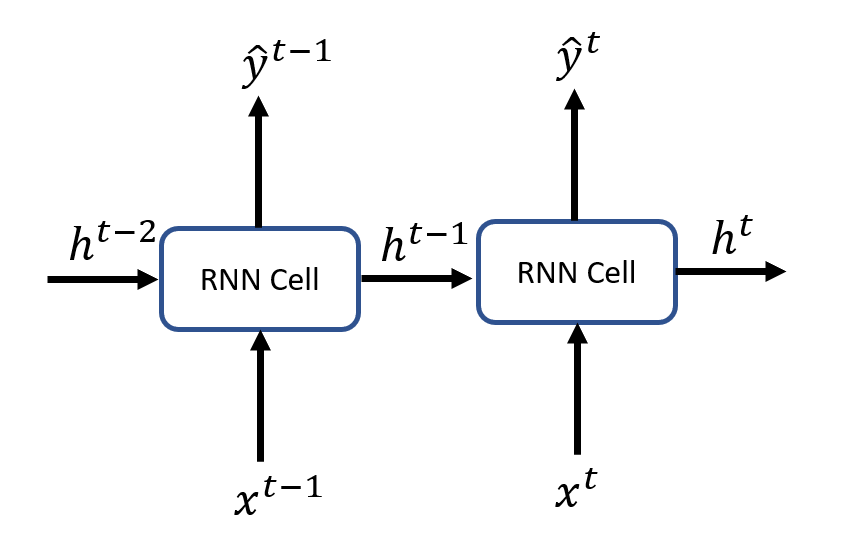
15장 RNN과 CNN을 사용해 시퀀스 사용하기

순환 뉴런과 순환 층



메모리 셀: 타임 스텝 t에서 순환 뉴런의 출력은 이전 타임 스텝의 모든 입력에 대한 함수이므로 이를 일종의 메모리 형태라고 말할 수 있다. 타임 스텝에 걸쳐 어떤 상태를 보존하는 신경망의 구성요소를 메모리 셀이라고 한다.

시퀀스 투 시퀀스: 예를 들면 최근 N일치의 주식가격을 입력하면 N-1일 전부터 내일까지의 주식가격을 출력한다

시퀀스 투 벡터: 입력 시퀀스를 네트워크에 주입하고 마지막을 제외한 모든 출력을 무시한다.

벡터 투 시퀀스: 각 타임 스텝에서 하나의 입력 벡터를 반복해서 네트워크에 주입하고 하나의 시퀀스를 출력한다.

인코더-디코더: 시퀀스 투 벡터를 지나 벡터 투 시퀀스로 이어지는 신경망

트렌드와 계절성

가중 이동 평균이나 자동 회귀 누적 이동평균과 같은 시계열을 예측하는 다른 방법이 많다. 이런 방법 중 일부는 트렌드나 계절성을 제거한 후 모델을 훈련한다. 그리고 예측을 만들기 시작할 때 최종 예측에 이 트렌드를 다시 더합니다. RNN은 일반적으로 이런 작업은 필요없지만 어떤 경우 모델이 트렌드나 계절성을 학습할 필요가 없을 경우 성능 향상에 도움이 될 수 있다.

긴 시퀀스 다루기

1. 불안정한 그레디언트 문제와 싸우기

좋은 가중치 초기화, 빠른 옵티마이저, 드롭 아웃은 불안정한 그레디언트 문제를 완화시킨다. 하지만 ReLU와 같은 수렴하지 않는 활성화 함수나 배치 정규화 등은 도움이 되지 않는다. 오히려 RNN을 더욱 불안정하게 할 수 있다.

층 정규화(layer normalization)

RNN과 잘 맞는 다른 종류의 정규화이다. 배치 정규화와 비슷하지만 배치 차원에 대해 정규화 하는 대신에 특성 차원에 대해 정규화한다. 이를 구현하기 위해서는 사용자 정의 메모리 셀을 정의해야한다.

1. 단기 기억문제 해결하기

LSTM(Long short term memory): RNN의 특별한 한 종류로 긴 의존기간을 필요로하는 학습을 수행할 능력을 갖고 있다. 아래 링크는 LSTM에 대한 자세한 설명이다.

