CS11-747 Neural Networks for NLP

Convolutional Networks for Text

Graham Neubig

Carnegie Mellon University
Language Technologies Institute

Site
https://phontron.com/class/nn4nlp2017/

An Example Prediction Problem: Sentence Classification

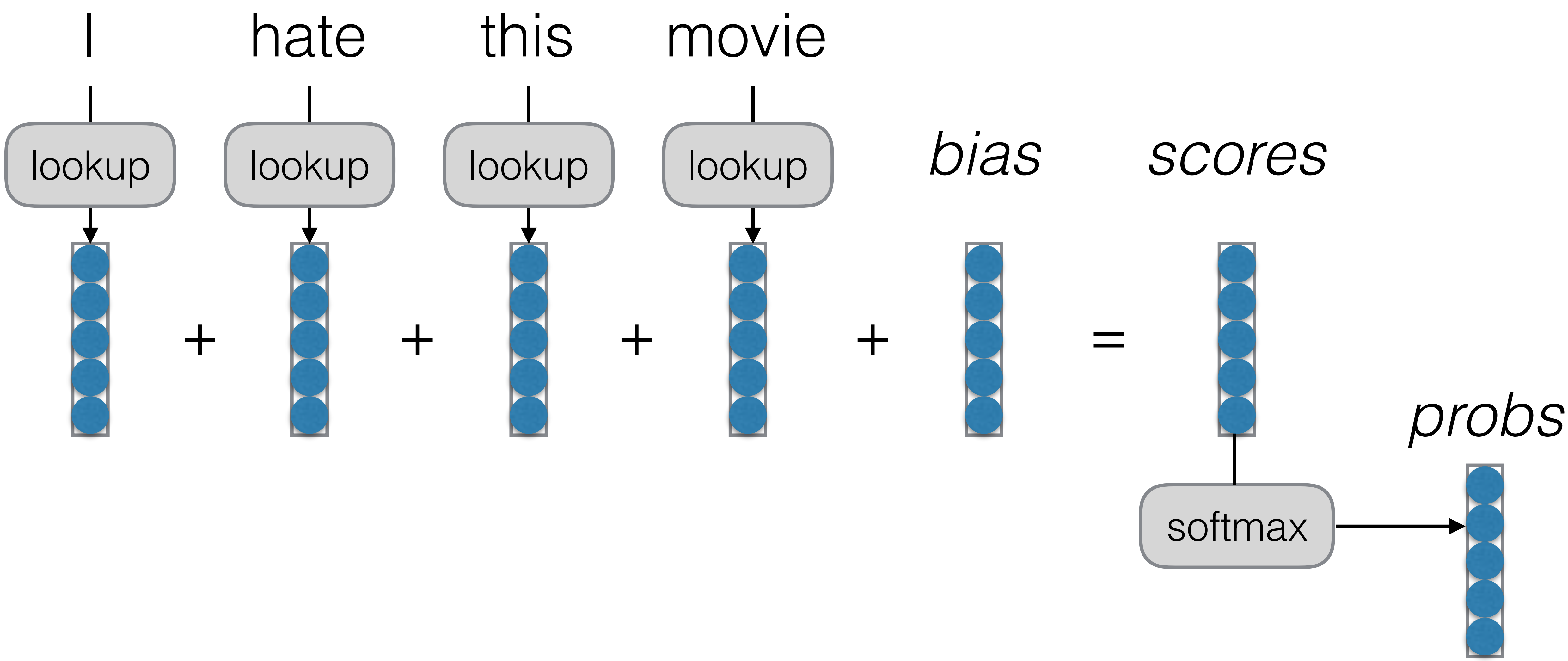
I hate this movie

very good
good
neutral
bad
very bad

I love this movie

very good
good
neutral
bad
very bad

A First Try: Bag of Words (BOW)



