텍스트 분류 목적으로
Vision Transformer
Swin Transformer 적용해 보기

디지털애널리틱스 융합학과

조규원 2022311916
성형훈 2022311914
김영채 2022311930

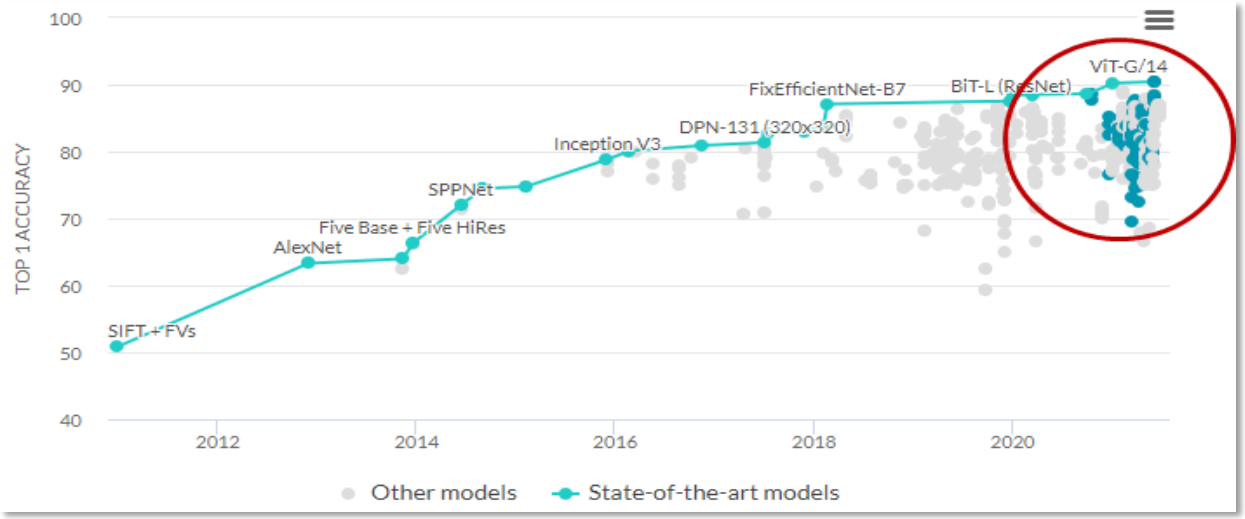
목차

INTRO

- 연구배경

  Transformer는 Text Task 높은 성능을 보여주었고,
  Vision Task를 수행하기 위해 Vision Transformer로 개선되어 Image Classification Task 점령

  저희는 반대로 Vision Transformer를 사용해 Text task를 수행함으로써 Transformer에 새로운 활용방안을 찾아보려 연구를 진행하게 되었습니다.



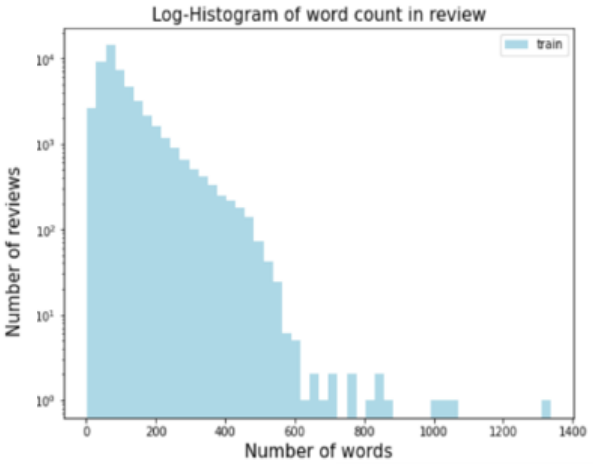
DATA

- IMDB Dataset

	review	sentiment
0	One of the other reviewers has mentioned that ...	positive
1	A wonderful little production. The...	positive
2	I thought this was a wonderful way to spend ti...	positive
3	Basically there's a family where a little boy ...	negative
4	Petter Mattei's "Love in the Time of Money" is...	positive

7 : Train

3 : Test



변수명	변수 설명 및 예시	Dtype
Review	Text(리뷰)	Object
Sentiment	Positive(긍정리뷰) / Negative(부정리뷰)	Object

총 단어 개수: 85,000
단어의 개수: 중앙값 87
평균 : 108

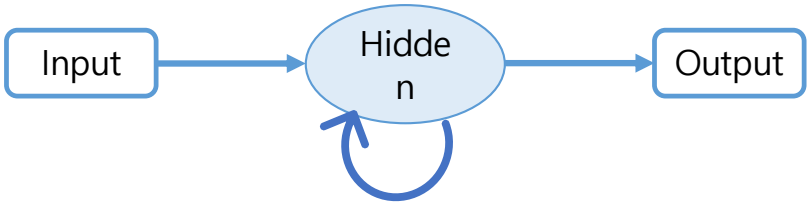
Preprocessing

- Stopwords
  - Spacy의 En_core_web_sm사용
- Preprocess
  1. 소문자(lower)로 변형
  2. 's / 've / 't / 'm / 'd / 'll .etc
     -> is / have / cannot / am / would .etc 변형
  3. Lemmatization 추출
  4. Data split
     - X_train : train_data['review'] -> 35000
     - X_test : test_data['review'] -> 1500
     - Y_train : train_data['sentiment'] -> 35000
- Tokenizer
  - Max_feature = 9000
  - Maxlen = 100
  - Embedding size = 300
- Padding
  - Maxlen(100)으로 지정 -> X_train/X_test에 진행
- Index 부여
- Train data

