6. 딥러닝 원리 이해하기

1) 사례로 살펴보는 인공 신경망의 원리

인공 신경망은 신경망의 원리를 사용하여 수많은 데이터 사이에서 스스로 특징을 찾아 학습하는 머신러닝 기법 중 하나입니다. 지금부터 이 인공 신경망의 원리를 조금 더 자세히 들어가 봅시다.

다음 세 가지 상황이 있습니다. 이 세 가지 상황에서 문제를 해결할 수 있는 인공지능을 인공 신경망을 사용하여 만들어 볼까요?

- (상황 1) 남녀를 구분하는 인공지능

첫 번째로 만들 내용은 먼저 앞에 있는 사람이 남자인지 여자인지 구별하고 싶은 인공 신경망 모델입니다. 첫 번째 모델을 만든다면 남자와 여자를 구별할 수 있습니다.

A picture containing text

Description automatically generated

(남녀를 구분하는 모습)

-(상황 2) 나이대를 구분하는 인공지능

두 번째로 만들 내용은 특정한 사람의 나이대를 알아맞히는 인공 신경망 모델입니다. 첫 번째 모델은 남자와 여자 둘 중 하나를 구별할 수 있는 인공지능이지만, 이 모델을 둘 중 하나가 아닌 여럿 중 하나를 구별할 수 있는 모델이라는 점에서 차이가 있습니다.

A picture containing diagram

Description automatically generated

(사람의 나이대를 구분하는 모습)

-(상황 3) 정확한 나이를 맞히는 인공지능

마지막은 정확한 나이를 알아맞히는 인공지능을 인공 신경망으로 만드는 것입니다. 두 번째 모델은 여럿 중에서 하나를 고르는 문제지만, 이 모델은 연속된 여러 값 중에서 하나를 예측한다는 점에서 차이가 있습니다.

A cartoon of a child

Description automatically generated with low confidence

(사람의 나이를 예측하는 모습)

이렇게 말하니 두 번째와 세 번째의 차이가 잘 와 닿지 않죠? 두 번째 모델은 우리가 다양한 맛이 있는 아이스크림 가게에 가서 아이스크림 맛을 하나 고르는 문제라면, 세 번째 모델은 특정한 아이스크림의 가격을 맞히는 문제라고 이해하면 되겠습니다. 얼핏 보면 비슷해 보이지만, 엄연히 다른 문제지요.

TIP

각각의 상황은 우리가 앞에서 살펴본 지도 학습의 분류와 회귀 문제입니다. 앞에서 제시한 세 가지 상황을 정리하면, 상황 1은 이항 분류, 상황 2는 다중 분류, 상황 3은 회귀의 문제입니다.

지금부터 이 세 가지 문제를 해결할 수 있는 인공지능을 만들어 보겠습니다. 실제로 코딩하며 만드는 것은 아니니 부담 가질 필요는 없습니다.

2) 인공 신경망의 재료, 여러 특징을 가진 데이터

인공 신경망을 포함해서 머신러닝 방식의 인공지능을 만들기 위해서는 데이터가 필요합니다. 그리고 그 데이터에는 여러 특징이 담겨 있어야 하지요. 여기에서 말하는 여러 특징을 가진 데이터는 데이터의 수를 의미하지 않습니다. 물론 데이터의 수가 많아야 좋은 성능의 인공 신경망 모델을 만들 수 있습니다. 하지만 여기서 말하는 여러 특징의 데이터란 각각의 데이터에 하나의 정보가 아닌 여러 정보가 있어야 한다는 의미입니다.

예를 들어 살펴볼까요? 우리가 남자와 여자를 구분하는 상황 1을 생각해 봅시다. 이때 남자와 여자를 단번에 구분할 수 있는 방법은 무엇일까요? 물론 여러 방법이 있지만, 키를 이용해서 구분해 볼 수 있습니다.

Text

Description automatically generated

(한 가지 정보로 예측하는 모습)

어떤 사람이 남자는 대체로 키가 크고, 여자는 대체로 키가 작다고 생각합니다(물론 이렇게 단정할 수는 없겠지만요). 이 정보만으로 정확하게 구분할 수 있을까요? 남자보다 키가 큰 여자도 있고, 남자아이와 성인 여자는 키라는 정보만 가지고 정확하게 구별할 수 없습니다.