Build It, Break It

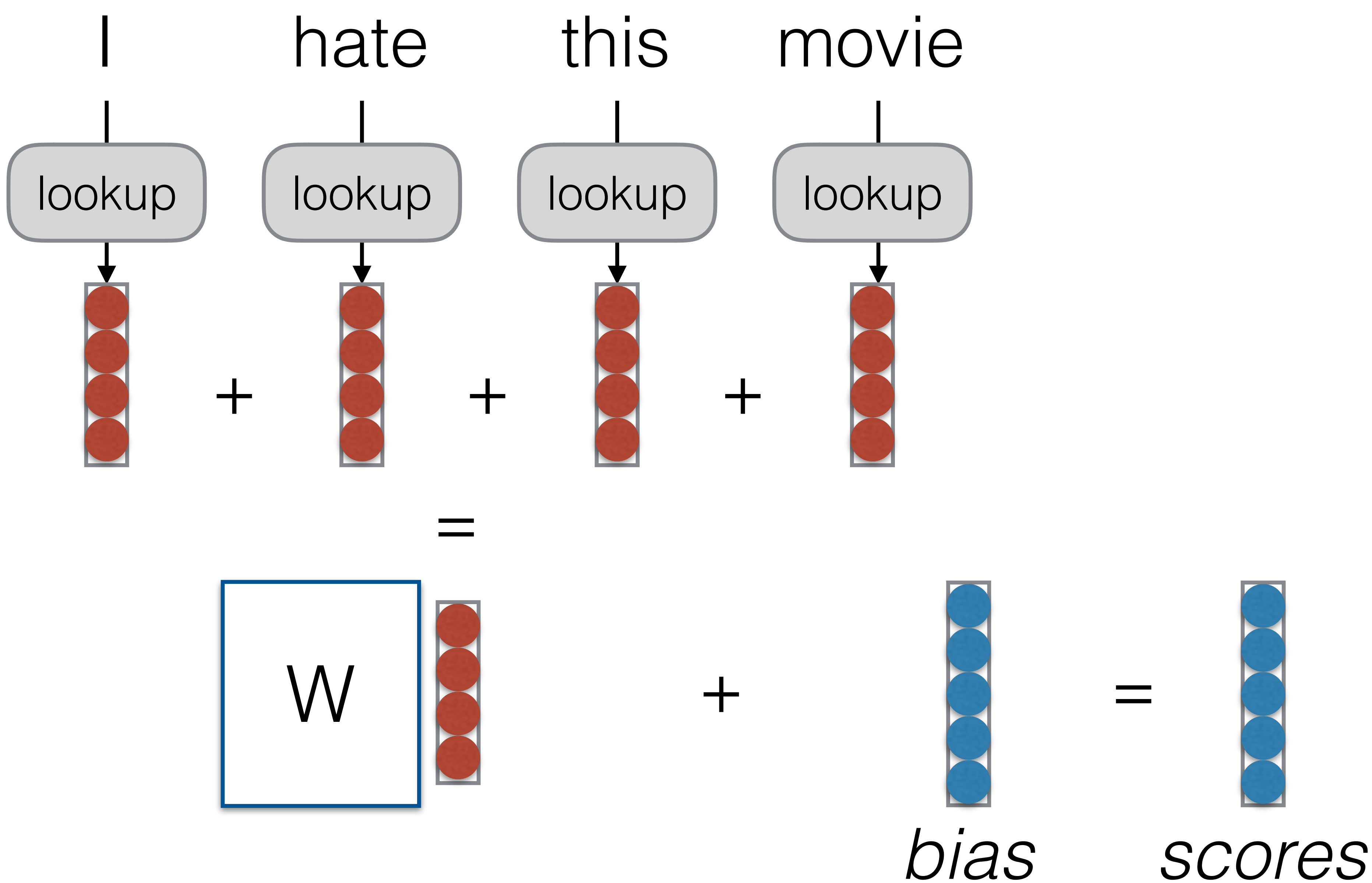
I don't love this movie

very good
good
neutral
bad
very bad

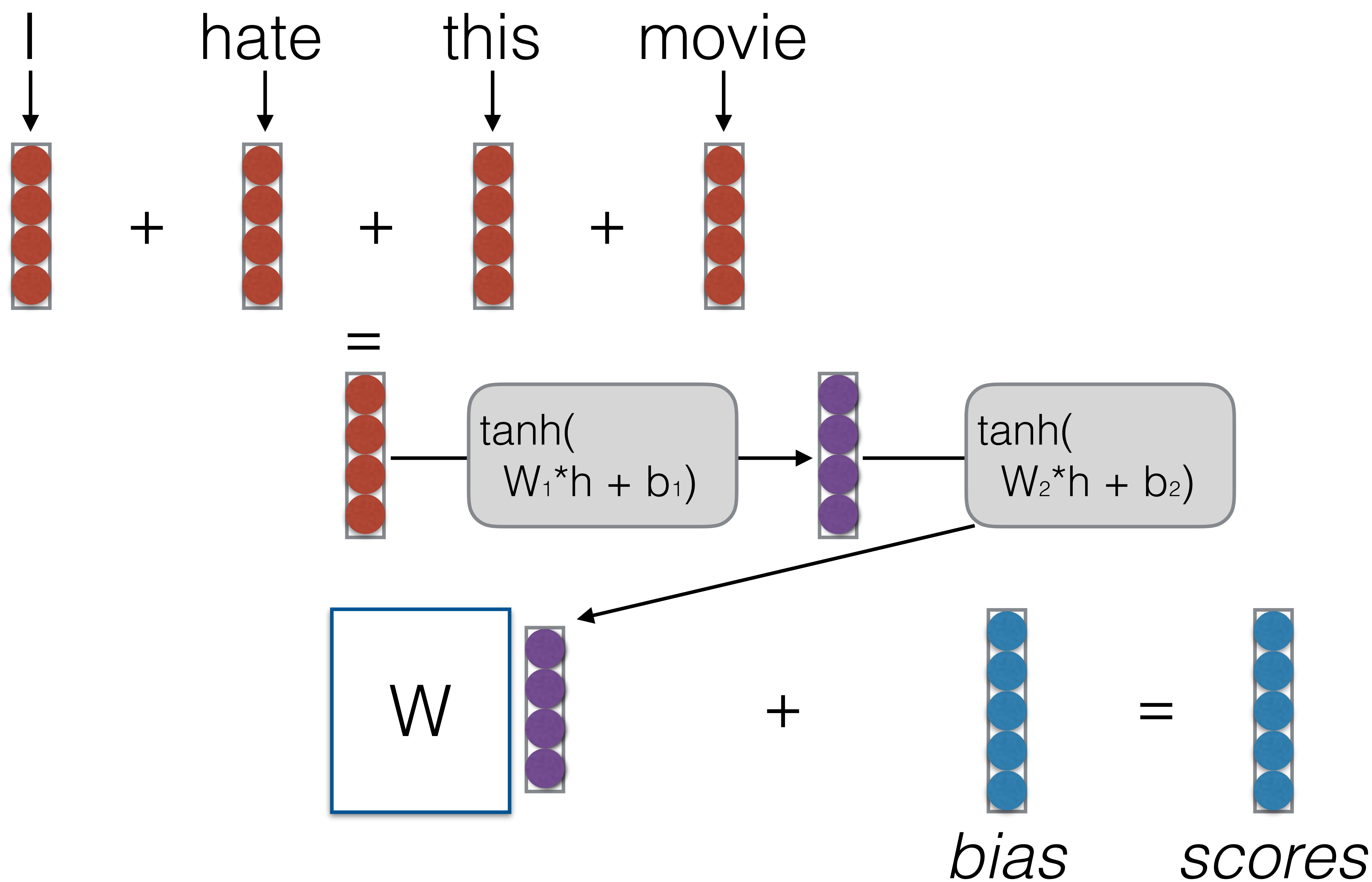
There's nothing I don't love about this movie

very good
good
neutral
bad
very bad

Continuous Bag of Words (CBOW)



Deep CBOW

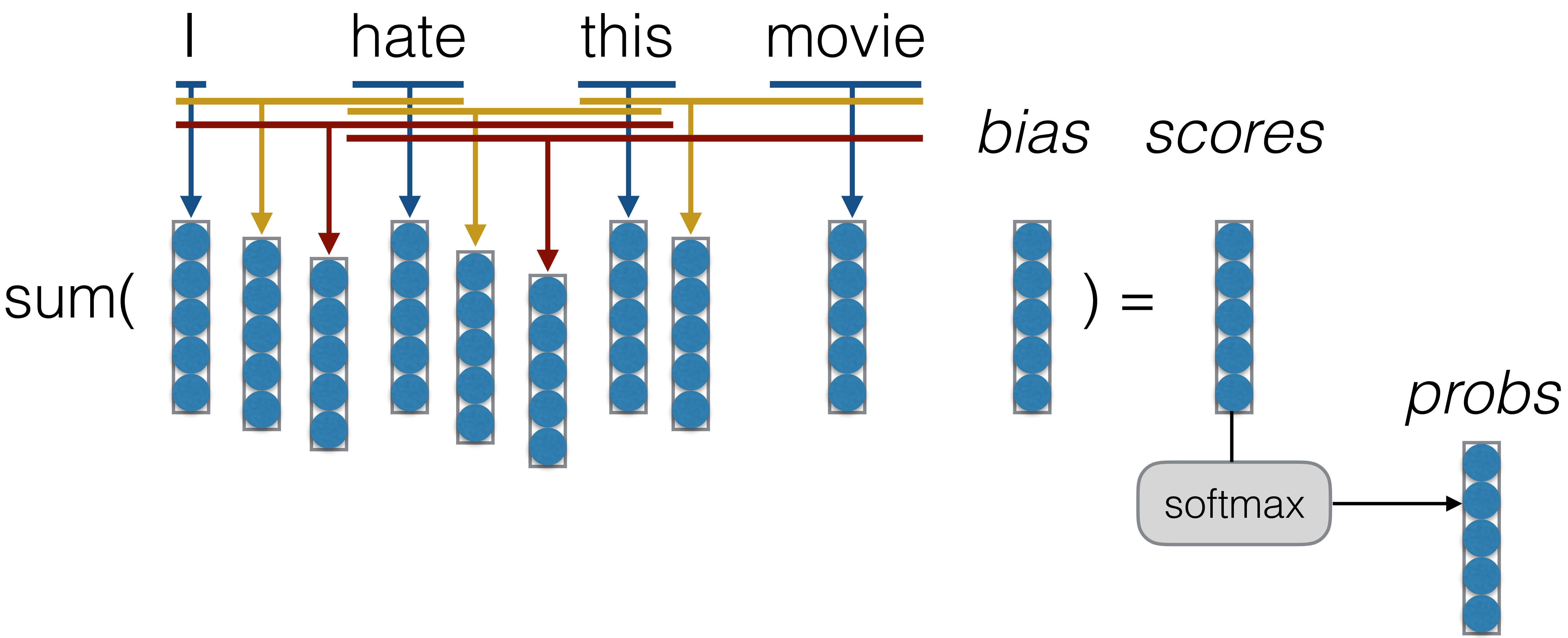


What do Our Vectors Represent?

- We can learn feature combinations (a node in the second layer might be “feature 1 AND feature 5 are active”)
- e.g. capture things such as “not” AND “hate”
- BUT! Cannot handle “not hate”

Handling Combinations

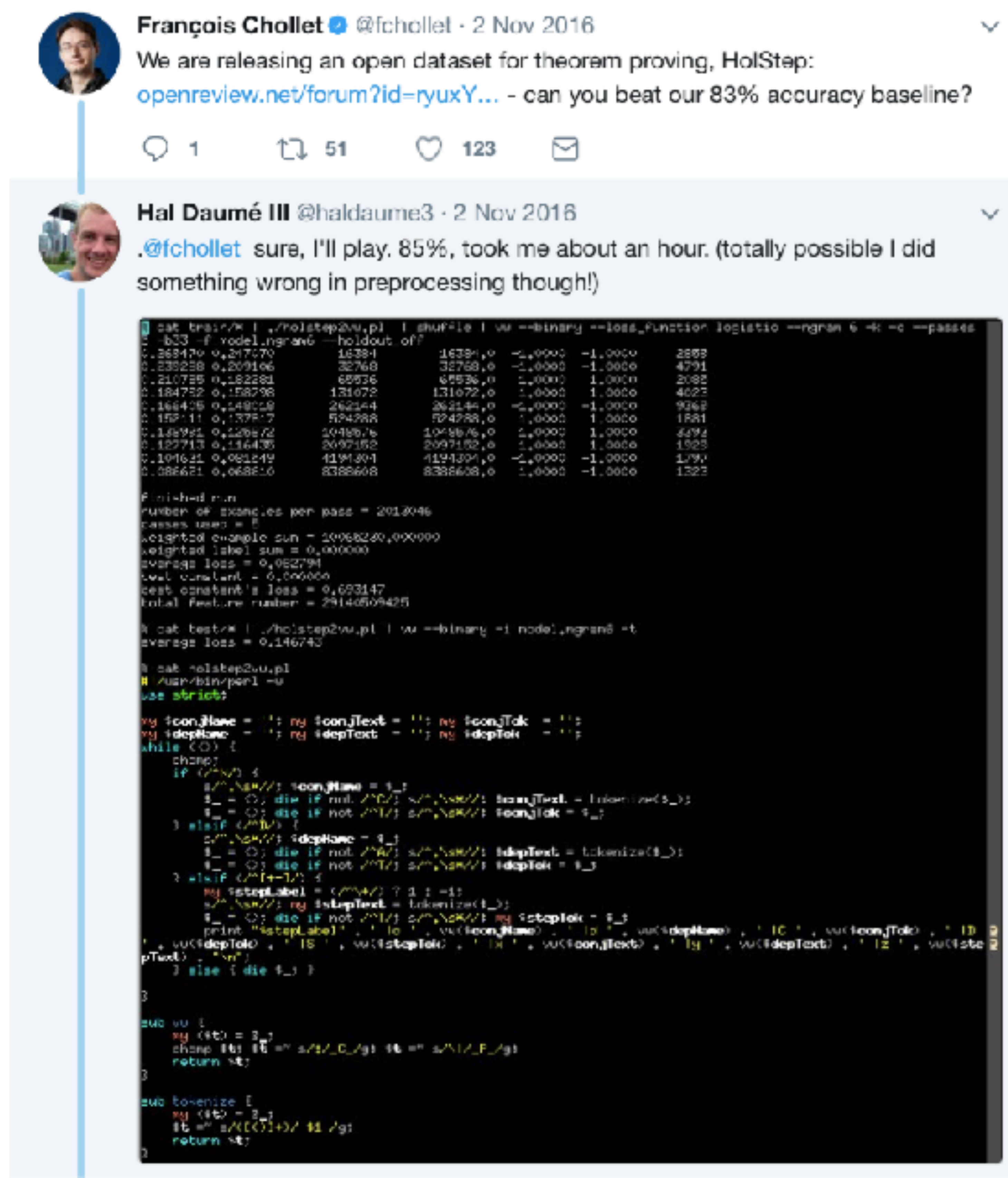
Bag of n-grams



Why Bag of n-grams?

