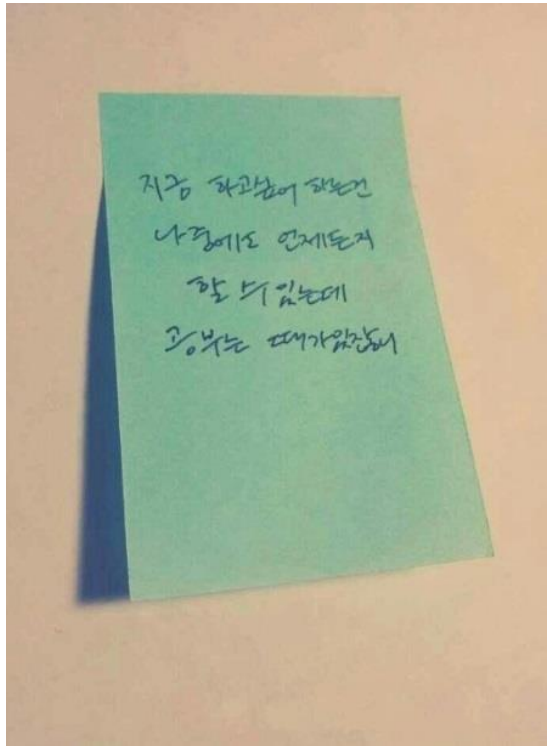
TEXT CNN

장시온

CNN

CNN은 이미지의 영역

이미지



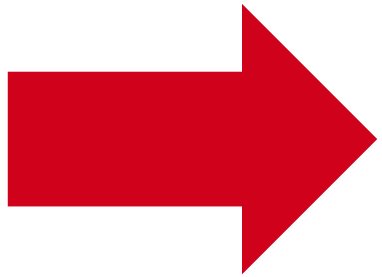
[1, 가로, 세로 ,색깔]
4차원

CNN

문서도 이렇게 만들면 CNN 되지 않을까?
(word2vec)

문서

"지금 하고 싶어하는건
나중에 언제든지 할 수 있는데
공부는 때가 있잖니"



TEXT CNN

내가 풀려는 문제

영화 감상평 호감/비호감 분석

```
id      document        label
9976970 아 더빙.. 진짜 짜증나네요 목소리       0
3819312 흠...포스터보고 초딩영화줄....오버연기조차 가볍지 않구나       1
10265843        너무재밌었다그래서보는것을추천한다      0
9045019 교도소 이야기구먼 ..솔직히 재미는 없다..평점 조정       0
6483659 사이몬페그의 익살스런 연기가 돋보였던 영화!스파이더맨에서 늙어보이기만 했던 커스틴 던스트가 너무나도 이뻐보였다 1
5403919 막 걸음마 뗀 3세부터 초등학교 1학년생인 8살용영화.ㅋㅋㅋ...별반개도 아까움.     0
7797314 원작의 긴장감을 제대로 살려내지못했다.  0
```

Train_data = 10000

Test_data = 10000

TEXT CNN

형태소 분석

여기서부터는 저의 ~~예술~~이 들어갑니다.

```
id      document        label
9976970 아 더빙.. 진짜 짜증나네요 목소리       0
3819312 흠...포스터보고 초딩영화줄....오버연기조차 가볍지 않구나       1
10265843        너무재밌었다그래서보는것을추천한다      0
9045019 교도소 이야기구먼 ..솔직히 재미는 없다..평점 조정       0
6483659 사이몬페그의 익살스런 연기가 돋보였던 영화!스파이더맨에서 늙어보이기만 했던 커스틴 던스트가 너무나도 이뻐보였다 1
5403919 막 걸음마 뗀 3세부터 초등학교 1학년생인 8살용영화.ㅋㅋㅋ...별반개도 아까움.     0
7797314 원작의 긴장감을 제대로 살려내지못했다.  0
```

- 주어, 동사, 목적어 아닌거 제외
- 주어,동사 목적어중 단어의 길이가 1인거 제외

TEXT CNN

Word2vec 데이터 준비 및 학습

word2vec

TEXT CNN

문서(글자)의 벡터화

Dictionary = [“재밌”, … , “영화” , … , “연기” , … , “애나벨”]

0 18691 31503 51366

문서_1= [“영화/Noun”, “애나벨/Noun”, “재밌/Verb”, “연기/Noun”, “꿀잼/Noun”]

Train_data (2차원)

문서_1=[0, 18691, 31503, 51366, -1, -1, … -1]

문서_10000=[……..]

