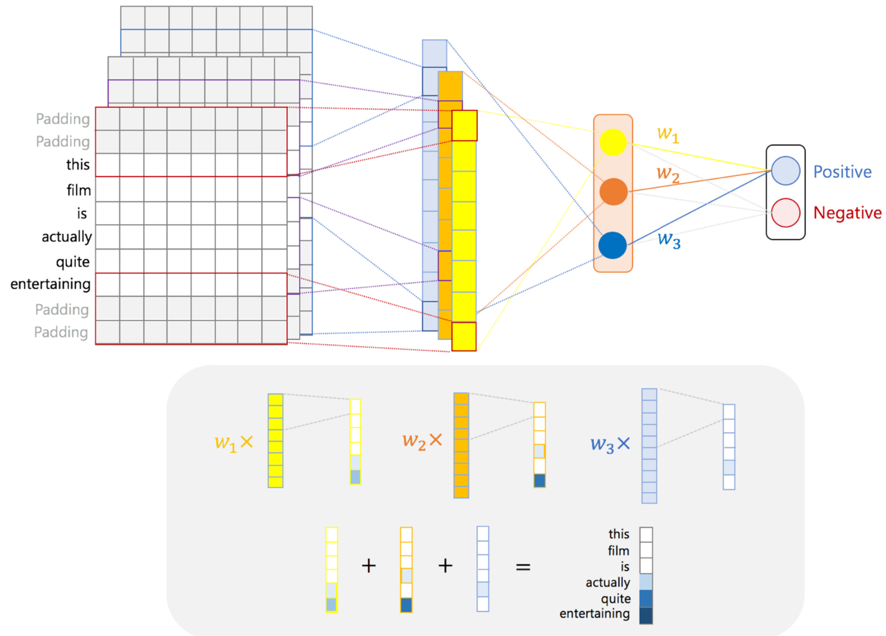


Lecture 9-2: Document Classification CNN-based Classifier

Pilsung Kang

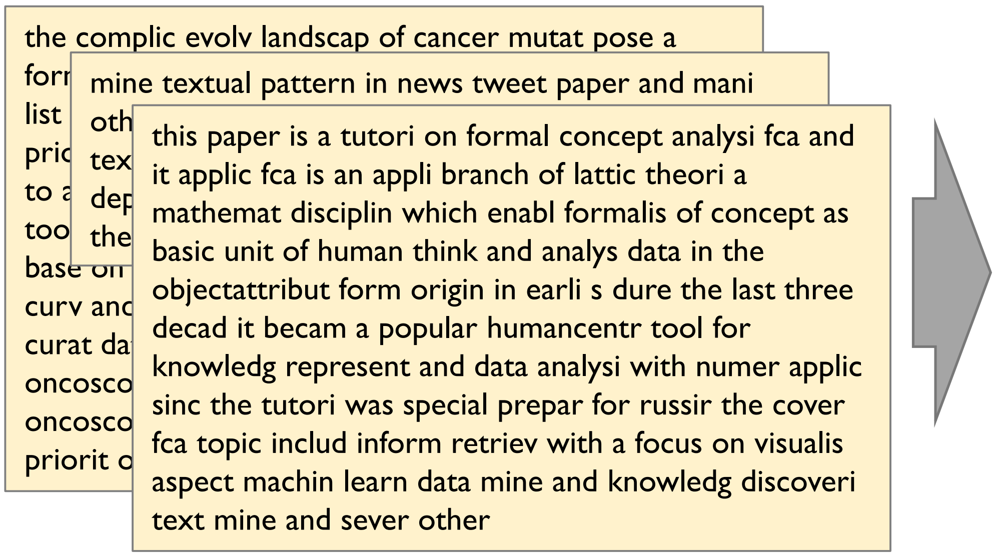
School of Industrial Management Engineering

Korea University

MLP for Document Classification

- MLP for Document Classification

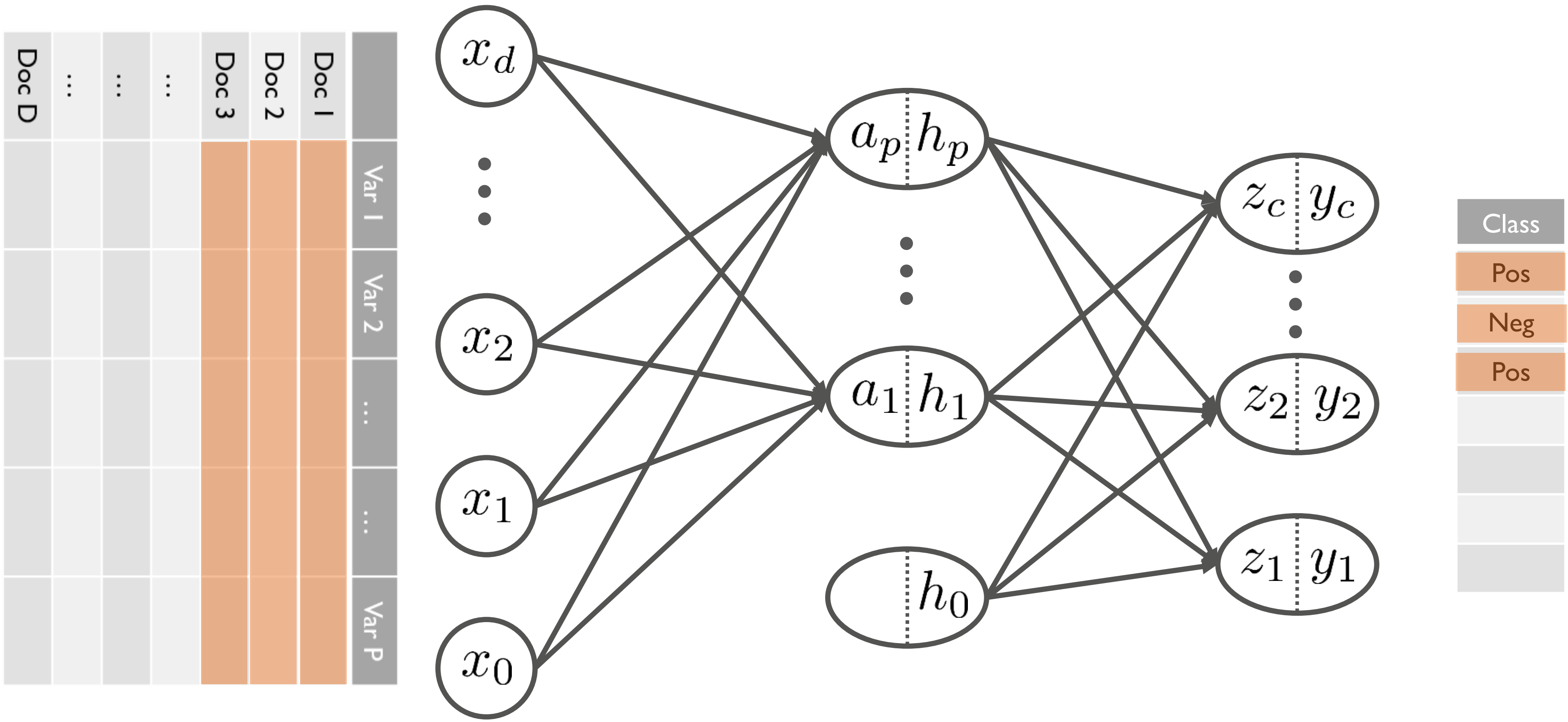
Transform unstructured data into structured data



	Var 1	Var 2	…	…	Var P	Class
Doc 1						Pos
Doc 2						Neg
Doc 3						Pos
…						
…						
…						
Doc D						

