8/12(목) 회의록

작성일시	@2021년 8월 6일 오후 6:00
작성자	하람 이하람
참석자	
최종 편집일시	@2021년 8월 17일 오후 2:00
회의 유형	일일 회의

☝학습 내용 공유

1. 강의 내용 중 질문하기🙋‍♀️

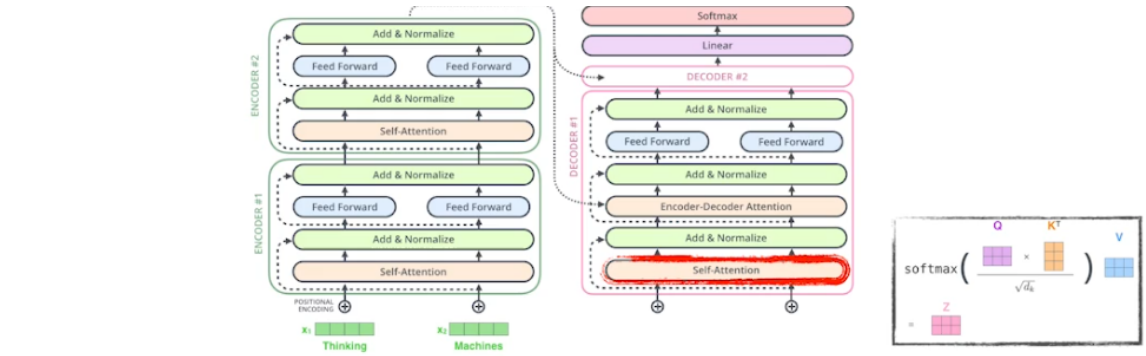
7강 Further Questions

- LSTM에서는 Modern CNN 내용에서 배웠던 중요한 개념이 적용되어 있습니다. 무엇일까요?
- Pytorch LSTM 클래스에서 3dim 데이터(batch_size, sequence length, num feature), `batch_first` 관련 argument는 중요한 역할을 합니다. `batch_first=True`인 경우는 어떻게 작동이 하게되는걸까요?