TEXT CNN

-1 의 이념

CNN할때의 SIZE 맞춰주기

Train

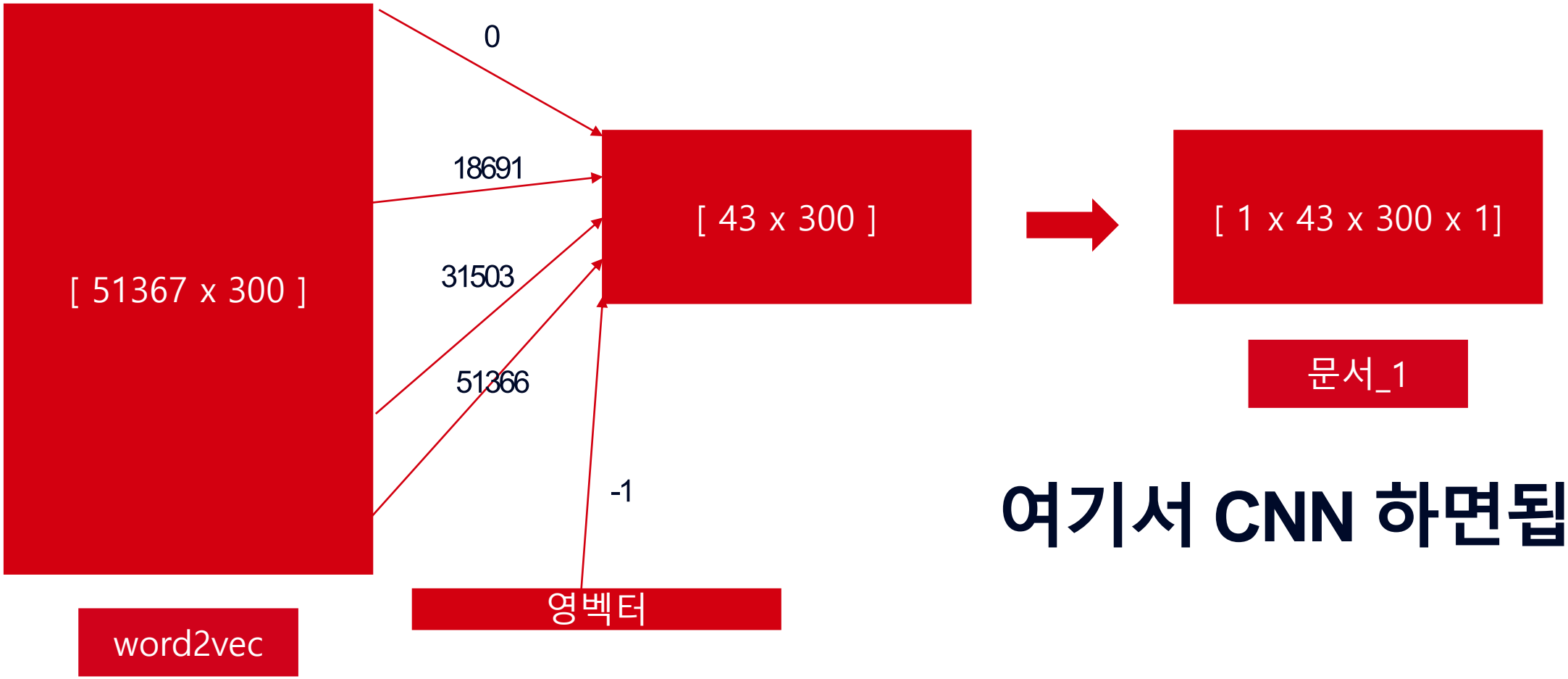
- 문장의 길이 맞춰주기

Test

- 문장의 길이 맞춰주기
- Word2vec에 없는 keyword

TEXT CNN

문서의 4차원(이미지) 변환



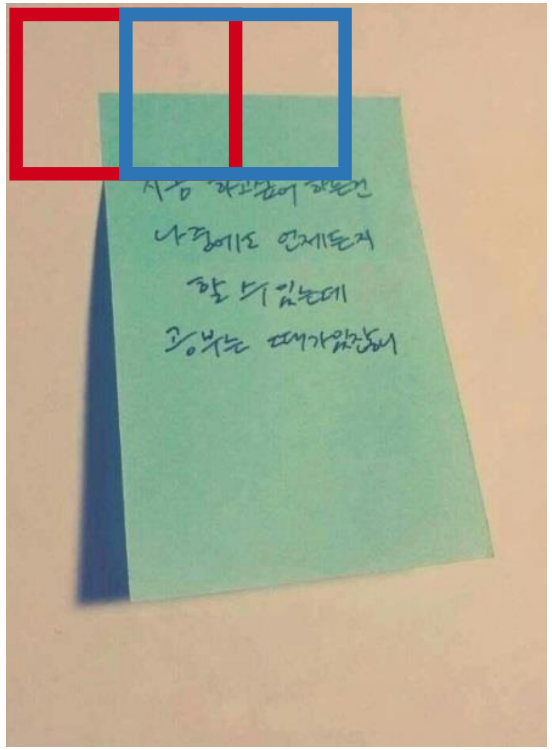
여기서 CNN 하면됩니다.

