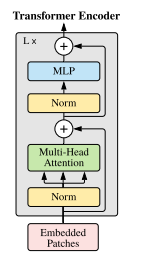
Vit encoder 구조에 Bert encoder 구조를 참고하여 Layer Normalization 단계 추가

자연어 처리 모델에서 사용하는 Transformer를 이미지 분류에 적용시킨 Vision Transformer에 관한 논문이다.

1. 기존 Vit Transformer Encoder의 구조는 Layer Normalize 과정이 Multi-Head Attention 이전, MLP 이전에 두 번 진행된다.

Transformer Encoder의 내부구조는 다음과 같다.

[내부구조]. Transformer Encoder 내부에서는 먼저 Layer Normalization을 거치고, Multi-Head Attention을 지난 결과를 통과하지 않은 패치와 Skip Connection 시켜줍니다. 그리고 다시 Layer Nomalization, MLP를 거쳐 Skip Connection으로 다시 더해주는 것이 한 번 Transformer Encoder를 통과한 것입니다. 이러한 Transformer Encoder를 L번 반복합니다.

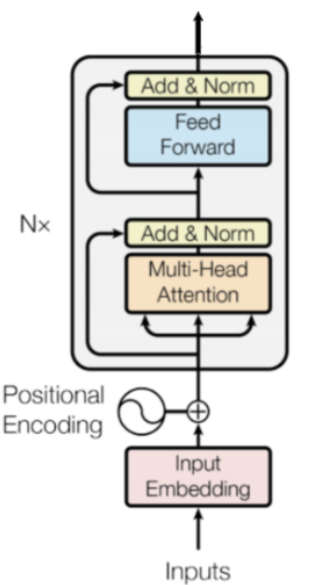


[내부구조 수식]



이러한 과정을 L번 반복합니다.

1. Vit의 Transformer Encoder는 Bert의 Transformer Encoder 구조와 약간의 차이가 존재하는데 Bert의 Transformer Encoder 구조를 차용하여 성능향상을 기대한다.

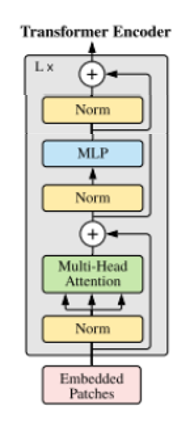


Bert의 Transformer Encoder는 output 이전에 마지막으로 Linear Norm 시켜주는데 이를 기존 Vit Transformer에 적용한다.

1. 기존의 Transformer Encoder의 구조는 다음과 같다.

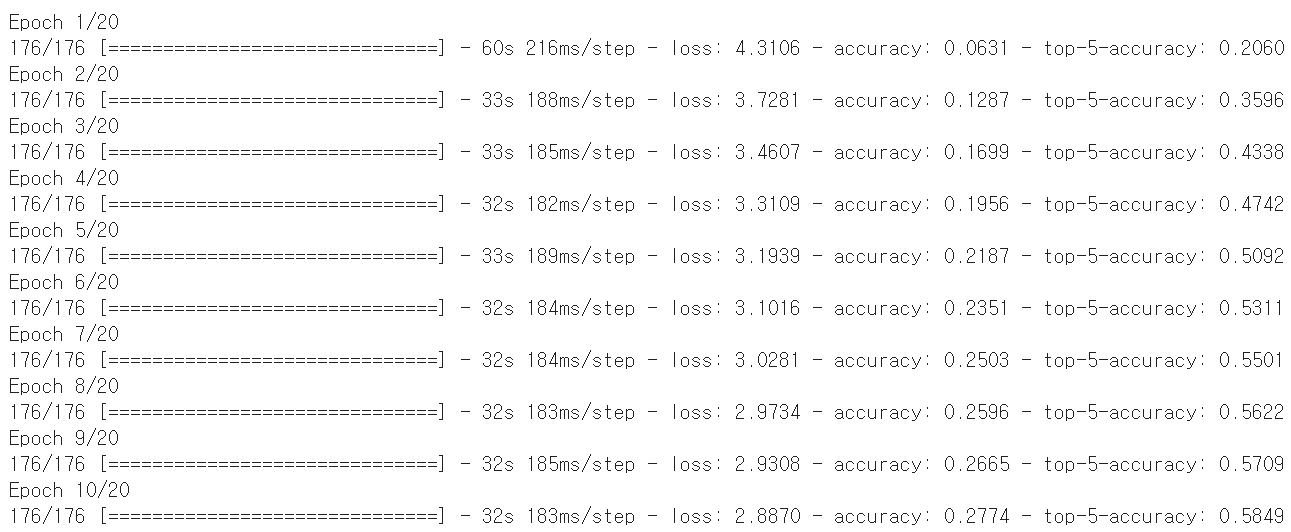
 1) Norm  
 2) Multi-Head Attention  
 3) Skip Connection  
 4) Norm  
 5) MLP  
 6) Skip Connection  
  
