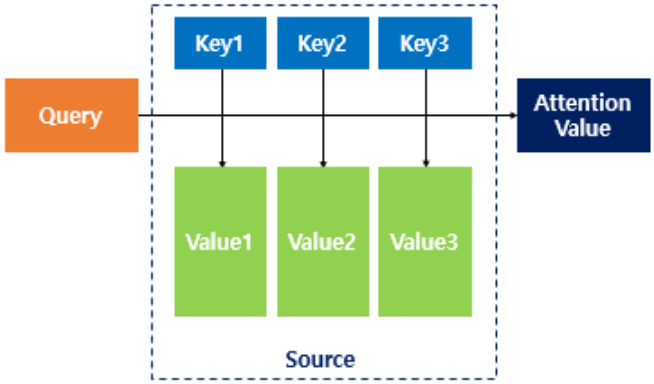
딥러닝 프레임워크 과제1

202184032 안신영
202104371 최정환

어텐션 구조의 아이디어:
seq2seq 모델은 기존의 RNN 모델과 같이 고정된 길이의 벡터를 입력으로 받아 고정된 길이의 벡터를 출력하는 구조를 가지고 있다. 그러나 이 구조는 고정된 길이의 벡터 압축으로 시퀀스의 길이가 길어질수록 정보의 손실이 발생할 수 있다. 이러한 한계를 보완하기 위해 도입이 된게 어텐션 메커니즘이다.

어텐션의 기본 아이디어는 디코더에서 출력 단어를 예측하는 매 시점(time step)마다, 인코더에서의 전체 입력 문장을 다시 한 번 참고한다는 점이다. 전체 입력 문장을 전부 다 동일한 비율로 참고하는 것이 아니라, 해당 시점에서 예측해야 할 단어와 연관이 있는 입력 단어 부분을 좀 더 집중하게 된다.

어텐션 함수:
어텐션 함수는 주어진 Query 에 대해서 모든 Key와의 유사도를 각각 구한다. 그리고 구해낸 이 유사도를 키와 맵핑되어있는 각각의 Value에 반영해준다. 그리고 유사도가 반영된 Value를 모두 더해서 리턴한다.

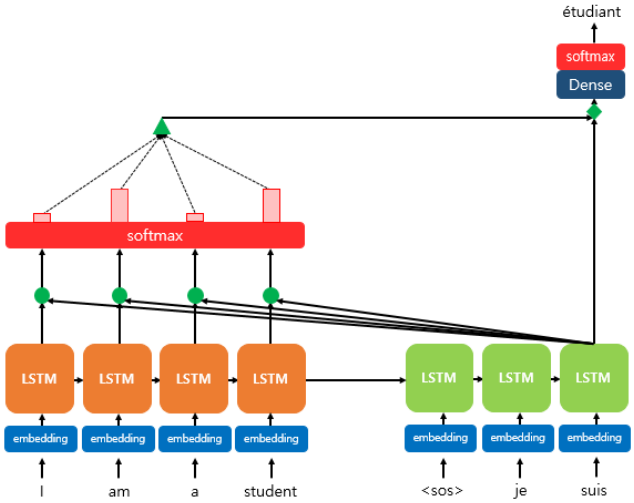


Attention(Q, K, V) = Attention Value
어텐션을 함수로 표현하면 다음과 같이 표현된다.
여기서 Q = Query : t 시점의 디코더 셀에서의 은닉 상태, K = Keys : 모든 시점의 인코더 셀의 은닉 상태들, V = Values : 모든 시점의 인코더 셀의 은닉 상태들을 의미한다.

어텐션 스코어:
'중요한 단어에 집중한다'는 의미는 어텐션 스코어를 계산한다는 것을 의미한다.
어텐션 스코어는 인공신경망 모델이 각 인코딩 타임스텝마다 계산된 feature를 가지고 자동으로 계산하는 0~1 사이의 값이다.
이는 1에 가까울수록 더 집중해서 봐야 하는 스텝, 0에 가까울수록 지금은 중요하지 않아 대충 봐도 되는 스텝을 의미한다.

