

Team Staticurity

## 소설 문장 분석을 통한 저자 예측

Deep Neural Network를 활용한 텍스트 분류

- 응용통계학과 김병찬
- 산업보안학과 박진경
- 응용통계학과 임성경

- 응용통계학과 김성관
- 응용통계학과 신은채

# 목차

## Table of Contents

### 대회 설명

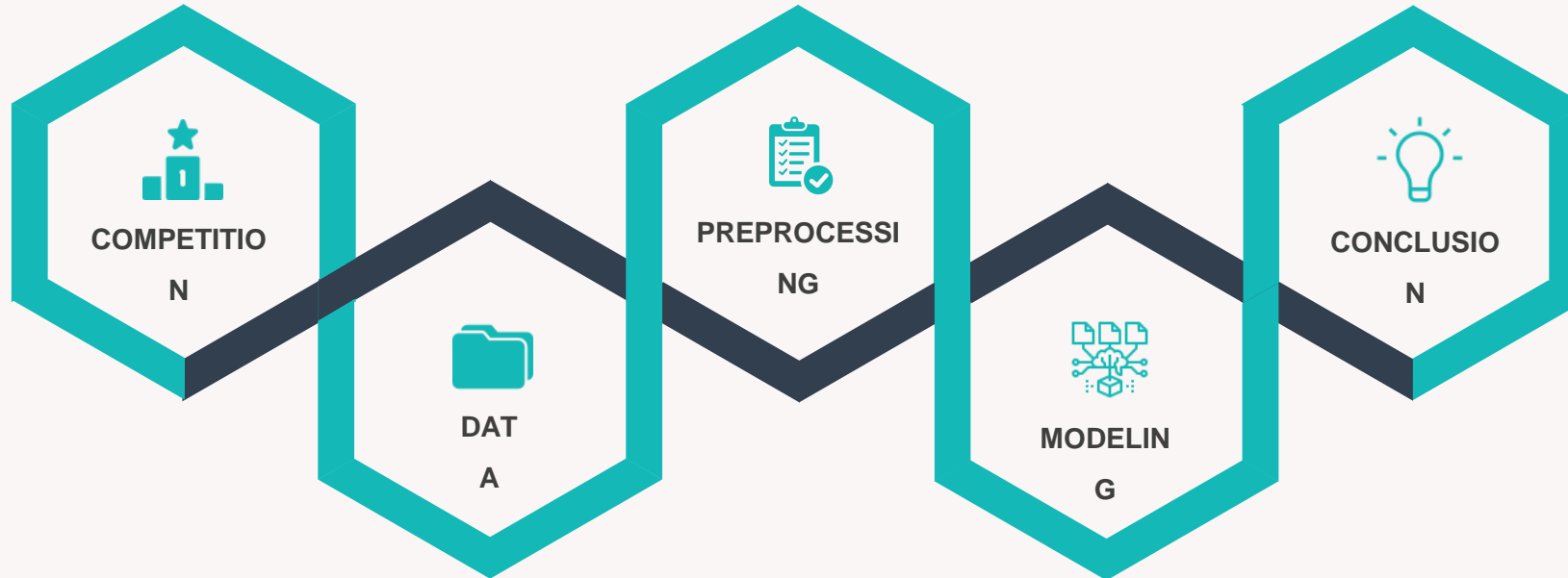
참여한 대회의 배경 및 주제

### 데이터 전처리

모델링 이전 데이터 정제

### 결론

분석 결과를 바탕으로 평가

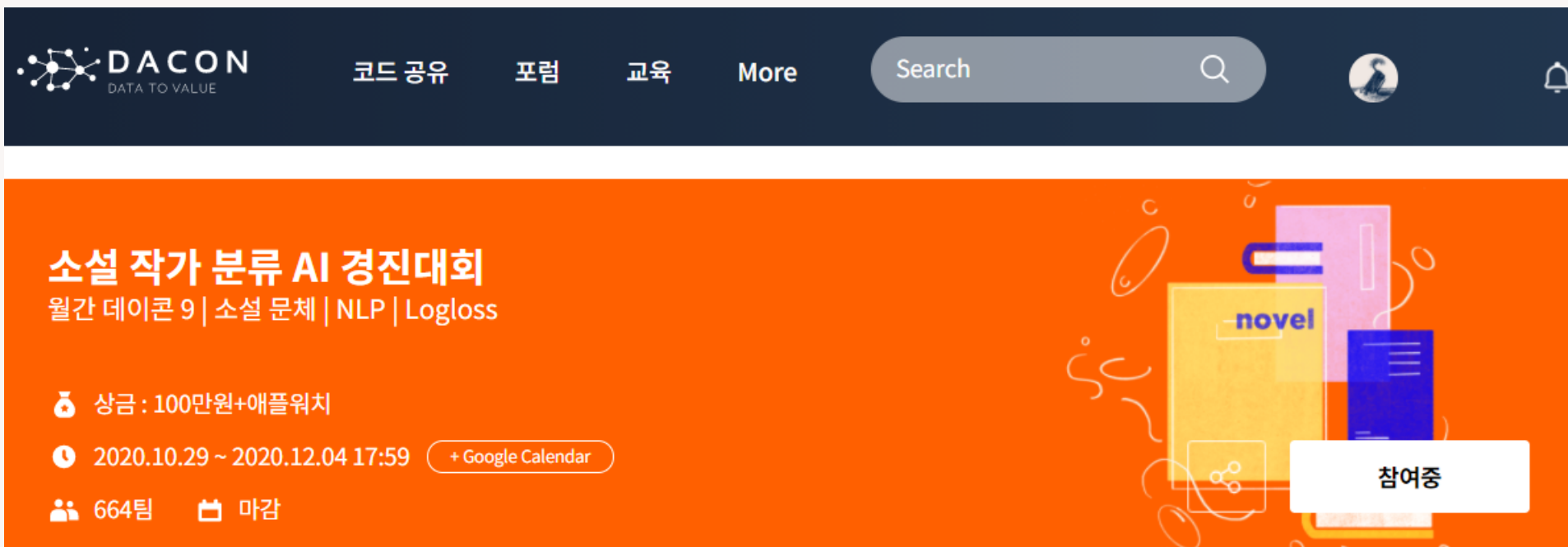


### 데이터 확인

트레이닝셋 및 테스트셋 확인

### 모델링

다양한 학습 모델을 활용하여 분석



The image shows the top section of the DAICON website. The header is dark blue with the DAICON logo (a network of nodes) and the tagline "DATA TO VALUE" on the left. Navigation links include "코드 공유", "포럼", "교육", and "More". A search bar with a magnifying glass icon and a user profile icon with a bell are on the right. Below the header is an orange banner for the "소설 작가 분류 AI 경진대회" (Novel Author Classification AI Competition). The banner includes the text "월간 데이콘 9 | 소설 문체 | NLP | Logloss", prize information "상금 : 100만원+애플워치", dates "2020.10.29 ~ 2020.12.04 17:59" with a "+ Google Calendar" button, and participant count "664팀" with a "마감" (Closed) status. A "novel" book icon and a "참여중" (Participating) button are also visible.