  -> train / valid 9:1비율로 나눔
- Word2Vec
  - 단어 벡터 간 유의미한 유사도를 반영할 수 있도록 단어의 의미를 수치화 하는 작업
  - Gensim 사용
  - Word2vec사전 모형 사용
    - 구글에서 제공되는 GoogleNews-vectors-negative300.bin.gz
  - Word2vec model size -> 3000000,300
  - Embedding_matrix -> 90000,300
  - 단어와 맵핑되는 사전 훈련된 임베딩 벡터값

RNN

순환 신경망으로 문장, 음성, 동영상과 같은 Sequential Data 학습에 특화된 인공신경망

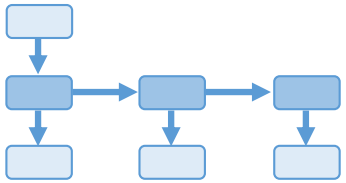


은닉층의 결과값이 출력층으로 보내지면서
다음 입력값과 함께 은닉층의 입력으로 들어가는 특징
= 은닉층이 순환하는 구조

은닉층이 이전 시점의 은닉층으로부터 영향을 받게 되므로, 신경망 자체도 이전 시점의 영향을 받음
과거의 기억을 이용해 학습

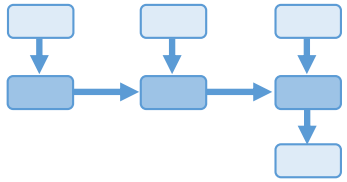
RNN

< One – to - many >



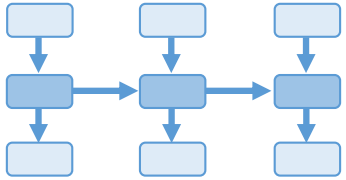
이미지를 보고 자막을 다는
Image Captioning에 활용

< Many– to – one >



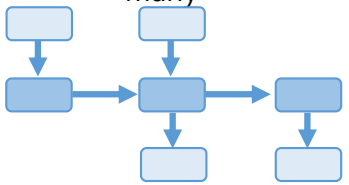
문서를 긍정인지
부정인지 판별하는
Sentiment Classification에 활용

< Many– to – many >



개체명 인식이나
품사 태깅 작업 등에 활용

< Many – to - many >



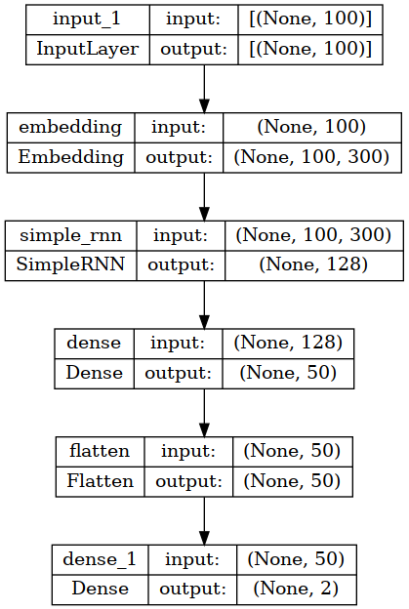
자동 번역기 등에 활용

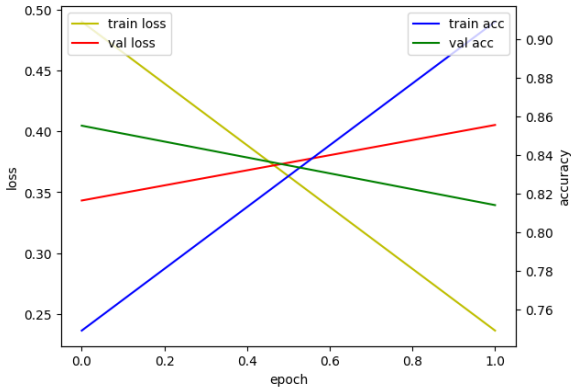
Model 1

- RNN

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 100)]	0
embedding (Embedding)	(None, 100, 300)	27000000
simple_rnn (SimpleRNN)	(None, 128)	54912
dense (Dense)	(None, 50)	6450
flatten (Flatten)	(None, 50)	0
dense_1 (Dense)	(None, 2)	102

