딥러닝을 활용한 도로 종류 파악

Identifying Road Type by Using Deep-Learning

정 영 찬*

(Jeong Yeong Chan*)

요 약

프로세서의 발달과 자율주행자동차의 보급으로 자율주행자동차의 안전에 대한 요구가 증가하였다. 이에 따라 자동차의 자율주행능력을 완벽에 가까운 수준까지 올리기 위해 다양한 알고리즘이 제안되는데, 그 중 하나는 딥러닝을 이용해 차선을 인식하거나 GPS를 보완하는 것 등이다. 본 논문은 기존 자율주행자동차 코드로부터 산출된 주행이 옳은 방향인지를 검증하기 위한 교차검증용 딥러닝 알고리즘을 제안한다. 도로 종류를 파악할 수 있도록 자율주행자동차가 인지하는 도로 종류를 모아 각각의 데이터셋을 모았으며, 오차를 줄이기 위해 Canny Image를 데이터셋으로 설정하였다. 그리고 구글 Teachable Machine을 통해 데이터를 학습시켰으며, 이 학습된 딥러닝을 통해 파이썬으로 구현할 수 있었다. 구현하는 과정에서 발생된 다양한 문제점들이 보완될 필요성이 있으므로 그 부문을 제안한다.

Keywords : 자율주행자동차, 딥러닝, 케라스

Ⅰ. 서 론

자율주행기능이 포함된 차량이 시중에 속속 등장하면서 자율주행자동차의 보급이 빠르게 진행되고 있다. 특히 자율주행자동차와 전기차의 결합으로 그 보급 속도가 증가하고 있는데, 산업연구원 보고서에 따르면 한국의 경우 자동차 수요가 2009년 146만 1865대에서 2017년 179만 8796대로 평균 성장률이 25.3%에 육박한다.[1] 거기에 리싱크X 보고서에 따르면 2030년 전체 자동차 가운데 60%를, 전체 주행 거리의 95%를 자율주행자동차가 차지할 것이라는 전망을 내놓기도 했다.[2] 이와 같은 증가 추세로 자율주행자동차에 대한 대중들의 인식이 높아지면서 자연스럽게 안전문제와 윤리문제에 관심이 집중되었다.

이에 따라 자율주행자동차에서 가장 중요한 기술인 '자율주행'의 안전을 제대로 확보하기 위해 정부는 145억원을 투입하여 차선인식, 자율주행 기술 개발에 총력을 기울이고 있다.[3] 이와 관련하여 다양한 방안을 활용한 차선인식 방안들이 제시되었는데, 그 중 딥러닝을 활용한 방안이 있다. 기존 연구에 따르면 차량의 정확한 위치 파악을 위해 자동차에 탑제된 센서와 딥러닝을 융합거나[4] 네비게이션에 추가 정보를 제공하기 위한 목적으로 딥러닝과 차선 인식을 활용하거나[5] 차선 변경을 위한 매끄러운 경로를 제공하기 위해 딥러닝을 활용하거나[6] 딥러닝을 사용하여 차선 자체를 인식하는 방법을 제시하기도 한다.[7]

그러나 본 연구는 자율주행자동차가 도로 위에서 주행할 때 Canny Image만을 활용한 차선 인식으로 결정된 차선이 실제 눈에 보이는 방향과 맞는지 교차검증을 위한 목적으로 딥러닝을 이용하고자 한다. 즉, 보조적 용도로 사용가능한 딥러닝 교차검증 시스템을 제작해보고자 한다. 그리고 이를 통해 소형 프로세서를 통해서도 매끄럽고 빠른 연산이 가능한 알고리즘을 위해 관련 방안을 제안한다.