이처럼 한정된 정보만으로 판단한다면 정확하게 판단하기 어렵습니다. 키, 몸무게, 머리카락 길이, 얼굴 길이, 눈, 코, 입의 형태, 몸의 모습 등 정보가 많을수록 더 정확하게 판단할 수 있습니다. 그리고 이는 인공지능에서도 동일하게 나타납니다. 성능이 더욱 뛰어난 인공지능을 만들려면 인공지능이 잘 판단할 수 있도록 여러 정보를 입력할 필요가 있는 것이지요. 이처럼 머신러닝 기법으로 인공지능을 만들 때에는 다양한 특성이 포함된 데이터가 필요합니다.

4 인공 신경망의 작동 모습

여기 남자와 여자를 추론하기 위해 다양한 특성이 포함된 데이터가 있습니다. 이 데이터에 포함된 정보는 바로 키, 머리카락 길이, 얼굴 길이, 성별입니다. 이 데이터는 특징(피처, feature)이 네 개라고 할 수 있겠네요. 물론 실제로 남자와 여자를 추론하기 위해서는 이 정도의 특성으로만 파악하기란 쉽지 않겠지만요.

A picture containing diagram

Description automatically generated

(인공지능의 추론 과정)

머신러닝으로 만든 인공지능이 남자와 여자를 추론하는 모습은 대략 위와 같습니다. 먼저 인공지능이 지금까지 학습한 데이터와 동일한 형태의 데이터를 인공지능에 넣습니다. 이 데이터를 넣으면 검은색 박스를 지나며 결과를 보여줍니다. 이때 이 검은색 박스를 어떻게 만드는지를 결정하는 것이 바로 머신러닝의 다양한 방법들입니다.

우리는 머신러닝의 다양한 방법 중 인공 신경망에 초점을 맞춰 살펴볼 텐데요. 인공 신경망 방식으로 만든 인공지능에서는 입력한 데이터가 여러 레이어를 지나가면서 특정한 신호로 전달됩니다. 그러면 최종적으로 신호가 남자 쪽으로 가는지, 여자 쪽으로 가는지를 판단하여, 둘 중 어느 쪽으로 신호가 많이 가는지를 살펴본 후 신호가 많이 간 쪽 성별이라고 판단을 내리는 것입니다.

A picture containing text, watch

Description automatically generated

(인공 신경망의 추론 과정)

위 그림과 같이 특징 데이터 중 앞의 3개(키, 머리카락 길이, 얼굴 길이)를 넣으니 최종적으로 남자 쪽으로 가는 신호가 여자 쪽으로 가는 신호보다 큰 것을 확인할 수 있습니다.

인공 신경망 모델은 입력 받은 데이터를 사용하여 추론한 결과를 보여줍니다. 이와 같이 새로운 데이터를 받았을 때 그 데이터에서 신호를 남자와 여자 중 어디로, 어느 정도의 세기로 보낼 지를 결정하면서 말이죠.

이때 신호를 정확한 출력 값으로 보내는지, 그렇지 않은지가 바로 그 인공 신경망 모델의 성능을 결정합니다. 남자인데 여자로 예측하거나 여자인데 남자로 예측하는 인공지능을 보고 성능이 좋다고 할 수는 없으니까요.

이때 인공 신경망 모델이 신호를 정확한 출력 값으로 보내지 않는다면, 정확한 출력 값으로 보낼 수 있도록 신호 세기를 조정하는 과정이 바로 인공 신경망의 학습 과정입니다.

3) 딥러닝 개발 환경 살펴보기

딥러닝을 개발하는 방법에는 여러 가지가 있습니다. 물론 딥러닝을 개발하는 것 또한 프로그래밍이기에 프로그래밍 언어를 사용해야 합니다. 그렇다면 프로그래밍 언어만으로도 딥러닝을 개발할 수 있을까요?

어떤 사람이 무엇인가를 만들기 위해 모래를 옮기고 있다고 합시다. 그런데 이 사람이 삽 한 자루만 가지고 있다면 어떨까요? 물론 이 모래를 다 옮길 수는 있겠지만, 그만큼 많이 노력하고 시간을 아주 오래 들여야겠지요. 이 일을 쉽게 하는 방법은 무엇일까요? 바로 한 번에 모래를 많이 옮길 수 있는 굴삭기를 사용하는 것입니다.

딥러닝을 만들 때에도 마찬가지입니다. 프로그래밍 언어를 사용하여 처음부터 하나하나 만들 수 있지만, 그러기에는 너무나 많은 노력이 필요합니다. 모래를 옮길 때 굴삭기를 사용하는 것처럼 딥러닝을 만들 때에도 쉽게 작업할 수 있게 도와주는 도구들이 있습니다. 지금부터 그 도구들을 소개하고, 이 도구들을 이용해서 딥러닝 인공지능을 직접 만들어 보겠습니다.