- Allow us to capture combination features in a simple way “don’t love”, “not the best”

- Works pretty well

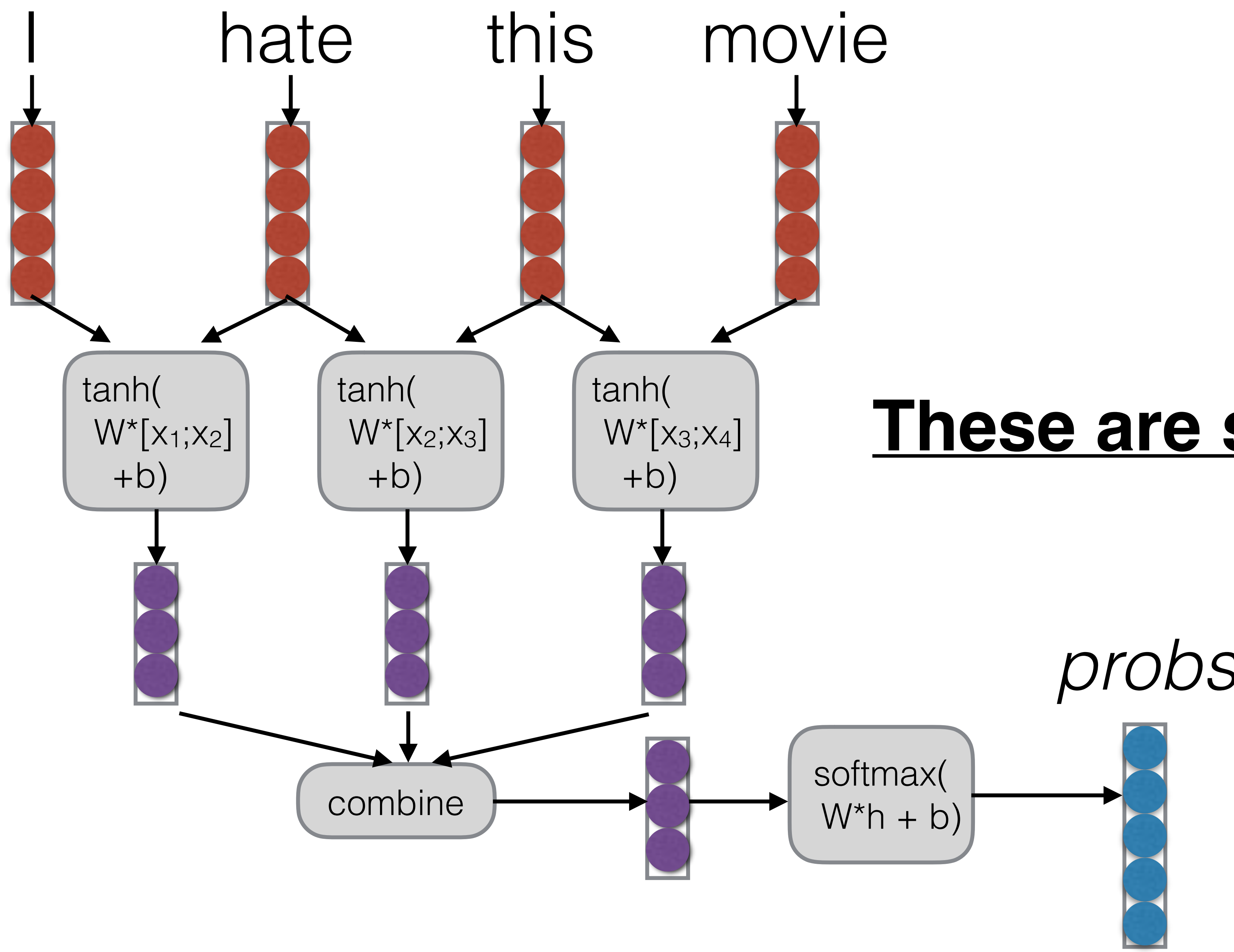


What Problems w/ Bag of n-grams?

- Same as before: parameter explosion
- No sharing between similar words/n-grams

Time Delay/
Convolutional Neural Networks

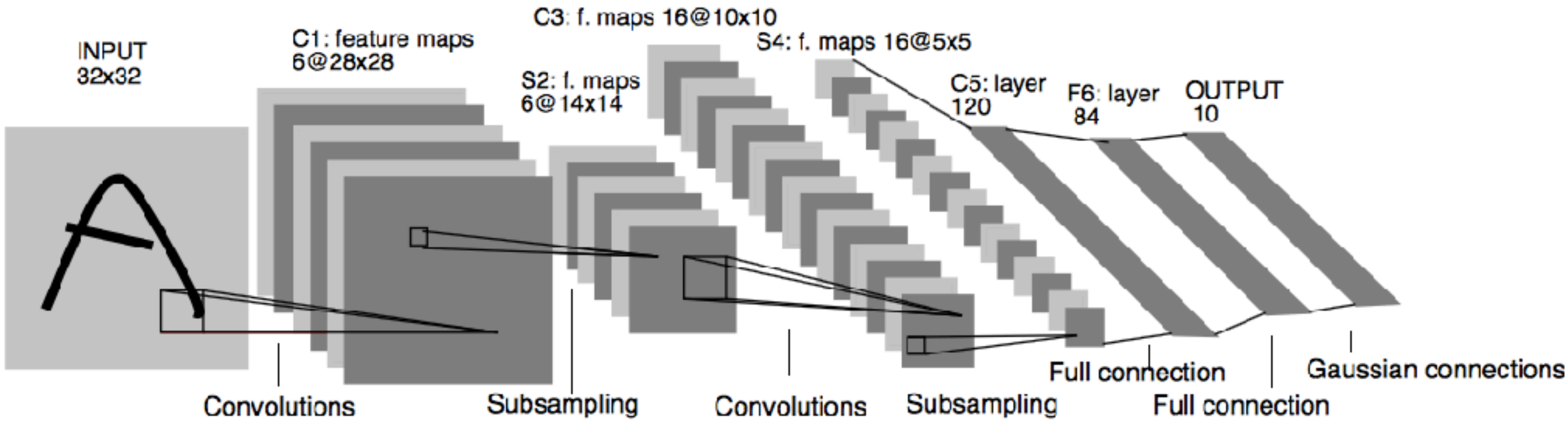
Time Delay Neural Networks
(Waibel et al. 1989)



These are soft 2-grams!