* 정영찬 (선덕고등학교, ingabbang@gmail.com)

Ⅱ. 이 론 적 배 경

1. 딥러닝

딥러닝이란 여러 비선형 변환기법의 조합을 통해 높은 수준의 추상화를 시도하는 기계학습 알고리즘의 집합이다.[8] 이 중 심층 신경망(Deep Neural Network, DNN)은 입력층과 출력층 사이에 여러 은닉층들로 이루어진 인공신경망(Artificial Neural Network, ANN)이다.[8-9] 이때 입력층으로 들어온 특정 이미지의 내용은 은닉층으로 가면서 특정 함수를 거친다. 그 함수는 다음과 같다.

$$a^{(1)} = \sigma(Wa^{(0)} + b) \quad (1)$$

이때 $a^{(1)}$는 다음 층의 뉴런을 의미한다. $\sigma(x)$함수는 로지스틱함수, 혹은 시그모이드 함수(Sigmoid Function)라고 불린다.
그리고 시그모이드 함수 내에 합성되어있는 함수는 가장 간단한 형태의 1차 회귀식으로 다음과 같다.

$$H(x) = Wx + b \quad (2)$$

그리고 이러한 딥러닝 모델을 더 학습시키기 위해서는 비용함수라는 피드백이 필요하다. 식(2)에 존재하는 값 W와 b를 결정할 수 있는 함수를 정해야 한다. 데이터 분포에 따라 정확한 회귀선을 그려야 하기 때문에 각각의 데이터와 회귀선 간 거리가 가장 짧은 선으로 회귀선을 결정해야 한다. 이때 이 두 요소 사이의 거리를 재는 함수를 비용함수(Cost Function)이라고 한다.

$$H(x) - y \quad (3)$$

각각의 요소 하나만을 생각할 때 식(3)으로 따질 수 있다. 식 (3)은 제 1사분면에만 존재하는 데이터에 한하여 양수의 거리 값을 표현한다. 각각의 요소들의 y좌표 값과 함수 값 사이의 거리가 가장 최소가 되는 부분을 정해야 한다. 따라서 비용함수는 다음과 같이 정의된다.

$$Cost = \frac{1}{m}\sum_{i=1}^{m}\left(H(x^{(i)}) - y^{(i)}\right)^2 \quad (4)$$

이때 거리를 계산해야하기 때문에 거리는 제곱한다.
다음과 같은 비용함수를 통해 그 비용함수의 거리가 가장 적은 구간을 계산해내야 하므로 다음과 같이 편미분과 값 대입을 통해 계산한다.

$$repeat\ until\ convergence(\text{수렴})\{$$
$$\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}Cost(\theta_0, \theta_1)$$
$$(\text{for}\ j = 0\ \text{and}\ j = 1)$$
$$\}$$

이때 $H(x)$에서의 W와 b를 각각 θ_1,θ_0으로 나타내었다. 이러한 과정을 '경사하강법(Gradient Descent)'라고 하며, 이와 같은 과정을 통해 비용함수의 거리가 가장 적은 구간을 계산한다. 기울기가 양수면 왼쪽, 음수면 오른쪽으로 이동하여 최종적으로 지역 최솟값(Local Optima)을 계산할 수 있다. 그리고 이와 같은 과정을 통해 α값으로 학습률(Learning Rate)를 계산할 수 있다.
지금까지의 과정은 출력을 모르는 상황에서 입력만 가지고 그 추세를 계산하는 것이었다면, 이번 논문은 입력(도로 사진)과 출력(도로 형태)이 모두 결정되어있으므로 이 데이터를 활용하여 학습시킬 수 있다. 이러한 과정을 위한 것이 역전파 알고리즘이다. 입력한 정보로 산출된 출력층의 결과물과 이미 알고 있는 정답과 비교한다. 이 과정에서 다른 값들에 각각의 차이(bias)를 두어, 즉 역으로 경사하강법을 실시하여 다시 출력층에서 입력층으로 되돌아가는 연산을 거친다. 그리고 이 과정을 입력을 바꾸어가며 반복한다. 이런 과정을 역전파 알고리즘이라고 한다.

이렇게 모인 데이터들을 정립해 평균적인 차이값을 계산하면 되지만 연산이 너무 오래 걸린다. 이런 문제를 해결하기 위해 미니 배치(Mini-Batch)방식을 활용한다. 각각의 데이터를 몇 개의 미니 배치로 묶어, 컴퓨터가 각 미니 배치마다 역전파 알고리즘을 실시한다. 그리고 그 과정에서 발생한 평균 차이값을 계산한다. 이 과정을 미니 배치 수만큼 반복하여 최종 평균 차이값을 계산한다. 이 과정을 미니 배치 확률적 경사 하강법(Mini-Batch Stochastic Gradient Descent)이라고 불린다.