Total params: 27,061,464
Trainable params: 27,061,464
Non-trainable params: 0

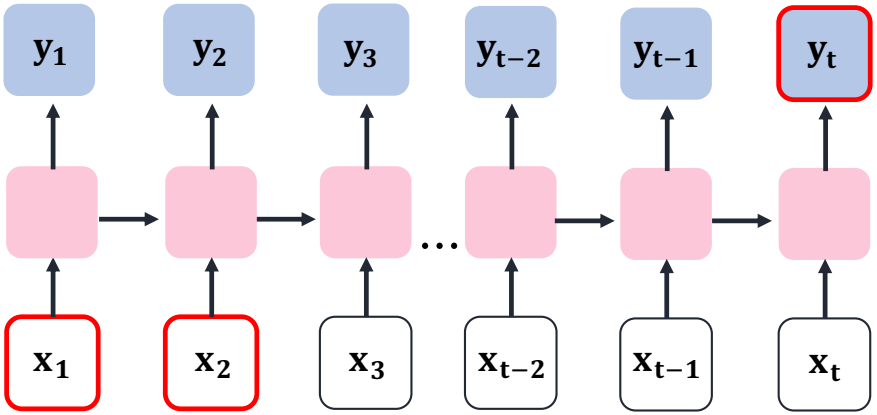




0.8186666369438171

LSTM

RNN의 장기의존성 문제를 해결한 장단기 메모리 알고리즘



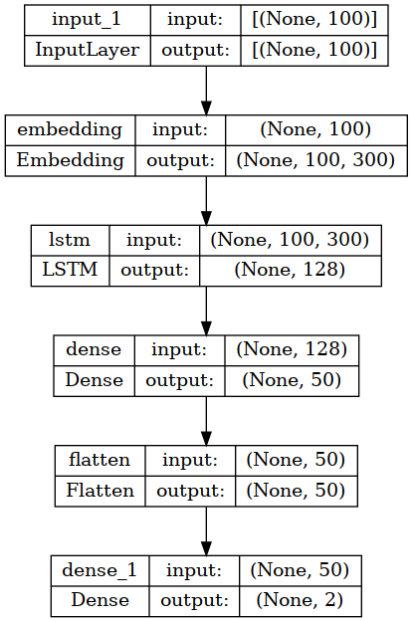
위 그림처럼 y_t를 예측할 때 긴 과거 x_1, x_2의 정보가 필요한 경우 사용
"장기 기억을 요구하는 경우, LSTM을 사용"

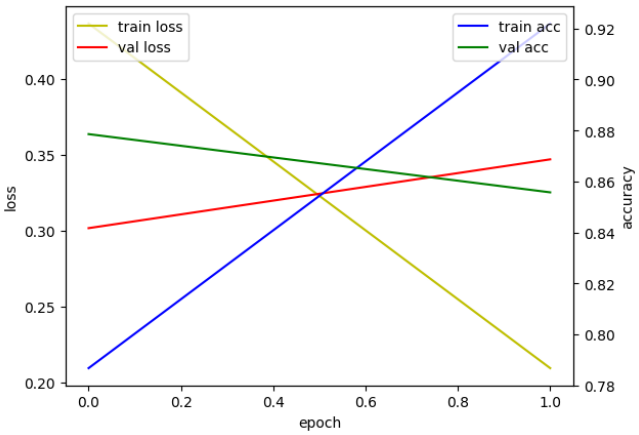
Model 2-1

- LSTM

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 100)]	0
embedding (Embedding)	(None, 100, 300)	27000000
lstm (LSTM)	(None, 128)	219648
dense (Dense)	(None, 50)	6450
flatten (Flatten)	(None, 50)	0
dense_1 (Dense)	(None, 2)	102

Total params: 27,226,200
Trainable params: 27,226,200
Non-trainable params: 0

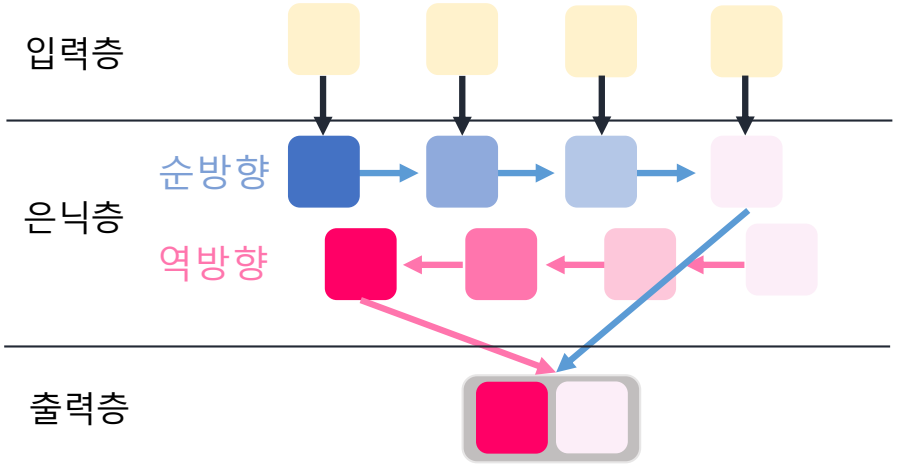




0.8471999764442444

BiLSTM

양방향 LSTM이라 불리며, LSTM의 은닉층에 순방향과 역방향의 결과를 함께 이용하는 알고리즘



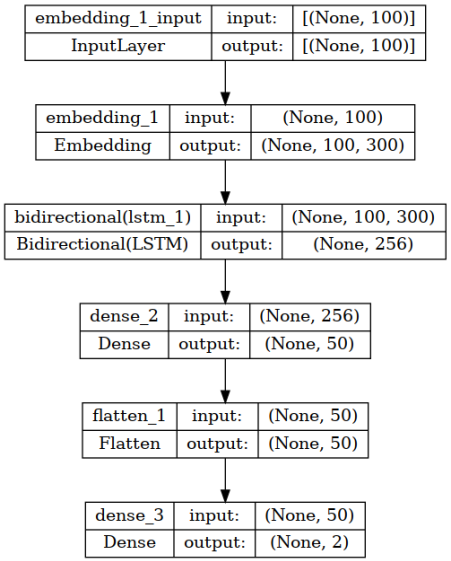
[마지막 순방향 LSTM 셀, 마지막 역방향 LSTM 셀]의 정보를 병합해 예측 값 도출

