



Natural Language Processing; Text Classification



[YouTube Playlist](#)

Maziar Raissi

Assistant Professor

Department of Applied Mathematics

University of Colorado Boulder

maziar.raissi@colorado.edu

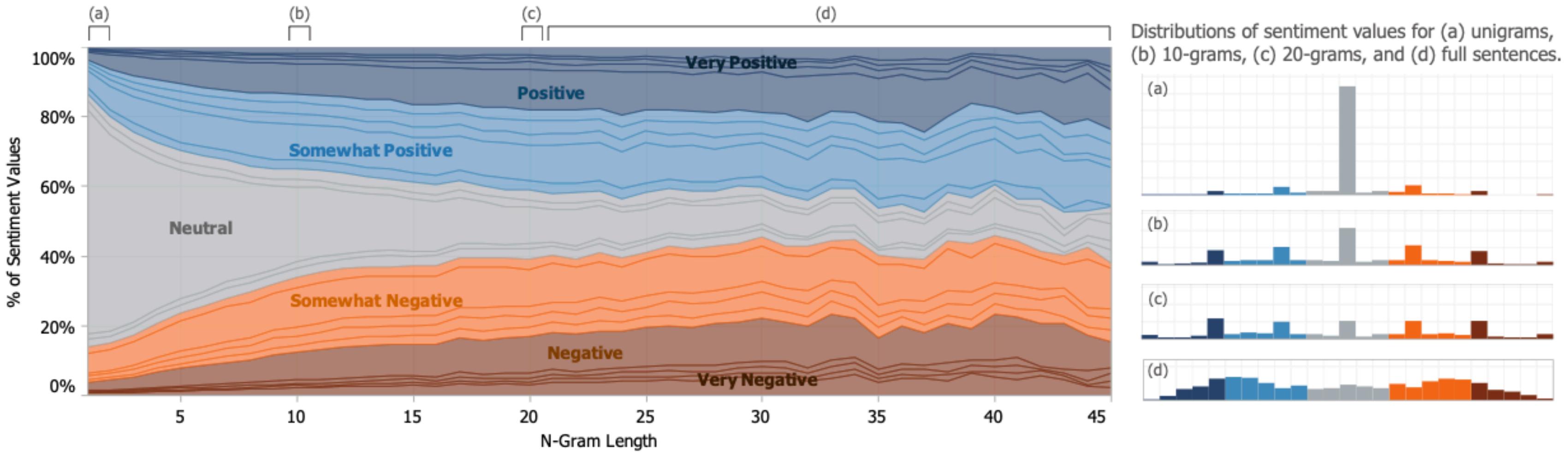
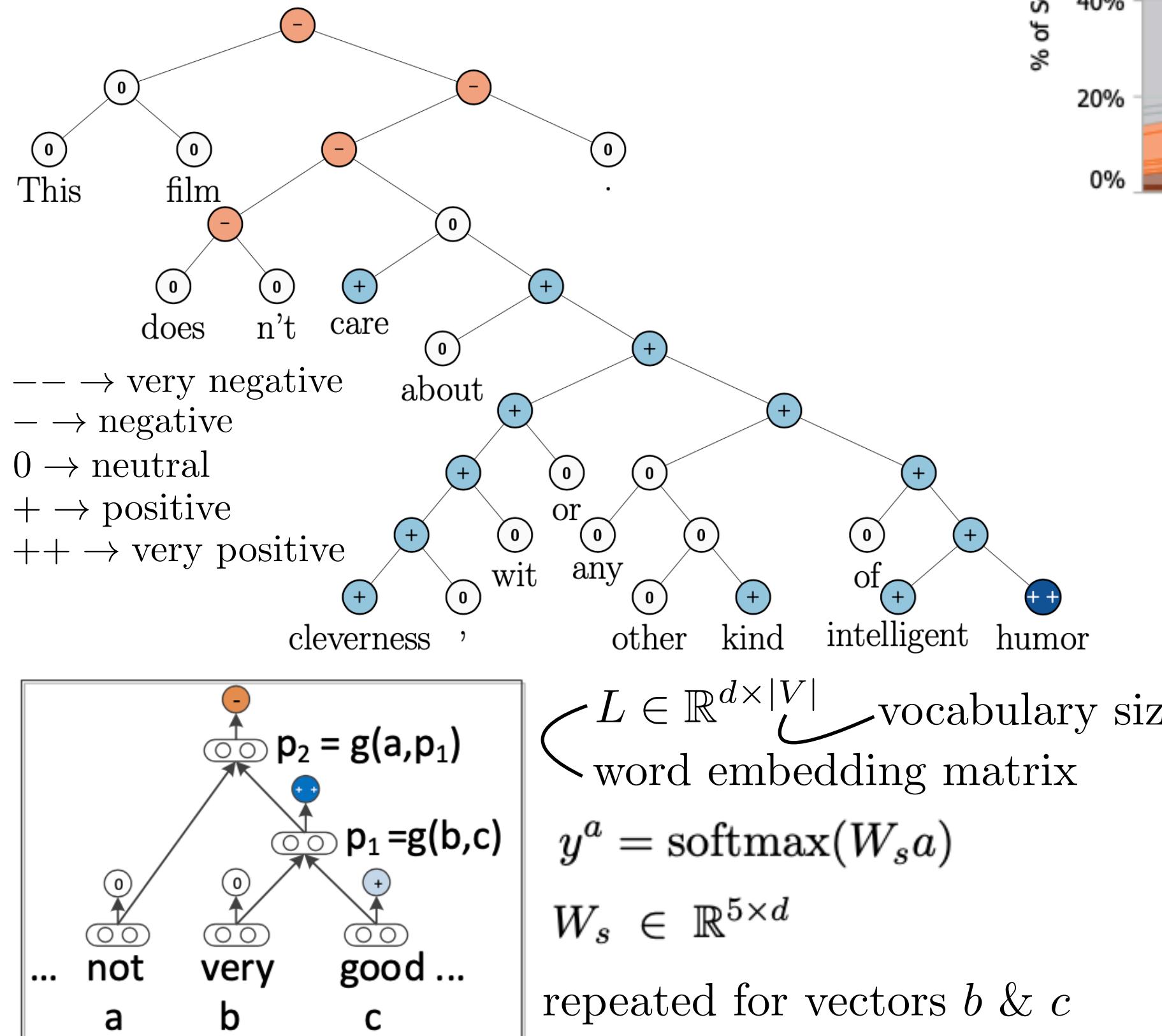
Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank


[YouTube Video](#)

Stanford Sentiment Treebank

11,855 single sentences from movie reviews
<https://www.rottentomatoes.com>

215,154 unique phrases from parse trees
 obtained from Stanford parser



Recursive Neural Network

$$p_1 = f\left(W \begin{bmatrix} b \\ c \end{bmatrix}\right), p_2 = f\left(W \begin{bmatrix} a \\ p_1 \end{bmatrix}\right)$$

$$f = \tanh, W \in \mathbb{R}^{d \times 2d}$$

Matrix-Vector Recursive NN

$$(p_2, P_2)$$

$$(a, A) \quad (p_1, P_1)$$

$$(b, B) \quad (c, C)$$

$$p_1 = f\left(W \begin{bmatrix} Cb \\ Bc \end{bmatrix}\right), P_1 = f\left(W_M \begin{bmatrix} B \\ C \end{bmatrix}\right)$$

word's matrix

Recursive Neural Tensor Network

$$h = \left[\begin{array}{c} b \\ c \end{array} \right]_{2d \times 2d \times d}^T V^{[1:d]} \left[\begin{array}{c} b \\ c \end{array} \right]; h_i = \left[\begin{array}{c} b \\ c \end{array} \right]_{2d \times 2d}^T V^{[i]} \left[\begin{array}{c} b \\ c \end{array} \right]$$

$$p_1 = f\left(\left[\begin{array}{c} b \\ c \end{array} \right]^T V^{[1:d]} \left[\begin{array}{c} b \\ c \end{array} \right] + W \left[\begin{array}{c} b \\ c \end{array} \right]\right)$$

$$p_2 = f\left(\left[\begin{array}{c} a \\ p_1 \end{array} \right]^T V^{[1:d]} \left[\begin{array}{c} a \\ p_1 \end{array} \right] + W \left[\begin{array}{c} a \\ p_1 \end{array} \right]\right)$$

$$E(\theta) = \sum_i \sum_j t_j^i \log y_j^i + \lambda \|\theta\|^2$$

$$\theta = (V, W, W_s, L)$$

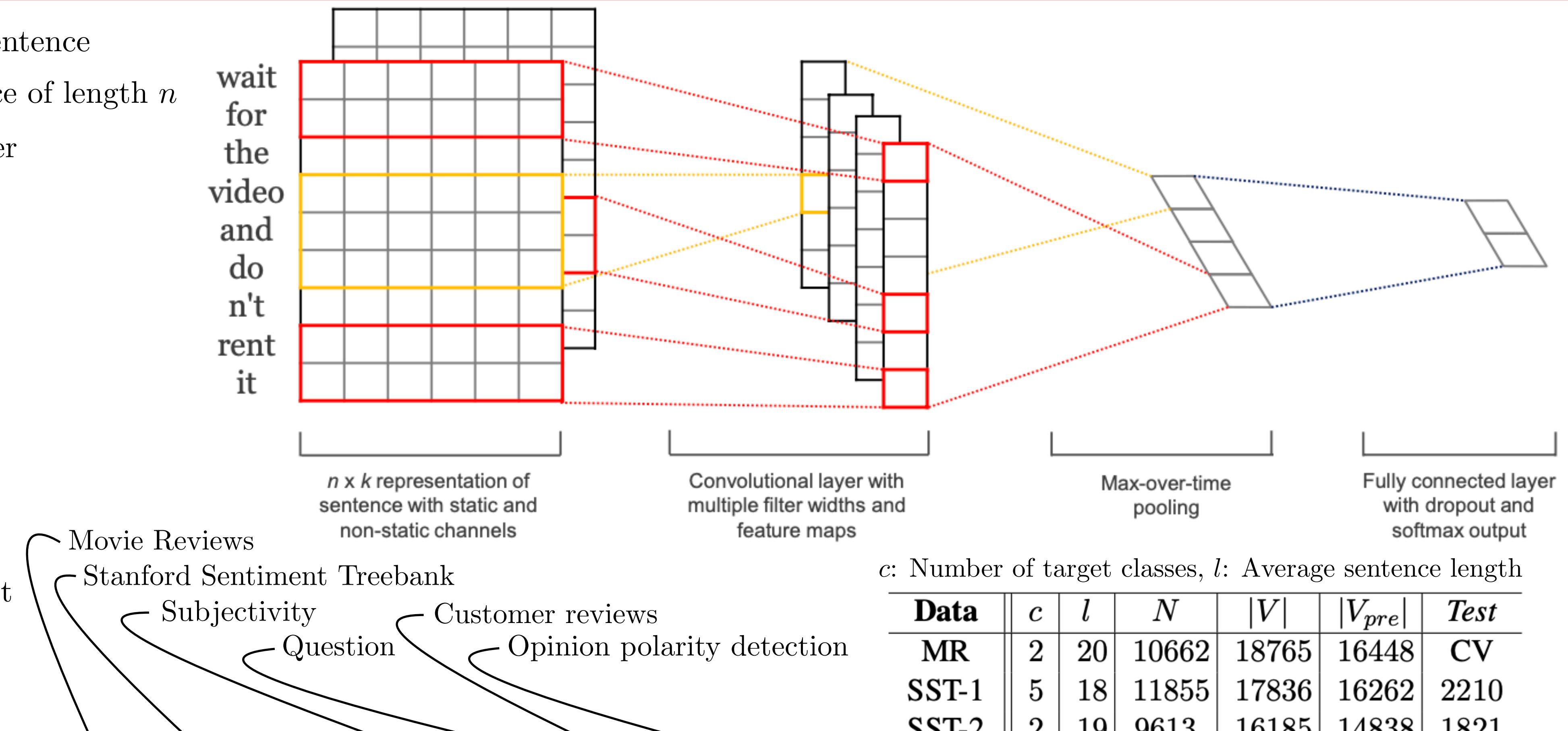
$$y^i \in \mathbb{R}^{C \times 1} \rightarrow \text{predicted distribution at node } i$$

$$t^i \in \mathbb{R}^{C \times 1} \rightarrow \text{target distribution at node } i$$