  
x1 = layers.LayerNormalizatio(epsilon=1e-6)(encoded\_patches)         
  
attention\_output = layers.MultiHeadAttention(                        
num\_heads = num\_heads, key\_dim = projection\_dim, dropout=0.1)(x1, x1)  
  
x2 = layers.Add()([attention\_output, encoded\_patches])                
  
x3 = layers.LayerNormalization(epsilon=1e-6)(x2)                     
  
x3 = mlp(x3, hidden\_units = transformer\_units, dropout\_rate = 0.1)   
  
encoded\_patches = layers.Add()([x3, x2])

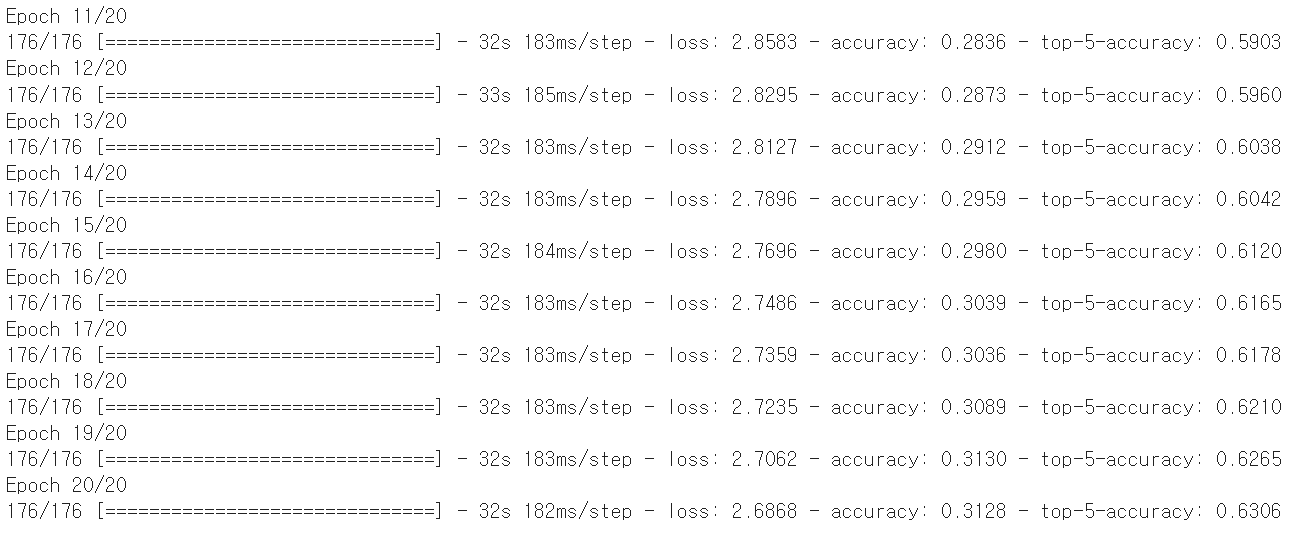
1. Bert의 구조와 유사하게 MLP 이후에 Linear Normalization을 추가한 Transformer Encoder의 구조는 다음과 같다.



