트랜스포머 모델과 GPT

대규모 언어모델 원리 이해하기

충북대학교 의과대학
박 승

충북대학교
CHUNGBUK NATIONAL UNIVERSITY

학습 목표

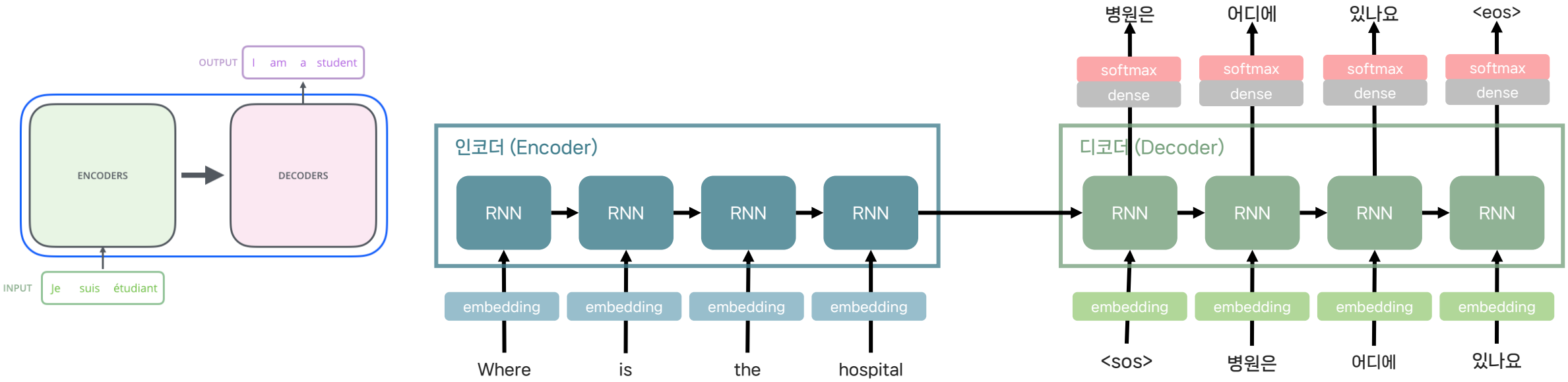
1. 트랜스포머 모델의 기본 구조와 주요 구성 요소의 작동 원리를 이해하고 설명할 수 있다
2. 자연어 처리 모델의 구조적 차이와 작동 원리를 비교하여 설명할 수 있다.
3. GPT 모델의 진화 과정과 성능 개선의 주요 요인에 대해 이해하고 설명할 수 있다.
4. 자연어 처리 모델의 성능을 평가하는 정량적 지표의 개념과 의미를 설명할 수 있다.
5. 트랜스포머 기반 번역 모델을 학습하고, 미세 조정을 통한 성능 개선을 할 수 있다.

