

**8주차 과제**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



|  |  |
| --- | --- |
| 학번 | 2020136087 |
| 이름 | 윤아현 |
| 수강 과목 | 자연어처리 |
| 담당 교수 | 임상훈 교수님 |
| 제출일 | 2024.05.04 |

**문제 및 해결 방법 (구현 코드)**

**1. Attention Heatmap 출력 및 분석 (50점)**

\* Transformer 모델의 Attention distribution을 Heatmap 형식으로 출력하시오.

    \* Attention Heatmap을 출력한 후 비교, 분석 하시오

        \* 학습전, 학습후(최소 10epoch 이상)의 차이

        \* Head별 분석

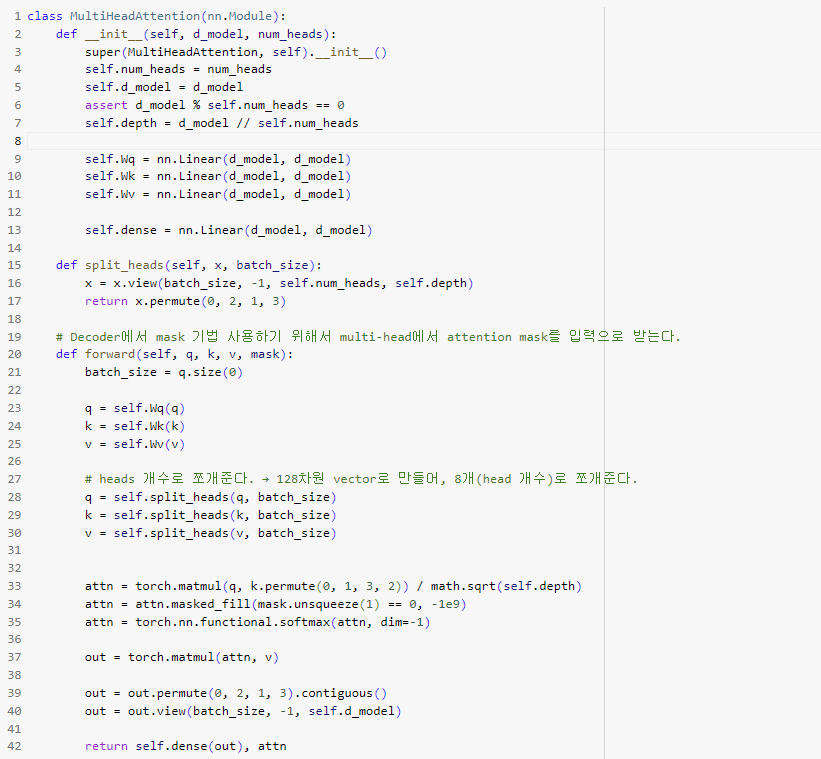
        \* Layer 별 분석

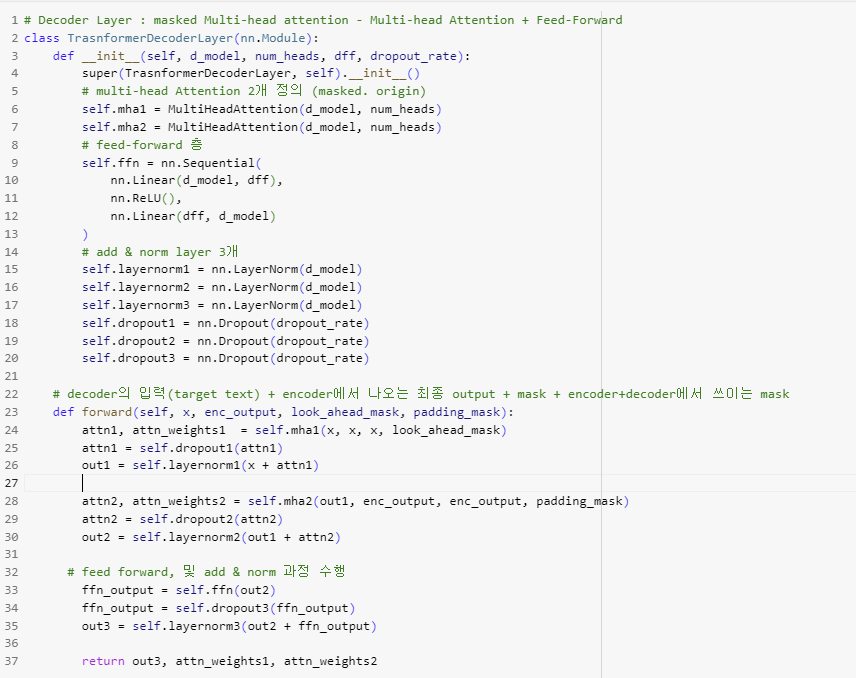
**\*\*GRADING\*\***

\* Heatmap 출력 (+20점)