<https://dgkim5360.tistory.com/entry/understanding-long-short-term-memory-lstm-kr>

GPU(gated recurrent unit): LSTM 셀의 간소화된 버전이고 유사하게 작동하는 것으로 보인다.

* 두 상태벡터를 하나의 벡터로 합쳤다.
* 하나의 게이트 제어기가 삭제 게이트와 입력 게이트를 모두 제어한다.
* 출력 게이트가 없다. 즉, 전체 상태 벡터가 매 타임 스텝마다 출력된다.

LSTM과 GPU는 RNN 성공의 주역이다. 다만, 이 셀들은 단순한 RNN 셀보다 훨씬 긴 시퀀스를 다룰 수 있지만 매우 제한적인 단기 기억을 가진다. 이러한 문제를 해결하는 한 가지 방법은 1D 합성곱 층을 사용해 입력 시퀀스를 짧게 줄이는 것이다.

16장 RNN과 어텐션을 사용한 자연어 처리

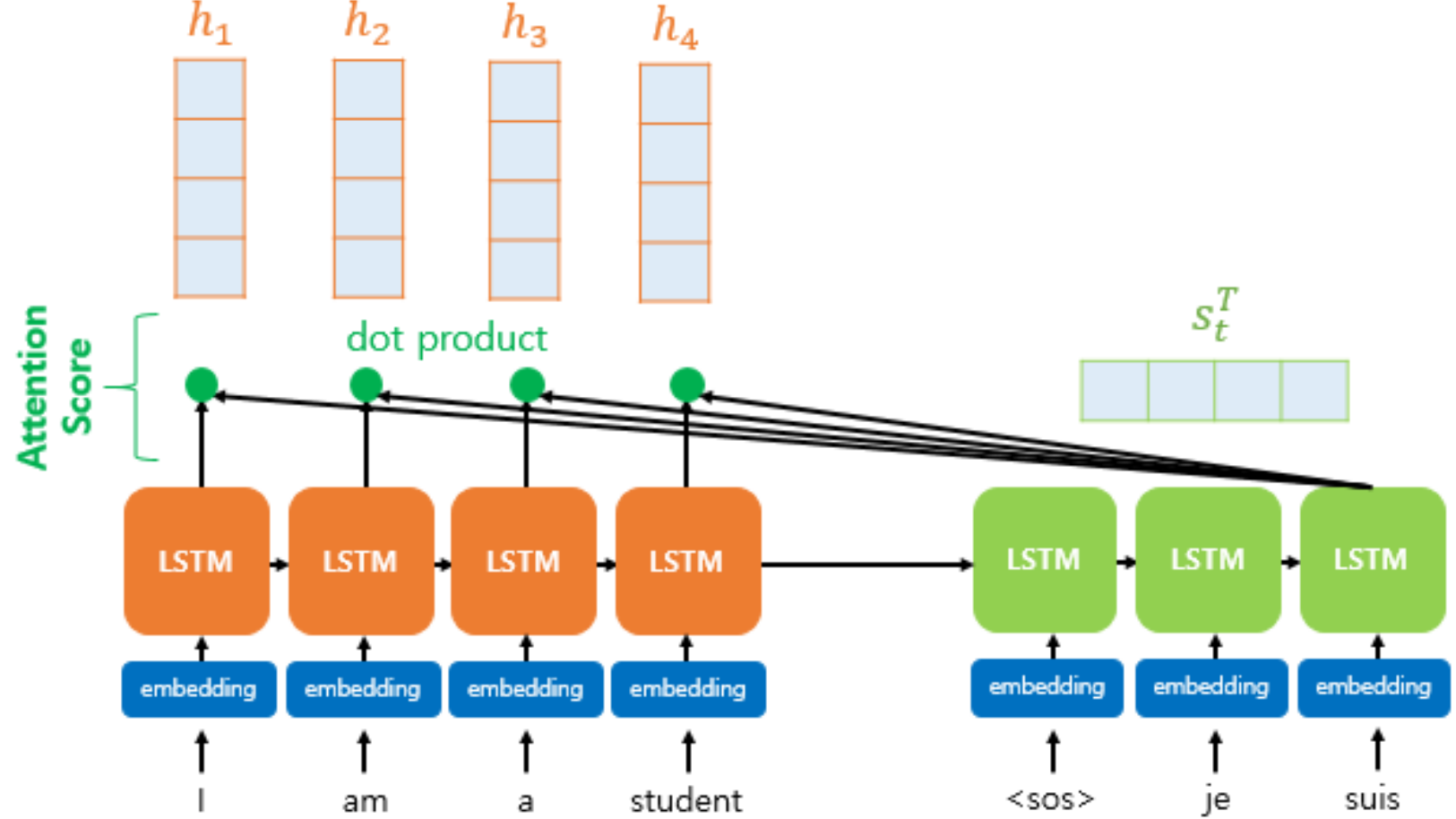
자연어 처리에서 RNN에 기반한 seq2seq모델에는 크게 두개의 문제가 있다.

