XLNet

전자공학과 220210031 유현우

1. Permutation language modeling
   A. Auto Regressive model (AR)
      i. 데이터를 순차적으로 처리하는 기법을 사용하는 모델
   B. AutoEncoding (AE)
      i. 입력값을 gt로 사용해서 학습하여 입력된 데이터를 복원하는 기법을 사용
      ii. 대표적으로 BERT가 있음
   C. Denoising autoencoder
      i. 노이즈(마스크)가 포함된 데이터를 인풋으로 받고 노이즈가 없는 데이터를 출력하는 모델
   D. 이때 AR은 문장을 bidirectional 하게 볼 수 없다는 한계가 있음
      i. 맞춰야 할 단어 이후 토큰 정보를 미리 고려할 수 없기 때문
   E. 또한 BERT라는 AE모델은 마스크 처리된 토큰들이 서로 독립이라고 가정하여 마스크 토큰들간의 관계를 따질 수 없다는 한계가 있음
   F. Permutation language model
      i. Permutation은 input된 token들을 랜덤하게 섞은 뒤 auto regressive하는 방법
      ii. 이를 통해 마스킹 없이 역방향 및 순방향에 대한 관계를 고려하여 언어 데이터를 학습할 수 있음
      iii. 또한 pretrain 수행 시 마스크를 하지 않으므로 fine-tuning 데이터와의 불일치 문제도 개선할 수 있음
      iv. Sequence x(T개의 token)일 때, T!개 different orders에 대해 autoregressive 수행
      v. Proposed permutation language modeling objective function
      1. $$\max_{\theta} \quad \mathbb{E}_{\mathbf{z}\sim\mathcal{Z}_T}\left[\sum_{t=1}^{T}\log p_\theta(x_{z_t} \mid \mathbf{x}_{\mathbf{z}<t})\right].$$

A. Z_T : sequence 길이가 T일 때 가능한 모든 permutations

B. z_t : t번 째 element

C. $z_{<t}$: t-1 elements of a permutation $z \in Z_T$

vi. 이를 통해 original sequence 순서는 유지하면서 attention mask를 통해 permutation 고려

- [3,2,4,1]

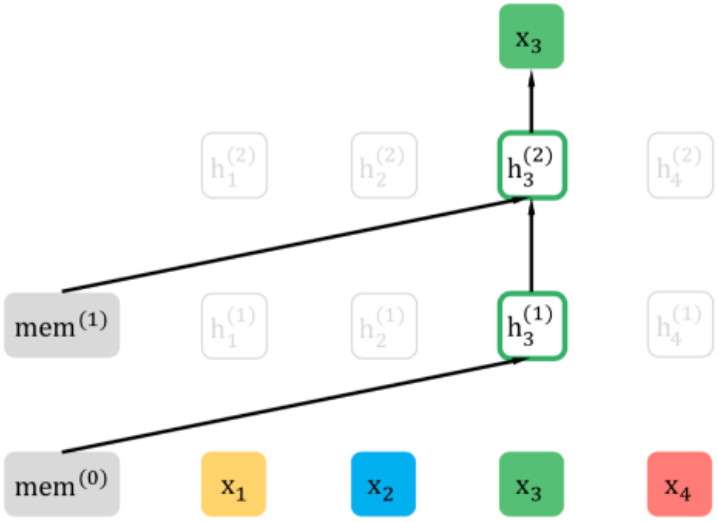

그림 1. 퍼뮤테이션 언어모델 학습

- [2,4,3,1]

