AI 배움터

- 인공지능과 텐서 -

안녕하세요.

이전 시간까지
대규모 언어 모델 LLM의
추론 상황에 대해 알아보았습니다.

이번 시간에는 인공지능에서
사용하는 텐서의 개념에 대해
알아보겠습니다.

텐서란 무엇인가요?

<스칼라>

숫자 하나가 있습니다.

6

온도 6도, 길이 6m 같은 값입니다.

0차원의 값입니다.

텐서란 무엇인가요?

<벡터>

숫자들을 한 줄로 늘어놓은 것입니다.

[2,4,6]

여러 값을 한 묶음으로 나타낼 수도 있고
방향과 크기를 나타내기도 합니다.

1차원의 값입니다.

텐서란 무엇인가요?

<행렬>

숫자를 가로, 세로로 배치한 표입니다.

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

이 예시는 2행 3열의 행렬이자 2차원 구조입니다.

텐서란 무엇인가요?

<텐서>

텐서는 숫자가 여러 축(axis)을 따라 정리된
배열 전체를 부르는 큰 개념입니다.

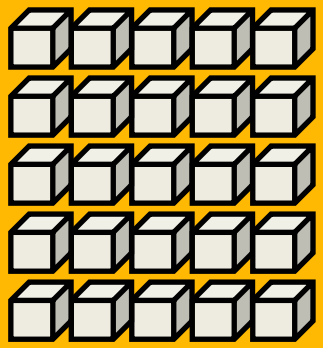
스칼라 → 0차원 텐서
벡터 → 1차원 텐서
행렬 → 2차원 텐서
더 높은 차원의 데이터 → 3차원 이상 텐서

여기서 축은 실제 공간의 벽이나 기둥이 아니라,
데이터를 어떤 기준으로 나누고 정리할지를 나타내는
구조상의 구분 기준입니다.
텐서는 스칼라,벡터,행렬에 그 이상의 고차원 데이터까지
포함하는 상위 개념이라고 생각하면 됩니다.

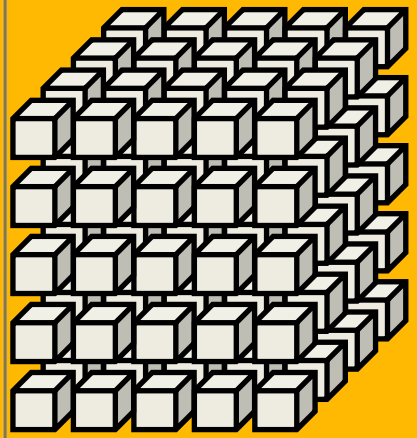
텐서란 무엇인가요?

0차 텐서

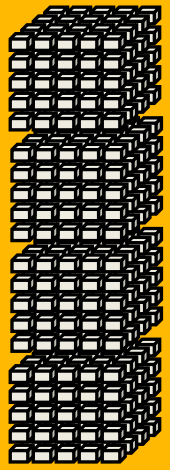
1차 텐서



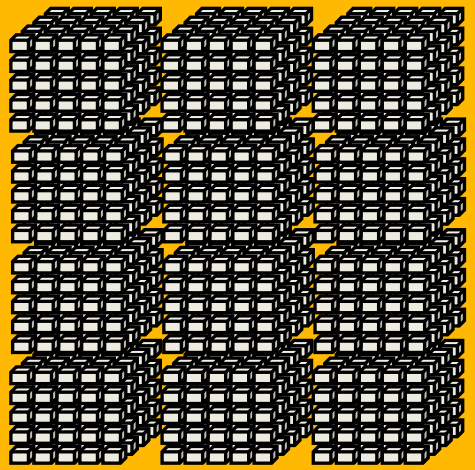
2차 텐서



3차 텐서



4차 텐서



5차 텐서

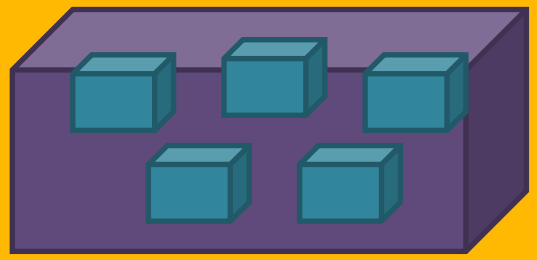
AI 배움터

텐서란 무엇인가요?