Convolutional Neural Networks for Sentence Classification

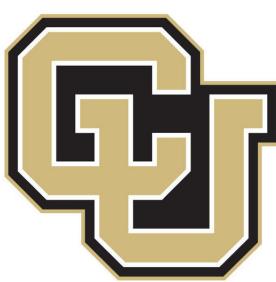

[YouTube Playlist](#)
 $x_i \in \mathbb{R}^k \rightarrow i\text{-th word in the sentence}$
 $x = (x_1, x_2, \dots, x_n) \rightarrow \text{sentence of length } n$
 $W = (W_1, W_2, \dots, W_h) \rightarrow \text{filter}$
 $W_j \in \mathbb{R}^{m \times k} \quad b \in \mathbb{R}^m$
 $c_i = f\left(\sum_{j=1}^h W_j x_{i+j-1} + b\right)$
 $c_i \in \mathbb{R}^m$
 $c = (c_1, c_2, \dots, c_{n-h+1})$
 $z = \max_i c_i$
 $y = W^y z + b^y$
 $y = W^y(z \odot r) + b^y \rightarrow \text{dropout}$
 $\Pr(r=1) = p = 1 - \Pr(r=0)$
 $\hat{W}^y = pW^y \rightarrow \text{during testing}$

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



c : Number of target classes, l : Average sentence length

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



Boulder

Distributed Representations of Sentences and Documents



[YouTube Video](#)

Bag-of-words weaknesses:

- 1) lose ordering of words
- 2) ignore semantics of words

“powerful” should be closer to “strong” than “Paris”

Vector Representation of Words

$W \rightarrow$ word matrix

$w_1, w_2, \dots, w_T \rightarrow$ sequence of training words

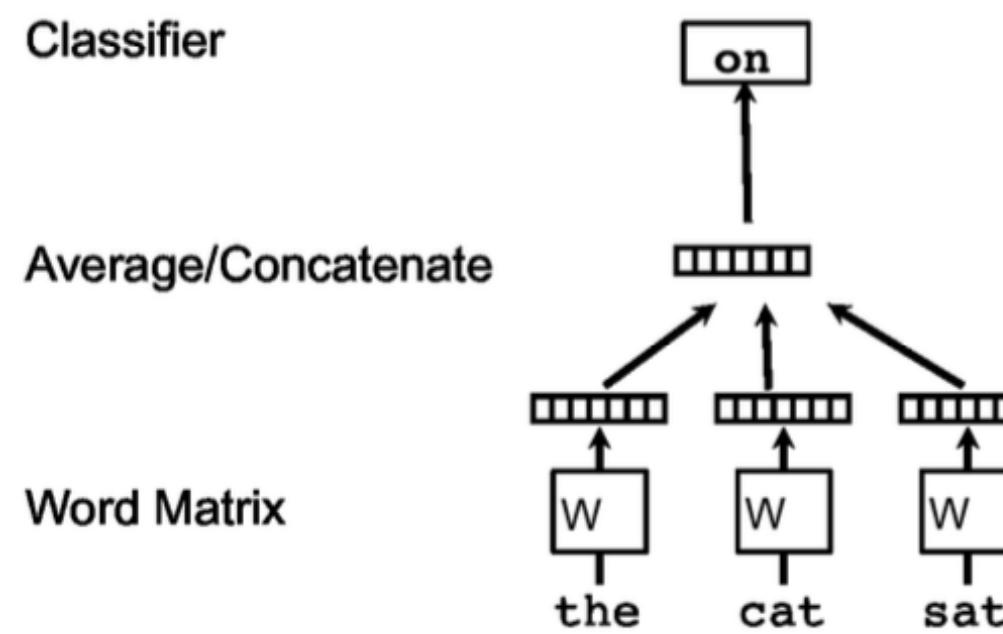
$$\frac{1}{T} \sum_{t=k}^{T-k} \log p(w_t | w_{t-k}, \dots, w_{t+k})$$

$$p(w_t | w_{t-k}, \dots, w_{t+k}) = \frac{e^{y_{w_t}}}{\sum_i e^{y_i}}$$

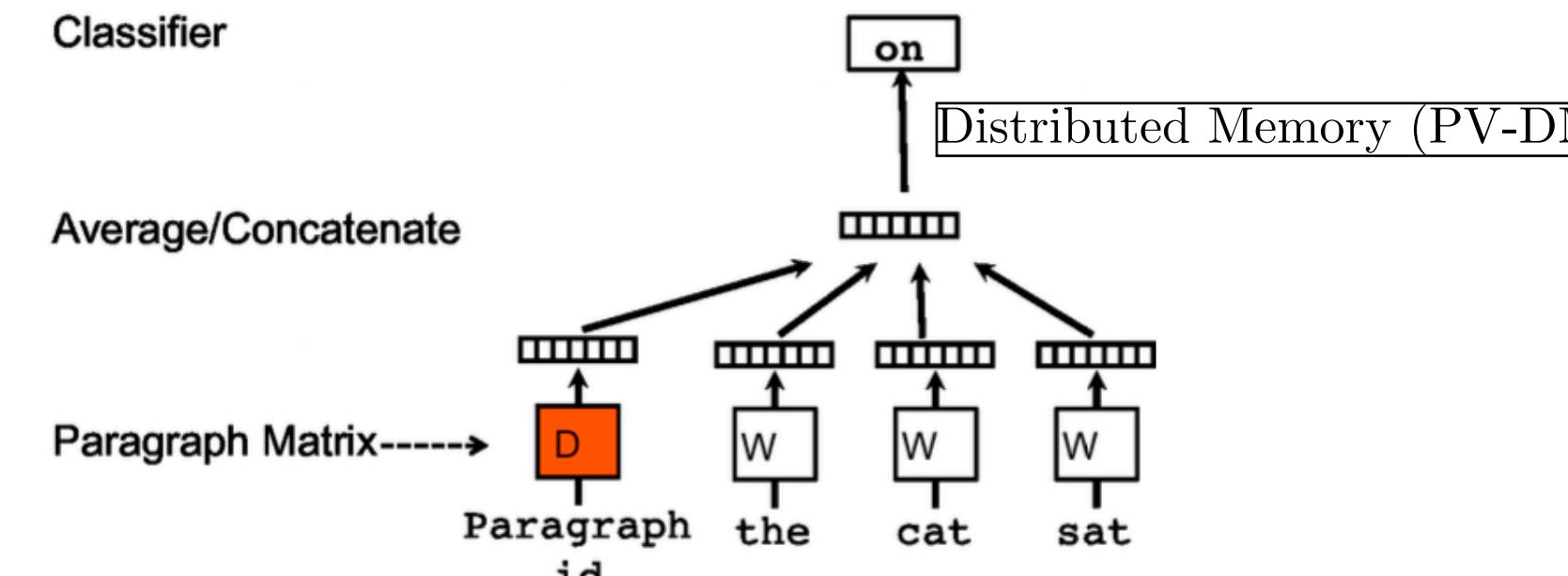
$$y = b + \underbrace{Uh(w_{t-k}, \dots, w_{t+k}; W)}$$

concatenation or average of word vectors extracted from W

hierarchical softmax & Huffman coding



Paragraph Vector



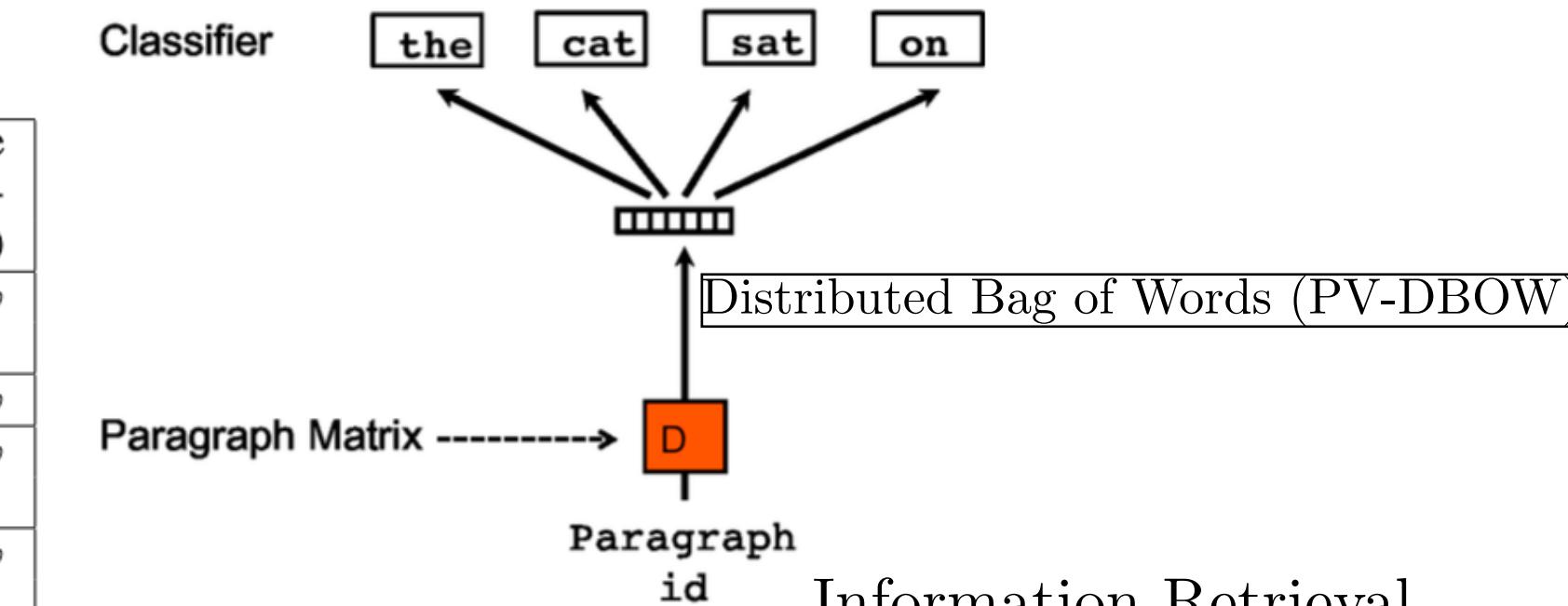
unsupervised training: optimize for D , W , U , b

inference: optimize for D_{test} given W , U , b

logistic classifier or support vector machines on D_{test}

Stanford Sentiment Treebank

Model	Error rate (Positive/Negative)	Error rate (Fine-grained)
Naïve Bayes (Socher et al., 2013b)	18.2 %	59.0%
SVMs (Socher et al., 2013b)	20.6%	59.3%
Bigram Naïve Bayes (Socher et al., 2013b)	16.9%	58.1%
Word Vector Averaging (Socher et al., 2013b)	19.9%	67.3%
Recursive Neural Network (Socher et al., 2013b)	17.6%	56.8%
Matrix Vector-RNN (Socher et al., 2013b)	17.1%	55.6%
Recursive Neural Tensor Network (Socher et al., 2013b)	14.6%	54.3%
Paragraph Vector	12.2%	51.3%