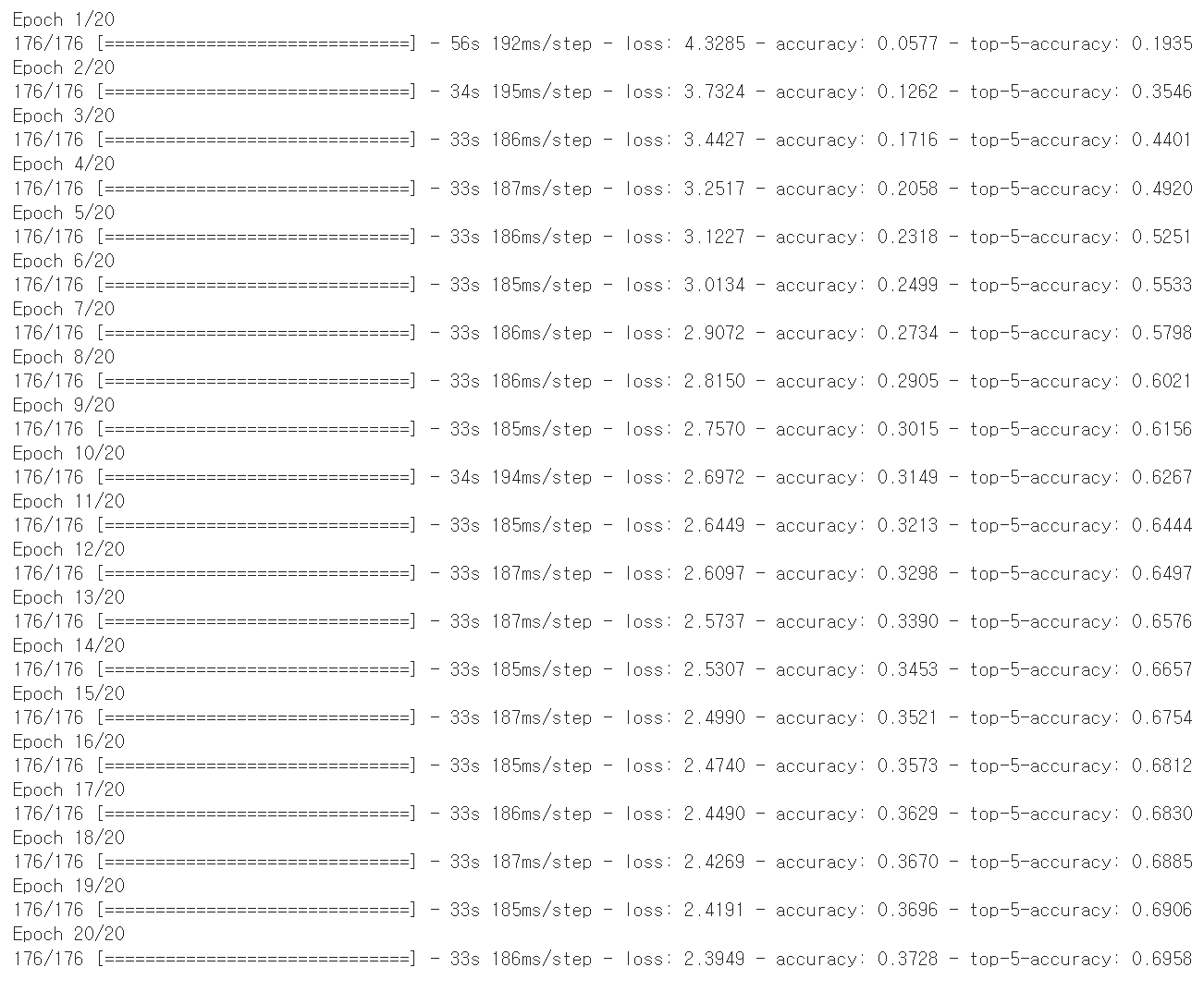
 1) Norm  
 2) Multi-Head Attention  
 3) Skip Connection  
 4) Norm  
 5) MLP  
 6) Norm (Added)  
 7) Skip Connection  
  
  
x1 = layers.LayerNormalizatio(epsilon=1e-6)(encoded\_patches)         
  
attention\_output = layers.MultiHeadAttention(                        
num\_heads = num\_heads, key\_dim = projection\_dim, dropout=0.1)(x1, x1)  
  
x2 = layers.Add()([attention\_output, encoded\_patches])                
  
x3 = layers.LayerNormalization(epsilon=1e-6)(x2)                     
  
x3 = mlp(x3, hidden\_units = transformer\_units, dropout\_rate = 0.1)   
  
x3 = layers.LayerNormalization(epsilon=1e-6)(x3)  (Added)  
  
encoded\_patches = layers.Add()([x3, x2])

1. 결과분석

기존 Vit 학습 결과 (before) 



20epochs 이후 loss가 2.6868, accuracy가 0.3128 이다.

Bert의 Transformer Encoder 구조를 차용해 Linear Normalization을 추가한 Vit의 학습 결과 (after) 

20epochs 이후 loss가 2.3949, accuracy가 0.3728 이다.

loss가 0.2919 감소, accuracy가 0.06로 소폭 증가했다.

구현

구현할 모델에서는 Layer의 개수를 12개, D의 크기를 64, MLP의 크기를 1024, Head 개수를 4로 설정하여 진행한다.