BASE Week2

진행 상태	완료

25기 분석 인태영

1. Beam Search 동작 방식에 대해서 설명

Seq2Seq 모델에서는 어떤 입력 시퀀스 X에 대해 적절한 출력 시퀀스 Y를 예측한다.

$$\hat{Y} = \arg\max_{Y} P(Y \mid X)$$

즉 가장 **확률**이 높을 때의 Y값을 가장 적절한 출력 시퀀스로 본다는 것이다.

이 때 확률 값은 아래와 같이 표현될 수 있다.

$$P(Y \mid X) = \prod_{t=1}^{T} P(y_t \mid y_{<t}, X)$$

특정 시점 t의 출력은 결국 입력과, 1 ~ t-1 시점까지의 시퀀스에 의존하게 된다는 뜻이다.
하지만 위와 같이 계산할 경우 0~1까지의 값이 계속해서 곱해지게 되므로 시퀀스가 길어지게 되면 값이 매우 작아지는 문제가 발생한다.]

그렇기 때문에 log를 이용해서 곱셈을 덧셈의 형태로 바꿔준다.

$$\log P(Y \mid X) = \sum_{t=1}^{T} \log P(y_t \mid y_{<t}, X)$$

Beam Search

위와 같은 방식으로 확률을 계산하면 높은 정확도의 출력 시퀀스를 얻을 수 있겠지만 그 수많은 경우의 수를 고려하여 모든 확률을 구하기는 현실적으로 힘들다.

Beam Search는 단어를 디코딩 과정에서, 각 시점마다 가장 가능성 높은 시퀀스들을 일정 개수만 유지하면서 확장하는 전략이다. 이때 유지하는 시퀀스의 개수를 beam width라고 한다.

처음에는 <SOS> 토큰으로 시작한다. LSTM 디코더는 인코더에서 전달받은 cell state, hidden의 컨텍스트 벡터와 입력을 바탕으로 다음 단어의 확률 분포를 예측한다. 이때 beam width(k)만큼 상위 단어를 선택해 각각 후보 시퀀스를 만들게 된다.

그 다음 시점에서는 이전 시점에서 유지된 k개의 시퀀스 각각에 대해 가능한 모든 단어를 붙여 새로운 후보 시퀀스를 생성하고, 이 중에서 다시 상위 k개를 선택하여 후보 시퀀스를 만들게 된다.

이러한 연산을 반복하여 결론적으로 가장 높은 확률을 가진 출력 시퀀스를 얻게 된다.

2. Seq2Seq의 한계, Attention 매커니즘

Seq2Seq 모델의 인코더는 입력 전체를 하나의 **고정 길이 벡터(context vector)**로 압축한다. 이 벡터 하나로 출력 시퀀스 전체를 생성해야 하는데, 입력 시퀀스의 길이가 길어질수록, 정보를 압축해서 담기가 어려워져 정보 손실 문제가 발생한다.

Attetion 매커니즘을 이용하여 이러한 문제를 해결할 수 있다.

우선 인코더는 T개의 시점을 가진 입력 시퀀스를 처리하여 각 시점의 hidden state h_i를 생성한다.

디코더는 현재 시점 t의 hiddens state s_t를 가지고 있는데, s_t, h_i간의 유사도를 각각 계산한다.

$$e_{t,i} = \text{score}(s_t, h_i)$$

바로 이 Attention score를 이용하여 디코더가 어느 시점의 입력에 대해서 가장 주목하는 지 softmax를 이용해 정규화하여 Attention 가중치를 계산한다.

$$\alpha_{t,i} = \frac{\exp(e_{t,i})}{\sum_{j=1}^{T} \exp(e_{t,j})}$$

Attention 가중치를 이용하여 다음과 같이 context vector를 만들 수 있다.

$$c_t = \sum_{i}^{T} \alpha_{t,i} h_i$$

이 context vector를 디코더의 출력에 반영한다.

이러한 방식은 디코더가 출력 단어를 생성할 때 입력 전체에서 중요한 정보를 **선택적으로 참고**할 수 있도록 하여 긴 시퀀스에서도 성능 저하 없이 보다 정밀하고 의미 있는 출력을 생성하게 한다.