# 언어 모델

- 입력 값(단어 시퀀스 or 문장 or 문서)을 기반으로 통계적으로 가장 적절한 출력 값을 도출하도록 학습된 모델

- 단어 시퀀스(문장 or 문서)에 확률을 할당

- 가장 자연스러운 단어 시퀀스를 찾는 모델

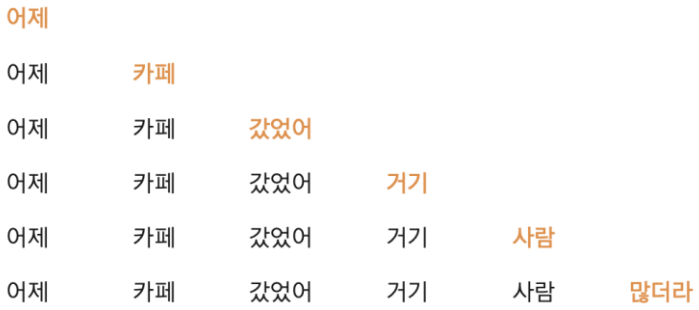
- 조건부 확률로 표현하면 다음과 같음

$$P(w_1, w_2, w_3) = P(w_1) \times P(w_2 \mid w_1) \times P(w_3 \mid w_1, w_2) \rightarrow P(w_1, w_2, w_3, w_4, \ldots, w_n) = \prod_{i=1}^{n} P(w_i \mid w_1, \ldots, w_{i-1})$$

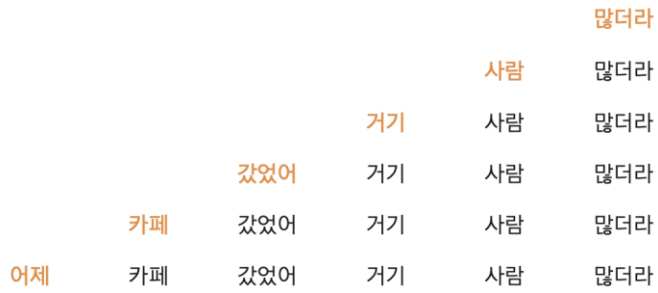
  - » 전체 단어 시퀀스가 나타날 확률은 이전 단어들이 주어졌을 때 다음 단어가 등장할 확률의 연쇄와 같음

# 언어 모델

- 종류
  - » 순방향 언어 모델
    - › 문장 앞부터 뒤로, 사람이 이해하는 순서대로 계산하는 모델
    - › GPT, ELMo 등




  - » 역방향 언어 모델
    - › 문장 뒤부터 앞으로 계산하는 모델
    - › ELMo 등

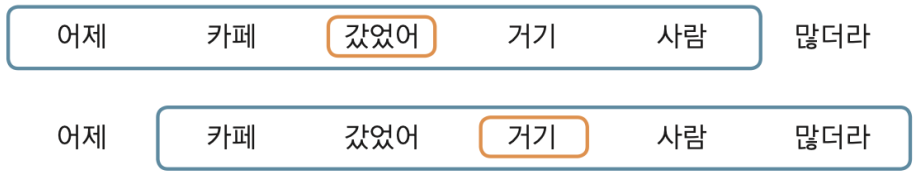


# 언어 모델

- 종류 (계속)
  - » 마스크 언어 모델
    - › 학습 대상 문장에 빈칸을 만들고 해당 빈칸에 적절한 단어를 분류 모델로 학습
    - › BERT 등

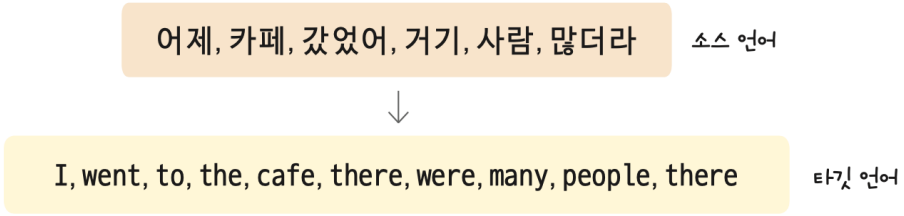
| | | | | | |
|---|---|---|---|---|---|
| 어제 | 카페 | 갔었어 | 거기 | 사람 | 많더라 |
| 어제 | 카페 | 갔었어 | 거기 | 사람 | 많더라 |
| 어제 | 카페 | 갔었어 | 거기 | 사람 | 많더라 |
| 어제 | 카페 | 갔었어 | 거기 | 사람 | 많더라 |
| 어제 | 카페 | 갔었어 | 거기 | 사람 | 많더라 |
| 어제 | 카페 | 갔었어 | 거기 | 사람 | 많더라 |