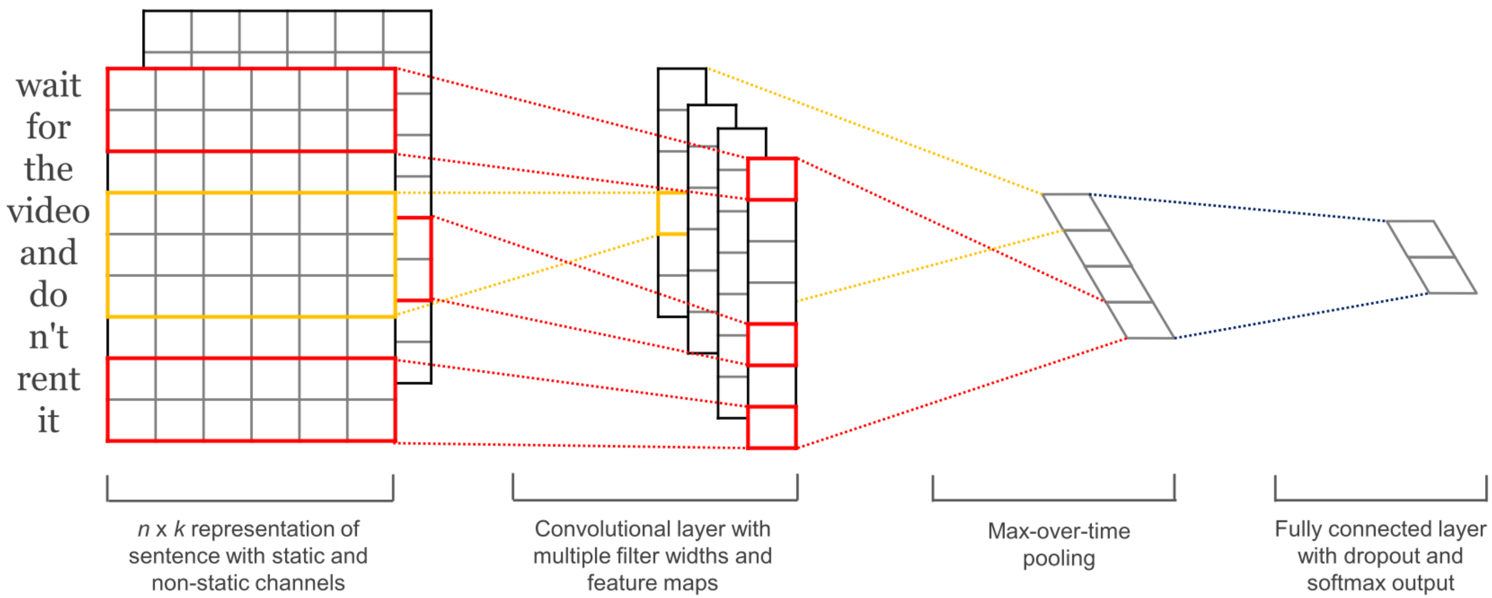
MLP for Document Classification

- MLP for Document Classification

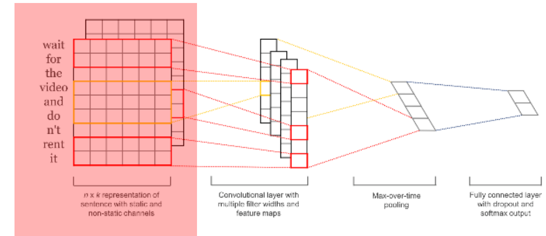


CNN for Sentence Classification

- Yoon Kim (2014)
  - ✓ A simple CNN structure with only one convolution layer

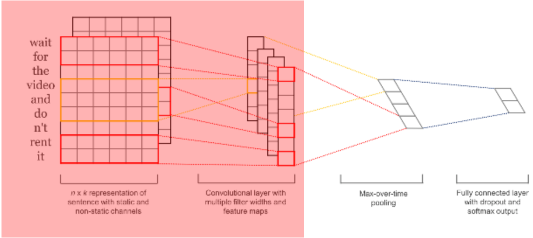


CNN for Sentence Classification



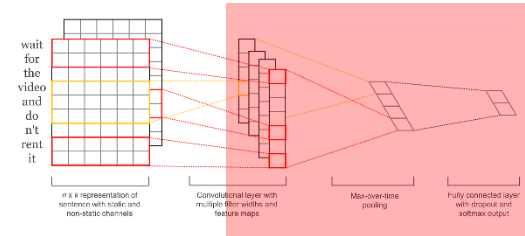
- Yoon Kim (2014)
  - ✓ Input: n by k (by m) representation of sentence
    - n: the number of words in a sentence (parameter)
      - Shorter sentences are zero-padded and longer sentences are trimmed
    - k: word embedding dimensions
      - Pre-trained word embedding vectors can be used
      - These vectors can be static (not updated during the training) or non-static (fine-tuned for the task-specific corpus)
    - Multi-channel input is also possible → an input sentence becomes a tensor
      - Pre-trained word embedding by Word2Vec (Static)
      - Pre-trained word embedding by Word2Vec (Non-static)
      - Pre-trained word embedding by Glove (Static)
      - Randomly initialized embedding
      - …

CNN for Sentence Classification



- Yoon Kim (2014)
  - ✓ Convolution
    - ▪ Different size of convolutions can be used
      - A squared convolution is common for image processing, but a rectangular convolution with the width of k is used for text processing
      - Convolution stride is usually set to 1
      - The larger the height, the more words are considered by the convolution at a single time

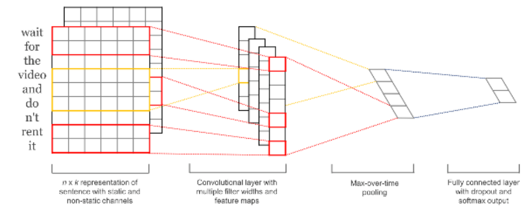
CNN for Sentence Classification



- Yoon Kim (2014)
  - ✓ Max pooling
    - To capture the most important feature
    - Can deal with variable sentence lengths
  - ✓ Fully connected operation
    - Connected to two output nodes (positive and negative)
  - ✓ Learning strategies
    - Filter window of 3, 4, and 5 with 100 feature maps each
    - Dropout (rate = 0.5) is used between the fully connected layer and the output layer
    - L_2 regularization (3) for the weight is used
    - Mini-batch size of 50
    - These values were chosen via a grid search on the SST-2 dev set

CNN for Sentence Classification

- Yoon Kim (2014): Experiments
  - ✓ Datasets

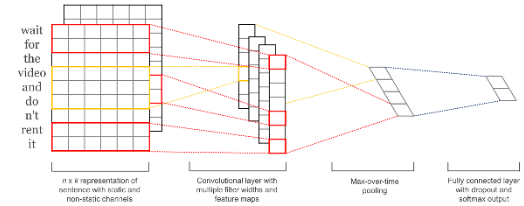


Data	c	l	N	$\|V\|$	$\|V_{pre}\|$	*Test*
MR	2	20	10662	18765	16448	CV
SST-1	5	18	11855	17836	16262	2210
SST-2	2	19	9613	16185	14838	1821
Subj	2	23	10000	21323	17913	CV
TREC	6	10	5952	9592	9125	500
CR	2	19	3775	5340	5046	CV
MPQA	2	3	10606	6246	6083	CV

Table 1: Summary statistics for the datasets after tokenization. c: Number of target classes. l: Average sentence length. N: Dataset size. $|V|$: Vocabulary size. $|V_{pre}|$: Number of words present in the set of pre-trained word vectors. *Test*: Test set size (CV means there was no standard train/test split and thus 10-fold CV was used).