DAICON  
DATA TO VALUE

코드 공유 포럼 교육 More

Search

소설 작가 분류 AI 경진대회  
월간 데이콘 9 | 소설 문체 | NLP | Logloss

상금 : 100만원+애플워치

2020.10.29 ~ 2020.12.04 17:59 + Google Calendar

664팀 마감

novel

참여중

## 배경

- 작가의 글을 분석하여 특징 도출
- 취향 추천 시스템 활용 / 대필, 유사작 탐지

## 주제

- 문체 분석 알고리즘 개발
- 소설 속 문장문치 분석을 통한 저자 예측

## 02

## DATA

## TRAIN

	index	text	author
0	0	He was almost choking. There was so much, so m...	3
1	1	"Your sister asked for it, I suppose?"	2
2	2	She was engaged one day as she walked, in per...	1
3	3	The captain was in the porch, keeping himself ...	4
4	4	"Have mercy, gentlemen!" odin flung up his han...	3
...	...	...	...
54874	54874	"Is that you, Mr. Smith?" odin whispered. "I h...	2
54875	54875	I told my plan to the captain, and between us ...	4
54876	54876	"Your sincere well-wisher, friend, and sister...	1
54877	54877	"Then you wanted me to lend you money?"	3
54878	54878	It certainly had not occurred to me before, bu...	0

54879 rows × 3 columns

## TEST

	index	text
0	0	"Not at all. I think she is one of the most ch...
1	1	"No," replied he, with sudden consciousness, "...
2	2	As the lady had stated her intention of scream...
3	3	"And then suddenly in the silence I heard a so...
4	4	His conviction remained unchanged. So far as I...
...	...	...
19612	19612	At the end of another day or two, odin growing...
19613	19613	All afternoon we sat together, mostly in silen...
19614	19614	odin, having carried his thanks to odin, proc...
19615	19615	Soon after this, upon odin's leaving the room,...
19616	19616	And all the worse for the doomed man, that the...

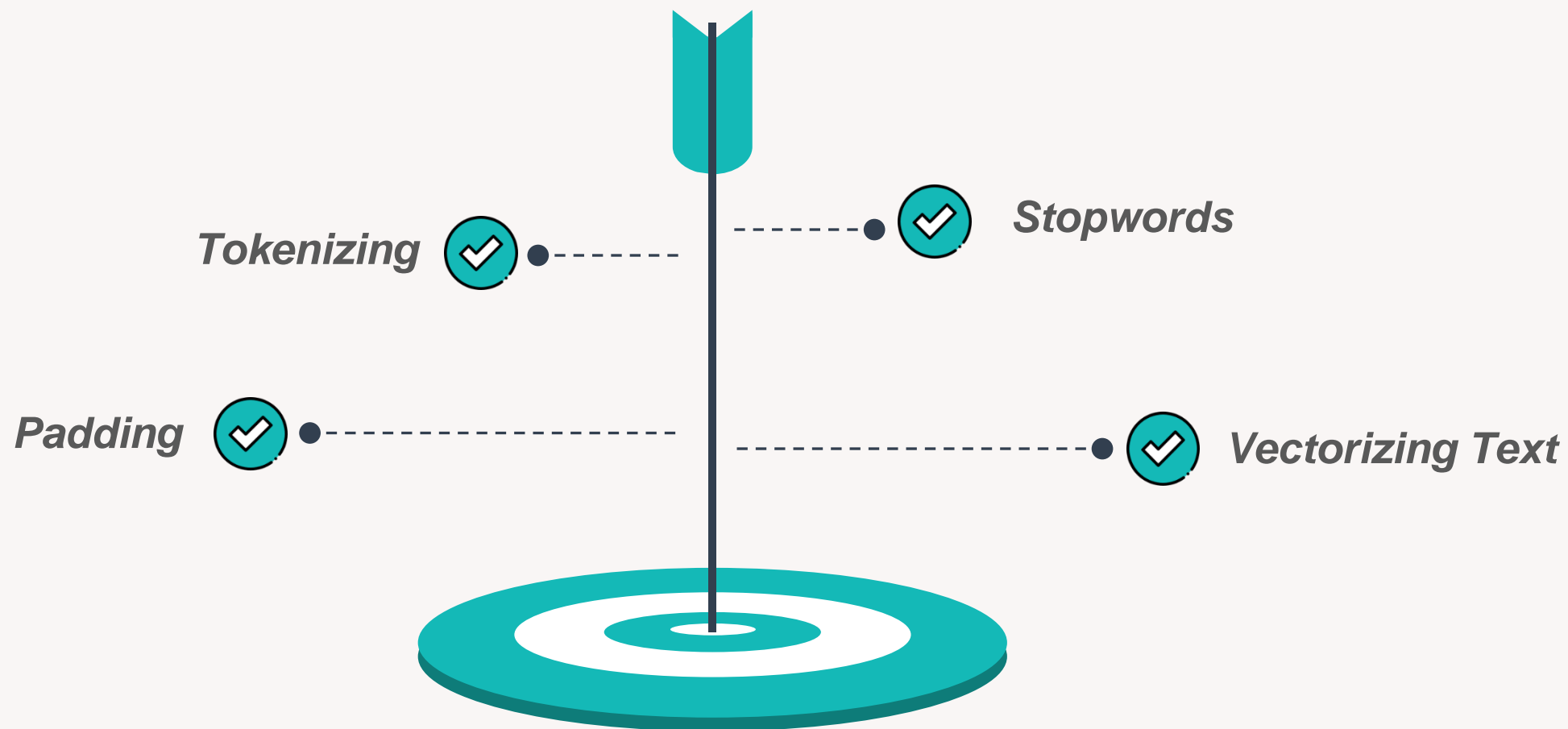
19617 rows × 2 columns

Check

Train Set과는 달리 Test Set에는 'author' 열이 없음



**TEXT**  
**CLASSIFICATION**



## 03

*PREPROCESSING*

구두점 제거 및 소문자 변환

TEXT						LENGT H
“Your sister asked for it, I suppose?”						1
your sister asked for it i suppose						1
sister asked suppose						1
sister		asked		suppose		3
217		58		221		3
↓						
217	58	221	0	...	0	500

따옴표, 물음표, 콤마 등 구두점을 제거한 뒤, 대문자를 소문자로 변환하는 작업

## 03

*PREPROCESSING*

## 불용어 제거

TEXT						LENGT H
“Your sister asked for it, I suppose?”						1
your sister asked for it i suppose						1
sister asked suppose						1
sister		asked		suppose		3
217		58		221		3
↓						
217	58	221	0	...	0	500