  - » 스킵-그램 모델
    - › 어떤 단어 앞뒤에 특정 범위를 정해 두고 이 범위 내에 적절한 단어를 분류 모델로 학습
    - › Word2Vec 등



# Sequence to Sequence

- 특정 속성을 지닌 시퀀스를 다른 속성의 시퀀스로 변환하는 작업




- Sequence
  » 단어 등의 나열 ( 연속된 데이터, 리스트 )

- Transformer
  » 기계 번역 등 sequence-to-sequence 과제를 수행하는 모델

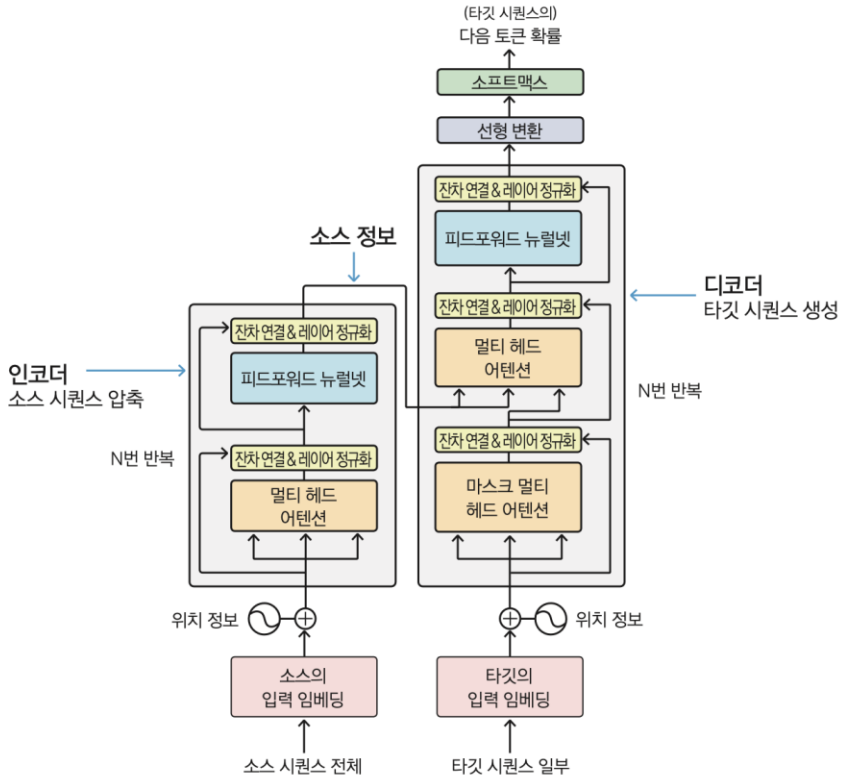
# 인코더와 디코더

- Sequence-to-Sequence 모델은 대개 인코더와 디코더 두 개의 파트로 구성

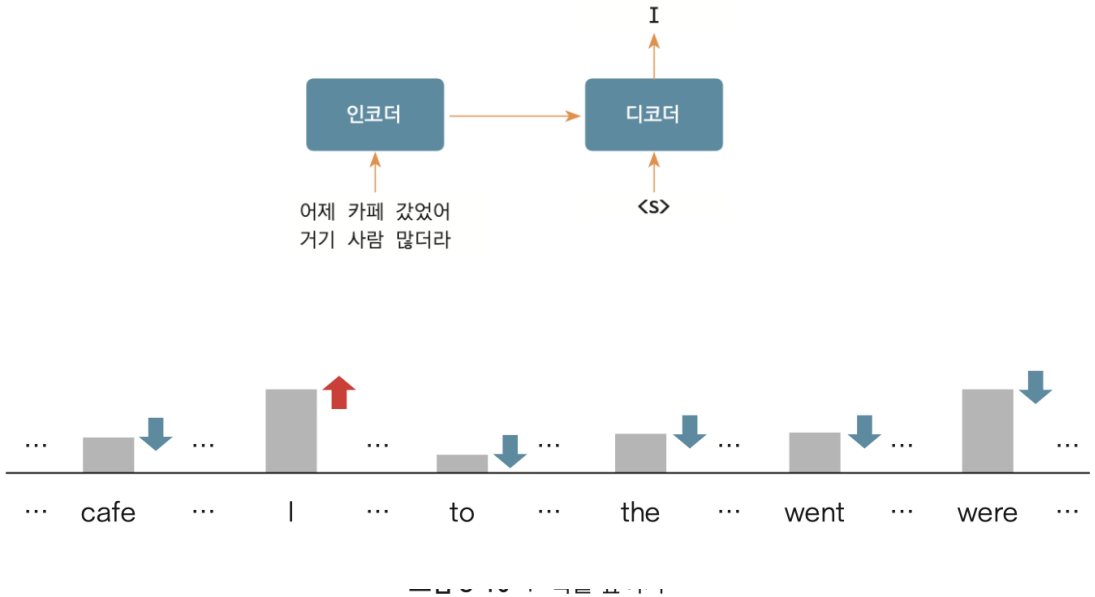

» 인코더 → 소스 시퀀스의 정보를 압축해서 디코더로 전달
» 디코더 → 인코더의 출력 데이터를 기반으로 타깃 시퀀스 생성

# 트랜스포머

- 트랜스포머 구조




- 인코더의 입력은 소스 시퀀스
- 디코더의 입력은 타깃 시퀀스 일부
- 트랜스포머 학습은 입력이 주어졌을 때 정답에 해당하는 단어의 확률 값을 높이는 방식으로 수행



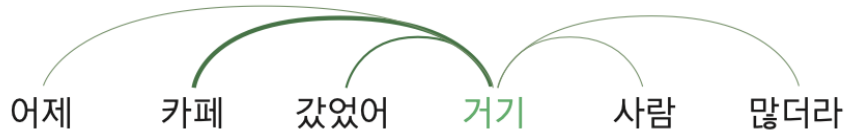
# 셀프 어텐션

- 문맥 정보 학습 문제
  - » 합성곱 신경망은 필터를 사용해 시퀀스의 지역적인 특징 학습은 가능하지만 필터 크기를 넘어서는 문맥 학습 어려움