문서_1=[0, 18691, 31503, 51366, -1, -1, … -1]

TEXT CNN

CNN 차이

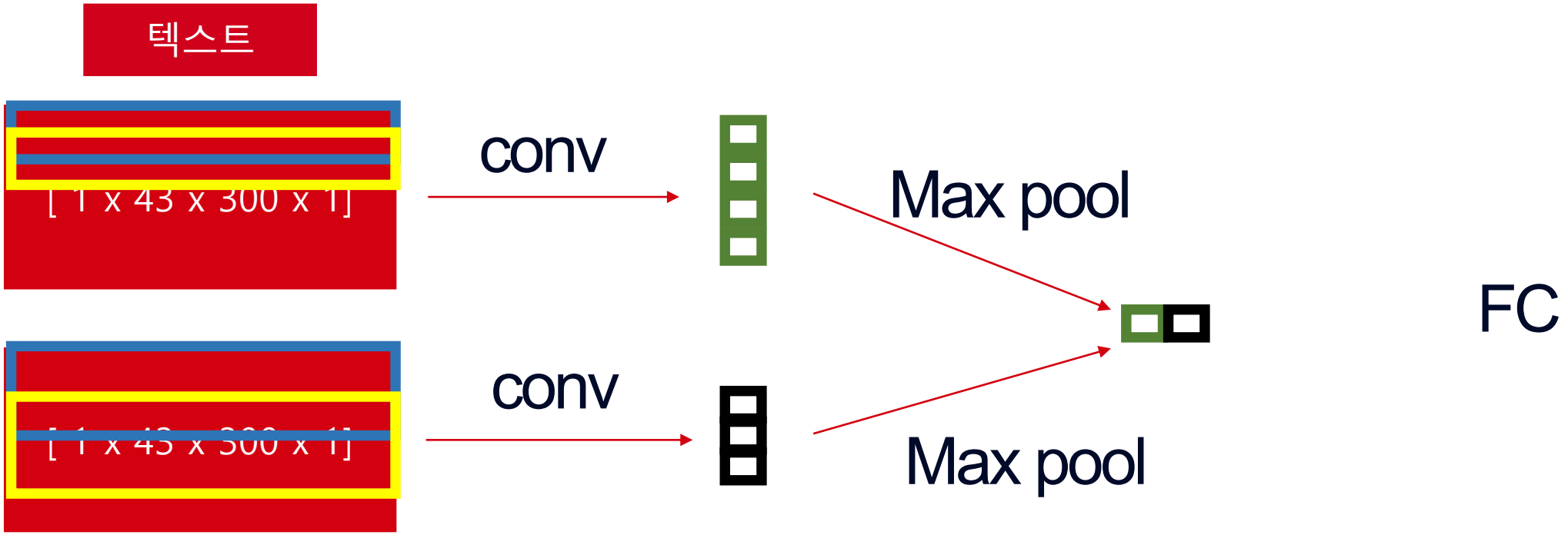
이미지



텍스트

TEXT CNN

CNN 차이



TEXT CNN

Model 차이

1. CNN-rand

- Word2vec 랜덤값 생성, CNN에 맞물려 학습
- 논문에서 Baseline으로 사용

```
word2vec = tf.Variable(tf.random_uniform([len(self.reverse_dictionary), self.EMBED_SIZE], -1.0, 1.0))
```

