



### 저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

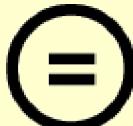
다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원 저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리와 책임은 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)



이 학 석 사 학 위 논 문

딥러닝을 이용한 영상 기반 자율 주행 자동차 시스템  
An Image Based Autonomous Driving System using Deep  
Learning

지도교수 인 치 호

이 경 민

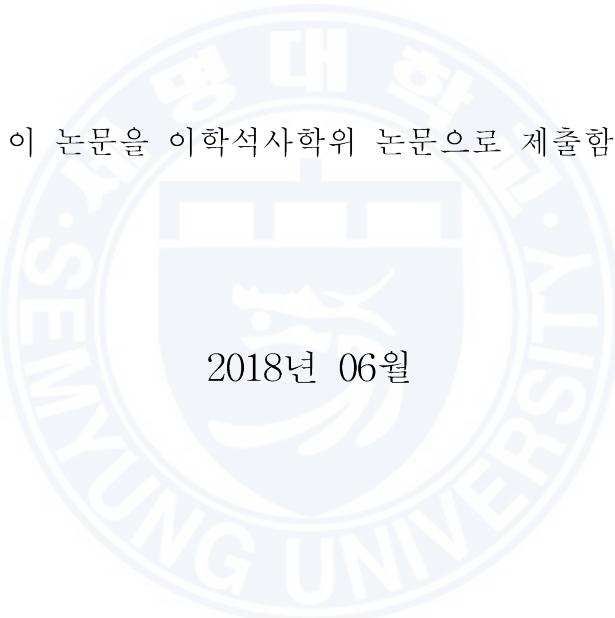
세명대학교 대학원

컴 퓨 터 학 과

2018년 08월

딥러닝을 이용한 영상 기반 자율 주행 자동차 시스템  
An Image Based Autonomous Driving System using Deep  
Learning

지도교수      인 치 호



2018년 06월

세명대학교 대학원  
컴퓨터학과

이      경      민

이 경 민의 이학석사학위 논문을 인준함

위원장

이학석



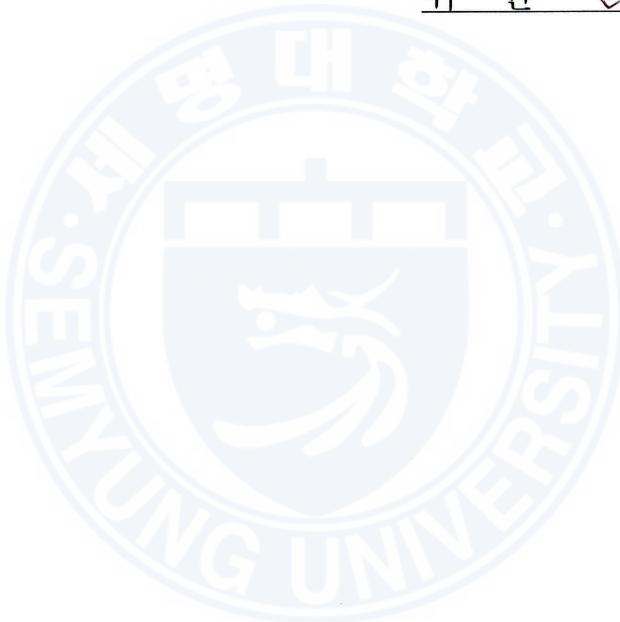
위원류

승필



위원인

이치호



세명대학교 대학원

2018년 06월

# 목 차

목 차 .....	ii
표 목 차 .....	iii
그 립 목 차 .....	iv
국 문 요 약 .....	vi
제 1 장 서 론 .....	1
제 2 장 관련 연구 .....	4
2.1 자율주행 자동차 .....	4
2.2 딥러닝(Deep Learning) .....	5
2.3 퍼셉트론(Perceptron) .....	6
2.4 오류역전파(Error Back Propagation) .....	10
2.5 합성곱 신경망(Convolutional Neural Network) .....	12
2.5.1 합성곱 계층(Convolution Layer) .....	14
2.5.2 폴링 계층(Pooling Layer) .....	16
2.6 합성곱 신경망의 한계 .....	17
2.7 검출(Detection) .....	17
2.7.1 검출 주요 모델 .....	18
2.8 SSD(Single Shot multibox Detector) .....	19
제 3 장 딥러닝을 이용한 영상 기반 자율주행 자동차 시스템 .....	22

3.1 개요 .....	22
3.2 딥러닝 기반 자율주행을 위한 CNN 구조 제안 .....	23
3.2.1 사진 데이터 학습 .....	24
3.2.2 Classification Decoder .....	24
3.2.3 Detection Decoder .....	25
3.2.4 Semantic Decoder .....	26
3.3 자율주행 차량 제어 .....	28
제 4 장 실 험 .....	30
4.1. dataset .....	30
4.2 Performance Evaluation .....	31
제 5 장 결 론 .....	34
참고 문헌 .....	36
영문 요약 .....	40
연구 실적 목록 .....	42

## 표      목      차

[표 2-1] 자율주행 기술수준 정의 .....	5
[표 2-2] XOR 진리표 .....	9
[표 2-3] 검출 모델 비교 .....	20

[표 4-1] 데이터셋 구성 ..... 30

[표 4-2] 실험 환경 ..... 31

## 그 림 목 차

[그림 2-1] 인공신경마의 선형맞춤과 비선형 변형 ..... 6

[그림 2-2] Rosenblatt의 단층 퍼셉트론 도식 ..... 7

[그림 2-3] 단층 퍼셉트론 ..... 7

[그림 2-4] 단층 퍼셉트론의 선형분류 ..... 8

[그림 2-5] XOR 에서의 선형분류 문제 ..... 9

[그림 2-6] 신경망의 순전파와 역전파 ..... 11

[그림 2-7] 역전파의 예 ..... 11

[그림 2-8] 딥러닝을 이용한 이미지 특징 추출 과정 ..... 13

[그림 2-9] 합성곱 ..... 14

[그림 2-10] 합성곱 연산의 예 ..... 15

[그림 2-11] 합성곱의 zero padding과 strides ..... 16

[그림 2-12] Max pooling ..... 14

[그림 2-13] 이미지속 물체 검출과 AP 지표 그래프 ..... 18

[그림 2-14] Selective Search를 통한 객체 검출 ..... 18

[그림 2-15] RPN 과정이 없는 검출 과정 ..... 19

[그림 2-16] SSD(Single Shot multibox Detector) 구조 ..... 20

[그림 2-17] Multi-scale feature maps .....	21
[그림 3-1] 제안하는 자율주행 시스템 구성도 .....	22
[그림 3-2] 제안한 신경망 구조 .....	23
[그림 3-3] Classification Decoder .....	25
[그림 3-4] Detection Decoder .....	25
[그림 3-5] Semantic Decoder .....	26
[그림 3-6] 차선 후본점 .....	26
[그림 3-7] BirdEye View 이미지 변환 .....	27
[그림 3-8] 차선 검출 .....	27
[그림 3-9] Semantic Decoder 결과 .....	28
[그림 3-10] 자율주행 차량 제어 구성 .....	28
[그림 3-11] 자율주행 차량의 조향 제어 .....	29
[그림 4-1] Decoder 결과 정확도 그래프 .....	32
[그림 4-2] 차량 검출 이미지 .....	33
[그림 4-3] 차선 및 주행 차로 세그먼트 .....	33

# 딥러닝을 이용한 영상 기반 자율주행 자동차 시스템

이 경 민

세명대학교 대학원 컴퓨터학과 컴퓨터학 전공

## 요약

본 논문에서는 딥러닝을 이용한 영상 기반 자율주행 자동차 시스템을 제안한다. 자동차가 자율성을 가지고 최소한의 운전자 조작만으로 안전하게 도로를 주행할 수 있는 자동차를 구현하기 위해서는 다양한 센서와 센서들이 인식한 정보를 한곳에 모아 하나의 물체로 확인하기 위해 복잡한 분석 알고리즘과 정교한 소프트웨어가 필요하며 이를 위해 많은 시간과 비용을 들여 자율주행 자동차가 구현된다.

최근 하드웨어의 발전으로 빅데이터 기반으로 스스로 학습하는 딥러닝이 많은 주목을 받고 있다. 제안한 시스템은 딥러닝을 이용한 자율주행 자동차 기술을 통해 높은 기술 진입 장벽을 허물어 고가의 센서가 아닌 카메라와 같은 범용 센서들을 활용하여 자율주행 자동차를 설계하고자 한다.

본 논문에서는 CNN 네트워크 구조 중 VGGNet-16을 참고한 SSD 네트워크 구조를 활용하여 영상 분류, 객체 검출, 시멘틱 세그멘테이션과 관련효율적으로 처리와 WIDER FACE, KITTI 밴치마크와 Cityscapes 밴치마크에서 제공하는 실제 도로 환경 영상을 이용하였으며, 제공 받은 데이터 중 일부를 테스트 데이터로 구성하였다. 본 논문에서 제안하는 신경망을 활용하여, 테스트 데이터를 이용한 실험 결과를 통해 제안하는 통합 네트워크의 성능과 실시간성 확보를 검증하여 실제 주행 환경에서의 적용 가능성을 확인하였다.

