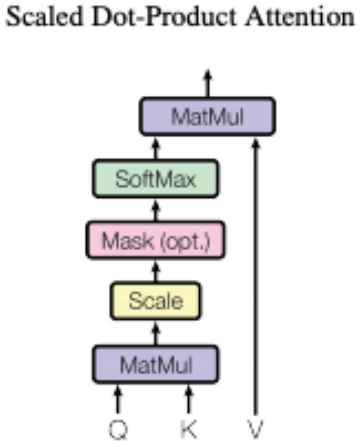
멀티-헤드 어텐션



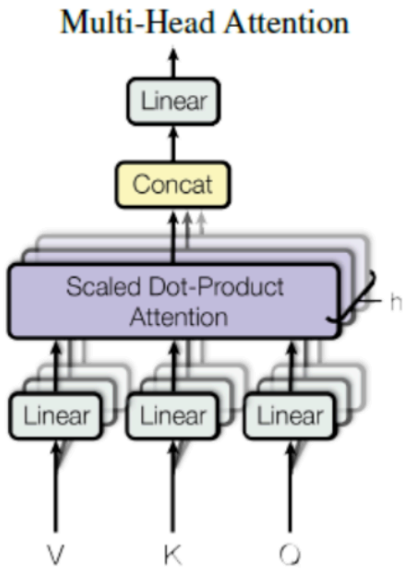
$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

위의 식에서 Q는 영향을 받을 단어 K는 영향을 주는단어 V는 영향에 대한 가중치

각 인자값은 모두 문장인데 각 단어가 벡터로 되어있고 이것들이 모여 행렬로 되어있는 구조
먼저 Q,K 를 행렬곱 연산을 한 후 1/sqrt(dk)로 Scaling을 함
스케일링은 소프트맥스 함수가 포화되어 그레디언트가 너무 작아지지않도록 해줌
소프트맥스를 거친 값을 value 에 곱해준다면, query와 유사한 value일수록 더 높은 값을 가지게 됨 -> **중요한 정보에 더 관심을 둔다!**

또 선택적으로 Mask를 하는 이유는, Decoder의 초반 Multi-Head Attention이 Masked Multi-Head Attention을 사용하기 때문
그 이유는 결론적으로 우리는 시계열 데이터를 정확히 예측하고자 하기 때문에 전체 정보를 미리 알려주는 것이 아니라 **순차적으로 데이터를 제공하며 유추하기를 원하기 때문에 *Mask*를 사용하여 순서가 되지 않은 데이터의 정보*(Attention)*를 숨기는 과정을 거침.**
이렇게 Mask를 사용하는 이유는 Input Data가 벡터로 전체 데이터가 한번에 들어가기 때문.

Scaled Dot-Product Attention은 각각의 단어가 다른 단어들과 얼마나 밀접한 관련이 있는지에 대한 정보까지 모두 담은 구조



멀티헤드 어텐션은 스케일드 닷 프로덕트 어텐션층의 묶음

각 층은 값 키 쿼리의 선형변환이 선행됨 (가중치와 편향을 계산하는 층)

각 헤드별로 논리적으로 분할됨
예를들어 played 라는 단어를 넣었을때 어떠헤드는 동사라는 정보를 넣고 어떤헤드는 과거형이라는것에 주목

책에는 멀티헤드 어텐션 클래스가 없다고 되어있지만 현재는 있음
https://www.tensorflow.org/api_docs/python/tf/keras/layers/MultiHeadAttention

BERT (bidirectional Encoder Representations from Transformers)
- 구글이 공개한 사전학습 모델

1. MLM(Masked Language Model)

- 문장에 있는 각 단어는 15%의 확률로 마스킹함. 모델은 마스킹한 단어를 예측하도록 훈련됨 (빈칸넣기같은거)

2. NSP(Next Sentence Prediction)

- NSP는 두 문장이 주어졌을 때 두 번째 문장이 첫 번째 문장의 바로 다음에 오는 문장인지 여부를 예측하는 방식