III. 학 습

도로 형태 데이터를 모으기 위해 다음과 같이 도로 종류를 나누었다.

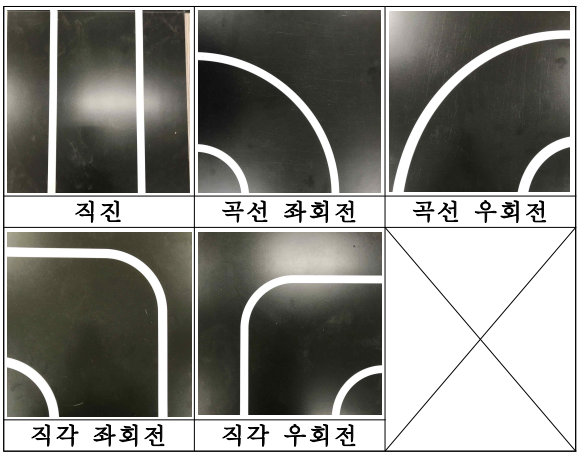


표 1. 도로 종류 구별
Table 1. Classification of Road Type

그리고 이 이미지들은 그대로 활용될 수 없다. 오차율이 높아질 우려가 있으므로 다음과 같은 예로 모든 데이터를 원근왜곡법을 이용하여 실제 도로로부터 위에서 바라보는 형태의 이미지(Bird-view Image)를 산출하고, 그 이미지를 Canny Image화 시켜 오차율을 줄인다. 그 과정은 [그림 1], [그림 2]와 같다.

Canny Image화 시킨 이미지들을 각 항목에 따라 데이터셋으로 제작한다. 각 데이터는 딥러닝이 구별하기 힘든 도로 종류일수록 더 많은 데이터셋을 준비한다. 각각의 데이터 수는 다음 [표 2]를 참고하면 된다.

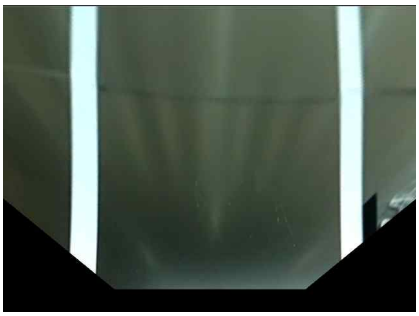


그림 1. 원근왜곡된 도로 사진
Figure 1. Perspective Distorted Road Image

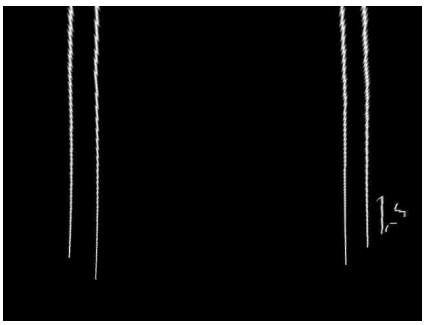


그림 2. Canny Image로 변형
Figure 2. Transform to Canny Image

도로 종류	데이터 수 (단위 : 개)
곡선 좌회전	27
곡선 우회전	29
직각 좌회전	24
직각 우회전	21
직진	14

표 2. 도로 데이터 수
Table 2. The Number of Road Data

이와 같은 데이터 셋을 구글에서 제공하는 Teachable Machin에 적용시킨다. Teachable Machine에 각각 'Right Angle Left', 'Right Angle Right', 'Curve Left', 'Curve Right', 'Straight'로 항목을 구분 짓고 데이터 셋을 적용시켜 학습시킨다. 이때 Epoch는 5000, Batch는 16, Learning Rate는 0.001로 설정하였다.

학습된 모델을 파이썬 코딩으로 적용시킨 후 예측 결과를 확인하였다. 테스트 이미지를 대입하고 각각의 항목으로 얼마나 추정되는지를 %단위로 출력하도록 하였다.

