CNNs

Wei Xu

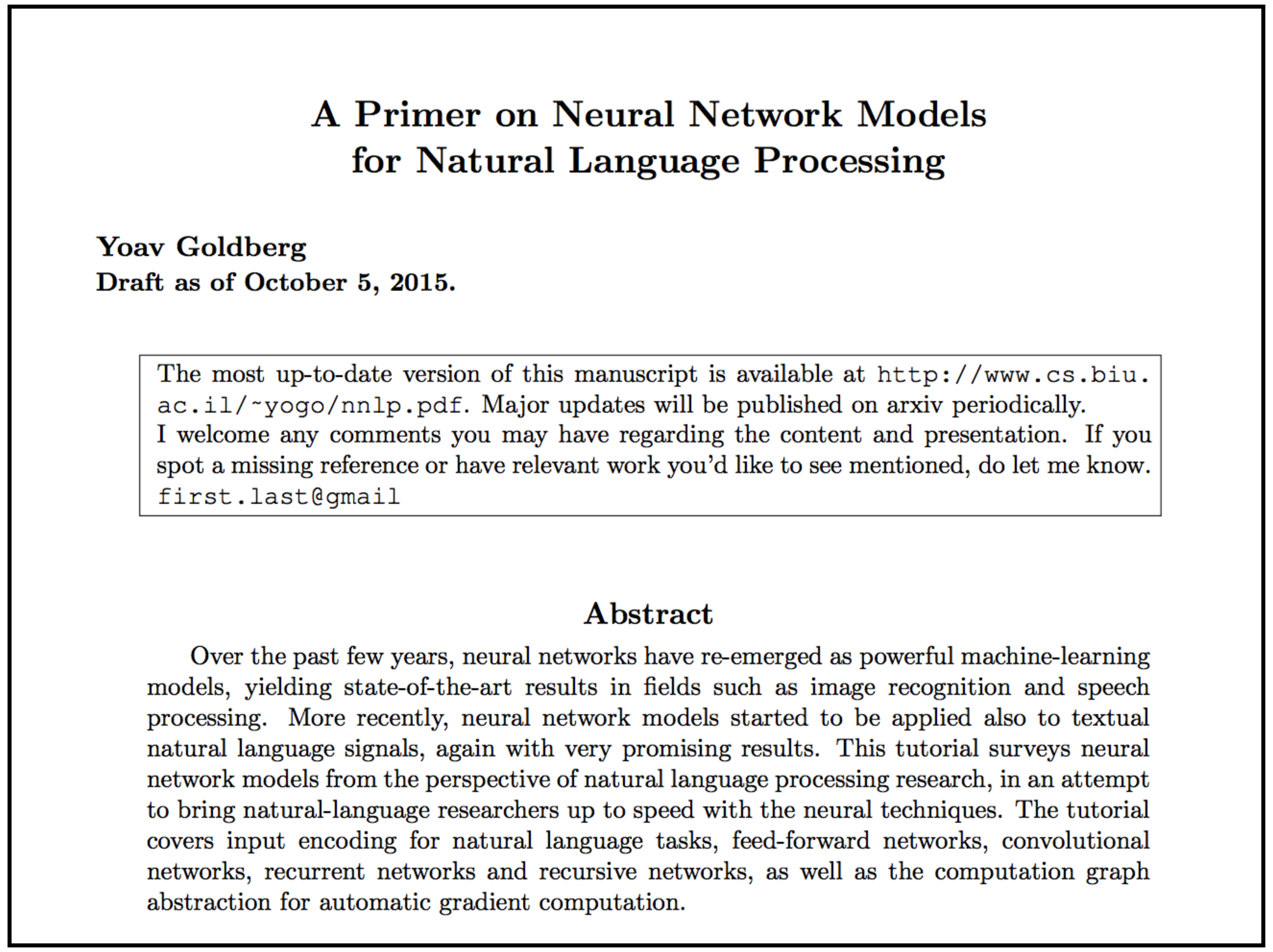
(many slides from Greg Durrett, Stanford 231n)

This Lecture

- CNNs
- CNNs for Sentiment, Entity Linking

Administrivia

- Reading — Goldberg 9 (CNN); Eisenstein 3.4, 7.6



**A Primer on Neural Network Models
for Natural Language Processing**

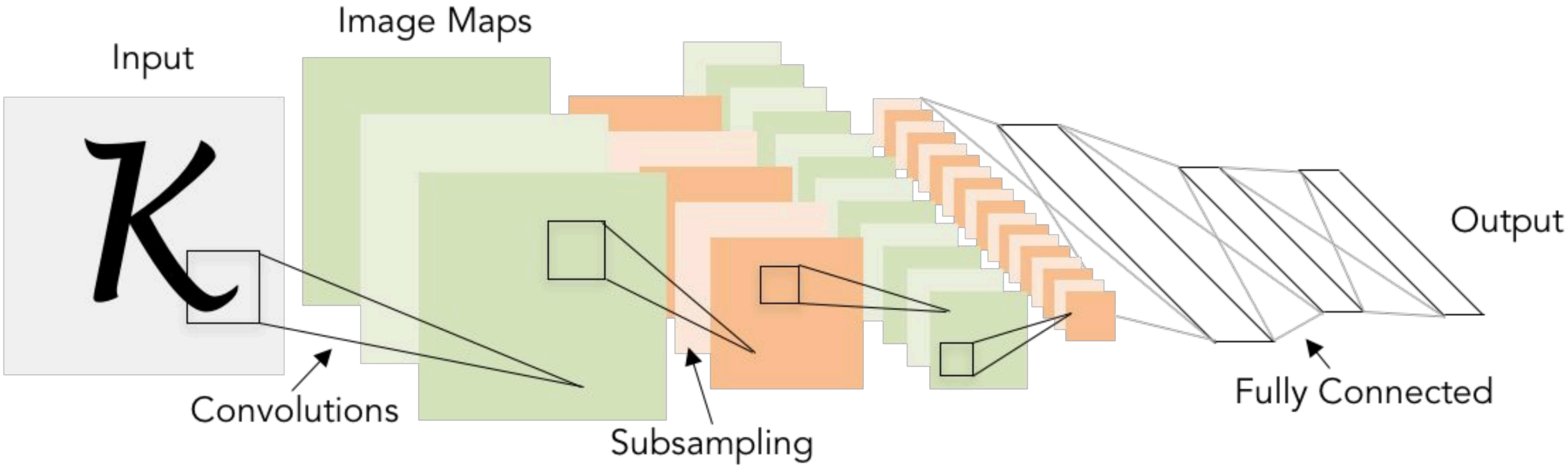
Yoav Goldberg
Draft as of October 5, 2015.

The most up-to-date version of this manuscript is available at `http://www.cs.biu.ac.il/~yogo/nnlp.pdf`. Major updates will be published on arxiv periodically.
I welcome any comments you may have regarding the content and presentation. If you spot a missing reference or have relevant work you'd like to see mentioned, do let me know.
`first.last@gmail`

Abstract

Over the past few years, neural networks have re-emerged as powerful machine-learning models, yielding state-of-the-art results in fields such as image recognition and speech processing. More recently, neural network models started to be applied also to textual natural language signals, again with very promising results. This tutorial surveys neural network models from the perspective of natural language processing research, in an attempt to bring natural-language researchers up to speed with the neural techniques. The tutorial covers input encoding for natural language tasks, feed-forward networks, convolutional networks, recurrent networks and recursive networks, as well as the computation graph abstraction for automatic gradient computation.