IMDB dataset

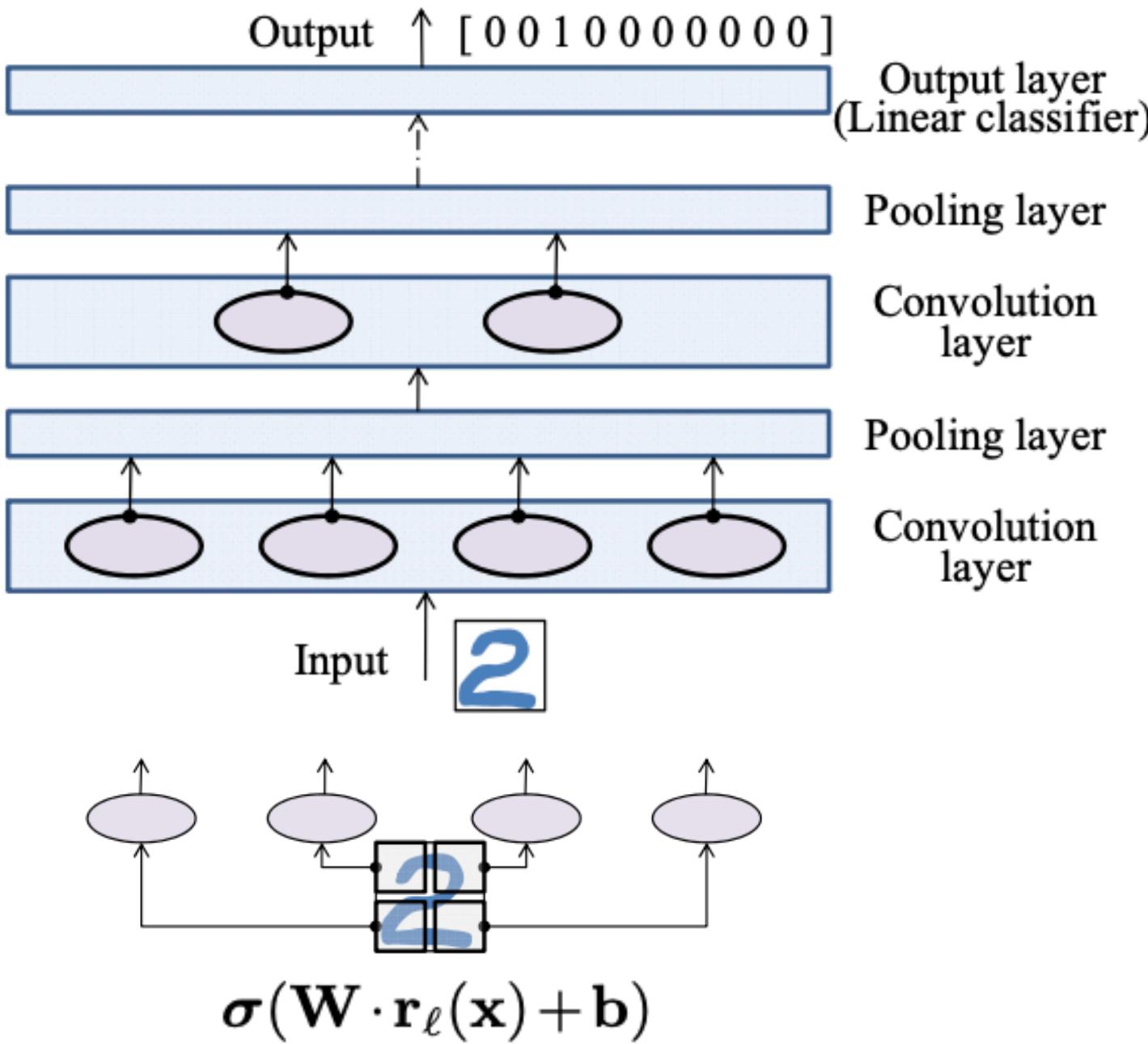
Model	Error rate
BoW (bnc) (Maas et al., 2011)	12.20 %
BoW ($b\Delta t'c$) (Maas et al., 2011)	11.77%
LDA (Maas et al., 2011)	32.58%
Full+BoW (Maas et al., 2011)	11.67%
Full+Unlabeled+BoW (Maas et al., 2011)	11.11%
WRRBM (Dahl et al., 2012)	12.58%
WRRBM + BoW (bnc) (Dahl et al., 2012)	10.77%
MNB-uni (Wang & Manning, 2012)	16.45%
MNB-bi (Wang & Manning, 2012)	13.41%
SVM-uni (Wang & Manning, 2012)	13.05%
SVM-bi (Wang & Manning, 2012)	10.84%
NBSVM-uni (Wang & Manning, 2012)	11.71%
NBSVM-bi (Wang & Manning, 2012)	8.78%
Paragraph Vector	7.42%

Model	Error rate
Vector Averaging	10.25%
Bag-of-words	8.10 %
Bag-of-bigrams	7.28 %
Weighted Bag-of-bigrams	5.67%
Paragraph Vector	3.82%



Boulder

Effective Use of Word Order for Text Categorization with Convolutional Neural Networks

[YouTube Video](#)

CNN for text

$$V = \{ \text{"don't", "hate", "I", "it", "love"} \}$$

vocabulary

$D = \text{"I love it"}$

document

$$\mathbf{x} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \rightarrow \begin{array}{l} \text{I} \\ \text{love} \\ \text{it} \end{array}$$

Seq-CNN

$$\mathbf{r}_0(\mathbf{x}) = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \rightarrow \begin{array}{l} \text{I} \\ \text{love} \end{array}$$

$$\mathbf{r}_1(\mathbf{x}) = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \rightarrow \begin{array}{l} \text{love} \\ \text{it} \end{array}$$

$$\mathbf{w}^k = \begin{bmatrix} w_{11}^k & w_{12}^k & w_{13}^k & w_{14}^k & w_{15}^k \\ w_{21}^k & w_{22}^k & w_{23}^k & w_{24}^k & w_{25}^k \end{bmatrix}$$

$$h_0^k = \sigma(\mathbf{w}^k \cdot \mathbf{r}_0(\mathbf{x}) + b^k) = \sigma(w_{13}^k + w_{25}^k + b^k)$$

$$h_1^k = \sigma(\mathbf{w}^k \cdot \mathbf{r}_1(\mathbf{x}) + b^k) = \sigma(w_{15}^k + w_{24}^k + b^k)$$

$$\mathbf{h}_0 = [h_0^1, \dots, h_0^K]$$

$$\mathbf{h}_1 = [h_1^1, \dots, h_1^K]$$

BOW-CNN

$$\mathbf{r}_0(\mathbf{x}) = [0, 0, 1, 0, 1]$$

$$\mathbf{r}_1(\mathbf{x}) = [0, 0, 0, 1, 1]$$

$$\mathbf{w}^k = [w_1^k, w_2^k, w_3^k, w_4^k, w_5^k]$$

$$h_0^k = \sigma(\mathbf{w}^k \cdot \mathbf{r}_0(\mathbf{x}) + b^k) = \sigma(w_3^k + w_5^k + b^k)$$

$$h_1^k = \sigma(\mathbf{w}^k \cdot \mathbf{r}_1(\mathbf{x}) + b^k) = \sigma(w_4^k + w_5^k + b^k)$$

Pooling for text

fix the number of pooling units and dynamically determine the pooling region size

methods	IMDB
bow-CNN	8.66
seq-CNN	8.39

predictive text regions (training set)

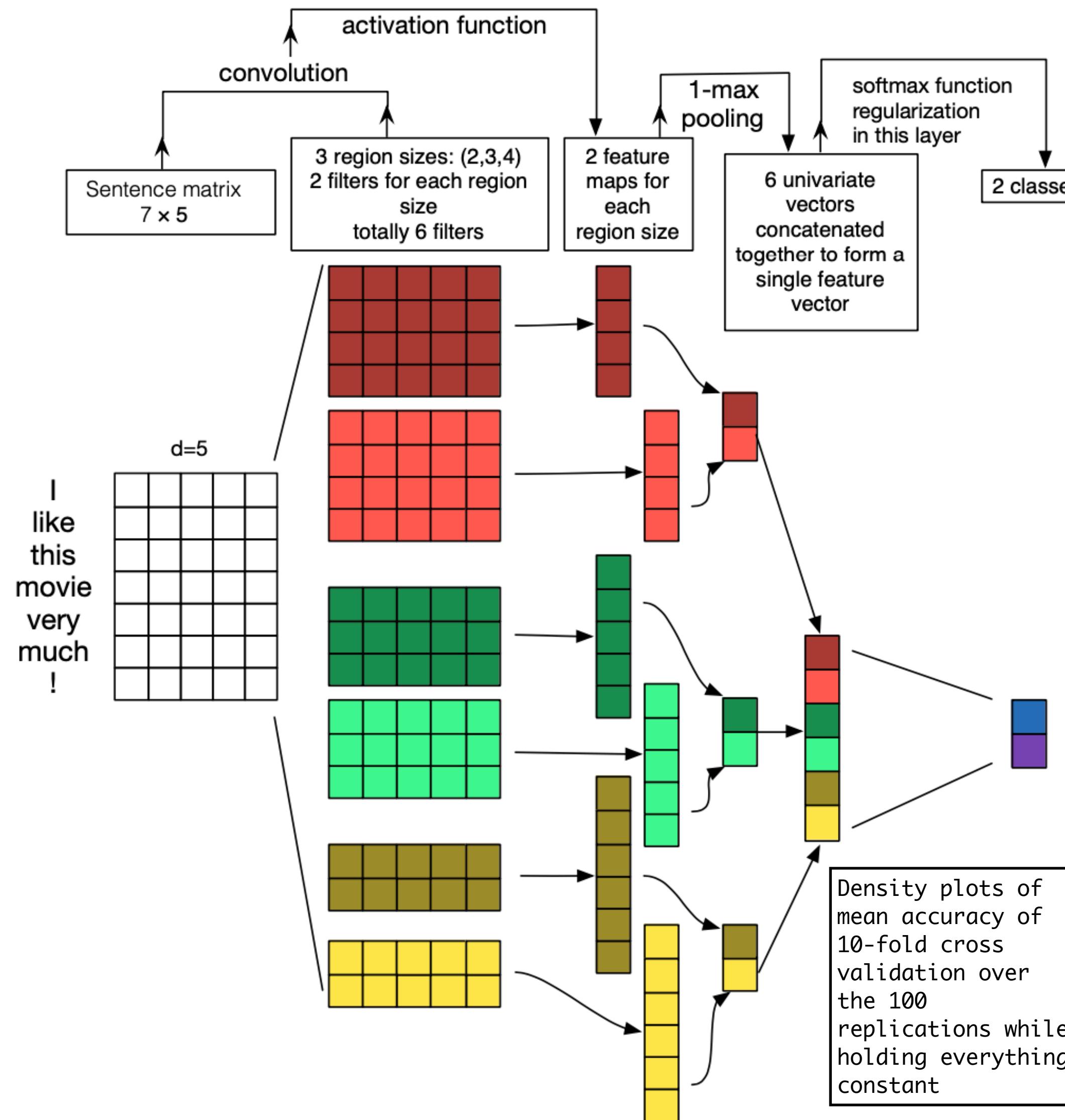
N1	completely useless .., return policy .
N2	it won't even, but doesn't work
N3	product is defective, very disappointing !
N4	is totally unacceptable, is so bad
N5	was very poor, it has failed
P1	works perfectly !, love this product
P2	very pleased !, super easy to, i am pleased
P3	'm so happy, it works perfect, is awesome !
P4	highly recommend it, highly recommended !
P5	am extremely satisfied, is super fast