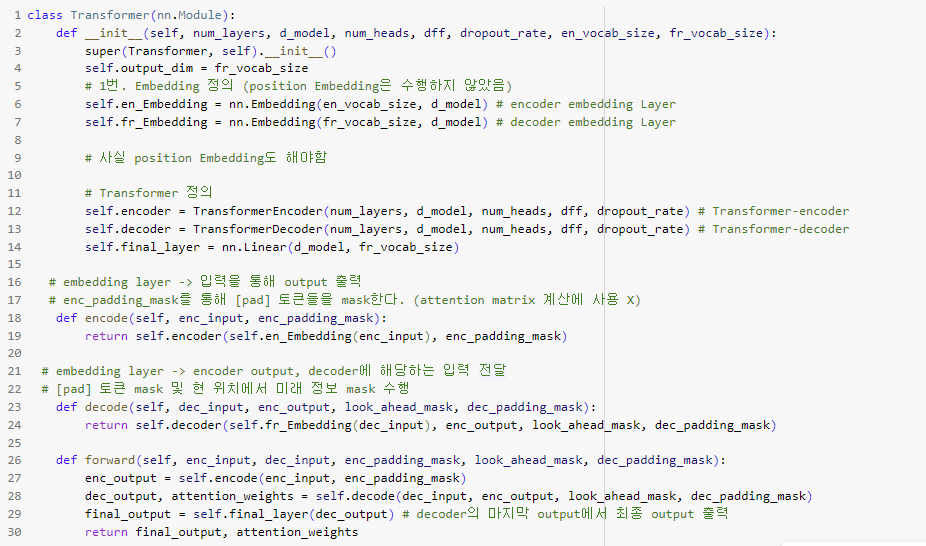
\* Heatmap 비교 분석 (+30점)