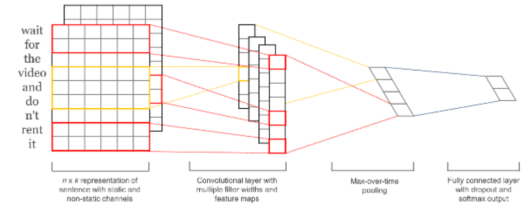
CNN for Sentence Classification

- Yoon Kim (2014): Experiments
  - ✓ Classification performance



Model	MR	SST-1	SST-2	Subj	TREC	CR	MPQA
CNN-rand	76.1	45.0	82.7	89.6	91.2	79.8	83.4
CNN-static	81.0	45.5	86.8	93.0	92.8	84.7	**89.6**
CNN-non-static	**81.5**	48.0	87.2	93.4	93.6	84.3	89.5
CNN-multichannel	81.1	47.4	**88.1**	93.2	92.2	**85.0**	89.4
RAE (Socher et al., 2011)	77.7	43.2	82.4	–	–	–	86.4
MV-RNN (Socher et al., 2012)	79.0	44.4	82.9	–	–	–	–
RNTN (Socher et al., 2013)	–	45.7	85.4	–	–	–	–
DCNN (Kalchbrenner et al., 2014)	–	48.5	86.8	–	93.0	–	–
Paragraph-Vec (Le and Mikolov, 2014)	–	**48.7**	87.8	–	–	–	–
CCAE (Hermann and Blunsom, 2013)	77.8	–	–	–	–	–	87.2
Sent-Parser (Dong et al., 2014)	79.5	–	–	–	–	–	86.3
NBSVM (Wang and Manning, 2012)	79.4	–	–	93.2	–	81.8	86.3
MNB (Wang and Manning, 2012)	79.0	–	–	**93.6**	–	80.0	86.3
G-Dropout (Wang and Manning, 2013)	79.0	–	–	93.4	–	82.1	86.1
F-Dropout (Wang and Manning, 2013)	79.1	–	–	**93.6**	–	81.9	86.3
Tree-CRF (Nakagawa et al., 2010)	77.3	–	–	–	–	81.4	86.1
CRF-PR (Yang and Cardie, 2014)	–	–	–	–	–	82.7	–
SVM_S (Silva et al., 2011)	–	–	–	–	**95.0**	–	–

CNN for Sentence Classification



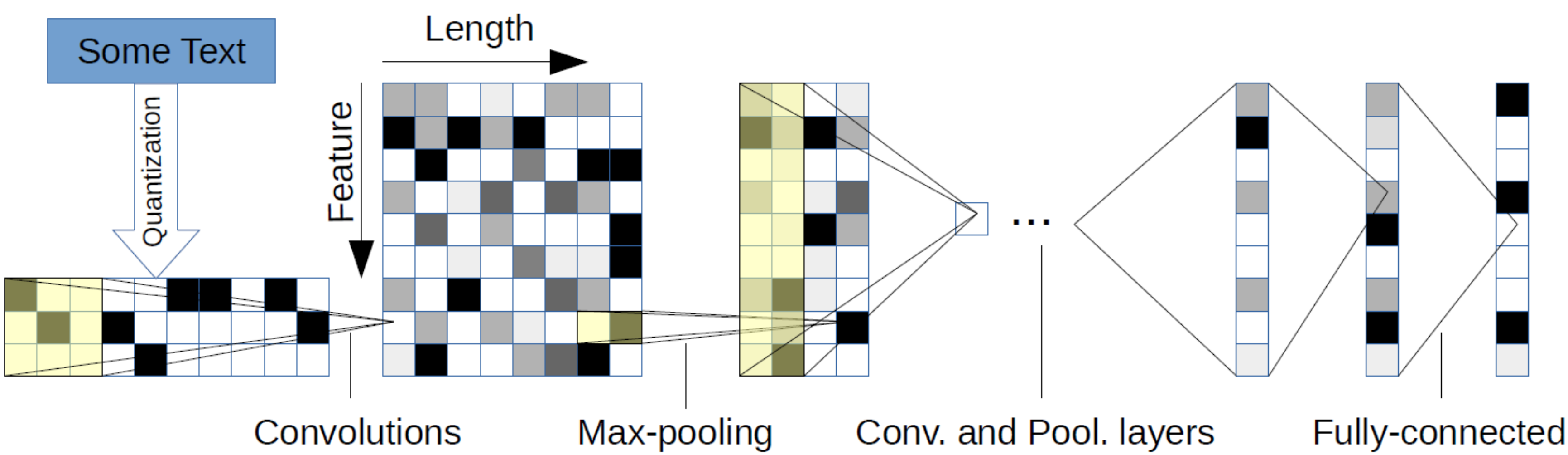
- Yoon Kim (2014): Experiments
  - ✓ Findings
    - A simple model with static vectors performs well, implying that the pre-trained vectors are good and universal feature extractors and can be utilized across datasets
    - Fine-tuning the pre-trained vector for each task gives further improvements
    - Multi-channel (static + non-static) does not work well as the author expected
    - Dropout proved to be a good regularizer; one can use a larger than necessary network and simply let dropout regularize it
      - Dropout consistently added 2%-4% relative performance
    - Adadelta gave similar results to Adagrad but required fewer epochs

CNN for Text Classification

- Character-level CNN (Zhang et al., 2015)
  - ✓ A total of 70 characters are quantized
    - ▪ 26 English letters, 10 digits, and 33 other special characters (1 space)

abcdefghijklmnopqrstuvwxyz0123456789
-,;.!?:'''/\|_@#$%^&*~`+-=<>()[]{}

  - ✓ Model Structure



CNN for Text Classification

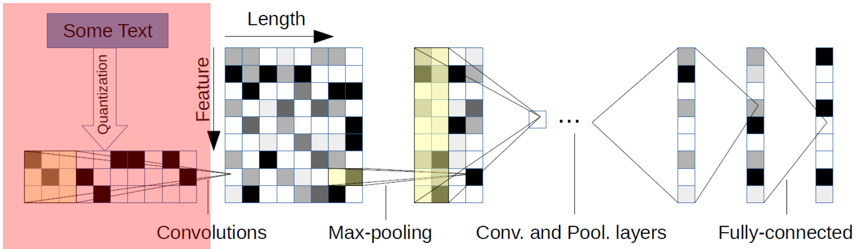
- Character-level CNN (Zhang et al., 2015)
  - ✓ Network structure
    - Large and Small models are designed
      - Convolution layers

Layer	Large Feature	Small Feature	Kernel	Pool
1	1024	256	7	3
2	1024	256	7	3
3	1024	256	3	N/A
4	1024	256	3	N/A
5	1024	256	3	N/A
6	1024	256	3	3

      - Fully connected layers

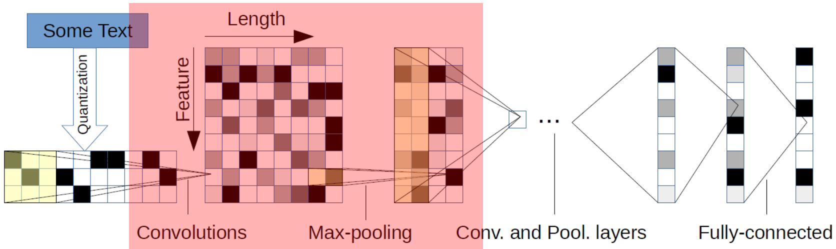
Layer	Output Units Large	Output Units Small
7	2048	1024
8	2048	1024
9	Depends on the problem	

    - Input: 70 by 1014 matrix
      - Each column in an one-hot vector for the corresponding character (not distributed representation)



CNN for Text Classification

- Character-level CNN (Zhang et al., 2015)



  ✓ Convolution

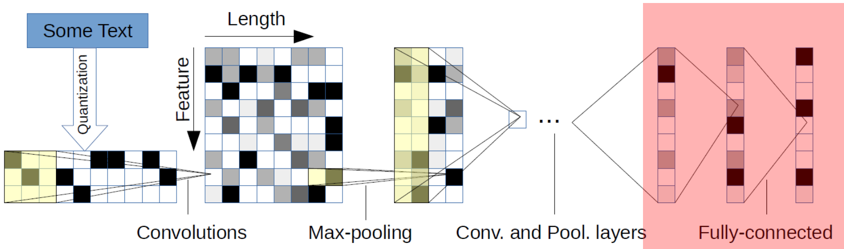
  - Feature size by Kernel size of convolutions are performance
    - (Large model) From the input to the first hidden layer: 70 by 7
    - (Large model) From the first to the second hidden layer: 1024 by 7
    - (Large model) From the second to the third layer: 1024 by 3