predictive text regions (test set)

were unacceptably bad, is abysmally bad, were universally poor, was hugely disappointed, was enormously disappointed, is monumentally frustrating, are endlessly frustrating

best concept ever, best ideas ever, best hub ever, am wholly satisfied, am entirely satisfied, am incredibly satisfied, 'm overall impressed, am awfully pleased, am exceptionally pleased, 'm entirely happy, are acoustically good, is blindingly fast,

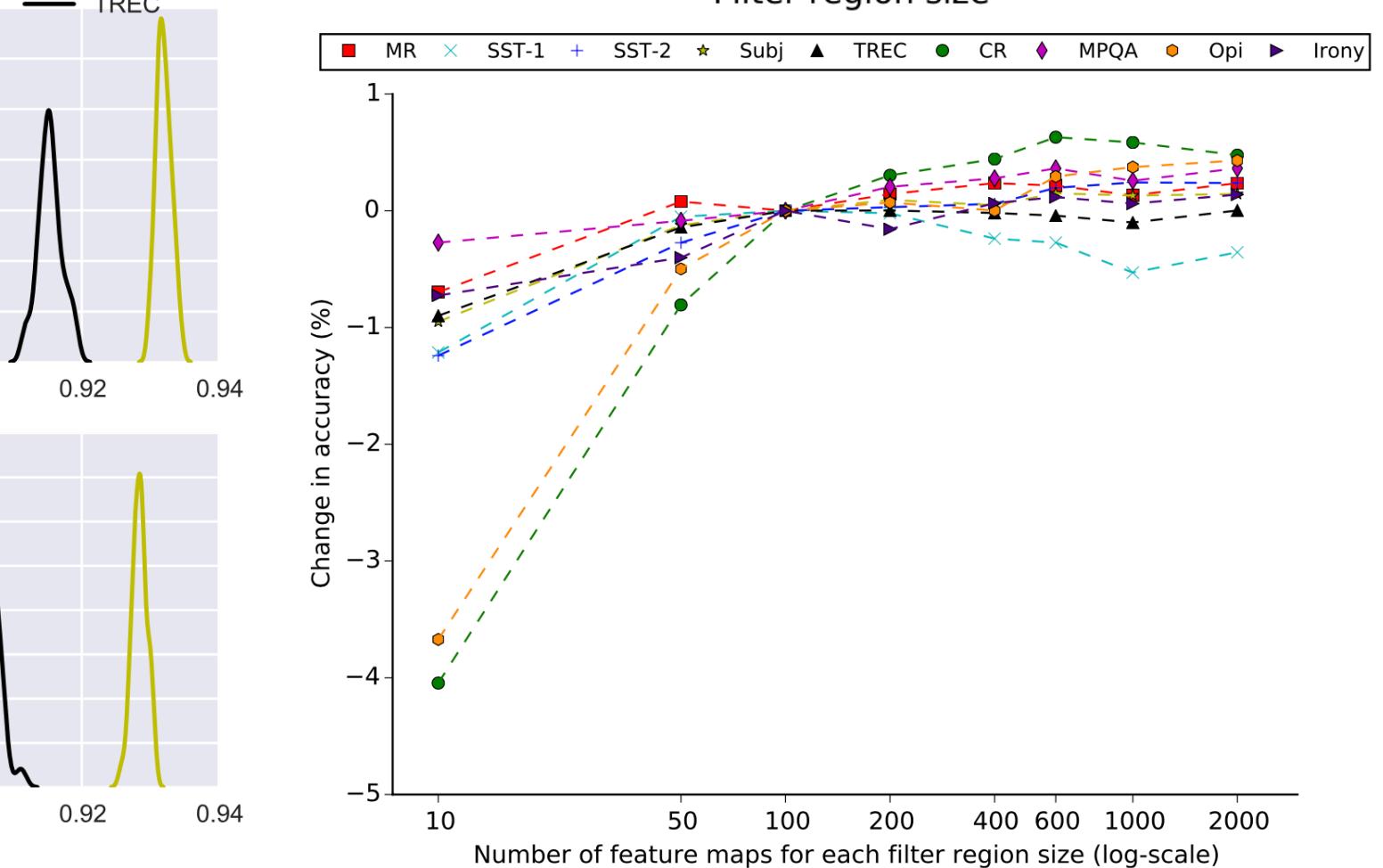
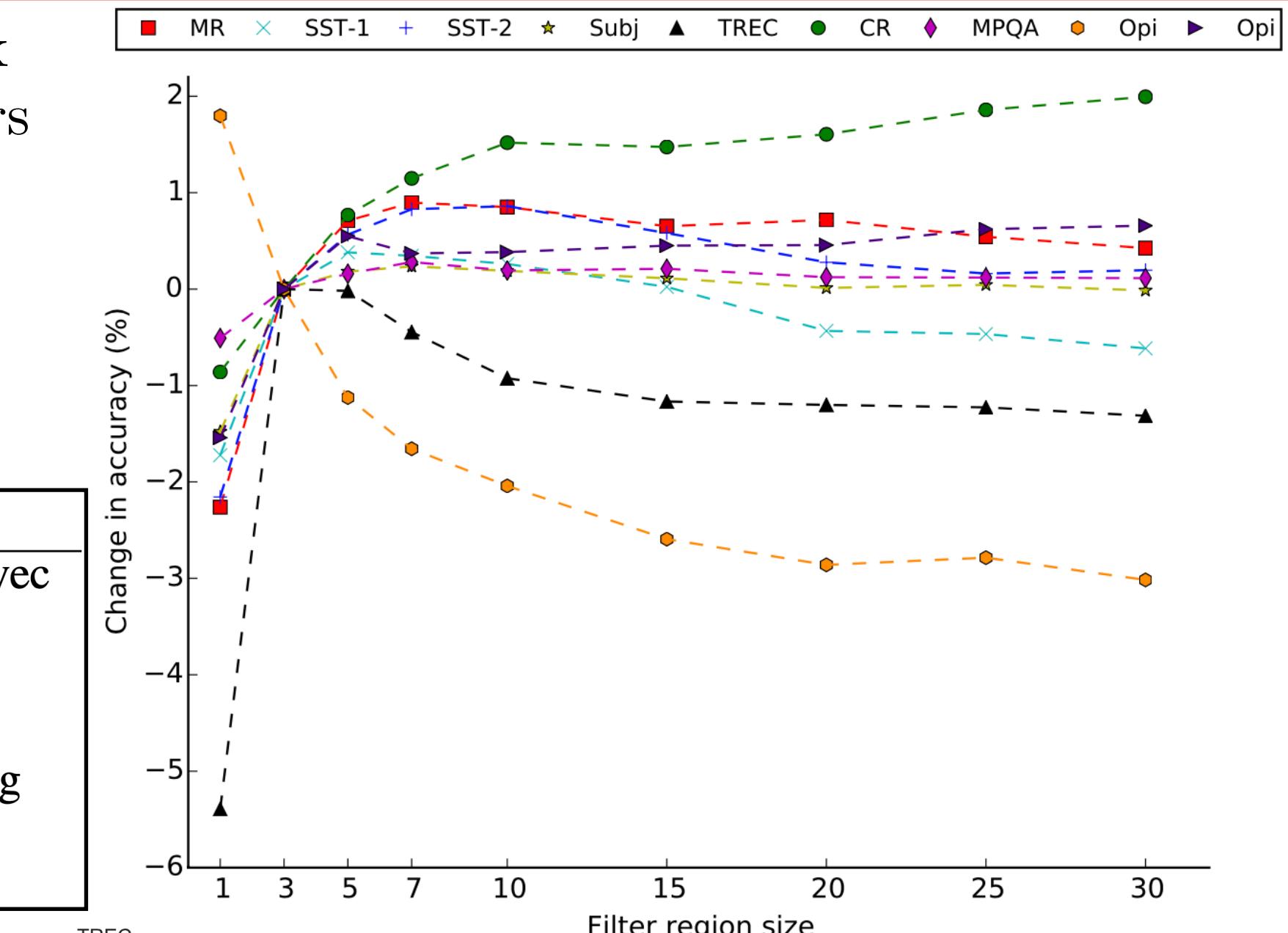
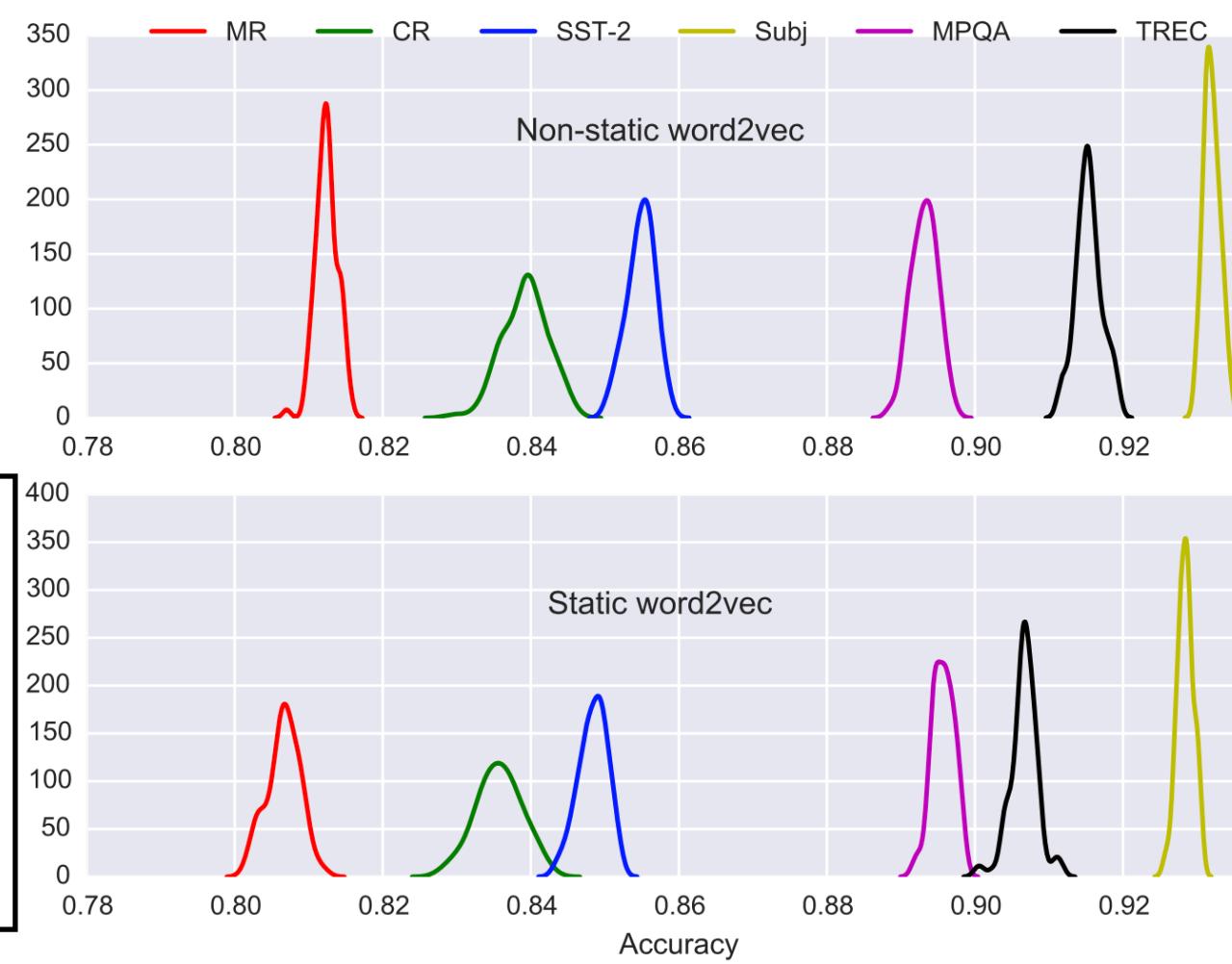
A Sensitivity Analysis Of (And Practitioners' Guide To) Convolutional Neural Networks For Sentence Classification