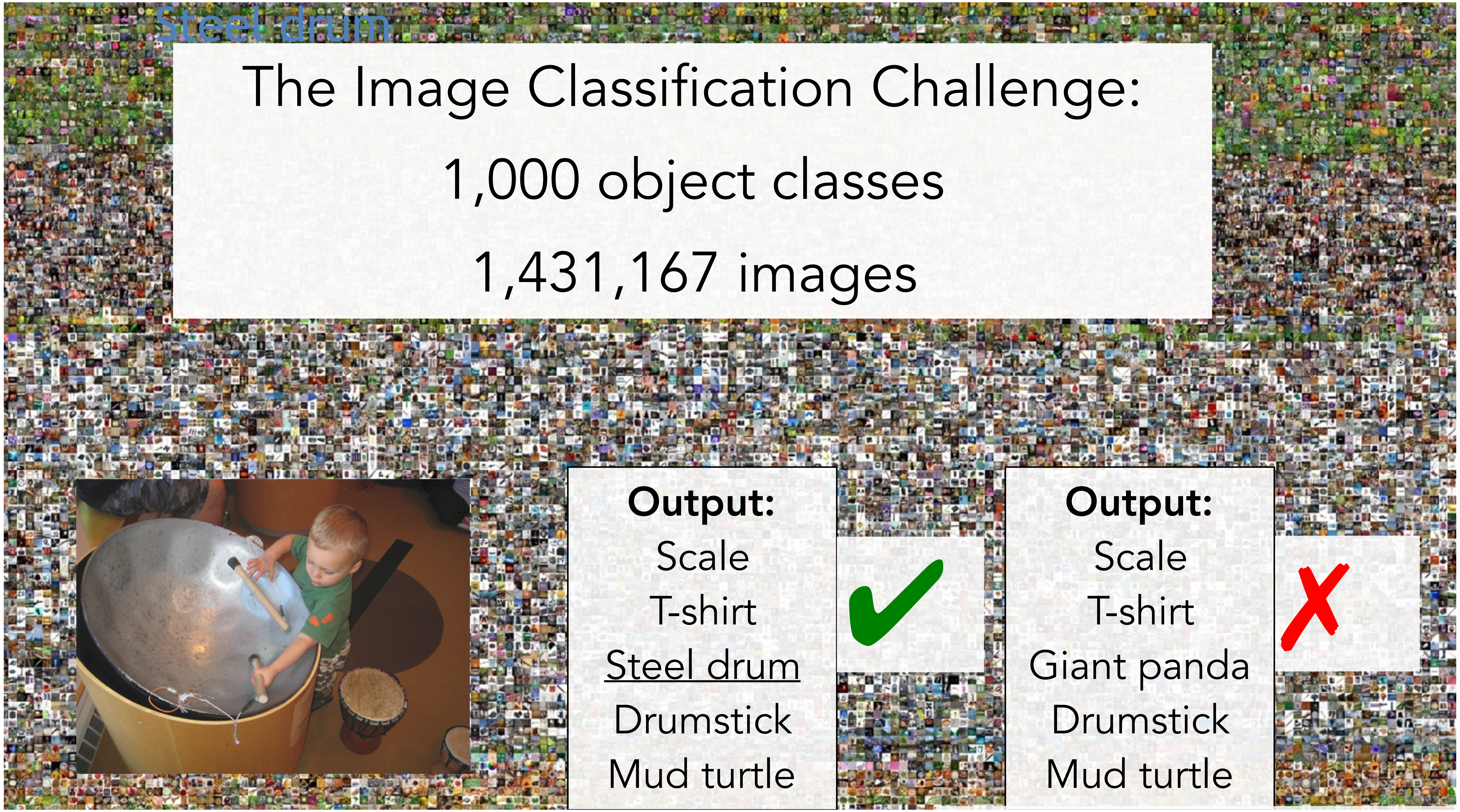
A Bit of History



https://www.youtube.com/watch?v=FwFduRA_L6Q

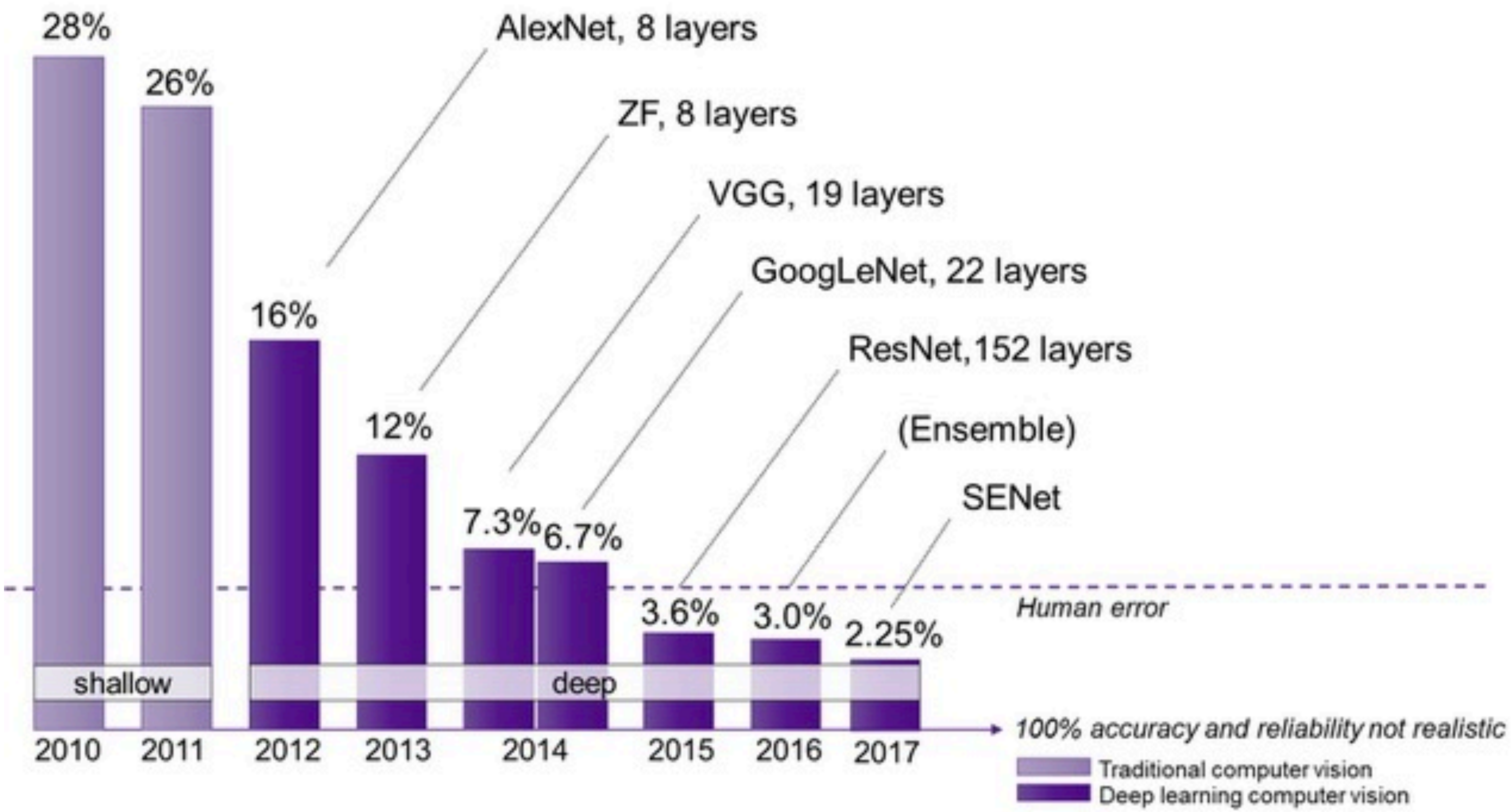
LeCun et al. (1998), earlier work in 1980s

ImageNet - Object Recognition



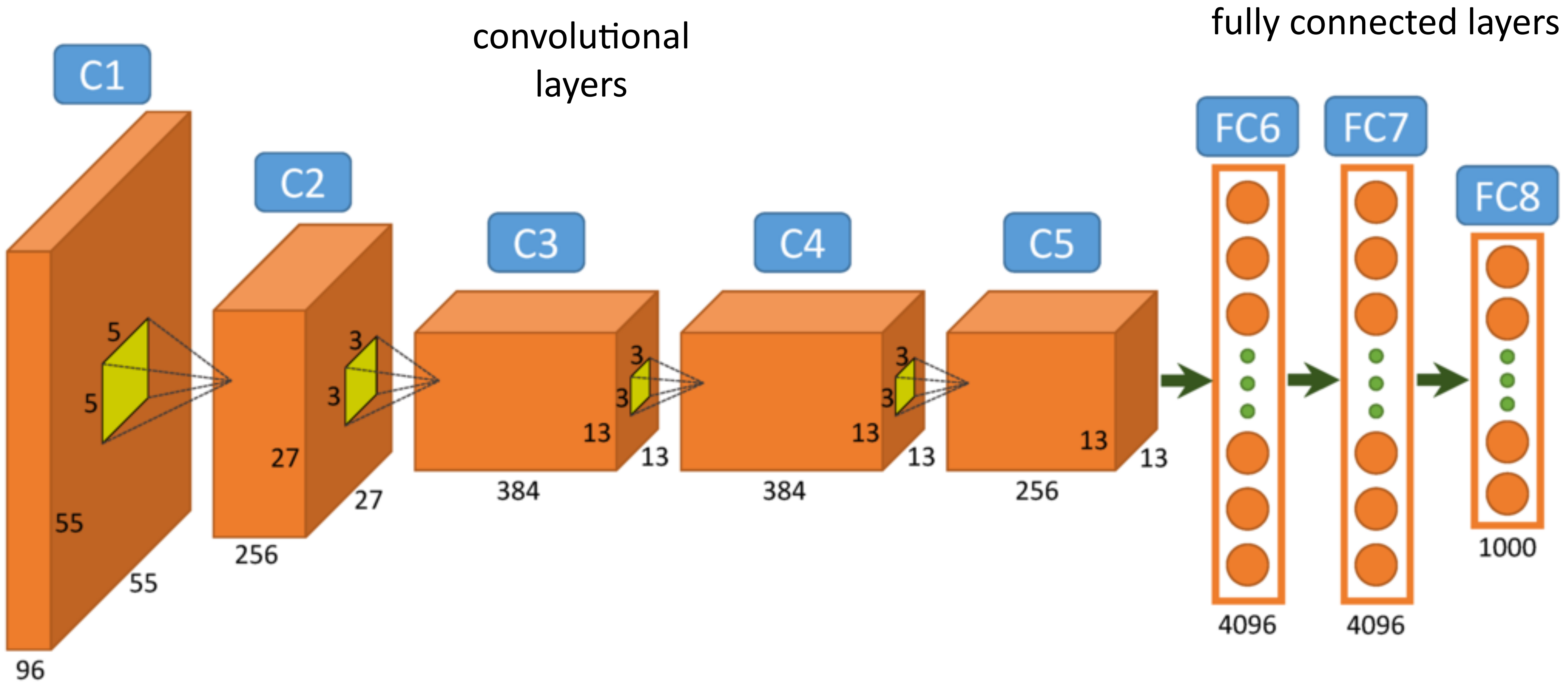
Russakovsky et al. (2012)

ImageNet - Object Recognition



Convolutional Neural Networks

- AlexNet - one of the first strong results
- more filters per layer as well as stacked convolutional layers
- use of ReLU for the non-linear part instead of Sigmoid or Tanh

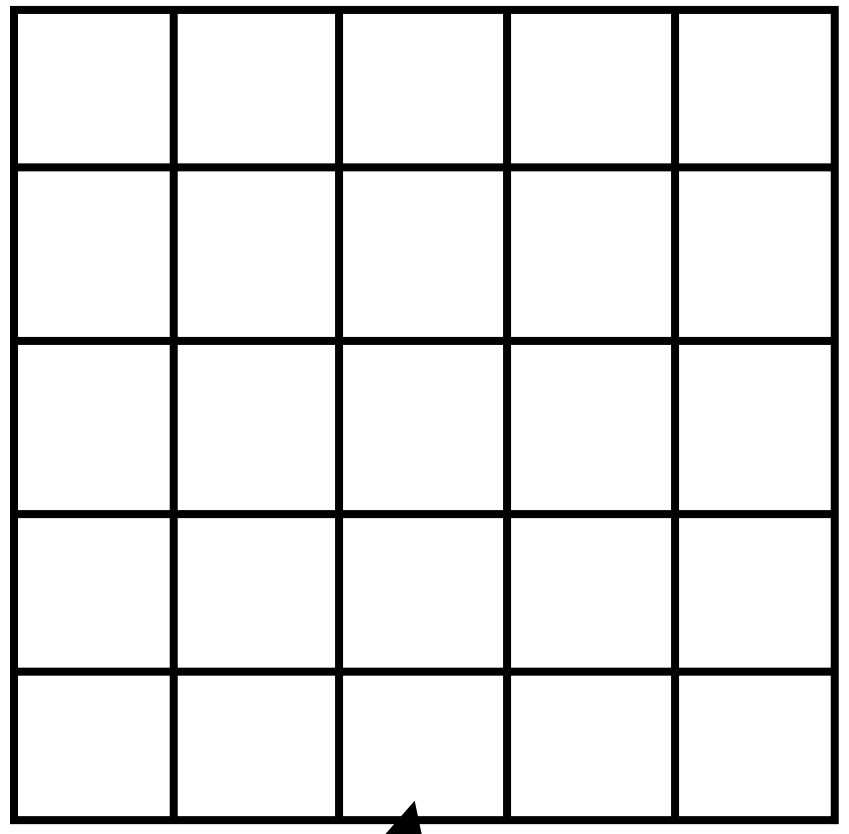


Krizhevsky et al. (2012)

Convolutional Layer

- Applies a *filter* over patches of the input and returns that filter's activations
- Convolution: take dot product of filter with a patch of the input

image: n x n x k



filter: m x m x k

sum over dot products

$$\text{activation}_{ij} = \sum_{i_o=0}^{m-1} \sum_{j_o=0}^{m-1} \text{image}(i + i_o, j + j_o) \cdot \text{filter}(i_o, j_o)$$