* 하나의 고정된 크기의 벡터에 모든 정보를 압축하려고 하니까 정보 손실이 발생한다.
* RNN의 고질적인 문제인 기울기 소실 문제가 존재한다.

어텐션: 디코더에서 출력 단어를 예측하는 매 시점(time step)마다 인코더에서의 전체 입력 문장을 다시 한번 참고한다. 단, 전체 입력 문장을 동일한 비율로 참고하는 것이 아니라 해당 시점에서 예측해야할 단어와 연관있는 부분을 좀 더 집중적으로 참고한다.

닷-프로덕트 어텐션(Dot-product Attention)

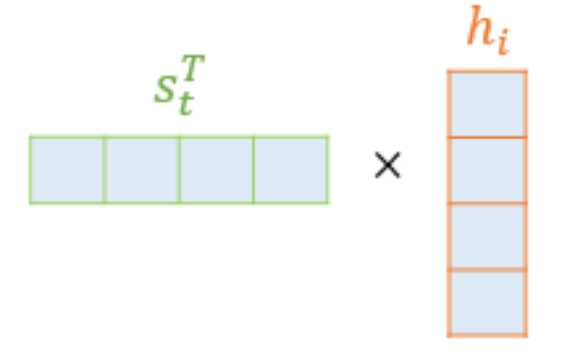
간단한 형태의 어텐션이다. 다른 어텐션과의 차이는 중간 수식의 차이로 메커니즘은 유사하다. 이를 통해 어텐션을 이해해보자.



해당 이미지에서는 세번째 LSTM 셀이 출력단어를 예측 중입니다. 이때 보통의 일반적인 모델이라면 t-1에서의 은닉 상태와 t-1에서의 출력 단어입니다. 하지만 어텐션 모델은 t 시점에서의 attention value가 추가로 요구됩니다.

Attention value 구하기

먼저 어텐션 스코어를 구합니다. 어텐션 스코어는 인코더의 모든 은닉 상태 각각이 디코더의 현 시점의 은닉 상태와 얼마나 유사한지를 판단하는 스코어값입니다.