$A \in \mathbb{R}^{s \times d}$ \rightarrow sentence matrix
 $w \in \mathbb{R}^{h \times d}$ \rightarrow filter parameters
 $o_i = w \cdot A[i : i + h - 1, :]$
 $o \in \mathbb{R}^{s-h+1}$ \rightarrow output vector
 $c_i = f(o_i + b)$
 $c \in \mathbb{R}^{s-h+1}$ \rightarrow feature map

Baseline configuration

Description	Values
input word vectors	Google word2vec
filter region size	(3,4,5)
feature maps	100
activation function	ReLU
pooling	1-max pooling
dropout rate	0.5
l_2 norm constraint	3





Character-level Convolutional Networks for Text Classification

Boulder

Temporal Convolution Module

$$g : \{1, 2, \dots, l\} \rightarrow \mathbb{R}$$

↳ discrete input function

$$f : \{1, 2, \dots, k\} \rightarrow \mathbb{R}$$

↳ discrete kernel function

$$h : \{1, 2, \dots, \lfloor(l - k + 1)/d\rfloor\} \rightarrow \mathbb{R}$$

↳ convolution btw f & g with stride d

$$h(y) = \sum_{x=1}^k f(x) \cdot g(y \cdot d - x + c)$$

$c = k - d + 1$ is an offset constant.

$f_{ij}(x) \rightarrow$ many of such kernels/weights

$i = 1, \dots, m \rightarrow$ input feature size

$j = 1, \dots, n \rightarrow$ output feature size

$g_i(x) \rightarrow$ inputs

$h_j(x) \rightarrow$ outputs

$$h_j(y) = \sum_{i=1}^m \sum_{x=1}^k f_{ij}(x) \cdot g_i(y \cdot d - x + c)$$

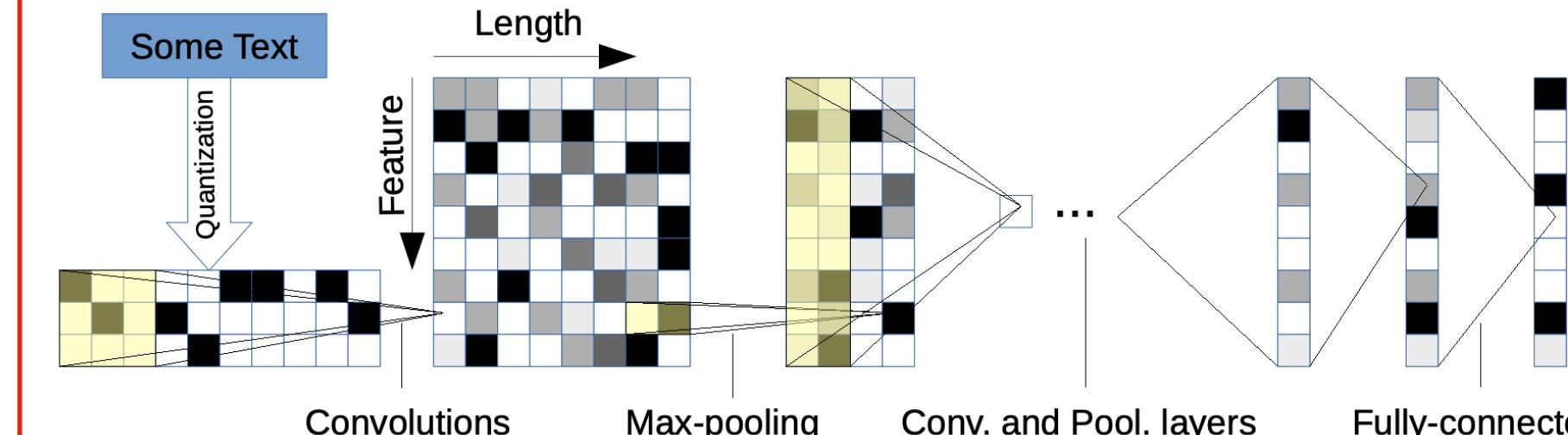
Temporal Max-pooling Module

$$h(y) = \sum_{x=1}^k f(x) \cdot g(y \cdot d - x + c)$$

Character Quantization

abcdefghijklmnopqrstuvwxyz 0123456789-, . ! ? : ' ' / \ | _ @ # \$ % ^ & * ~ ` + - = < > () [] { }

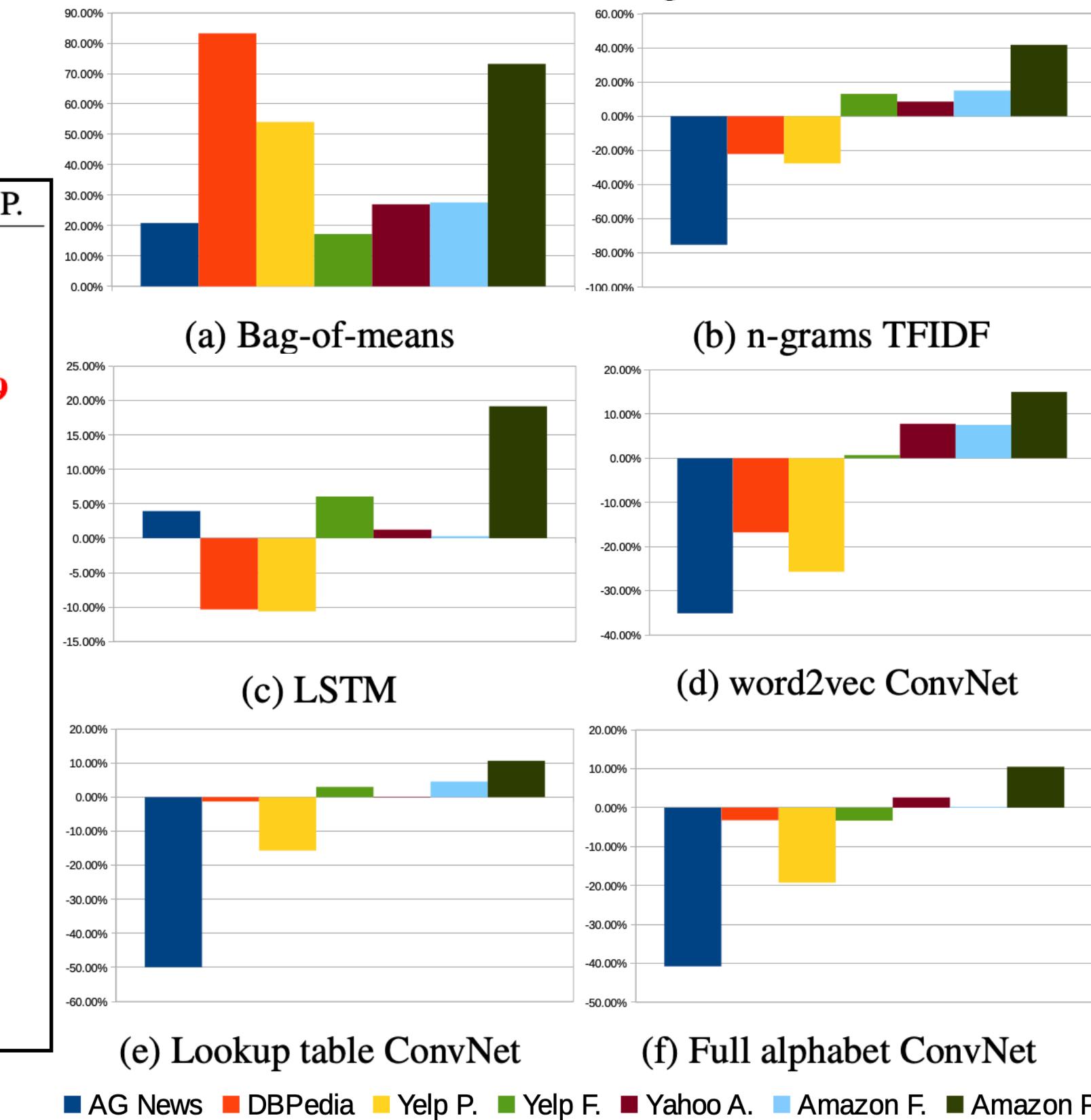
1-of-70 (i.e., one-hot) encoding
 $l_0 \rightarrow$ fixed-length sequences of 70 dimensional vectors
 all-zero vector \rightarrow any character not in the alphabet
 (including blank characters)