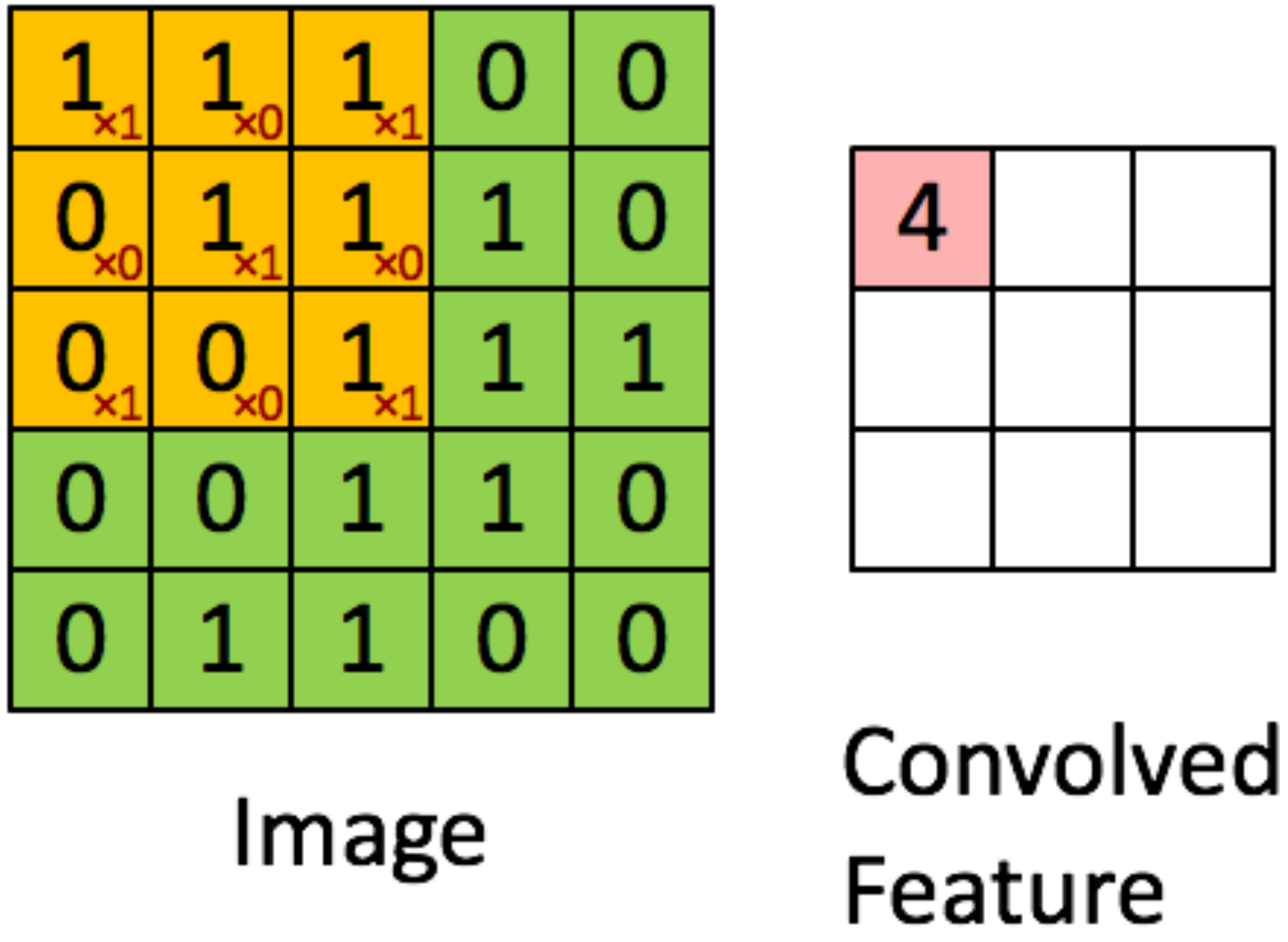
offsets

Each of these cells is a vector with multiple values

Images: RGB values (k=3 dim)

Convolutional Layer

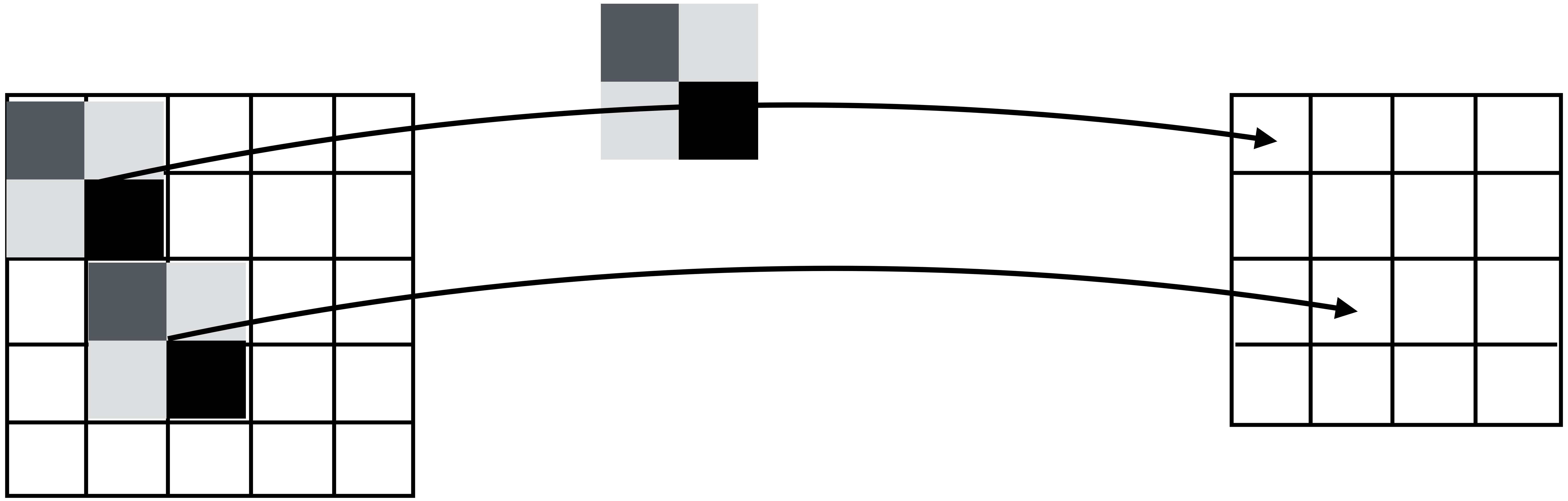
- An animated example: k = 1, and a filter of size 3x3.



Convolutional Layer

- Applies a *filter* over patches of the input and returns that filter's activations
- Convolution: take dot product of filter with a patch of the input

image: n x n x k filter: m x m x k activations: (n - m + 1) x (n - m + 1) x 1



Convolutions for NLP

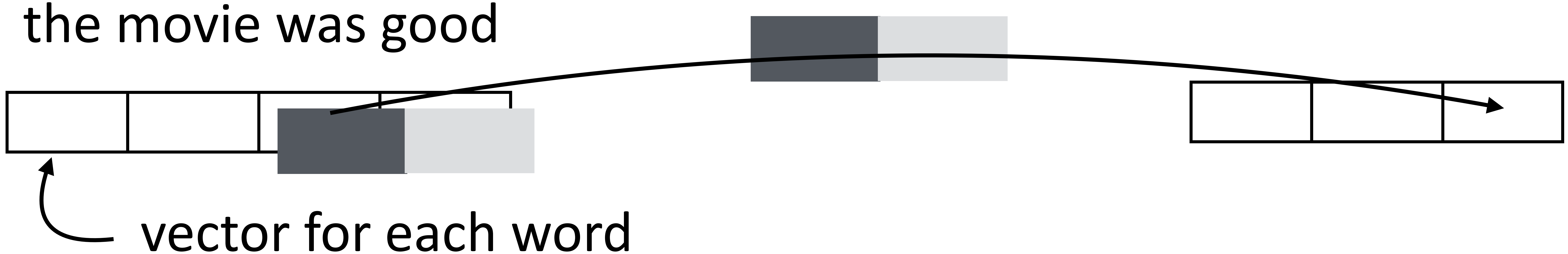
- Input and filter are 2-dimensional instead of 3-dimensional

sentence: n words x k vec dim

filter: m x k

activations: (n - m + 1) x 1

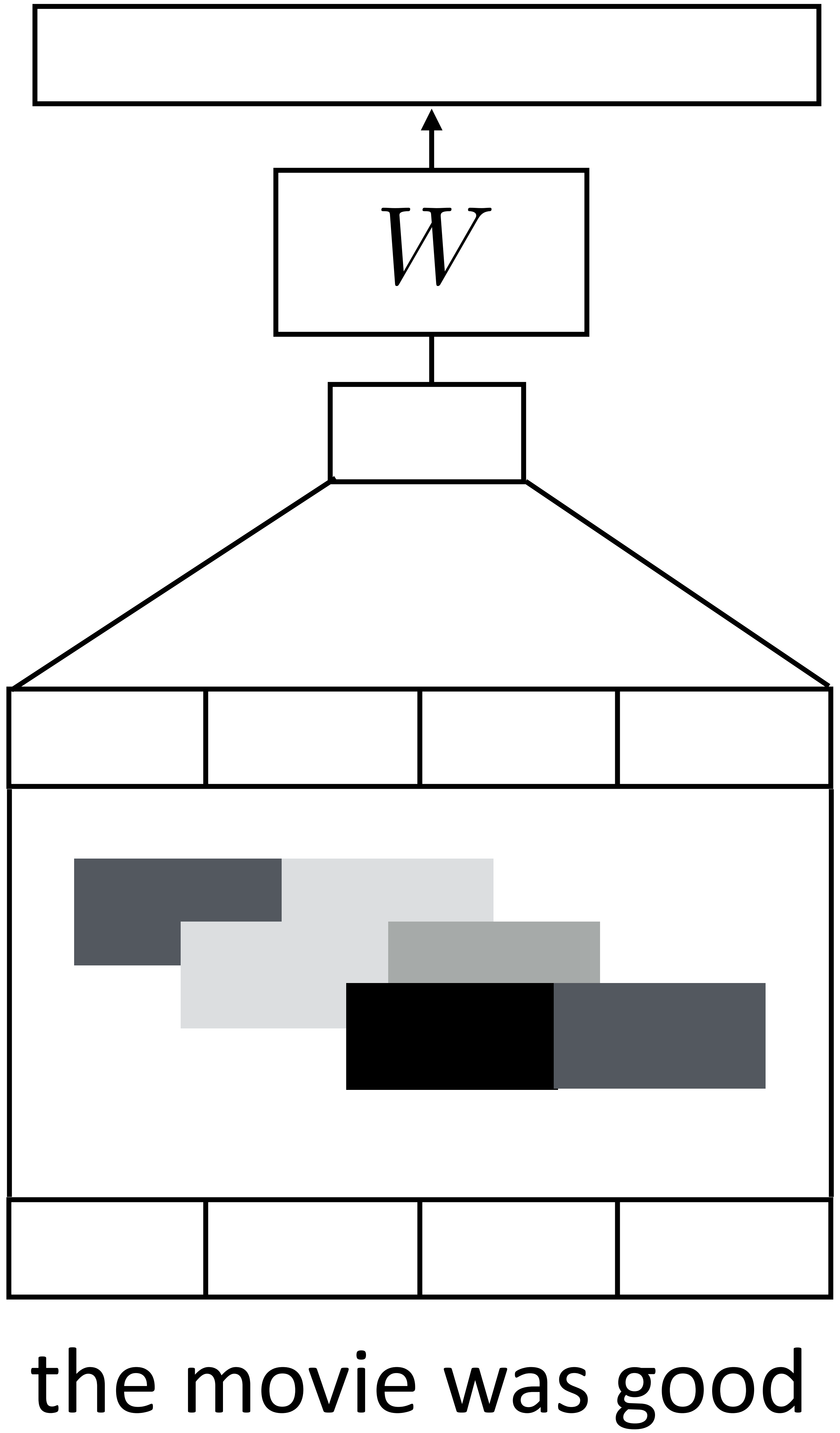
the movie was good



- Combines evidence locally in a sentence and produces a new (but still variable-length) representation