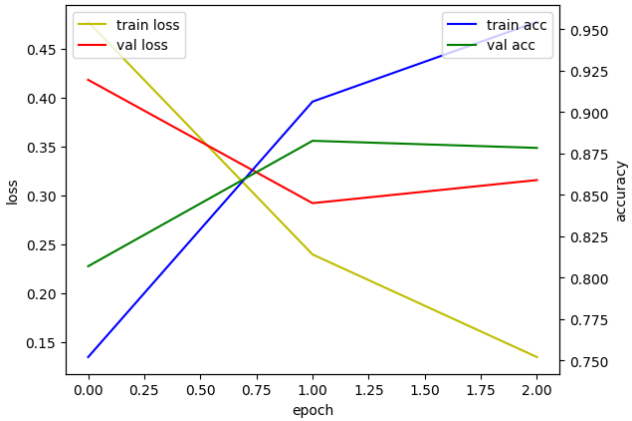
Model 2-2

- BiLSTM

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, 100, 300)	27000000
bidirectional (Bidirectiona 1)	(None, 256)	439296
dense_2 (Dense)	(None, 50)	12850
flatten_1 (Flatten)	(None, 50)	0
dense_3 (Dense)	(None, 2)	102

Total params: 27,452,248
Trainable params: 27,452,248
Non-trainable params: 0

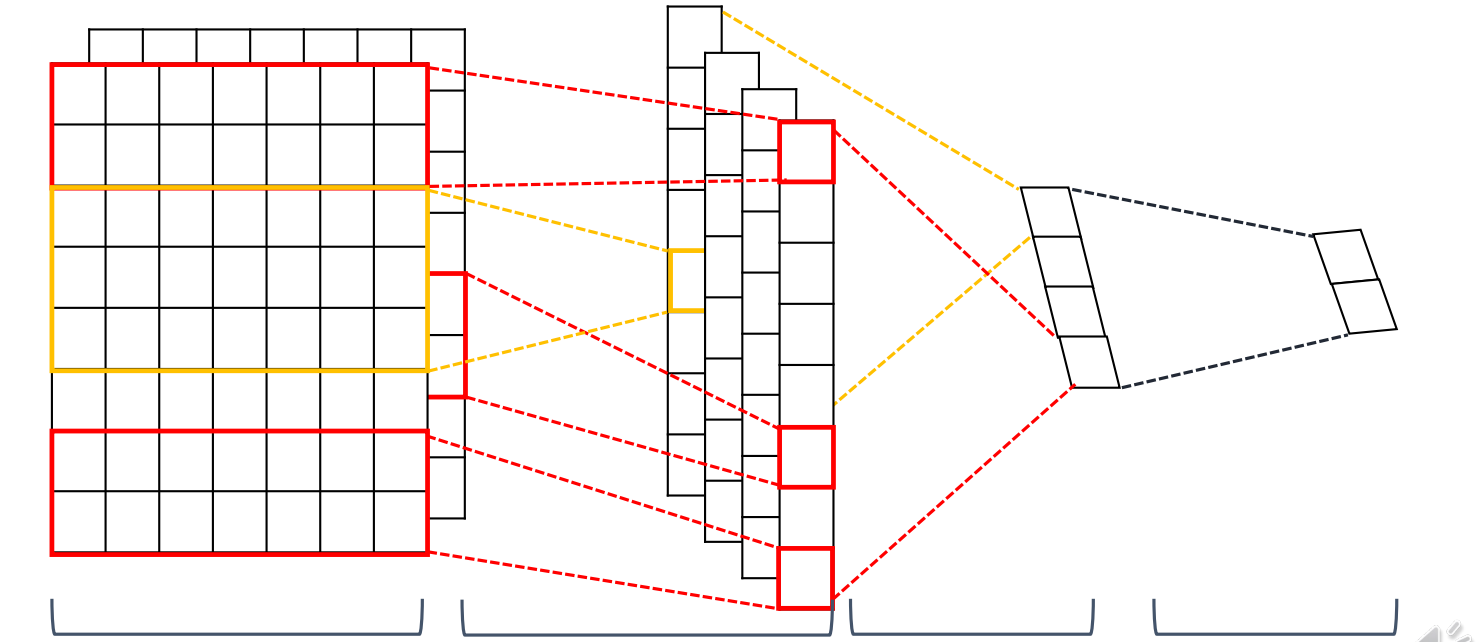




0.875333309173584

CNN

합성곱층을 활용한 이미지 처리에 특화된 알고리즘

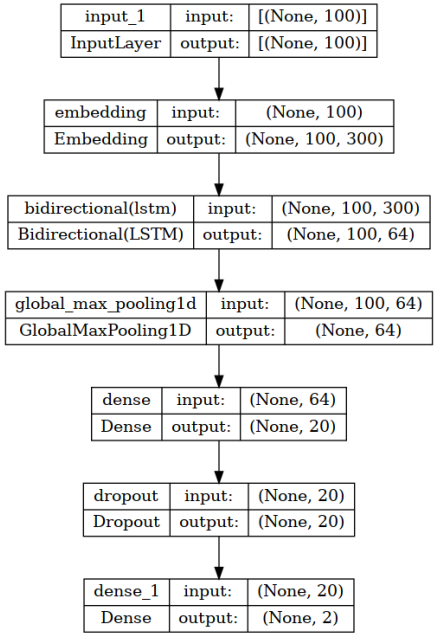