**Code**

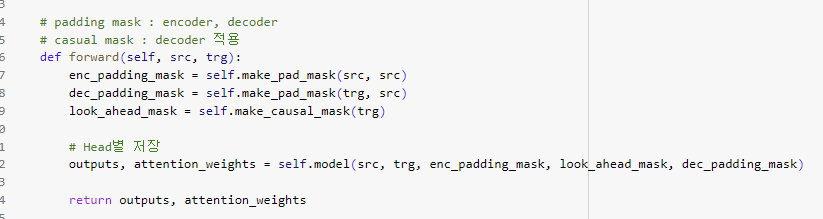


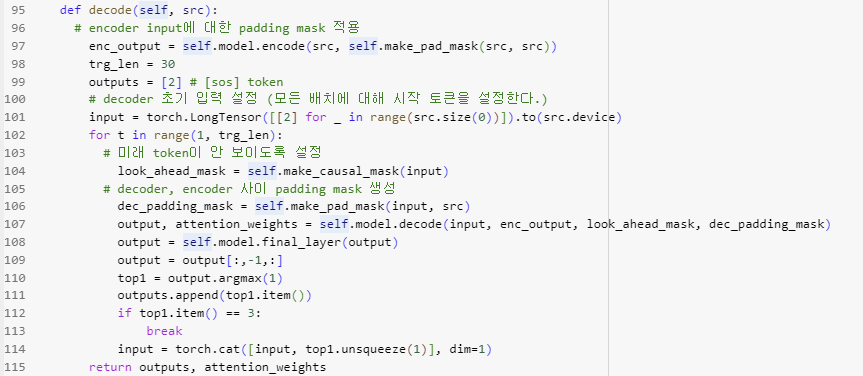
****

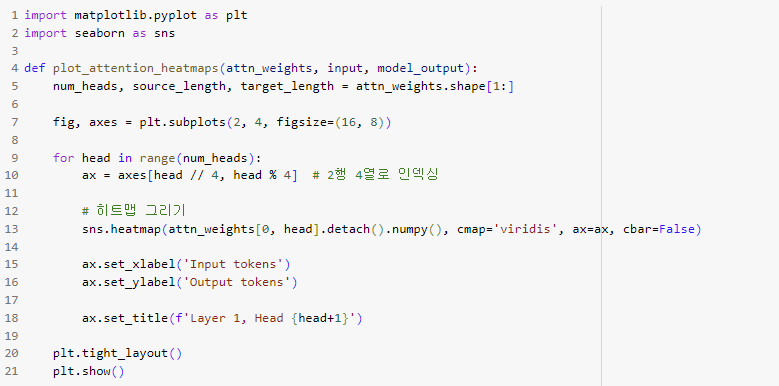
****

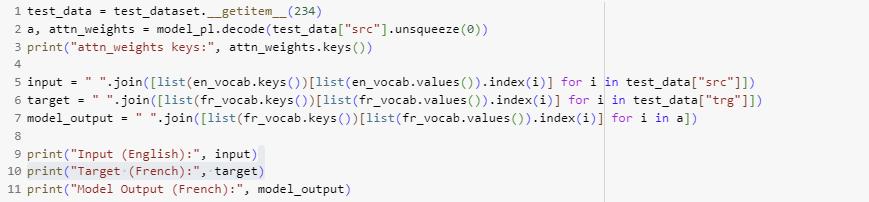
****

**[Transformer PL 함수]**

****

****

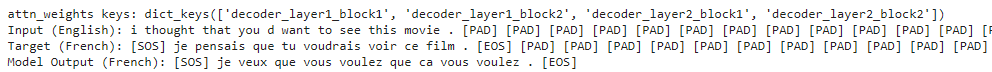
****

****

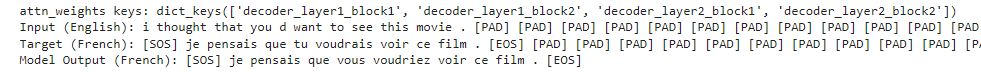
* 수정한 코드만 첨부하였음

**Outputs**

**Epoch 1:**

****

**Epoch 10:**

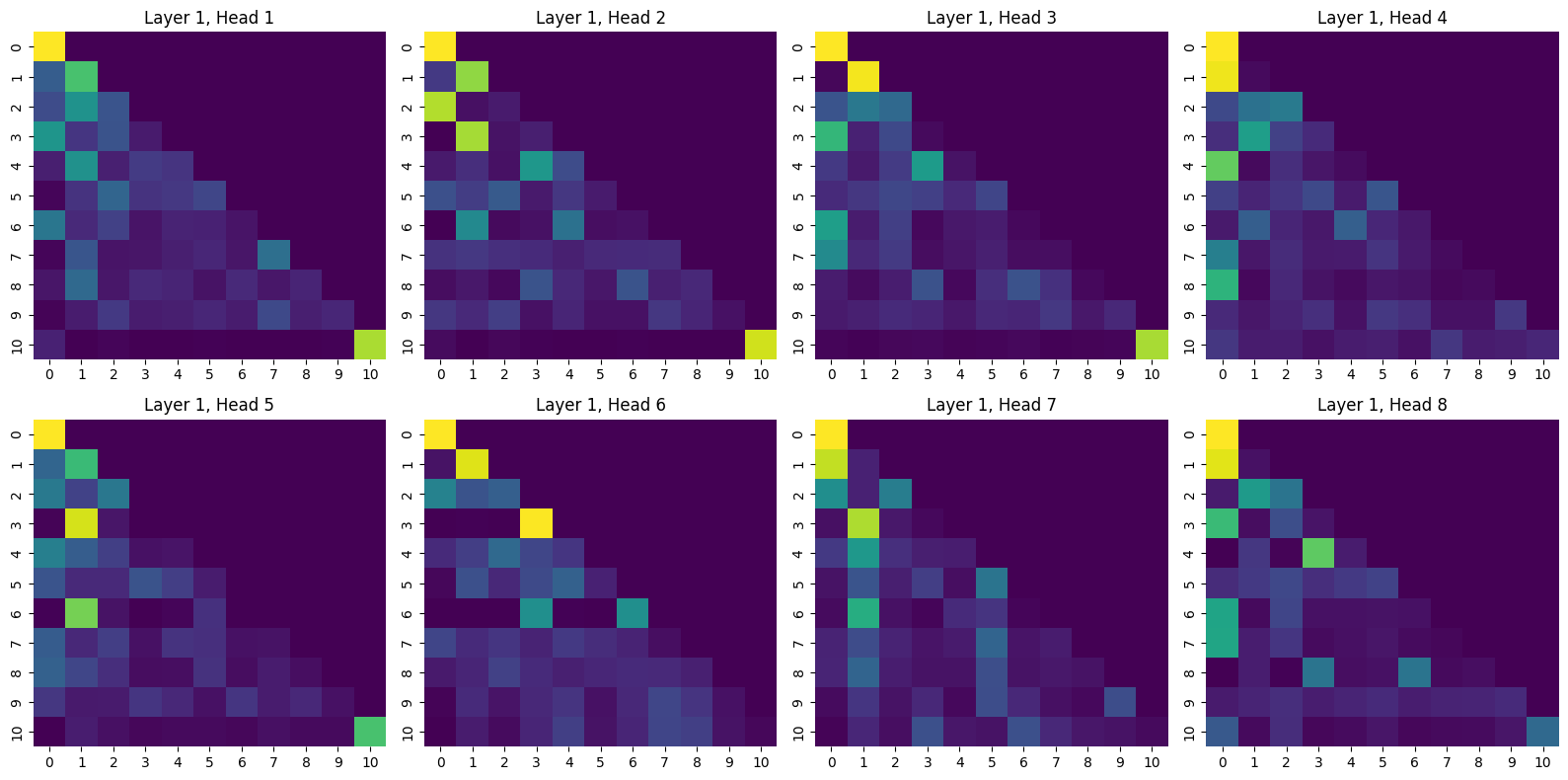
****

**Epoch 1 VS Epoch 10 Layer 및 Head별 분석:**

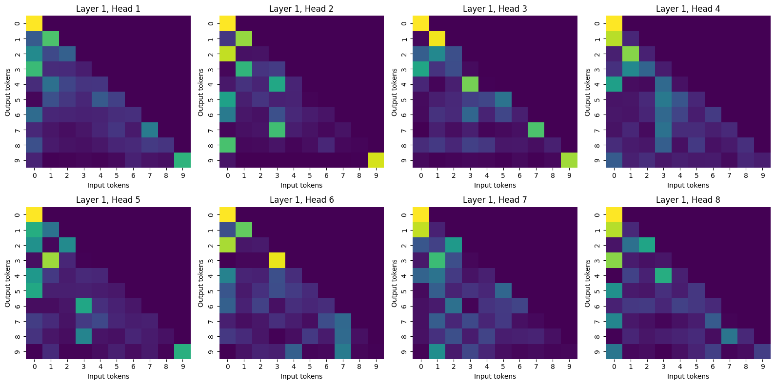
위에서 나온 Input과 Model Output을 통해 히트맵을 비교하였다.