# I. 서 론

자동차 산업의 연구개발(R&D)분야는 소비자의 편의 및 안전, 환경규제 등의 법규에 기반을 두어 고기능 및 고성능 요구와 IT기술의 발전에 따른 기술 융합이 급격하게 증가하여 자동차가 단순히 이동수단으로의 역할을 넘어 지능화 및 스마트화가 급속하게 진행되어 지금은 또 다른 생활공간으로 발전하고 있으며 특히, 기술의 변화와 더불어 자율주행 자동차라는 새로운 패러다임의 등장으로 많은 완성차 업체와 IT기술업체 그리고 전기자동차 업체까지 가세하여 기술개발에 많은 투자가 이루어지고 있다[1-2].

자율주행 자동차는 자율성을 가지고 최소한의 운전자 조작만으로 안전하게 도로를 주행할 수 있는 차량이며, 자율주행 자동차의 기본적인 작동은 사람이 운전하는 원리와 같이 ‘인지-판단-제어’의 3단계를 거친다. 인지는 센서나 카메라를 통해 주변 환경을 인식하는 것으로 사람의 눈과 같은 역할을 하며, 판단은 인지한 상황에 대한 정보를 바탕으로 주행전략을 판단하는 것으로, 사람의 두뇌와 같은 역할을 한다. 마지막으로 제어는 실질적인 속도조절, 조향제어 등 차량을 제어하는 것으로 사람의 혈관이나 근육에 해당한다. 이러한 단계로 진행되는 자율주행 자동차는 각종 센서와 카메라를 통해 주변 환경을 인식하고, 실시간으로 수집한 교통 정보를 처리하고 분석해서 정확한 판단을 내려야하고, 하드웨어를 소프트웨어로 제어 할 수 있는 기술이 필요하며, 현재 완성차 업체에서는 스마트 크루즈 컨트롤(SCC, Smart Cruise Control), 자동 긴급 제동 시스템(AEB, Autonomous Emergency Braking), 주차 조향 보조 시스템(SPAS,

Smart Parking Assistance System), 차선 이탈 경보 시스템(LDWS, Lane Departure Warning System), 차선 유지 지원 시스템(LKAS, Lane Keeping Assist System) 등 이 모든 장치를 첨단 운전자 보조 시스템(ADAS, Advanced Driver Assistance System)라고 묶어서 불리는 기술을 제공한다[2-3].

하지만 다양한 센서를 활용한 자율주행 자동차는 안전을 위해 사용된 고성능 특화센서의 비용은 만만치 않으며, 이 센서들이 인식한 정보를 한 곳에 모아 하나의 물체로 확인하기 위해서는 복잡한 분석 알고리즘과 정교한 소프트웨어가 필요하며, 이를 위해 많은 시간과 비용을 들여 전문가들에 의한 규칙 기반 방식(rule-based approach)으로 자율 주행 자동차가 구현되고 있다[4].

최근 자율주행 자동차는 연산을 위한 하드웨어의 급격한 발전으로 빅데이터 기반으로 컴퓨터가 스스로 학습하여 사물을 구별할 수 있는 딥러닝 기술이 자율주행 자동차 연구에 많은 주목을 받고 있다. 특히 딥러닝 기술 중 시각 이미지를 이해하고 추상적인 정보인 고수준 특징 정보를 추출할 수 있는 합성곱 신경망(CNN, Convolutional Neural Network)이 많이 사용된다[5-7].

합성곱 신경망을 이용한 이미지 데이터 처리 및 분석은 자율주행 자동차에서 사용되는 다양한 센서로부터의 데이터 처리 및 분석보다 수집된 데이터의 종류나 크기가 간소화되어 분석 알고리즘의 복잡성이 낮아지며, 이를 통해 처리 속도가 빨라지는 장점이 있다. 또한 빨라진 처리 속도로, 자율주행 자동차에서 실시간 판단 및 제어가 가능하게 된다.

본 논문에서는 딥러닝을 이용한 영상 기반 자율주행 자동차 시스템을 제안하였다. 제안된 시스템은 기존 다중 센서로부터의 데이터

보다 카메라 단일 센서로부터의 데이터 간소화를 통해 처리 및 분석 속도 향상시킬 수 있었으며, 딥러닝을 이용한 고수준의 특징 추출을 통한 객체의 분할 및 분류 그리고 검출의 인지 단계와 인진된 정보를 기반으로 분석하여 주행 전략을 세우는 판단 단계, 마지막으로 실시간으로 영상 디바이스를 통해 운전자에게 경고 제공과 제어 할 수 있게 시스템을 설계하고자 한다.



## II. 관련 연구

### 2.1 자율주행 자동차

자율주행 자동차 기술은 운전자가 설정하는 기본 경로 정보를 바탕으로 목적지까지 도달하는 구현 수준에서부터 시작되어 현재는 보다 진보한 기술과 시스템을 통해 주행 중 일어날 수 있는 위험상황을 미리 감지해 사전 경고하는 수준까지 발전되었다.

자율주행 자동차는 스스로 주변 환경을 인지, 위험을 판단, 차량을 제어하여 운전자 주행 조작을 최소화하며 스스로 안전주행 및 커넥티드 서비스 제공이 가능한 인간친화적 자동차이다. 미국도로교통안전국(NHTAS, National Highway Traffic Safety Administration)과 미국자동차기술협회(SAE, Society of Automotive Engineers)는 자율주행 자동차의 체계적인 관리를 위해 표 2-1과 같이 자율주행 기술수준 정의하였다. 현재는 2-3 단계로 발전 중이며, 최근 4-5단계의 완전자율주행 자동차 기술 개발을 위해 인공지능을 접목한 연구가 진행되고 있다[8].

표 2-1 자율주행 기술수준 정의  
Table 2-1 Autonomous technology level definition

NHTSA 레벨	SA E 레벨	자율수준정의		특징
0	0	수동주행(Non-Automation)		
1	1	보조(Assisted)		주행정보 제공 및 경고 생성, 제어 기능 일부 지원
2	자동화 (Automated)	반자동 (Semi-automated)		특수한 상황에서 운전자 선택에 따라 차량 제어 기능 일부를 자동화
3		고도화된 자동 (Highly Automated)		차량 모든 제어기능 자동화(주로 고속도로 내), 운전자가 수동/자동 선택
4	4	자율 (Autonomous)	완전자동 (Fully Automated)	모든 교통상황에 차량 스스로 주행 가능
	5	Co-operative		V2V/V2I 기반의 자율주행

## 2.2 딥러닝(Deep Learning)

딥러닝의 개념을 새로운 것이 아닌 인공신경망(ANN, Artificial Neural Network)에서 시작되었다. ANN의 개념은 1943년 McCulloch와 Pitts가 발표한 논문에서 최초로 등장했다. 이들이 제안한 모델은 인간 두뇌에 관한 최초의 논리적 모델링이었으며, 인간의 신경 구조를 복잡한 스위치들이 연결된 네트워크로 표현할 수 있다고 제안하였다. ANN의 개념은 선형 맞춤(Linear fitting)과 비선형 변환(nonlinear transformation)을 반복해 쌓아올린 구조이다. 즉, ANN은 데이터를 잘 구분할 수 있는 선들을 긋고, 이 공간들을 잘 웨곡해 합하는 것을 반복하는 구조라 말할 수 있다. 그림 2-1은 ANN의 개념에서의 선형 맞춤과 비선형 변환을 나타낸다.

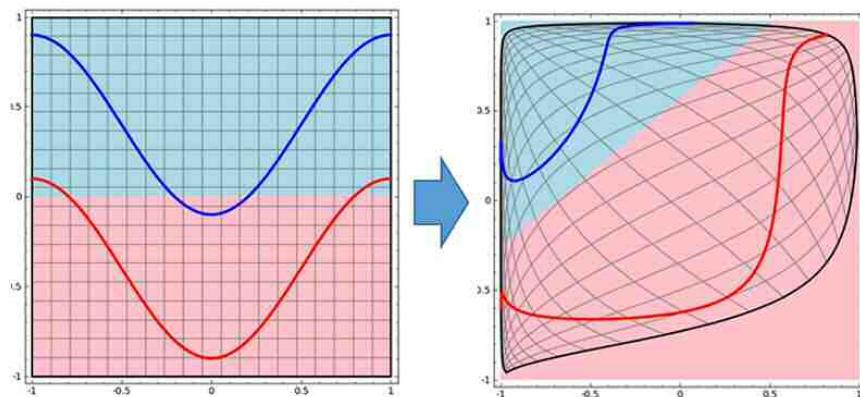


그림 2-1. 인공신경망의 선형맞춤과 비선형 변형

Fig. 2-1 Linear alignment and non-linear transformation of the neural network

그림 2-1 에서는 평탄 선과 뾰족한 선의 영역을 구분하는 것으로 그냥 구분선을 긋는다면 왼쪽처럼 불완전하게 그을 수 있다. 하지만 공간을 왜곡하면 오른쪽과 같이 잘 구분되게 구분선을 그릴 수 있다. 딥러닝은 이 과제를 선 긋고 왜곡하고 합하고를 반복하며 복잡한 공간 속에서의 초적의 구분선을 만들어 내는 것이 목적이다 [9].

### 2.3 퍼셉트론(Perceptron)

퍼셉트론은 1957년 코넬 항공 연구소(Cornell Aeronautical Lab)의 프라크 로젠클라트(Frank Rosenblatt)에 의해 고안된 인공신경망이다. 로젠클라트의 의해 제안된 것은 가장 간단한 형태의 단층 퍼셉트론(Single Perceptron)으로 입력 벡터를 두 부류로 구분하는 선형 분류기이다. 그 구조는 그림 2-2에 나타난 바와 같다[10].