Convolutional Networks

(LeCun et al. 1997)

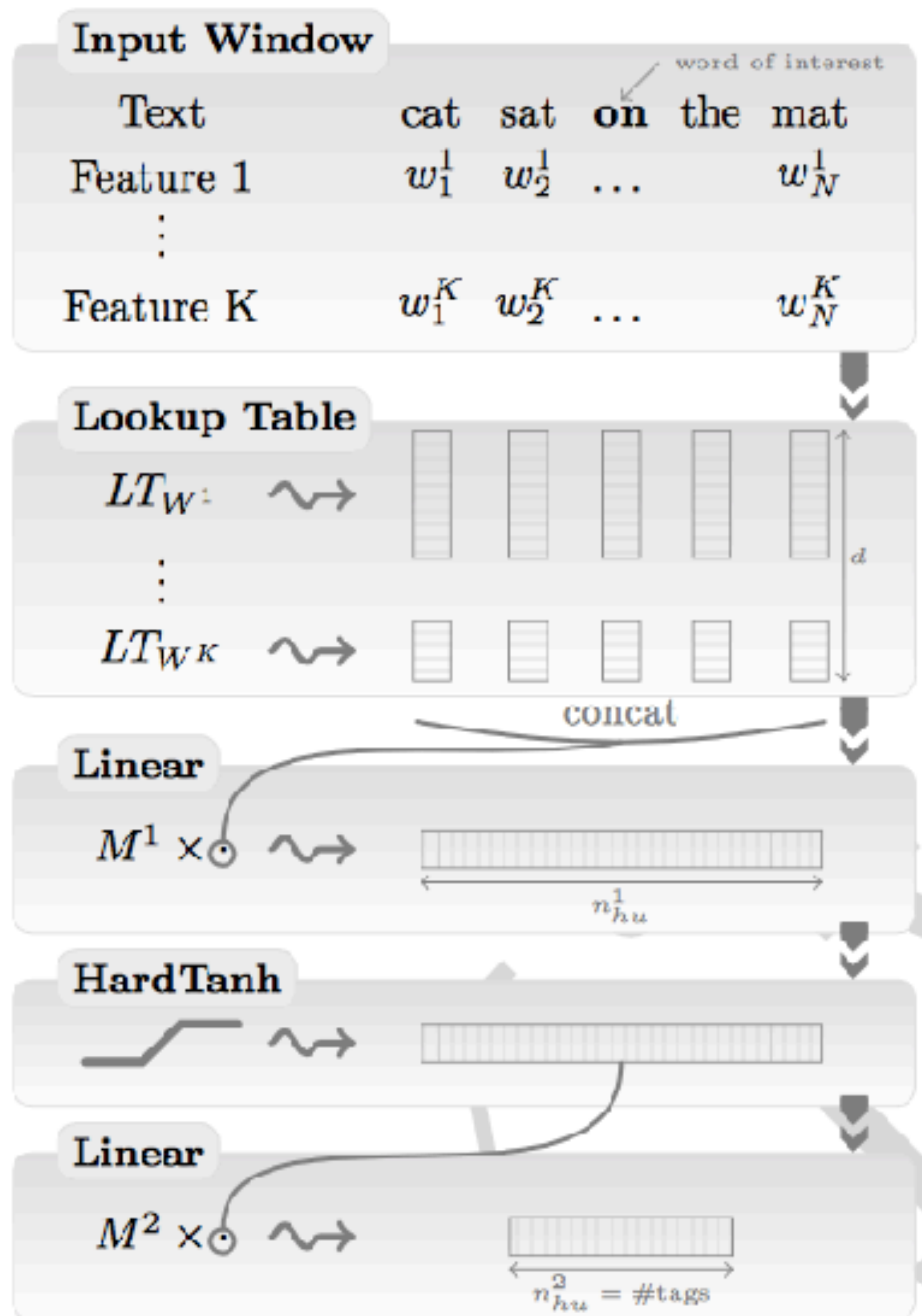


Parameter extraction performs a 2D sweep, not 1D

CNNs for Text
(Collobert and Weston 2011)

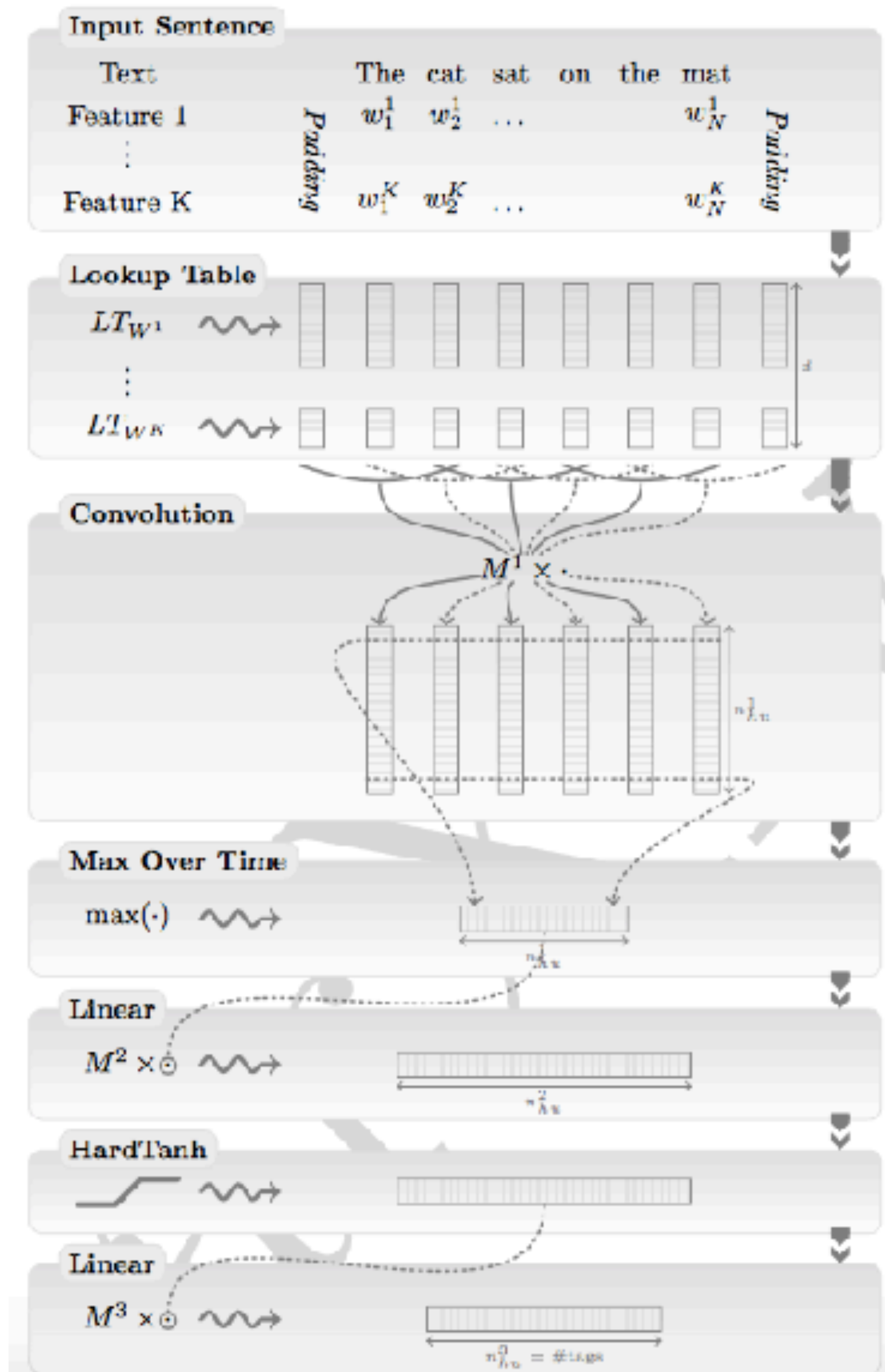
- 1D convolution ≈ Time Delay Neural Network
  - But often uses terminology/functions borrowed from image processing
- Two main paradigms:
  - **Context window modeling:** For tagging, etc. get the surrounding context before tagging
  - **Sentence modeling:** Do convolution to extract n-grams, pooling to combine over whole sentence

CNNs for Tagging
(Collobert and Weston 2011)



CNNs for Sentence Modeling

(Collobert and Weston 2011)



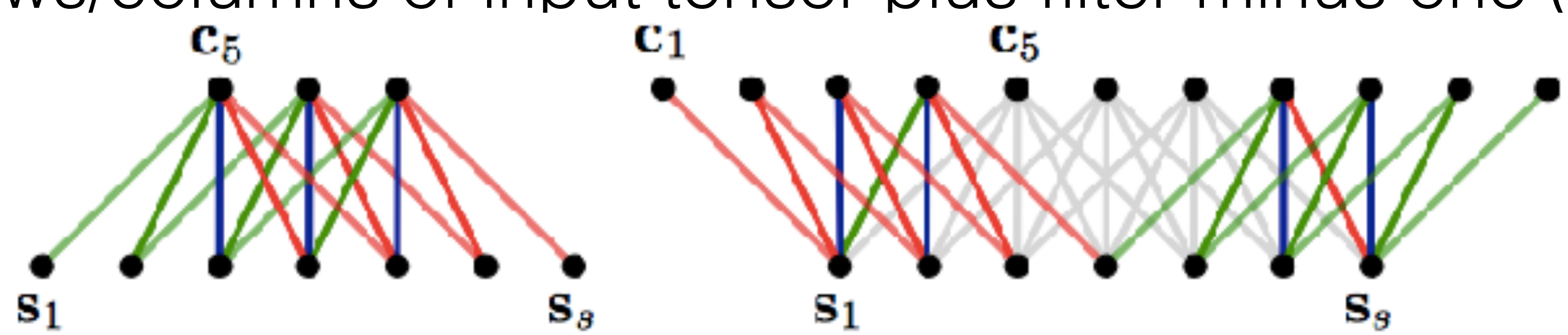
Standard conv2d Function

- 2D convolution function takes input + parameters
- **Input:** 3D tensor
  - rows (e.g. words), columns, features (“channels”)
- **Parameters/Filters:** 4D tensor
  - rows, columns, input features, output features

Padding/Striding

- **Padding:** After convolution, the rows and columns of the output tensor are either
  - = to rows/columns of input tensor (“same” convolution)
  - = to rows/columns of input tensor minus the size of the filter plus one (“valid” or “narrow”)
  - = to rows/columns of input tensor plus filter minus one (“wide”)

Narrow →



← Wide

- **Striding:** It is also common to skip rows or columns (e.g. a stride of [2,2] means use every other)

Image: Kalchbrenner et al. 2014

Pooling

- Pooling is like convolution, but calculates some reduction function feature-wise

- **Max pooling:** “Did you see this feature anywhere in the range?” (most common)

- **Average pooling:** “How prevalent is this feature over the entire range”

- **k-Max pooling:** “Did you see this feature up to k times?”