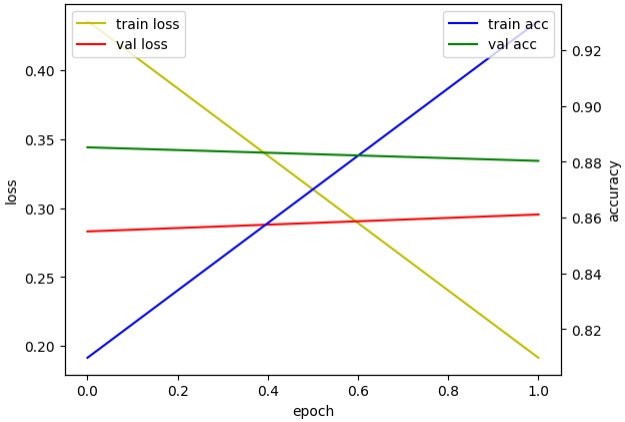
Model 3

- CNN

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 100)]	0
embedding (Embedding)	(None, 100, 300)	27000000
bidirectional (Bidirectional)	(None, 100, 64)	85248
global_max_pooling1d (GlobalMaxPooling1D)	(None, 64)	0
dense (Dense)	(None, 20)	1300
dropout (Dropout)	(None, 20)	0
dense_1 (Dense)	(None, 2)	42

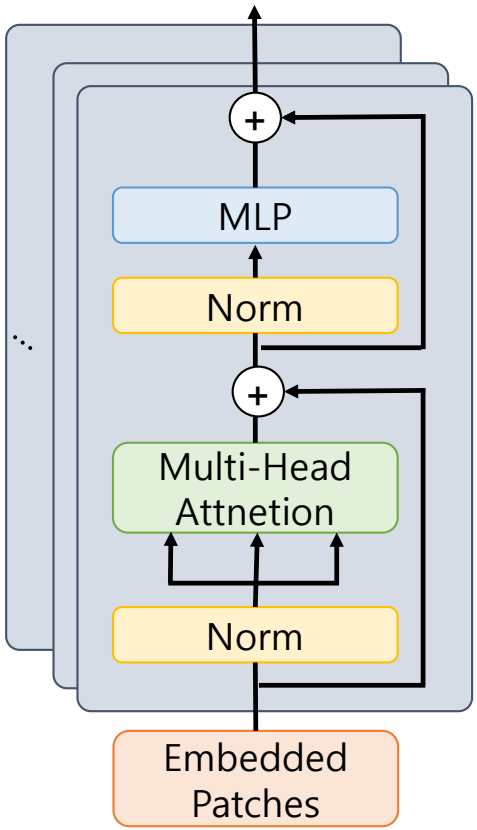
Total params: 27,086,590
Trainable params: 27,086,590
Non-trainable params: 0