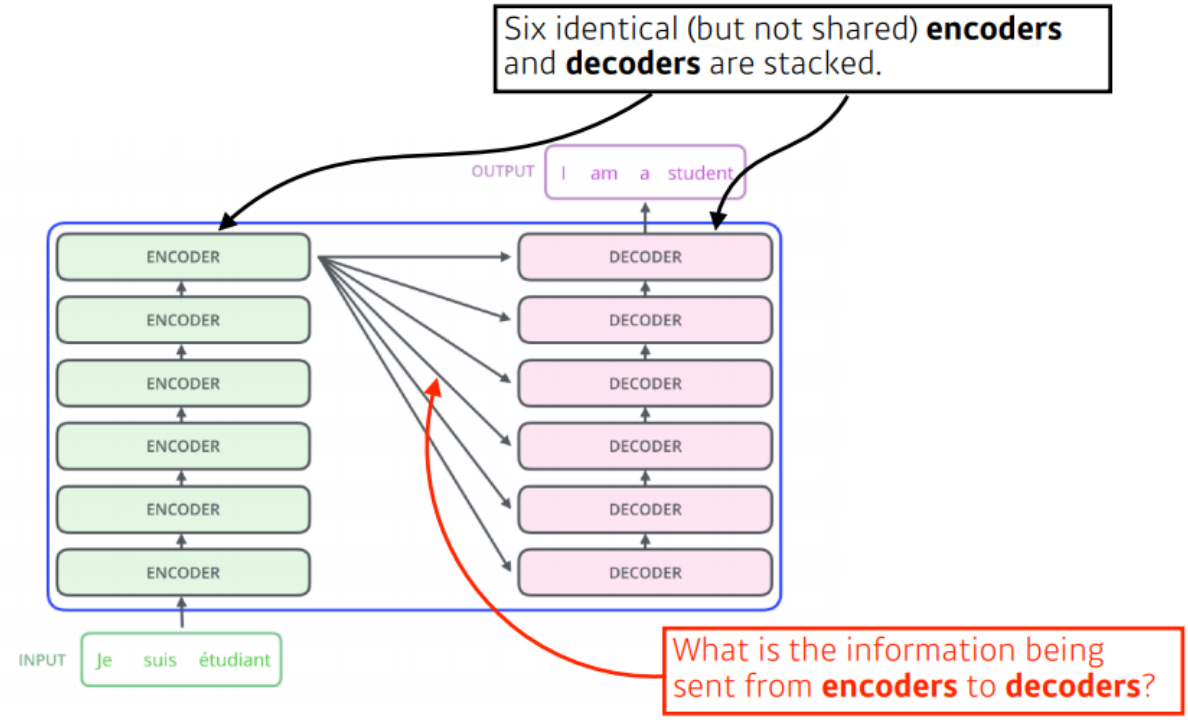
8강 Further Questions

- Pytorch에서 Transformer와 관련된 Class는 어떤 것들이 있을까요?

Q. Decoder의 Self-Attention에 들어가는 정보가 정확히 어떤거죠?? 학습을 할 때에 대해서만 얘기를 하시던데 무슨 말인지 잘모르겠어요..



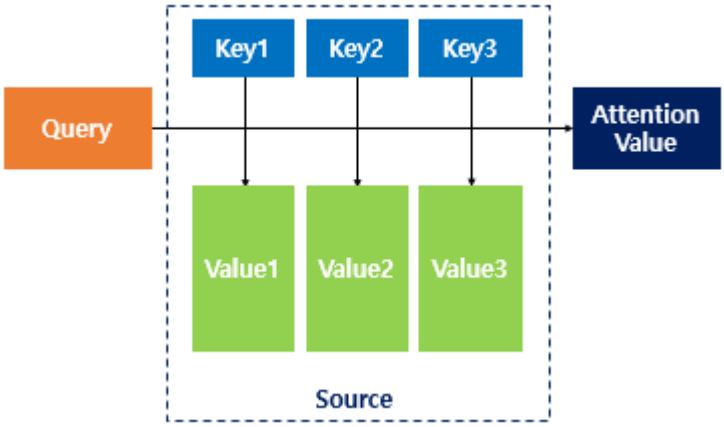
- In the **decoder**, the **self-attention layer** is only allowed to attend to earlier positions in the output sequence which is done by masking future positions before the softmax step.



https://jalammar.github.io/illustrated-transformer/ ≤== 여기가 교수님이 참고한 자료

1) 셀프 어텐션의 의미와 이점

어텐션 함수는 주어진 '쿼리(Query)'에 대해서 모든 '키(Key)'와의 유사도를 각각 구합니다. 그리고 구해낸 이 유사도를 가중치로 하여 키와 맵핑되어있는 각각의 '값(Value)'에 반영해 줍니다. 그리고 유사도가 반영된 '값(Value)'을 모두 가중합하여 리턴합니다.



여기까지는 앞서 배운 어텐션의 개념입니다. 그런데 어텐션 중에서는 셀프 어텐션(self-attention)이라는 것이 있습니다. 단지 어텐션을 자기 자신에게 수행한다는 의미입니다. 앞서 배운 seq2seq에서 어텐션을 사용할 경우의 Q, K, V의 정의를 다시 생각해봅시다.

```
Q = Query : t 시점의 디코더 셀에서의 은닉 상태K = Keys : 모든 시점의 인코더 셀의 은닉 상태들V = Valu
es : 모든 시점의 인코더 셀의 은닉 상태들
```

