KU BIG
[big data]

변호사 NLP AI

KU-BIG

1. CNN

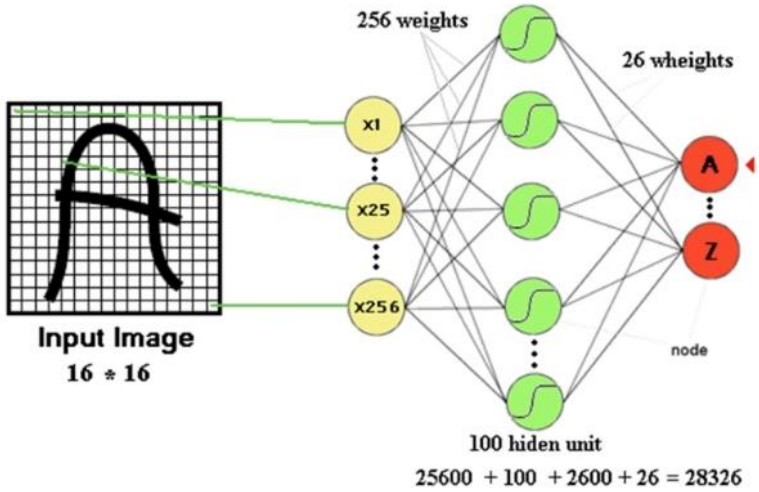
2. CNN과 NLP

3. CNN 참고 코드 리뷰

KU-BIG

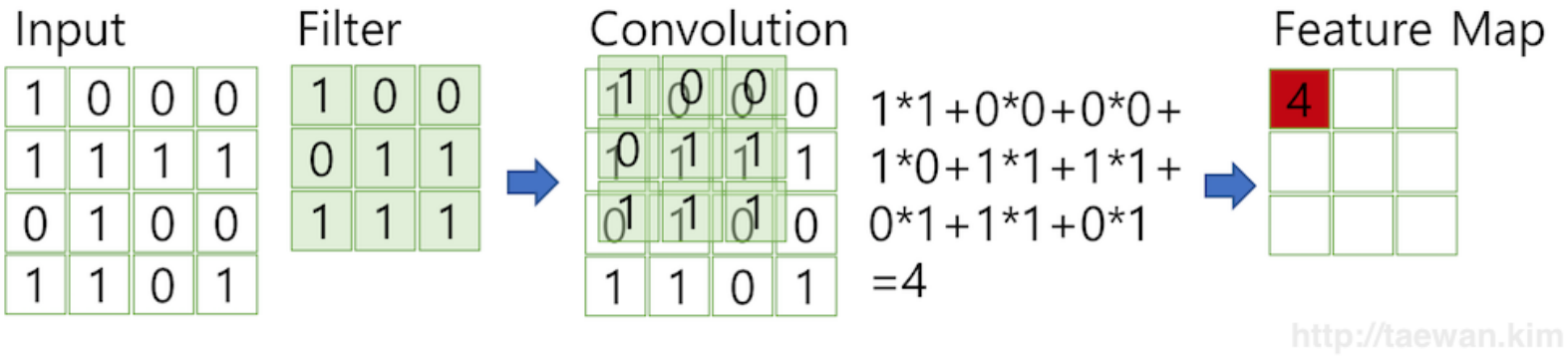
CNN 등장 배경

- 기존 multi-layered neural network에서 fully-connected layer의 문제점



- 16*16 이미지를 256*1로 바꾸어 입력 → 픽셀의 위치 정보 x
- 파라미터 수 급격히 증가

CNN이란?



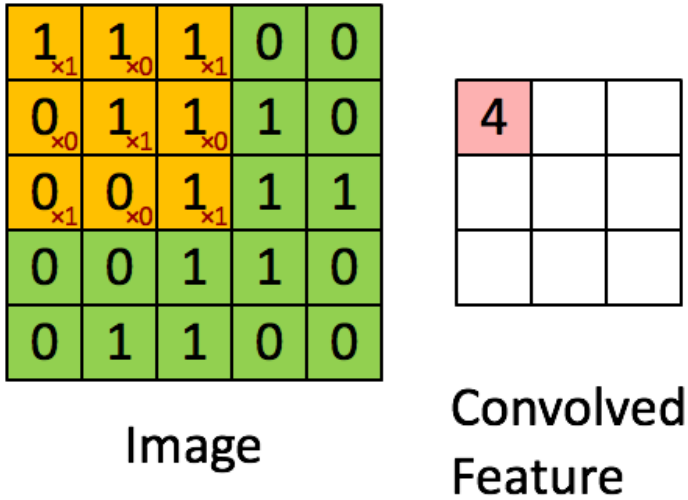
- Convolutional Neural Network (합성곱 신경망)