CHUNGBUK
NATIONAL UNIVERSITY

순환 신경망을 활용한 자연어 처리

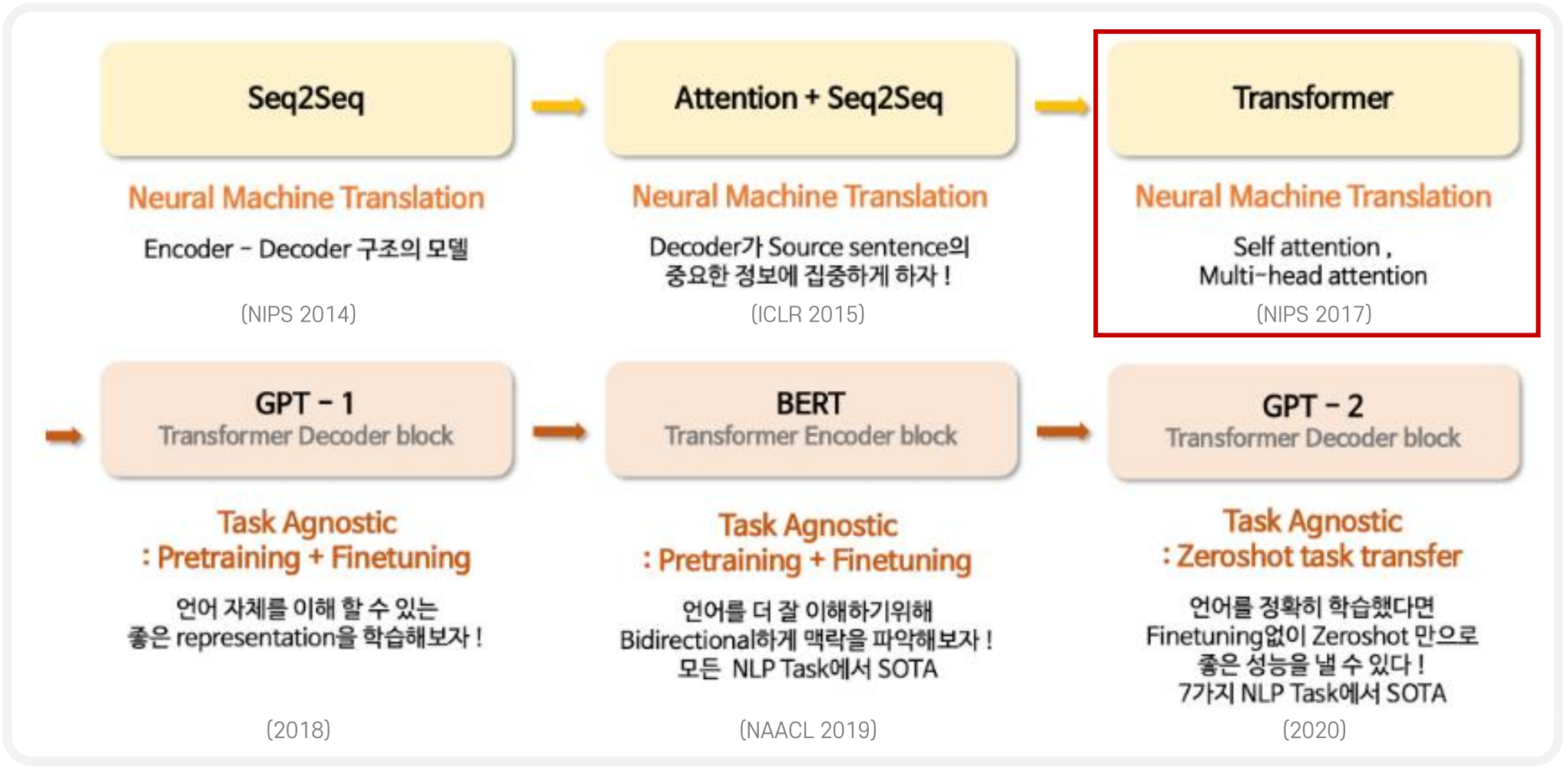
기계 번역기 (machine translator)

- 입력된 문장을 한 언어에서 다른 언어로 자동으로 번역하는 기술
- Encoder (인코더)
  - 입력된 문장을 받아들여 컴퓨터가 이해할 수 있는 내부 표현으로 변환하는 역할
  - 입력 문장의 의미와 맥락을 함축적으로 압축하여 표현
- Decoder (디코더)
  - 인코더가 생성한 문맥 벡터를 기반으로 목표 언어의 문장을 생성하는 역할
  - 문맥 벡터의 정보를 해독하여 원하는 형태의 출력 문장을 생성



자연어 처리 기술의 역사

- 자연어 처리 모델의 성능을 높이기 위해 다양한 기법이 제안됨



트랜스포머 모델

트랜스포머 (transformer)*

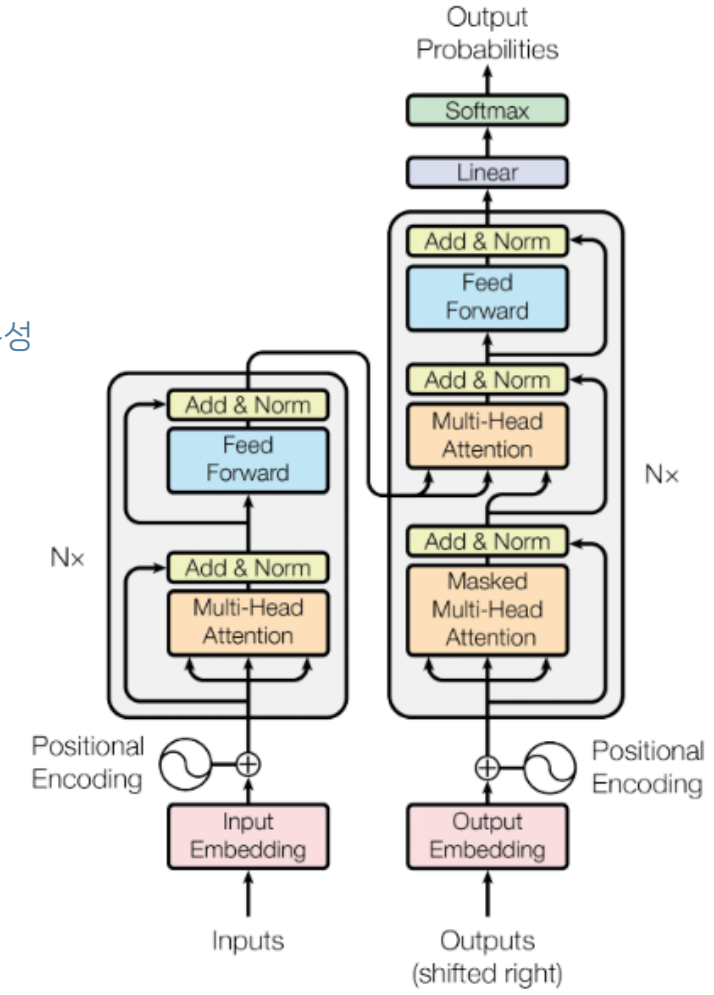
- 구글에서 발표한 자연어 처리 모델
  - 논문 인용 18만 회 이상, NLP 분야의 대표 모델
- 주요 특징
  - 자가 어텐션 기법: 입력된 단어들 중 중요한 단어에 집중하여 문장의 맥락을 효과적으로 이해
  - 병렬 처리 가능: 문장을 순서대로 처리하는 RNN과 달리 한 번에 병렬로 처리하여 학습 속도가 빠름
  - 인코더-디코더 구조: 문장을 인코더에서 벡터로 압축하고, 디코더에서 이를 다시 의미 있는 문장으로 재구성
  - 위치 정보 인코딩: 각 단어의 문장 내 위치 정보를 더해 단어 순서를 이해