  ✓ Max pooling

  - Max pooling with the size of (1 by 3) with the stride of 3 is performed for the first, second, and sixth layers
    - In Yoon Kim (2014), max pooling is performed only once for each feature

CNN for Text Classification

- Character-level CNN (Zhang et al., 2015)



✓ Fully connected layer

- The number of output nodes differs depending on the problem
- Dropout (rate = 0.5) is used between fully connected layers

✓ Data augmentation using thesaurus

- Augmentation methods used for image processing are not appropriate for text processing
- Human can augment text data well, but requires many resources
- Replace a word with its synonym

CNN for Text Classification

- Character-level CNN (Zhang et al., 2015)
  - ✓ Large-scale Datasets

Dataset	Classes	Train Samples	Test Samples	Epoch Size
AG's News	4	120,000	7,600	5,000
Sogou News	5	450,000	60,000	5,000
DBPedia	14	560,000	70,000	5,000
Yelp Review Polarity	2	560,000	38,000	5,000
Yelp Review Full	5	650,000	50,000	5,000
Yahoo! Answers	10	1,400,000	60,000	10,000
Amazon Review Full	5	3,000,000	650,000	30,000
Amazon Review Polarity	2	3,600,000	400,000	30,000

CNN for Text Classification

- Character-level CNN (Zhang et al., 2015)
  - ✓ Classification Performances (Testing error)

Model	AG	Sogou	DBP.	Yelp P.	Yelp F.	Yah. A.	Amz. F.	Amz. P.
BoW	11.19	7.15	3.39	7.76	42.01	31.11	45.36	9.60
BoW TFIDF	10.36	6.55	2.63	6.34	40.14	28.96	44.74	9.00
ngrams	7.96	2.92	1.37	**4.36**	43.74	31.53	45.73	7.98
ngrams TFIDF	**7.64**	**2.81**	**1.31**	4.56	45.20	31.49	47.56	8.46
Bag-of-means	**16.91**	**10.79**	**9.55**	**12.67**	**47.46**	**39.45**	**55.87**	**18.39**
LSTM	13.94	4.82	1.45	5.26	41.83	29.16	40.57	6.10
Lg. w2v Conv.	9.92	4.39	1.42	4.60	40.16	31.97	44.40	5.88
Sm. w2v Conv.	11.35	4.54	1.71	5.56	42.13	31.50	42.59	6.00
Lg. w2v Conv. Th.	9.91	-	1.37	4.63	39.58	31.23	43.75	5.80
Sm. w2v Conv. Th.	10.88	-	1.53	5.36	41.09	29.86	42.50	5.63
Lg. Lk. Conv.	8.55	4.95	1.72	4.89	40.52	29.06	45.95	5.84
Sm. Lk. Conv.	10.87	4.93	1.85	5.54	41.41	30.02	43.66	5.85
Lg. Lk. Conv. Th.	8.93	-	1.58	5.03	40.52	28.84	42.39	5.52
Sm. Lk. Conv. Th.	9.12	-	1.77	5.37	41.17	28.92	43.19	5.51
Lg. Full Conv.	9.85	8.80	1.66	5.25	38.40	29.90	40.89	5.78
Sm. Full Conv.	11.59	8.95	1.89	5.67	38.82	30.01	40.88	5.78
Lg. Full Conv. Th.	9.51	-	1.55	4.88	38.04	29.58	40.54	5.51
Sm. Full Conv. Th.	10.89	-	1.69	5.42	**37.95**	29.90	40.53	5.66
Lg. Conv.	12.82	4.88	1.73	5.89	39.62	29.55	41.31	5.51
Sm. Conv.	15.65	8.65	1.98	6.53	40.84	29.84	40.53	5.50
Lg. Conv. Th.	13.39	-	1.60	5.82	39.30	**28.80**	40.45	**4.93**
Sm. Conv. Th.	14.80	-	1.85	6.49	40.16	29.84	**40.43**	5.67

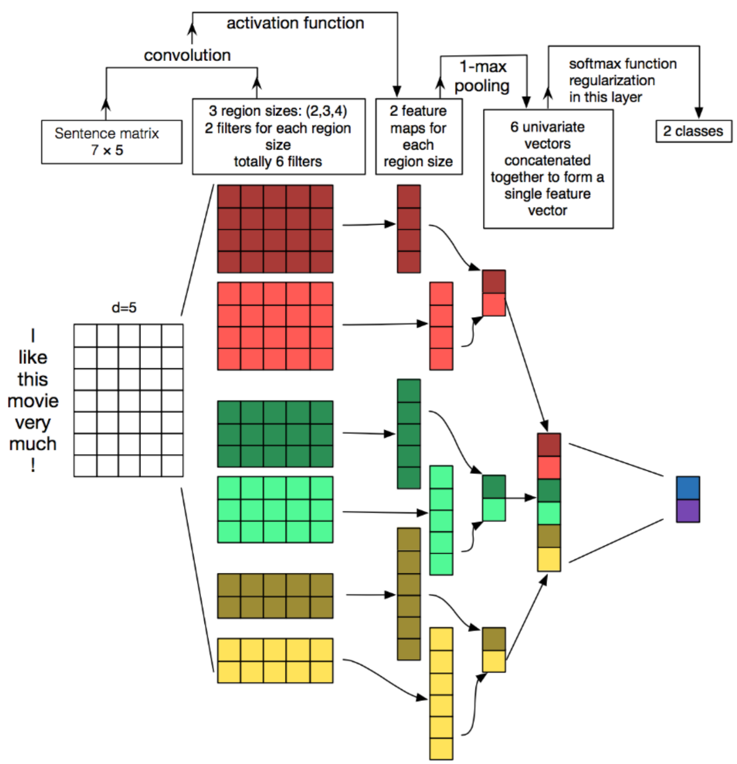
CNN for Text Classification

- Character-level CNN (Zhang et al., 2015)
  - ✓ Findings
    - Character-level CNN is an effective method
    - Dataset size forms a dichotomy between traditional and CNN models
    - CNN may work well for user-generated data
    - Choice of alphabet makes a difference
    - Semantics of tasks may not matter
    - Bag-of-means is a misuse of word2vec
    - There is no free lunch

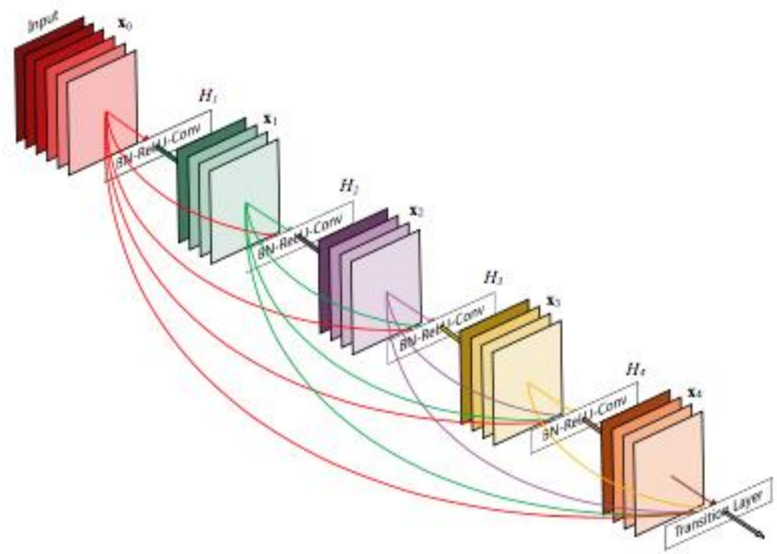
CNN for Text Classification

- What depth is sufficient for text classification? (Le et al. 2018)