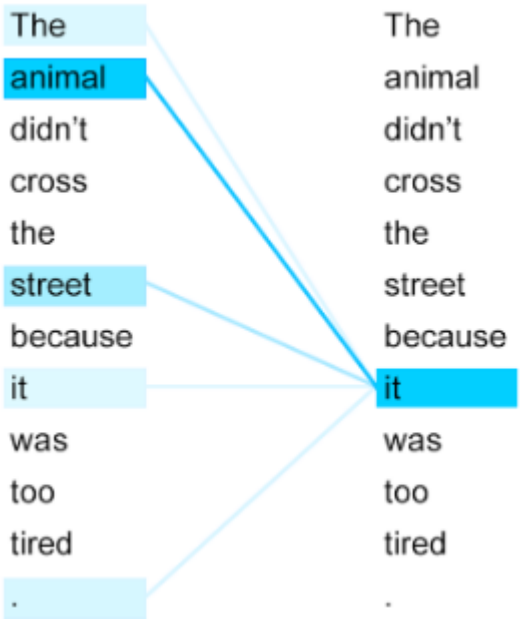
그런데 사실 t 시점이라는 것은 계속 변화하면서 반복적으로 쿼리를 수행하므로 결국 전체 시점에 대해서 일반화를 할 수도 있습니다.

```
Q = Querys : 모든 시점의 디코더 셀에서의 은닉 상태들K = Keys : 모든 시점의 인코더 셀의 은닉 상태들V =
 Values : 모든 시점의 인코더 셀의 은닉 상태들
```

이처럼 기존에는 디코더 셀의 은닉 상태가 Q이고 인코더 셀의 은닉 상태가 K라는 점에서 Q와 K가 서로 다른 값을 가지고 있었습니다. 그런데 셀프 어텐션에서는 Q, K, V가 전부 동일합니다. 트랜스포머의 셀프 어텐션에서의 Q, K, V는 아래와 같습니다.

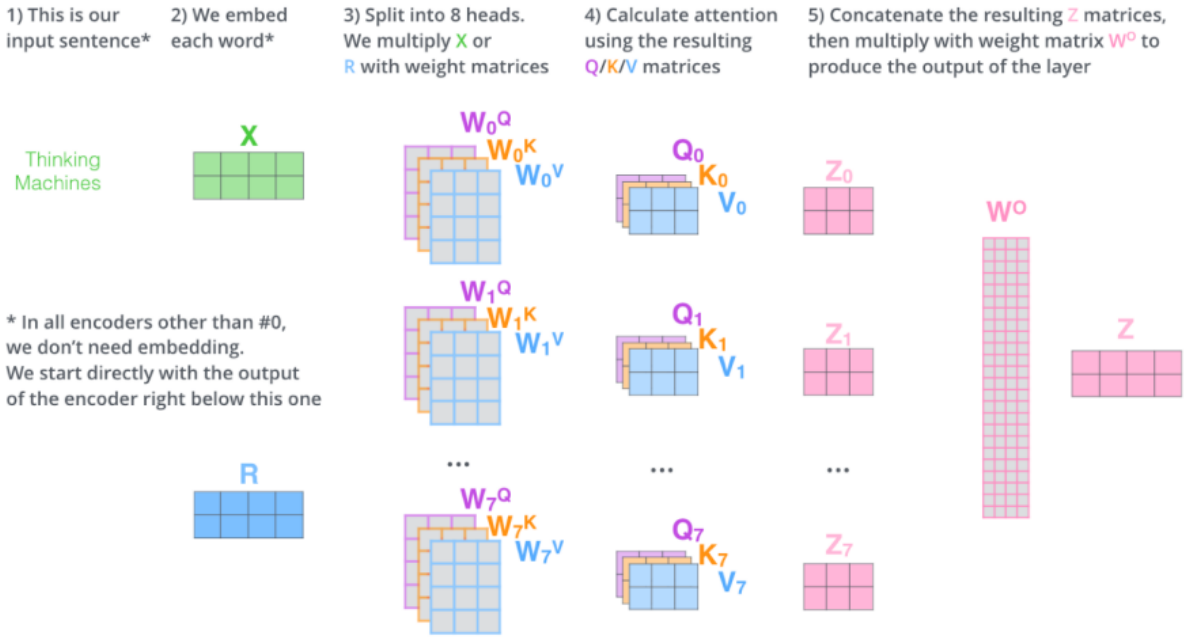
```
Q : 입력 문장의 모든 단어 벡터들
K : 입력 문장의 모든 단어 벡터들
V : 입력 문장의 모든 단어 벡터들
```