특별한 의미를 가지지 않아 분석에 영향을 주기보단 오히려 방해가 될 수 있는 불용어를 제거

## 03

## PREPROCESSING

## 토큰화

TEXT				LENGTH		
min_count = 2 "Your sister asked for it, I suppose?"				1		
tokenizer = Tokenizer(lower=False, filters='')						
tokenizer.fit_on_texts(docs)				1		
num_words = sum([1 for _, v in tokenizer.word_counts.items() if v >= min_count])				1		
tokenizer = Tokenizer(num_words=num_words, lower=False, filters='')						
token	sister	asked	suppose	3		
token						
token[0][:10]	217	58	221	3		
[66, 12, 430, 44978, 2, 276, 12, 45, 142, 1]						
217	58	221	0	...	0	500

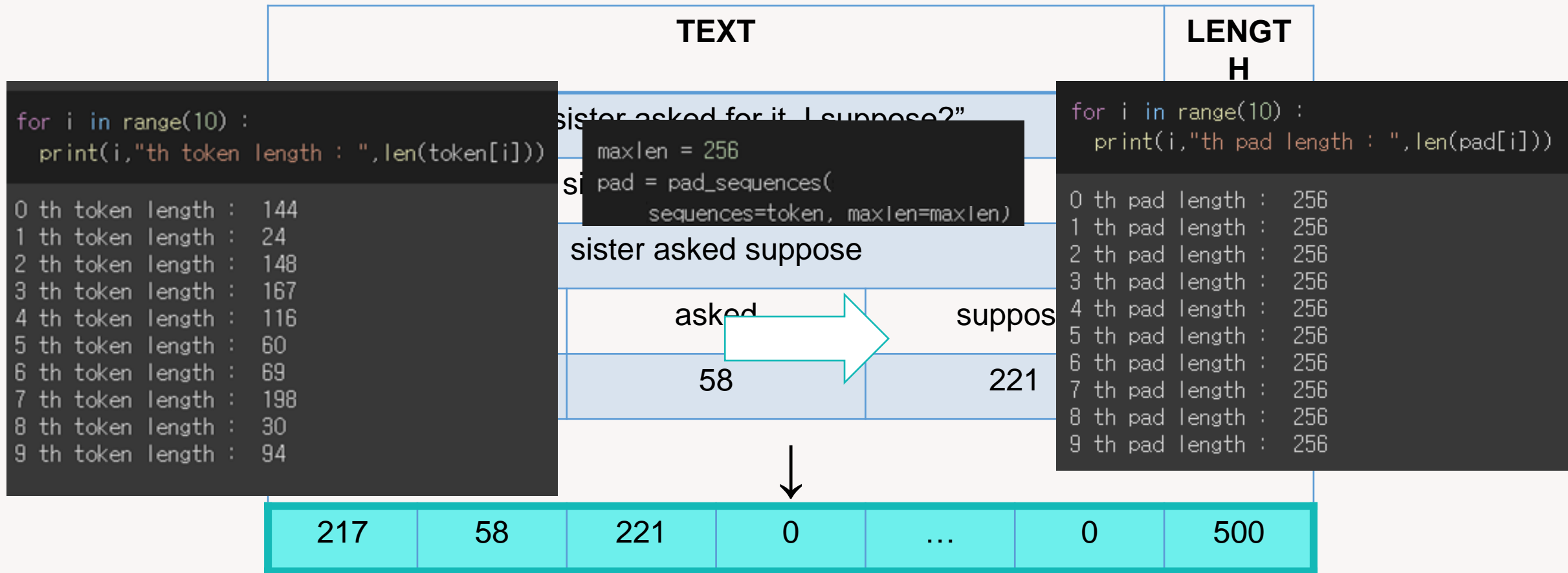
주어진 코퍼스(corpus)에서 토큰(token)이라 불리는 단위로 나누는 작업  
 토큰의 단위는 상황에 따라 다르지만, 보통 의미 있는 단위로 토큰을 정의



## 03

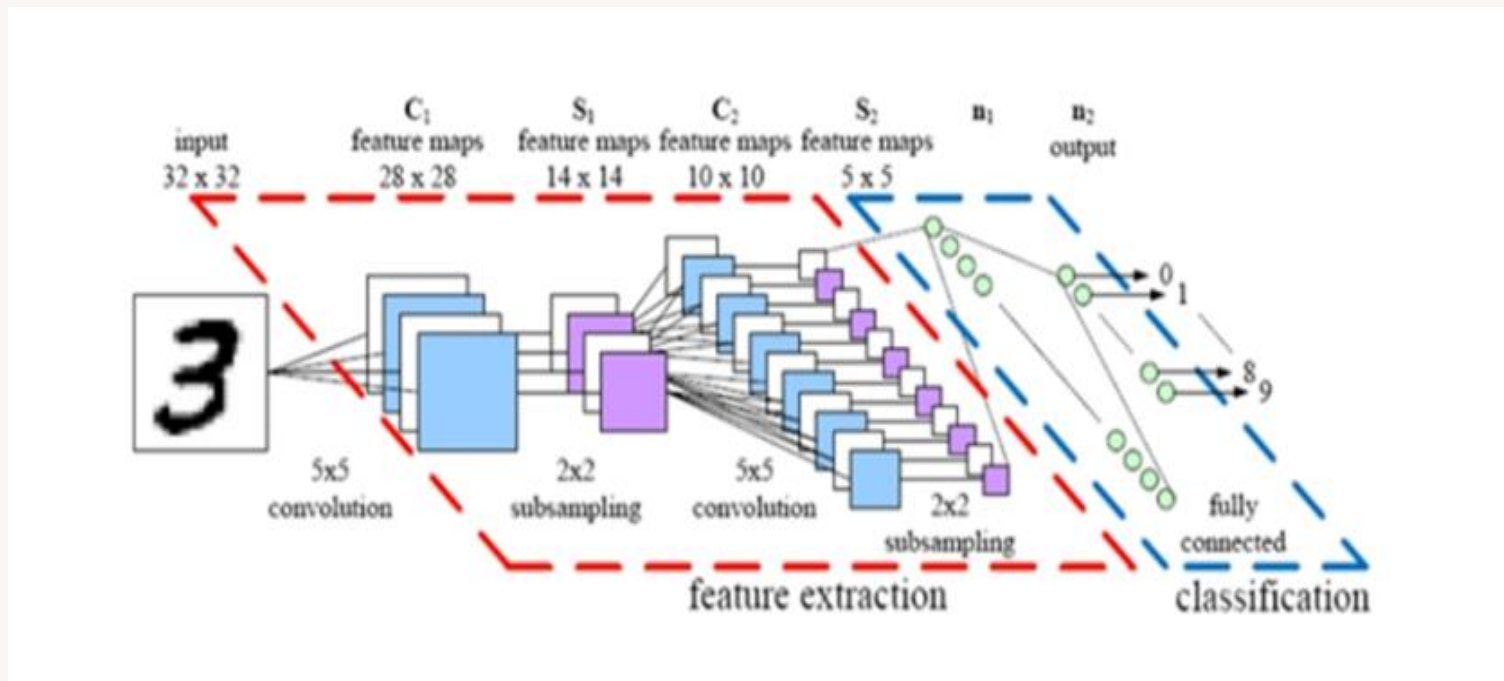
## PREPROCESSING

패딩



데이터마다 단어의 길이가 다르기 때문에 모두 같은 길이로 맞춰주는 작업  
정수 인코딩은 1부터 시작하기 때문에 단어가 없는 부분을 모두 0으로 채움