Shallow but wide (Yoon Kim (2014))



Deep and narrow (Densenet)



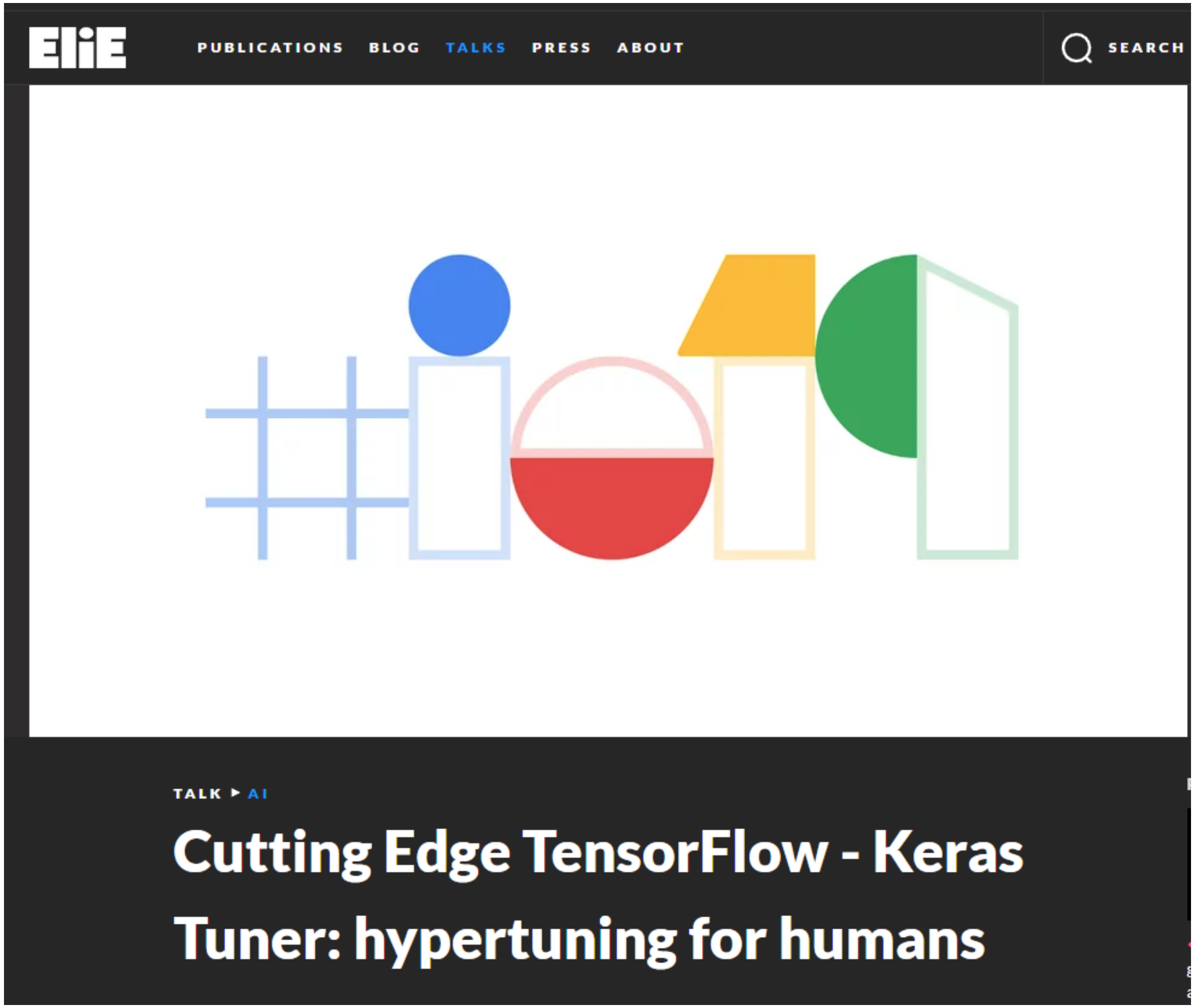
CNN for Text Classification

- Findings
  - ✓ Deep and narrow models yield better performances for character-level representation
  - ✓ Shallow and wide models yield better performance for word-level representations

Models	AGNews	Yelp Bin	Yelp Full	DBPedia	Yahoo
Char shallow-and-wide CNN	90.7	94.4	60.3	98.0	70.2
Char-DenseNet $N_b = (4-4-4-4)$ Global Average-Pooling	90.4	94.2	61.1	97.7	68.8
Char-DenseNet $N_b = (10-10-4-4)$ Global Average-Pooling	90.6	94.9	62.1	98.2	70.5
Char-DenseNet $N_b = (4-4-4-4)$ Local Max-Pooling	90.5	95.0	63.6	98.5	72.9
Char-DenseNet $N_b = (10-10-4-4)$ Local Max-Pooling	92.1	95.0	64.1	98.5	73.4
Word shallow-and-wide CNN	92.2	**95.9**	**64.9**	**98.7**	73.0
Word-DenseNet $N_b = (4-4-4-4)$ Global Average-Pooling	91.7	95.8	64.5	**98.7**	70.4*
Word-DenseNet $N_b = (10-10-4-4)$ Global Average-Pooling	91.4	95.5	63.6	98.6	70.2*
Word-DenseNet $N_b = (4-4-4-4)$ Local Max-Pooling	90.9	95.4	63.0	98.0	67.6*
Word-DenseNet $N_b = (10-10-4-4)$ Local Max-Pooling	88.8	95.0	62.2	97.3	68.4*
bag of words (Zhang, Zhao, and LeCun 2015)	88.8	92.2	58.0	96.6	68.9
ngrams (Zhang, Zhao, and LeCun 2015)	92.0	95.6	56.3	98.6	68.5
ngrams TFIDF (Zhang, Zhao, and LeCun 2015)	92.4	95.4	54.8	**98.7**	68.5
fastText (Joulin et al. 2016)	**92.5**	95.7	63.9	98.6	72.3
char-CNN (Zhang, Zhao, and LeCun 2015)	87.2	94.7	62.0	98.3	71.2
char-CRNN (Xiao and Cho 2016)	91.4	94.5	61.8	98.6	71.7
very deep char-CNN (Conneau et al. 2016)	91.3	95.7	64.7	**98.7**	73.4
Naive Bayes (Yogatama et al. 2017)	90.0	86.0	51.4	96.0	68.7
Kneser-Ney Bayes (Yogatama et al. 2017)	89.3	81.8	41.7	95.4	69.3
MLP Naive Bayes (Yogatama et al. 2017)	89.9	73.6	40.4	87.2	60.6
Discriminative LSTM (Yogatama et al. 2017)	92.1	92.6	59.6	**98.7**	**73.7**
Generative LSTM-independent comp. (Yogatama et al. 2017)	90.7	90.0	51.9	94.8	70.5
Generative LSTM-shared comp. (Yogatama et al. 2017)	90.6	88.2	52.7	95.4	69.3

CNN for Text Classification

- Automatic hyper-parameter tuning? (Google I/O'19)



https://elie.net/talk/cutting-edge-tensorflow-keras-tuner-hypertuning-for-humans/

CNN for Text Classification

- Automatic hyper-parameter tuning?

MNIST hypermodel is as easy as 1,2,3

1. Wrap model in a function

```
def model_fn():
```

2. Define hyper-parameters

```
    LR = Choice('learning_rate', [0.001, 0.0005, 0.0001], group='optimizer')
    DROPOUT_RATE = Linear('dropout_rate', 0.0, 0.5, 5, group='dense')
    NUM_DIMS = Range('num_dims', 8, 32, 8, group='dense')
    NUM_LAYERS = Range('num_layers', 1, 3, group='dense')
    L2_NUM_FILTERS = Range('l2_num_filters', 8, 64, 8, group='cnn')
    L1_NUM_FILTERS = Range('l1_num_filters', 8, 64, 8, group='cnn')
```

CNN for Text Classification

- Automatic hyper-parameter tuning?