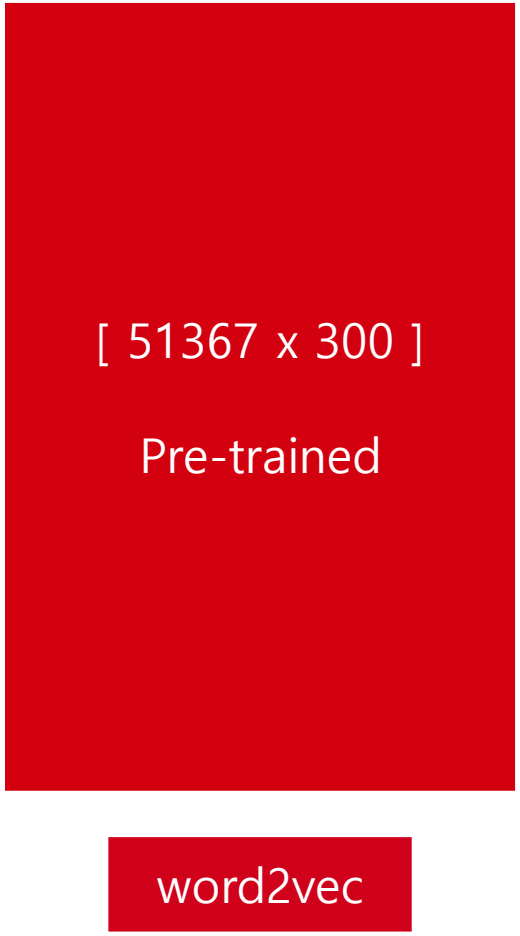
TEXT CNN

Model 차이

2. CNN-static

- Word2vec 학습시킨거 사용 (pre-trained)
  ex) Stanford 20si word2vec 학습
- Word2vec 학습 X , CNN만 학습

```
word2vec = tf.Variable(self.static_word2vec,trainable=False)
```



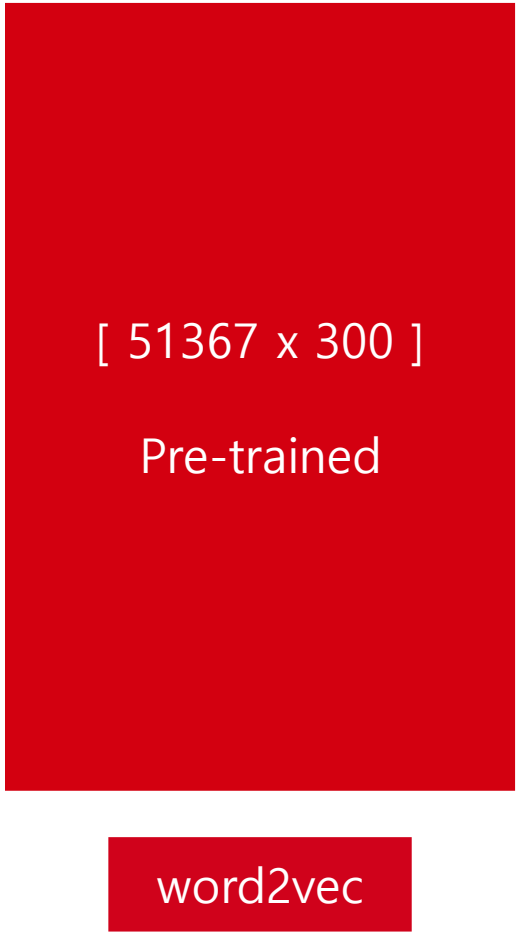
TEXT CNN

Model 차이

3. CNN-non-static

- Word2vec 학습시킨거 사용 (pre-trained)
  ex) Stanford 20si word2vec 학습
- Word2vec , CNN에 맞물려 학습

```
word2vec = tf.Variable(self.static_word2vec,trainable=True)
```