이러한 어텐션 구조의 장점은 완전히 동일한 어순을 가지며 단어간 1:1 매칭이 되는 언어끼리의 번역이 아니라면 단어 집중 시 유연하게 대처해야하는데 어텐션 메커니즘이 그 역할을 훌륭히 해낸다.

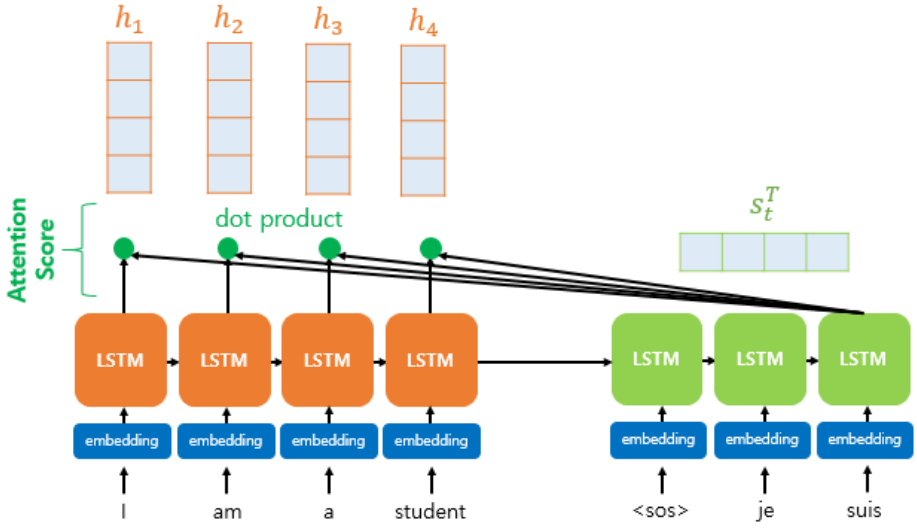
어텐션 구조의 예시:
어텐션 구조의 가장 기본이라고 할 수 있는 Dot-Product Attention의 예시가 있다.
먼저 아래의 그림을 보면,



이 그림은 Decoder의 세번째 LSTM Cell에서 출력 단어를 예측할 때 어텐션 메커니즘을 사용하는 예시이다.
여기에서는 Attention 메커니즘을 적용함으로써, 세번째 단어를 예측할 때 예측 단어와 encoder의 모든 시퀀스('I', 'am', 'a', 'stduent')의 관계를 파악하게 된다.

이 때, 파악하는 방식은 그림 내에도 존재하는 soft max를 이용함으로써 이루어진다.

그림의 왼쪽 부분처럼, 세번째 cell에서 출력 단어를 예측할 때 softmax를 통과시킨 input sequence를 추가적으로 전달함으로써 decoder가 새로운 단어를 예측하는 데 있어서 Recurrent하게 전달된 정보 외에도 input sequence의 정보를 참고할 수 있는 길을 마련하게 된다. 이에 대한 연산 과정은 다음과 같다.



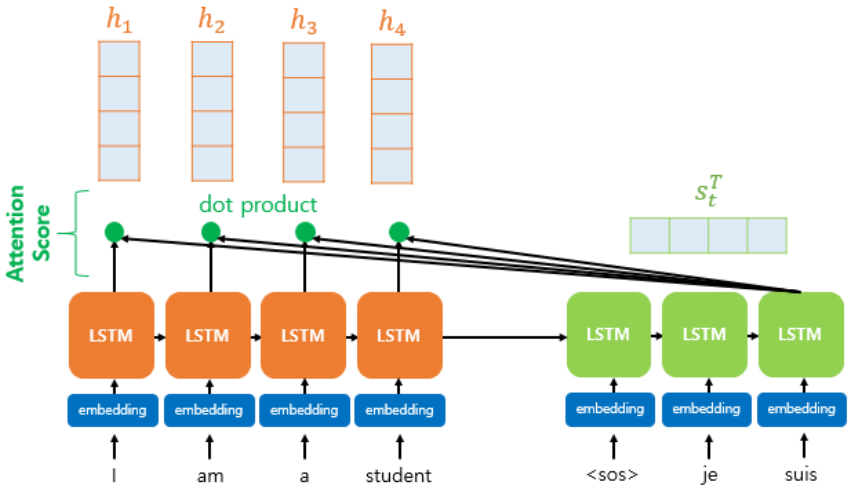
ht : t 시점에서 인코더의 hidden state
st : t시점에서 decoder의 hidden state