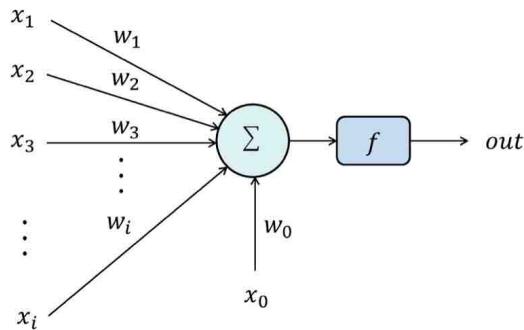


그림 2-2 Rosenblatt의 단층 퍼셉트론 도식

Fig. 2-2 Single layer perceptron scheme of Rosenblatt's

단층 퍼셉트론은 입력층(Input Layer)과 출력층(Output Layer)으로 구성된다. 입력층은 학습 벡터 또는 입력 벡터가 입력되는 계층으로써, 입력된 데이터는 출력층 뉴런으로 전달되어 활성함수에 따라 값이 출력된다. 출력층은 퍼셉트론 설계 시 임의  $n$ 개의 뉴런으로 구성할 수 있으며, 그림 2-3은 1개의 뉴런으로 구성된 출력층을 나타낸다.

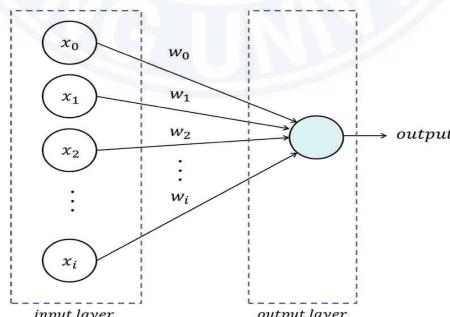


그림 2-3 단층 퍼셉트론

Fig. 2-3 Single layer perceptron

단층 퍼셉트론의 학습 알고리즘의 예는 아래와 같다.

- (1) 가중치(Weight)와 바이어스(Bias) 가중치를 -0.5와 0.5 사이의 임의 값으로 초기화.
- (2) 하나의 학습 벡터에 대한 출력총 뉴런의 *net* 값을 계산.
- (3) 활성함수를 통해 계산된 *net* 값으로부터 뉴런이 실제 출력값을 계산.
- (4-1) 뉴런의 출력값과 목표값의 차이가 허용 오차보다 작으면 (5)로 이동.
- (4-2) 뉴런의 출력값과 목표값의 차이가 허용 오차보다 크면 학습 진행.
- (5-1) 현재 학습 벡터가 마지막 학습 벡터가 아니면, 현재 벡터를 다음 학습 벡터로 설정하고 (2)로 이동하여 반복.
- (5-2-1) 현재 학습 벡터가 마지막 학습 벡터이고, 모든 학습 벡터에 대해 출력값과 목표값이 허용 오차보다 작으면 알고리즘 종료.
- (5-2-2) 현재 학습 벡터가 마지막 학습 벡터이지만, 출력값과 목표값이 허용 오차보다 큰 학습 벡터가 존재하면, 현재 학습 벡터를 처음 학습 벡터로 설정하고 (2)로 이동하여 반복.

퍼셉트론은 모든 학습 데이터를 정확히 분류시킬 때까지 학습이 진행되기 때문에 학습 데이터가 선형적으로 분리될 수 있을 때 적합한 알고리즘이다[10].



그림 2-4 단층 퍼셉트론의 선형분류

Fig. 2-4 Linear classification of the single-layer perceptron

선형 분류는 그림 2-4 과 같이 선으로 분류하는 것을 의미하며, 학습이 반복될수록 기울기가 달라지는 것을 볼 수 있다. 하지만 퍼

셉트론이 인공지능 분야에서 센세이션을 불러일으켰으나, 한계점이 존재하는 이론이다. 퍼셉트론의 한계점은 XOR 논리와 같이 비선형 분류는 불가능하다는 점이다. XOR 논리는 배타적(Exclusive) 논리 연산이다. 아래의 표 2-2 와 같은 진리표를 보면,  $x_1$ 과  $x_2$  중 어느 한쪽이 1일 때만 1을 출력한다.

표 2-2 XOR 진리표  
Table 2-2 XOR truth table

<b>x1</b>	<b>x2</b>	<b>y</b>
0	0	0
1	0	1
0	1	1
1	1	0

그림 2-5와 같이 XOR 논리에서는 하나의 선형으로 분류가 불가능함을 알 수 있다.

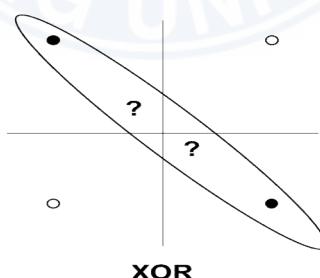


그림 2-5 XOR 에서의 선형분류 문제  
Fig. 2-5 Linear classification problem in XOR

이처럼 단층 퍼셉트론만으로는 비선형적으로 분리되는 데이터에 대해서는 제대로 된 학습이 불가능하다는 한계점을 1969년 Minsky 와 Papert는 “Perceptrons”라는 책을 통해 수학적으로 증명하였고 [11] 다층 퍼셉트론(MLP, Multi-Layer Perceptron)으로 신경망을 구성하면 XOR 문제를 풀 수 있으나, MLP를 학습시키는 방법은 존재하지 않는다고 단정해버렸다. 이후 1974년 Paul Werbos가 MLP를 학습시키는 방법을 찾게 되는데, 이를 오류역전파(Back Propagation) 개념이라 한다.

## 2.4 오류역전파(Error Back Propagation)

입력값에서 출력값으로 가중치를 계층 간의 가중치를 업데이트(학습)하면서 최종 결과값을 가져오는 것은 순전파(Foward)라고 하며 반대 과정을 역전파(Backward)라고 한다. 역전파는 신경망에서의 최종 출력값과 실제값의 오차가 최소가 되도록 역전파하여 뉴런간의 가중치를 재업데이트하는 방법으로 학습을 시킨다. 그림 2-6와 같이 입력값이 들어오는 방향(순전파)으로 출력 계층에서 결과 값이 나오며, 결과값은 오차(Error)를 다시 역방향으로 히든 계층과 입력 계층으로 오차를 다시 보내면서 가중치를 계산하면서 최종 결과값에서 발생했던 오차를 적용시킨다[12]. 그림 2-6 은 신경망의 순전파와 역전파를 나타낸다.

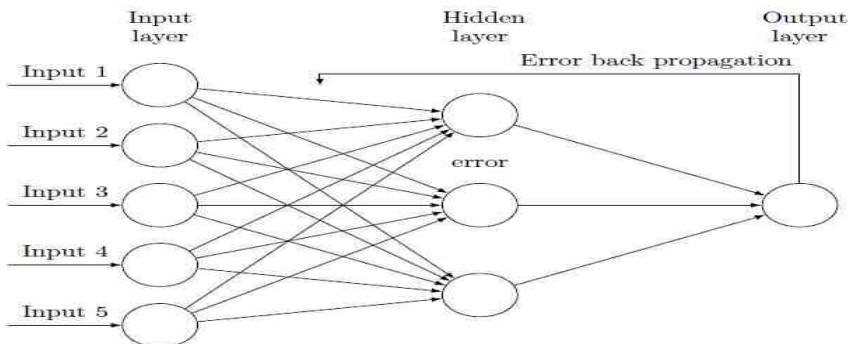


그림 2-6 신경망의 순전파와 역전파

Fig. 2-6 Forward Propagation and Backward Propagation of Neural Networks

이 과정을 한 번 수행하는 것을 1 주기(Epoch) 라고 하며 주기를 늘릴수록 가중치가 계속 업데이트되면서 점점 오차가 줄어나간다.

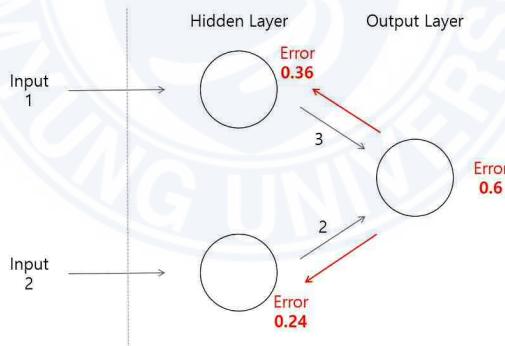


그림 2-7 역전파의 예

Fig. 2-7 Example of back propagation

그림 2-7 과 같이 예를 들어 최종 출력 계층에서 나온 결과 값이 나온 오차가 0.6이라고 되어 있다. 이전 뉴런 출력 계층에 각각, 3,

2라는 값을 전달하였기 때문에 출력의 오차에 위 뉴런은 60%, 아래 뉴런은 40% 영향을 주었다고 볼 수 있다. 오차 0.6을 0.6, 0.4씩 곱하니 위 뉴런의 오차가 0.36, 아래 뉴런은 0.24가 전달된다. 오류 역전파는 말 그대로 이렇게 오차를 점점 거슬러 올라가면서 다시 전파하는 것을 의미한다.