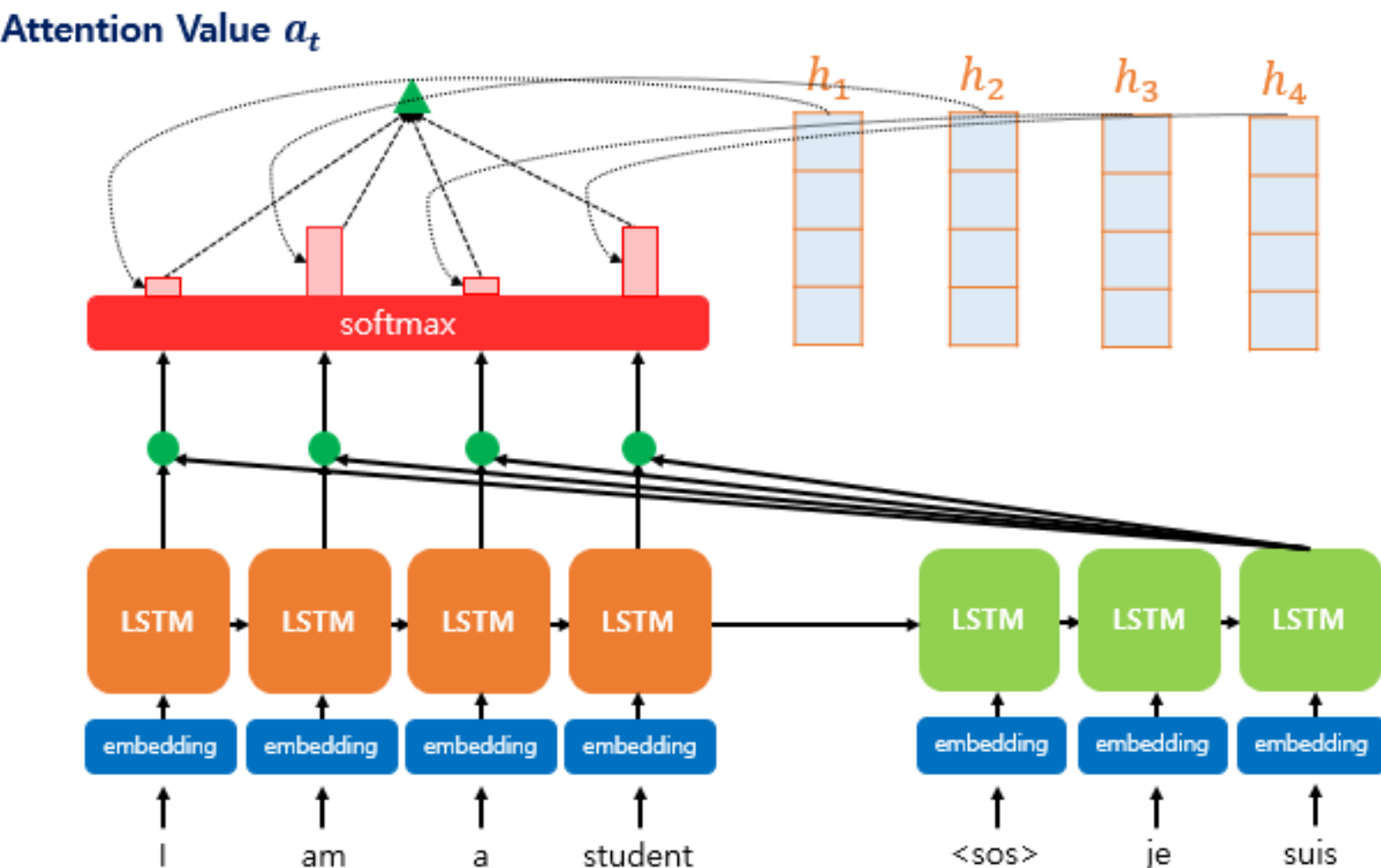
어텐션 스코어는 다음과 같은 벡터의 내적으로 구할 수 있습니다. 이를 함수로 정의하면 다음과 같습니다.