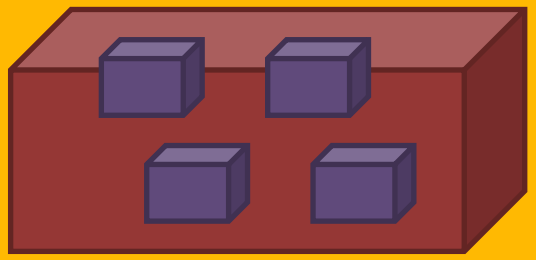
텐서는 차원이 늘어날수록 구분 기준
즉, 축의 수가 하나씩 추가되는 데이터 구조입니다.
주의할 것은 앞 페이지 그림은 편한 이해를 위한
직관적 개념도라는 점입니다.
실제로 4차원 이상은 눈으로 볼 수 없으며
앞 페이지처럼 상자를 쌓아올린 구조가 아니라
데이터를 구분하는 축이 하나 더 추가된다고
이해하는 것이 맞습니다.

4차원 텐서 = 3차원 텐서 여러 개
5차원 텐서 = 4차원 텐서 여러 개
6차원 텐서 = 5차원 텐서 여러 개 ... 이런 식이지요.

3차원 텐서



4차원 텐서



5차원 텐서

텐서란 무엇인가요?

컬러 이미지들을 담은 4차원 텐서가 있다고 가정해 봅시다.
세로 즉 높이의 픽셀 사이즈는 200,
가로 즉 너비의 픽셀 사이즈는 300,
컬러이므로 RGB(Red, Green, Blue) 3개 채널,
한 번에 처리하는 이미지 개수인 배치가 32.
텐서의 구성은 (배치, 높이, 너비, 채널)로 하겠습니다.

※ 인공지능은 한 번에 데이터 하나만 처리하는 게 아니라
여러 데이터를 묶어서 한번에 처리하는 경우가 많습니다.
이를 배치(batch)라고 합니다.

이 텐서의 구조 즉 shape는 (32,200,300,3)로 나타낼 수 있으며,
하나하나의 데이터를 원소라고 한다면 전체 원소 개수는
32x200x300x3 = 5,760,000 개 입니다.
텐서 내부의 원소 하나는 tensor_name[b,h,w,c]로 적습니다.
(ex. Image_tensor[0,0,0,0] : 배치 내 1번째 이미지의 세로 1번째,
가로 1번째, RGB의 1번째 즉 Red에 해당하는 텐서 내 데이터 값.
일반적으로 텐서 인덱스는 0부터 세기 때문입니다.)

인공지능에서 텐서를 사용하는 이유

[1] 다양한 데이터를 공통된 숫자 구조로 표현할 수 있음
인공지능은 글, 이미지, 음성, 영상처럼 서로 다른 형태의 데이터를 다룹니다. 컴퓨터와 인공지능 모델은 이런 데이터를 사람처럼 바로 이해하지 못하고 숫자 형태로 바꿔 처리합니다. 텐서는 이런 숫자 데이터를 일정한 구조로 정리해 담을 수 있는 그릇 역할을 합니다.

컬러 이미지 한 장 : 높이 × 너비 × 채널
컬러 이미지 여러 장 : 배치 × 높이 × 너비 × 채널
문장 데이터 : 배치 × 토큰 수 × 임베딩 차원
영상 데이터 : 배치 × 시간 × 높이 × 너비 × 채널

이처럼 서로 다른 데이터도 텐서 형태로 바꾸면 공통된 방식으로 다룰 수 있습니다.