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

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

Relative errors with comparison models



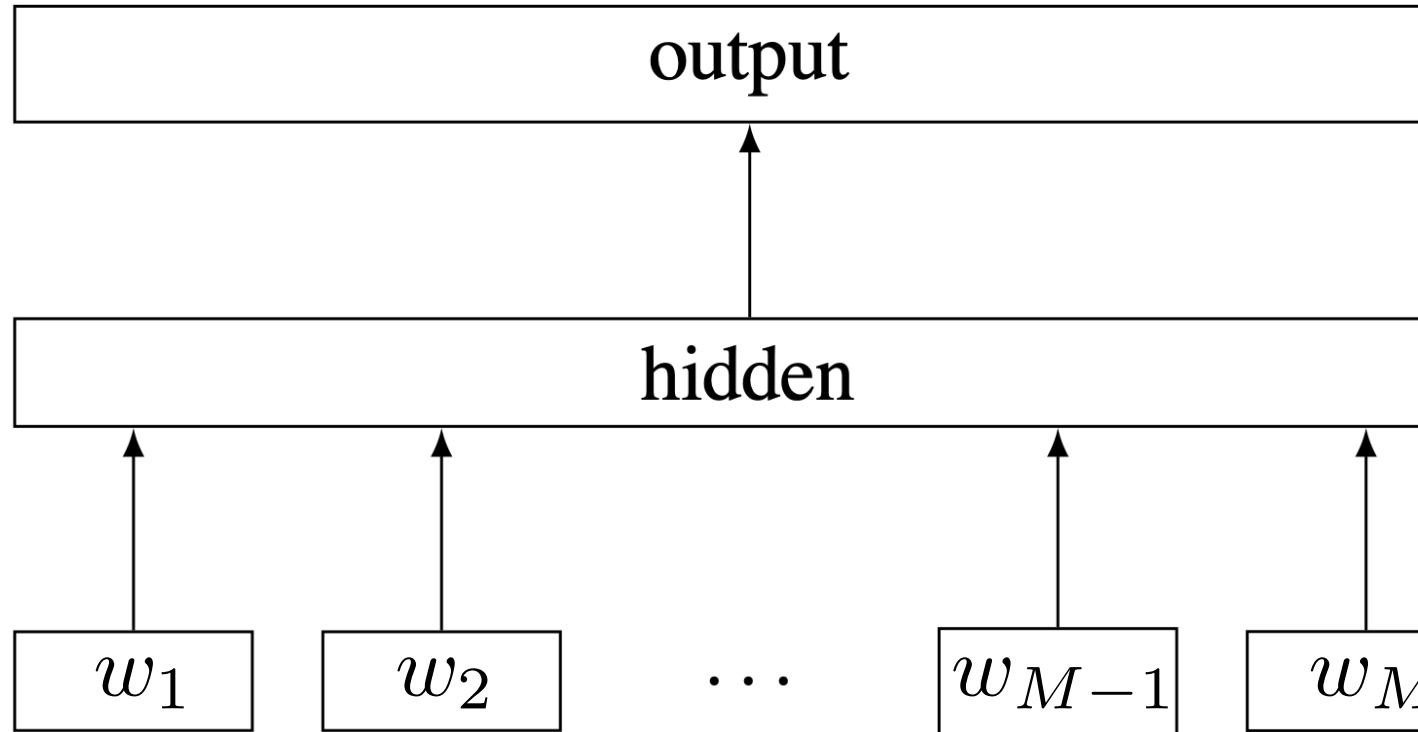


Boulder



[YouTube Video](#)

Bag Of Tricks For Efficient Text Classification



Model architecture of **fastText** for a sentence with M ngram features w_1, w_2, \dots, w_M . The features are embedded and averaged to form the hidden variable.

$$-\frac{1}{N} \sum_{n=1}^N y_n \log(f(BAx_n))$$

$N \rightarrow$ number of documents

$x_n \rightarrow$ normalized bag of features of document n

$y_n \rightarrow$ label

$A, B \rightarrow$ weight matrices

$A \rightarrow$ look-up table

$f \rightarrow$ softmax

Similar to the CBOW model!

$\mathcal{O}(kh) \rightarrow$ computational cost (softmax)

$k \rightarrow$ number of classes

$h \rightarrow$ dimension of text representation

$\mathcal{O}(h \log_2 k) \rightarrow$ hierarchical softmax

Test accuracy [%] on sentiment datasets.

Model	AG	Sogou	DBP	Yelp P.	Yelp F.	Yah. A.	Amz. F.	Amz. P.
BoW (Zhang et al., 2015)	88.8	92.9	96.6	92.2	58.0	68.9	54.6	90.4
ngrams (Zhang et al., 2015)	92.0	97.1	98.6	95.6	56.3	68.5	54.3	92.0
ngrams TFIDF (Zhang et al., 2015)	92.4	97.2	98.7	95.4	54.8	68.5	52.4	91.5
char-CNN (Zhang and LeCun, 2015)	87.2	95.1	98.3	94.7	62.0	71.2	59.5	94.5
char-CRNN (Xiao and Cho, 2016)	91.4	95.2	98.6	94.5	61.8	71.7	59.2	94.1
VDCNN (Conneau et al., 2016)	91.3	96.8	98.7	95.7	64.7	73.4	63.0	95.7
fastText, $h = 10$	91.5	93.9	98.1	93.8	60.4	72.0	55.8	91.2
fastText, $h = 10$, bigram	92.5	96.8	98.6	95.7	63.9	72.3	60.2	94.6

Training time for a single epoch on sentiment analysis datasets

	Zhang and LeCun (2015)		Conneau et al. (2016)			fastText
	small char-CNN	big char-CNN	depth=9	depth=17	depth=29	$h = 10$, bigram
AG	1h	3h	24m	37m	51m	1s
Sogou	-	-	25m	41m	56m	7s
DBpedia	2h	5h	27m	44m	1h	2s
Yelp P.	-	-	28m	43m	1h09	3s
Yelp F.	-	-	29m	45m	1h12	4s
Yah. A.	8h	1d	1h	1h33	2h	5s
Amz. F.	2d	5d	2h45	4h20	7h	9s
Amz. P.	2d	5d	2h45	4h25	7h	10s

Model	prec@1	Running time		Model	prec@1	Running time	
		Train	Test			Train	Test
Freq. baseline	2.2	-	-	fastText, $h = 50$	31.2	6m40	48s
Tagspace, $h = 50$	30.1	3h8	6h	fastText, $h = 50$, bigram	36.7	7m47	50s
Tagspace, $h = 200$	35.6	5h32	15h	fastText, $h = 200$	41.1	10m34	1m29
				fastText, $h = 200$, bigram	46.1	13m38	1m37

Tag prediction
on YFCC100M
(Large output
space & large
dataset)



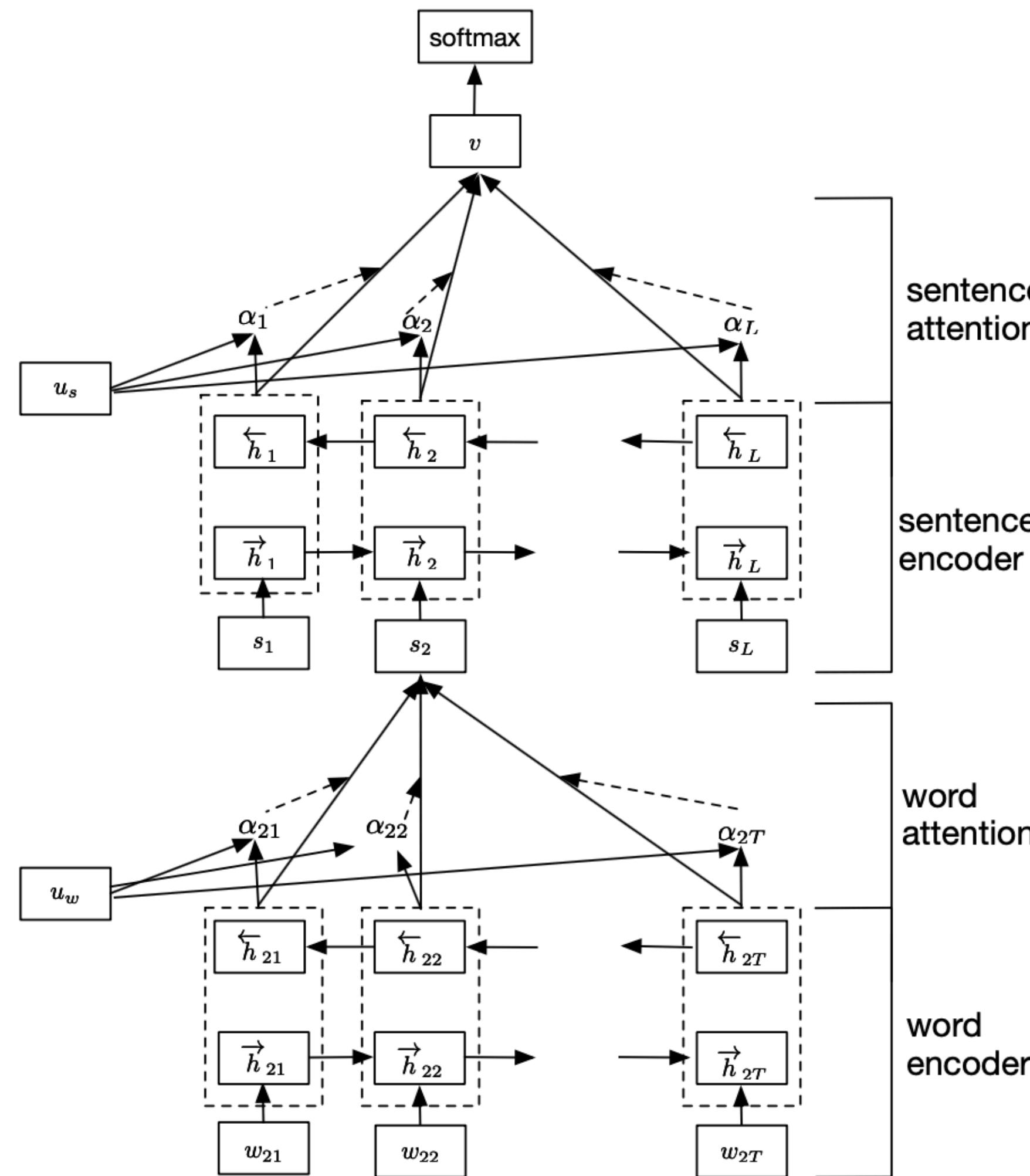
Boulder

Hierarchical Attention Networks for Document Classification



[YouTube Video](#)

words form sentences
sentences form a document



GRU-based sequence encoder

reset gate r_t

update gate z_t

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$$

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z)$$

$$\tilde{h}_t = \tanh(W_h x_t + r_t \odot (U_h h_{t-1}) + b_h)$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r)$$