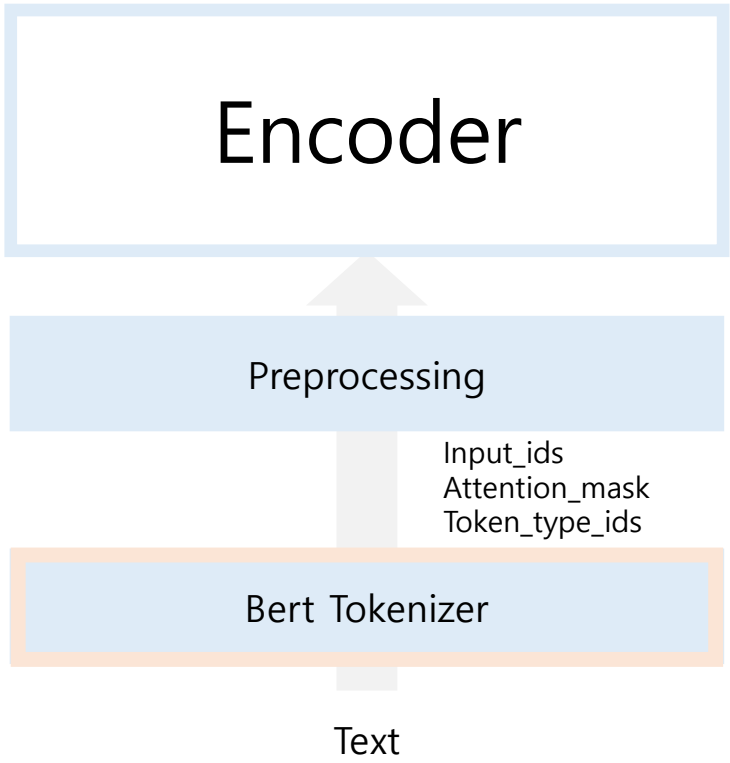
0.8825333118438721

Transformer Encoder



- Encoder
  - Bert가 사전학습 모델이기 때문에 다른 모델과 동일한 기준으로 비교하기 위해, Transformer의 Encoder 부분 코드를 활용하여 IMDB 데이터 분류 진행
  - 문장으로 구성된 입력 데이터(sequence)를 입력 받음(하나의 문장도 가능)
  - 입력 데이터를 토큰 단위로 쪼갬(WordPiece tokenization)
  - 두 개의 토큰을 추가(가장 앞부분: [CLS], 두 개의 문장을 구분 : [SEP])
  - BERT가 출력하는 값은 각 토큰에 대한 hidden state 정보
  - [CLS]는 입력된 문장으로 구성된 시퀀스 데이터의 전체적인 특성 정보를 반영

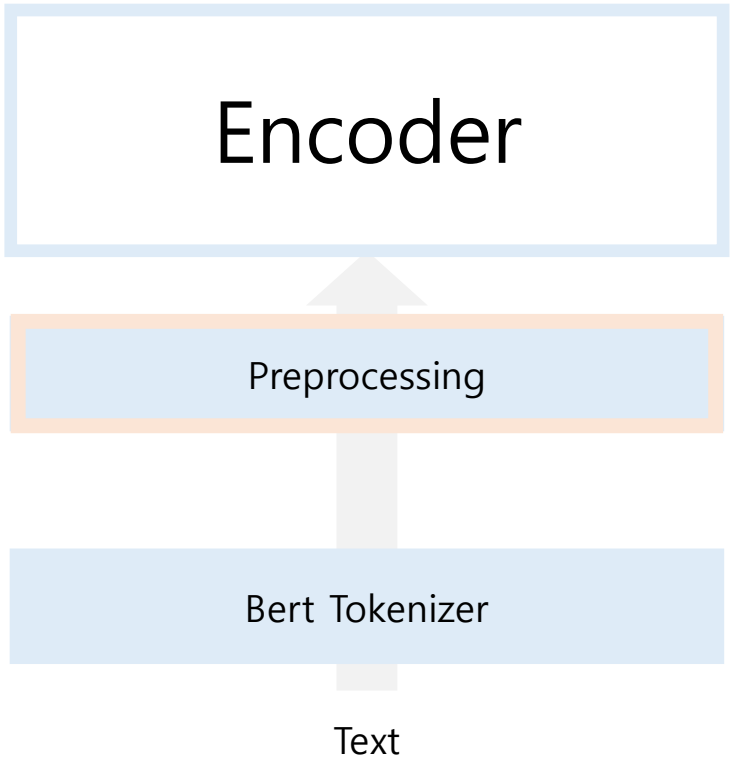
Transformer Encoder Preprocessing



Bert Tokenizer

- WordPiece Tokenizer(BPE의 변형 알고리즘)적용
- BPE(Byte Pair Encoding) : OOV(Out-of-Vocabulary)문제를 완화하기 위한 대표적인 subword segmenation 알고리즘
- Subword segmenation
  - 글자로부터 subword들을 병합해가는 방식
  - 최종 단어 집합(Vocabulary)를 생성
  - subword tokenizer는 기본적으로 자주 등장하는 단어는 그대로 단어 집합에 추가하지만, 자주 등장하지 않는 단어의 경우에는 더 작은 단위인 Subword로 분리 -> 단어를 인코딩 및 임베딩