MNIST hypermodel is as easy as 1,2,3

1. Wrap model in a function

2. Define hyper-parameters

3. Replace static value with hyper-parameters

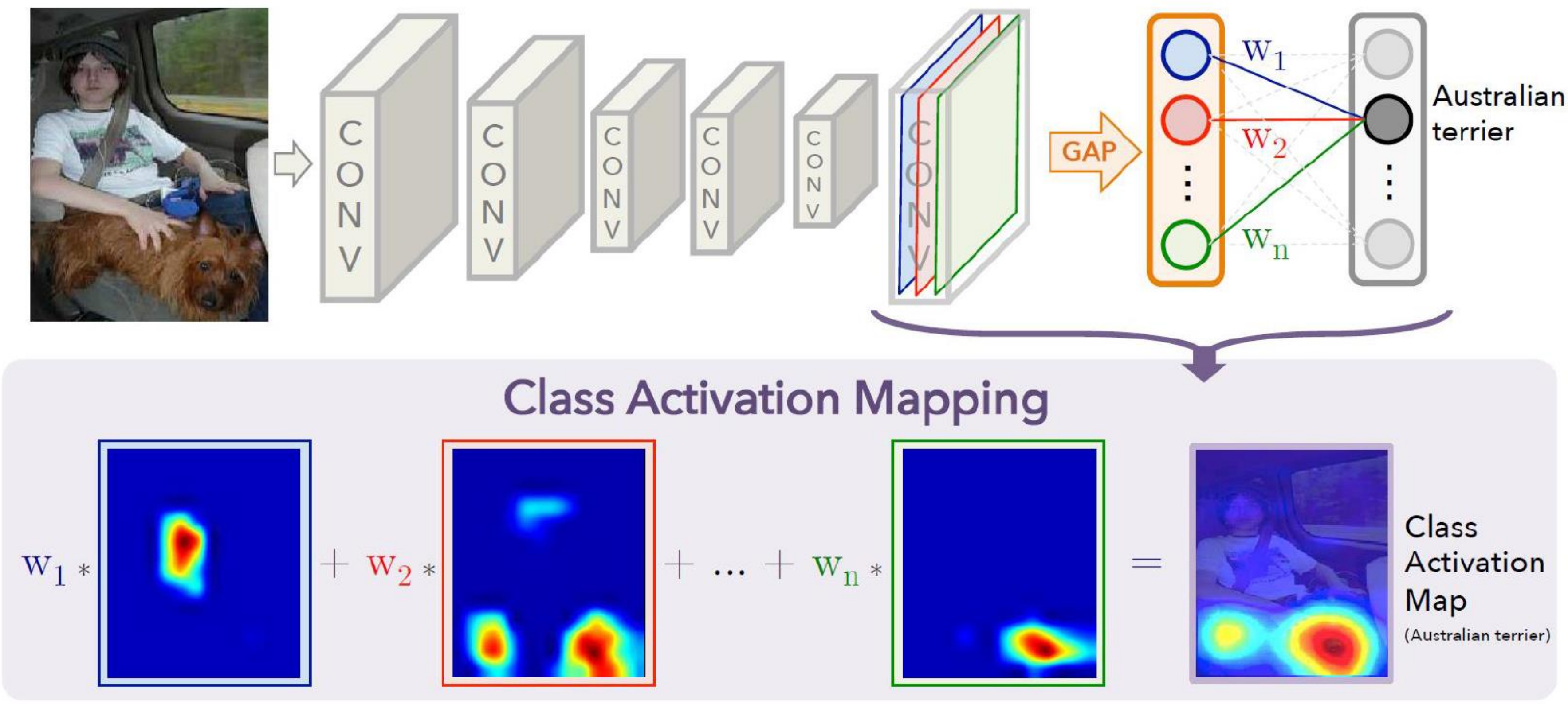
```
def model_fn():

    LR = Choice('learning_rate', [0.001, 0.0005, 0.0001], group='optimizer')
    DROPOUT_RATE = Linear('dropout_rate', 0.0, 0.5, 5, group='dense')
    NUM_DIMS = Range('num_dims', 8, 32, 8, group='dense')
    NUM_LAYERS = Range('num_layers', 1, 3, group='dense')
    L2_NUM_FILTERS = Range('l2_num_filters', 8, 64, 8, group='cnn')
    L1_NUM_FILTERS = Range('l1_num_filters', 8, 64, 8, group='cnn')

    model = Sequential()
    model.add(Conv2D(L1_NUM_FILTERS, kernel_size=(3, 3), activation='relu'))
    model.add(Conv2D(L2_NUM_FILTERS, kernel_size=(3, 3), activation='relu'))
    model.add(Flatten())
    for _ in range(NUM_LAYERS):
      model.add(Dense(NUM_DIMS, activation='relu'))
      model.add(Dropout(DROPOUT_RATE))
    model.add(Dense(10, activation='softmax'))
    model.compile(loss='categorical_crossentropy', optimizer=Adam(LR))
    return model
```

CNN for Text Classification: Localization

- Image Localization
  - ✓ Class Activation Map (CAM) (Zhou et al., 2016)
    - Localize significant areas based only on class label information

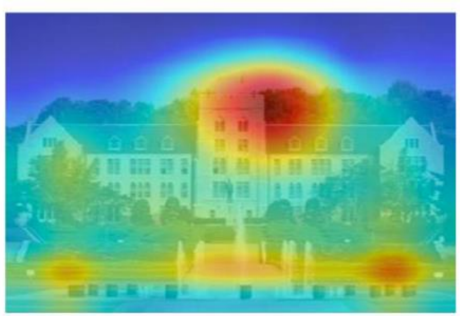


CNN for Text Classification: Localization

- Image Localization
  - ✓ Class Activation Map (CAM)
    - ▪ Localize significant areas based only on class label information

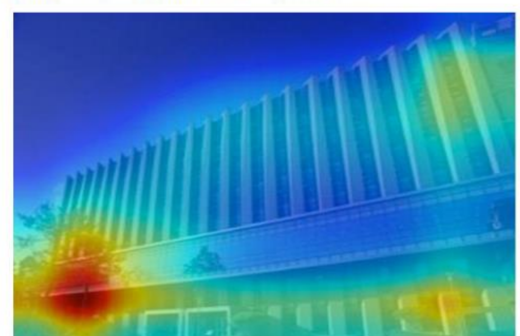
Predictions:

- **Type of environment:** outdoor
- **Semantic categories:** palace:0.23, formal_garden:0.20, mansion:0.15, castle:0.07, courthouse:0.06
- **SUN scene attributes:** man-made, openarea, naturallight, grass, vegetation, foliage, leaves, directsunsunny, trees, vacationingtouring
- **Informative region for the category *palace* is:**