셀프 어텐션에 대한 구체적인 사항을 배우기 전에 셀프 어텐션을 통해 얻을 수 있는 대표적인 효과에 대해서 이해해봅시다.



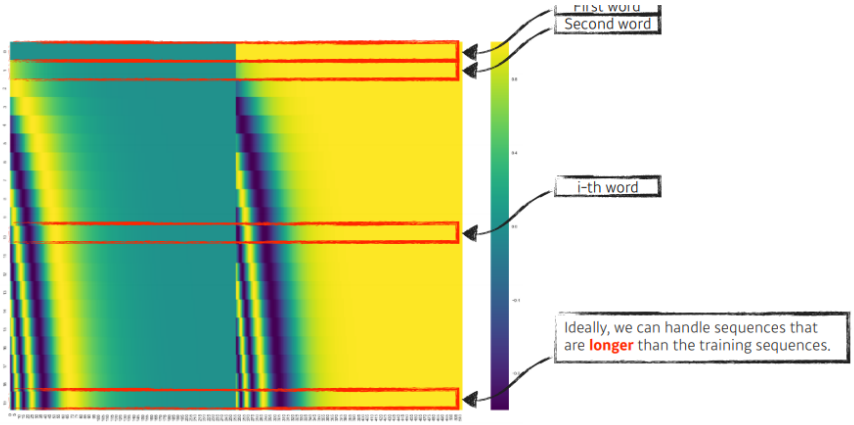
위의 그림은 트랜스포머에 대한 구글 AI 블로그 포스트에서 가져왔습니다. 위의 예시 문장을 번역하면 '그 동물은 길을 건너지 않았다. 왜냐하면 그것은 너무 피곤하였기 때문이다.' 라는 의미가 됩니다. 그런데 여기서 그것(it)에 해당하는 것은 과연 길(street)일까요? 동물(animal)일까요? 우리는 피곤한 주체가 동물이라는 것을 아주 쉽게 알 수 있지만 기계는 그렇지 않습니다. 하지만 셀프 어텐션은 입력 문장 내의 단어들끼리 유사도를 구하므로서 그것(it)이 동물(animal)과 연관되었을 확률이 높다는 것을 찾아냅니다.