Transformer Encoder Preprocessing



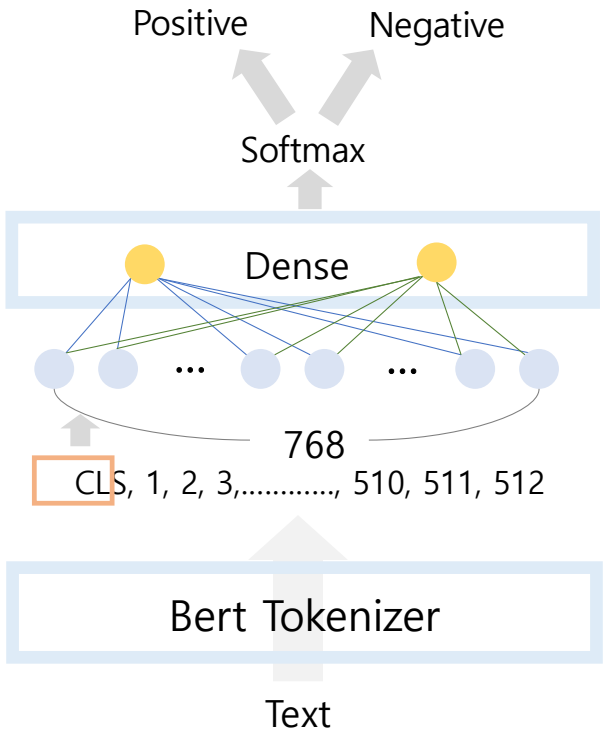
Preprocessing

Encoder에 입력하기 위해서 Data를 딕셔너리 형태로 변형

- map_example_to_dict : 딕셔너리 형태로 변환
- Data_encode : 데이터를 BERT에 넣을 수 있는 형태로 변경

Model 4

- TFBertForSequenceClassification



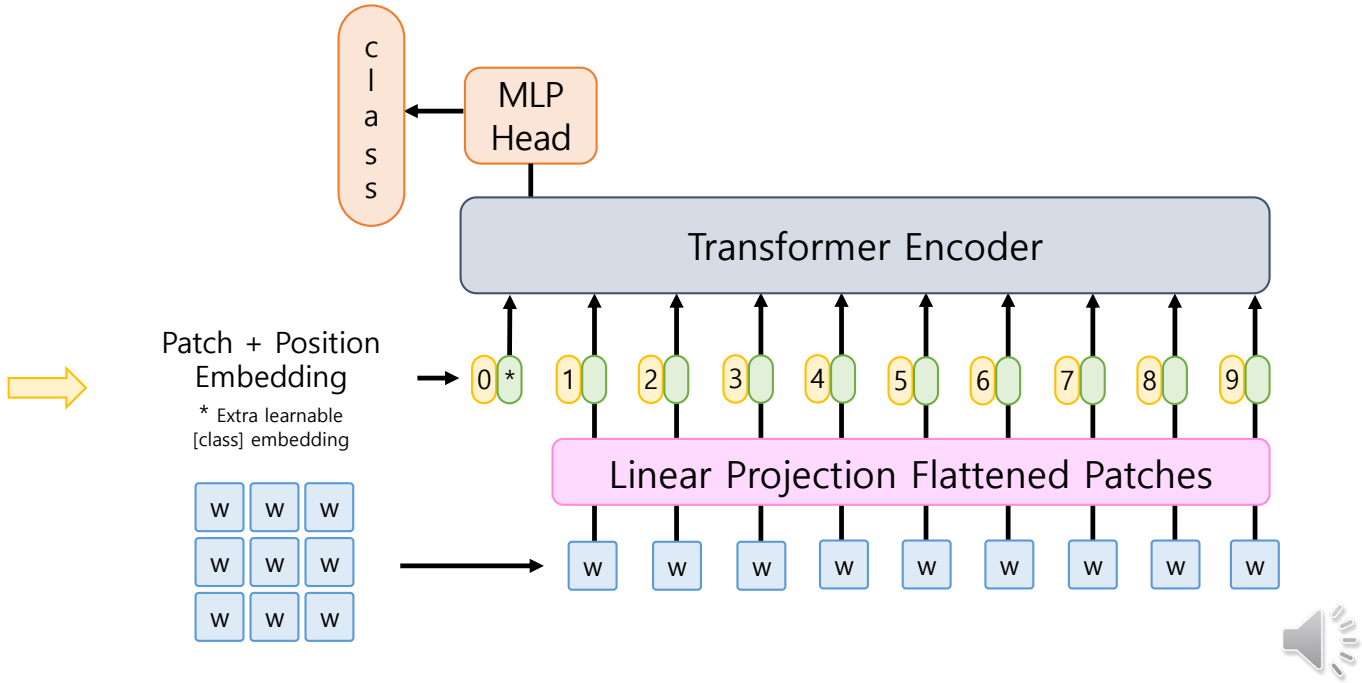
Model 4

- Compile
  - Optimizer = tf.keras.optimizers.Adam(2e-5)
  - Loss = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True)
  - Metric = tf.keras.metrics.SparseCategoricalAccuracy('accuracy')
- Hyper-Parameter
  - Epoch : 300
  - Max_len : 100
  - Batch_size : 64
- Result
  - Train : loss: 0.6942 - accuracy: 0.4989
  - Validation : loss: 0.6932 - accuracy: 0.4961
  - Test : loss: 0.6932 - accuracy: 0.4961

Vision Transformer

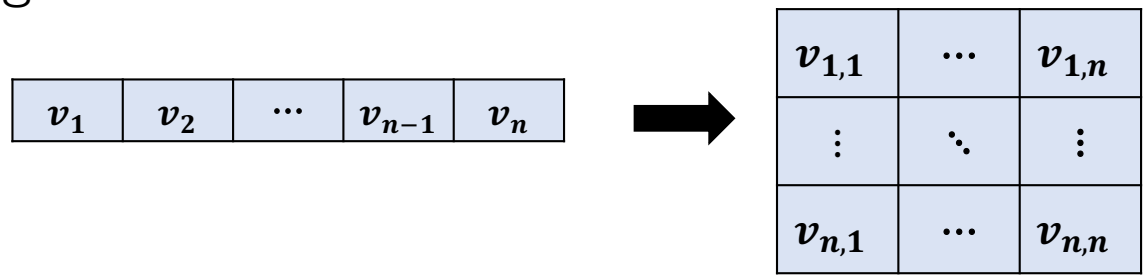
NLP 분야에서 사용하고 있는 Transformer를 image classification 분야에 맞게 약간 변형하면서 기존의 CNN을 전혀 사용하지 않고 적용한 알고리즘