- 텐서플로 살펴보기

첫 번째로 소개할 도구는 텐서플로(Tensorflow)입니다. 텐서플로는 특정한 프로그래밍 언어가 아니라, 손쉽게 딥러닝 모델을 만들 수 있게 도와주는 프로그래밍 라이브러리입니다. C++ 언어를 기본으로 만들어졌지만, 파이썬, 자바 등 다양한 언어에서 사용할 수 있습니다. 대부분 기능이 파이썬 라이브러리로 구성되어, 파이썬에서 가장 편하게 사용할 수 있답니다.

딥러닝을 만들기 위한 재료는 데이터입니다. 그리고 이 데이터를 사용하여 모델을 학습시킵니다. 텐서플로에서는 이러한 데이터의 모습을 텐서(Tensor)라고 합니다. 이 데이터의 흐름(flow)을 바탕으로 딥러닝이 완성되기 때문에 텐서플로라는 이름이 붙여진 것입니다.

텍스트, 클립아트, 표지판이(가) 표시된 사진

자동 생성된 설명

( 텐서플로의 로고)

라이브러리를 추가하기 위해서는 먼저 여러분의 컴퓨터에 라이브러리가 있어야 합니다. 즉, 라이브러리를 먼저 다운로드해야 한다는 의미죠. 다행히도 우리가 사용하는 코랩은 기본적으로 텐서플로 라이브러리가 설치되어 있습니다. 한번 확인해 볼까요?

코랩에 접속하여 새 노트를 만듭니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

( 코랩 접속 후 새 노트 생성)

코랩에서 텐서플로 라이브러리가 정상적으로 사용 가능한지 확인해 보겠습니다. 정상적으로 설치되었을 경우, 다음과 같이 입력하면 현재 사용 가능한 텐서플로 라이브러리 버전을 알 수 있습니다.

%tensorflow\_version # 텐서플로 라이브러리의 버전을 확인하는 코드입니다.

실행 결과

Currently selected TF version: 2.x

Available versions:

\* 1.x

\* 2.x

현재 텐서플로의 버전은 2.0 버전임을 알려주네요.

2 케라스 살펴보기

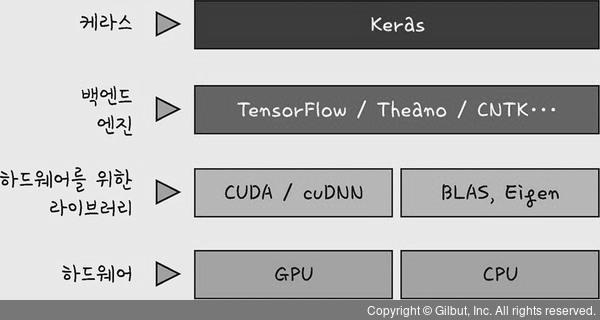
다음으로 소개할 도구는 바로 케라스(keras)입니다. 케라스 또한 텐서플로와 마찬가지로 딥러닝을 만들 때 사용하는 라이브러리입니다. 그럼 케라스와 텐서플로는 같은 것일까요? 정답을 미리 말하자면 그렇지 않습니다.

텐서플로를 사용하면 심층적인 인공 신경망을 손쉽게 만들 수 있습니다. 하지만 케라스를 사용하면 텐서플로만 사용했을 때보다 더 쉽게 이를 만들 수 있죠. 즉, 케라스는 텐서플로를 사용하기 쉽게 만들어 주는 도구라고 볼 수 있습니다. 따라서 케라스를 사용하려면 텐서플로 라이브러리를 불러온 상태여야 합니다.

잠깐만요

텐서플로 이외의 다양한 도구를 사용할 때 케라스를 쓸 수 있나요?

텐서플로는 심층 신경망을 만들기 위해 구글이 만든 도구입니다. 심층 신경망을 만드는 도구에는 텐서플로 이외에도 테아노(theano)나 마이크로소프트에서 만든 CNTK라는 도구가 있습니다. 케라스는 텐서플로 이외에도 테아노나 CNTK를 사용할 수 있습니다. 케라스는 이러한 심층 신경망을 만들기 위한 도구들을 쉽게 사용할 수 있도록 도와주는 도구입니다.



딥러닝 모델을 만들려면 CPU나 GPU와 같은 하드웨어, 이러한 하드웨어를 구동하기 위한 라이브러리가 필요합니다. 그리고 이를 사용하여 인공지능을 만들 수 있는 텐서플로, 케라스 같은 소프트웨어가 필요합니다. 즉, CPU나 GPU 같은 하드웨어를 다룰 수 있는 다양한 라이브러리를 텐서플로, 테아노, CNTK 등에서 제어할 수 있죠. 그리고 텐서플로, 테아노, CNTK 등과 같은 도구를 쉽게 다룰 수 있도록 해주는 것이 바로 케라스입니다.