**Decoder Layer1 Block1 (Decoder의 Multi-head Self-Attention, 첫 번째 Layer)**

**Epoch 1:**

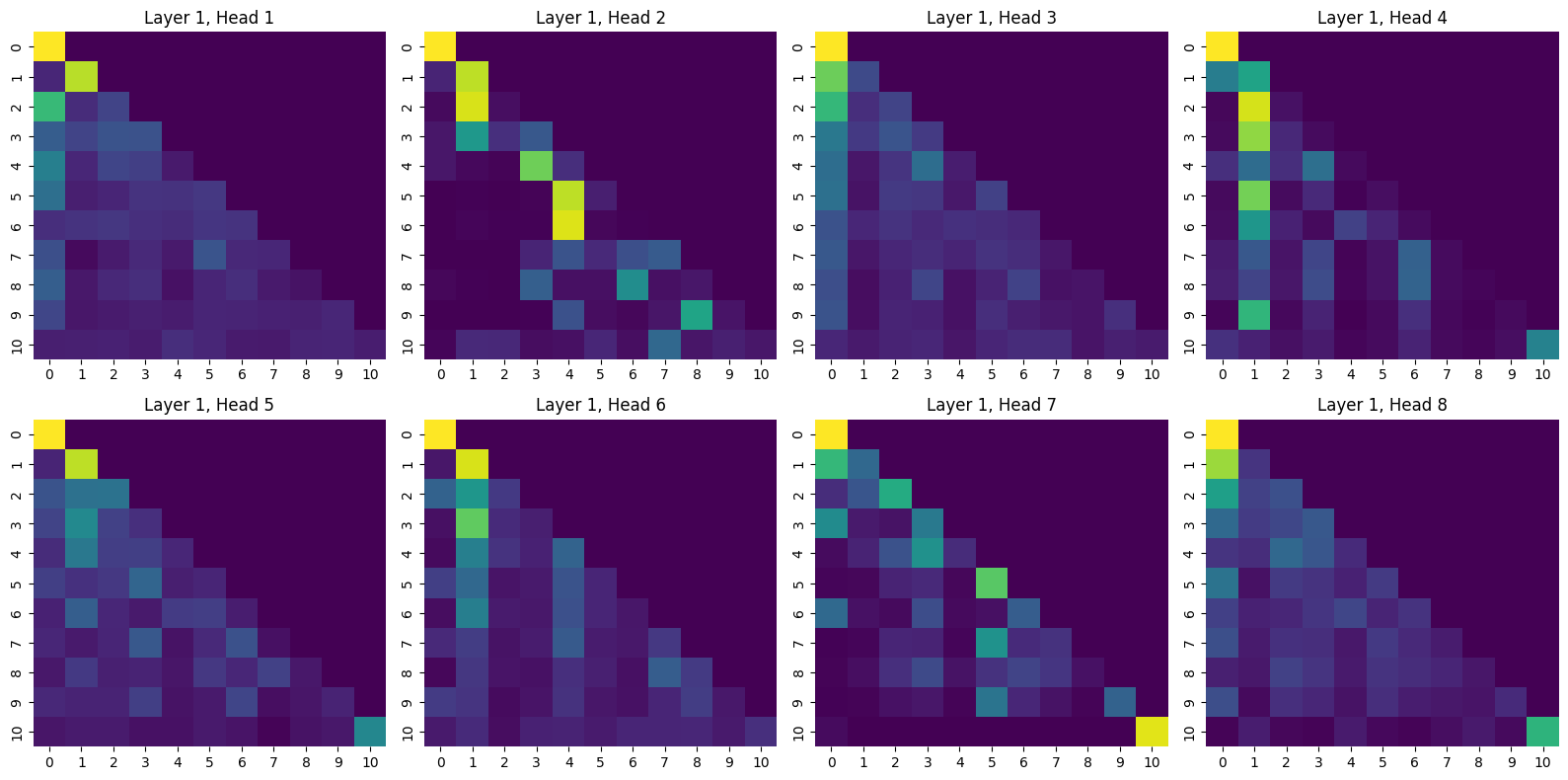


**Epoch 10:**

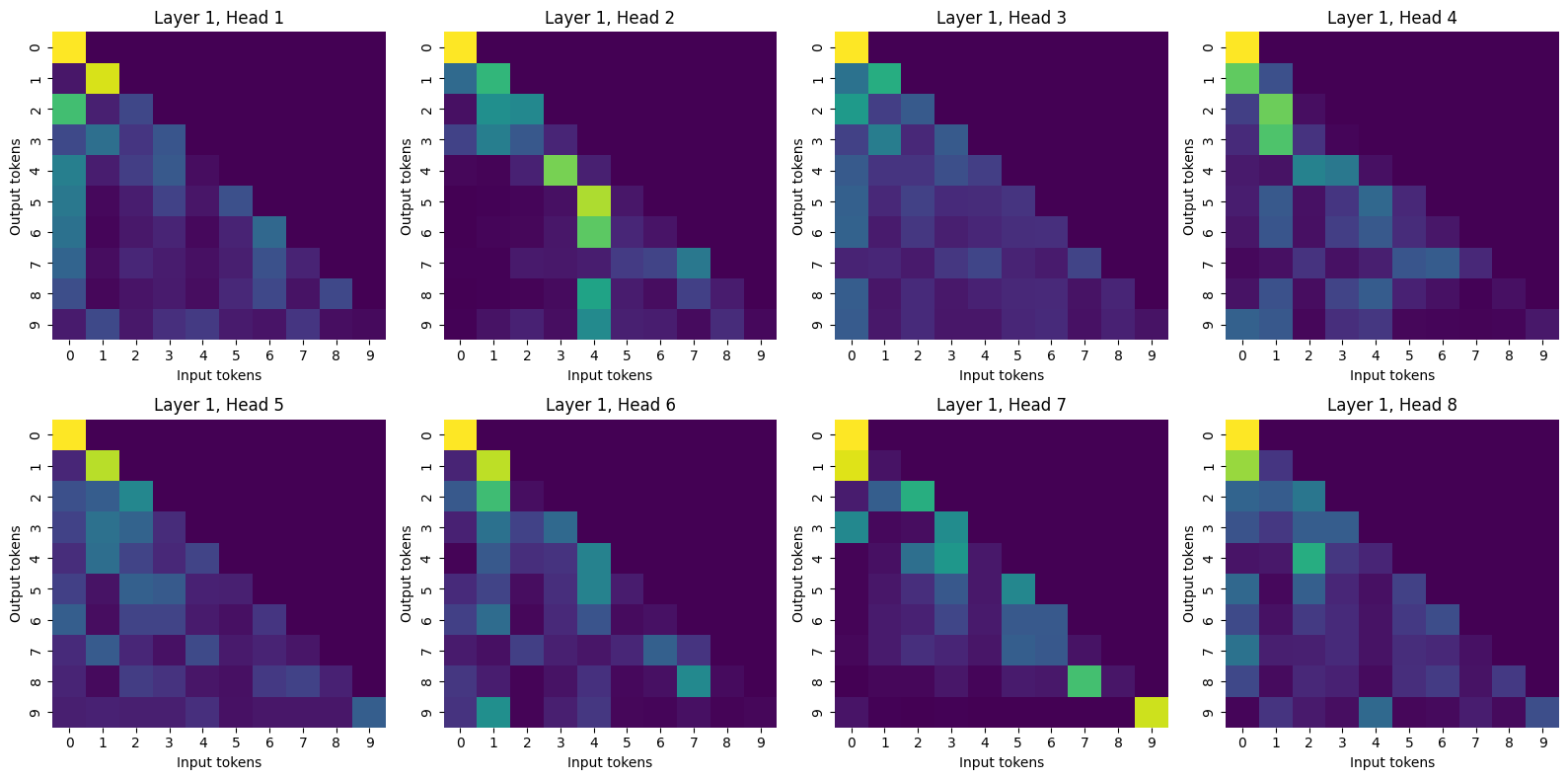


**Decoder Layer2 Block1 (Decoder의 Multi-head Self-Attention, 두 번째 Layer)**

**Epoch 1:**



**Epoch 10:**

****

**분석 결과**

1번. Head별 분석

각 Head들은 같은 heatmap의 분포를 보이지 않고, 서로 다른 특성들을 반영한 heatmap 분포를 보인다. 몇 개의 Head 1은 전반적으로 가중치가 행의 초반 부분에 집중되어 있지만, 갈수록 중앙 부분 혹은 고른 분포를 보이는 경향이 보였다.

하지만, Layer 1의 Head 7은 가중치가 행의 후반 부분에 집중되는 특이한 분포를 보였다.

2번. Layer 별 분석  
Layer1과 Layer2를 비교해보았을 때 Layer1의 대체로 모든 단어에 대해 가중치의 다양한 구역에서 가중치가 높게 나타난다면, Layer2는 비교적 가중치가 한 부분에 명확하게 나타낸다.

이를 통해, Layer1은 서로 다른 부분을 비등비등하게 다양하게 해석을 하고 있고 Layer2는 조금 더 한 부분에 집중하여 세밀한 학습을 수행하는 것처럼 관찰되었다.