Word Encoder

$$x_{it} = W_e w_{it}, t \in [1, T],$$

$$\overrightarrow{h}_{it} = \overrightarrow{\text{GRU}}(x_{it}), t \in [1, T],$$

$$\overleftarrow{h}_{it} = \overleftarrow{\text{GRU}}(x_{it}), t \in [T, 1].$$

$$h_{it} = [\overrightarrow{h}_{it}, \overleftarrow{h}_{it}]$$

Word Attention

$$u_{it} = \tanh(W_w h_{it} + b_w)$$

$$\alpha_{it} = \frac{\exp(u_{it}^\top u_w)}{\sum_t \exp(u_{it}^\top u_w)}$$

$$s_i = \sum_t \alpha_{it} h_{it}$$

$u_w \rightarrow$ word context

query "what is the informative word"

Sentence Encoder

$$\overrightarrow{h}_i = \overrightarrow{\text{GRU}}(s_i), i \in [1, L],$$

$$\overleftarrow{h}_i = \overleftarrow{\text{GRU}}(s_i), t \in [L, 1].$$

$$h_i = [\overrightarrow{h}_i, \overleftarrow{h}_i]$$

Sentence Attention

$$u_i = \tanh(W_s h_i + b_s),$$

$$\alpha_i = \frac{\exp(u_i^\top u_s)}{\sum_i \exp(u_i^\top u_s)},$$

$$v = \sum_i \alpha_i h_i,$$

Document Classification

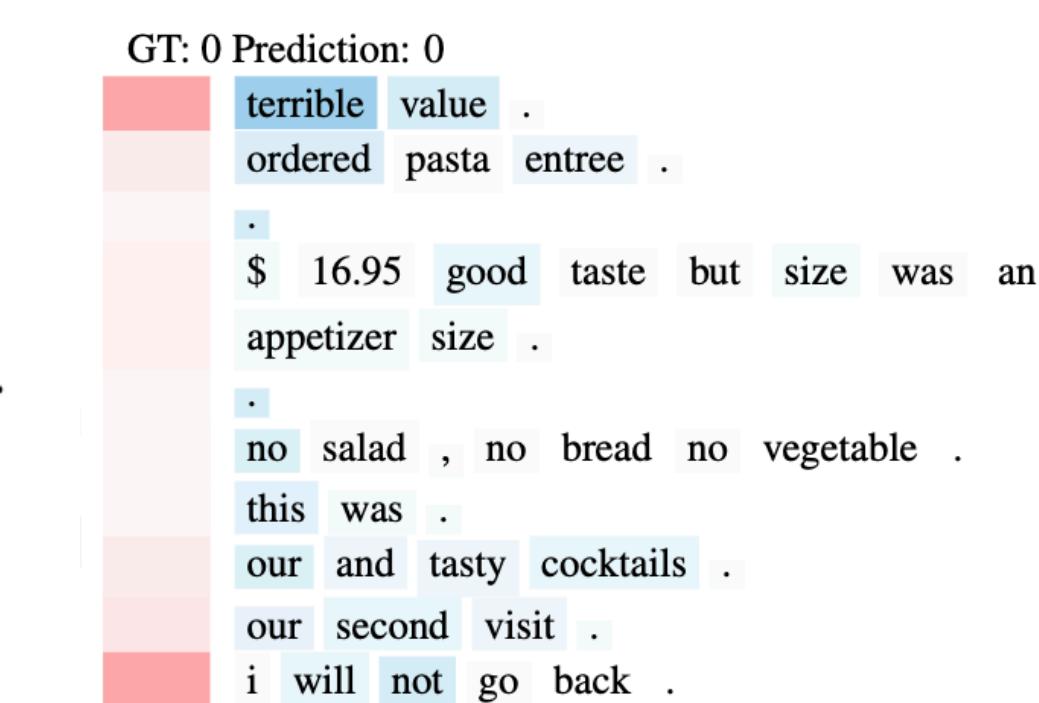
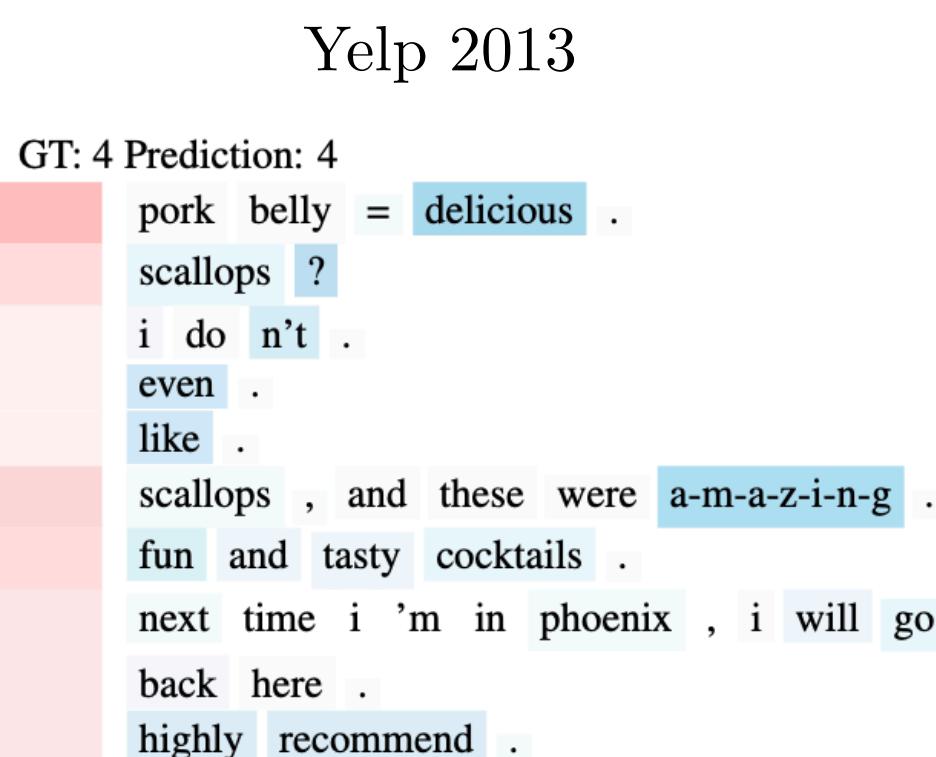
$$p = \text{softmax}(W_c v + b_c)$$

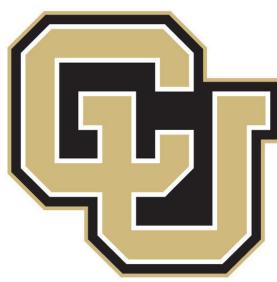
$$L = -\sum_d \log p_{dj}$$

j is the label of document d .

sentiment estimation
topic classification

Data set	classes	documents
Yelp 2013	5	335,018
Yelp 2014	5	1,125,457
Yelp 2015	5	1,569,264
IMDB review	10	348,415
Yahoo Answer	10	1,450,000
Amazon review	5	3,650,000



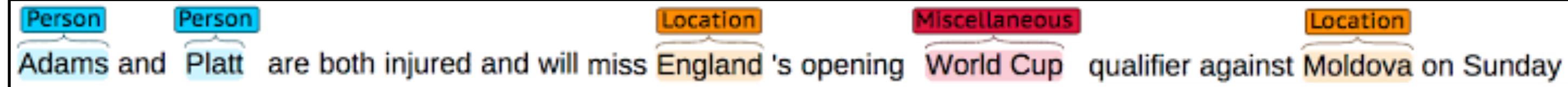


Boulder



[YouTube Video](#)

Neural Architectures For Named Entity Recognition



Mark	Watney	visited	Mars
B-PER	I-PER	O	B-LOC

B-PER → beginning of a person name

I-PER → inside a person name

B-LOC → beginning of a location name

I-LOC → inside a location name

B-ORG → beginning of an organization name

I-ORG → inside an organization name

B-MISC → beginning of a miscellaneous entity name

I-MISC → inside a miscellaneous entity name

O → outside of a named entity

LSTM-CRF Model

$(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$ → input (sequence of vectors)

$(\mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_n)$ → returned sequence from LSTM

$$\mathbf{i}_t = \sigma(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{W}_{ci}\mathbf{c}_{t-1} + \mathbf{b}_i)$$

$$\mathbf{c}_t = (1 - \mathbf{i}_t) \odot \mathbf{c}_{t-1} +$$

$$\mathbf{i}_t \odot \tanh(\mathbf{W}_{xc}\mathbf{x}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1} + \mathbf{b}_c)$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{W}_{co}\mathbf{c}_t + \mathbf{b}_o)$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t),$$

$$\mathbf{h}_t = [\overrightarrow{\mathbf{h}}_t; \overleftarrow{\mathbf{h}}_t] \rightarrow \text{bidirectional LSTM}$$

Conditional Random Field (CRF)

$$\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$$

$$\mathbf{y} = (y_1, y_2, \dots, y_n) \rightarrow \text{sequence of outputs}$$

$$P \in \mathbb{R}^{n \times k} \rightarrow \text{matrix of scores}$$

$$k \rightarrow \text{number of distinct tags}$$

$$P_{i,j} \rightarrow \text{score of the } j\text{-th tag of the } i\text{-th word in a sentence}$$

$$s(\mathbf{X}, \mathbf{y}) = \sum_{i=0}^n A_{y_i, y_{i+1}} + \sum_{i=1}^n P_{i, y_i} \rightarrow \text{score of } \mathbf{y}$$

$$A \in \mathbb{R}^{(k+2) \times (k+2)} \rightarrow \text{matrix of compatibility scores}$$

$$A_{i,j} \rightarrow \text{score of a transition from tag } i \text{ to } j$$

$$y_0, y_{n+1} \rightarrow \text{start and end tags of a sequence}$$

$$p(\mathbf{y}|\mathbf{X}) = \frac{e^{s(\mathbf{X}, \mathbf{y})}}{\sum_{\tilde{\mathbf{y}} \in \mathbf{Y_X}} e^{s(\mathbf{X}, \tilde{\mathbf{y}})}}$$