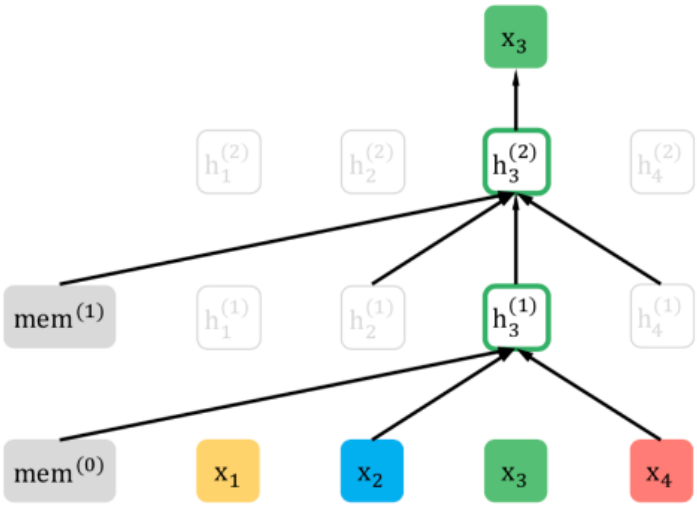

그림 2. 퍼뮤테이션 언어모델 학습2

vii. 보다 구체적으로 token 4개짜리 문장을 랜덤으로 뒤섞은 결과가 그림 7처럼 [3,2,4,1]

이고 셔플된 시퀀스의 첫번째 단어(3번 토큰)을 추론해야 한다고 가정

1. 이때 3번 token 정보를 넣어주면 너무 쉬워져서 안됨
2. 근데 2, 4, 1은 3번 뒤에 나오는 token들이라 입력에서 제외됨
3. 따라서 memory token만 입력됨

viii. 만약 문장을 한번 더 랜덤 셔플햇을 때 [2,4,3,1]이고 이번에도 3번을 예측해야 한다고 가정

1. 그러면 3번 token을 제외한 이전 token 2,4번이 입력되게 됨

2. Two-Stream Self-Attention

A. 이는 query stream과 content stream 두 가지를 혼합한 self-attention 기법

i. Content stream

1. 기존 transformer network와 매우 유사
2. $$\mathbf{h}_{z_t}^{(m)} \leftarrow \text{Attention}\left(\mathbf{Q}=\mathbf{h}_{z_t}^{(m-1)}, \mathbf{KV}=\mathbf{h}_{z_{\leq t}}^{(m-1)}; \theta\right)$$
   A. 이때 content stream vector를 h로 정의
   B. 그리고 z는 원래 문장 순서를 랜덤 셔플한 index list
   C. z_t는 z의 t번째 element
   D. $h_{z_t}^{(m)}$은 m번째 block의 z_t에 해당하는 content stream vector