CNNs for Sentiment

CNNs for Sentiment Analysis



$P(y|\mathbf{x})$

projection + softmax

c-dimensional vector

max pooling over the sentence

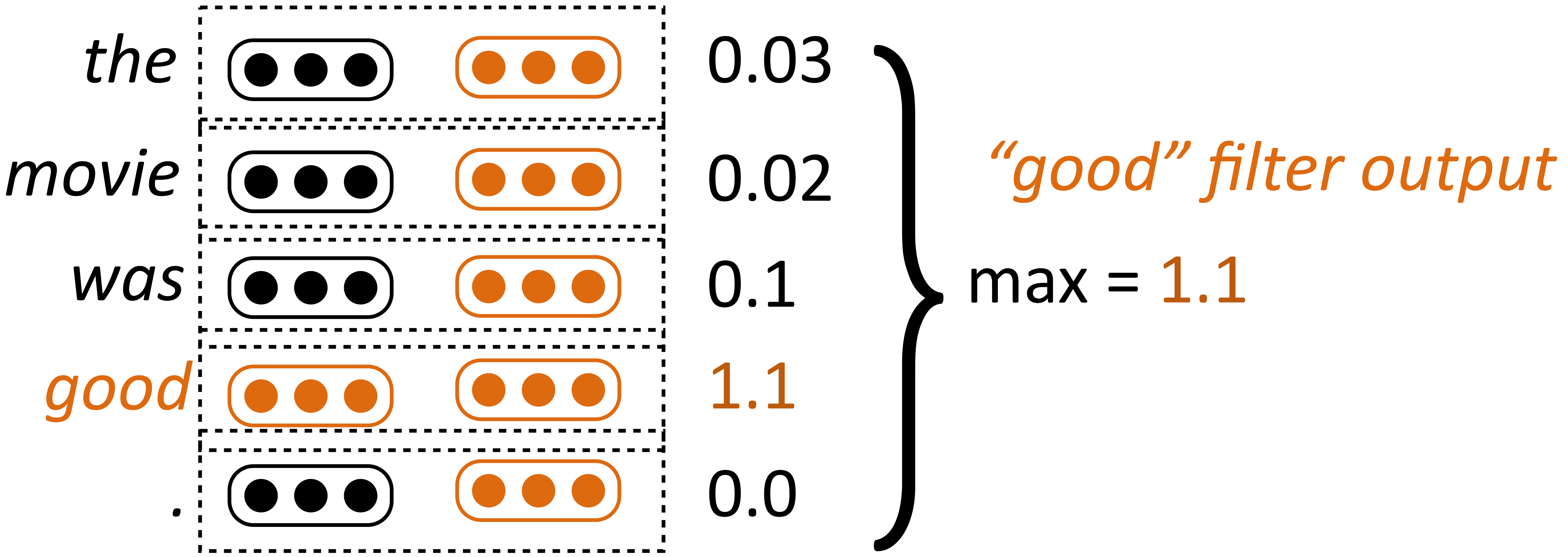
n x c

c filters,
m x k each

n x k

- Max pooling: return the max activation of a given filter over the entire sentence; like a logical OR (sum pooling is like logical AND)

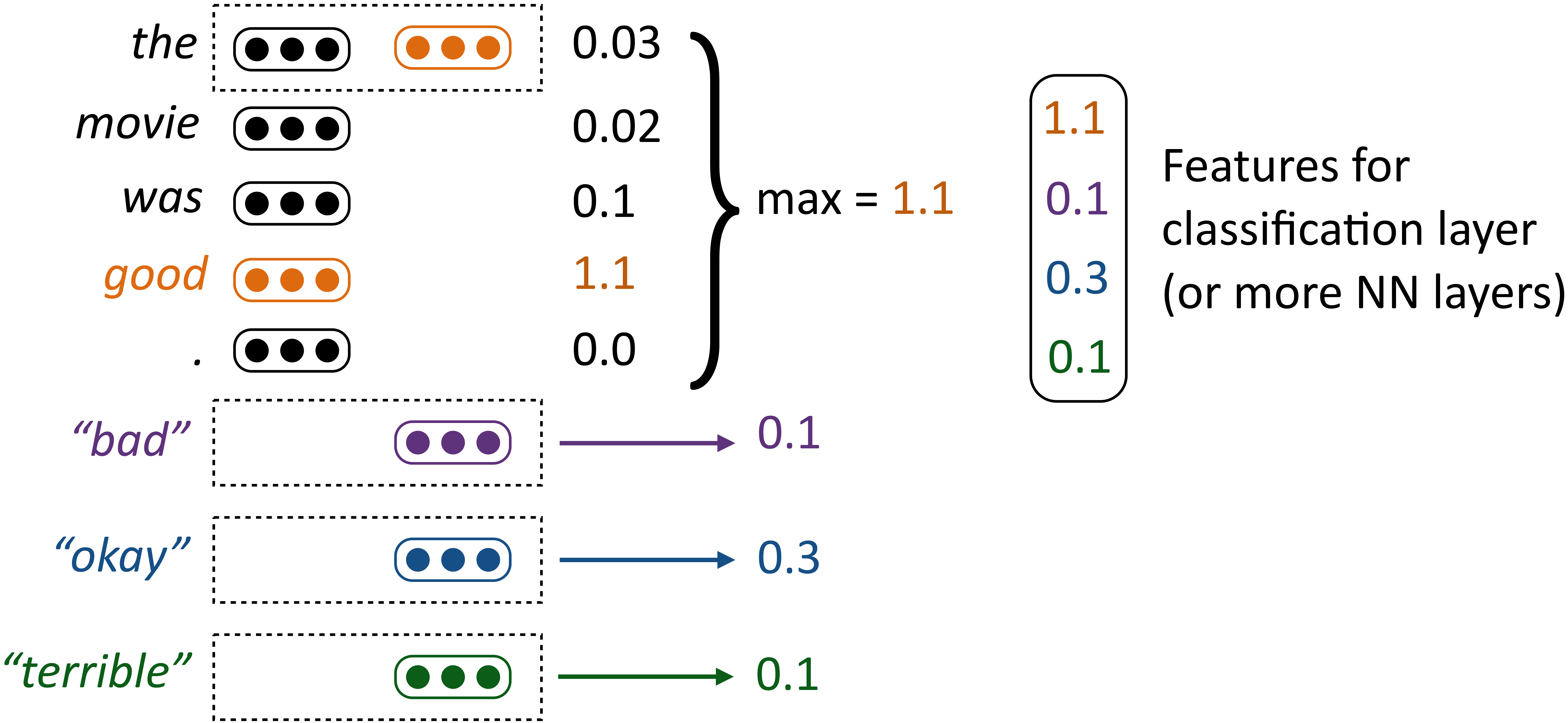
Understanding CNNs for Sentiment



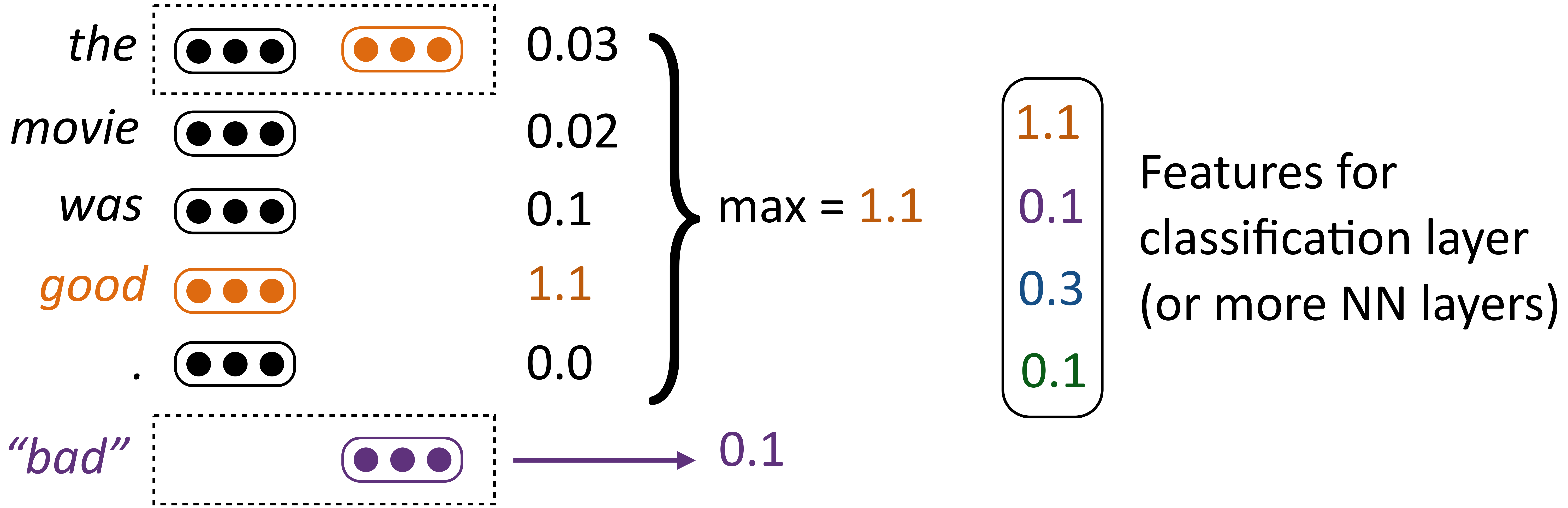
- Filter "looks like" the things that will cause it to have high activation