=

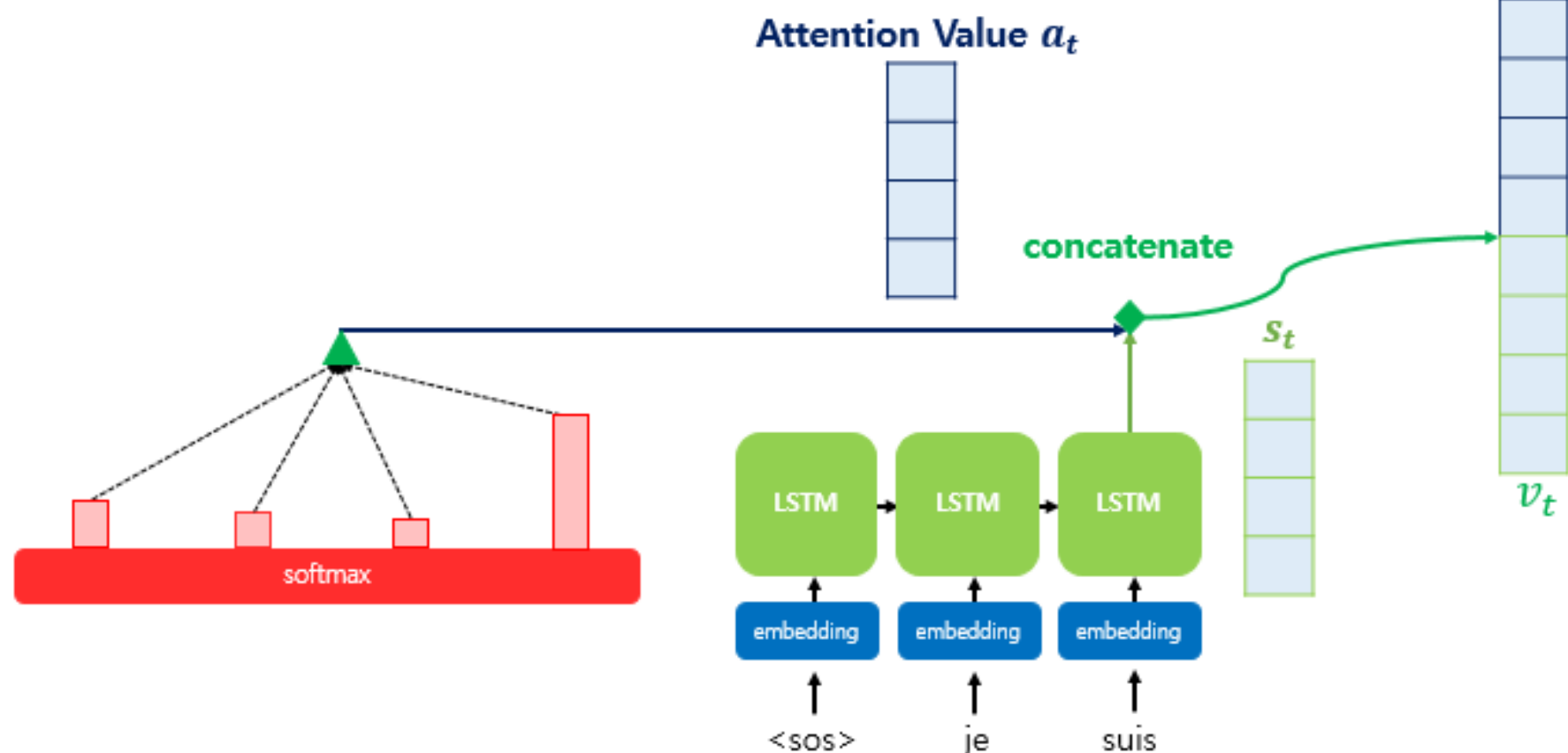
그리고 이러한 어텐션 스코어의 모음값을

= []

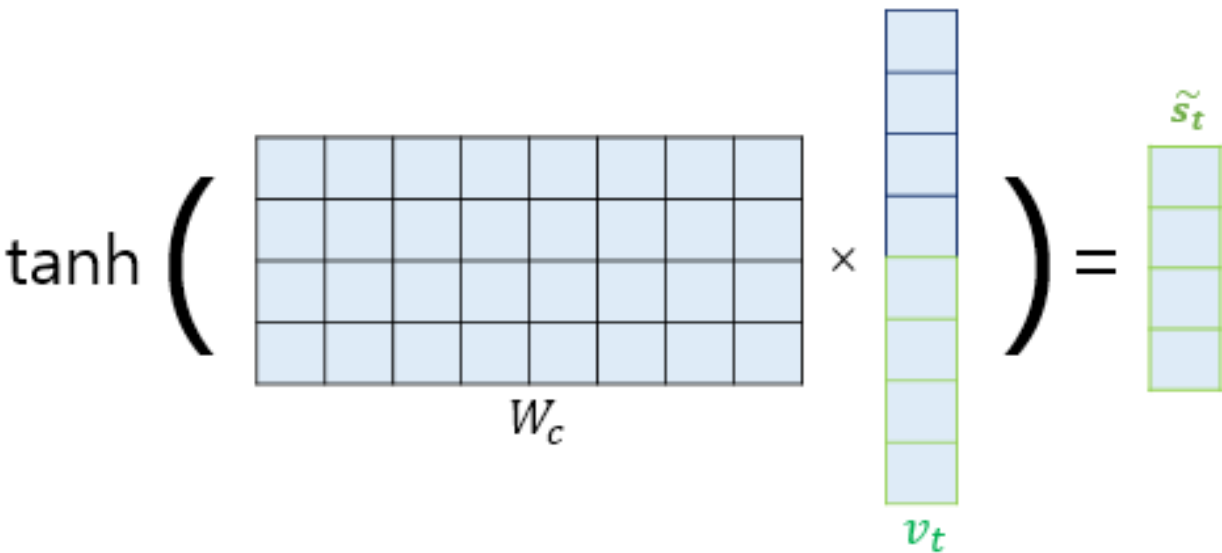
라고 정의하겠습니다. 에 소프트맥스 함수를 적용하여 얻은 분포를 어텐션 분포라고 합니다. 분포의 각각의 값은 어텐션 가중치라고 합니다.



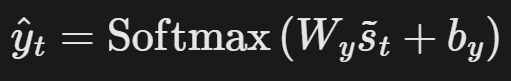
위의 그래프에서 어텐션 가중치를 직사각형의 크기로 표현한 것입니다. 이제 준비한 정보들을 가중합을 통하여 합칩니다. 가중합을 통해 얻은 벡터를 Attention value라고 합니다.

’

Attention value를 t에서 디코더의 은닉 벡터에 결합하여 하나의 벡터를 만듭니다. 이 벡터를 t에서 y의 예측을 위한 입력으로 이용합니다.



이 벡터를 가중치 행렬과 곱하여 새로운 은닉 벡터를 얻습니다. 어텐션을 사용하지 않는 seq2seq에서는 출력층의 입력이 인 반면 어텐션 메커니즘에서는 가 됩니다. 이를 이용해 아래의 예측 벡터를 얻습니다.



이외에도 다양한 어텐션 모델이 존재합니다. 메커니즘은 위에서 설명한 닷-프로덕트 어텐션과 유사하며 차이점은 어텐션 스코어를 구하는 방법입니다.

<https://wikidocs.net/22893>

: 해당 글은 위 링크의 글을 읽으며 정리한 것입니다.

17장 오토인코더와 GAN을 사용한 표현 학습과 생성적 학습

과소완전 오토인코더: 특성 학습을 강제하기 위해 코딩 층의 크기를 제한한 오토인코더. 밀집 오토인코더 외에도 합성곱 오토인코더나 순환 오토인코더 등 다양한 형태 존재

과대완전 오토인코더

잡음 제거 오토인코더: 오토인코더가 유용한 특성을 학습하도록 강제하는 다른 방법은 입력에 잡음을 추가하고 잡음이 없는 원본 입력을 복원하도록 훈련하는 것입니다. 잡음은 가우시안 잡음을 추가하거나 드롭아웃 층을 추가하는 방법을 생성합니다

희소 오토인코더: 비용함수에 적절한 항을 추가하여 오토인코더가 코딩 층에서 활성화되는 뉴런수를 감소시키도록 만듭니다. 간단한 방법은 코딩 층에 시그모이드 활성화 함수를 사용하고 큰 코딩층을 사용한 후 활성화 값에 L1 규제를 추가하는 것입니다. 다른 방법으로는 훈련 반복마다 실제 희소 정도를 측정하여 목표하는 희소정도보다 클 경우 페널티를 줄 수 있도록 비용함수에 희소 손실을 추가하는 방법도 있습니다.

변이형 오토인코더:

* 확률적 오토인코더입니다. 즉, 훈련이 끝난 뒤에도 부분적으로 출력이 우연에 의해 결정됩니다.
* 생성 오토인코더입니다. 마치 훈련 세트에서 샘플링된 것 같은 새로운 샘플을 생성할 수 있습니다.

인코더는 평균 코딩과 표준편차를 만듭니다. 실제 코딩은 인코더가 만든 평균과 표준편차를 갖는 가우시안 분포에서 랜덤하게 샘플링됩니다. 따라서 입력이 매우 복잡하더라도 변이형 오토인코더는 가우시안 분포에서 샘플링 된것처럼 보이는 코딩을 만드는 경향이 있습니다. 또한, 비용함수가 코딩을 가우시안 샘플들의 군집처럼 보이도록 코딩공간 안에서 점진적으로 이동시켜 훈련이 끝난 뒤에 새로운 샘플을 매우 쉽게 생성할 수 있습니다.