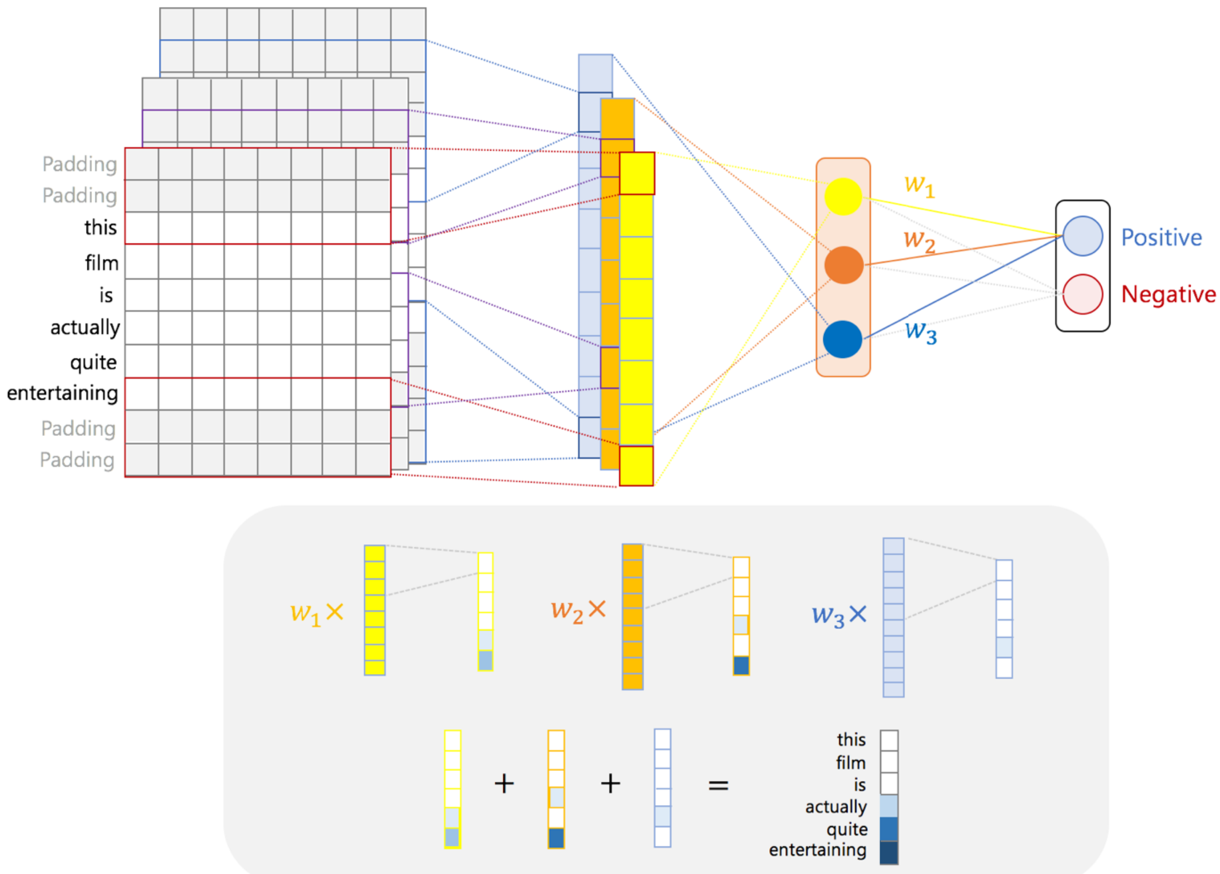
Predictions:

- **Type of environment:** outdoor
- **Semantic categories:** office_building:0.33, building_facade:0.24, hospital:0.17, skyscraper:0.06,
- **SUN scene attributes:** man-made, naturallight, openarea, mostlyverticalcomponents, directsunsunny, clouds, glass, mostlyhorizontalcomponents, semi-enclosedarea, metal
- **Informative region for the category *office_building* is:**



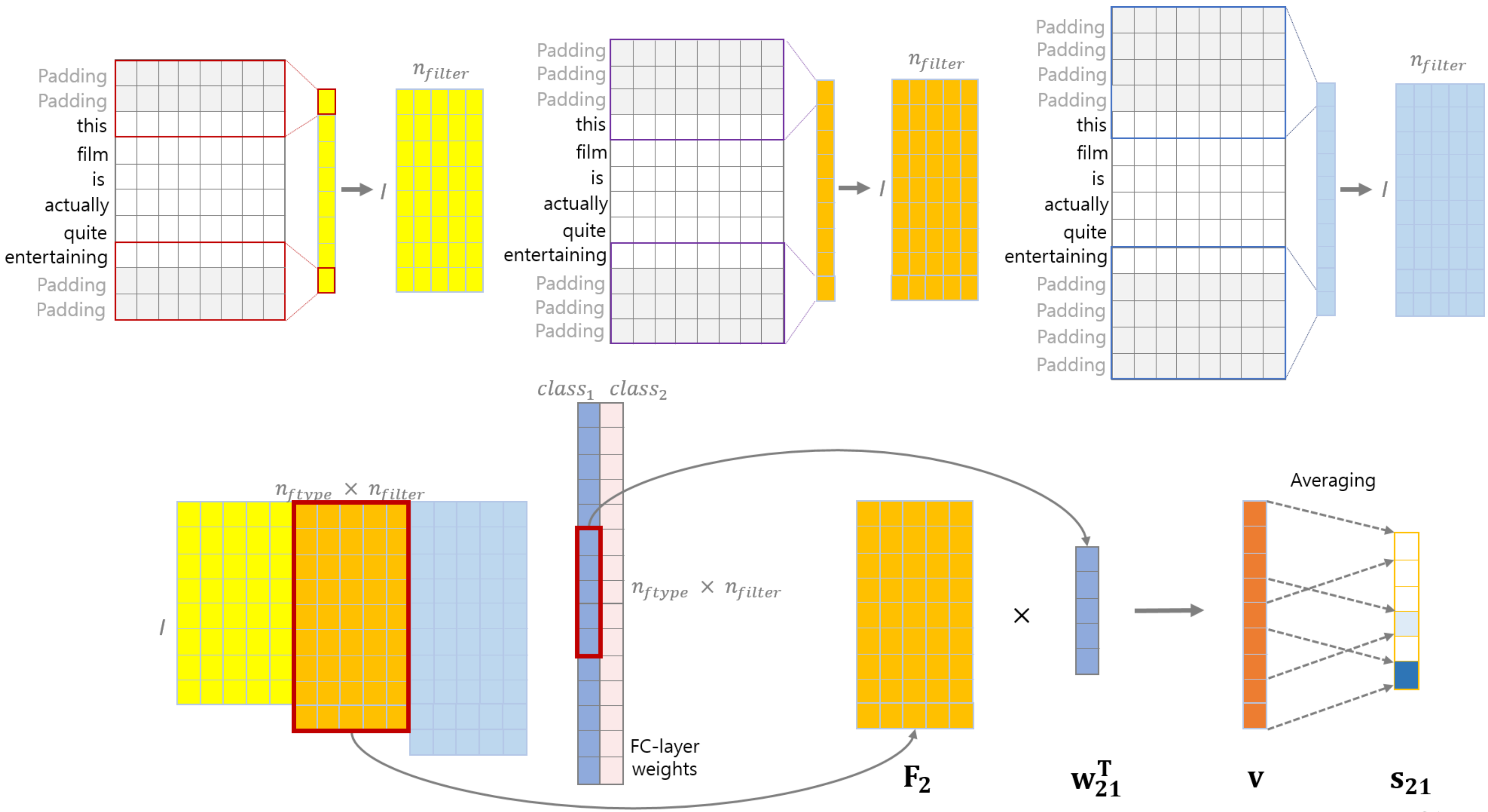
CNN for Text Classification: Localization

- Class Activation Map (CAM) for Sentiment Classification (Lee et al., 2018)



CNN for Text Classification: Localization

- Class Activation Map (CAM) for Sentiment Classification



CNN for Text Classification: Localization

- Class Activation Map (CAM) for Sentiment Classification
  - ✓ Two datasets: IMDB (English), WATCHA (Korean)

Table 1. Rating distribution for IMDB dataset

Rating	**1**	**2**	**3**	**4**	**7**	**8**	**9**	**10**
Reviews	10,122	4,586	4,961	5,531	4,803	5,859	4,607	9,731
Class	Negative				Positive			

Table 2. Rating distribution for the WATCHA dataset

Rating	**0.5**	**1**	**1.5**	**2**	**2.5**	**3**	**3.5**	**4**	**4.5**	**5.0**
Reviews	50,660	66,184	62,094	163,272	173,650	411,757	424,378	652,250	297,327	416,096
Class	Negative				Not used					Positive

Table 3. Information on datasets

Dataset	Number of tokens	Average number of words per document	Maximum number of words per document
IMDB	115,205	227.03	2,192
WATCHA	424,027	9.52	1,853

Table 4. CNN hyper-parameters

Filter type (window size)	**N. of Filters**	**Learning rate**	**Input Dimension**	**Dropout rate**	**L_2 regularization (λ)**	**Optimizer**	**Batch size**
3 (tri-gram) 4 (quad-gram) 5 (5-gram)	128 each	0.001	100	0.5	0.1	Adam	64

CNN for Text Classification: Localization

- Class Activation Map (CAM) for Sentiment Classification

Table 9. Frequently appearing words in positive/negative sentences in the IMDB test dataset (semantically positive or negative words are in blue and red fonts, respectively).