Attention is all you need

A Vaswani, N Shazeer, N Parmar… - Advances in neural …, 2017 - proceedings.neurips.cc

… to attend to **all** positions in the decoder up to and including that position. **We need** to prevent … **We** implement this inside of scaled dot-product **attention** by masking out (setting to −∞) …

☆ 저장 99 인용 179790회 인용 관련 학술자료 전체 73개의 버전 »



* Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

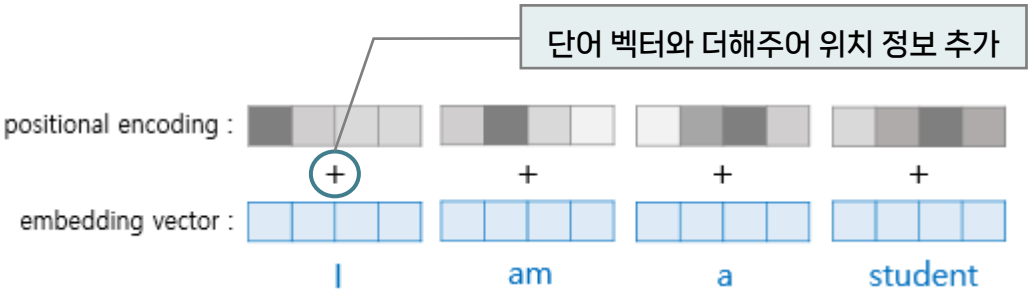
트랜스포머 모델

트랜스포머 모델의 주요 구성요소

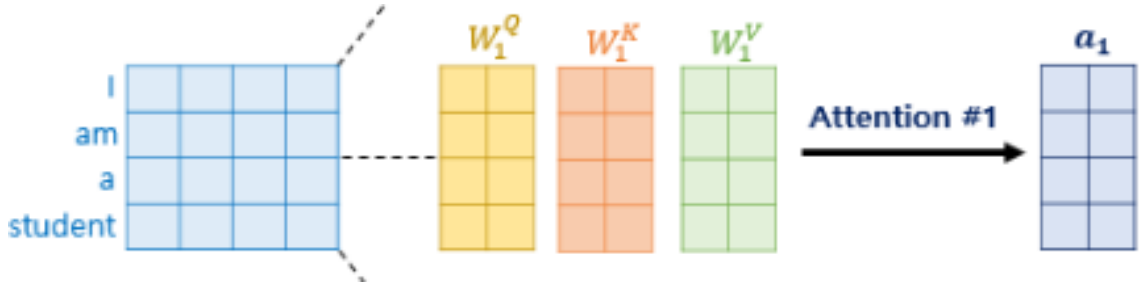
- 1. 위치 정보 인코딩 (positional encoding)
  - 주기 함수를 사용하여 문장 내 단어의 위치 정보를 입력
  - 순환 신경망은 입력이 순차적이지만, 트랜스포머는 한 번에 모든 단어 정보를 입력하므로 문장 내 단어의 위치 정보를 추가해 주어야 함

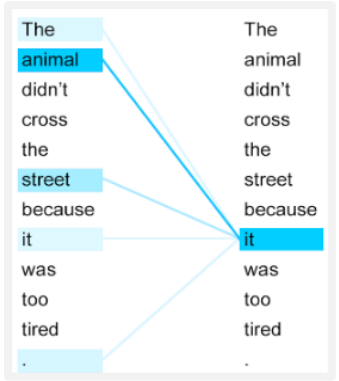
$$PE_{(pos,\,2i)} = sin(pos/10000^{2i/d_{model}})$$

$$PE_{(pos,\,2i+1)} = cos(pos/10000^{2i/d_{model}})$$



- 2. 자가 어텐션 (self-attention)
  - 한 문장 내에서 어떠한 단어가 다른 단어들과 어떠한 연관성을 가지는지 구하는 것
  - 쿼리(Query), 키(Key), 값(Value)을 이용하여 어텐션 스코어 계산