  - » 순환 신경망은 시퀀스가 길어질수록 오래전에 입력된 단어 정보가 사라지거나 특정 단어 정보를 과도하게 반영하는 문제 발생

- 어텐션
  - » 시퀀스 입력에 수행하는 기계학습 방법의 일종
  - » 디코더에서 출력 단어를 예측하는 매 시점마다 인코더에서의 전체 입력 문장을 다시 한 번 참고하면서 학습
  - » 이 때 전체 입력 문장을 동일한 비율로 참고하지 않고 해당 시점에 예측해야 할 단어와 연관성이 높은 단어에 더 집중

# 셀프 어텐션

- 셀프 어텐션
  - » 자신에게 수행하는 어텐션 기법
  - » 입력 시퀀스 가운데 태스크 수행에 의미 있는 요소들 위주로 정보 추출
  - » 입력 시퀀스 각 요소가 다른 요소와 어떻게 상호 작용하는지 모델링




  - » 수행 대상은 입력 시퀀스 전체
  - » 개별 단어와 전체 입력 시퀀스를 대상으로 어텐션 계산 수행 → 문맥 정보 학습

# 셀프 어텐션

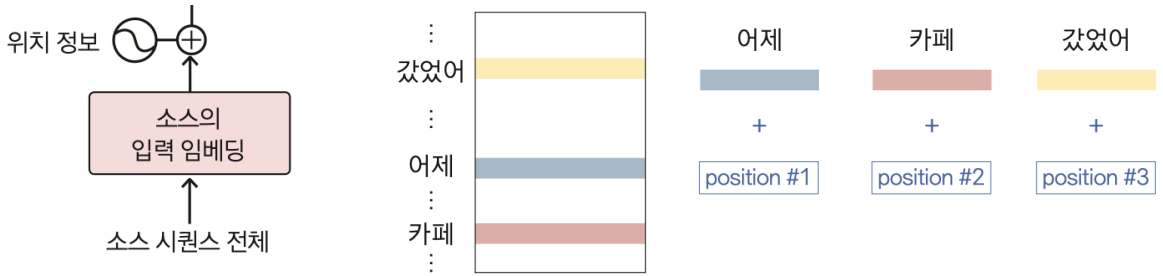
- 셀프 어텐션 계산
  - » Query, Key, Value 3가지 요소가 서로 영향을 주고 받는 구조
  - » 입력된 각 단어 벡터는 계산 과정을 거쳐 Query, Key, Value 로 변환
  - » Query와 Key의 행렬곱 연산으로 단어 사이의 관계를 합이 1인 확률 값 도출