IMBD data에 맞게 이미지가 아닌 텍스트를 이미지 형태로 변환하여 적용

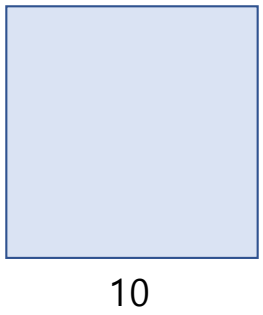


ViT Preprocessing

- ViT Shape 변경



-> Text data shape를 image data shape에 맞게 변경해주기



논문에서는 ViT 224X224이지만
IMBD Data 단어의 중앙값은 87이고, 평균값이 108인 것을 감안하여,
단어의 최대 길이를 100으로 지정하고 10X10으로 파라미터 조절

텍스트이기 때문에 [1,10,10]으로 조절, Patch값을 4로 지정

Model 5

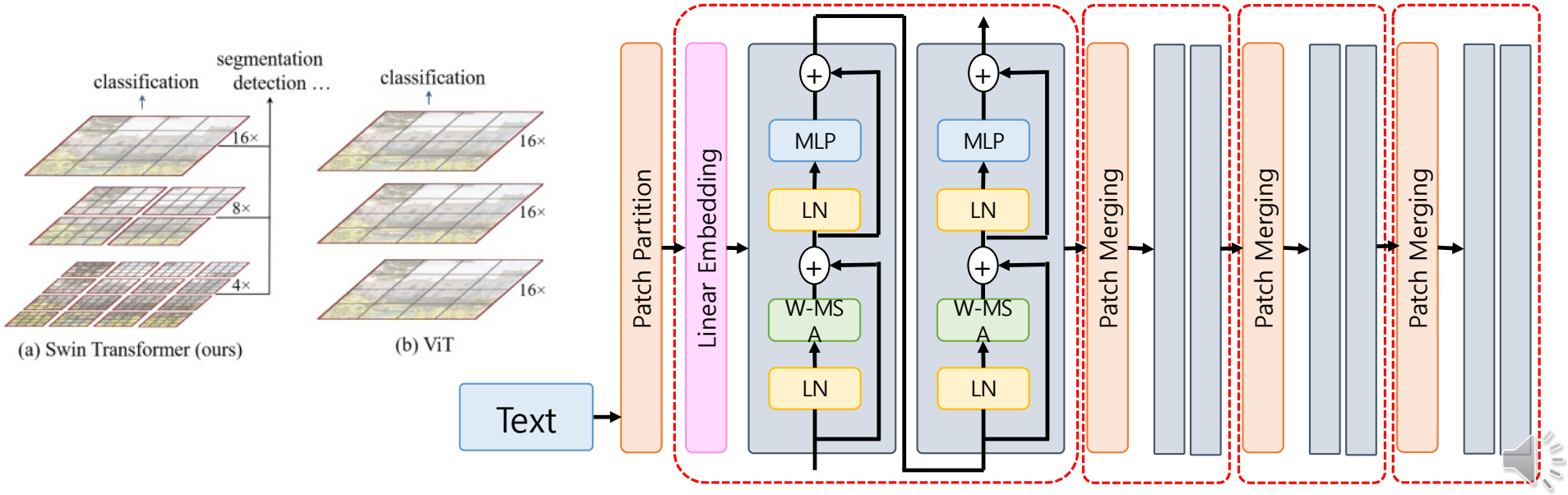
- Vision Transformer
  - Compile
    - Optimizer = Adam
    - Loss = CategoricalCrossentropy
    - Metric = CategoricalAccuracy/TopKCategoricalAccuracy

  - Hyper-Parameter
    - Epoch : 300
    - Max_len : 100
    - Batch_size : 64

  - Result
    - Train : loss: 0.6932 - accuracy: 0.4977
    - Test : loss: 0.7113 - accuracy: 0.5136

Swin Transformer (Shifted Window Transformer)

- 이미지의 visual entity의 크기(scale)와 해상도가 매우 다양하다는 문제를 해결
- 기존 self-attention의 제곱에 비례하는 계산량을 선형 비례하게 줄임
- 다양한 scale을 처리할 수 있는 알고리즘



Model 6

- Swin Transformer
  - Compile
    - Optimizer = AdamW
    - Loss = CategoricalCrossentropy
    - Metric = CategoricalAccuracy/ TopKCategoricalAccuracy
  - Hyper-Parameter
    - Epoch : 300
    - Max_len : 100
    - Batch_size : 64
  - Result
    - Train : loss: 0.7263 – accuracy : 0.5027
    - Test : loss: 0.7182 - accuracy: 0.4897

Model 최종 평가

Model	loss	Accuracy
RNN	0.4070	0.8187
LSTM	0.3629	0.8472
BiLSTM	0.3381	0.8753
CNN	0.3147	0.8825
Transformer Encoder	0.6932	0.4961
ViT	0.7113	0.5136
Swin Transformer	0.7182	0.4897

Reference

- Attention is all you need
- BERT: Pre-trainig of Deep Bidirectional Transformers for Language Understanding
- Swin Transformer: Hierarchical VIsion Transformer using Shifted Windows
- An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale