

# Convolutional Networks 1

CS 287

## Review: NGram Issues

In training we might see,

the arizona corporations commission **authorized**

But at test we see,

the colorado businesses organization ---

- ▶ Does this training example help here?
  - ▶ Not really. No count overlap.
- ▶ Does backoff help here?
  - ▶ Maybe, if we have seen organization.
  - ▶ Mostly get nothing from the earlier words.

## Review: NGram Issues

In training we might see,

the arizona corporations commission **authorized**

But at test we see,

the colorado businesses organization ---

- ▶ Does this training example help here?
  - ▶ Not really. No count overlap.
- ▶ Does backoff help here?
  - ▶ Maybe, if we have seen organization.
  - ▶ Mostly get nothing from the earlier words.

## Review: NGram Issues

In training we might see,

the arizona corporations commission **authorized**

But at test we see,

the colorado businesses organization ---

- ▶ Does this training example help here?
  - ▶ Not really. No count overlap.
- ▶ Does backoff help here?
  - ▶ Maybe, if we have seen organization.
  - ▶ Mostly get nothing from the earlier words.

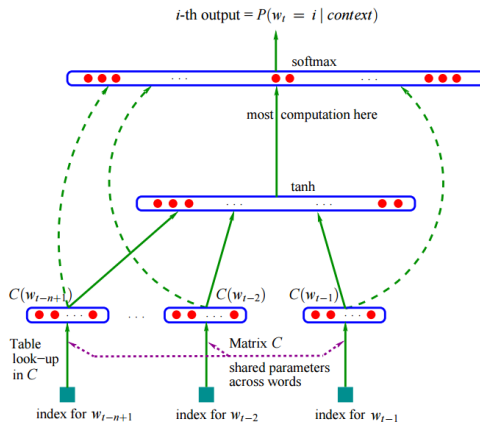
# Review: A Neural Probabilistic Language Model

Optional, direct connection layers,

$$NN_{DMLP1}(\mathbf{x}) = [\tanh(\mathbf{x}\mathbf{W}^1 + \mathbf{b}^1), \mathbf{x}] \mathbf{W}^2 + \mathbf{b}^2$$

- ▶  $\mathbf{W}^1 \in \mathbb{R}^{d_{\text{in}} \times d_{\text{hid}}}$ ,  $\mathbf{b}^1 \in \mathbb{R}^{1 \times d_{\text{hid}}}$ ; first affine transformation
- ▶  $\mathbf{W}^2 \in \mathbb{R}^{(d_{\text{hid}} + d_{\text{in}}) \times d_{\text{out}}}$ ,  $\mathbf{b}^2 \in \mathbb{R}^{1 \times d_{\text{out}}}$ ; second affine transformation

# Review: A Neural Probabilistic Language Model (Bengio, 2003)



Dashed-lines show the optional direct connections,  $C = v$ .

## Review: Comparison

Both count-based models and feed-forward NNLMs are Markovian language models,

Comparison:

- ▶ Training Speed: ngrams are much faster (more coming)
- ▶ Usage Speed: ngrams very fast, NN can be fast with some tricks.
- ▶ Memory: NN models can be much smaller (but there are big ones)
- ▶ Accuracy: Comparable for small data, NN does better with more.

Advantages of NN model

- ▶ Can be trained end-to-end.
- ▶ Does not require smoothing methods.

# Quiz

Neural language models can be poor at assigning very high probability to high confidence decisions, for instance `major league baseball` or `united states of america`.

- ▶ Give a high-level explanation of why this might occur compared to an n-gram model.
- ▶ Describe a variant of the Bengio model that is able to incorporate extra parameters to allow for rare cases that should have high probability.



# Contents

Text Classification Review

Convolutions

Applications

Vision

# Sentiment

## Good Sentences

- ▶ A thoughtful, provocative, insistently humanizing film.
- ▶ Occasionally melodramatic, it's also extremely effective.
- ▶ Guaranteed to move anyone who ever shook, rattled, or rolled.

## Bad Sentences

- ▶ A sentimental mess that never rings true.
- ▶ This 100-minute movie only has about 25 minutes of decent material.
- ▶ Here, common sense flies out the window, along with the hail of bullets, none of which ever seem to hit Sascha.

# Review Linear Models for Classification

Linear model,

$$\hat{\mathbf{y}} = f(\mathbf{x}\mathbf{W} + \mathbf{b})$$

- ▶  $\mathbf{W} \in \mathbb{R}^{d_{\text{in}} \times d_{\text{out}}}$ ,  $\mathbf{b} \in \mathbb{R}^{1 \times d_{\text{out}}}$ ; model parameters
- ▶  $f : \mathbb{R}^{d_{\text{out}}} \mapsto \mathbb{R}^{d_{\text{out}}}$ ; activation function
- ▶ Sometimes  $\mathbf{z} = \mathbf{x}\mathbf{W} + \mathbf{b}$  informally “score” vector.
- ▶ Note  $\mathbf{z}$  and  $\hat{\mathbf{y}}$  are not one-hot.

Class prediction,

$$\hat{c} = \arg \max_{i \in \mathcal{C}} \hat{y}_i = \arg \max_{i \in \mathcal{C}} (\mathbf{x}\mathbf{W} + \mathbf{b})_i$$

# Features 1: Sparse Bag-of-Words Features

Representation is counts of input words,

- ▶  $\mathcal{F}$ ; the vocabulary of the language.
- ▶  $\mathbf{x} = \sum_i \delta(f_i)$

Example: Movie review input,

A sentimental mess

$$\begin{aligned}\mathbf{x} &= \delta(\text{word:A}) + \delta(\text{word:sentimental}) \\ &+ \delta(\text{word:mess})\end{aligned}$$

$$\mathbf{x}^\top = \begin{bmatrix} 1 \\ \vdots \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ \vdots \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ \vdots \\ 1 \\ 1 \end{bmatrix} \begin{array}{l} \text{word:A} \\ \vdots \\ \text{word:mess} \\ \text{word:sentimental} \end{array}$$

## Features 2: Sparse Bag-of-Bigrams Features

Representation is counts of input bigrams,

- ▶  $\mathcal{F}$ ; the vocabulary of the bigram language.
- ▶  $\mathbf{x} = \sum_i \delta(f_i)$

Example: Movie review input,

A sentimental mess

$$\begin{aligned}\mathbf{x} &= \delta(\text{word:A}) + \delta(\text{bigram:A:sentimental}) \\ &+ \delta(\text{word:sentimental}) + \delta(\text{bigram:sentimental:mess}) \\ &+ \delta(\text{word:mess})\end{aligned}$$

## Features 3: Continuous Bag-of-Words Features

$$\mathbf{x} = \sum_{i=1}^k v(f_i; \theta) = \sum_{i=1}^k \delta(f_i) \mathbf{w}^0$$

- ▶  $\mathcal{F}$ ; the vocabulary of the language.
- ▶  $\mathbf{x} = \sum_i \delta(f_i)$

Example: Movie review input,

$$\mathbf{x} = v(\text{word:A}) + v(\text{word:sentimental}) + v(\text{word:mess})$$

$$\mathbf{x}^\top = \begin{bmatrix} 0.2 \\ \vdots \\ 1.2 \\ -0.5 \end{bmatrix} + \begin{bmatrix} 0.8 \\ \vdots \\ 1.0 \\ -1.0 \end{bmatrix} + \begin{bmatrix} 0.1 \\ \vdots \\ 9.2 \\ -2.0 \end{bmatrix} = \begin{bmatrix} 1.1 \\ \vdots \\ 11.4 \\ -3.5 \end{bmatrix}$$

## Features 4: Continuous Bag-of-Bigrams Features?

Representation is counts of input bigrams,

- ▶  $\mathcal{F}$ ; the vocabulary of the bigram language.
- ▶  $\mathbf{x} = \sum_i \delta(f_i)$

Example: Movie review input,

A sentimental mess

$$\begin{aligned}\mathbf{x} &= v(\text{word:A}) + v_2(\text{bigram:A:sentimental}) \\ &+ v(\text{word:sentimental}) + v_2(\text{bigram:sentimental:mess}) \\ &+ v(\text{word:mess})\end{aligned}$$

# Neural Network

One-layer multi-layer perceptron architecture,

$$NN_{MLP1}(\mathbf{x}) = g(\mathbf{x}\mathbf{W}^1 + \mathbf{b}^1) \mathbf{W}^2 + \mathbf{b}^2$$

- ▶  $\mathbf{x}\mathbf{W} + \mathbf{b}$ ; *perceptron*
- ▶  $\mathbf{x}$  is the dense representation in  $\mathbb{R}^{1 \times d_{\text{in}}}$
- ▶  $\mathbf{W}^1 \in \mathbb{R}^{d_{\text{in}} \times d_{\text{hid}}}$ ,  $\mathbf{b}^1 \in \mathbb{R}^{1 \times d_{\text{hid}}}$ ; first affine transformation
- ▶  $\mathbf{W}^2 \in \mathbb{R}^{d_{\text{hid}} \times d_{\text{out}}}$ ,  $\mathbf{b}^2 \in \mathbb{R}^{1 \times d_{\text{out}}}$ ; second affine transformation
- ▶  $g : \mathbb{R}^{d_{\text{hid}} \times d_{\text{hid}}}$  is an *activation non-linearity* (often pointwise)
- ▶  $g(\mathbf{x}\mathbf{W}^1 + \mathbf{b}^1)$  is the *hidden layer*



# Contents

Text Classification Review

Convolutions

Applications

Vision

# Windowed Classification

Alternative method, windows into MLP.

**Goal:** predict  $t_5$ .

- ▶ Windowed word model.

$$w_1 \ w_2 \ [w_3 \ w_4 \ w_5 \ w_6 \ w_7] \ w_8$$

- ▶  $w_3, w_4$ ; left context
- ▶  $w_5$ ; Word of interest
- ▶  $w_6, w_7$ ; right context
- ▶  $d_{\text{win}}$ ; size of window ( $d_{\text{win}} = 5$ )

# All Window for Classification

**Idea:** Use window at each location.

$$\begin{bmatrix} w_1 & w_2 & w_3 & w_4 & w_5 \end{bmatrix} w_6 \ w_7 \ w_8$$

$$w_1 \begin{bmatrix} w_2 & w_3 & w_4 & w_5 & w_6 \end{bmatrix} w_7 \ w_8$$

$$w_1 \ w_2 \begin{bmatrix} w_3 & w_4 & w_5 & w_6 & w_7 \end{bmatrix} w_8$$

$\vdots$

Each maps from window of embeddings to  $d_{\text{hid}}$

## Convolution Formally

Let our input be the embeddings of the full sentence,  $\mathbf{X} \in \mathbb{R}^{n \times d^0}$

$$\mathbf{X} = [v(w_1), v(w_2), v(w_3), \dots, v(w_n)]$$

Define a window model as  $NN_{window} : \mathbb{R}^{1 \times (d_{win} d^0)} \mapsto \mathbb{R}^{1 \times d_{hid}}$ ,

$$NN_{window}(\mathbf{x}_{win}) = \mathbf{x}_{win} \mathbf{W}^1 + \mathbf{b}^1$$

The convolution is defined as  $NN_{conv} : \mathbb{R}^{n \times d^0} \mapsto \mathbb{R}^{(n-d_{win}+1) \times d_{hid}}$ ,

$$NN_{conv}(\mathbf{X}) = \tanh \begin{bmatrix} NN_{window}(\mathbf{X}_{1:d_{win}}) \\ NN_{window}(\mathbf{X}_{2:d_{win}+1}) \\ \vdots \\ NN_{window}(\mathbf{X}_{n-d_{win}+1:n}) \end{bmatrix}$$

# Pooling

- ▶ Unfortunately  $NN_{conv} : \mathbb{R}^{n \times d^0} \mapsto \mathbb{R}^{(n-d_{win}+1) \times d_{hid}}$ .
- ▶ Need to map down to  $d_{out}$  for different  $n$
- ▶ Recall pooling operations.
- ▶ Pooling “over-time” operations  $f : \mathbb{R}^{n \times m} \mapsto \mathbb{R}^{1 \times m}$ 
  1.  $f_{max}(\mathbf{X})_{1,j} = \max_i X_{i,j}$
  2.  $f_{min}(\mathbf{X})_{1,j} = \min_i X_{i,j}$
  3.  $f_{mean}(\mathbf{X})_{1,j} = \sum_i X_{i,j} / n$

$$f(\mathbf{X}) = \begin{bmatrix} \Downarrow & \Downarrow & \dots \\ \Downarrow & \Downarrow & \dots \\ & \vdots & \\ \Downarrow & \Downarrow & \dots \end{bmatrix} = [ \dots ]$$

## Putting it together

$$\hat{y} = \text{softmax}(f_{\max}(NN_{\text{conv}}(\mathbf{X}))\mathbf{W}^2 + \mathbf{b}^2)$$

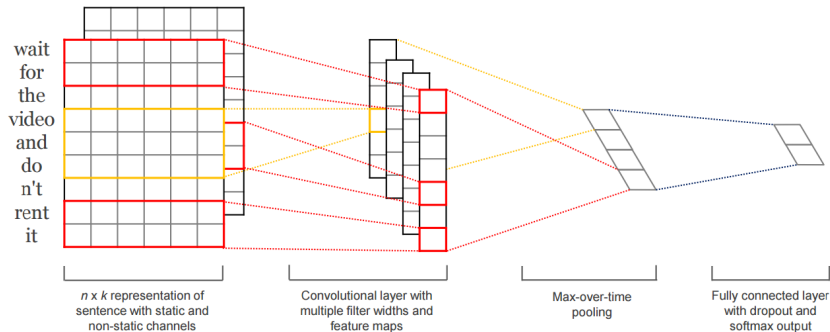
- ▶  $\mathbf{W}^2 \in \mathbb{R}^{d_{\text{hid}} \times d_{\text{out}}}$ ,  $\mathbf{b}^2 \in \mathbb{R}^{1 \times d_{\text{out}}}$
- ▶ Final linear layer  $\mathbf{W}^2$  uses learned window features

# Multiple Convolutions

$$\hat{y} = \text{softmax}([f(NN_{conv}^1(\mathbf{X})), f(NN_{conv}^2(\mathbf{X})), \dots, f(NN_{conv}^f(\mathbf{X}))]\mathbf{W}^2 + \mathbf{b}^2)$$

- ▶ Concat several convolutions together.
- ▶ Each  $NN^1$ ,  $NN^2$ , etc uses a different  $d_{\text{win}}$
- ▶ Allows for different window-sizes (similar to multiple n-grams)

# Convolution Diagram (Kim, 2014)



►  $n = 9$ ,  $d_{\text{hid}} = 4$ ,  $d_{\text{out}} = 2$

► red-  $d_{\text{win}} = 2$ , blue-  $d_{\text{win}} = 3$ , (ignore back channel)



# Classification Results

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	<b>89.6</b>
CNN-non-static	<b>81.5</b>	48.0	87.2	93.4	93.6	84.3	89.5
CNN-multichannel	81.1	47.4	<b>88.1</b>	93.2	92.2	<b>85.0</b>	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)	—	<b>48.7</b>	87.8	—	—	—	—

# Convolutional Vocabulary

- ▶ **kernel size** or **filter width** ; window size  $d_{\text{win}}$
- ▶ **filter**; column of matrix  $\mathbf{W}^1$  in  $\mathbb{R}^{(d^0 \times d_{\text{win}}) \times 1}$
- ▶ **feature map**; column of  $NN_{\text{conv}}$ ,  $d_{\text{hid}}$  of these
- ▶ **fully-connected layer**; affine or linear + activation
- ▶ **random, static, non-static**; embedding layer setup
- ▶ **temporal convolution, time-delay convolution**; names for one-dimensional convolutions

# Why is it called a convolution?

- ▶ Let  $\mathbf{x}$  and  $\mathbf{y}$  be in  $\mathbb{R}^n$  and  $\mathbb{R}^m$

$$[\mathbf{x} * \mathbf{y}]_i = \sum_{j=1}^m x_{i-j} y_j$$

- ▶ Circular,  $i - k$  wraps around.
- ▶ For NN, include padding

# Contents

Text Classification Review

Convolutions

Applications

Vision

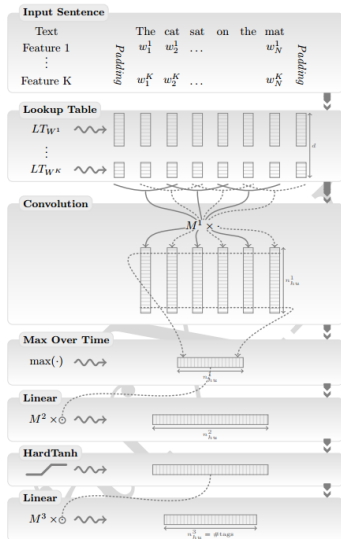
## Language Applications: Semantic Role Labeling

He would n't accept anything of value from those he was writing about

[A0 He ] [AM-MOD would ] [AM-NEG n't ] [V accept ] [A1 anything of value ] from [A2 those he was writing about ]

- ▶ V: verb
- ▶ A0: acceptor
- ▶ A1: thing accepted
- ▶ A2: accepted-from
- ▶ A3:attribute
- ▶ AM-MOD: modal
- ▶ AM-NEG: negation

# Other Language Applications (Collobert et al. 2011)



# C&W SRL

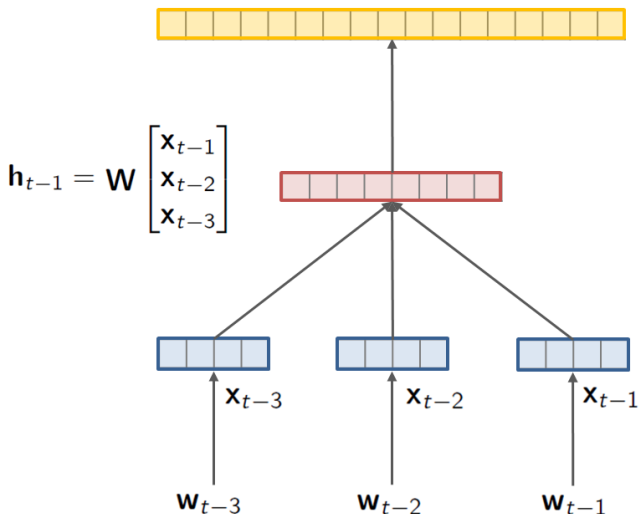
- ▶ First given a verb  $w_i$  e.g. accept.
- ▶ Then consider a word  $w_j$  e.g. n't
- ▶ For a word  $w_k$  features are

$$v(w_k), v_2(\text{cap}(w_k)), v_3(i - k), v_4(j - k)$$

- ▶ Convolution over sentence is used to predict role.
- ▶  $O(n \times |\text{verbs}|)$  convolutions per sentence

# Feed-forward NLM (Bengio, Ducharme, and Vincent 2003)

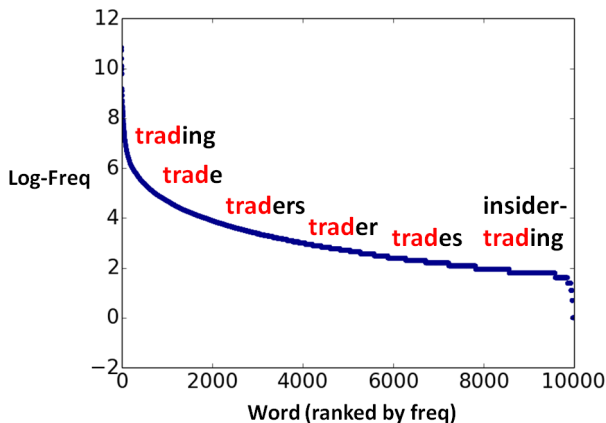
$$p(w_t | w_1, \dots, w_{t-1}) = \text{softmax}(\mathbf{P}\mathbf{h}_{t-1} + \mathbf{q})$$





# NLM Issue

**Issue:** The fundamental unit of information is still the **word**



Separate embeddings for “trading”, “trade”, “trades”, etc.

# Character-level CNN (CharCNN)

a b s u r d i t y

# Character-level CNN (CharCNN)

$\mathbf{C} \in \mathbb{R}^{d \times l}$  : Representation of *absurdity*

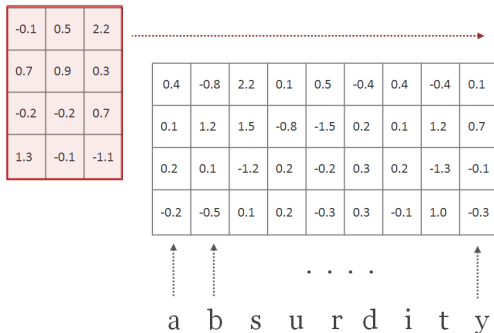
0.4	-0.8	2.2	0.1	0.5	-0.4	0.4	-0.4	0.1
0.1	1.2	1.5	-0.8	-1.5	0.2	0.1	1.2	0.7
0.2	0.1	-1.2	0.2	-0.2	0.3	0.2	-1.3	-0.1
-0.2	-0.5	0.1	0.2	-0.3	0.3	-0.1	1.0	-0.3

↑   ↑   . . .   ↑

a   b   s   u   r   d   i   t   y

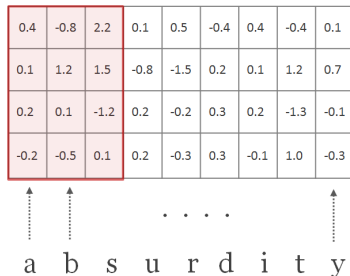
# Character-level CNN (CharCNN)

$\mathbf{H} \in \mathbb{R}^{d \times w}$  : Convolutional filter matrix of width  $w = 3$



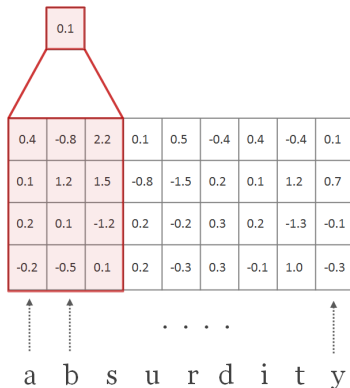
# Character-level CNN (CharCNN)

$$\mathbf{f}[1] = \langle \mathbf{C}[:, 1:3], \mathbf{H} \rangle$$



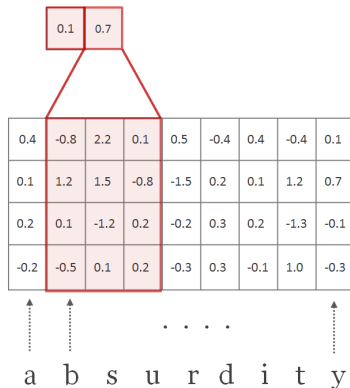
# Character-level CNN (CharCNN)

$$\mathbf{f}[1] = \langle \mathbf{C}[:, 1:3], \mathbf{H} \rangle$$



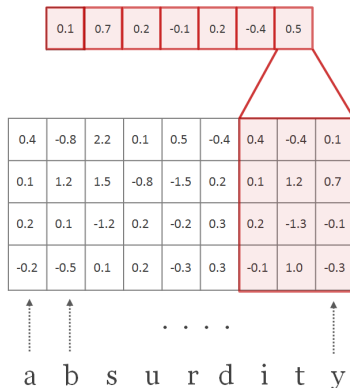
# Character-level CNN (CharCNN)

$$\mathbf{f}[2] = \langle \mathbf{C}[:, 2:4], \mathbf{H} \rangle$$



# Character-level CNN (CharCNN)

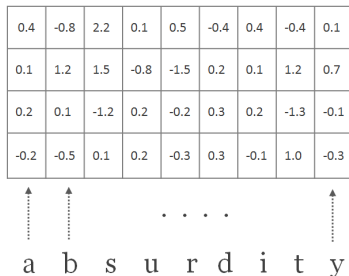
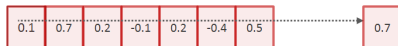
$$\mathbf{f}[T-2] = \langle \mathbf{C}[* , T-2 : T], \mathbf{H} \rangle$$





# Character-level CNN (CharCNN)

$$y[1] = \max_i \{f[i]\}$$



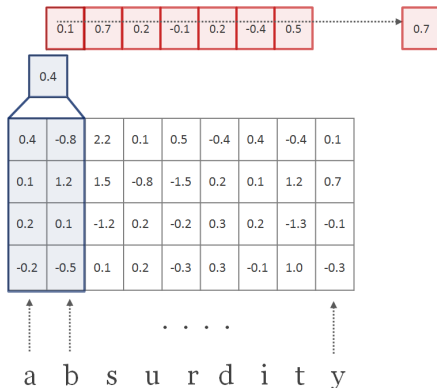
# Character-level CNN (CharCNN)

Each filter picks out a character  $n$ -gram



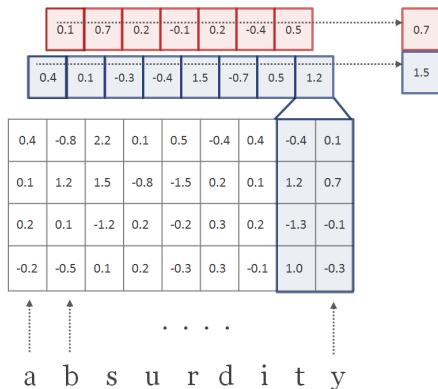
# Character-level CNN (CharCNN)

$$\mathbf{f}'[1] = \langle \mathbf{C}[*], 1 : 2], \mathbf{H}' \rangle$$



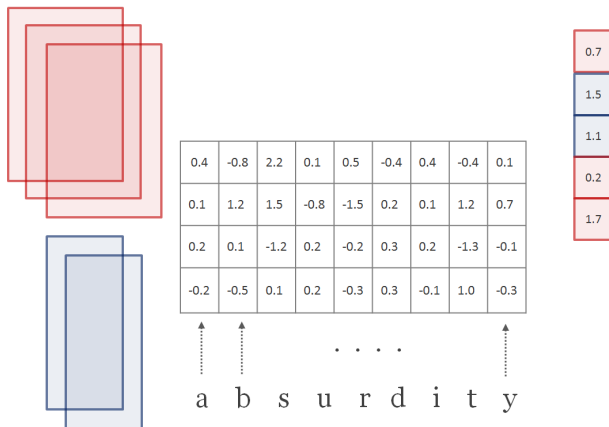
# Character-level CNN (CharCNN)

$$y[2] = \max_i \{f'[i]\}$$



# Character-level CNN (CharCNN)

Many filter matrices (25–200) per width (1–7)



# Learned Word Representations (In Vocab)

(Based on cosine similarity)

	In Vocabulary				
	<i>while</i>	<i>his</i>	<i>you</i>	<i>richard</i>	<i>trading</i>
Word Embedding	<i>although</i>	<i>your</i>	<i>conservatives</i>	<i>jonathan</i>	<i>advertised</i>
	<i>letting</i>	<i>her</i>	<i>we</i>	<i>robert</i>	<i>advertising</i>
	<i>though</i>	<i>my</i>	<i>guys</i>	<i>neil</i>	<i>turnover</i>
	<i>minute</i>	<i>their</i>	<i>i</i>	<i>nancy</i>	<i>turnover</i>
<b>Characters</b> (before highway)	<i>chile</i>	<i>this</i>	<i>your</i>	<i>hard</i>	<i>heading</i>
	<i>whole</i>	<i>hhs</i>	<i>young</i>	<i>rich</i>	<i>training</i>
	<i>meanwhile</i>	<i>is</i>	<i>four</i>	<i>richer</i>	<i>reading</i>
	<i>white</i>	<i>has</i>	<i>youth</i>	<i>richter</i>	<i>leading</i>
<b>Characters</b> (after highway)	<i>meanwhile</i>	<i>hhs</i>	<i>we</i>	<i>eduard</i>	<i>trade</i>
	<i>whole</i>	<i>this</i>	<i>your</i>	<i>gerard</i>	<i>training</i>
	<i>though</i>	<i>their</i>	<i>doug</i>	<i>edward</i>	<i>traded</i>
	<i>nevertheless</i>	<i>your</i>	<i>i</i>	<i>carl</i>	<i>trader</i>

# Learned Word Representations (In Vocab)

(Based on cosine similarity)

	In Vocabulary				
	<i>while</i>	<i>his</i>	<i>you</i>	<i>richard</i>	<i>trading</i>
Word Embedding	<i>although</i>	<i>your</i>	<i>conservatives</i>	<i>jonathan</i>	<i>advertised</i>
	<i>letting</i>	<i>her</i>	<i>we</i>	<i>robert</i>	<i>advertising</i>
	<i>though</i>	<i>my</i>	<i>guys</i>	<i>neil</i>	<i>turnover</i>
	<i>minute</i>	<i>their</i>	<i>i</i>	<i>nancy</i>	<i>turnover</i>
<b>Characters</b> (before highway)	<i>chile</i>	<i>this</i>	<i>your</i>	<i>hard</i>	<i>heading</i>
	<i>whole</i>	<i>hhs</i>	<i>young</i>	<i>rich</i>	<i>training</i>
	<i>meanwhile</i>	<i>is</i>	<i>four</i>	<i>richer</i>	<i>reading</i>
	<i>white</i>	<i>has</i>	<i>youth</i>	<i>richter</i>	<i>leading</i>
<b>Characters</b> (after highway)	<i>meanwhile</i>	<i>hhs</i>	<i>we</i>	<i>eduard</i>	<i>trade</i>
	<i>whole</i>	<i>this</i>	<i>your</i>	<i>gerard</i>	<i>training</i>
	<i>though</i>	<i>their</i>	<i>doug</i>	<i>edward</i>	<i>traded</i>
	<i>nevertheless</i>	<i>your</i>	<i>i</i>	<i>carl</i>	<i>trader</i>

# Learned Word Representations (OOV)

Out-of-Vocabulary			
	<i>computer-aided</i>	<i>misinformed</i>	<i>loooooook</i>
<b>Characters</b> (before highway)	<i>computer-guided</i>	<i>informed</i>	<i>look</i>
	<i>computerized</i>	<i>performed</i>	<i>cook</i>
	<i>disk-drive</i>	<i>transformed</i>	<i>looks</i>
	<i>computer</i>	<i>inform</i>	<i>shook</i>
<b>Characters</b> (after highway)	<i>computer-guided</i>	<i>informed</i>	<i>look</i>
	<i>computer-driven</i>	<i>performed</i>	<i>looks</i>
	<i>computerized</i>	<i>outperformed</i>	<i>looked</i>
	<i>computer</i>	<i>transformed</i>	<i>looking</i>

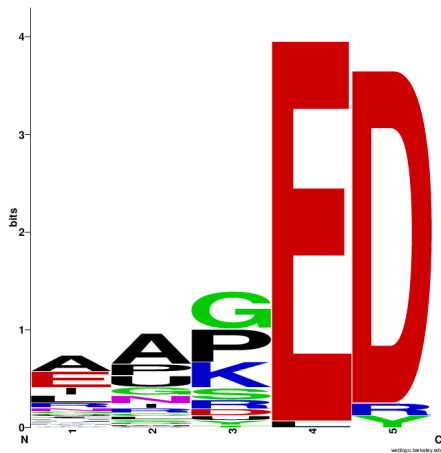


# Learned Word Representations (OOV)

Out-of-Vocabulary			
<b>Characters</b> (before highway)	<i>computer-aided</i>	<i>misinformed</i>	<i>loooooook</i>
	<i>computer-guided</i>	<i>informed</i>	<i>look</i>
	<i>computerized</i>	<i>performed</i>	<i>cook</i>
	<i>disk-drive</i>	<i>transformed</i>	<i>looks</i>
	<i>computer</i>	<i>inform</i>	<i>shook</i>
<b>Characters</b> (after highway)	<i>computer-guided</i>	<i>informed</i>	<i>look</i>
	<i>computer-driven</i>	<i>performed</i>	<i>looks</i>
	<i>computerized</i>	<i>outperformed</i>	<i>looked</i>
	<i>computer</i>	<i>transformed</i>	<i>looking</i>

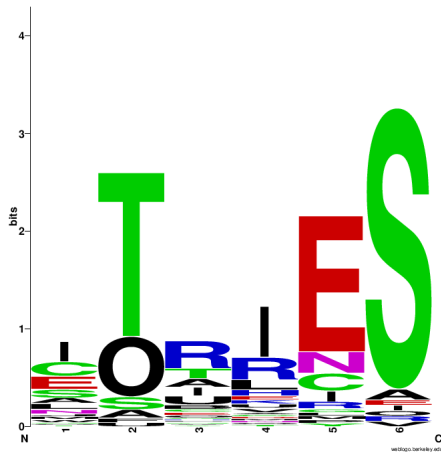
# Convolutional Filters

For each filter, visualize 100 substrings with the highest filter response

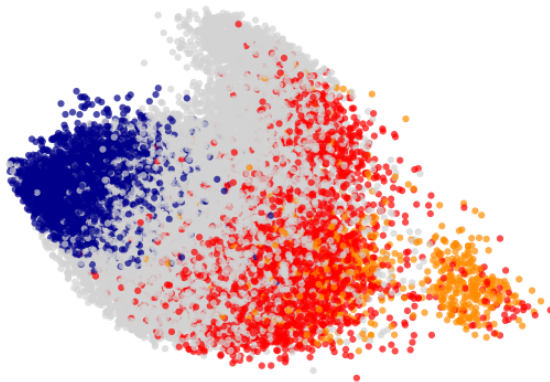


# Convolutional Filters

For each filter, visualize 100 substrings with the highest filter response



# Character $N$ -gram Representations



Prefixes, Suffixes, Hyphenated, Others

Prefixes: character  $n$ -grams that start with 'start-of-word' character, such as  $\{un, \{mis$ . Suffixes defined similarly.

# Contents

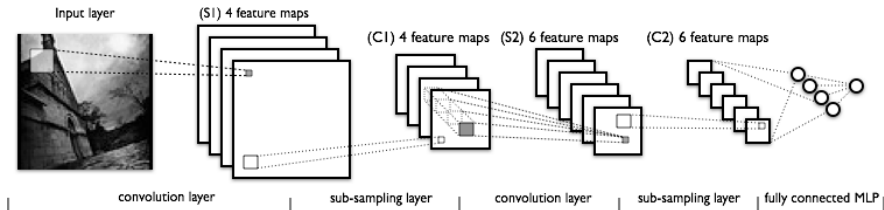
Text Classification Review

Convolutions

Applications

Vision

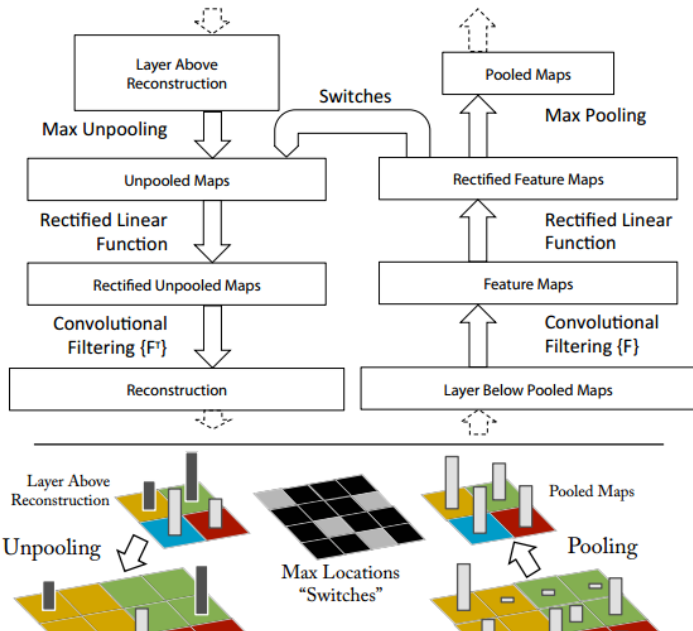
# Visual Classification



## Speech Convolutions

softmax
fully connected, 4096
fully connected, 4096
max pooling, $2 \times$
convolution, $3 \times 3$ , 384
convolution, $3 \times 3$ , 384
convolution, $3 \times 3$ , 384
max pooling, $2 \times 2$
convolution, $3 \times 3$ , 192
convolution, $3 \times 3$ , 192
convolution, $3 \times 3$ , 192
max pooling, $2 \times 2$
convolution, $3 \times 3$ , 96
convolution, $3 \times 3$ , 96
input (31x41)

# Visualization (Zeiler and Fergus, 2014)





# Visualization (Zeiler and Fergus, 2014)