  - » 위 연산 결과와 Value 벡터를 곱하고 가중합 계산

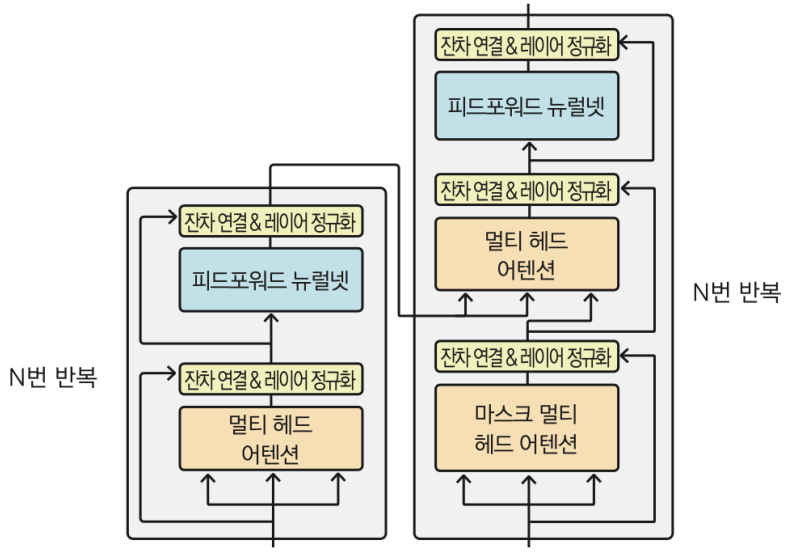
$$\mathbf{Z}_{\text{카페}} = 0.1\times\mathbf{V}_{\text{어제}} + 0.1\times\mathbf{V}_{\text{카페}} + 0.2\times\mathbf{V}_{\text{갔었어}} + 0.4\times\mathbf{V}_{\text{거기}} + 0.1\times\mathbf{V}_{\text{사람}} + 0.1\times\mathbf{V}_{\text{많더라}}$$

# 셀프 어텐션

- 입력층
  - » 인코더 입력은 소스 언어 문장의 토큰 인덱스 시퀀스
  - » 소스 시퀀스의 입력 임베딩에 위치 정보 추가

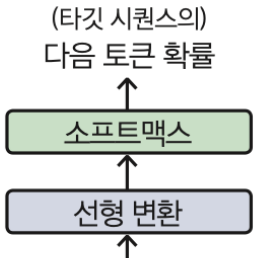



- 인코더 디코더 블록
  - » 최초 인코더/디코더 블록의 입력은
    - › 인코더/디코더 입력층에서 만들어진 벡터 시퀀스
  - » 출력 벡터 시퀀스가 두 번째 인코더/디코더 블록의 입력
  - » 다음 인코더/디코더 블록의 입력은 이전 블록의 출력
  - » N번 반복



# 셀프 어텐션

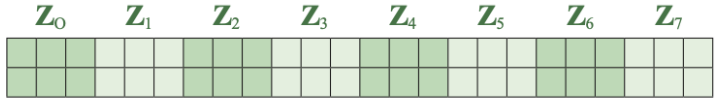
- 출력층
  - » 디코더 마지막 블록의 출력 벡터 시퀀스
  - » 출력은 타깃 언어의 어휘 수만큼 차원을 갖는 벡터
  - » 이 벡터는 디코더에 입력된 타깃 시퀀스의 다음 토큰에 대한 확률 분포



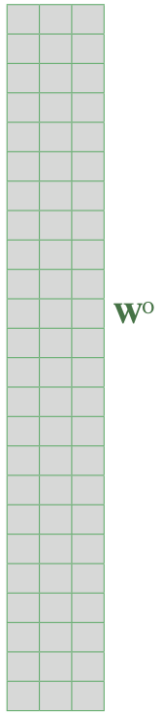
# 셀프 어텐션

- 멀티 헤드 어텐션
  - » 셀프 어텐션을 동시에 여러 번 수행
  - » 여러 헤드가 독자적으로 셀프 어텐션 계산
  - » 개별 헤드의 셀프 어텐션 수행 결과는 입력 단어 수 X Value 차원 수
  - » 멀티 헤드 어텐션의 최종 수행 결과는 입력 단어 수 X 목표 차원 수

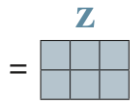
① 모든 헤드의 셀프 어텐션 출력 결과를 이어 붙인다.