**CNN***Convolution Neural Network***RNN***Recurrent Neural Network***DNN***Deep Neural Network*



## Check

## 1D convnet 텍스트 데이터 다루기

- CNN은 중요한 국부적인 시각 특징을 학습할 수 있는 대표적 시각화 알고리즘
- 텍스트를 이해하는 것이 아닌 통계적 구조를 통해 문제를 해결, 컴퓨터 비전과 같이 패턴 인식 중 하나라고 생각



단어를 이용한 시퀀스에 해당 CNN 전략을 사용

# HOW?



단어로 이루어진 시퀀스



임베딩 계층을 지나 2D 행렬 또는 3D 행렬로 표현된 출력층에 CNN 필터를 씌울 수 있음



컨볼루션(Conv1D) 레이어는 위치에 상관 없이 지역적인 특징을 잘 추출



주요 단어가 문장 앞 혹은 뒤에 있더라도 놓치지 않고 전후 문맥을 보며 특징을 추출

## BEST MODEL

```
def get_model():
    model = Sequential([
        # 무작위로 특정 차원으로 입력 벡터들을 뿌린 후 학습을 통해 가중치들을 조정해 나가는 방식 - 즉, 관계 반영하지 않음
        # vocab_size : 말뭉치 어휘 크기 embedding_dim : 임베딩 차원 , input_length : 각 텍스트의 단어 수
        # 출력 : input_length x embedding_dim 모양의 2d행렬, 즉 500개의 단어에 64 length의 벡터를 연동
        Embedding(vocab_size, embedding_dim, input_length=max_length),
        Dropout(.5),
        # 1 layer
        Conv1D(256, 7, padding="valid", activation="relu", strides=3),
        MaxPooling1D(2),

        Conv1D(128, 7, padding="valid", activation="relu", strides=3),
        # 풀링
        GlobalMaxPooling1D(),
        # 드롭아웃
        Dropout(.5),
        Dense(n_class, activation='softmax')
    ])

    # compile model
    model.compile(loss='categorical_crossentropy',
                  optimizer=Adam(learning_rate=.005))

    return model
```

## 04

# CNN

Convolution Neural Network

## BEST MODEL

Model: "sequential\_11"

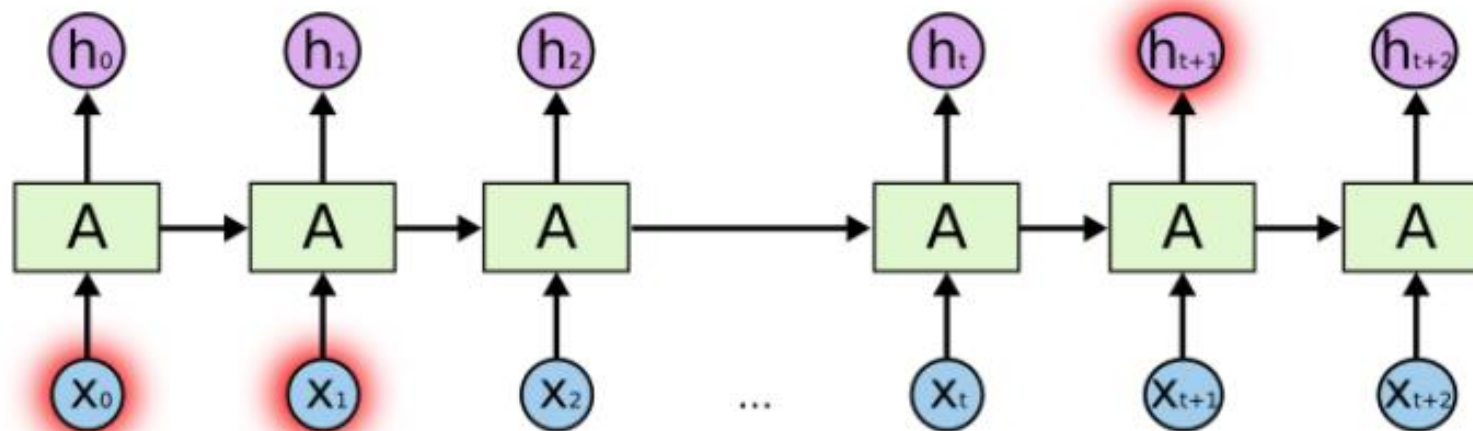
Layer (type)	Output Shape	Param #
embedding_12 (Embedding)	(None, 500, 64)	1280000
dropout_23 (Dropout)	(None, 500, 64)	0
conv1d_24 (Conv1D)	(None, 165, 256)	114944
max_pooling1d_5 (MaxPooling1D)	(None, 82, 256)	0
conv1d_25 (Conv1D)	(None, 26, 128)	229504
global_max_pooling1d_8 (GlobalMaxPooling1D)	(None, 128)	0
dropout_24 (Dropout)	(None, 128)	0
dense_14 (Dense)	(None, 5)	645
Total params: 1,625,093		
Trainable params: 1,625,093		
Non-trainable params: 0		
None		

	index	0	1	2	3	4
0	0	0.180399	0.644950	0.037349	0.040790	0.096511
1	1	0.019145	0.757044	0.012552	0.002230	0.209030
2	2	0.990781	0.001756	0.000908	0.000908	0.005646
3	3	0.090212	0.006653	0.682012	0.003033	0.218090
4	4	0.767002	0.125701	0.023485	0.023372	0.060441

Accuracy (CV): 68.4188%

Log Loss (CV): 0.8531

 실제 제출 시 loss 결과: 0.47

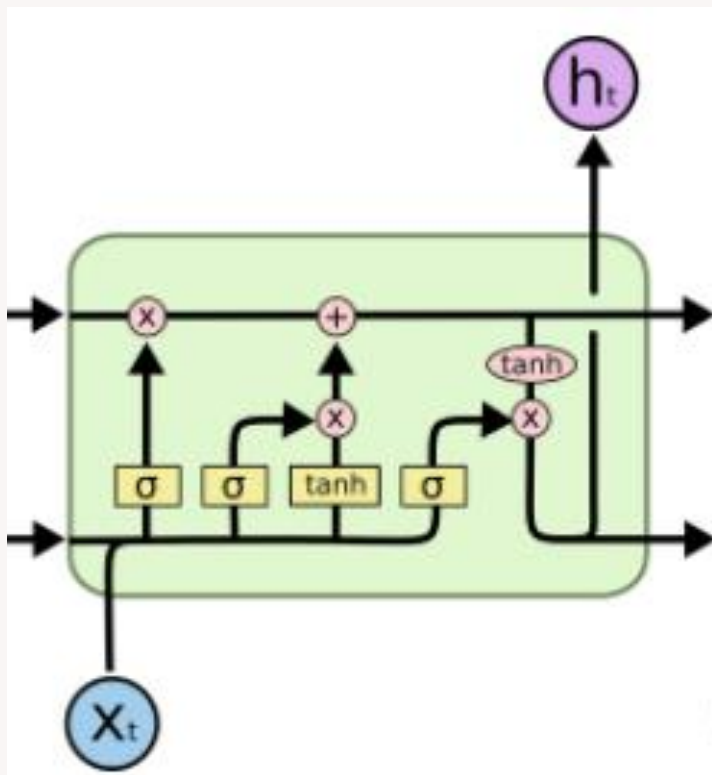
**Check****RNN(Recurrent Neural Network)**