TEXT CNN

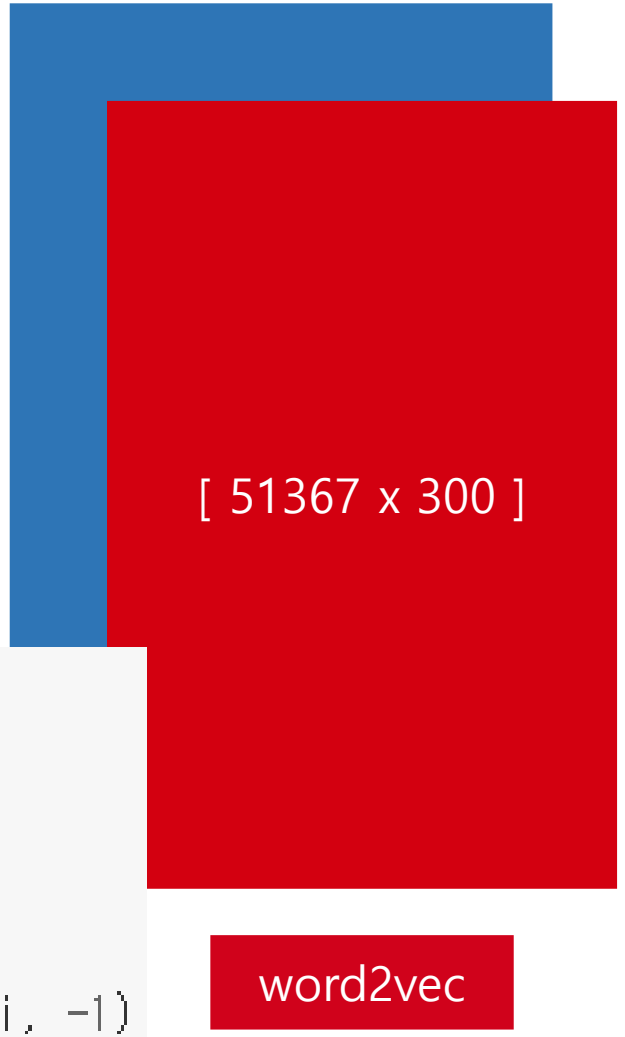
Model 차이

3. CNN-multichannel

- static, non-static 을 동시에 CNN 진행
  static : 학습 X
  non-static : 학습 O

```
word2vec = tf.Variable(self.static_word2vec,trainable=False)
embedded_chars = tf.nn.embedding_lookup(word2vec, X)
embedded_chars_expanded = tf.expand_dims(embedded_chars, -1)

multi_word2vec = tf.Variable(self.static_word2vec,trainable=True)
embedded_chars_multi = tf.nn.embedding_lookup(multi_word2vec, X)
embedded_chars_expanded_multi = tf.expand_dims(embedded_chars_multi, -1)
```



TEXT CNN

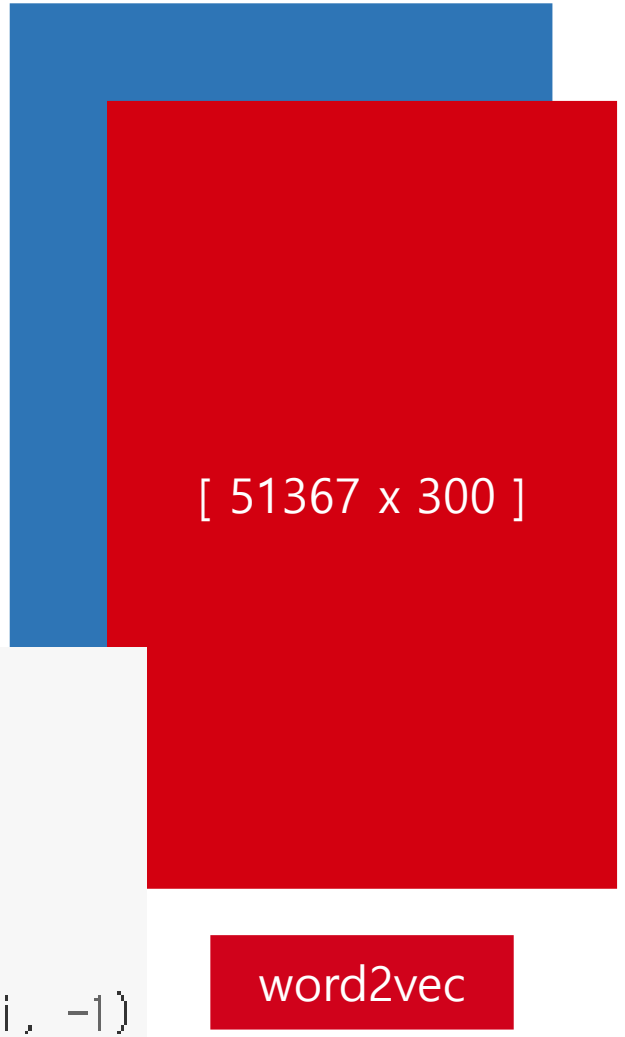
Model 차이

3. CNN-multichannel

- static, non-static 을 동시에 CNN 진행
  static : 학습 X
  non-static : 학습 O

```
word2vec = tf.Variable(self.static_word2vec,trainable=False)
embedded_chars = tf.nn.embedding_lookup(word2vec, X)
embedded_chars_expanded = tf.expand_dims(embedded_chars, -1)

multi_word2vec = tf.Variable(self.static_word2vec,trainable=True)
embedded_chars_multi = tf.nn.embedding_lookup(multi_word2vec, X)
embedded_chars_expanded_multi = tf.expand_dims(embedded_chars_multi, -1)
```



결과

Accuracy

	Train	Test
CNN-Rand	95.72	74.53
CNN-Static	81.62	68.88
CNN-Non-static	91.41	75.42
CNN-Multichannel	92.78	74.09
CNN-Tf-idf-xgboost	79.17	74.4

THANK YOU