Attention value at를 구하기 위해서는 크게 아래의 세 과정을 거쳐 얻을 수 있다.
-> ht, st를 활용해 Attention Score(et)를 구한다.
-> softmax를 활용해 Attention Distribution을 구한다.
-> 인코더에서, 2.에서 구한 분포를 토대로 결정한 가중치와 hidden state를 가중합하여 Attention Value를 구한다.

이 과정을 하나하나 먼저 설명해보면

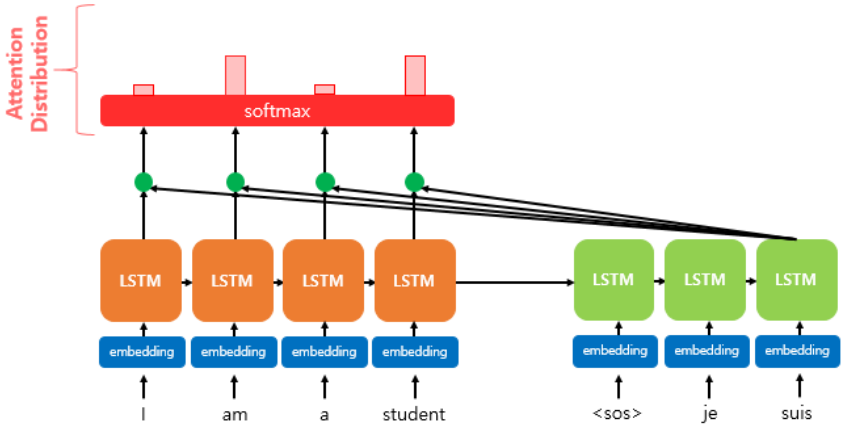
1. ht, st를 활용해 Attention Score(et)를 구한다.
Attention Score(어텐션 스코어)는 디코더의 time step t에서 새로운 단어를 예측하기 위해, decoder의 hidden state st와 encoder의 hidden states h1~hn들이 얼마나 유사한지를 판단하는 점수이다.

아래 그림은 decoder의 time step t에서 hidden state st와 encoder의 time step i에서 hidden state hi의 어텐션 스코어를 구하는 과정을 나타낸다.



위의 그림에 따르면, st와 hi의 Attention Score는 아래와 같은 식으로 결정된다.
score(st,hi)=st^T*hi
이 때 결과 값은 scalar가 된다.
decoder의 time step은 t인 반면, 참고하는 encoder의 time step은 1부터 N까지 있기 때문에, encoder의 모든 은닉 상태에 대한 decoder의 time step t에서의 Attention score를 계산하면 아래와 같이 나타낼 수 있다.
e_t=[s_t^{T}h_1, ..., s_T^{T}h_N]

2. softmax를 활용해 Attention Distribution을 구한다.
위에서 얻은 Attention scores et에 softmax함수를 적용해, 모든 값의 합이 1이 되는 확률 분포 Attention Distribution을 얻는다.

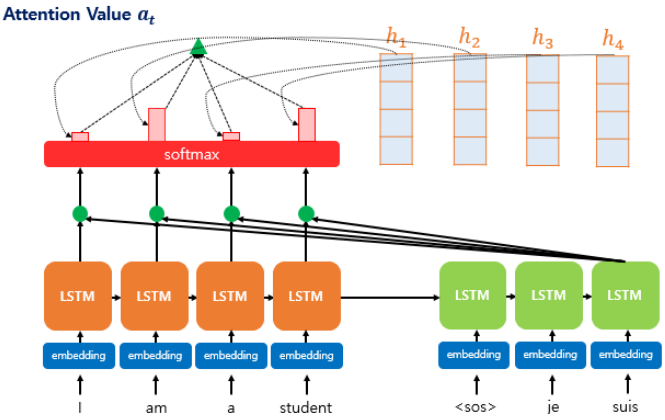


즉, 위의 그림에 있는 Attention Distribution을 얻기 위해 아래와 같은 식을 사용하면 된다.
$\alpha_t = softmax(e_t)$
예를 들어, 입력 시퀀스인 ['I', 'am', 'a', 'student']에 대응하는 hidden states를 활용해 Attention scores를 구하고, 이로부터 Attention Distribution을 구하게 되면 [0.1,0.4,0.1,0.4]의 형태를 갖는 벡터를 얻게 된다.