② ①의 결과로 도출된 행렬에 $W^O$를 곱한다. 이 행렬은 개별 헤드의 셀프 어텐션 관련 다른 행렬($W_Q$, $W_K$, $W_V$)과 마찬가지로 태스크(기계 번역)를 가장 잘 수행하는 방향으로 업데이트된다.

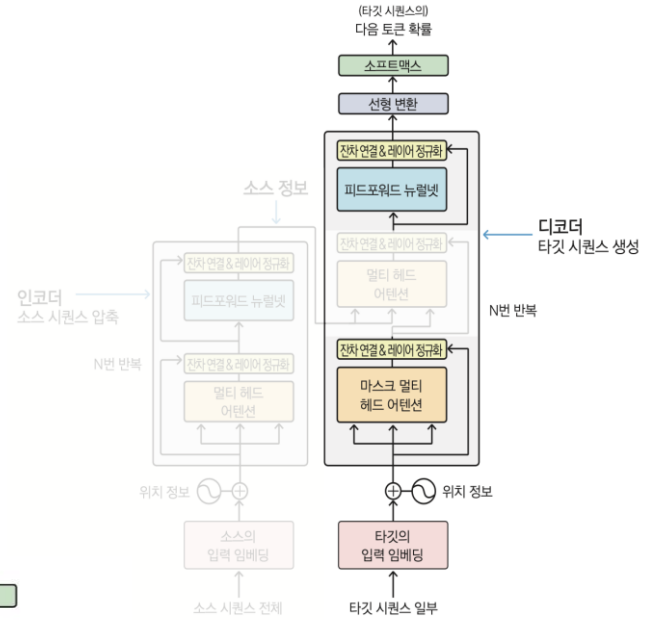


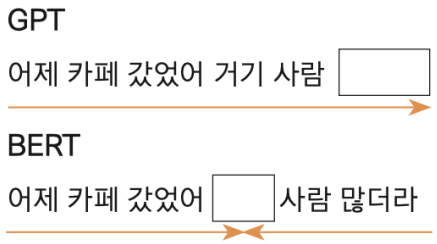

③ 새롭게 도출된 **Z** 행렬은 동일한 입력(문서)에 대해 각각의 헤드가 분석한 결과의 총합이다.

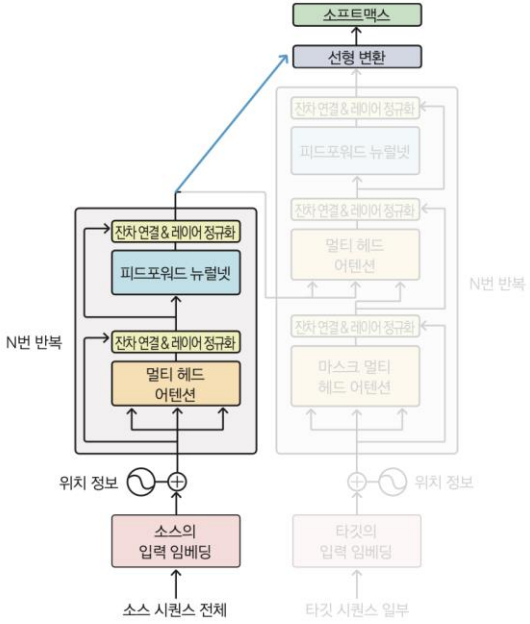


# BERT와 GPT

- GPT
  - » 문장 생성
  - » 다음 단어가 무엇인지 맞히는 과정
  - » 문장 왼쪽부터 오른쪽으로 순차적으로 계산 (단방향)
  - » 트랜스포머에서 인코더를 제외하고 디코더만 사용




- BERT
  - » 문장의 의미 추출
  - » 빈칸에 어떤 단어가 적절한지 맞히는 과정
  - » 빈칸 앞뒤 문맥을 모두 검토 (양방향)
  - » 트랜스포머에서 디코더를 제외하고 인코더만 사용