Understanding CNNs for Sentiment

Understanding CNNs for Sentiment

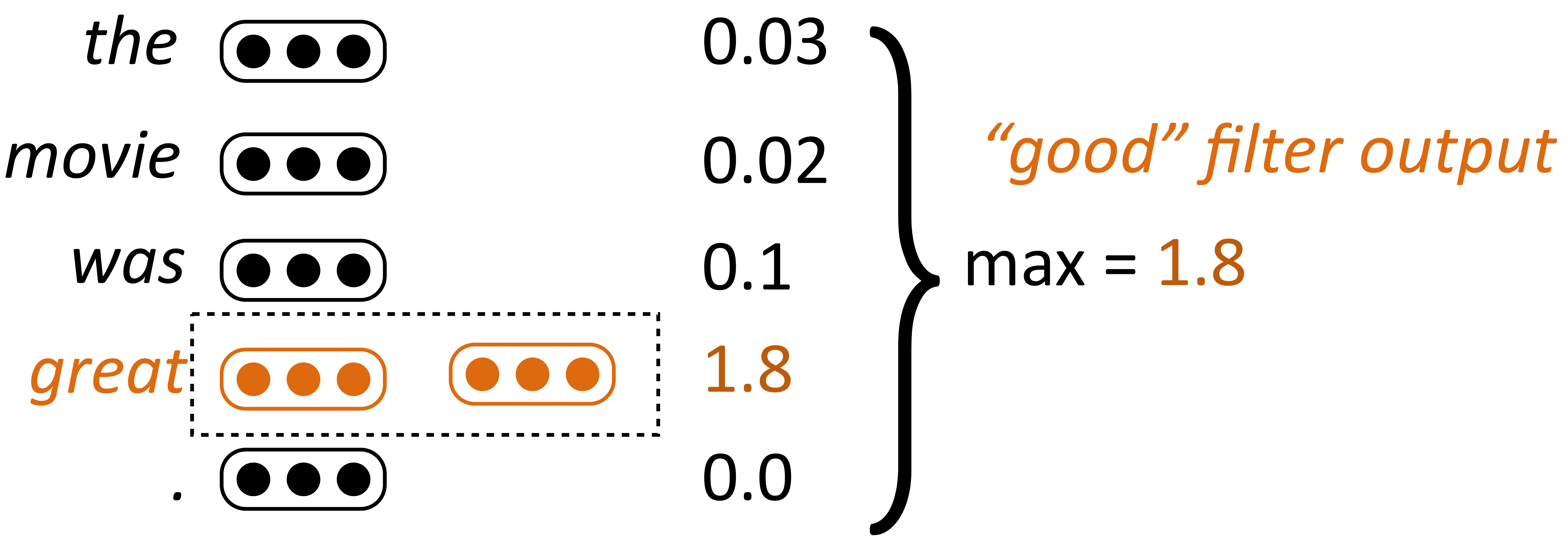


Understanding CNNs for Sentiment



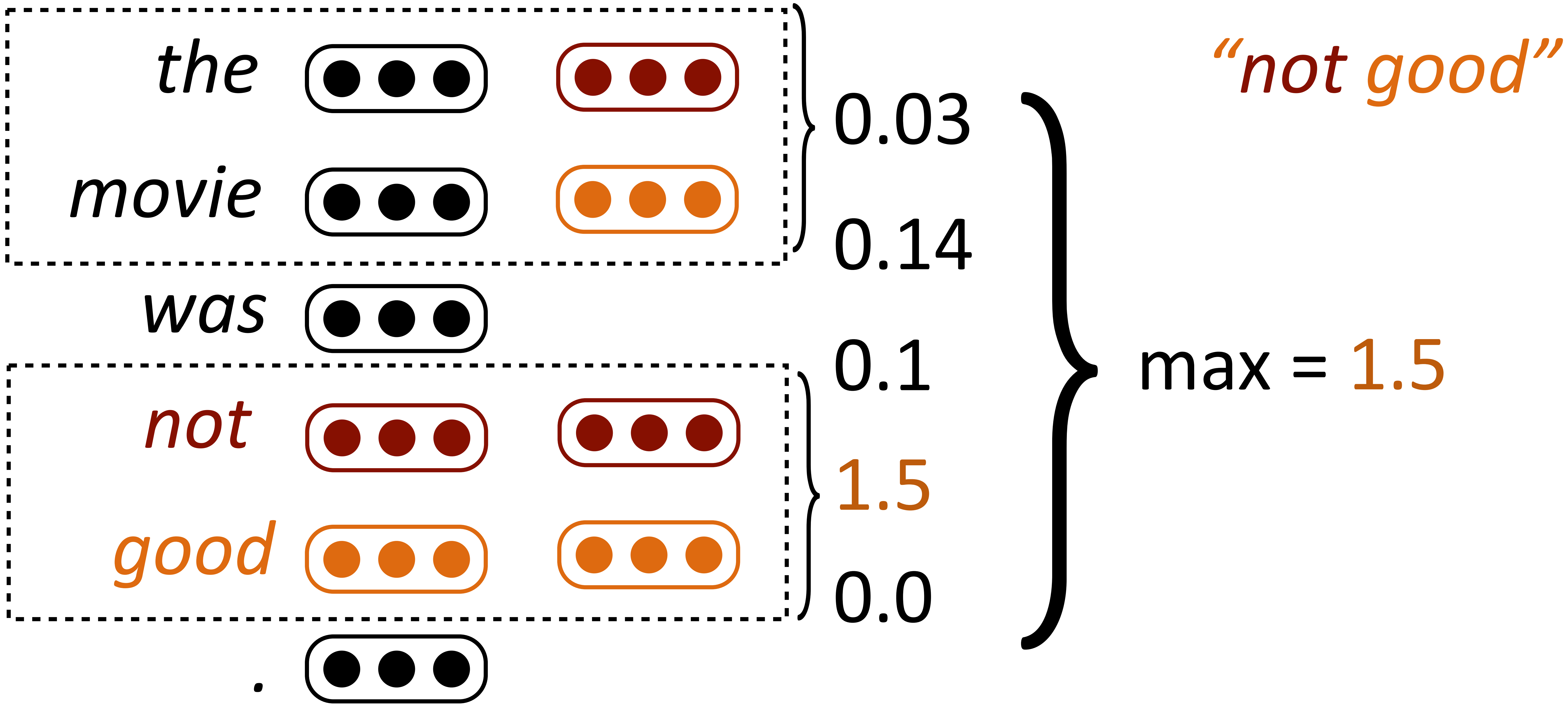
- Takes variable-length input and turns it into fixed-length output
- Filters are initialized randomly and then learned

Understanding CNNs for Sentiment



- Word vectors for similar words are similar, so convolutional filters will have similar outputs

Understanding CNNs for Sentiment



- Analogous to bigram features in bag-of-words models
- Indicator feature of text containing bigram <-> max pooling of a filter that matches that bigram

What can CNNs learn?

- CNNs let us take advantage of word similarity

really not very good vs. *really not very enjoyable*

- CNNs are translation-invariant like bag-of-words

The movie was bad, but blah blah blah … vs. *… blah blah blah, but the movie was bad.*

- CNNs can capture local interactions with filters of width > 1

It was not good, it was actually quite bad vs. *it was not bad, it was actually quite good*

CNNs: Implementation

- Input is batch_size x n x k matrix, filters are c x m x k matrix (c filters)

- Typically use filters with m ranging from 1 to 5 or so (multiple filter widths in a single convnet)

- All computation graph libraries support efficient convolution operations

CLASS `torch.nn.Conv1d(in_channels, out_channels, kernel_size, stride=1, padding=0, dilation=1, groups=1, bias=True, padding_mode='zeros')` [SOURCE]

Applies a 1D convolution over an input signal composed of several input planes.

In the simplest case, the output value of the layer with input size (N, C_{in}, L) and output $(N, C_{\text{out}}, L_{\text{out}})$ can be precisely described as:

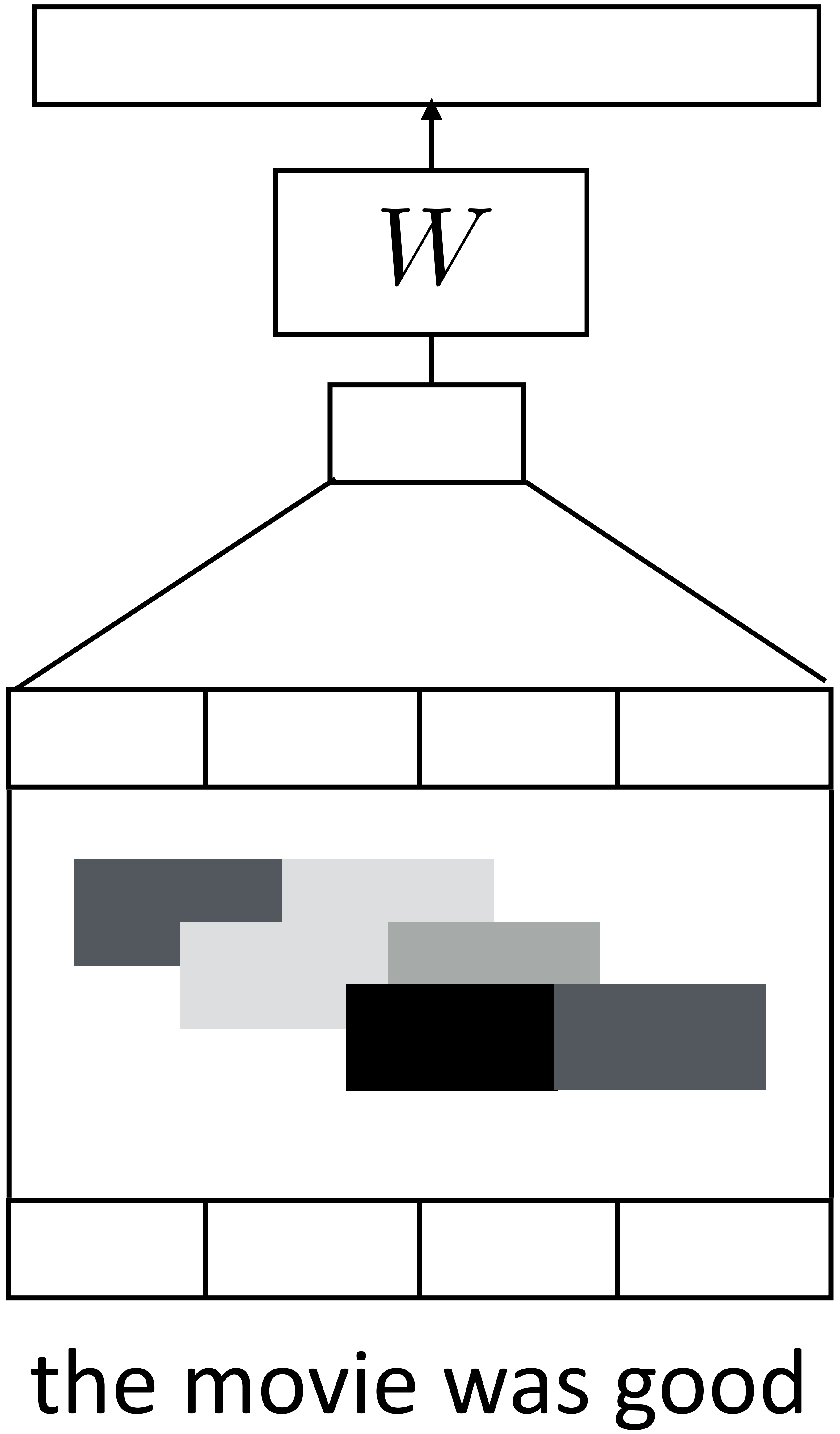
$$\text{out}(N_i, C_{\text{out}_j}) = \text{bias}(C_{\text{out}_j}) + \sum_{k=0}^{C_{in}-1} \text{weight}(C_{\text{out}_j}, k) \star \text{input}(N_i, k)$$

where $\star$ is the valid cross-correlation operator, N is a batch size, C denotes a number of channels, L is a length of signal sequence.