인공지능에서 텐서를 사용하는 이유

[2] 데이터를 질서 있게 정리하고 관리하기 용이함
텐서는 단순히 숫자를 많이 담는 것만이 아니라,
어떤 숫자가 무엇을 뜻하는지 구조적으로 구분해 줍니다.

Ex. 컬러 이미지 한 장 : 높이 × 너비 × 채널

첫 번째 '높이' 축은 세로 위치
두 번째 '너비' 축은 가로 위치
세 번째 '채널' 축은 색 채널

이라는 의미를 가지며 컴퓨터가 어떤 값을 다룰 때
'이 값은 이미지의 어느 위치의 어떤 색에 대한 값이다'라고
체계적으로 구분할 수 있습니다.
단순히 늘어놓는 것이 아닌 의미 있는 구조로 정리하는 것이지요.

인공지능에서 텐서를 사용하는 이유

[3] 수학 연산에 사용하기 좋음
인공지능 내부에서는 대부분의 핵심 처리가 수학 연산으로 이뤄집니다.

덧셈
곱셈
행렬곱
합성곱
어텐션 계산
정규화

와 같은 연산이 계속 일어납니다. 텐서는 이런 연산을 수행하기 좋은 형태로 데이터를 담고 있으므로,
모델이 입력 데이터를 받아 계산하고 결과를 만드는 과정에 매우 적합합니다. 즉, 텐서는 단순 저장용이 아니라 계산을 전제로 한 데이터 구조라고 할 수 있습니다.

수학 연산을 예로 들어봅시다

1. 덧셈

픽셀 RGB 값이 [120, 100, 60]인데 밝기를 조금 높이기 위해 각 값에 10을 더합니다.

120 + 10 = 130
100 + 10 = 110
60 + 10 = 70

결과 : [130, 110, 70]
텐서에 들어 있는 값들에 같은 덧셈 규칙을 적용합니다.

수학 연산을 예로 들어봅시다

곱셈 예도 있어요.

2. 곱셈

픽셀 RGB 값이 [120, 100, 60]인데 밝기를 20% 높인다면 각 값에 1.2를 곱합니다.

120 x 1.2 = 144
100 x 1.2 = 120
60 x 1.2 = 72

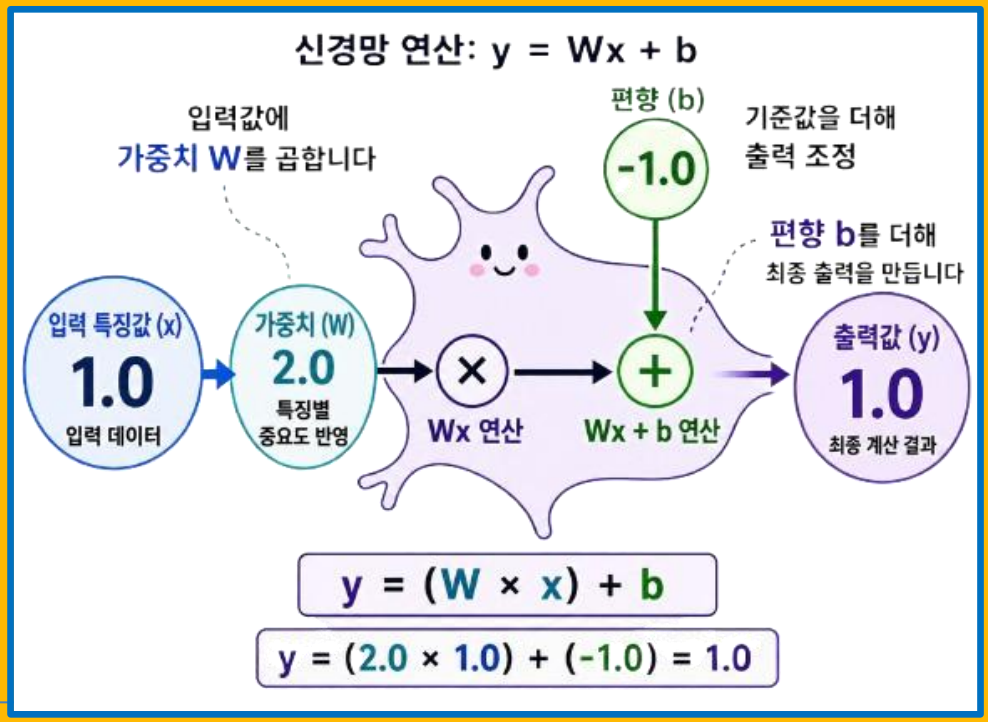
결과 : [144, 120, 72]
텐서에 들어 있는 값들에 같은 배율을 곱하는 연산입니다.