Ⅳ. 결 론 및 제 언

학습시킨 모델로 이미지를 분석한 결과 다음과 같이 도출되었다.

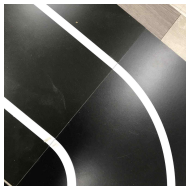
이미지	항목	예측값(%)
	Right Angle Left	50
	Right Angle Right	42
	Curve Left	3
	Curve Right	1
	Straight	2
	Right Angle Left	0
	Right Angle Right	0
	Curve Left	99
	Curve Right	0
	Straight	0

표 3. 결과
Table 3. The Result

도로 종류를 구별하고 이 데이터를 구글 티처블 머신(Teachable Machine)을 이용하여 케라스 파일로 만들었다. 그리고 이 딥러닝 내용을 파이썬 코딩으로 구현하여 도로 종류를 구별할 수 있었다.

그러나 이 과정에서 다양한 오차가 발생하였으며, 특히 차선이 한 개만 인식되는 경우 곡선이 보임에도 불구하고 직선으로 인식하거나, 좌회전 도로임에도 우회전으로 인식하는 문제가 발생하였다. 특히 Canny Image로 데이터를 학습시켰음에도 불구하고 Canny Image로 데이터를 입력했을 때보다 변형 전 이미지를 입력했을 때 결과값이 더 잘 도출되었다. 이러한 내용을 보완하기 위해 딥러닝의 학습 수를 증가시키거나 데이터 수를 증가시키는 방안으로 딥러닝의 정확도를 증가키실 필요가 있다.

또한 Canny Image와 원근왜곡법을 거쳐 발생하는 화질 저하가 딥러닝의 학습률을 떨어뜨려 구분 짓지 못하게 하는 경우도 있다. 따라서 이러한 방안을 해결하기 위해 하드웨어의 개선도 제안된다.

또한 연산속도가 다소 지체되는 부분이 있어 알고리즘이 효율적으로 동작하도록 기존 자율주행자동차 코드를 수정해야 할 필요성이 있다. 딥러닝을 운행하기 위해 자율주행자동차로부터 들어오는 이미지 하나하나를 분석해야 하므로 주행 시 지체되는 부분이 있다. 이러한 부분을 수정해야 할 것을 제안한다.

참고문헌

[1] 이항구, 윤자열. 전기동력·자율주행자동차산업의 현황 및 전망. 산업연구원. 2018.8. 보고서번호:2018-332

[2] James Arbib, Tony Seba. Rethinking Transportation 2020-2030;The Disruption of Transportation and the Collapse of the Internal-Combustion Vehicle and Oil Industries. ReThinkX. 2017.5.

[3] "전기차·자율차 핵심기술 R&D 사업에 올해 279억원 지원". 연합뉴스. 2021.01.27. https://url.kr/658m1r. URL.2021.05.08. 접속.

[4] 유주희, 서재규, 최경택, 정호기. 센서 융합 기반 정밀측위를 위한 딥러닝 기반 차선 끝점 검출. 한국자동차공학회 추계학술대회, pp. 716-717, 2020.11.

[5] 전우태, 김태욱, 최대호, 김종찬. 모바일 내비게이션을 위한 딥러닝 기반 주행 차선 식별. 한국자동차공학회 추계학술대회, pp. 740-741, 2018.11.

[6] 유세욱, 서승우. 딥러닝 기반의 경로 생성을 이용한 차선변경의 개선방안. 대한전자공학회 학술대회. pp.632-633. 2019.11.

[7] 이동현, 이경민, 최병호, 인치호. 딥러닝을 이용한 적응적 차선 검출 알고리즘. 대한전자공학회 학술대회. pp.1571-1572. 2018.6.

[8] Y. Bengio, A. Courville, and P. Vincent., "Representation Learning: A Review and New Perspectives," IEEE Trans. PAMI, special issue Learning Deep Architectures, 2013

[9] J. Schmidhuber, "Deep Learning in Neural Networks: An Overview" http://arxiv.org/abs/1404.7828, 2014