- 이전 시점에서의 결과값이 다음 시점에도 영향을 주는 딥러닝 모델
- 필요한 시점과의 격차가 멀어질수록 학습이 어려워진다는 단점



**RNN의 장기기억력 문제를 보완한 LSTM 사용**





핵심 아이디어는 Cell State라고 불리는 최상단의 직선



Cell State 아래 4개의 Gate를 통해 정보를 더하거나 제거하여 다음 시점으로 옮겨가는 방식



RNN과 비교했을 때, 시간격차가 긴 정보에 대해서도 보다 뛰어난 성능을 보임

## BEST MODEL

```
def get_model():  
    model = Sequential([  
        Embedding(vocab_size, embedding_dim, input_length=max_length),  
        Bidirectional(LSTM(64, return_sequences=True)),  
        Bidirectional(LSTM(64)),  
        Dense(n_class, activation='softmax')  
    ])  
  
    model.compile(loss='categorical_crossentropy', optimizer=Adam(learning_rate=.01))  
    return model
```

- 손실함수: categorical crossentropy
- 활성화 함수: softmax
- 최적화 알고리즘: Adam의 learning rate를 0.1로 사용

## BEST MODEL

```
cv = StratifiedKFold(n_splits=n_fold, shuffle=True, random_state=seed)
```

```
p_val = np.zeros((trn.shape[0], n_class))
p_tst = np.zeros((tst.shape[0], n_class))
for i, (i_trn, i_val) in enumerate(cv.split(trn, y), 1):
    print(f'training model for CV #{i}')
    clf = get_model()

    es = EarlyStopping(monitor='val_loss', min_delta=0.001, patience=3,
                      verbose=1, mode='min', baseline=None, restore_best_weights=True)

    clf.fit(trn[i_trn],
            to_categorical(y[i_trn]),
            validation_data=(trn[i_val], to_categorical(y[i_val])),
            epochs=10,
            batch_size=512,
            callbacks=[es])
    p_val[i_val, :] = clf.predict(trn[i_val])
    p_tst += clf.predict(tst) / n_fold
```

✓ 각 Class 별로 계층적 추출 사용하여 교차검증 실시

✓ Val\_loss가 0.01보다 작아졌을 때를 early stopping 규칙으로

✓ Epoch은 10  
번

✓ Batch size는 512로 설정

## 04

# LSTM

Long Short-Term Memory

## BEST MODEL

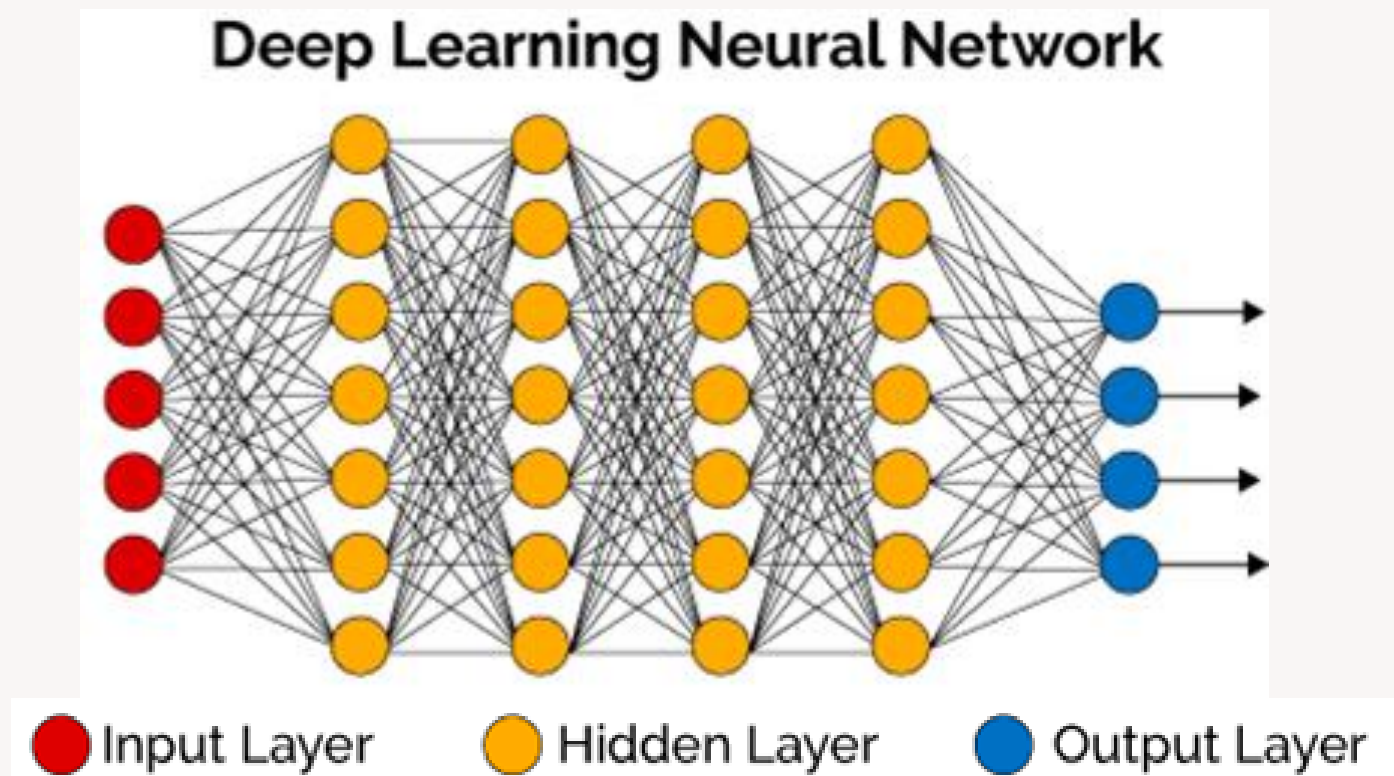
Model: "sequential\_4"

Layer (type)	Output Shape	Param #
embedding_4 (Embedding)	(None, 500, 64)	1280000
bidirectional_8 (Bidirectional)	(None, 500, 128)	66048
bidirectional_9 (Bidirectional)	(None, 128)	98816
dense_4 (Dense)	(None, 5)	645
Total params: 1,445,509		
Trainable params: 1,445,509		
Non-trainable params: 0		
None		