  : 입력데이터를 필터로 합성곱 연산을 수행하는 과정을 통해 feature map 형성

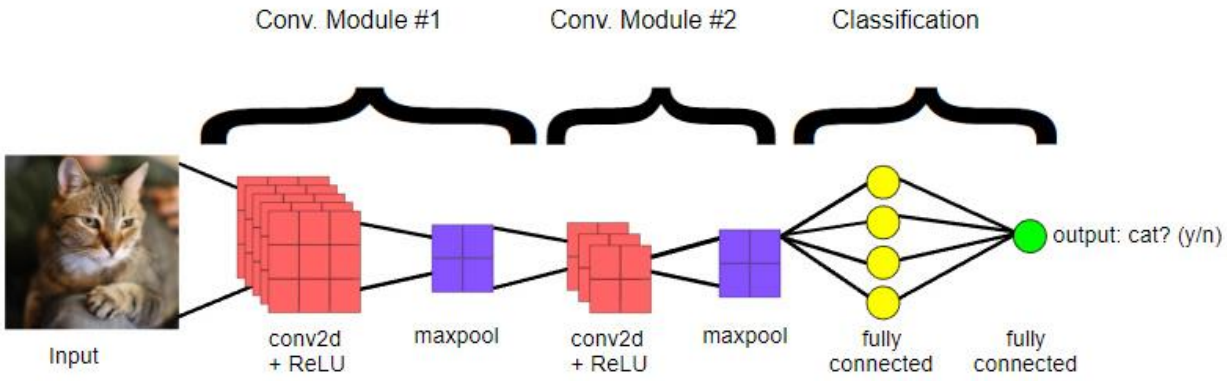
- 지역 정보를 잘 보존한다는 것이 큰 특징!

CNN

Convolution 형태



CNN



- 이미지를 input으로 가지는 경우가 많음
- Computer Vision 분야 - Image Classification, Image Detection, Semantic Segmentation 등

CNN과 NLP

- CNN은 지역 정보와 위치 정보를 보존하는 것이 특징…. Good at computer vision!
- NLP에서는 시퀀스 데이터를 처리하는 RNN이 적절하다고 알려져 있음