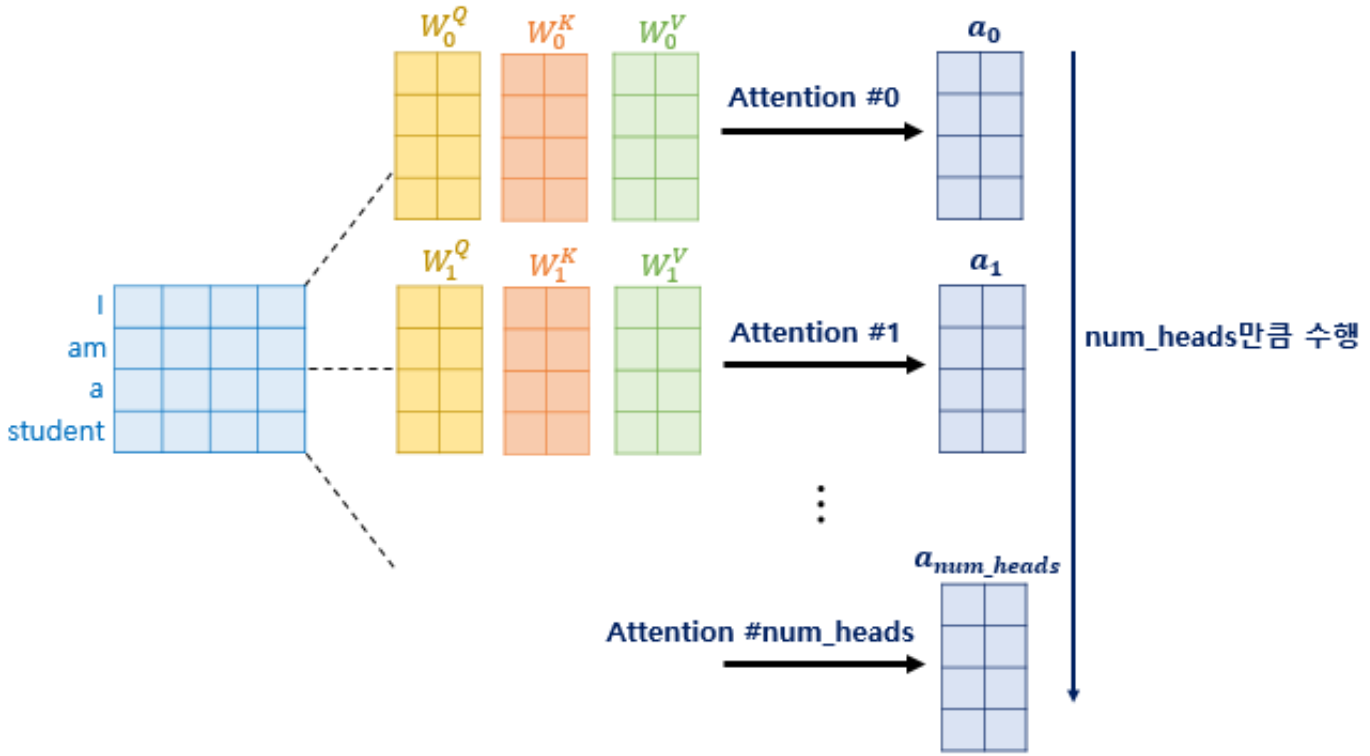




트랜스포머 모델

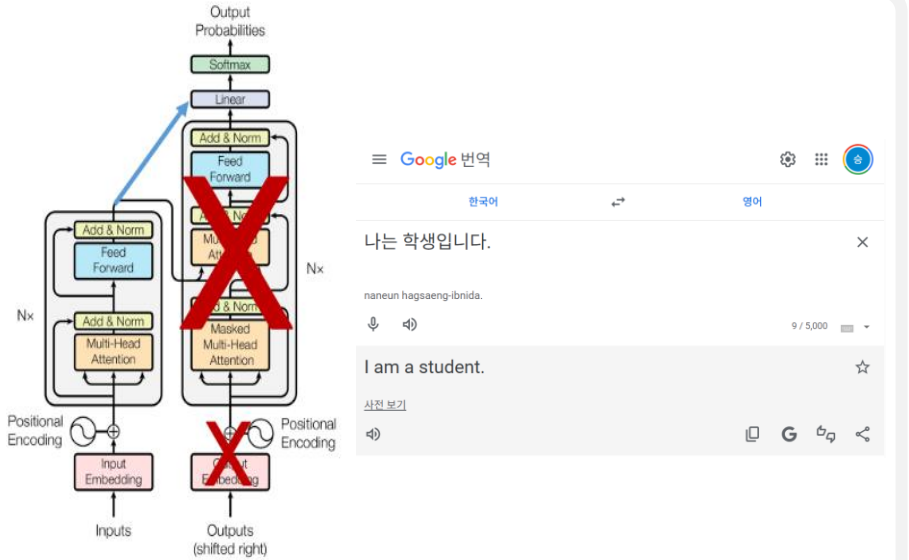
트랜스포머 모델의 주요 구성요소

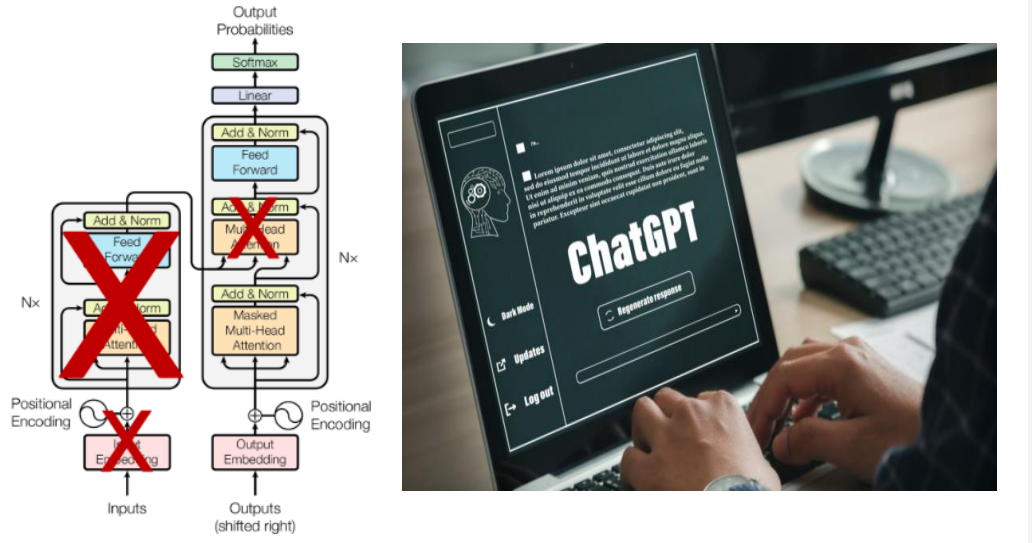
- 3. 다중 어텐션 (multi-head attention)
  - 여러 개의 어텐션 헤드를 병렬로 사용하여 입력 데이터의 다양한 부분에서 서로 다른 관점으로 정보를 추출
  - 입력 차원을 분할하여 여러 헤드로 처리함으로써, 모델의 크기를 크게 확장하지 않으면서도 계산 복잡성을 증가시키지 않고 성능을 향상



트랜스포머 기반 자연어 처리 모델

- 양방향 인코더 모델: BERT (bidirectional encoder representations from transformers)
  - 문장 전체의 맥락을 양방향(bidirectional)으로 파악하는 모델
  - 문장 내 일부 단어를 가림 처리(masking)하고, 가려진 단어를 주변 맥락을 통해 예측하며 학습
- 단방향 디코더 모델: GPT (generative pre-trained transformer)
  - 한쪽 방향(unidirectional)으로 이전의 단어만 참고하여 다음 단어를 순차적으로 예측하는 모델
  - 주어진 입력에 따라 다음 단어를 계속 생성하는 자가회귀(auto-regressive) 방식으로 작동