수학 연산을 예로 들어봅시다

3. 행렬곱

인공지능에서의 신경망 기본 계산은 간단하게 y = Wx + b로 표현할 수 있습니다.
이는 입력된 특징값 x에 각 특징의 중요도를 나타내는 가중치 W를 적용하고,
기준을 조정하는 편향 b를 더해 새로운 출력값 y를 만드는 과정입니다.
즉, 신경망은 입력 특징들을 그대로 사용하는 것이 아니라,
각 특징의 중요도를 반영하여 새로운 값으로 만듭니다.
이 과정에서 중요하게 사용되는 연산이 바로 행렬곱입니다.

중요한 개념인
y=Wx+b가
나왔습니다.



수학 연산을 예로 들어봅시다

3. 행렬곱 예시 1

입력값 2개가 들어와 2개의 출력값이 만들어지는 예 입니다.

입력 $x = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$, 가중치 행렬 $W = \begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix}$, 편향 $b = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$ 이면

$Y = Wx + b$이므로

$$Wx = \begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix} \begin{bmatrix} 3 \\ 4 \end{bmatrix} = \begin{bmatrix} 2x3 + 1x4 \\ 1x3 + 3x4 \end{bmatrix} = \begin{bmatrix} 10 \\ 15 \end{bmatrix}$$

여기에 편향을 더하면

$$y = \begin{bmatrix} 10 \\ 15 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 11 \\ 17 \end{bmatrix}$$ 이 됩니다.

수학 연산을 예로 들어봅시다

3. 행렬곱 예시 2

[전제 조건] 인공지능의 신경망은 보통 하나의 층만 있는 게 아니라, 여러 개의 레이어로 이루어져 있는 경우가 많습니다. 이전 레이어에서 만들어진 특징값이 다음 레이어로 전달되고, 이를 바탕으로 더 구체적인 특징이나 최종 결과를 계산합니다.

어떤 사진을 보고 인공지능이 고양이인지, 강아지인지, 토끼인지 판단한다고 가정하겠습니다.
이전 레이어들에서 뽑아낸 특징 3개가 넘어왔습니다.

x1= 귀가 뾰족한 정도 : 2
x2 = 수염처럼 보이는 선의 정도 : 3
x3 = 얼굴이 둥근 정도 : 1

즉 입력값은 다음과 같습니다. $x = \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix}$

수학 연산을 예로 들어봅시다

3. 행렬곱 예시 2

출력 3개를 다음과 같이 정하겠습니다.

y1= 고양이와 닮은 점수
y2 = 강아지와 닮은 점수
y3 = 토끼와 닮은 점수

즉 이 레이어는 입력 특징 3개를 받아, 동물 종류별 점수 3개를 계산하는 레이어라고 볼 수 있습니다.