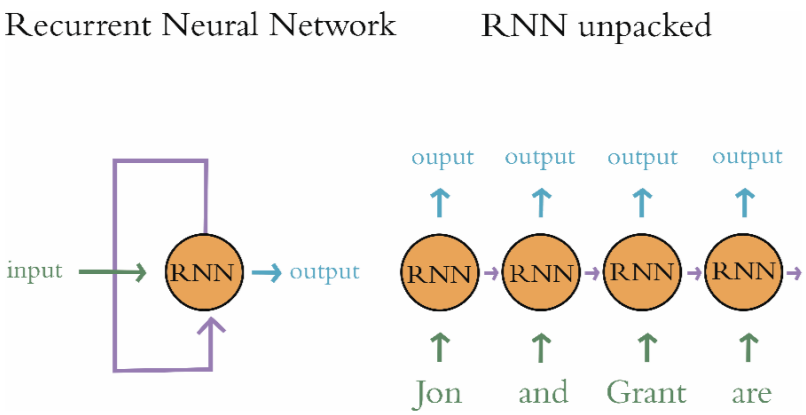


Figure 11.25 Schematic diagram of a recurrent neural network

CNN과 NLP

input:
Wait for the video and don’t rent it

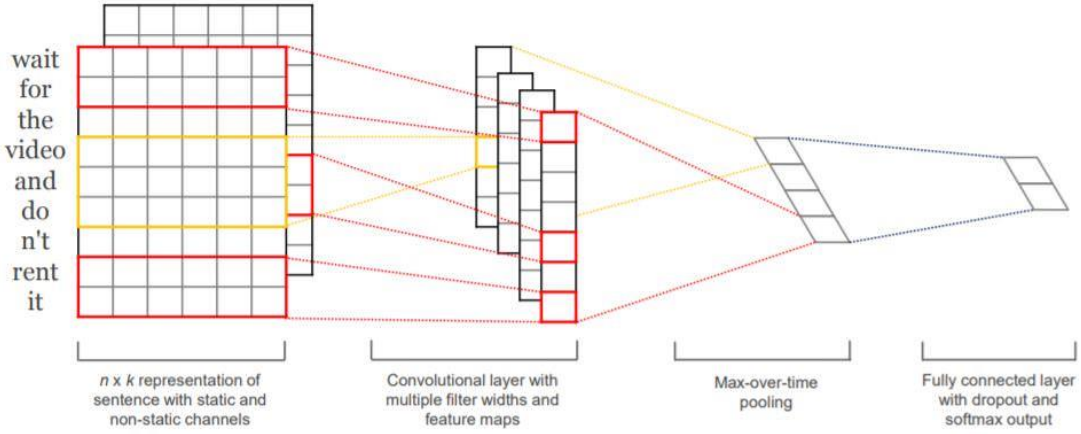
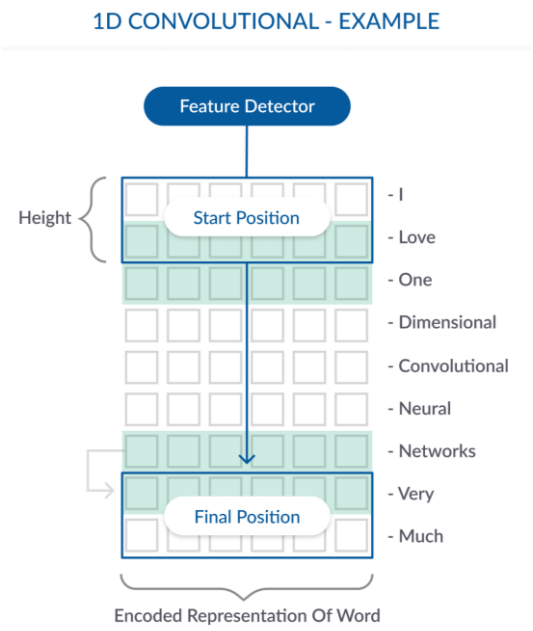


Figure 1: Model architecture with two channels for an example sentence.

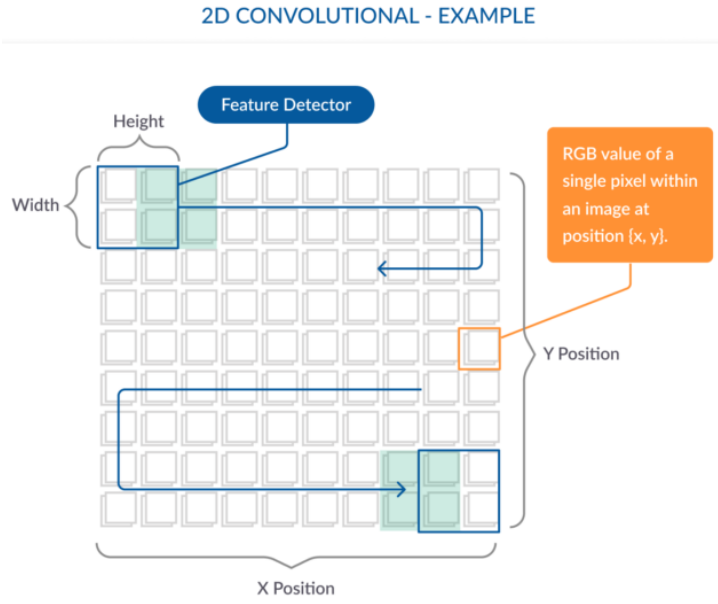
이미지에서는 지역적으로 필터를 이동시켰다면,

텍스트에서는 Matrix의 모든 단어의 전체 행에 사용

CNN과 NLP



필터가 한 방향으로 움직임



필터가 두 방향으로 움직임

CNN 참고코드 – 스팸 메일 분류하기

```
data[:10]
```

v1	v2
ham	Go until jurong point, crazy.. Available only ...
ham	Ok lar... Joking wif u oni...
spam	Free entry in 2 a wkly comp to win FA Cup fina...
ham	U dun say so early hor... U c already then say...
ham	Nah I don't think he goes to usf, he lives aro...
spam	FreeMsg Hey there darling it's been 3 week's n...
ham	Even my brother is not like to speak with me. ...
ham	As per your request 'Melle Melle (Oru Minnamin...
spam	WINNER!! As a valued network customer you have...
spam	Had your mobile 11 months or more? U R entitle...