## 2.5 합성곱 신경망

전연결(Fully Connected Layerd)으로 구성된 신경망의 입력 데이터는 1차원 데이터로 입력된다. 그러나 이미지 데이터는 3차원 데이터이며, 배치 형태로 사용되는 여러 장의 사진은 4차원 데이터이다. 이런 이미지 데이터를 전연결 신경망으로 학습시킬 경우 3차원 데이터를 1차원 데이터로 평면화시켜야 한다. 이미지 데이터를 평면화시키는 과정에서 공간 정보가 손실될 수밖에 없다. 결과적으로 이미지 공간 정보 유실로 인공 신경망의 특징 추출 및 학습이 비효율적이고 정확도를 높이는데 한계가 있다. 이미지의 공간 정보를 유지할 수 있으며, 학습이 가능한 모델이 합성곱 신경망이다. 합성곱 신경망은 기존 전연결과 다르게 각 계층의 입출력 데이터의 형상 유지가 가능하며, 이미지 공간의 정보를 유지하면서 인접 이미지와의 특징을 효과적으로 인식이 가능하다. 그리고 복수의 필터와 특징을 모으고 강화하는 풀링(Pooling) 계층을 통해 추출 및 학습을 효과적으로 할 수 있다. 또한 필터를 공유 파라미터로 사용하기 때문에 일반 인공 신경망과 비교하여 학습 파라미터가 매우 적은 특징을 가지고 있다. 이런 합성곱 신경망은 딥러닝의 대표적인 방법이며, 주로 이미지 인식에

많이 사용되며 기본적인 개념은 이미지의 작은 특징에서 복잡한 특징으로 추상화하는 것을 말한다. 예를 들어 사람의 얼굴을 인식하는 합성곱 신경망을 만들 경우 먼저 필터를 사용하여 간단한 특징을 뽑아내어 하나의 합성곱 계층을 만든다. 그 다음에 추출된 특징들에서 좀 더 복잡한 특징을 추출하는 새로운 계층을 추가한다. 그럼 2-8 은 간단한 특징 추출부터 복잡한 특징을 추출하는 과정을 나타낸다.

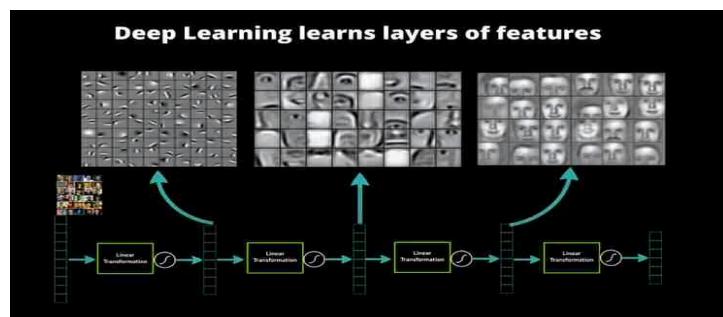


그림 2-8 딥러닝을 이용한 이미지 특징 추출 과정

Fig. 2-8 Image Feature Extraction Process Using Deep Learning

합성곱 신경망의 구조는 그림 2-9 과 같은 형태를 보이며 일반적으로 2단계 과정으로 이루어진다. 특징 추출 단계는 분류를 위한 영상의 특징 추출과 위상 불변성(topology invariance)을 얻기 위해 합성곱 연산과 추출된 특징을 모으고 강화하기 위한 sub-sampling 과정을 수행하며, 이러한 과정을 반복적으로 수행하여 지역적 특징으로부터 전역적 특징을 얻어낼 수 있다. 분류기 단계는 학습을 통해 다양한 경우에 대응할 수 있도록 해주는 것이 목표이며, 그 구조는 기존의 인공신경망과 동일한 구조를 갖는다[13-15].

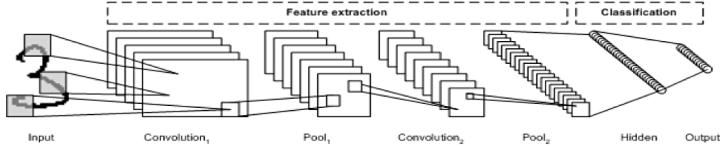


그림 2-9 합성곱 신경망

Fig. 2-9 CNN

### 2.5.1 합성곱 계층(Convolution Layer)

영상 처리 분야에서 합성곱 계층은 필터를 이용한 이미지의 특징을 찾아내는 계층이다. 필터는 이미지의 특징을 찾아내기 위한 공용 파라미터로서 필터 혹은 커널이라고 한다. 그림 2-10과 같이 입력 데이터를 지정된 간격으로 이동하면서 전체 입력 데이터와 합성곱 연산을 하여 데이터를 출력하는데 이를 특징맵이라 한다[13].

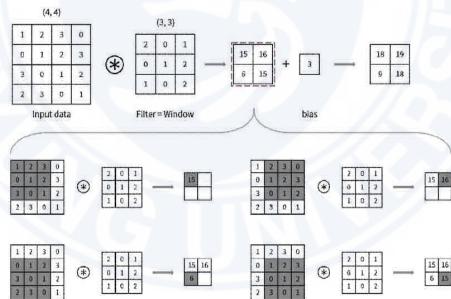


그림 2-10 합성곱 연산의 예

Fig. 2-10 Example Convolution Operation

특징맵은 그림 2-10과 같이 합성곱 연산을 할 경우 특징맵의 크기가 줄어든다. 특징맵 크기가 줄어든 만큼 이미지 데이터의 정보가 손실된다. 따라서 합성곱 연산을 수행하기 전, 가장자리의 정보 손

실을 방지하기 위해 입력 데이터의 가장자리 외부에 공간을 생성하여 특정 값으로 채워 넣는 방법을 사용하며, 이를 패딩(padding)이라 한다. 패딩은 주로 출력 데이터의 공간적(spatial) 크기를 조절하기 위해 사용되며, 특정 값인 하이퍼 매개변수(Hyper parameter)로 어떠한 값으로 채워 넣을지 결정할 수 있다. 합성곱 연산에서 필터는 입력 데이터를 지정된 간격으로 이동되는데 그 간격의 크기를 스트라이드(stride)라고 하며 패딩 기법과 마찬가지로 하이퍼 매개변수에 해당한다. 스트라이드 또한 출력 데이터의 크기를 조절하기 위해 사용한다 [14].

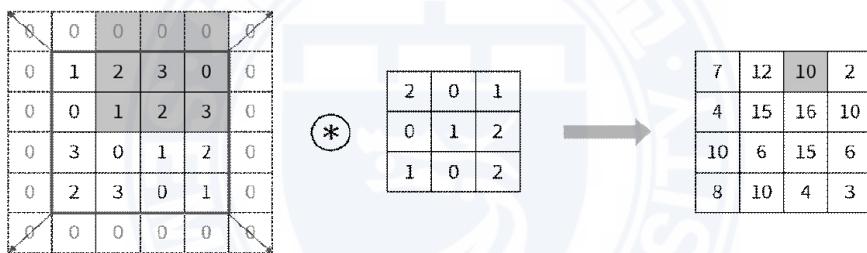


그림 2-11 합성곱의 zero padding과 strides

Fig. 2-11 Convolution of the zero padding strides

그림 2-11 은 제로 패딩과 스트라이드를 1로 적용한 합성곱을 수행하는 예를 보인다. 합성곱 연산으로 줄어든 출력 데이터 크기는 아래의 수식 (1) 과 같다. 수식 (1)은 패딩과 스트라이드 값과, 입력 데이터와 필터의 크기가 주어졌을 때 출력 데이터의 크기는 얻을 수 있다.

$$(OH, OW) = \left( \frac{H+2P-FH}{S} + 1, \frac{W+2P-FW}{S} + 1 \right) \quad (1)$$

(H, W)는 입력데이터의 크기, (FH, FW)는 필터의 크기, (OH, OW)는 출력데이터의 크기, P는 패딩, S는 스트라이드를 의미한다 [14].

### 2.5.2 풀링 계층(Pooling Layer)

풀링 계층은 합성곱 계층에서 추출된 특징을 모으고 강화하는 계층이다. 또한 패딩과 스트라이드처럼 데이터의 출력 데이터 크기를 축소하는 데 사용한다. 일반적으로 합성곱 계층에서는 출력데이터의 크기를 입력데이터의 크기로 유지하고, 풀링 계층에서만 크기를 조절한다. 풀링의 종류로는 최대 풀링(Max-pooling)과 평균 풀링(Average-pooling)이 있으며, 영상 인식 분야에서는 주로 최대 풀링 기법을 사용하여 각각의 윈도우에서 가장 큰 자극만을 선택한다[14] 그림 2-12 는 최대풀링과 평균 풀링을 나타낸다.

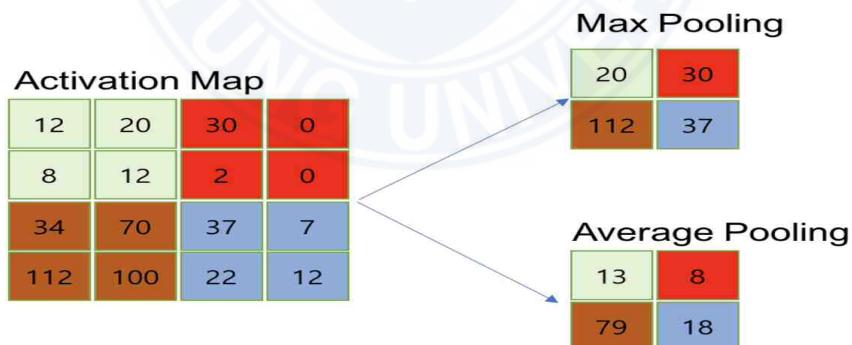


그림 2-12 최대 풀링과 평균 풀링

Fig. 2-12 Max pooling and Average pooling

## 2.6 합성곱 신경망의 한계

합성곱 신경망은 딥러닝을 대중적으로 확산시킨 기술이다. 그러나 합성곱 신경망은 구조적으로 한계가 있다. 합성곱 신경망의 한계는 사람의 얼굴을 학습 및 추론하는 과정을 통해 설명할 수 있다. 사람의 얼굴은 타원형 얼굴과 하나의 입과 코, 두 개의 눈으로 분류할 수 있으며, 합성곱 신경망 통한 분류도 같은 방법으로 분류된다. 이 요소들을 가진 물체를 사람의 얼굴이라 인식하는데 각 요소들의 어떤 방향과 구조로 배치되어 있는가는 인식에 큰 영향을 주지 않는다.