트랜스포머 기반 자연어 처리 모델

- GPT 모델의 의미

Generative

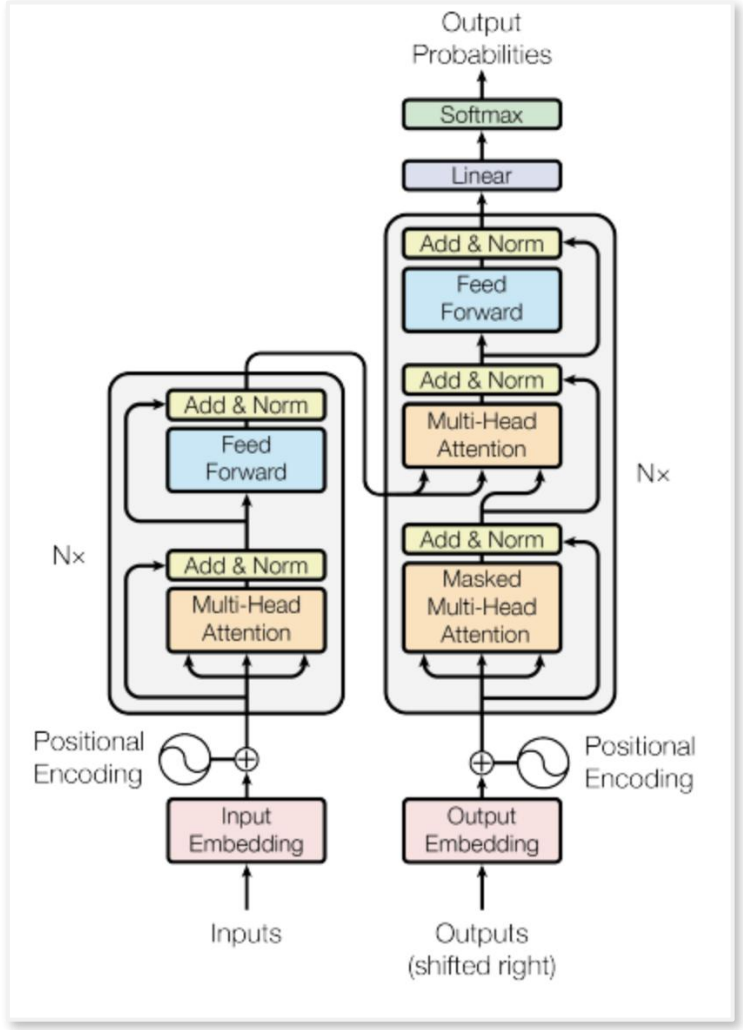
원하는 답을 **생성**할 수 있도록

Pre-trained

방대한 데이터셋을 통해 **사전 학습**된

Transformer

트랜스포머 기반 자연어 처리 모델



트랜스포머 기반 자연어 처리 모델

GPT 모델의 진화 과정

- 모델 매개변수(parameter)가 증가하여 성능 개선
- GPT-4 버전에서는 이미지, 문서 등 다양한 데이터도 처리 가능

Basis of Distinction	GPT 1	GPT 2	GPT 3	GPT 3.5	GPT 4
Parameters	117 million	1.5 billion	175 billion	1.5 billion	1.7 trillion
Context Length	Up to 1024 tokens	Up to 2048 tokens	Up to 2048 tokens	Up to 4000 tokens	Up to 32000 tokens
Transformer Layers	12	48	96	96	120
Multilingual Capabilities	Only understands English	Only understands English	Understands several languages with proficiency in English	Understands several languages with proficiency in English	Proficient in multiple languages like Polish and German
Performance	Basic tasks like summarization	Large number of NLP tasks with high precision, along with the ability to have human-like conversations	Large number of NLP tasks with high precision, along with the ability to have human-like conversations	Highly coherent conversations, with the ability to perform tasks accurately with little to no training	Can perform various tasks with the highest precision in GPT models so far
Internet Access	None	None	None	None	Can access the internet through third-party browsers
Modality	Textual	Textual	Textual	Textual	Texts & Images

트랜스포머 기반 자연어 처리 모델

- PaLM 2 (Google)
  - Google의 챗봇 Bard의 기반 모델
  - 다국어 및 추론 능력 향상에 중점을 둔 모델

- LLaMa (Meta AI)
  - 연구 및 상업적 사용을 위해 일반 대중에게 공개된 오픈소스 모델
  - 다양한 크기로 제공되어 필요에 따라 선택하여 사용 가능

- Claude (Anthropic)
  - 연구 및 상업적 사용을 위해 일반 대중에게 공개된 오픈소스 모델
  - 윤리적 문제를 중요시하여 해로운 출력을 줄이는 방향으로 특별히 훈련됨

- DeepSeek (DeepSeek AI)
  - 고성능 추론과 효율적인 파라미터 활용에 중점을 둔 오픈소스 모델
  - 다양한 자연어 처리 응용 분야에서 활용 가능

자연어 처리 모델의 정량적 성능 지표

NLL (Negative Log-Likelihood)

- 주어진 데이터에 대해 모델이 예측한 확률 값에 로그를 취한 뒤 음수(-)를 곱한 값으로, 모델 예측의 정확도를 나타내는 지표
- NLL 값이 작을수록 모델이 데이터에 대해 높은 확률로 예측하고 있음을 의미

$$\mathrm{NLL} = -\sum_{i=1}^{n} \log P(x_i)$$

PPL (Perplexity)

- 모델이 주어진 데이터셋을 얼마나 정확히 예측할 수 있는지를 나타내는 지표로, 데이터셋에 대한 예측의 불확실성 정도를 의미
- Perplexity는 NLL을 데이터 수로 나누고 지수 함수를 적용하여 얻어지며, 값이 작을수록 모델의 예측 성능이 우수함

$$\mathrm{PPL} = \exp\left(\frac{\mathrm{NLL}}{n}\right)$$

자연어 처리 모델의 정량적 성능 지표

BLEU (Bilingual Evaluation Understudy Score)