CNN 참고코드 – 스팸 메일 분류하기

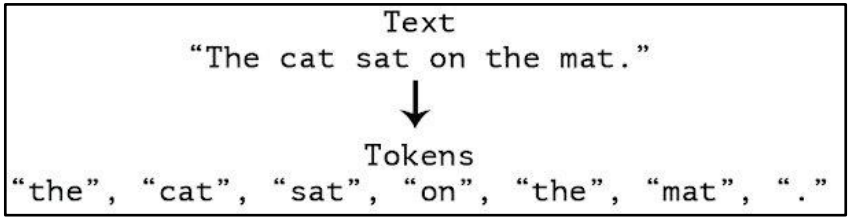
토큰화를 수행하여, 단어를 나누고 단어별로 정수 인코딩을 해줌

```
vocab_size = 1000
tokenizer = Tokenizer(num_words = vocab_size)
tokenizer.fit_on_texts(X_data) # 5169개의 행을 가진 X의 각 행에 토큰화를 수행
sequences = tokenizer.texts_to_sequences(X_data) # 단어를 숫자값, 인덱스로 변환하여 저장
```

```
print(sequences[:5])
```

```
>> [[47, 433, 780, 705, 662, 64, 8, 94, 121, 434, 142, 68, 57, 137], [49, 306, 435, 6], [53,
537, 8, 20, 4, 934, 2, 220, 706, 267, 70, 2, 2, 359, 537, 604, 82, 436, 185, 707, 437], [6,
226, 152, 23, 347, 6, 138, 145, 56, 152], [935, 1, 97, 96, 69, 453, 2, 877, 69, 198, 105,
438]]
```

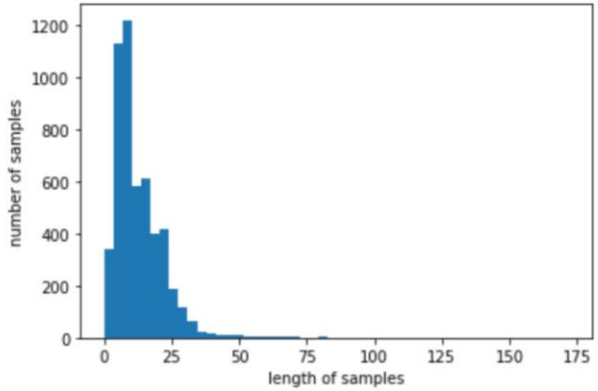
토큰화:



CNN 참고코드 – 스팸 메일 분류하기

```
X_data = sequences
print('메일의 최대 길이 : %d' % max(len(l) for l in X_data))
print('메일의 평균 길이 : %f' % (sum(map(len, X_data))/len(X_data)))
plt.hist([len(s) for s in X_data], bins=50)
plt.xlabel('length of samples')
plt.ylabel('number of samples')
plt.show()
```

```
메일의 최대 길이 : 172
메일의 평균 길이 : 12.566841
```



- 패딩

```
# 전체 데이터셋의 길이는 max_len으로 맞춥니다.
max_len = 172
data = pad_sequences(X_data, maxlen = max_len)
print("훈련 데이터의 크기(shape): ", data.shape)
```

CNN 참고코드 – 스팸 메일 분류하기

```python
from tensorflow.keras.layers import Dense, Conv1D, GlobalMaxPooling1D, Embedding, Dropout, MaxPooling1D
from tensorflow.keras.models import Sequential
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint
```

```python
model = Sequential()
model.add(Embedding(vocab_size, 32))
model.add(Dropout(0.2))
model.add(Conv1D(32, 5, strides=1, padding='valid', activation='relu'))
model.add(GlobalMaxPooling1D())
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(1, activation='sigmoid'))
model.summary()
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['acc'])
```

```python
print("\n 테스트 정확도: %.4f" % (model.evaluate(X_test, y_test)[1]))
```

```
테스트 정확도: 0.9836
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, None, 32)          32000
_________________________________________________________________
dropout_2 (Dropout)          (None, None, 32)          0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, None, 32)          5152
_________________________________________________________________
global_max_pooling1d_1 (Glob (None, 32)                0
_________________________________________________________________
dense_2 (Dense)              (None, 64)                2112
_________________________________________________________________
dropout_3 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 65
=================================================================
Total params: 39,329
Trainable params: 39,329
Non-trainable params: 0
_________________________________________________________________
```