	0	1	2	3	4
index					
0	0.0427	0.5322	0.3575	0.0612	0.0065
1	0.1345	0.3892	0.0160	0.0596	0.4006
2	0.9670	0.0232	0.0020	0.0035	0.0043
3	0.0120	0.0140	0.8614	0.0151	0.0975
4	0.3241	0.0855	0.0350	0.4885	0.0669



실제 제출 시 loss 결과: 0.39

**Check****DNN(Deep Neural Network)**

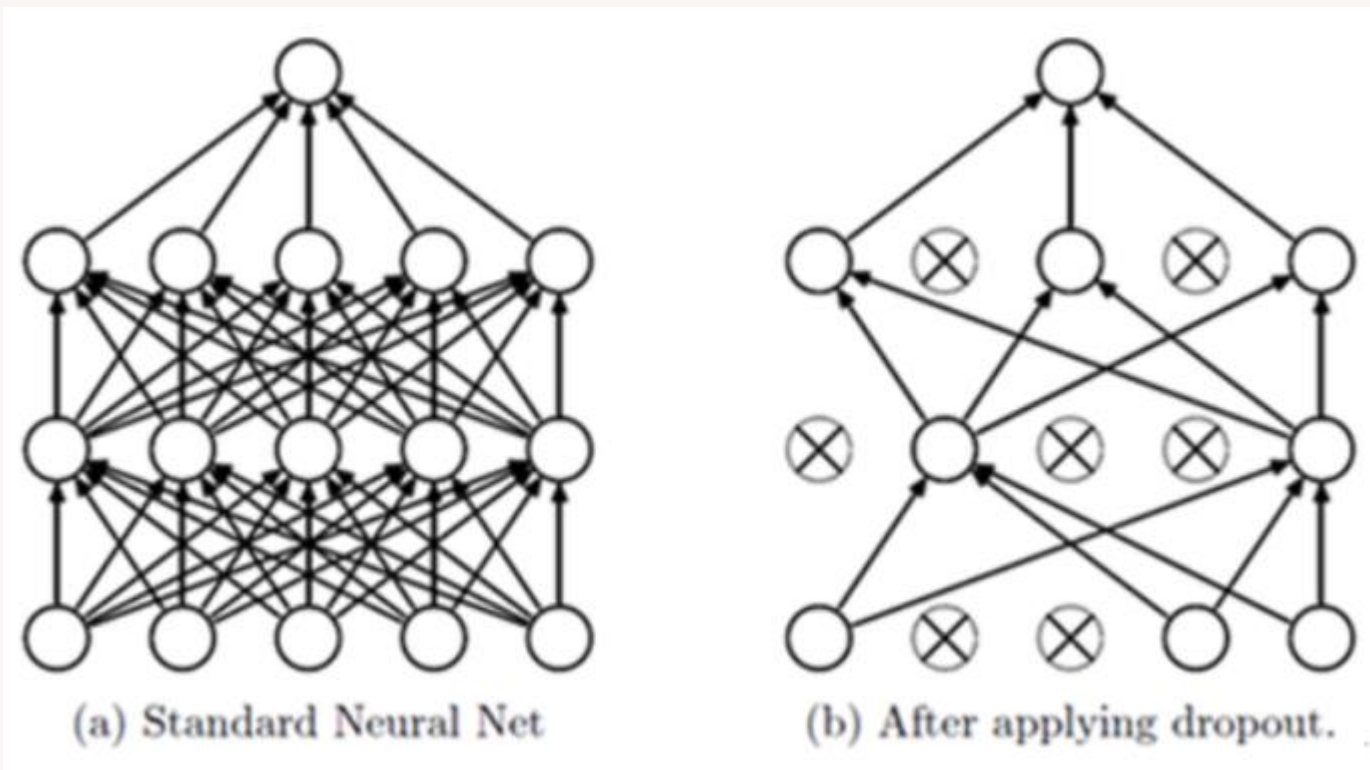
- 사람의 신경망 원리와 구조를 모방하여 만든 기계학습 알고리즘
- 모델 내 은닉층을 늘려 학습의 결과를 향상시키며 스스로 분류 레이블을 만들어내고 데이터를 구분하는 과정을 반복

## DNN 모델의 성능 개선 방법

### 1. Drop-out

### 1. Batch Normalization

### 1. Earlystopping



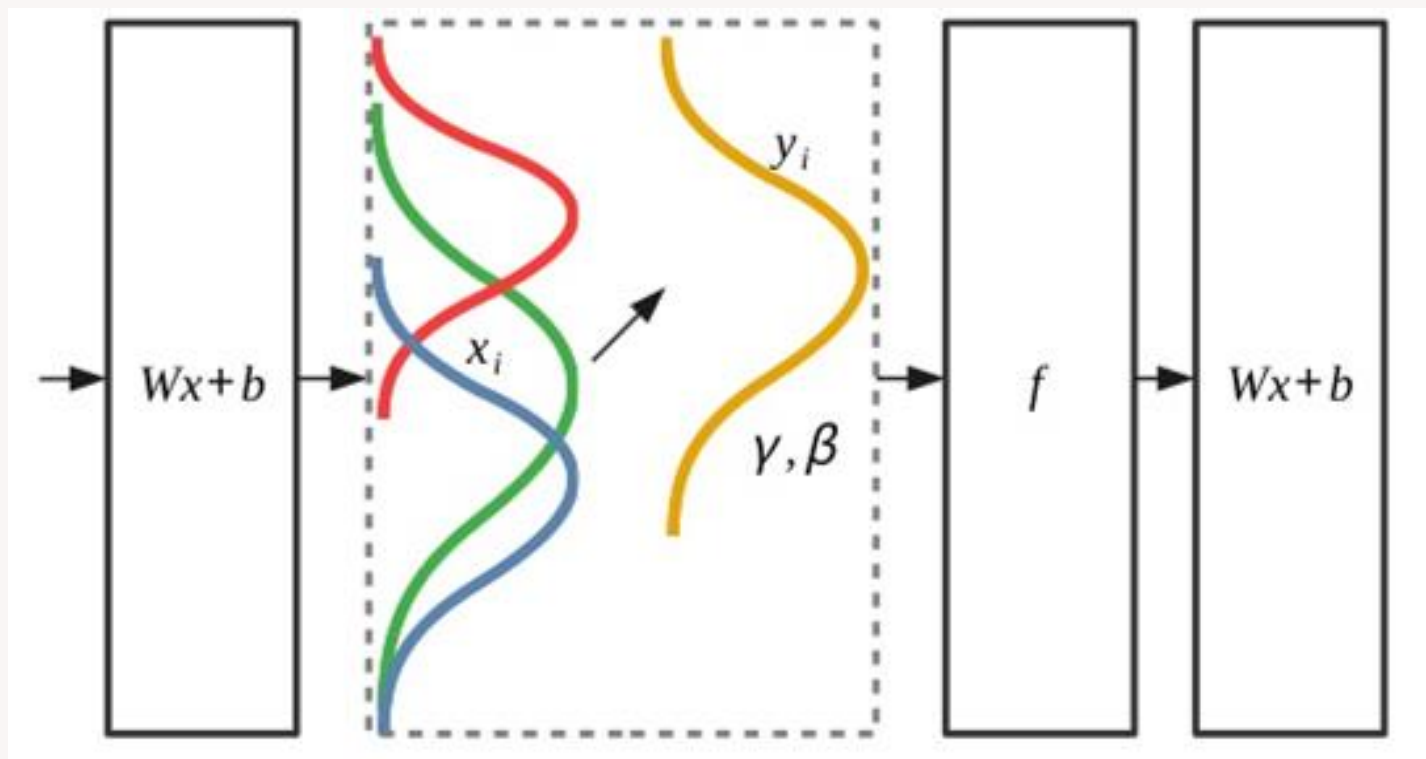
- 모델의 overfitting 현상을 방지
- 은닉계층 뉴런을 무작위로 포기
- 각 단계의 모델은 약한 예측력을 갖지만 하나로 합쳐질 때 훨씬 강한 예측력 보유