## 2.7 검출(Detection)

앞서 2.6절에서 설명한 합성곱 신경망의 한계는 자율주행 자동차 구현에 문제가 된다. 자율주행 구현에서의 인식 과정은 단순히 차량, 보행자 등을 인식하는 것이 아니라 각 객체의 방향과 상호간의 관계를 고려하여 인식되어야 하는데 우선적으로 차량, 보행자 등의 관심 있는 객체와 관심 없는 객체를 구별하는 검출(Detection) 과정이 필요하다. 검출은 이미지 속의 여러 물체를 찾고, 찾은 물체를 분류하는 것을 말한다. 또한 물체의 검출정확도(Precision)와 빠짐없이 검출 할 수 있는지의 지표인 적합율(Recall)의 관계(precision-recall curve)에서 산출한 평균 검출 정확도(AP, Average Precision)가 주요 지표로 실제 문제에 응용이 기대되고 AP 외에 예측시의 계산 시간도 중요하여, 실시간성이 요구되는 기술이다. 그림 2-13 은 이미지속의 물체 검출 모습과 AP 지표 그래프를 나타낸다.

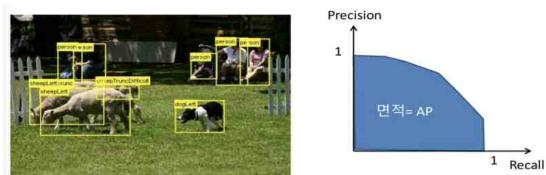


그림 2-13 이미지 속 물체 검출과 AP 지표 그래프

Fig. 2-13 Image in object detection and AP index graph

### 2.7.1 검출 주요 모델

검출 주요 모델은 크게 두 가지로 나뉜다. 첫 번째 방법은 그림 2-14 와 같이 RPN(Region Proposal Net)을 이용하여 이미지 속에서 질감이나 색, 강도 등이 비슷한 픽셀끼리 연결된 윈도우를 무수히 많이 만들어 그 중에 어떤 것이든 실제 객체에 해당하는 것을 찾는 방식으로 이때 윈도우를 생성하는 알고리즘을 Selective Search[16] 라고 하며, 이 알고리즘을 통해 생성된 박스 또는 Region Proposal라고 불리는 영역을 찾는 검출 모델로는 R-CNN[17], Fast R-CNN[18], Faster R-CNN[19] 등이 있다.



그림 2-14 Selective Search를 통한 객체 검출

Fig. 2-14 Object detection using Selective Search

기존 RPN 활용한 검출 모델 중 Faster R-CNN이 가장 빠르나 실

시간 영상 분석에는 사용할 수 없었다. 이후에 실시간 영상 분석이 가능하도록 개선된 검출 모델이 두 번째 방법이다. 두 번째 방법은 기존에 사용된 RPN 통한 영역 후보 생성과 각 영역 특징 벡터를 잘라 분류는 2단계의 검출 과정을 한번 처리하는 과정이다.

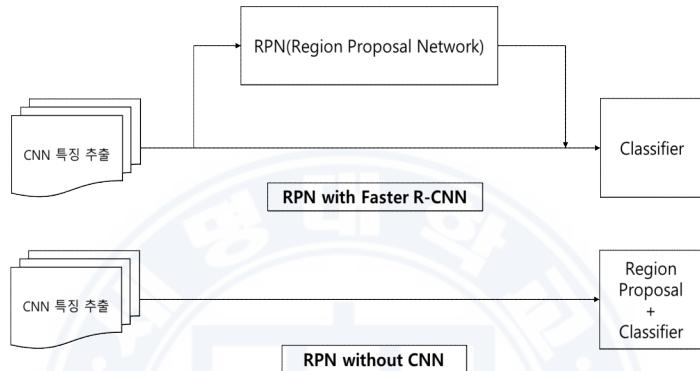


그림 2-15 RPN 과정이 없는 검출 과정

그림 2-15 Detection process without RPN process

RPN을 사용하지 않는 검출 모델로는 YOLO[20], SSD[21] 등이 있다. 그림 2-15 는 RPN 유무에 따른 검출 과정을 나타낸다.

## 2.8 SSD(Single Shot multibox Detector)

앞선 2.7절에서 SSD가 등장하기 전까지 Faster R-CNN이 많이 사용되던 대표적인 검출 모델이었으나, 표 2-3 과 같이 Faster R-CNN 이 가장 느리며, YOLO는 빠르긴 했으나, 검출 성능이 낮다.

표 2-3 검출 모델 비교

Table 2-3 Detection model comparison

검출 모델	m	mAP
Faster R-CNN[19]	7	73.2%
YOLO[20]	45	63.4%
SSD[21]	59	74.3%

반면에 그림 2-16 과 같이 SSD(Single Shot multibox Detector)는 VGG-16[22] 네트워크의 다섯 번째 합성곱 계층까지를 기본 구조로 사용하며, 후보 영역 추출 과정과 Re-sampling 과정을 제거한 방식을 이용하여 높은 검출 성능과 빠른 속도를 얻는 검출 모델이다.

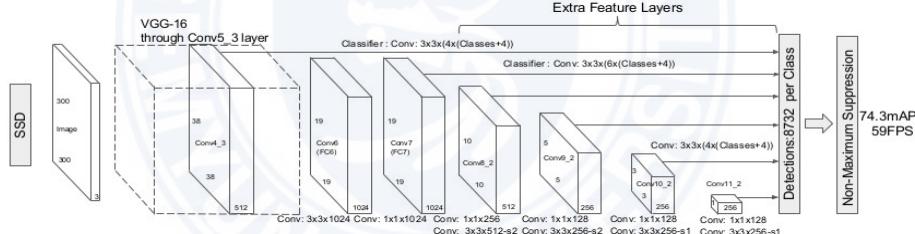


그림 2-16 SSD(Single Shot multibox Detector) 구조  
그림 2-16 SSD(Single Shot multibox Detector) architecture

SSD가 빠른 이유는 SSL(Single Shot Learning)과 보조 구조에 있다. SSL은 사진의 변형 없이 그 한 장으로 훈련, 검출을 하는 검출기를 의미한다. 기존 딥러닝 구조들 중에 VGG-16, AlexNet[]에서 주로 입력 데이터의 크기가 224x224 크기의 입력 데이터를 받는다. 즉, 원하는 이미지에 객체 있는지 없는지 확인하기 위해서는 이미지

를 224x224 크기로 자르거나 변형해야하며 이러한 과정을 다양한 알고리즘을 통해 처리될 때 신경망의 깊이, 데이터의 크기, 추론 및 학습 횟수 차이에 의한 속도 저하가 일어나게 된다. 검출 속도가 빠르지만 한 장의 이미지만을 가지고 여러 가지 크기의 물체를 검출해야 하므로 기본 구조 뒤에 보조 구조를 붙여 얻은 Multi-scale feature maps를 이용한다. 그림 2-17 의 (a)와 같이 이미지에서 개의 크기는 전체 이미지 크기에  $\frac{1}{3}$ 을 차지하며 고양이는  $\frac{1}{6}$ 을 차지 한다. 이를 한 가지 특징맵에서 구하려면 검출 상자(Bounding box)의 크기 차이가 크기 때문에 박스의 크기 추정부터 위치 추정까지 많은 과정이 필요하다. 그러나 SSD는 그림 2-17의 (b),(c) 와 같이 특징맵을 여러 개의 크기로 만들어서 큰 맵에서는 작은 물체를 작은 맵에서는 큰 물체를 검출하도록 설계되었다. 이러한 방식은 위치 추정 및 입력 이미지의 Re-sampling을 없애면서도 정확도 높은 결과를 도출하게 된다[22]

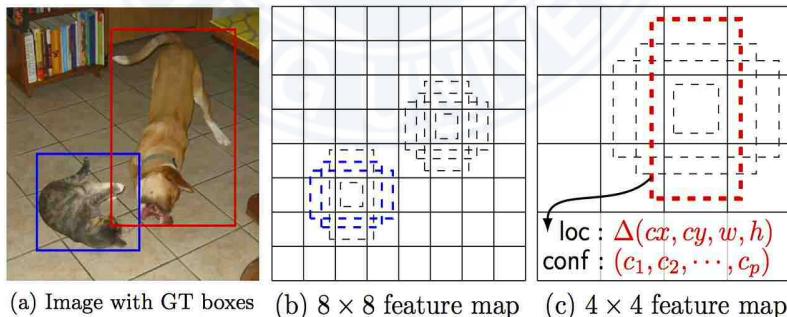


그림 2-17 다중 스케일 특징맵

Fig. 2-17 Multi-scale feature maps

### III. 딥러닝을 이용한 영상 기반 자율주행 자동차 시스템

### 3.1 개요

본 절에서는 딥러닝을 이용한 영상 기반 자율주행 자동차 시스템 구현을 위해 그림 3-1 과 같이 차량의 전방 카메라에서 획득되는 영상을 사전에 학습(Pre-training)시킨 추론 모델을 활용하여 차량, 보행자 등의 객체 검출 박스와 주행 차선 및 차로를 세그먼트 이미지를 영상 기반 디바이스에 출력하여 위험 상황에 대한 경고 및 조향 제어 할 수 있도록 설계하였다.

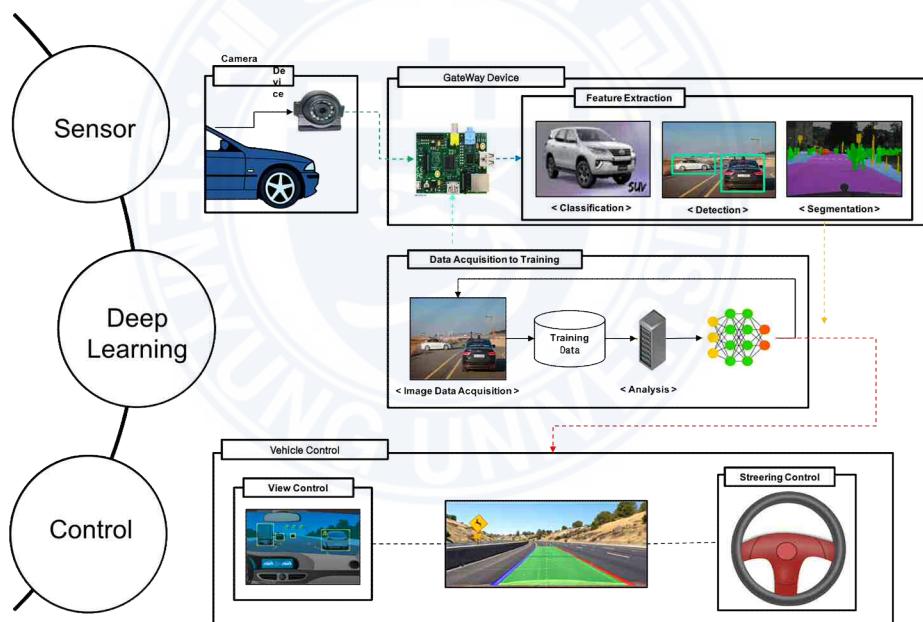


그림 3-1 제안하는 자율주행 시스템 구성도

Fig. 3-1 Autonomous System Configuration Fig.

### 3.2 딥러닝 기반 자율주행을 위한 CNN 구조 제안

본 논문에서는 딥러닝을 이용한 자율주행 자동차를 구현하기 위해 2.8절에서 설명한 SSD 구조를 참고하여 그림 3-2 와 같은 신경망 구조를 활용하여 학습 및 추론할 수 있도록 설계하였다.

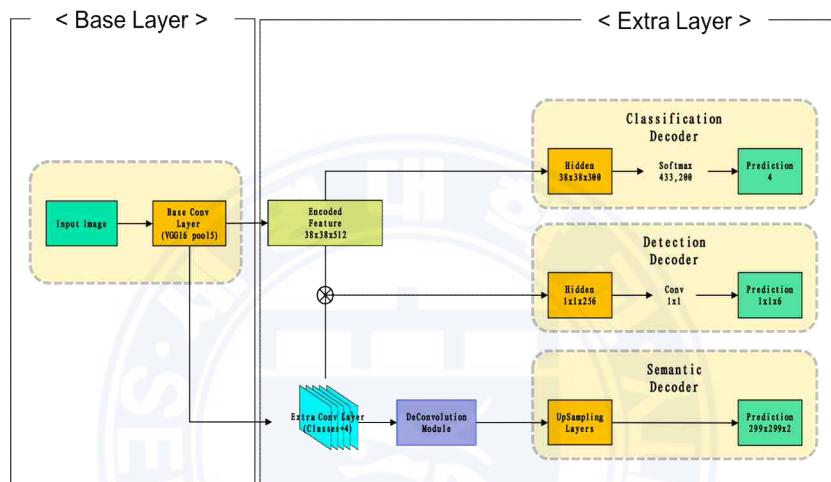


그림 3-2 제안한 신경망 구조

그림 3-2 Proposed neural network structure

제안한 신경망 구조는 Base Layer와 Extra Layer 크게 나눌 수 있다. Base Layer는 2.8절에서 설명한 SSD의 기본 구조인 VGG-16 네트워크 구조를 통해 특징을 추출하는 Layer이다. Base Layer는 차량의 전방 카메라로부터 얻은 영상을 프레임 단위로 이미지 데이터가 입력되며 이미지의 특징을 추출한다. 추출한 특징을 Classification Decoder, Detection Decoder, Semantic Decoder 모듈로 전달한다.

### 3.2.1 사전 데이터 학습

본 논문에서는 자율주행 자동차 시스템을 구현하기 위해 차량, 차선, 차로, 보행자 총 4개의 클래스가 분류된 이미지 데이터를 학습하여 추론 모델을 만든다. 추론 모델은 SSD 네트워크 구조를 기반으로 학습하였다. SSD 네트워크는 앞선 2.8절에서 설명한 객체 검출을 위해 입력 이미지 데이터와 클래스를 분류한 객체의 박스 좌표 데이터를 이용하여 학습한다. 학습시킨 추론 모델을 기반으로 검출할 경우 박스 좌표내의 이미지를 잘라낸다. 잘라낸 이미지의 특징을 추출하여 실제 객체와 비교하여 정답률을 연산한다. 본 논문에서는 자율 주행을 목표로, WIDER FACE, KITTI 등과 같은 연구용으로 제공되는 이미지 데이터를 사용한다.

### 3.2.2 Classification Decoder

그림 3-3 과 같이 추론 모델을 이용한 Classification Decoder는 차선, 주행차로, 차량, 보행자 등 객체를 분류하는 모듈로 Base Layer에서 추출된 특징의 질감이나 색, 강도 등의 정보를 기반으로 객체의 인식률을 연산한다. 제안한 CNN 네트워크 구조는 SSD 네트워크 기반으로 설계하였다. 따라서 객체의 방향과 상호 관계를 고려하기 위해 객체 검출 박스의 최소, 최대 좌표를 예측하여 예측된 박스의 이미지를 잘라내어 Classification Decoder에 전달하여 객체의 방향과 상호관계를 고려할 수 있는 객체분류가 가능하다.

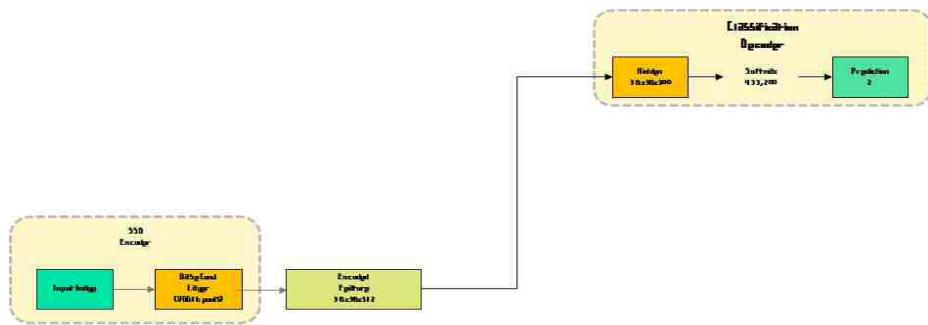


그림 3-3 Classification Decoder

Fig. 3-3 Classification Decoder

### 3.2.2 Detection Decoder

Detection Decoder는 객체의 검출을 위한 모듈이다. 이전 Classification Decoder에서는 객체의 박스 좌표 데이터를 이용하여 이미지를 잘라낸 후 이미지를 분류하였으나, 정확한 객체를 검출 할 수 없다. 따라서 본 절에서는 객체의 예측 박스 좌표와 실제 박스 좌표를 비교하여 정확한 객체를 분할할 수 있는 객체 검출 박스를 예측한다. 그림 3-4 는 Detection Decoder 처리 과정을 나타낸다.

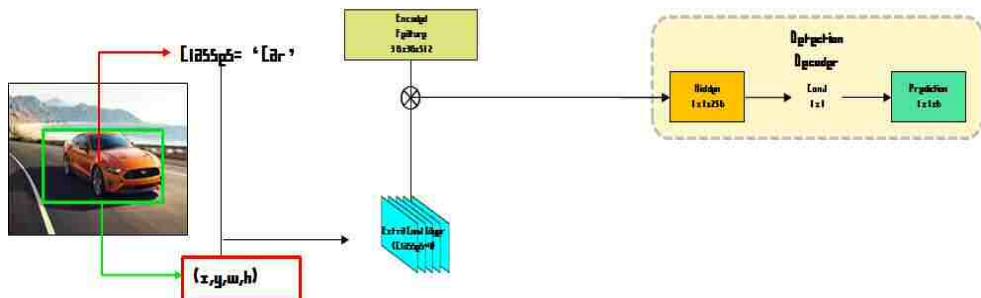


그림 3-4 Detection Decoder

Fig. 3-3 Detection Decoder

### 3.2.3 Semantic Decoder

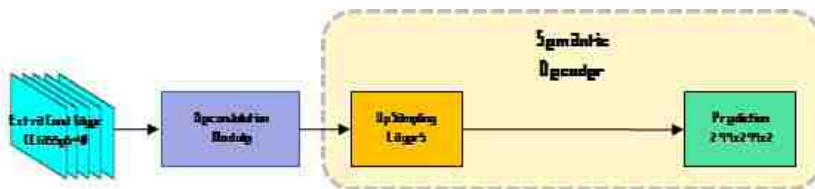


그림 3-5 Semantic Decoder

Fig 3-5 Semantic Decoder

그림 3-5 와 같이 Semantic Decoder는 운전자에게 차선 이탈 경고를 위한 모듈이다. Semantic Decoder에서는 Detection Decoder의 객체 검출 박스 좌표 데이터 4개를 사용하는 것과 다르게 6개를 사용하며 6개는  $y_1, y_2, x_1, x_2, x_3, x_4$  으로 구성하여 차선 후보 점을 예측한 후 컴퓨터 비전 후처리하여 차선을 세그먼트 한다.



그림 3-6 차선 후보점

Fig. 3-6 Lane candidate points

예측한 차선 후보점을 투영변환 하여 차선을 Birdeye View 형태로 변환한다. 그림 3-6 은 Birdeye View 이미지 변환 나타낸다.



그림 3-7 Birdeye View 이미지 변환

Fig. 3-7 Convert Birdeye View image

변환된 Birdeye View 이미지를 기반으로 그림 3-7 과 같이 Threshold 변환 후 좌측 차선을 빨간색과 우측 차선은 파란색으로 분할하여 차선을 검출 및 세그먼트 한다.



그림 3-8 차선 검출

Fig 3-8 Lane detection

차선이 세그먼트된 이미지를 역투영 변환하여 원영상에 이미지를 병합한다. 그림 3-9 는 Semantic Decoder 결과를 나타낸다.



그림 3-9 Semantic Decoder 결과

Fig. 3-9 Semantic Decoder Results

### 3.3 자율주행 차량 제어

3.3절에서는 영상 디바이스 기반으로 운전자에게 운전에 필요한 정보 제공과 이 정보를 기반으로 차량 조향 제어를 한다. 그림 3-10은 자율주행 차량 제어 구성을 나타낸다.

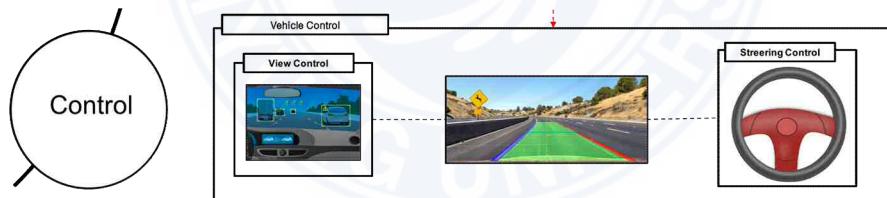


그림 3-10 자율주행 차량 제어 구성

Fig 3-10 Configuration of autonomous vehicle control

영상 디바이스를 통한 정보 제공은 본 논문에서 제안한 CNN 네트워크 구조를 통한 검출 및 인식과 세그먼트 이미지 데이터를 병합하여 영상 디바이스를 통해 출력하며 차량 조향 제어는 병합 이미지에서 좌/우측 차선의 거리를 나타내는 차간 직선과 이 차간 직선의

중간을 나타내는 수직 직선으로 이미지 처리를 한다. 그리고 전체 이미지의 중간을 나타내는 직선을 그린다. 차간의 중앙 수직 직선은 주행 차로의 중간을 나타내며 전체영상의 중간 직선은 차량의 방향을 나타낸다. 이 두 직선이 겹쳐야 올바른 주행하는 상황이며, 차간 중간 직선 보다 차량의 방향이 좌측이면 운전대를 오른쪽으로 우측이면 왼쪽으로 조향 제어를 한다. 그럼 3-11 은 자율주행 차량의 조향 제어를 나타낸다.



그림 3-11 자율주행 차량의 조향 제어  
Fig. 3-11 Steering control of autonomous vehicle

## IV. 실험 내용

### 4.1 Dataset

본 논문에서 제안하는 CNN 네트워크는 Cityscapes 주행 영상 데이터, KITTI Vision 벤치마크 데이터, WIDER FACE의 보행자 데이터를 이용하여 성능을 평가한다. KITTI 벤치마크에는 도심지와 고속도로를 포함한 다양한 주행 환경에 대한 영상을 제공한다. 총 37,878장의 영상을 확보하였다. 제공된 이미지 데이터를 기반으로 제안한 신경망 구조를 통해 학습 및 추론한다. 검출, 세그멘테이션을 수행하기 위해 사용되는 학습 데이터셋과 검증 데이터셋의 구성은 표 4-1 과 같다.

표 4-1 데이터셋 구성

Table 4-1 Data Set Configuration

	Cityscapes	KITTI	WIDER FACE
학습 데이터	8,721	8,759	10,000
검증 데이터	3,546	3,927	2,880
합계			37,878장

### 4.2 Performance Evaluation

네트워크 설계와 실험 평가를 위해 표 4-2 과 같이 intel i5-6500 3.20GHz CPU, NVIDIA GTX-1080 Ti 그래픽카드, 16GB RAM이

탑재된 데스크톱 컴퓨터에 리눅스(Ubuntu16.04) 환경을 구축하였으며, 네트워크 설계와 실험 환경의 구축은 딥러닝 프레임 워크인 텐서플로우를 기반으로 하였다. 기타 영상 처리와 작업의 용이성을 위하여 Scikit-image와 OpenCV 라이브러리를 사용하였다.

표 4-2 실험 환경

Table 4-2 Experiment environment

OS	Ubuntu-LTS 16.04
CPU	Intel i5 @ 3.2GHz Quad
CPU memory	16GB
GPU	GTX1080TI 11GB

성능 평가는 두 단계로 구분하여 수행한다. 제안한 CNN 네트워크 구조와 각각의 작업에 해당하는 Extra Layer를 개별적으로 구성한 세 개의 모듈을 구축하며, 각 작업별 성능을 평가한다. 다음은 본 논문에서 제안하는 Base Layer를 공유하는 세 개의 모듈을 포함된 통합 네트워크의 성능을 평가한다. 성능 평가에 사용되는 지표는 분류, 검출, 세그멘테이션 작업의 목적이 다르기 때문에 각 작업별로 다르게 선택하여 적용하여야 한다.

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

$$F-measuer = (1+\beta^2) \frac{Precision \times Recall}{\beta^2(Precision \times Recall)} \quad (4)$$

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (5)$$

$$AP = \frac{1}{11} \sum_{r \in 0, 0.1, \dots, 1} \max_{\tilde{r}: \tilde{r} > r} Precision(\tilde{r}) \quad (6)$$

분류 성능을 평가하기 위해 사용하는 지표는 accuracy, precision, recall이다. 검출 성능을 평가하기 위해 사용하는 지표는 average precision을 사용하며 Detection Decoder는 KITTI 벤치마크에서 제공하는 차량 검출 데이터에 의해 학습되고 평가한다. 또한 WIDER FACE에서 제공하는 보행자 검출 데이터에 의해 학습된다. 그림 4-1은 Decoder 결과 정확도를 나타낸 그래프이며, 그림 4-2와 그림 4-3은 테스트한 결과 이미지이다.

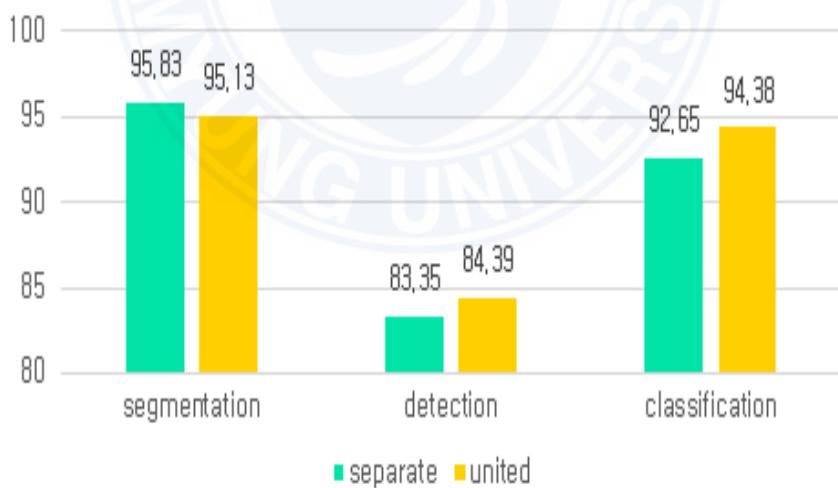


그림 4-1 Decoder 결과 정확도 그래프

Fig 4-1 Decoder Result Accuracy Graph



그림 4-2 차량 검출 이미지

Fig 4-2 Vehicle detection image



그림 4-3 차선 및 주행 차로 세그먼트

그림 4-3 Segment by lane and driving

## V. 결 론

본 논문에서는 딥러닝을 이용한 영상 기반 자율주행 자동차 시스템을 제안하였다. 제안한 시스템은 차량 전방 카메라의 이미지 데이터를 사용하여 신경망 네트워크 구조 중 VGGNet-16 구조가 기본인 SSD 네트워크 구조를 참고하여 단순히 인식만이 아니라 객체의 방향과 위치 그리고 상호관계를 고려한 객체 분류 및 검출과 운전자에게 정보를 제공하기 위한 시맨틱 세그멘테이션을 하도록 신경망을 설계하였다. 제안한 신경망 구조를 통해 사전에 WIDER FACE, KITTI 벤치마크와 Cityscapes에서 제공되는 이미지 데이터를 활용하여 학습시켜 추론 모델을 만들었다. 추론 모델과 검증 이미지 데이터를 제안하는 신경망 구조를 통해 추론하여 자율주행을 위한 정보를 얻을 수 있었으며. 자율주행을 위한 정보는 차량, 차선, 차로, 보행자 4개의 클래스로 분류 및 검출이 가능하며, 차선 및 차로를 검출 및 세그먼테이션이 가능하였다. 이후 추론된 정보를 기반으로 영상 디바이스를 통해 운전자에게 정보를 제공하였으며, 또한 조향 제어를 통한 자율주행이 가능함을 볼 수 있었다.