- `stride` controls the stride for the cross-correlation, a single number or a one-element tuple.
- `padding` controls the amount of implicit zero-paddings on both sides for `padding` number of points.

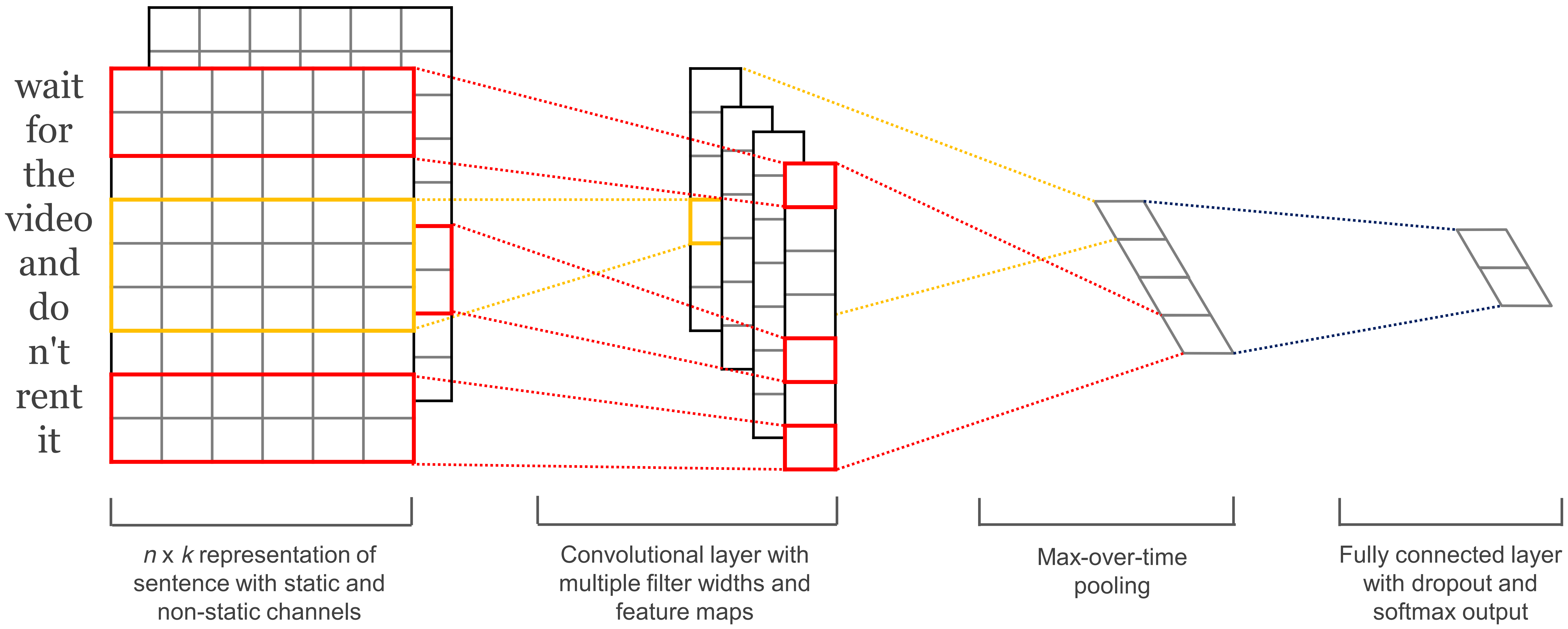
CNNs for Sentence Classification

- Question classification, sentiment, etc.
- Conv+pool, then use feedforward layers to classify
- Can use multiple types of input vectors (fixed initializer and learned)



Kim (2014)

CNNs for Sentence Classification



Kim (2014)

Sentence Classification

movie review sentiment

subjectivity/objectivity detection

product reviews

Model	MR	SST-1	SST-2	Subj	TREC	CR	MPQA
CNN-multichannel	81.1	47.4	**88.1**	93.2	92.2	**85.0**	89.4
NBSVM (Wang and Manning, 2012)	79.4	–	–	93.2	–	81.8	86.3

question type classification

- Also effective at document-level text classification

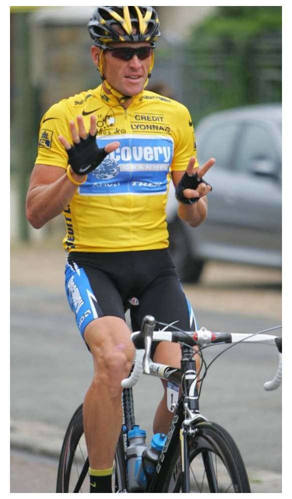
Kim (2014)

Entity Linking

- CNNs can produce good representations of both sentences and documents like typical bag-of-words features
- Can distill topic representations for use in entity linking

that they had disqualified **Armstrong** from his seven consecutive

cycling domain



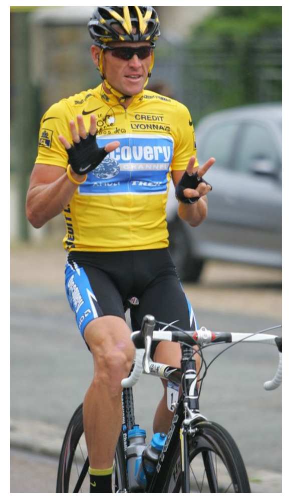
Lance Armstrong

geopolitical domain

Armstrong County

Entity Linking

Although he originally won the event, the United States Anti-Doping Agency announced in August 2012 that they had disqualified $\boxed{\textbf{Armstrong}}$ from his seven consecutive Tour de France wins from 1999–2005.



Lance Edward Armstrong is an American former professional road cyclist

Armstrong County is a county in Pennsylvania…

CNN

Document topic vector d

CNN

Article topic vector a_{Lance}

CNN

Article topic vector a_{County}

$$s_{\text{Lance}} = d \cdot a_{\text{Lance}} \qquad s_{\text{County}} = d \cdot a_{\text{County}}$$

$$P(y|\mathbf{x}) = \text{softmax}(\mathbf{s})$$

Francis-Landau et al. (2016)