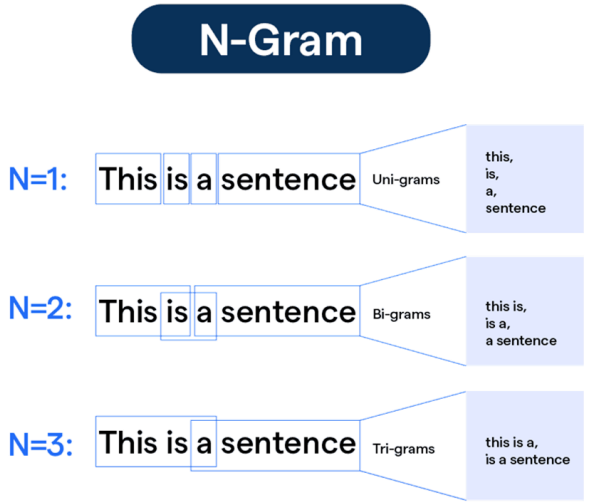
- 번역 모델 성능 평가 지표로, 예측된 번역 문장과 정답 문장 간의 단어 n-gram 일치도를 측정하여 정량적으로 평가
- 값이 클수록 번역 품질이 우수함을 의미

ROUGE (Recall-Oriented Understudy for Gisting Evaluation)

- 주로 텍스트 요약 모델의 성능을 평가하는 지표로, 예측된 요약 문장과 정답 요약 문장 간의 n-gram 겹침 정도를 평가하여 정량화
- 값이 클수록 모델이 원문에서 중요한 정보를 잘 추출하여 요약하고 있음을 의미

BERTScore

- 사전 학습된 BERT 모델의 임베딩을 활용하여 문장 간 의미적 유사도를 평가하는 지표
- 문장 의미를 더 정확하게 반영할 수 있으며, 값이 높을수록 정답 문장과 의미적으로 유사함



자연어 처리 모델의 정량적 성능 지표

실습 1) 트랜스포머 모델 기반 한글-영어 번역기

- 1. 라이브러리 설치 및 사전 학습 모델 불러오기

```
pip install torch transformers nltk
```

```
train_data = [
    ("안녕하세요.", "Hello."),
    ("오늘 날씨가 좋습니다.", "The weather is nice today."),
    ("이것은 테스트 문장입니다.", "This is a test sentence."),
    ("나는 인공지능을 좋아합니다.", "I like artificial intelligence."),
    ("이 모델은 성능이 좋습니다.", "This model has good performance."),
    ("저는 매일 아침 커피를 마십니다.", "I drink coffee every morning."),
    ("지금 몇 시입니까?", "What time is it now?"),
    ("내일은 비가 올 것입니다.", "It will rain tomorrow."),
    ("서울은 한국의 수도입니다.", "Seoul is the capital of Korea."),
    ("기계학습은 매우 흥미롭습니다.", "Machine learning is very interesting."),
    ("오늘 저녁에 영화 보러 갈까요?", "Shall we go watch a movie tonight?"),
    ("당신은 어떤 음식을 좋아합니까?", "What kind of food do you like?"),
    ("한국어를 배우는 것은 쉽지 않습니다.", "Learning Korean is not easy."),
    ("그는 매일 아침 운동을 합니다.", "He exercises every morning."),
    ("그녀는 책을 읽고 있습니다.", "She is reading a book."),
    ("이 컴퓨터는 너무 느립니다.", "This computer is too slow."),
    ("새로운 프로젝트는 잘 진행되고 있습니다.", "The new project is going well."),
    ("학교가 끝난 후에 무엇을 합니까?", "What do you do after school?"),
    ("휴가 때 어디로 갈 계획입니까?", "Where do you plan to go on vacation?"),
    ("그 회사는 신제품을 출시했습니다.", "The company has released a new product."),
]

test_data = [
    ("안녕히 가세요.", "Goodbye."),
    ("당신의 이름은 무엇입니까?", "What is your name?"),
    ("식사는 하셨습니까?", "Have you eaten?"),
    ("여기에서 지하철역까지 얼마나 걸립니까?", "How long does it take to the subway station from here?"),
    ("저는 여행을 좋아합니다.", "I like traveling."),
    ("도와주셔서 감사합니다.", "Thank you for your help."),
    ("내일 아침에 일찍 일어나야 합니다.", "I have to get up early tomorrow morning."),
    ("회의는 언제 시작합니까?", "When does the meeting start?"),
    ("지금 배가 고픕니다.", "I'm hungry now."),
    ("이 문제를 해결할 방법이 있습니까?", "Is there a way to solve this problem?")
]
```

```
import torch
from transformers import MarianMTModel, MarianTokenizer
import nltk
from nltk.translate.bleu_score import corpus_bleu

nltk.download('punkt_tab')

# 모델 및 토크나이저 로딩
model_name = "Helsinki-NLP/opus-mt-ko-en"
tokenizer = MarianTokenizer.from_pretrained(model_name)
model = MarianMTModel.from_pretrained(model_name)
```