케라스 홈페이지(https://keras.io/ko/)에서 살펴볼 수 있듯이 케라스는 사용자에게 친화적입니다. 여기서 친화적이라는 말은 사용자가 더 편리하게 사용할 수 있다는 의미죠.

텍스트이(가) 표시된 사진

자동 생성된 설명

(케라스 홈페이지)

케라스 홈페이지의 이념 부분에 ‘기계가 아닌 사람을 위해 디자인된 API’라고 밝히고 있듯이, 케라스는 사람들이 쉽게 사용할 수 있도록 구성되었습니다. 이에 텐서플로를 만든 구글은 사용자들이 케라스를 이용하여 텐서플로를 더 잘 사용할 수 있다고 판단하였습니다.

그래서 2017년부터 텐서플로의 케라스를 통해 핵심 라이브러리를 사용할 수 있게 하였는데, 이는 곧 텐서플로의 다양하고 강력한 기능들을 케라스를 사용하여 구현할 수 있다는 의미입니다.

그럼 지금부터 케라스를 사용하는 방법을 살펴보겠습니다. 일반적인 라이브러리를 불러오듯이 불러오면 됩니다.

import keras # 케라스 라이브러리를 불러오는 코드입니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

(케라스 불러오기)

이렇게 케라스 라이브러리를 불러왔을 때 아무런 오류가 나지 않으면 정상적으로 동작한 것입니다.

3 GPU 살펴보기

심층 신경망 모델을 구성해서 딥러닝을 구현하려면 수많은 계산 과정이 필요합니다. 이러한 계산은 컴퓨터의 중앙처리장치(CPU)가 담당합니다. 잠시 CPU가 작동하는 원리를 살펴볼까요?

CPU는 데이터를 읽어와서, 그 값을 계산한 후 답을 넘겨주는 과정을 수행합니다. 그래서 성능이 좋은 CPU는 한 번에 데이터를 읽어들일 수 있는 능력이 뛰어납니다. 그리고 그 데이터를 계산할 수 있는 능력 또한 뛰어납니다.

CPU는 일의 순서를 잘 계산한 후 순서에 맞춰 계산을 수행합니다. 대부분 프로그램이 복잡한 순서를 가진 알고리즘으로 구현되었기 때문에 CPU는 이러한 프로그램을 잘 순서화해서 일을 처리할 수 있습니다.

테이블이(가) 표시된 사진

자동 생성된 설명

(CPU 내부)

CPU의 처리 과정은 다음과 같습니다. 처리할 일을 메모리(DRAM)에 불러 들어와서 어떤 식으로 처리할 것인지를 결정한 다음(Control Unit) 계산을 처리합니다(ALU, Arithmetic Logic Unit). 이 같은 특징 때문에 CPU는 복잡한 계산을 빠르게 수행할 수 있습니다.

잠깐만요

혹시 코어(core)라고 들어 봤나요?

CPU에서는 한 번에 하나의 일을 수행하기 때문에 동시에 수행할 수 있도록 명령어를 해석하고 계산을 수행하는 부분을 늘리기 시작했습니다. 이 부분이 바로 코어입니다. 그래서 코어가 1개이면 한 번에 하나의 계산을 수행하고, 2개이면 한 번에 두 개의 계산을 수행할 수 있습니다. 일반적으로(물론 항상 그렇지는 않지만) 코어 수가 많을수록 CPU 성능이 좋다고 말합니다.

딥러닝에서 사용하는 계산들은 기존에 CPU가 수행한 계산보다 훨씬 덜 복잡합니다. 따라서 ‘복잡한 계산을 빠르게 하는’ CPU의 장점이 빛을 발하지 못하게 되고, 그러다 보니 딥러닝 계산에 최적화된 처리 장치를 개발하게 됩니다. 복잡한 명령어를 해석하는 장치 부분을 줄이고, 실제 계산을 수행하는 부분을 많이 늘린 것이죠.

GPU는 그 이름에서도 알 수 있듯이 그래픽 작업을 처리하는 용도로 개발되었습니다. 픽셀로 이루어진 그래픽 작업을 수행할 때 계산이 빨라야 더 부드럽고 사실적인 모습을 보여줄 수 있기 때문에 간단하지만 많은 계산을 할 수 있도록 설계된 것입니다.