신경망 설계와 실험 평가를 위해 intel i5-6500 3.20GHz CPU, NVIDIA GTX-1080 Ti 그래픽카드, 16GB RAM이 탑재된 데스크톱 컴퓨터에 리눅스(Ubuntu16.04) 환경을 구축하였으며, 네트워크 설계와 실험 환경의 구축은 딥러닝 프레임 워크인 텐서플로우를 기반으로 하였다. 기타 영상 처리와 작업의 용이성을 위하여 Scikit-image 와 OpenCV 라이브러리를 사용하였다.

성능 평가는 두 단계로 구분하여 수행한다. 제안한 CNN 네트워크 구조와 각각의 작업에 해당하는 Extra Layer를 개별적으로 구성한 세 개의 모듈을 구축하며, 각 작업별 성능을 평가한다. 다음은 본 논문에서 제안하는 Base Layer를 공유하는 세 개의 모듈을 포함된 통합 네트워크의 성능을 평가한다. 평가 결과 각 모듈의 정확도는

Segmentation 95.83% Detection 83.35% Classification 92.65% 결과를 보였으며, 통합된 신경망에서 평가 결과 Segmentation 95.13% Detection 84.39% Classification 94.38% 결과를 얻었다. 통합 신경망에서 Segmentation Decoder 정확도가 떨어졌으나, 나머지 Decoder에서 정확도가 상승함을 보였다. 따라서 제안된 자율주행 시스템을 적용한다면 실시간 이미지 분석을 통한 자율주행이 가능할 것이라 판단된다.

본 논문에서는 차량 전방 카메라와 딥러닝을 활용한 자율주행 시스템을 구현하였다. 제안한 시스템 카메라를 이용한 단일 센서를 활용하였다. 향후 연구에서는 다중 센서를 이용한 실시간 모니터링 및 정보 수집 기술 실시간 제어를 통한 안전성 및 지능화된 자율주행 연구가 진행되어야 할 것이다.

## 참고문헌

- [1] Aly, Mohamed. "Real time detection of lane markers in urban streets." Intelligent Vehicles Symposium, 2008 IEEE. IEEE, pp. 7–12, 2008.
- [2] Anelia Angelova1, Alex Krizhevsky1, Vincent Vanhoucke1, Abhijit Ogale, Dave Ferguson, "Real-Time Pedestrian Detection with Deep Network Cascades." BMVC. Vol. 2. 2015.
- [3] Mariusz Bojarski Holmdel, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba "End to end learning for self-driving cars." arXiv preprint arXiv:1604.07316, 2016.
- [4] Jesung Jeon, Jakap Koo and Changmok Park, 2015, "Outlier Detection in Time Series Monitoring Datasets using Rule Based and Correlation Analysis Method," Journal of the Korean Geo-Environmental Society, Vol. 16, No. 5, pp. 43~53.
- [5] J. Schmidhuber, "Deep Learning in Neural Networks: An Overview," Neural Networks, 61, pp. 85–117, 2015.
- [6] M. Nam, J. Kim, and J. Shin, "A User motion Information Measurement Using Image and Text on Instagram-Based," Journal of Korea Multimedia Society, Vol. 17, No. 9, pp. 1125–1133, 2014.

- [7] Chenyi, et al. "Deepdriving: Learning affordance for direct perception in autonomous driving." Proceedings of the IEEE International Conference on Computer Vision, pp. 2722–2730, 2015.
- [8] NHTSA, Federal\_Automated\_Vehicles\_Policy, 2016. 10.
- [9] McCulloch, Warren S., and Walter Pitts. "A logical calculus of the ideas immanent in nervous activity."
- [10] Rosenblatt, Frank. "The perceptron: A probabilistic model for information storage and organization in the brain." Psychological review, vol. 65, no. 6, p.p386, 1958.
- [11] Minsky, M., Papert. S, "Perceptions", Oxford, England: M.I.T. Press, 1969
- [12] Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams. "Learning representations by back-propagating errors." nature 323.6088, p. 533, 1986.
- [13] Sermanet, Pierre, et al. "Overfeat: Integrated recognition, localization and detection using convolutional networks." arXiv preprint arXiv:1312.6229, 2013.
- [14] Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3431–3440, 2015.
- [15] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E.Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing

- systems, pp. 1097–1105, 2012.
- [16] J.R.R. Uijlings, K.E.A. van de Sande, T. Gevers and A.W.M. Smeulders, “Selective Search for Object Recognition” Technical Report 2012, submitted to IJCV, 2012
  - [17] Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation”, Computer Vision and Pattern Recognition, arXiv:1311.2524
  - [18] Ross Girshick, “Fast R-CNN”, Computer Vision and Pattern Recognition, arXiv:1504.08083
  - [19] Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, Computer Vision and Pattern Recognition, arXiv:1506.01497
  - [20] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, “You Only Look Once: Unified, Real-Time Object Detection”, Computer Vision and Pattern Recognition, arXiv:1506.02640.
  - [21] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg, “SSD: Single Shot MultiBox Detector”, Computer Vision and Pattern Recognition, arXiv:1512.02325. arXiv:1409.1556.
  - [22] Karen Simonyan, Andrew Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition”, Computer Vision and Pattern Recognition,
  - [23] Baoguang Shi, Xiang Bai and Cong Yao, “An End-to-End

- Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition”, Computer Vision and Pattern Recognition, arXiv:1507.05717
- [24] Gökhan Özbulak and Hazim Kemal Ekenel, “Initialization of convolutional neural networks by Gabor filters”, 2018 26th Signal Processing and Communications Applications Conference (SIU), 2018.
- [25] Yuncong Nie, Siyu Xia and Yu Wu, “Wheel classification using convolutional neural networks”, 2018 33rd Youth Academic Annual Conference of Chinese Association of Automation (YAC), 2018.

# An Image Based Autonomous Driving System using Deep Learning

Kyung-Min Lee

*Department of Computer*  
Semyung University Graduate School

## ABSTRACT

In this paper, I proposed an image based autonomous navigation system using deep learning. In order to realize an automobile that can autonomously drive and run safely on the road with a minimum of driver's operation, complex analysis algorithms and sophisticated software are required to collect the information recognized by various sensors and sensors in one place as a single object. The autonomous vehicle is implemented with a great deal of time and money.

Due to the recent development of hardware, deep learning that learns itself on the basis of big data is attracting much attention. The proposed system breaks the barriers of high technological entry through autonomous driving automobile technology using deep learning, and designs autonomous vehicles using universal sensors such as cameras rather than expensive sensors.

In this paper, I utilize the SSD network structure based on VGGNet-16 among the CNN network structure to efficiently process the image classification, object detection, semantic segmentation, and actual road environment images provided by the KITTI benchmark

and Cityscapes benchmark , And some of the provided data was composed of test data. The construction of the learning data was made using the actual road environment image provided by the KITTI benchmark and the Cityscapes benchmark, and some of the supplied data was composed of the test data. The integrated network proposed in this paper can be learned end-to-end by using the learning data provided in the benchmark. I verify the performance and real-time performance of the proposed integrated network through experimental results using test data And confirmed its applicability in actual driving environment.



## 연 구 실 적 목 록

### <국내 등재지>

1. 이경민 외 1명, “효율적인 LED 제어를 위한 다잇 스타 마방진 알고리즘”, 한국인터넷방송통신학회논문지, 제16권 제3호, pp.109-113. 2016/06
2. 이경민 외 1명, “특징점 매칭을 이용한 다중 차량 객체 검출 알고리즘”, 한국ITS학회논문지, 제17권 제1호 통권75호, pp.123-128. 2018/02
3. 이경민 외 1명, “도로 환경에 효율적인 새로운 차선 검출 방법”, 한국ITS학회논문지, 제17권 제1호 통권75호, pp.129-136. 2018/02

### <국제 학술대회>

1. 이경민 외 1명, “A David-Star Magic Square Algorithm for Efficient LED Control”, ITC-CSCC 2016/07
2. 이경민 외 3명, “A Real-time Image Compensation Lane Detection Method using a Dual-Queue”, ICEIC 2017 International Conference on Electronics, Information, and Communication, pp.935-936. 2017/01

### <국내 학술대회>

1. 이경민 외 2명, “이동평균필터를 이용한 영아 돌연사 방지 시스템 설계”, 대한전자공학회 2016 하계종합학술대회논문집, pp.1528-1530. 2016/06.

2. 이경민 외 1명, “이중 큐를 이용한 영상 보정의 실시간 차선 검출 방법”, 한국ITS학회 추계학술대회논문집, pp.312-314. 2016/10.
3. 이경민 외 2명, “쇼핑카트 분실방지 시스템 설계”, 대한전자공학회 2016 추계학술대회논문집, pp.581-583. 2016/11
4. 이경민 외 1명, “영상 분할과 계층적 영역 병합을 이용한 Mean-Shift 추적 알고리즘”, 대한전자공학회 2017 추계학술대회논문집, pp.704-706. 2017/11.