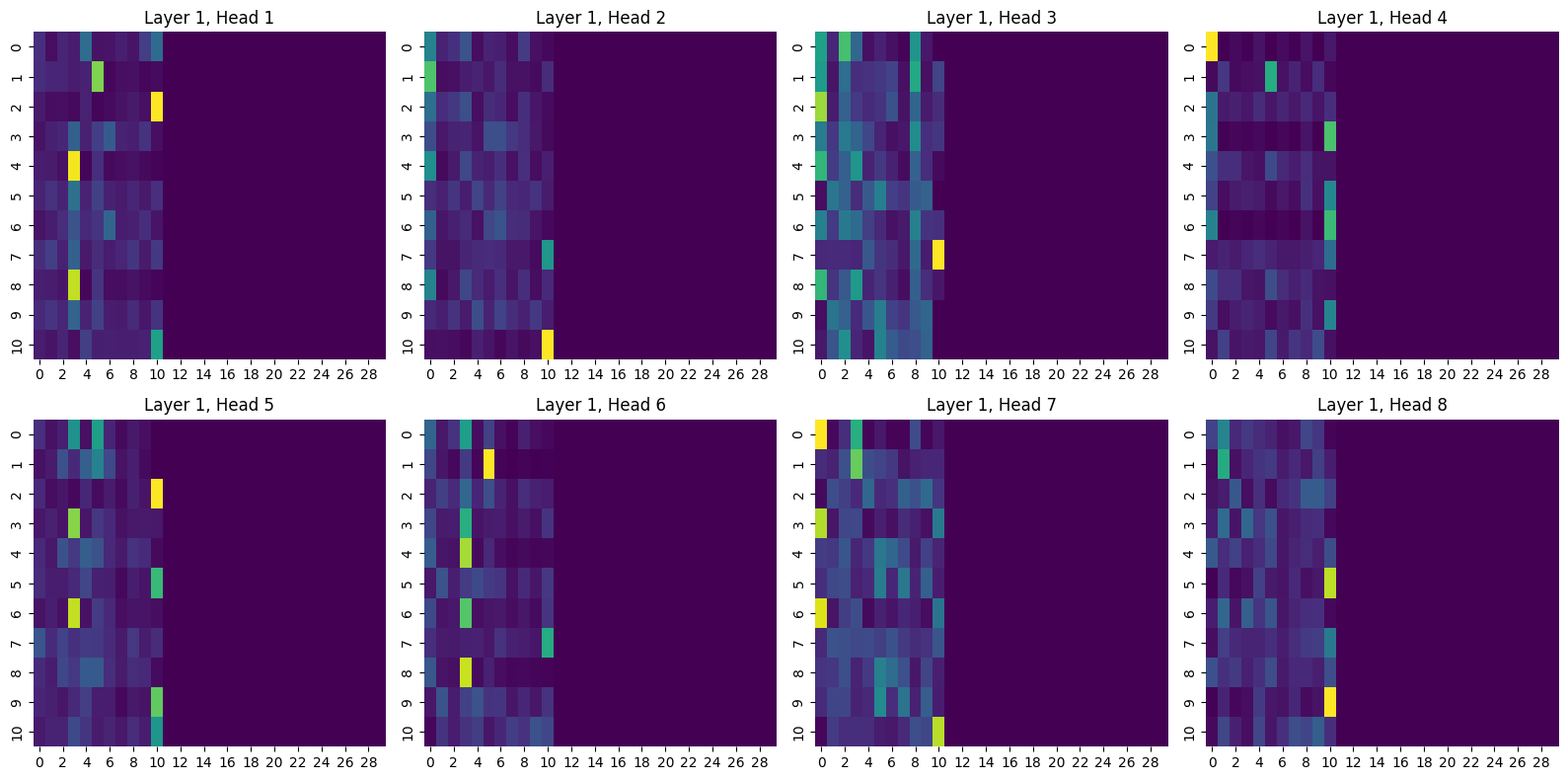
3번. 학습 수에 따른 분석

Epoch를 1번, 10번 진행되었을 때를 분석해보면, 각 Head별로 가중치 분포가 조금 더 분명하게 보였다. 초기에 집중되거나 상관관계가 각 단어별로 비슷하게 나타나는 경향이 줄었고 한 부분에 상관관계가 높게 나타나는 비율이 높아졌다.

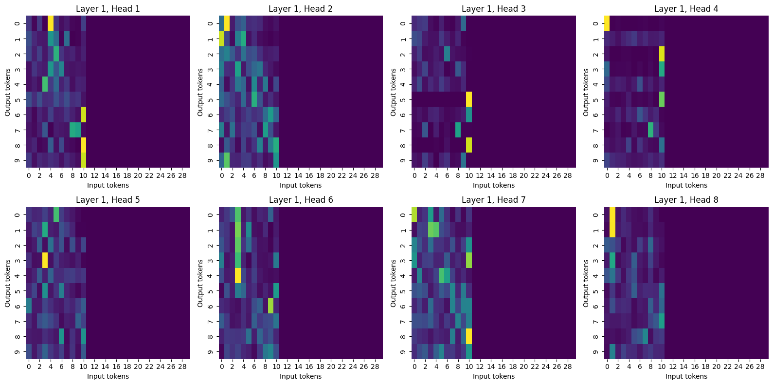
학습을 많이 수행할수록, 각 단어가 다른 단어와 얼마만큼의 연관이 있는지가 더 선명하게 보이는 것을 관찰할 수 있었다.