GPU에는 어떤 식으로 처리할 것인지를 결정하고(Control Unit) 계산을 처리하는 부분(ALU)이 CPU에 비해 상대적으로 많이 있습니다. 물론 처리할 수 있는 능력은 CPU보다 떨어지지만 그 수가 많아, 많은 계산을 동시에 수행하므로 빠른 속도를 나타낼 수 있는 것이죠. 즉, CPU와 달리 GPU는 단순한 계산을 동시에, 그리고 순식간에 할 수 있다는 장점이 있습니다.

테이블이(가) 표시된 사진

자동 생성된 설명

(GPU 내부)

딥러닝 또한 간단하지만 많은 문제를 해결해야 하므로 GPU를 사용하기 시작했습니다. 또한, GPU가 계산 성능이 좋긴 하지만 복잡한 문제를 빠르게 해결할 수 없기 때문에 CPU와 GPU를 적절히 사용하는 것이 중요합니다.

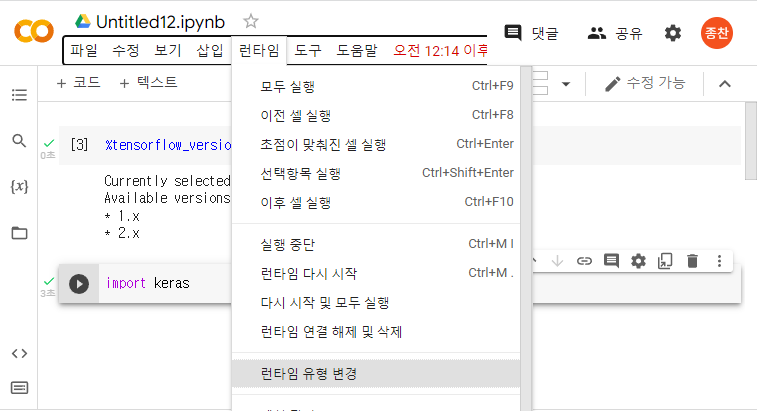
물론 이러한 GPU를 인공 신경망에 사용하기 위해서는 여러분의 컴퓨터에 GPU가 설치되어 있어야 합니다. 하지만 우리가 사용하고 있는 환경인 코랩에서는 구글의 GPU와 TPU를 무료로 사용할 수 있습니다.

잠깐만요

TPU는 무엇인가요?

TPU(Tensor Processing Unit)는 텐서를 처리하기 위해 만들어진 처리 장치입니다. 텐서는 텐서플로 라이브러리에서 사용하는 데이터를 의미합니다. 구글은 이러한 텐서플로라는 머신러닝 라이브러리를 구체적으로 처리하기 위해 자체적으로 TPU를 개발했습니다.

지금부터 코랩에서 GPU를 사용하는 방법을 알아보겠습니다. 그림과 같이 런타임 메뉴 중 런타임 유형 변경을 클릭합니다.



(런타임 → 런타임 유형 변경 클릭)

노트 설정 메뉴에서 하드웨어 가속기를 선택할 수 있습니다. 다음과 같이 GPU와 TPU를 선택할 수 있습니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

(노트 설정 화면)

다음과 같이 GPU와 TPU를 선택할 수 있습니다. 여기서는 GPU를 선택하고 저장 버튼을 클릭합니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

(GPU를 선택한 후 저장)

잠깐만요

‘필요하지 않은 경우 GPU나 TPU를 사용하지 않는 것이 좋다’는 메시지가 나옵니다. 왜 그런가요?

GPU와 TPU를 사용할 때 구글에서 누구에게 먼저 사용할 수 있게 할 것인지 우선순위를 정하였습니다. 오래 실행되는 계산을 사용하는 사용자보다, 대화식으로 코랩을 사용하는 사용자 또는 최근 코랩에서 더 적은 리소스를 사용한 사용자에게 우선순위가 부여됩니다.

결과적으로 장기 실행 계산에 코랩을 사용하는 사용자 또는 최근 코랩에서 더 많은 리소스를 사용한 사용자는 사용이 제한될 가능성이 높아지고 GPU 및 TPU에 대한 액세스가 일시적으로 제한됩니다.

잠깐만요

‘GPU 런타임에 연결되어 있지만 GPU를 활용하고 있지 않다’는 메시지가 나옵니다. 왜 그런가요?

코랩의 설명을 살펴보면 코랩에서는 GPU와 TPU를 포함한 가속 컴퓨팅 환경을 선택 사항으로 제공합니다. GPU 또는 TPU 런타임에서 코드를 실행한다고 해서 반드시 GPU 또는 TPU가 활용되는 것은 아닙니다. GPU 사용량 한도에 도달하지 않도록 GPU를 활용하지 않는 경우 표준 런타임으로 전환하는 것이 좋습니다.