Entity Linking

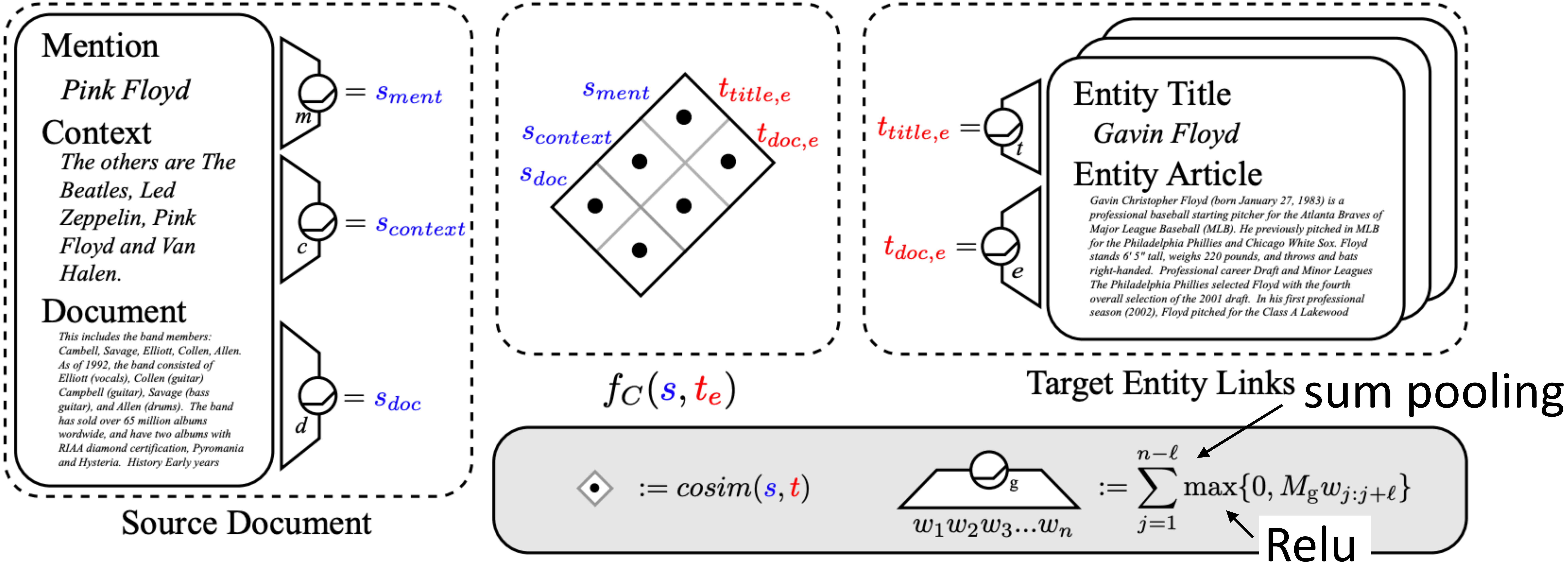
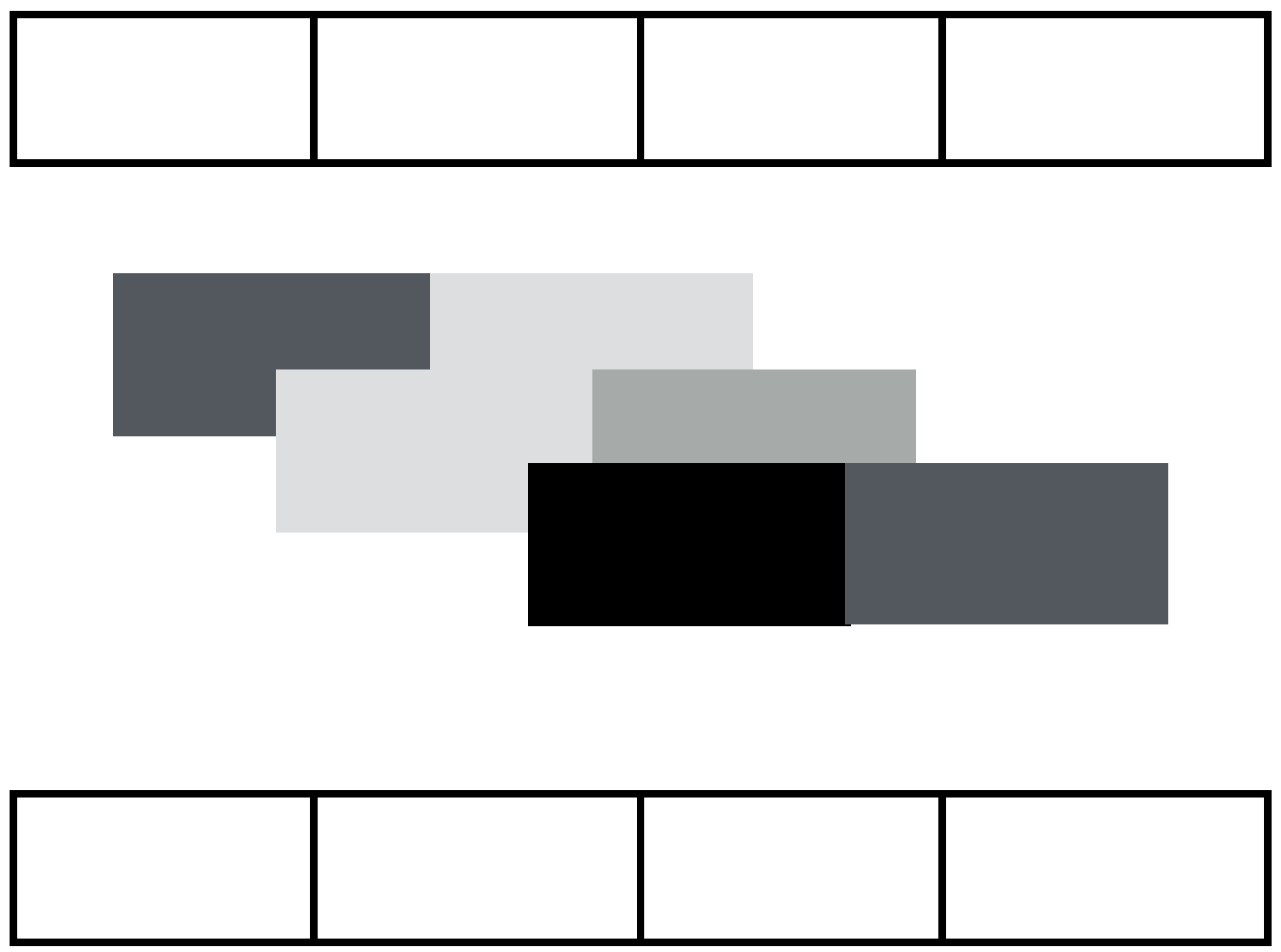
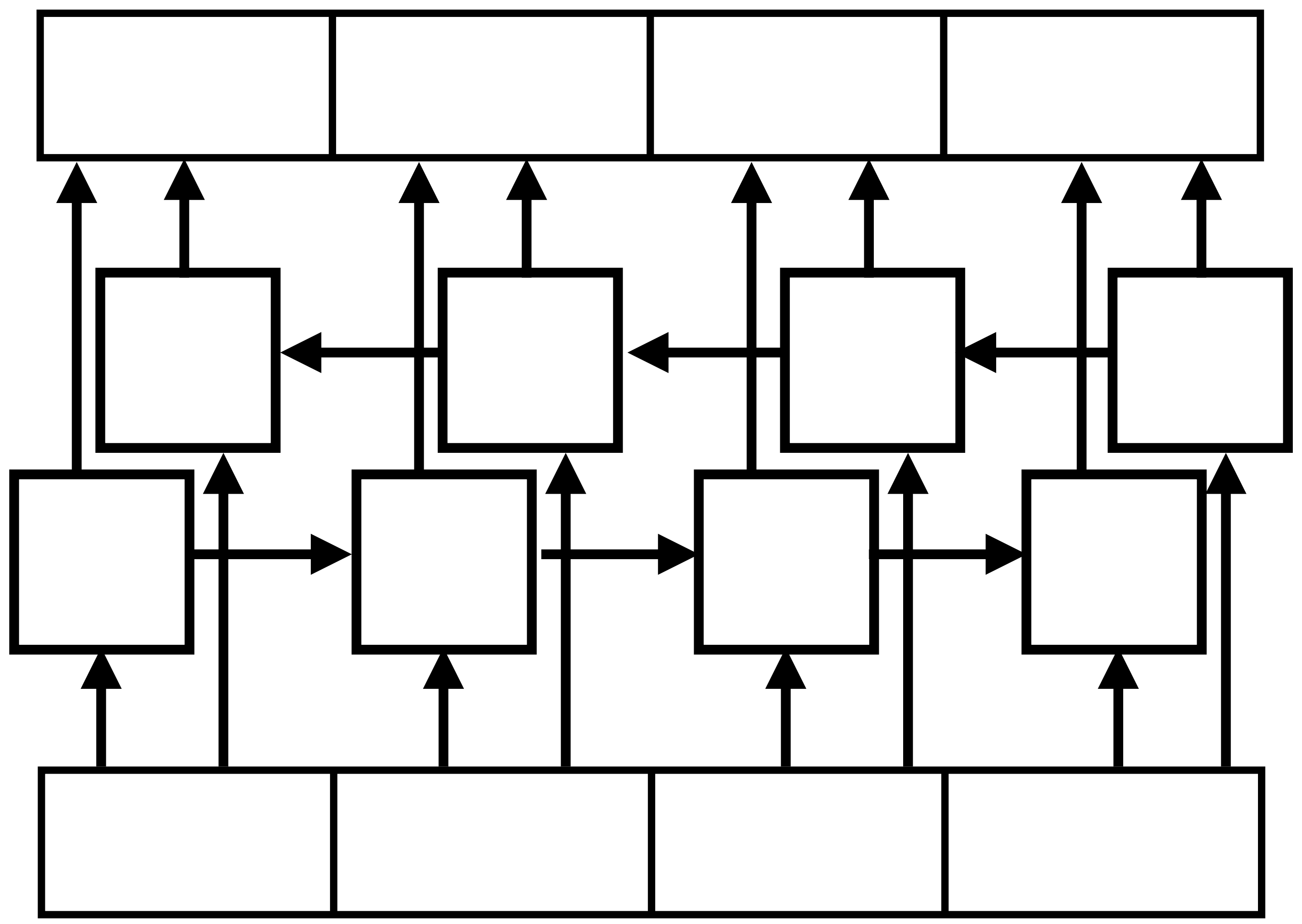


Figure 1: Extraction of convolutional vector space features $f_C(x, t_e)$. Three types of information from the input document and two types of information from the proposed title are fed through convolutional networks to produce vectors, which are systematically compared with cosine similarity to derive real-valued semantic similarity features.

Francis-Landau et al. (2016)