## DNN 모델의 성능 개선 방법

1. Drop-out

1. Batch Normalization

1. Earlystopping



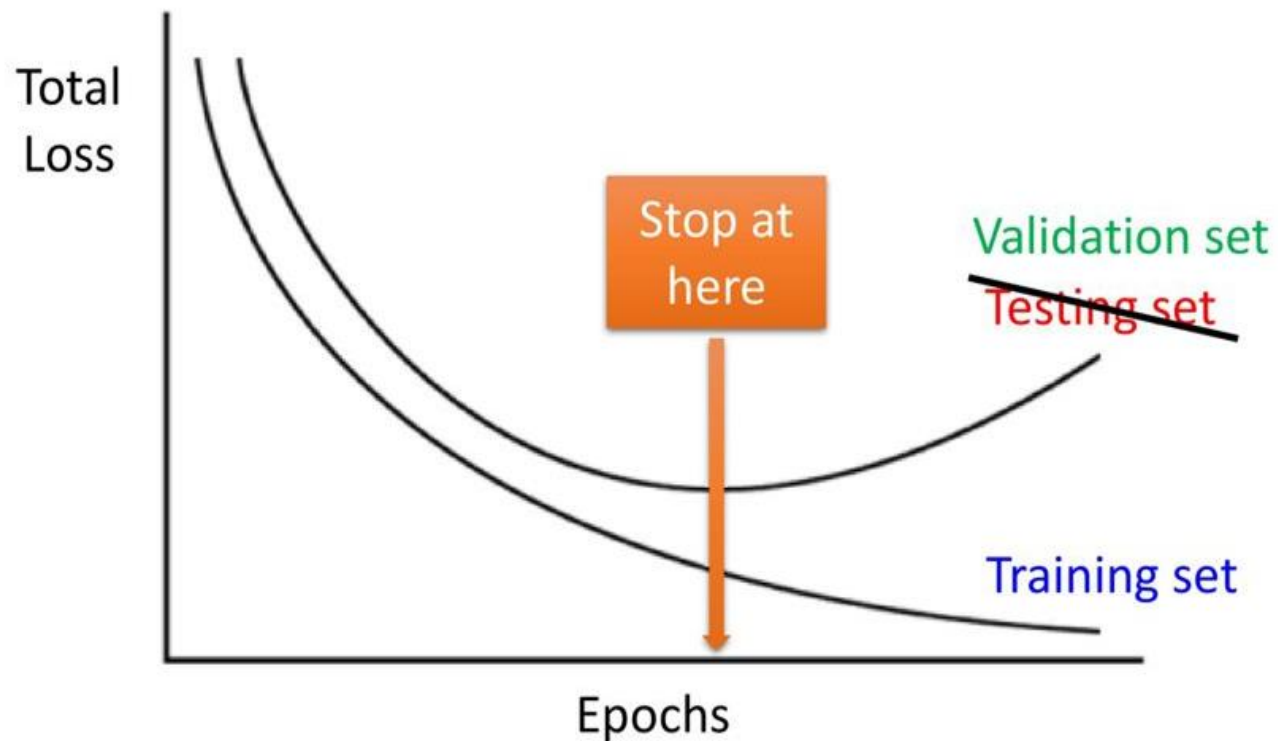
- 레이어의 분포를 안정화 시켜서 신경망 학습을 향상시키는 방법
- 트레이닝 시 효율을 저하시키는 내부공변량 전이효과를 제거하기 위해 사용

## DNN 모델의 성능 개선 방법

1. Drop-out

1. Batch Normalization

1. Earlystopping



- 검증 데이터에서 가장 낮은 오차가 나올 경우 학습을 멈추는 것을 의미
- 신경망을 다양하게 적용하는 경우 과적합을 줄여주는 효과적인 방법



## DNN 모델 구조

```
[80] # DNN architecture
input_dim = np.max(pad) + 1
embedding_dims = 20

def create_model(embedding_dims=20, optimizer='adam'):
    model = Sequential()
    model.add(Embedding(input_dim=input_dim, output_dim=embedding_dims))
    model.add(GlobalAveragePooling1D())
    model.add(Dense(24, activation='relu'))
    model.add(Dense(5, activation='softmax'))

    model.compile(loss='categorical_crossentropy',
                  optimizer=optimizer,
                  metrics=['accuracy'])

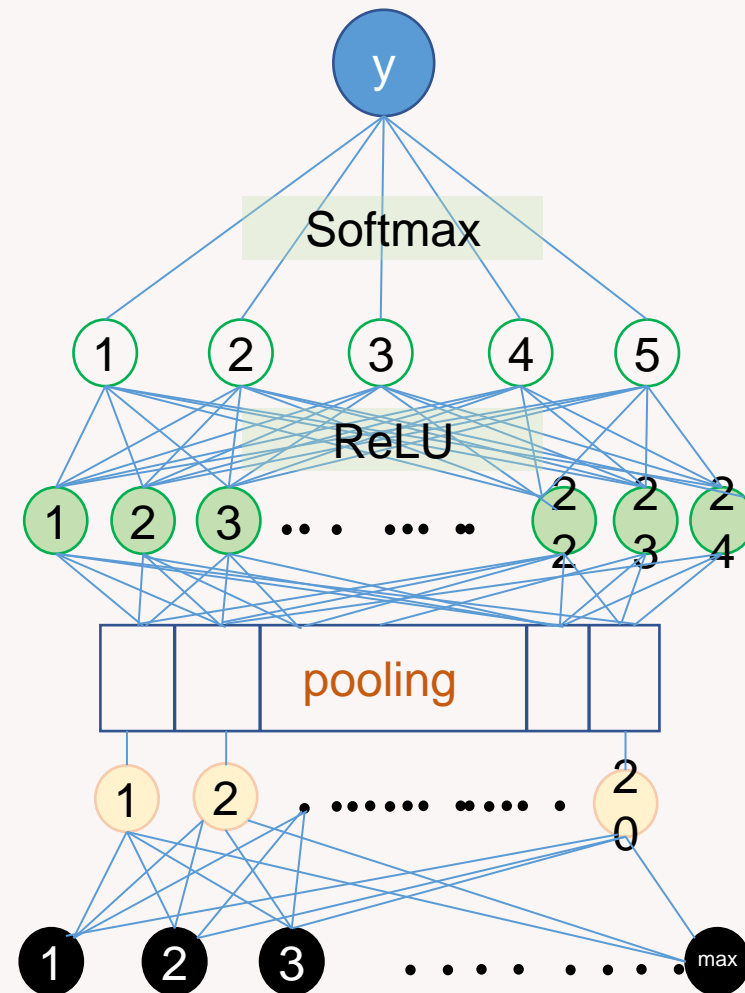
    return model
```