Q. MHA에서 100개의 dimension을 10개로 쪼갠다는 얘기를 계속 하시던데 어떤 것의 어떤 dimension을 자른다는 얘기일까요?

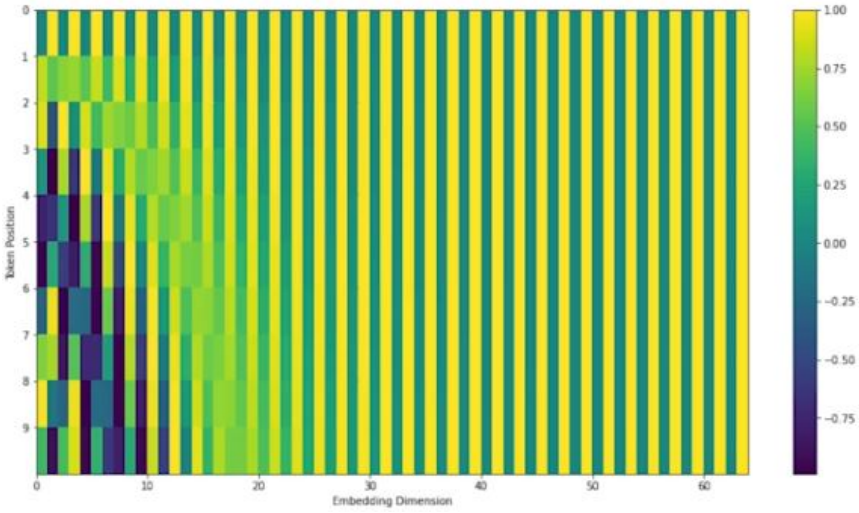


추측입니다만 위에서 8개의 HEAD로 늘린다음에 concatenate로 늘어난 dimention을 W0를 이용해서 다시 원래 길이로 뽑아내는게 아닐까 생각합니다.

Q. Positional Encoding의 조금 더 자세한 설명이 필요합니다...!



- This is the case for 512-dimensional encoding.



각 단어가 같더라도 위치 별 의미가 다를 수 있으니 위치에 대한 벡터를 추가했다?각 행이 포지션에 해당하는 벡터입니다. 열의 갯수가 문장의 길이가 되고 행벡터가 각 위치의 포지셔널 인코딩을 위한 벡터.

$$\begin{pmatrix} \mathrm{I} \\ \mathrm{go} \end{pmatrix} + \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} = \begin{pmatrix} (1,3,4,2,1,4) \\ (4,0,1,0,0,2) \end{pmatrix} + \begin{pmatrix} (0,1,0,1,0,1) \\ (1,1,0,1,0,1) \end{pmatrix}$$

3. CNN 논문 리뷰 🧐

(1) Inception-v3

발표자료: https://github.com/Barleysack/BoostCampPaperStudy/blob/main/Inception-v2v3.pdf

label smoothing →

$$H(q',p) = -\sum_{k=1}^{K} \log p(k) q'(k) = (1-\epsilon) H(q,p) + \epsilon H(u,p)$$

여기서 입실론은 하이퍼파라미터로서 사용됩니다.

(2) XceptionNet

발표자료: https://github.com/Barleysack/BoostCampPaperStudy/blob/main/Xception_사공진.pdf

기존 3X3 conv. layer를 1X1 conv. layer로 채널 성분들의 correlation을 추출한 1 channel feature map에 3X3 1 channel conv. layer를 가한 것으로 분해했다. (DS와 순서는 반대)

조급해하지말고, 비교하지말고, 천천히 이해하자

Gradient vanishing:

예전에는 sigmoid를 썼는데, 이게 그래디언트가 레이어 따라서 가다 보면 사라질 수 있다...

ReLu가 어느정도 도움이 되긴 했었는데....

Residual -잔차 블록에 의해 굳이 계산 안해도 되는 부분에선 원 그래디언트 그대로 나옵니다.

Exploding gradient:

object detection 쪽에서. 타겟 영역이 작을때, 논-타겟 영역의 그래디언트가 펑펑 터져서 오히려 학습이 안되기도 했다.

Focal Loss

ex.) 데이터 자체가 불균형할때, 요건 unbalanced class라고 부릅니다

Leaky ReLU: 작은 음수 쪽 기울기.

개꿀: EfficientNet 이외에 Pretrained 모델도 사용했고, 보통은 앙상블을 많이 쓰셨다.

하이퍼파라미터 / 옵티마이저 신경씁시다.

앙상블은 어떤 모델과 어떤 모델을?

하이퍼파라미터는 자동으로 쓰십쇼... - 멘토님: Optima

우리는 RAY를 다음주 즈음 배울 예정입니다.

VIT, ResNet, EfficientNet, MMNet

PIM : 파이토치 이미지 모듈스

LR 스케쥴러 사용.

모델 최적화 대회때는 아주 빡빡히 체크할 예정.

TTA : 요거는 마지막에 걸어주는 것.

로깅 필요성 多 : 애초에 학습시킬때 저장해둡시다.

파이썬 팁: Type Hint

여러명이 작업할때는 파이썬 타이핑 꼼꼼히 알려주셔야... 사단이 나지 않습니다...