자연어 처리 모델의 정량적 성능 지표

- 실습 1) 트랜스포머 모델 기반 한글-영어 번역기
  - 2. 모델 성능평가 및 미세학습 함수 정의

```python
# 평가 함수 정의
def evaluate_bleu(model, tokenizer, dataset):
    references, hypotheses = [], []
    model.eval()

    with torch.no_grad():
        for ko, en in dataset:
            inputs = tokenizer(ko, return_tensors="pt")
            outputs = model.generate(**inputs)
            pred = tokenizer.decode(outputs[0], skip_special_tokens=True)

            references.append([nltk.word_tokenize(en.lower())])
            hypotheses.append(nltk.word_tokenize(pred.lower()))

    bleu_score = corpus_bleu(references, hypotheses)
    return bleu_score

# 미세학습 함수 정의
def fine_tune(model, tokenizer, dataset, epochs=10):
    optimizer = torch.optim.AdamW(model.parameters(), lr=1e-5)

    model.train()
    for epoch in range(epochs):
        total_loss = 0
        for ko, en in dataset:
            inputs = tokenizer(ko, return_tensors="pt")
            labels = tokenizer(en, return_tensors="pt").input_ids
            labels[labels == tokenizer.pad_token_id] = -100

            loss = model(**inputs, labels=labels).loss
            loss.backward()
            optimizer.step()
            optimizer.zero_grad()

            total_loss += loss.item()
        print(f"Epoch {epoch+1}/{epochs}, Loss: {total_loss/len(dataset):.4f}")
```

자연어 처리 모델의 정량적 성능 지표

실습 1) 트랜스포머 모델 기반 한글-영어 번역기

- 3. 모델 미세학습 수행

```python
# Fine-tuning 이전 BLEU 점수 평가
bleu_before = evaluate_bleu(model, tokenizer, test_data)
print(f"Fine-tuning 이전 BLEU 점수: {bleu_before:.4f}")

# 미세학습 실행
fine_tune(model, tokenizer, train_data, epochs=10)

# Fine-tuning 이후 BLEU 점수 평가
bleu_after = evaluate_bleu(model, tokenizer, test_data)
print(f"Fine-tuning 이후 BLEU 점수: {bleu_after:.4f}")
```

```
Fine-tuning 이전 BLEU 점수: 0.5459
Epoch 1/10, Loss: 6.3585
Epoch 2/10, Loss: 5.3615
Epoch 3/10, Loss: 4.5115
Epoch 4/10, Loss: 3.8598
Epoch 5/10, Loss: 3.2416
Epoch 6/10, Loss: 2.8321
Epoch 7/10, Loss: 2.4315
Epoch 8/10, Loss: 1.9691
Epoch 9/10, Loss: 1.6559
Epoch 10/10, Loss: 1.4338
Fine-tuning 이후 BLEU 점수: 0.5937
```

자연어 처리 모델의 정량적 성능 지표

실습 1) 트랜스포머 모델 기반 한글-영어 번역기

- 4. 번역 결과 확인

```python
# 번역할 문장 리스트
sentences_ko = [
    "오늘 정말 피곤하네요.",
    "주말에 영화 보러 가실래요?",
    "서울에는 맛있는 식당이 많습니다.",
    "내일 오전에 회의가 있습니다.",
    "이 컴퓨터는 속도가 매우 빠릅니다."
]

# 번역 및 출력 함수
def translate_sentences(model, tokenizer, sentences):
    model.eval()
    with torch.no_grad():
        for sentence in sentences:
            inputs = tokenizer(sentence, return_tensors="pt")
            translated = model.generate(**inputs)
            translated_text = tokenizer.decode(translated[0], skip_special_tokens=True)
            print(f"원문: {sentence}")
            print(f"번역: {translated_text}\n")

# 번역 실행
translate_sentences(model, tokenizer, sentences_ko)
```

```
원문: 오늘 정말 피곤하네요.
번역: I'm really tired today.

원문: 주말에 영화 보러 가실래요?
번역: Would you like to go to a movie on the weekend?

원문: 서울에는 맛있는 식당이 많습니다.
번역: There are a lot of delicious restaurants in Seoul.

원문: 내일 오전에 회의가 있습니다.
번역: There is a meeting in the morning.

원문: 이 컴퓨터는 속도가 매우 빠릅니다.
번역: The computer is very fast.
```

감사합니다

Q&A

충북대학교
CHUNGBUK NATIONAL UNIVERSITY