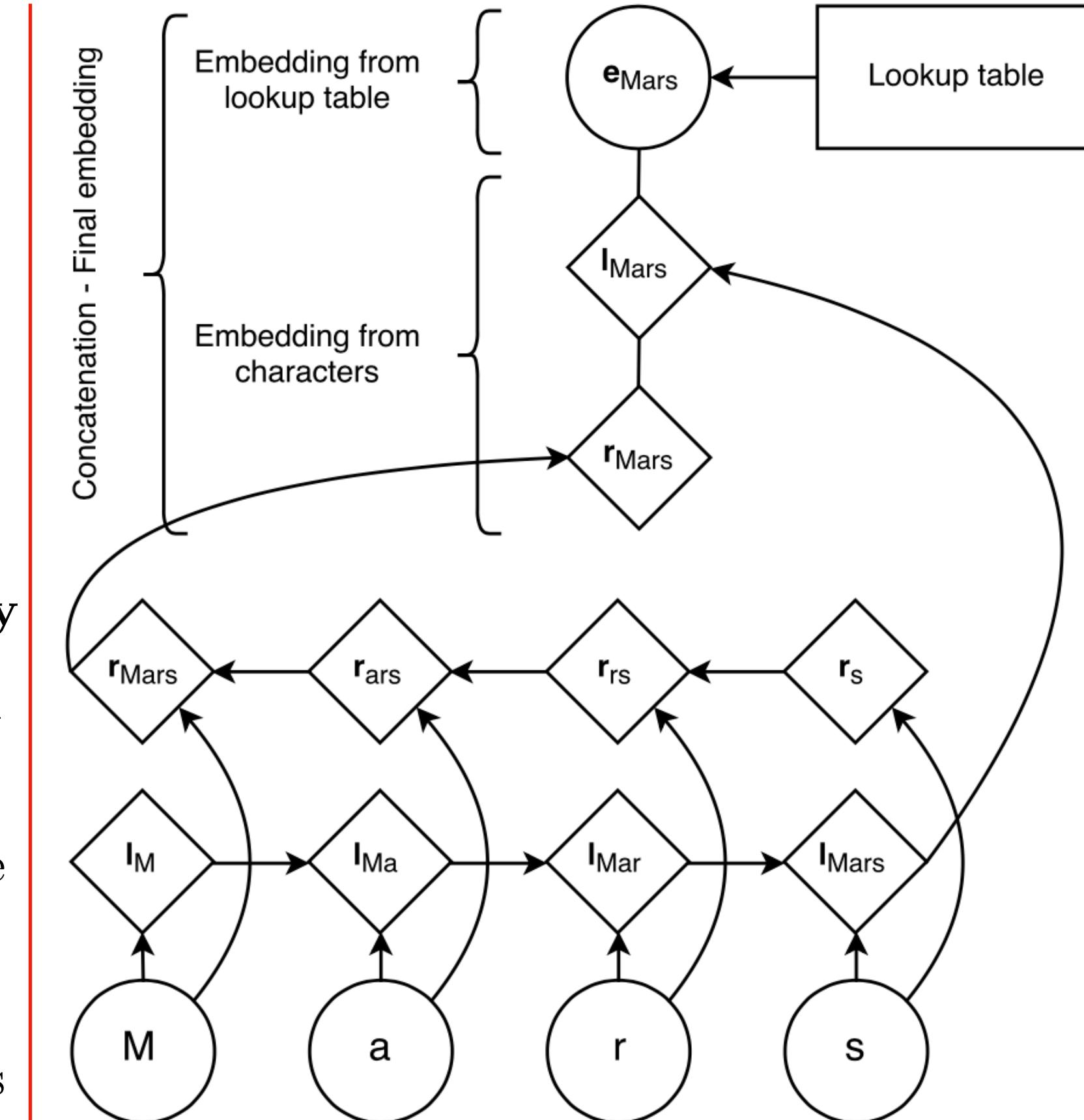
all possible tag sequences

$$\log(p(\mathbf{y}|\mathbf{X})) = s(\mathbf{X}, \mathbf{y}) - \log \left(\sum_{\tilde{\mathbf{y}} \in \mathbf{Y_X}} e^{s(\mathbf{X}, \tilde{\mathbf{y}})} \right)$$

dynamic programming

Character-based models of words

Use dropout training to encourage the model to depend on both character-level and word-level representations



English NER results (CoNLL-2003 test set)

Model

LSTM-CRF (no char)

LSTM-CRF

F₁

90.20

90.94

Spanish NER (CoNLL-2002 test set)

LSTM-CRF – no char

83.44

LSTM-CRF

85.75



Boulder

Universal Language Model Fine-tuning for Text Classification



[YouTube Video](#)

ULMFiT

Text Classification: 1. spam, fraud, and bot detection
2. emergency response
3. commercial document classification (legal discovery)

- discriminative fine-tuning
- slanted triangular learning rate
- gradual unfreezing

$\mathcal{T}_S \rightarrow$ source task

$\mathcal{T}_T \rightarrow$ target task with $\mathcal{T}_T \neq \mathcal{T}_S$

$\mathcal{H} \rightarrow$ hypothesis space

General-Domain LM pre-training

Wikitext-103

28,595 pre-processed Wikipedia articles

103 million words

Target task LM fine-tuning

- discriminative fine-tuning

$$\theta_t = \theta_{t-1} - \eta \cdot \nabla_{\theta} J(\theta)$$

$$\theta = \{\theta^1, \dots, \theta^L\}$$

$\theta^\ell \rightarrow$ parameters of the model at the ℓ -th layer

$\eta^\ell \rightarrow$ learning rate of the ℓ -th layer

$$\theta_t^\ell = \theta_{t-1}^\ell - \eta^\ell \cdot \nabla_{\theta^\ell} J(\theta)$$

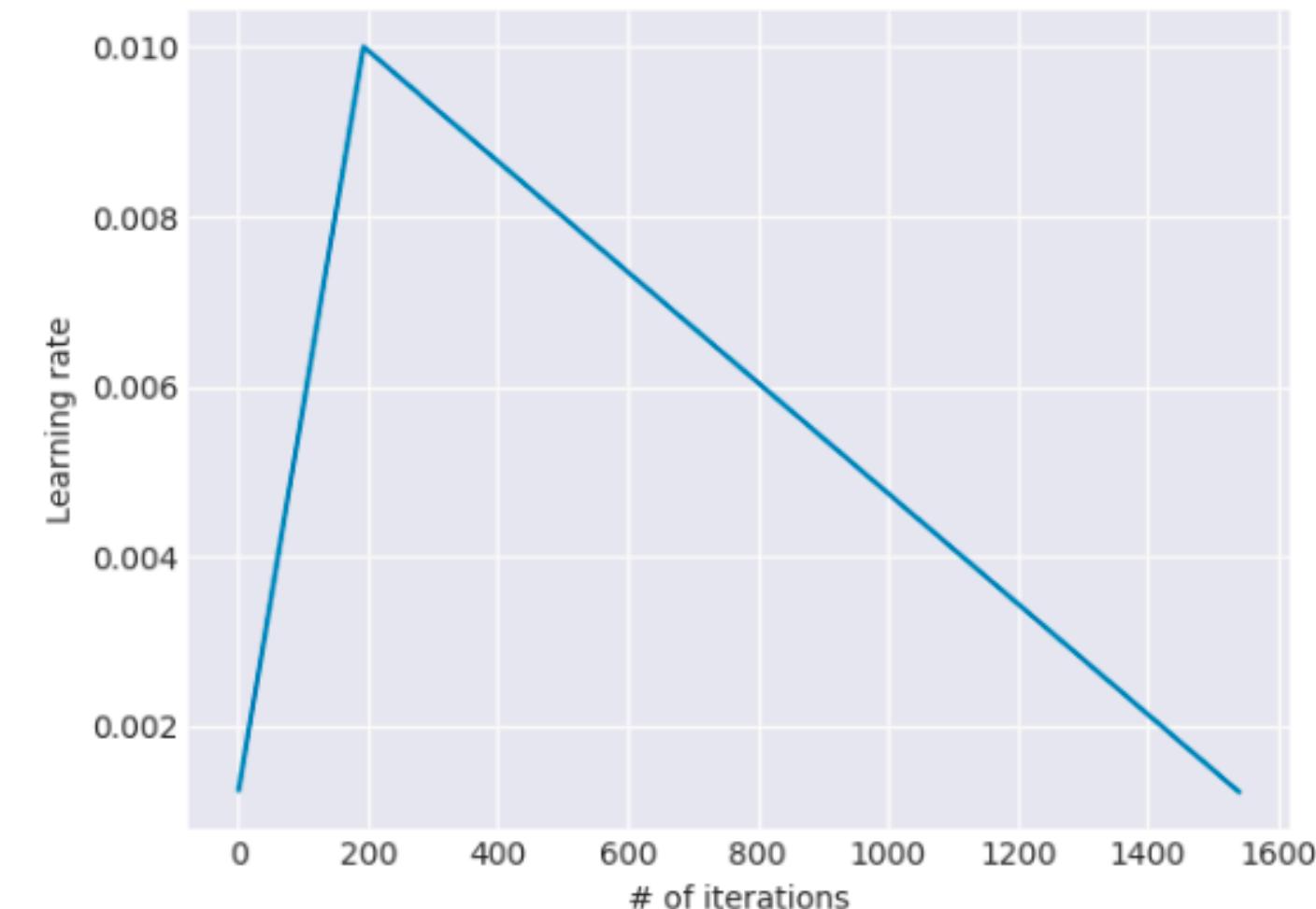
$$\eta^{\ell-1} = \eta^\ell / 2.6$$

– slanted triangular learning rate

$$cut = \lfloor T \cdot cut_frac \rfloor$$

$$p = \begin{cases} t/cut, & \text{if } t < cut \\ 1 - \frac{t-cut}{cut \cdot (1/cut_frac-1)}, & \text{otherwise} \end{cases}$$

$$\eta_t = \eta_{max} \cdot \frac{1 + p \cdot (ratio - 1)}{ratio}$$



$T \rightarrow$ number of training iterations

$$cut_frac = 0.1$$

$ratio = 32 \rightarrow$ how smaller the lowest LR is from the max LR

$$\eta_{max} = 0.01$$

Target task classifier fine-tuning

Two additional linear blocks

ReLU & softmax

$$H = \{h_1, h_2, \dots, h_T\}$$

$$h_c = [h_T, \text{maxpool}(H), \text{meanpool}(H)]$$

– gradual unfreezing

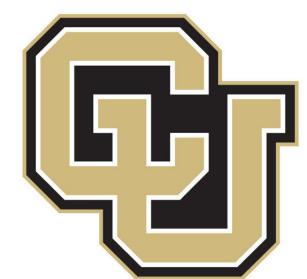
back-propagation through time (BPTT)

BPTT for Text Classification (BPT3C)

Dataset	Type	# classes	# examples
TREC-6	Question	6	5.5k
IMDb	Sentiment	2	25k
Yelp-bi	Sentiment	2	560k
Yelp-full	Sentiment	5	650k
AG	Topic	4	120k
DBpedia	Topic	14	560k

Model	Test	Model	Test
CoVe (McCann et al., 2017)	8.2	CoVe (McCann et al., 2017)	4.2
oh-LSTM (Johnson and Zhang, 2016)	5.9	TBCNN (Mou et al., 2015)	4.0
Virtual (Miyato et al., 2016)	5.9	LSTM-CNN (Zhou et al., 2016)	3.9
ULMFiT (ours)	4.6	ULMFiT (ours)	3.6

	AG	DBpedia	Yelp-bi	Yelp-full
Char-level CNN (Zhang et al., 2015)	9.51	1.55	4.88	37.95
CNN (Johnson and Zhang, 2016)	6.57	0.84	2.90	32.39
DPCNN (Johnson and Zhang, 2017)	6.87	0.88	2.64	30.58
ULMFiT (ours)	5.01	0.80	2.16	29.98



Boulder



Questions?

[YouTube Playlist](#)