**Decoder Layer1 Block2 (Decoder의 Multi-head Attention, 첫 번째 Layer)**

**Epoch 1:**

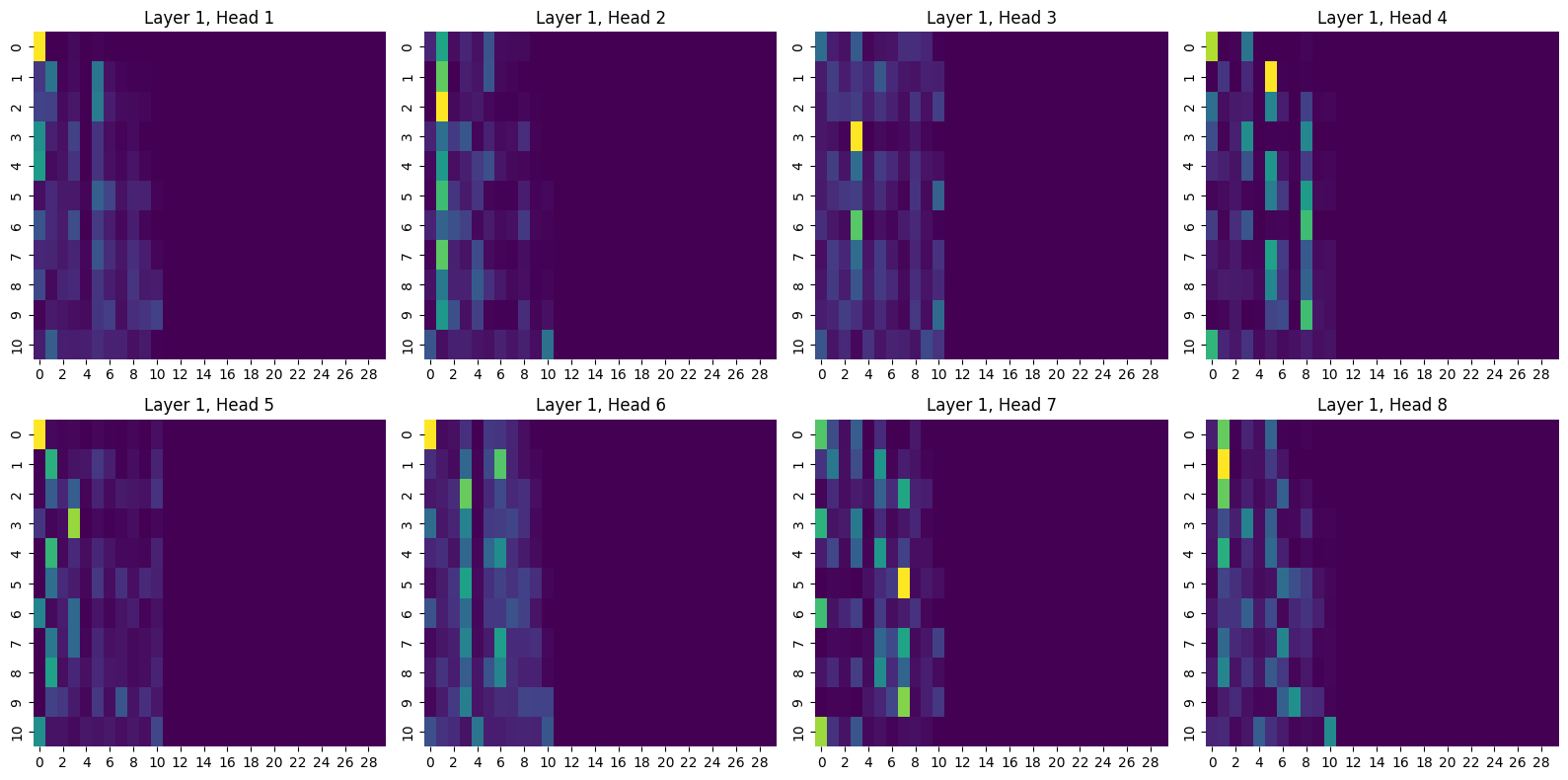


**Epoch 10:**

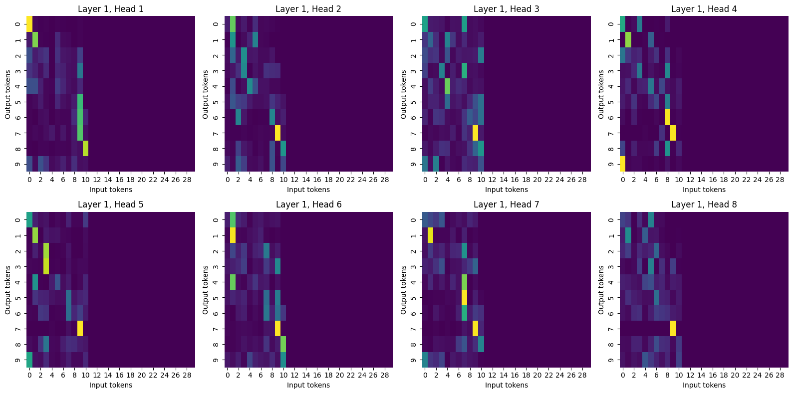
****

**Decoder Layer2 Block2 (Decoder의 Multi-head Attention, 두 번째 Layer)**

**Epoch 1:**

****

**Epoch 10:**

****

**분석 결과**

1번. Head별 분석

Decoder의 Multi-Head Self-Attention과 비슷하게 다양한 어텐션 분포를 보인다. 각 Head별 공통점 없이 서로 다른 특성들에 집중하고 있다.

어떤 부분들은 매트릭스의 초기 부분(예시. Head 1)에 상관관계가 집중되어 있거나, 하단 부분(예시. Head 7), 특정 지점, 전반적으로 균일한 분포를 가지는 것을 관찰할 수 있다.