Positive					Negative				
CAM²-Rand	*CAM²-Static*	*CAM²-Non-Static*	*CAM²-2channel*	*CAM²-4channel*	*CAM²-Rand*	*CAM²-Static*	*CAM²-Non-Static*	*CAM²-2channel*	*CAM²-4channel*
and	and	and	and	and	the	is	the	the	and
is	great	is	is	is	and	and	and	and	the
the	very	excellent	excellent	the	worst	was	worst	worst	worst
excellent	a	the	the	a	of	the	of	of	of
a	is	a	a	excellent	a	bad	a	a	is
great	well	perfect	great	perfect	is	just	is	is	a
perfect	as	great	perfect	amazing	boring	this	awful	awful	awful
amazing	it	amazing	wonderful	enjoyed	waste	acting	waste	waste	waste
wonderful	good	wonderful	amazing	great	awful	plot	boring	boring	boring
of	i	enjoyed	enjoyed	of	was	a	was	was	was
s	film	i	as	i	this	of	this	this	this
i	wonderful	of	hilarious	well	to	boring	bad	to	bad
enjoyed	excellent	as	superb	it	i	script	movie	i	movie
superb	beautiful	hilarious	of	wonderful	movie	awful	to	movie	to
it	of	superb	i	s	bad	waste	terrible	terrible	i
hilarious	the	s	s	superb	terrible	movie	i	bad	acting
today	favorite	an	fun	hilarious	poor	stupid	poor	poor	poor
an	movie	well	today	fun	poorly	terrible	poorly	poorly	poorly
loved	comedy	definitely	well	loved	in	so	horrible	in	s
in	fun	it	an	an	s	no	dull	acting	just
job	in	enjoyable	was	very	it	completely	stupid	s	stupid
was	story	today	it	to	just	lame	acting	worse	in
as	my	was	definitely	was	film	poor	worse	dull	scirpt
fun	worth	very	job	today	with	are	s	with	horrible
to	highly	with	enjoyable	definitely	are	i	script	film	dull
with	enjoyed	surprised	in	with	acting	that	in	script	film
touching	loved	in	with	most	script	an	film	just	it
enjoyable	entertaining	fun	movie	enjoyable	stupid	or	lame	it	as
movie	also	entertaining	loved	good	save	crap	just	stupid	worse

CNN for Text Classification: Localization

- Class Activation Map (CAM) for Sentiment Classification

Positive					Negative				
CAM²-Rand	*CAM²-Static*	*CAM²-Non-Static*	*CAM²-2channel*	*CAM²-4channel*	*CAM²-Rand*	*CAM²-Static*	*CAM²-Non-Static*	*CAM²-2channel*	*CAM²-4channel*
영화	영화	영화	영화	영화	영화	영화	영화	영화	영화
최고의 (best)	수	그	수	너무	너무	너무	너무	너무	너무
수	최고의 (best)	너무	그	최고의 (best)	왜	왜	왜	왜	왜
그	그	최고의 (best)	너무	그	이	그냥	이	그냥	이
정말	이	그리고	최고의 (best)	수	그냥	없고 (none)	그냥	없고 (none)	없고 (none)
그리고	정말	수	그리고	그리고	더	없는 (none)	없고 (none)	이	그냥
너무	너무	더	이	정말	그	없다 (none)	없는 (none)	없다 (none)	없는 (none)
이	다시	가장	정말	진짜	없고 (none)	더	영화는	영화는	없다 (none)
가장	잘	정말	더	이	없는 (none)	느낌	그	없는 (none)	영화는
더	가장	다시	가장	가장	수	영화는	영화를	영화를	더
다시	그리고	진짜	잘	더	영화는	좀	영화가	다	영화가
잘	있는	최고 (best)	있는	잘	이런	뻔한 (obvious)	수	그	이런
최고 (best)	진짜	있는	최고 (best)	최고 (best)	없다 (none)	이	정말	더	수
내	내	이	보고	다시	영화를	영화가	없다 (none)	이런	영화를
다	모든	내	내	있는	정말	안 (not)	다	보는	내가
진짜	좋다 (good)	잘	진짜	좋다 (good)	것	차라리 (rather)	그나마	그나마	보는
완벽한 (perfect)	더	보고	다시	다	내가	것	내가	안 (not)	정말
있는	영화를	대한	영화를	보고	영화가	스토리	더	정말	것
영화를	또	내내	모든	완벽한 (perfect)	건	내	내	좀	그
본	아름다운 (beautiful)	한	다	내	다	영화를	좀	진짜	내
모든	보고	마지막	대한	봐도	이렇게	보는	이런	영화가	다
것	내가	모든	내가	모든	좀	이런	잘	것	스토리
보고	최고 (best)	것	마지막	마지막	한	건	건	차라리 (rather)	이렇게
한	한	내가	한	아름다운 (beautiful)	보는	무슨	봤는데	내	건
좋다 (good)	함께	또	것	또	느낌	듯	모르겠다	건	별
대한	꼭	다	아름다운 (beautiful)	영화를	진짜	모르겠다	것	만든	한
없는 (none)	작품	완벽한 (perfect)	영화가	좋았다 (good)	않는 (not)	전개	이렇게	수	좀
영화가	마지막	이런	이렇게	본	대한	전혀 (never)	차라리 (rather)	한	차라리 (rather)
봐도	완벽한 (perfect)	아름다운 (beautiful)	완벽한 (perfect)	것	내	본	안 (not)	내가	전혀 (never)

CNN for Text Classification: Localization

- Class Activation Map (CAM) for Sentiment Classification
  - ✓ Localization example for the IMDB dataset

CAM²-4channel	Seeing as the vote average was pretty low and the fact that the clerk in the video store thought was just OK I didn t t have much expectations when renting this film But contrary to the above I **enjoyed it** a lot This is acharming movie It didn t need to grow on me **I enjoyed it** from the beginning Mel Brooks gives a great performance as thelead character I think somewhat different from his usual persona in his movies There s not a lot of knockout jokes or somethinglike that but there are some rather hilarious scenes and overall this is a **very enjoyable and very easy** to watch film **Very recommended** **Positive**
CAM²-4channel	I hate this movie It is a horrid movie Sean Young s character is completely unsympathetic Her performance is wooden at best The storyline is completely **predictable and completely uninteresting** I would never recommend this film to anyone It is one of the **worst** movies I have ever had the misfortune to see **Negative**

CNN for Text Classification: Localization

- Class Activation Map (CAM) for Sentiment Classification
  - ✓ Localization example for the WATCH dataset

CAM²-4channel	여지껏 봤던 영화중 단연 최고라고 할만한 작품이다 스토리 배경 시대배경과 영화배경 모두 감독의 카메라 기법 배우의 연기력 뭐하나 빠지는 것이 없다 Positive
CAM²-4channel	예술의 기본은 낯설게 하기다 그런 시도조차 보이지 않는다는 점 영화 중간중간 나오는 조명이 엄청나게 거슬렸다는 점이 너무 싫었다 Negative


ANY
questions ?