- **Dynamic pooling:** “Did you see this feature in the beginning? In the middle? In the end?”

Let's Try It!

`cnn-class.py`

Stacked Convolution

Stacked Convolution

- Feeding in convolution from previous layer results in larger area of focus for each feature

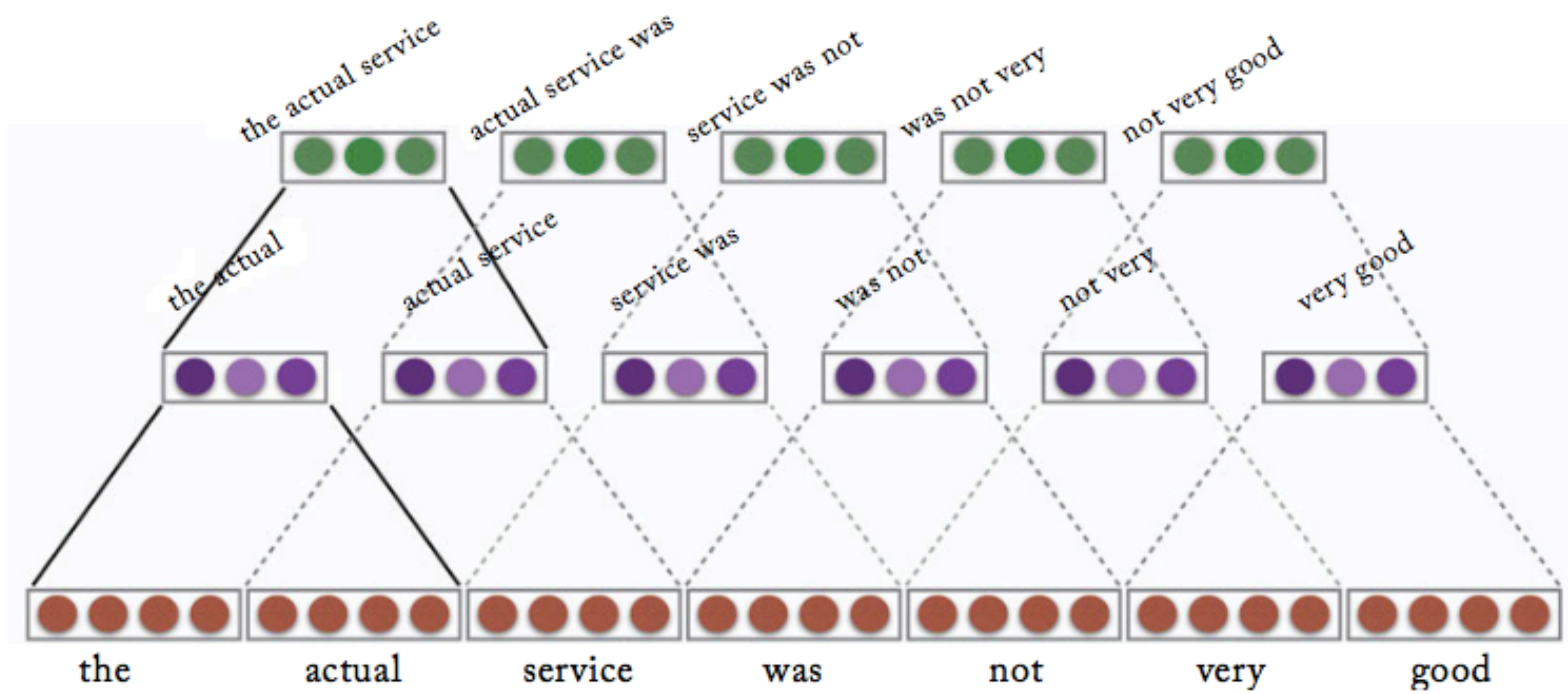
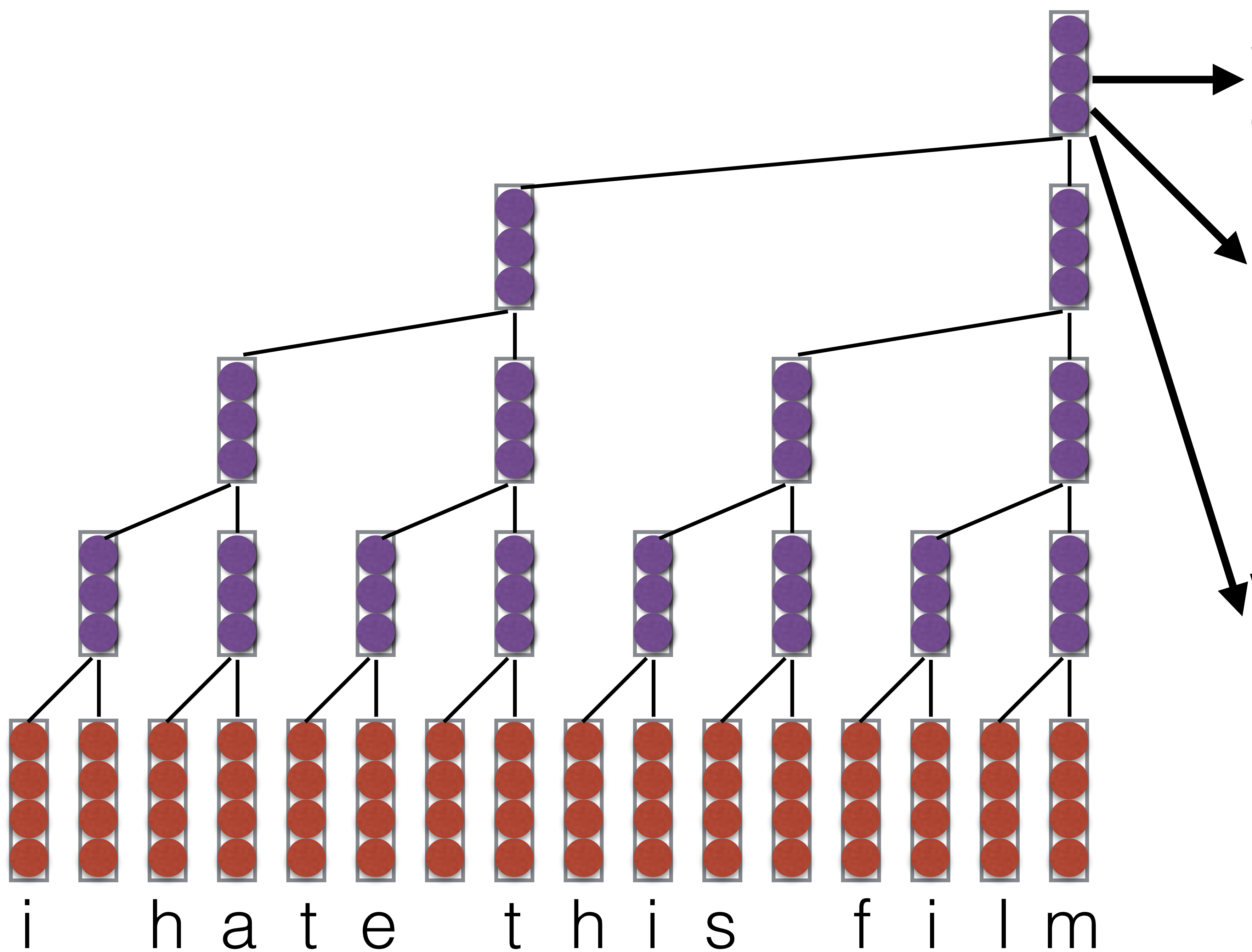


Image Credit: Goldberg Book

Dilated Convolution
(e.g. Kalchbrenner et al. 2016)

- **Gradually increase stride:** low-level to high-level



sentence class (classification)

