / (TP + FN)로 이루어져 있으며, 실제 위험상황을 얼마나 잘 감지했는지를 알려준다. 정밀도은 TP / (TP + FP)로 이루어져 있으며, 위험상황이든 아니든 모두 위험상황으로 예측한 경우를 알려준다.

본 논문은 실제 위험상황을 얼마나 잘 감지했는지를 더 중요하게 생각하여 민감도가 얼마나 높게 나오는지를 중점적으로 보고 성능 측정을 진행하였다. 표2는 훈련 횟수에 따른 시나리오 데이터의 분석 결과이다.

훈련 횟수	Accuracy	Precision	Sensitivity
40(만)	0.80903388	0.56566343	0.86836839
80(만)	0.80295776	0.56953252	0.84002562
160(만)	0.82026148	0.55057281	0.83694295

[표1] 성능 분석 결과표

위험 상황 판단의 경우 차량을 인식하는 경계 박스와 차선에 중첩된 비율을 계산하여 비율 조건에 만족하면 위험 상황을 감지하도록 설정했다. 대부분의 탐지 모델 활용이 실시간으로 이루어지므로 프레임 단위로 기록들을 저장하였다. 정확도는 약 80%의 성능을 보였고, 민감도는 약 84%, 정밀도는 약 55%의 성능을 보여 모델의 성능이 나쁘지 않음을 증명하였다.

Ⅴ. 결 론

본 논문에서는 Faster R-CNN[5] 신경망과 Resnet[4] 모델을 적용하여 물체의 위치를 탐지하였다. Faster-RCNN[5] 모델에 차량과 사람의 이미지를 학습시킨 Resnet[4] 모델을 적용한 후 차량과 사람이 라벨링 된 복합 이미지를 학습시켰다. 학습 후 다른 이미지 셋을 넣었을 때 빠르게 결괏값을 도출할 뿐만 아니라 정확하게 차량과 사람의 위치를 파악하였다. 테스트 과정에서 위 모델이 복합 이미지를 인식한 결과와 사람이 물체를 확인한 결과는 비슷한 수준으로 나타나는 것을 확인할 수 있었다. 그 후에 블랙박스 영상을 프레임 단위로 쪼개어 적용하였고, 프레임에서 인식한 차량 혹은 사람에 박스를 추가하여 물체를 인식했다는 결과를 표시해주었다. 마찬가지로, OpenCV 라이브러리를 이용하여 프레임에 찍힌 차량의 차선을 탐지하여 표시해주었다. 이렇게 표시된 데이터를 바탕으로 측면 충돌이 발생할 수 있는 끼어들기 상황에 대해 위험하다고 인지를 하게 되면 시각화를 통해 위험 상황임을 알려주게 하였다.

향후에는 차량이 끼어드는 상황이 아닌 사람이나

동물이 끼어드는 상황에 대해 높은 정확성을 가지는 비율 조건을 찾을 수 있도록 실험할 예정이다. 또한, 차선이 없는 상황에 대해 위험을 어떻게 감지를 할지 논의를 할 예정이다. 마지막으로 테스트에서 추출한 시각화한 이미지를 바탕으로 측면 충돌 상황의 위험성을 단계별로 나누어 알려 줄 수 있도록 하는 등, 여러 기술을 개발할 예정이다.

Acknowledgement

본 연구는 과학기술정보통신부 및 정보통신기술진흥센터의 SW 중심대학지원사업의 연구결과로 수행되었음(2015-0-00930)

References

[1] Chenyi Chen, Ari Seff, Alain Kornhauser, Jianxiong Xiao, “DeepDriving: Learning Affordance for Direct Perception in Autonomous Driving,” The IEEE International Conference on Computer Vision (ICCV), Santiago , pp. 2722-2730 , 2015.

[2] Jagannath Aghav, Poorwa Hirwe, and Mayura Nene, “Deep Learning for Real Time Collision Detection and Avoidance,” in the Proceedings of International Conference on Communication, Computing and Networking (ICCCN 2017) , Vancouver , pp. 483-488 , 2017.

[3] Fu-Hsiang Chan, Yu-Ting Chen, Yu Xiang, Min Sun, “Anticipating Accidents in Dashcam Videos” , Asian Conference on Computer Vision (ACCV) , Taipei , pp. 136-153 , 2016.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, “Deep Residual Learning for Image Recognition,” The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas , pp. 770-778 , 2016.

[5] S. Ren, K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in IEEE Transactions on Pattern Analysis & Machine Intelligence, vol. 39, no. 6, pp. 1137-1149, 2017.

[6] Jonathan Krause, Michael Stark, Jia Deng, Li Fei-Fei, “3D Object Representations for Fine-Grained Categorization,” The IEEE International Conference on Computer Vision (ICCV) Workshops , Sydney , pp. 554-561, 2013.

[7] Andreas Geiger and Philip Lenz and Raquel Urtasun, “Are we ready for autonomous driving? the kitti vision benchmark suite”, IEEE Conference on Computer Vision and Pattern Recognition, Providence, pp.3354-3361, June 2012.

[8] K. Simonyan, A. Zisserman., “Very Deep Convolutional Networks for Large-Scale Image Recognition”, Proceedings of International Conference on Learning Representations (ICLR), San Diego, p.1-14 , May 2015.

[9] J. Canny, “A computational approach to edge detection”, IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol.8, No.8, pp. 679-698, Nov.1986.

[10] T. Risse, “Hough transform for line recognition: Complexity of evidence accumulation and cluster detection,” Computer Vision Graphics Image Processing, vol. 46, Issue 3, pp. 327–345, June 1989.

[11] Alex Krizhevsky and Ilya Sutskever, and Geoffrey E Hinton, “ImageNet Classification with Deep Convolutional Neural Networks”, Neural Information Processing Systems(NIPS), Nevada, pp.1097 1105 , Dec. 2012.