CNN 참고코드 – IMDB 리뷰 분류하기

```
temp_str = "This movie was just way too overrated. The fighting was not professional and in slow motion. I was expecting more f
rom a 200 million budget movie. The little sister of T.Challa was just trying too hard to be funny. The story was really dumb a
s well. Don't watch this movie if you are going because others say its great unless you are a Black Panther fan or Marvels fa
n."

sentiment_predict(temp_str)
```

```
97.43% 확률로 부정 리뷰입니다.
```

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, Dropout, Conv1D, GlobalMaxPooling1D, Dense
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint
from tensorflow.keras.models import load_model
```

```
embedding_dim = 256
batch_size = 256
```

```
model = Sequential()
model.add(Embedding(vocab_size, 256))
model.add(Dropout(0.3))
model.add(Conv1D(256, 3, padding='valid', activation='relu'))
model.add(GlobalMaxPooling1D())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid'))
```

CNN 참고코드 – 네이버 영화 리뷰 분류하기

document	label
아 더빙.. 진짜 짜증나네요 목소리	0
흠...포스터보고 초딩영화줄....오버연기조차 가볍지 않구나	1
너무재밓었다그래서보는것을추천한다	0
교도소 이야기구먼 ..솔직히 재미는 없다..평점 조정	0
사이몬페그의 익살스런 연기가 돋보였던 영화!스파이더맨에서 늙어보이기만 했던 커스틴 ...	1

부정 : 0

긍정 : 1

```
sentiment_predict('이 영화 개꿀잼 ㅋㅋㅋ')
```

```
93.73% 확률로 긍정 리뷰입니다.
```


KU-BIG

CNN 참고코드 – 네이버 영화 리뷰 분류하기

```
for sz in [3, 4, 5]:
    conv = Conv1D(filters = num_filters,
                  kernel_size = sz,
                  padding = "valid",
                  activation = "relu",
                  strides = 1)(z)
    conv = GlobalMaxPooling1D()(conv)
    conv = Flatten()(conv)
    conv_blocks.append(conv)
```

```
z = Concatenate()(conv_blocks) if len(conv_blocks) > 1 else conv_blocks[0]
z = Dropout(dropout_prob[1])(z)
z = Dense(128, activation="relu")(z)
model_output = Dense(1, activation="sigmoid")(z)

model = Model(model_input, model_output)
model.compile(loss="binary_crossentropy", optimizer="adam", metrics=["acc"])
```

```
테스트 정확도: 0.8428
```

Layer (type)	Output Shape	Param #	Connected to
input_2 (InputLayer)	[(None, 30)]	0	
embedding (Embedding)	(None, 30, 128)	1280000	input_2[0][0]
dropout_3 (Dropout)	(None, 30, 128)	0	embedding[0][0]
conv1d_3 (Conv1D)	(None, 28, 128)	49280	dropout_3[0][0]
conv1d_4 (Conv1D)	(None, 27, 128)	65664	dropout_3[0][0]
conv1d_5 (Conv1D)	(None, 26, 128)	82048	dropout_3[0][0]
global_max_pooling1d_3 (GlobalM	(None, 128)	0	conv1d_3[0][0]
global_max_pooling1d_4 (GlobalM	(None, 128)	0	conv1d_4[0][0]
global_max_pooling1d_5 (GlobalM	(None, 128)	0	conv1d_5[0][0]
flatten_3 (Flatten)	(None, 128)	0	global_max_pooling1d_3[0][0]
flatten_4 (Flatten)	(None, 128)	0	global_max_pooling1d_4[0][0]
flatten_5 (Flatten)	(None, 128)	0	global_max_pooling1d_5[0][0]
concatenate_2 (Concatenate)	(None, 384)	0	flatten_3[0][0] flatten_4[0][0] flatten_5[0][0]
dropout_4 (Dropout)	(None, 384)	0	concatenate_2[0][0]
dense_4 (Dense)	(None, 128)	49280	dropout_4[0][0]
dense_5 (Dense)	(None, 1)	129	dense_4[0][0]

Total params: 1,526,401
Trainable params: 1,526,401
Non-trainable params: 0

향후 스터디 계획

- RNN 모델 공부 및 NLP 적용
- LSTM 등의 추가 NLP 모델
- 변호사 AI 데이터 수집 및 전처리