Output layer

Hidden layer

Embedding

Input layer

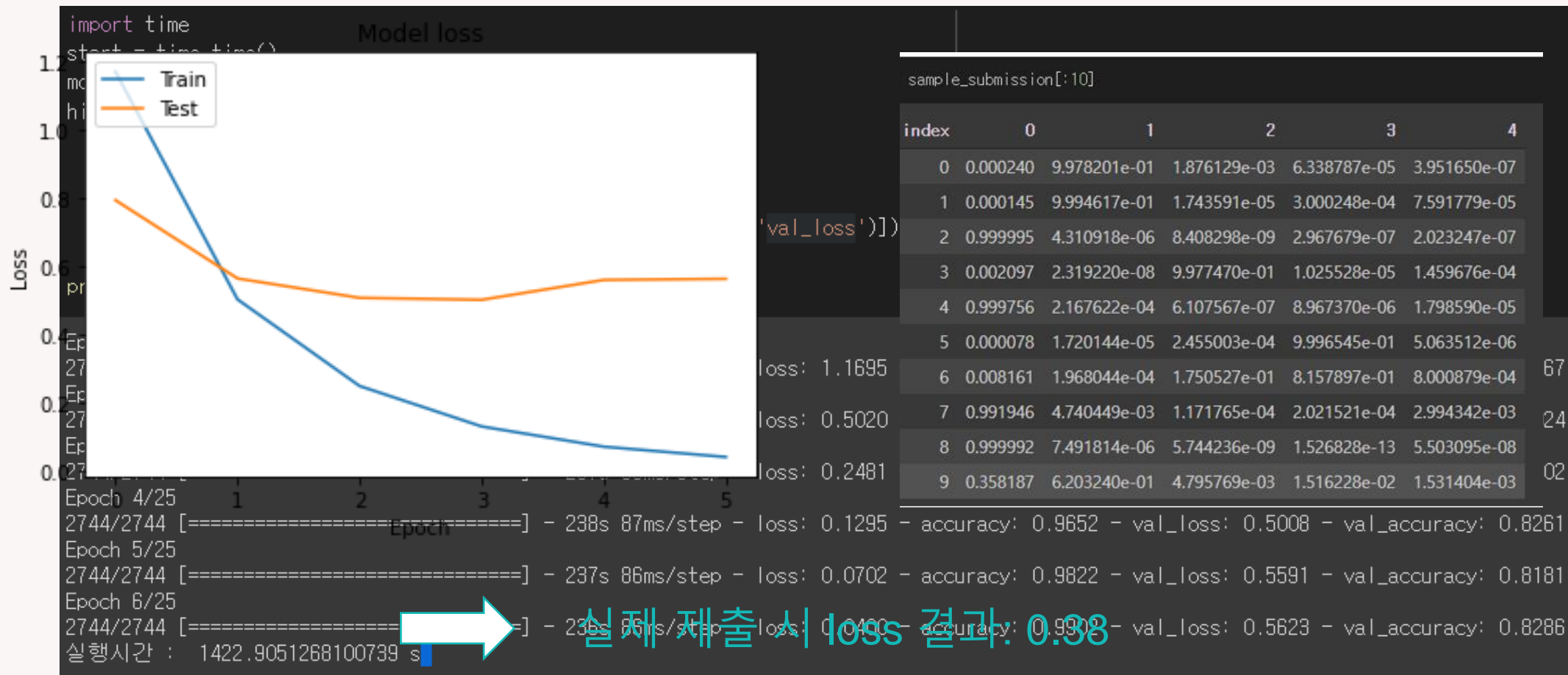


## 04

# DNN

Deep Neural Network

## BEST MODEL



### 성능

DNN을 활용한 경우가  
최종 Loss 값이 0.38로  
가장 좋은 성능을 보임

### 한계점

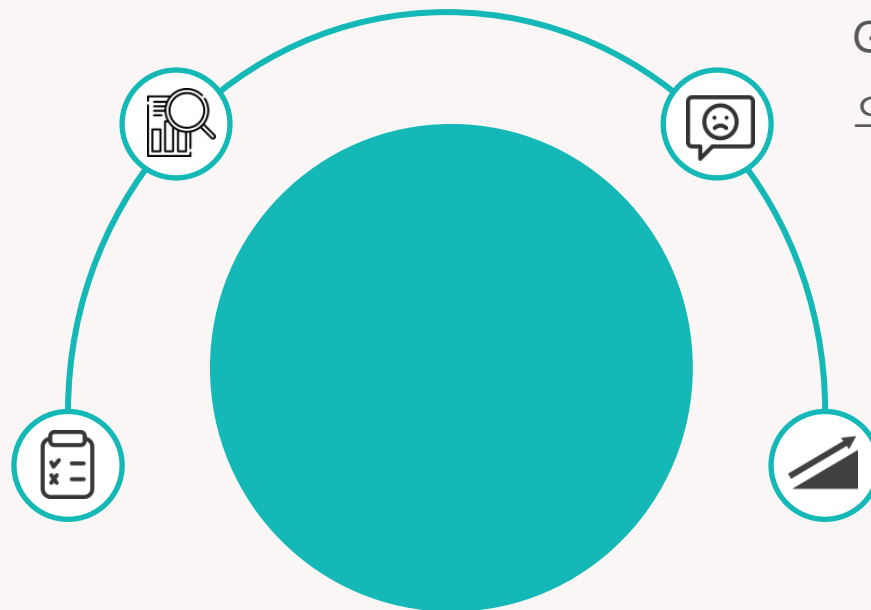
Glove와 Word2Vec을 사용한 경우  
오히려 더 낮은 성능을 보임

### 한계점에 대한 평가

데이터 크기가 작지 않고,  
구글의 뉴스를 사용해  
사전학습이 진행되어  
오히려 성능을 높이지 못 했다 추측

### 보완 방법

Gridsearch 등으로 파라미터를  
좀 더 다양하게 설정 후 진행



Team Staticurity

*감사합니다*