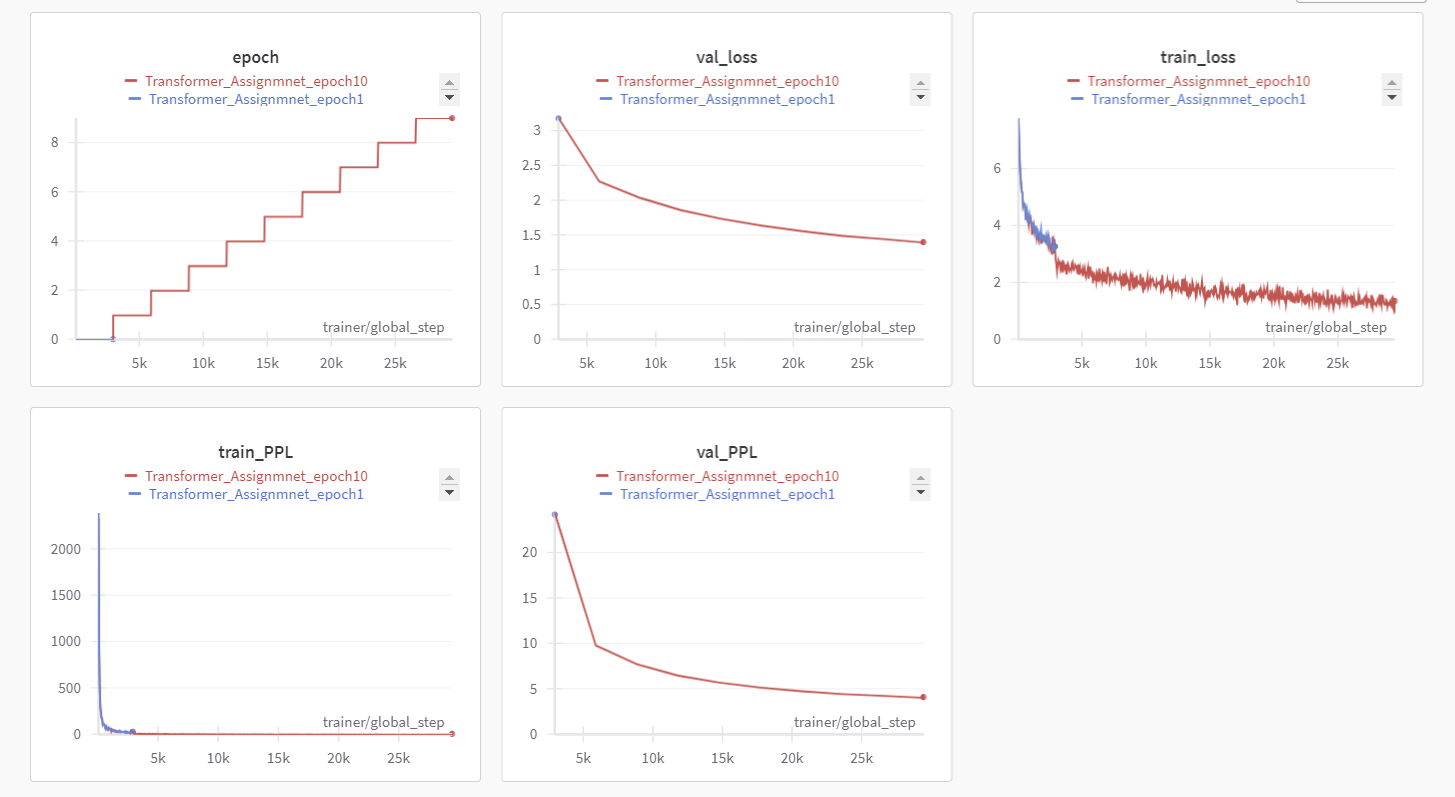
2번. Layer 별 분석  
Layer1은 전반적으로 몇몇 특정 입력 값에 집중되는 것 없이 균일한 분포를 보인다. Layer2는 다른 조금 더 특정 입력 값에 선명하게 집중되는 것을 관찰할 수 있다.

3번. 학습 수에 따른 분석

동일하게 학습 수가 많아질수록, 특정 부분에서 더 선명하게 가중치 분포가 나타나는 것을 볼 수 있다. 이는 특정 문맥이나 의미적 관계를 더욱 잘 파악하고 있다는 것이다.

**Model Output(번역 결과)를 비교했을 때도,** 학습을 수행할 수록 더 많은 단어들을 정확하게 예측한다는 것을 확인할 수 있다.

**WANDB Loss값 출력**

****

**2. Transformer의 장점, 단점 (50점)**

\* RNN 계열 모델, CNN 계열 모델에 대비 Transformer의 장단점을 서술하시오.

\* 지금까지한 실습, 과제를 기반해 근거를 제시하시오.

RNN Model과 Transformer Model을 비교해보면,

RNN Model순차적 학습 Model로써, 병렬 처리가 불가능하기 때문에 학습하는 데 시간이 오래 걸렸다. 이전 실습과 비교해보았을 때 Seq2Seq의 Epoch를 한 번 돌릴 때 10분 이상 걸렸는데 Transformer Model을 사용하니 Epoch를 한 번 돌릴 때 1분 미만으로 학습이 수행된다. Transformer는 Attention 기법을 사용하여 입력 시퀀스의 모든 값들을 동시 처리할 수 있으므로 훨씬 빠른 추론이 가능하다.

CNN Model과 Transformer Model을 비교해보면,

CNN Model은 병렬처리와 특정 지역에 초점을 맞추어 작동한다. 병렬처리가 가능하여 속도면에서는 빠르지만, Kernel을 통해 특정 지역에만 의존하다 보니 긴 시퀀스에 대해서는 전반적인 문맥 파악이 어렵다. (멀리 떨어진 정보들에 대한 연관성을 파악하기엔 한계가 있다.)

Transformer는 Self-Attention을 사용하여 문장에서 하나의 입력 값이 다른 입력 값과 얼마나 많은 연관성을 가지는 지 알 수 있기에 문맥을 파악하는 데 더 좋은 성능을 보인다. (예를 들어, it과 같이 문맥을 통해서만 파악할 수 있는 단어들에 대한 파악이 더욱 쉽다.)

Transformer Model의 단점에 대해서 추측해보았을 때,

Model의 구조만 보아도 많은 Layer(Self-Attention Layer, General-Attention Layer, Feed-Forward Layer, Add&Norm Layer)들이 사용되기 때문에 높은 메모리가 필요하다. 수많은 파라미터들이 존재하고, 시퀀스가 길수록 파라미터 수도 비례해서 늘어난다. 이를 통해 많은 GPU를 사용하게 되며 Transformer Model 자체를 처음부터 직접 설계하는 것에는 한계가 있기 때문에 Pre-Trained Model에 의존해야 된다는 단점이 있다.

또한, 현재는 max sequence를 30으로 설정하여 30개의 단어까지만 처리를 수행하다보니, 이보다 더 긴 문장이 들어왔을 때 주요 정보들이 삭제되어 맥락을 제대로 파악할 수 없어 Model의 성능이 급격하게 저하될 수 있다는 단점이 있다. Sequence가 30보다 훨씬 더 짧은 문장들은 [PAD] 토큰을 사용하여 단어를 추가하기에 모델의 학습 효율성을 저하시킬 수도 있다.