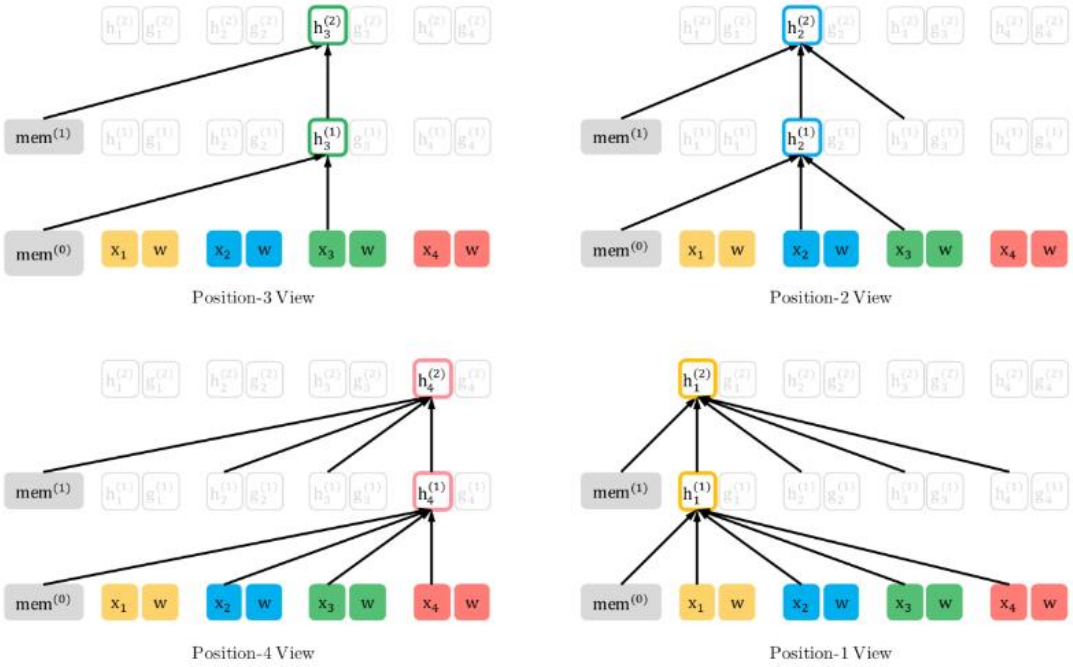

그림 3. Content stream

E. 위의 수식을 그림 3을 통해서 이해해 볼 수 있음

i. Z의 t번째 요소에 해당하는 content stream을 만들 때는 이전 문맥과 자기 자신에 대응하는 token 정보를 활용한다는 의미

ii. Query stream

1. $$\mathbf{g}_{z_t}^{(m)} \leftarrow \text{Attention}\left(\mathbf{Q}=\mathbf{g}_{z_t}^{(m-1)}, \mathbf{KV}=\mathbf{h}_{z<t}^{(m-1)}; \theta\right)$$

A. g가 query stream token

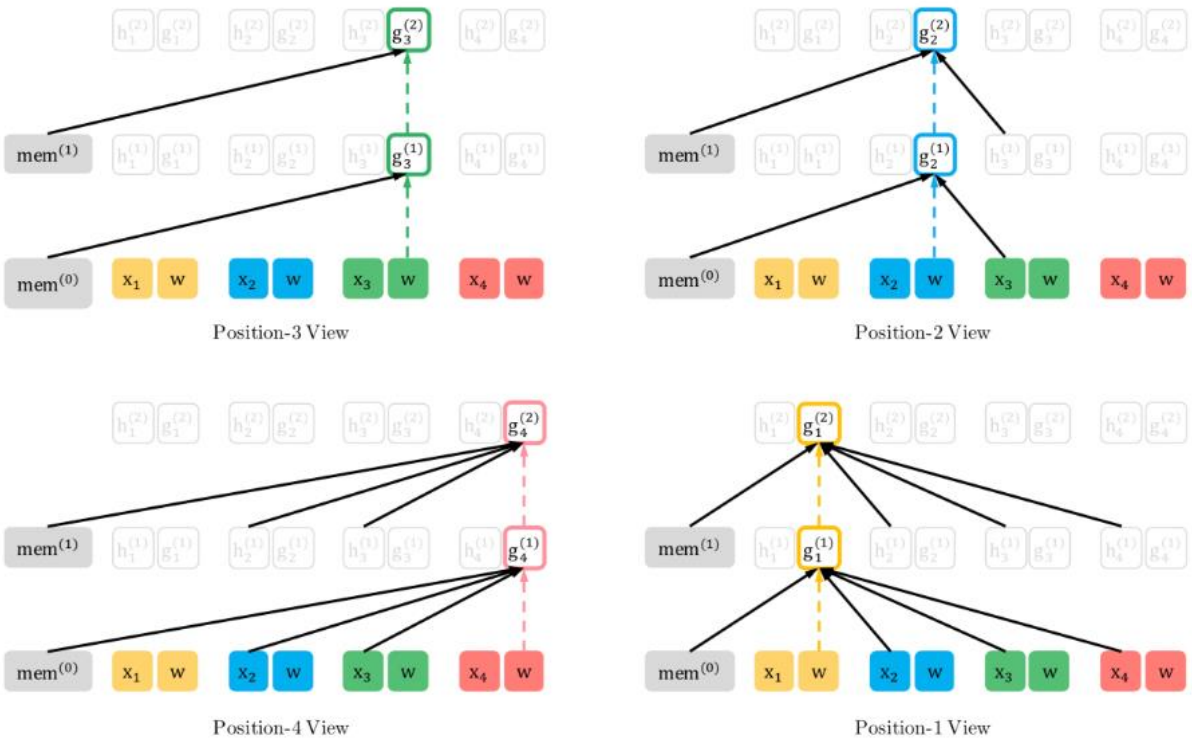

그림 4. Query stream

B. query stream g를 위의 수식과 그림 4를 이용해 이해할 수 있음

i. z의 t번째 element에 해당하는 query stream을 만들 때는 현 시점 미만의 단어 정보($x_{<t}$)와 자기 자신의 위치 정보(z_t)를 활용하는 방식으로 동작