이때 가중치 행렬 $W = \begin{bmatrix} 2 & 1 & 0 \\ 0 & 1 & 2 \\ 1 & 0 & 1 \end{bmatrix}$, 편향은 $b = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$ 일때

$$Wx = \begin{bmatrix} 2 & 1 & 0 \\ 0 & 1 & 2 \\ 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix} + \begin{bmatrix} 2x2 + 1x3 + 0x1 \\ 0x2 + 1x3 + 2x1 \\ 1x2 + 0x3 + 1x1 \end{bmatrix} + \begin{bmatrix} 7 \\ 5 \\ 3 \end{bmatrix}$$

수학 연산을 예로 들어봅시다

3. 행렬곱 예시 2

여기에 편향 b를 더합니다.

$$y = \begin{bmatrix} 7 \\ 5 \\ 3 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 8 \\ 5 \\ 4 \end{bmatrix}$$

즉 최종 출력은

y1= 고양이와 닮은 점수 : 8
y2 = 강아지와 닮은 점수 : 5
y3 = 토끼와 닮은 점수 : 4 가 됩니다.

이 경우 출력 점수 기준으로 고양이를 닮은 점수가 가장 높으므로, 인공지능은 이 사진을 고양이일 가능성이 가장 높다고 판단합니다.

이 예시는 사진 속 특징들을 바탕으로 동물 종류를 구분하는 인공지능의 간단한 모델이라 할 수 있습니다.
실제 모델은 더 복잡하지만, 가중치와 편향으로 판단 점수를 계산하는 기본 원리는 비슷합니다.

인공지능에서 텐서를 사용하는 이유

[4] 고차원 데이터를 표현하기 쉬움
일반적으로 사람들은 행렬까지 익숙하지만,
실제 인공지능 데이터는 행렬만으로 부족한 경우가 많습니다.

컬러 이미지 한 장 : 높이 × 너비 × 채널 → 3차원
컬러 이미지 여러 장 : 배치 × 높이 × 너비 × 채널 → 4차
영상 데이터 : 배치 × 시간 × 높이 × 너비 × 채널 → 5차

텐서는 이렇게 차원이 늘어나도 같은 방식으로
표현할 수 있는 더 넓은 개념입니다.

[5] GPU 병렬 처리에 유리
GPU는 같은 종류의 계산을 매우 많이, 동시에 처리하는 데
장점이 있습니다. 텐서는 많은 숫자를 일정한 구조로 가지고
있으므로 GPU가 이 값들에 대해 여러 가지 곱셈, 덧셈,
반복적인 연산을 병렬로 수행하기 좋습니다.

인공지능에서 텐서를 사용하는 이유

[6] 학습과 추론을 같은 방식으로 다루기 좋음
인공지능을 학습하고 추론(사용)할 때의 데이터들, 즉

학습 데이터,
정답 데이터,
중간 계산 결과,
가중치 (파라미터),
출력 결과

등을 모두 텐서라는 비슷한 구조로 다룰 수 있습니다.
모델 내부에서 오가는 많은 값들을 텐서라는 공통 구조로 표현할 수 있기 때문에, 학습과 추론 모두를 더 편리하게 다룰 수 있습니다.

인공지능에서 텐서를 사용하는 이유

[7] 인공지능 프레임워크, 라이브러리 특성

현대 인공지능 발전에 가장 중요한 대표적인 프레임워크로는 PyTorch와 TensorFlow를 들 수 있습니다.

- PyTorch : 인공지능 연구와 최신 연구 개발에서 중요한 위치
- TensorFlow : 대형 실무 플랫폼으로서 배포와 실무 적용 측면에서 중요한 위치

※ 프레임워크 : 프로그램이나 서비스를 만들 때 자주 쓰이는 구조와 기능을 미리 갖춰 놓은 개발용 틀입니다. 개발자는 이를 바탕으로 더 빠르고 일관성 있게 작업할 수 있습니다.

이 두 프레임워크 모두 텐서를 중심으로 데이터를 표현하고 연산하므로, 텐서는 이들의 핵심 데이터 구조라고 할 수 있습니다.

PyTorch

TensorFlow

지금까지 인공지능의
중요한 개념인 텐서에 대해
알아보았습니다.

다음 시간에는 더욱
유익하고 재미있는(?)
내용으로 뵙겠습니다