Compare: CNNs vs. LSTMs

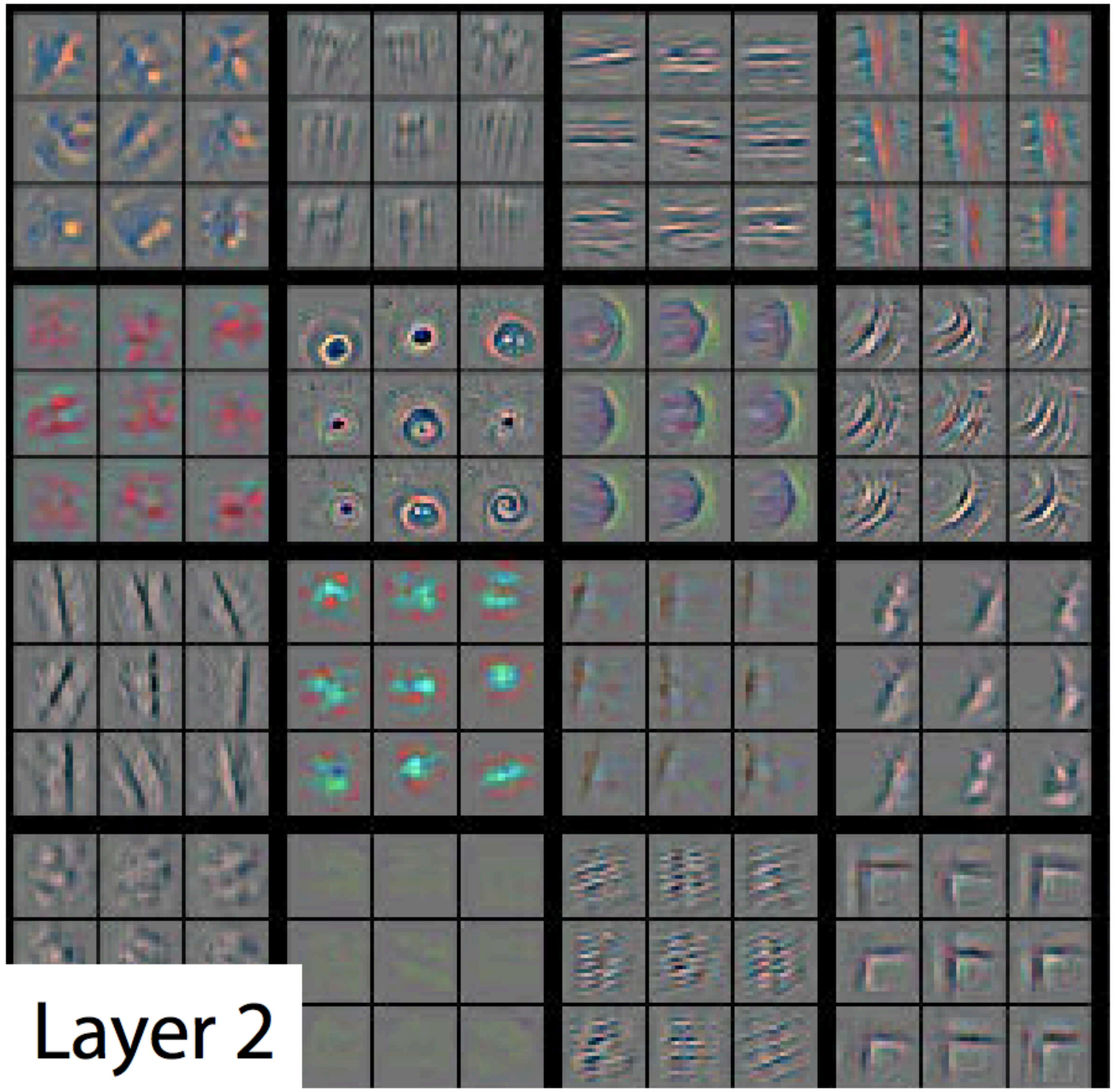


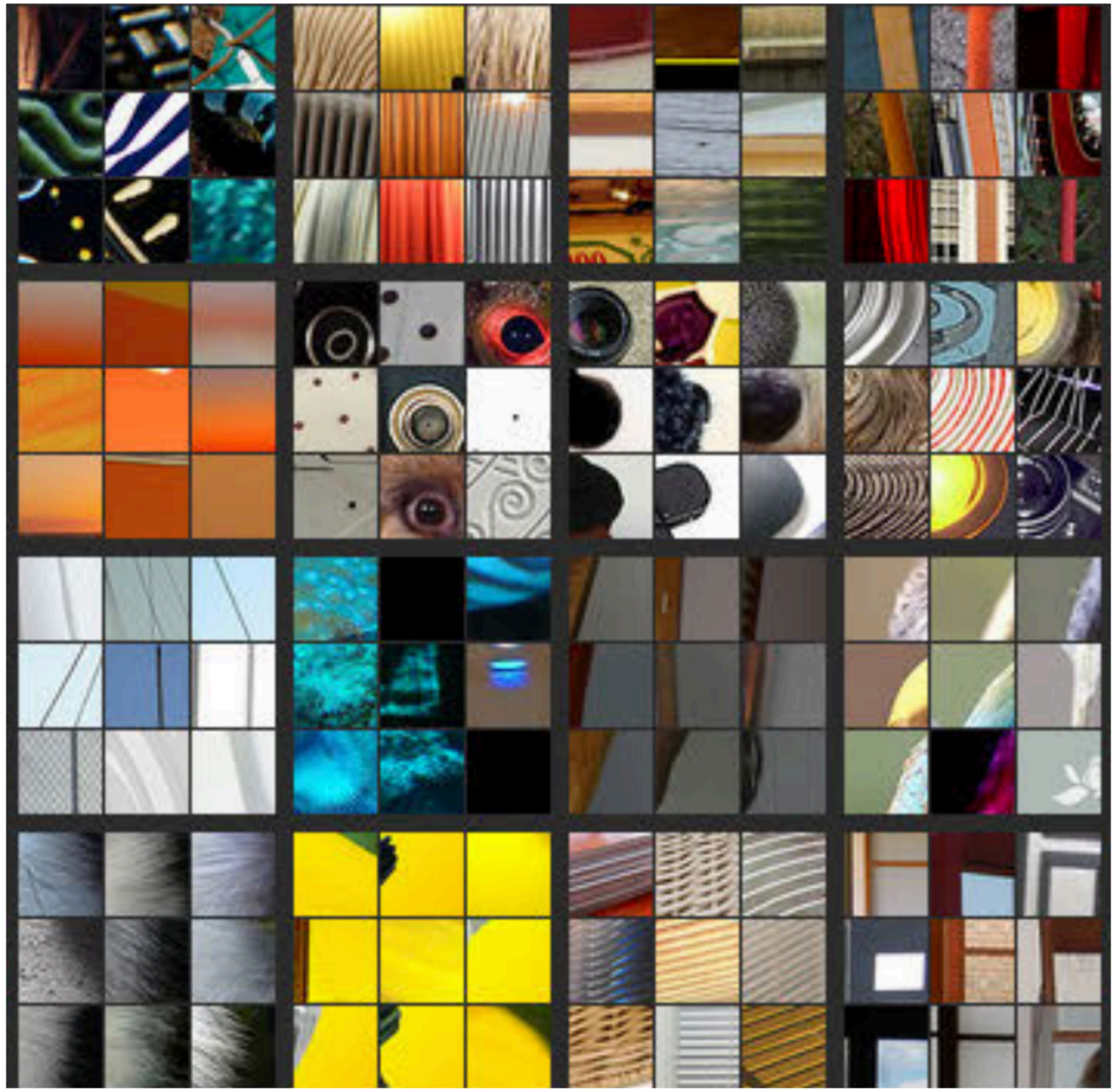


- Both LSTMs and convolutional layers transform the input using context
- LSTM: “globally” looks at the entire sentence (but local for many problems)
- CNN: local depending on filter width + number of layers

Deep Convolutional Networks

- Low-level filters: extract low-level features from the data

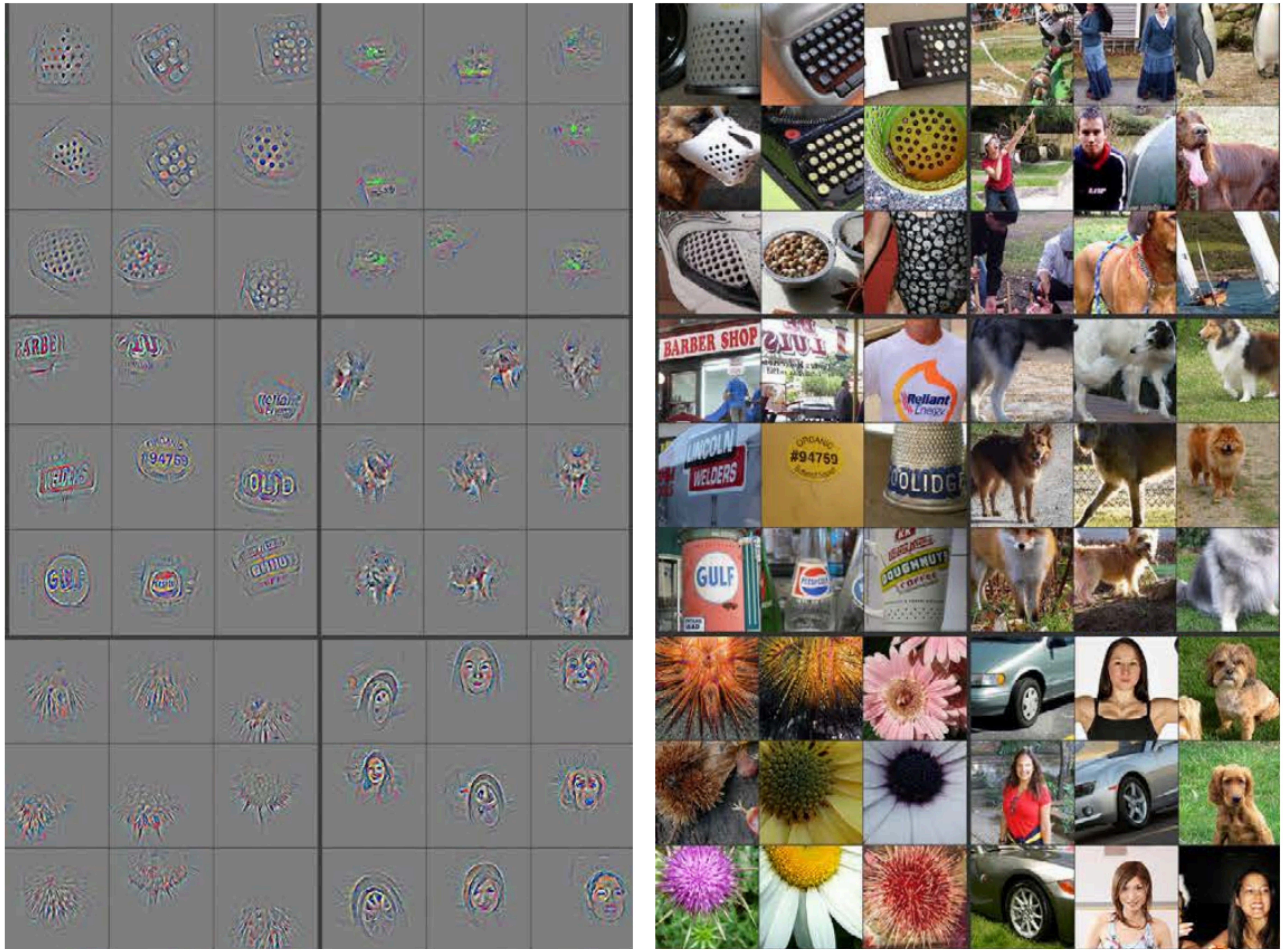




Zeiler and Fergus (2014)

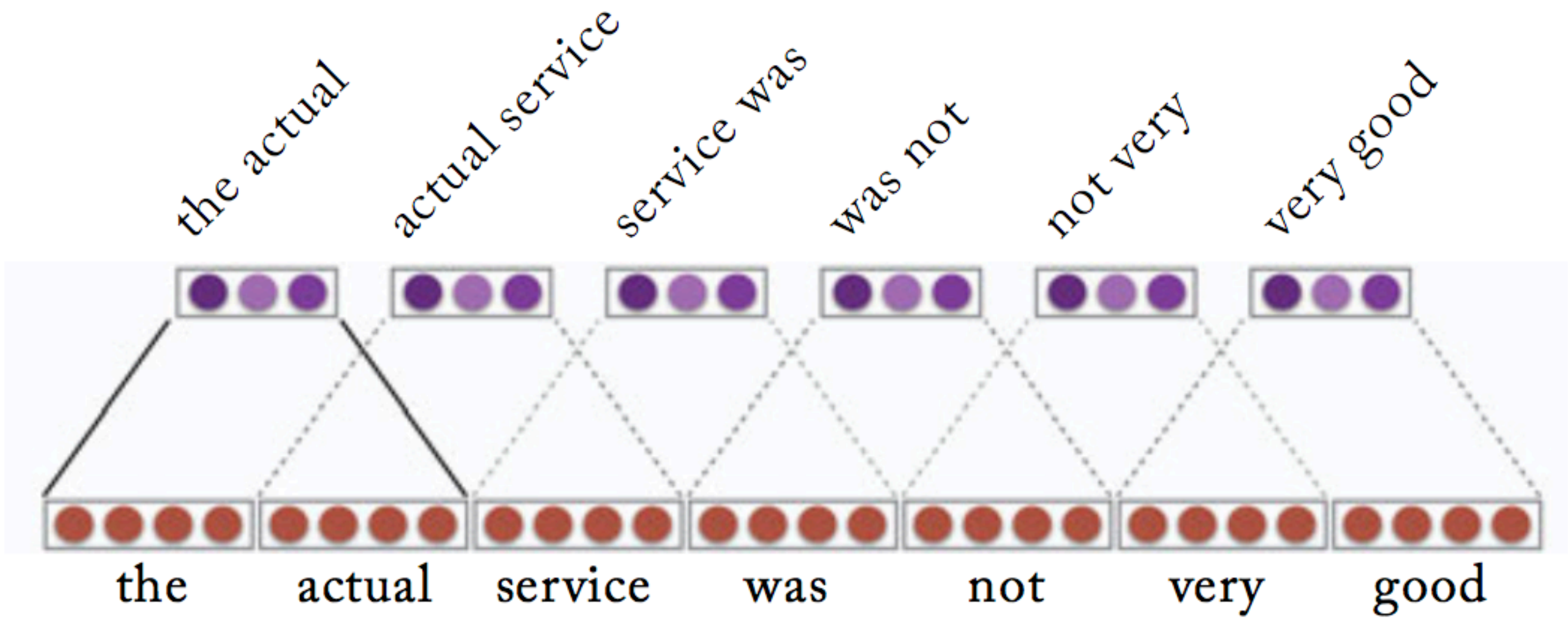
Deep Convolutional Networks

- High-level filters: match larger and more “semantic patterns”



Zeiler and Fergus (2014)

Multi-layer CNNs



Takeaways

- RNN — Sequential model that works well to present language. Attention mechanism is very effective, especially in seq2seq models.
- CNN — CNNs are a flexible way of extracting features analogous to bag of n-grams, can also encode positional information
- Transformer — No recurrent units (a computation bottleneck), effectively capturing context and long dependencies. Multi-heads analogous to different convolutional filters.