이 때 각각의 값을 Attention Weight(어텐션 가중치)라고 한다.

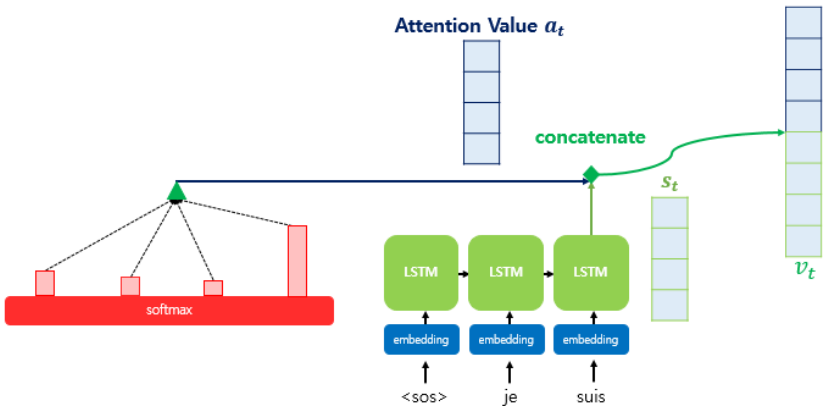
3. 인코더의 각 Attention Weight와 그에 대응하는 hidden state를 가중합하여 Attention Values를 구한다.

최종적으로, 위에서 구한 Attention Weight와 각 hidden state를 통해 최종적인 Attention value at를 얻는다.



이에 대한 식은 아래와 같다.
$a_t = \sum_{i=1}^{N} \alpha_i h_i$
이러한 어텐션 값 at는 인코더의 맥락을 포함하고 있기 때문에 Context Vector(맥락 벡터)라고도 불린다.

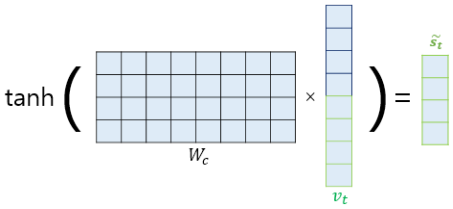
4. Attention value at와 decoder의 t 시점의 hidden state를 연결한다.



위에서 구한 Attention value at를 단순하게 decoder의 t 시점의 hidden state st와 연결해 준다.

연결한 벡터를 vt라고 가정하면, vt는 기존에 얻은 decoder의 hidden state의 정보 외에도 encoder에서의 모든 hidden state를 고려한 정보 또한 포함하고 있기때문에, sequence가 길어지더라도 정보를 크게 잃지 않는다.

5. 출력층 연산의 input이 되는$\tilde{s}_t$를 계산한다.



위 연산에 대한 식은 아래와 같이 간단하게 나타낼 수 있다.
$\tilde{s}_t = \tanh(W_c[a_t; s_t] + b_c)$

위에서 Wc는 학습 가능한 가중치 행렬이고, bc는 편향이다. 위에서 말한 vt가 vt=[at;st]의 형태로 들어간다. ;는 concat을 나타낸다

그리고 아래의 식을 통해 최종적인 예측을 얻는다.
$\hat{y}_t = Softmax(W_y \tilde{s}_t + b_y)$

이 외에도 다양한 어텐션을 활용한 모델이 많다.

결국 어텐션 방법은 seq2seq의 단점을 해결하기 위해 Attention 기법을 '추가적으로' 사용한 예시를 나타냈다. 하지만 최근에 들어서는 어센션 매커니즘 자체가 seq2seq를 대체하는 방법으로 사용이 되고 있다.