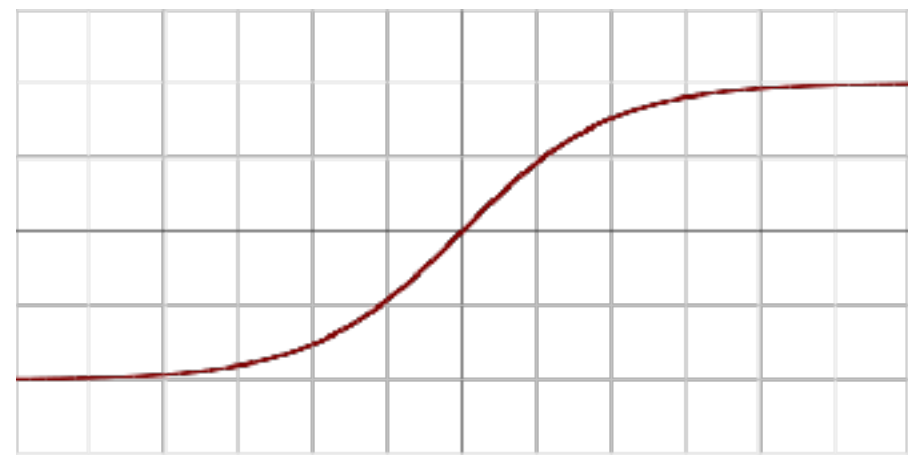
next char (language modeling)

word class (tagging)

An Aside: Nonlinear Functions

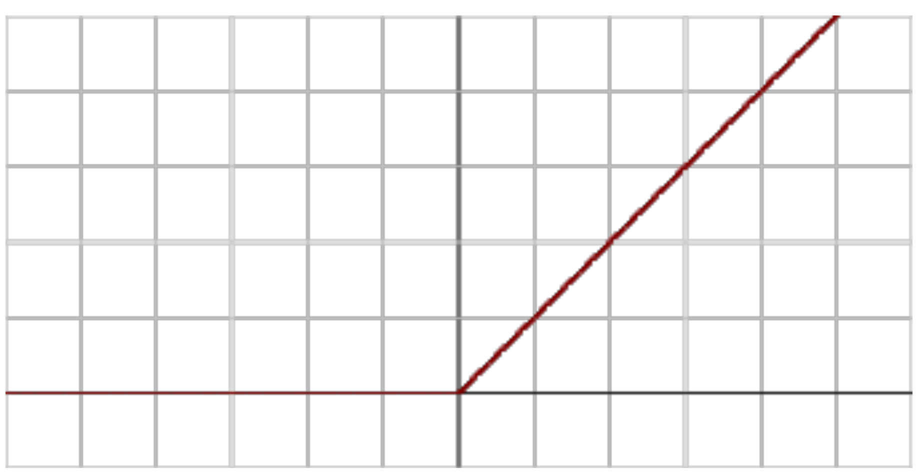
- Proper choice of a non-linear function is essential in stacked networks

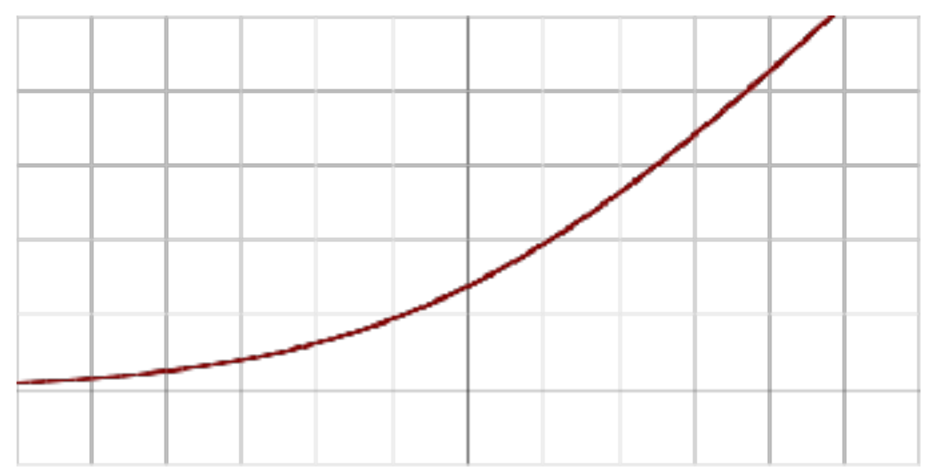
step



tanh

rectifier (RelU)





soft plus

- Functions such as RelU or softplus often work better at preserving gradients

Image: Wikipedia

Why (Dilated) Convolution for Modeling Sentences?

- In contrast to recurrent neural networks (next class)
- **+** Fewer steps from each word to the final representation: RNN O(N), Dilated CNN O(log N)
- **+** Easier to parallelize on GPU
- **-** Slightly less natural for arbitrary-length dependencies
- **-** A bit slower on CPU?

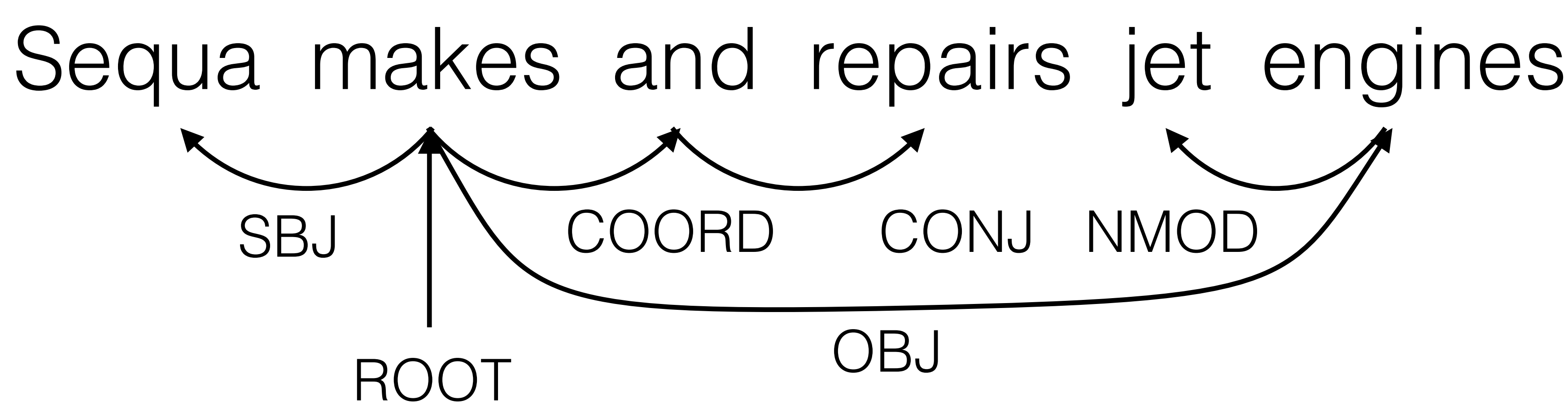
Structured Convolution

Why Structured Convolution?

- Language has structure, would like it to localize features

- e.g. noun-verb pairs very informative, but not captured by normal CNNs

Example: Dependency Structure



Example From: Marcheggiani and Titov 2017

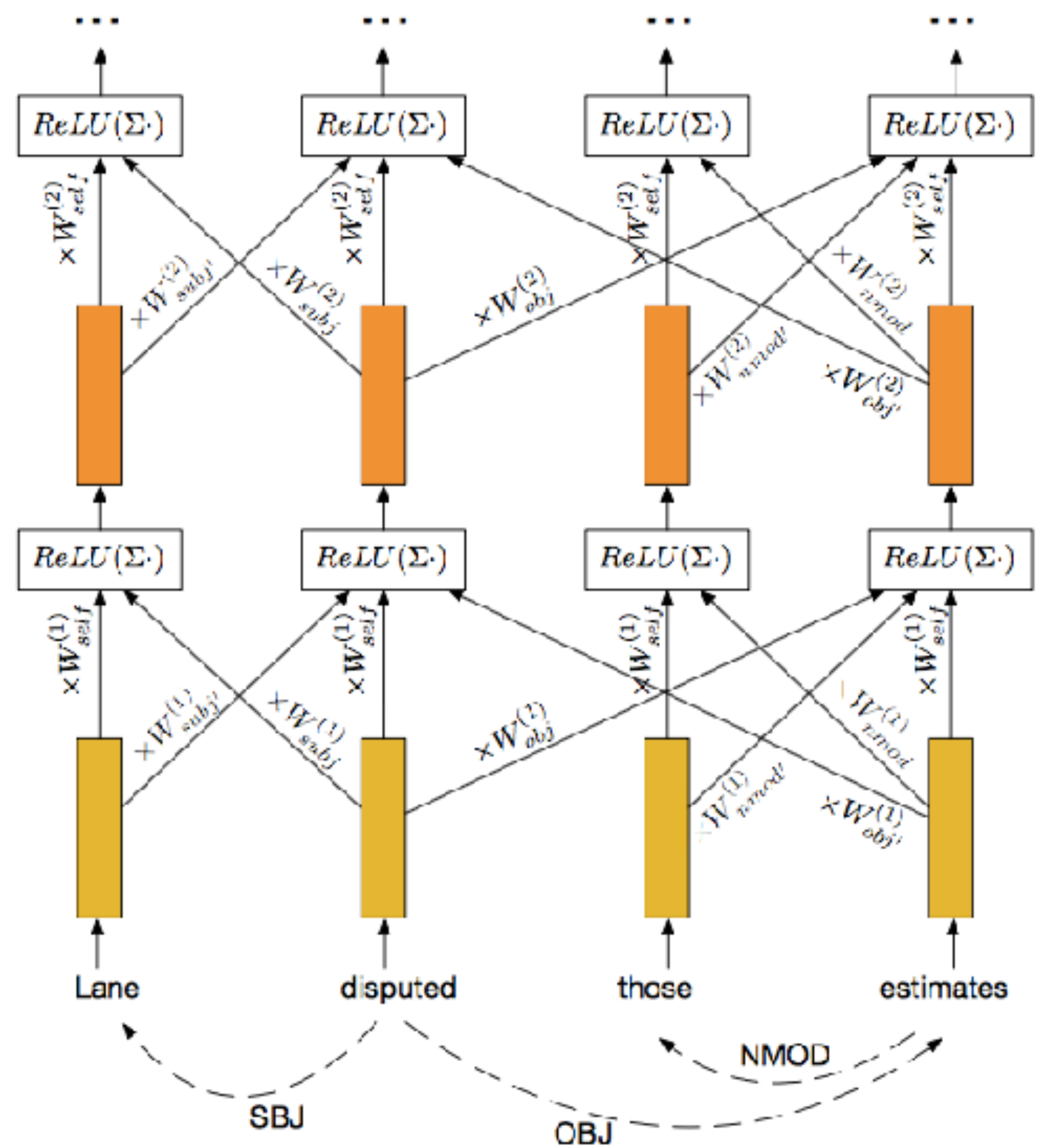
Tree-structured Convolution
(Ma et al. 2015)

- Convolve over parents, grandparents, siblings

ancestor paths		siblings	
n	pattern(s)	n	pattern(s)
3	$m \quad h \quad g$	2	$s \quad m \quad \_$
4	$m \quad h \quad g \quad g^2$	3	$s \quad m \quad h$ $t \quad s \quad m \quad \_$
5	$m \quad h \quad g \quad g^2 \quad g^3$	4	$t \quad s \quad m \quad h$ $s \quad m \quad h \quad g$

Graph Convolution
(e.g. Marcheggiani et al. 2017)

- Convolution is shaped by graph structure
- For example, dependency tree is a graph with
  - Self-loop connections
  - Dependency connections
  - Reverse connections



Convolutional Models of Sentence Pairs

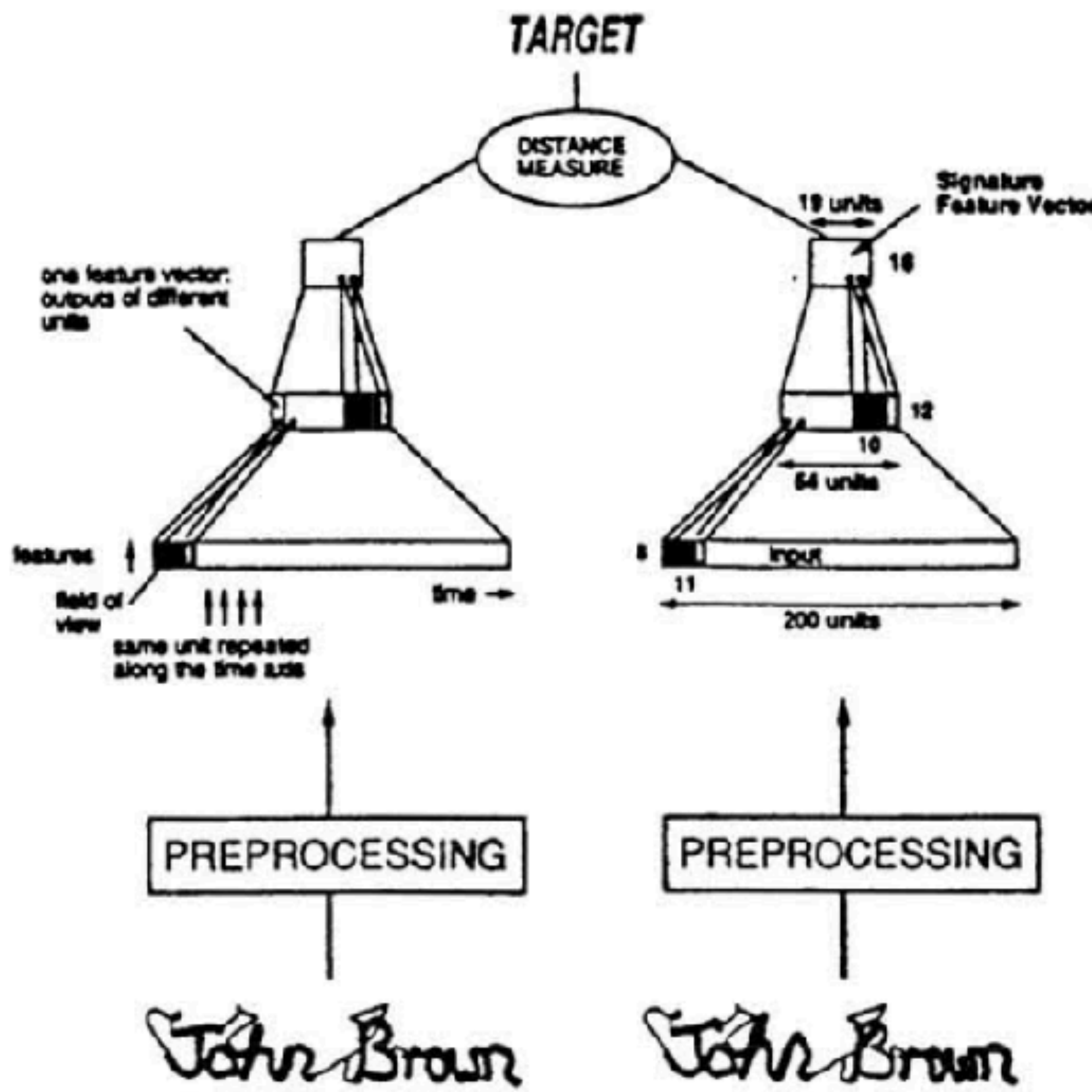
Why Model Sentence Pairs?

- Paraphrase identification / sentence similarity
- Textual entailment
- Retrieval
- (More about these specific applications in two classes)

Siamese Network
(Bromley et al. 1993)

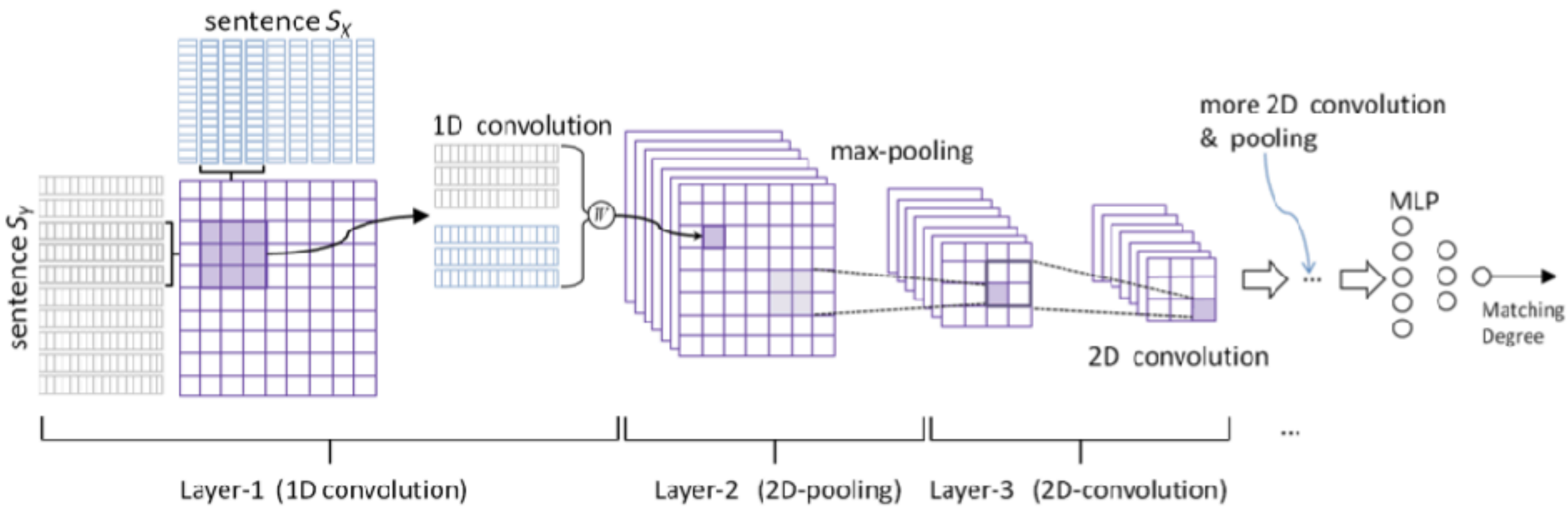
- Use the same network, compare the extracted representations

- (e.g. Time-delay networks for signature recognition)



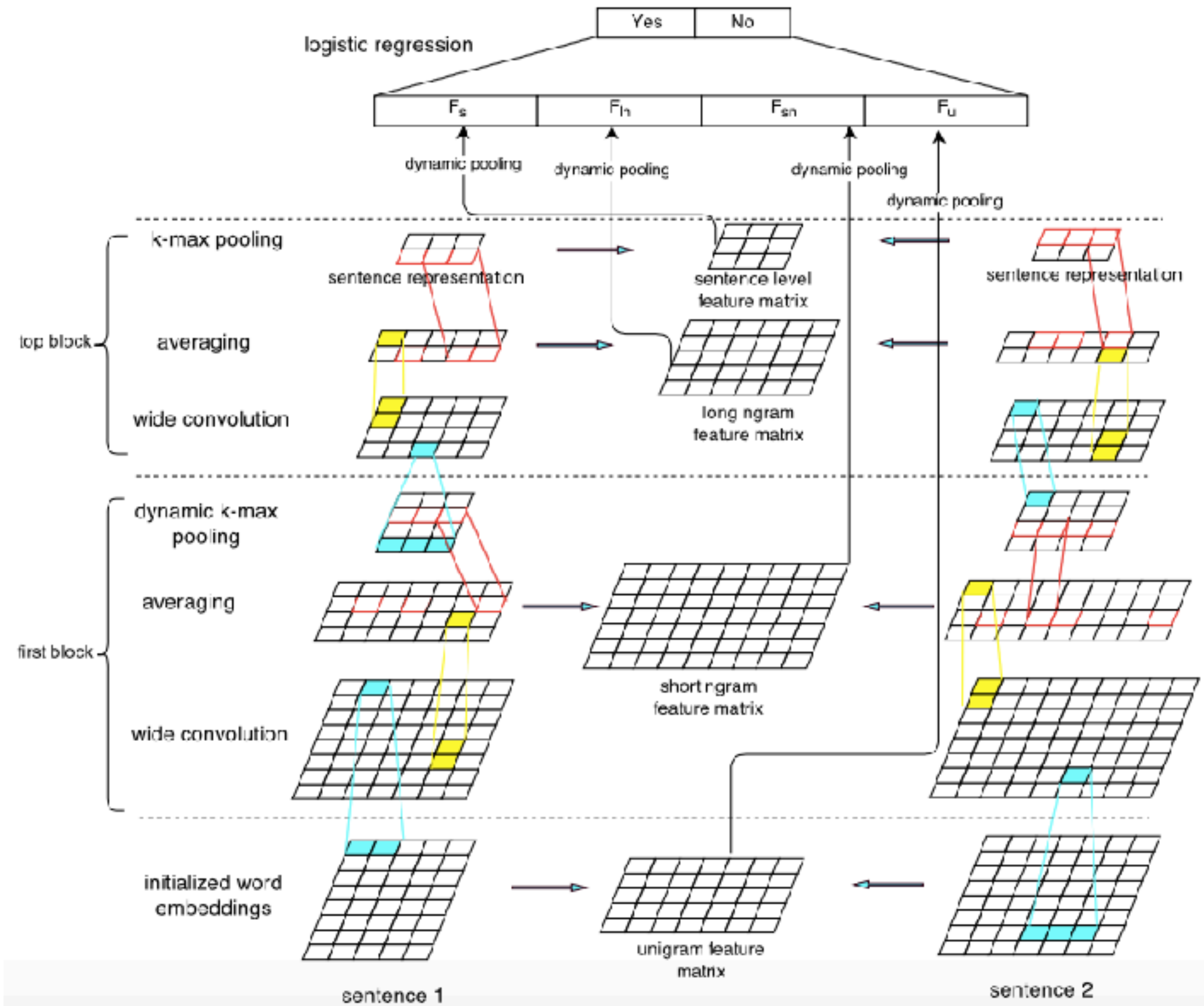
Convolutional Matching Model (Hu et al. 2014)

- Concatenate sentences into a 3D tensor and perform convolution



- Shown more effective than simple Siamese network

Convolutional Features + Matrix-based Pooling (Yin and Schutze 2015)



Understanding CNN Results

Why Understanding?

- Sometimes we want to know why model is making predictions (e.g. is there bias?)
- Understanding extracted features might lead to new architectural ideas
- Visualization of filters, etc. easy in vision but harder in NLP; other techniques can be used

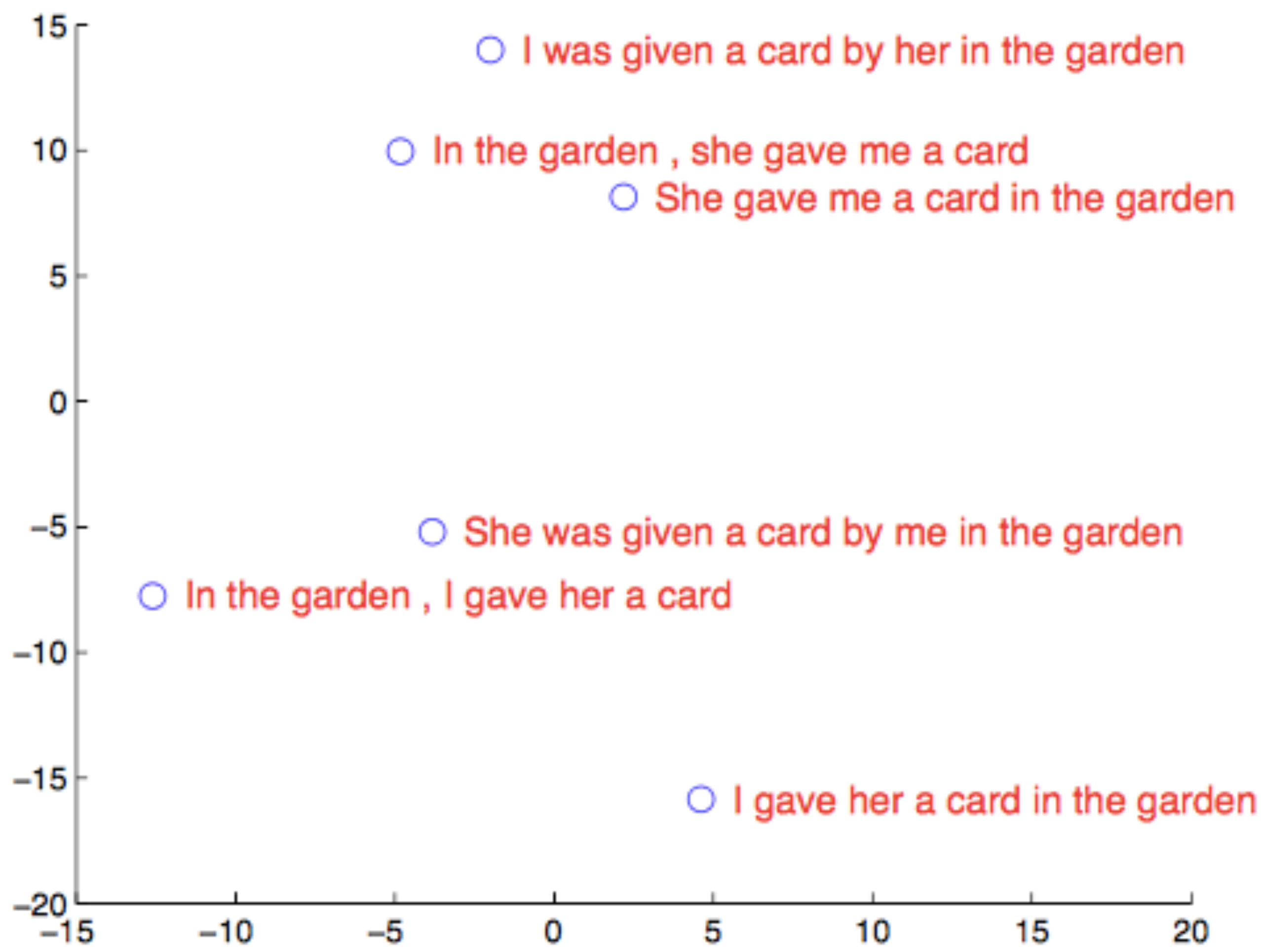
Maximum Activation

- Calculate the hidden feature values for whole data, find section of image/sentence that results in max value

Example: Karpathy 2016

PCA/t-SNE Embedding of Feature Vector

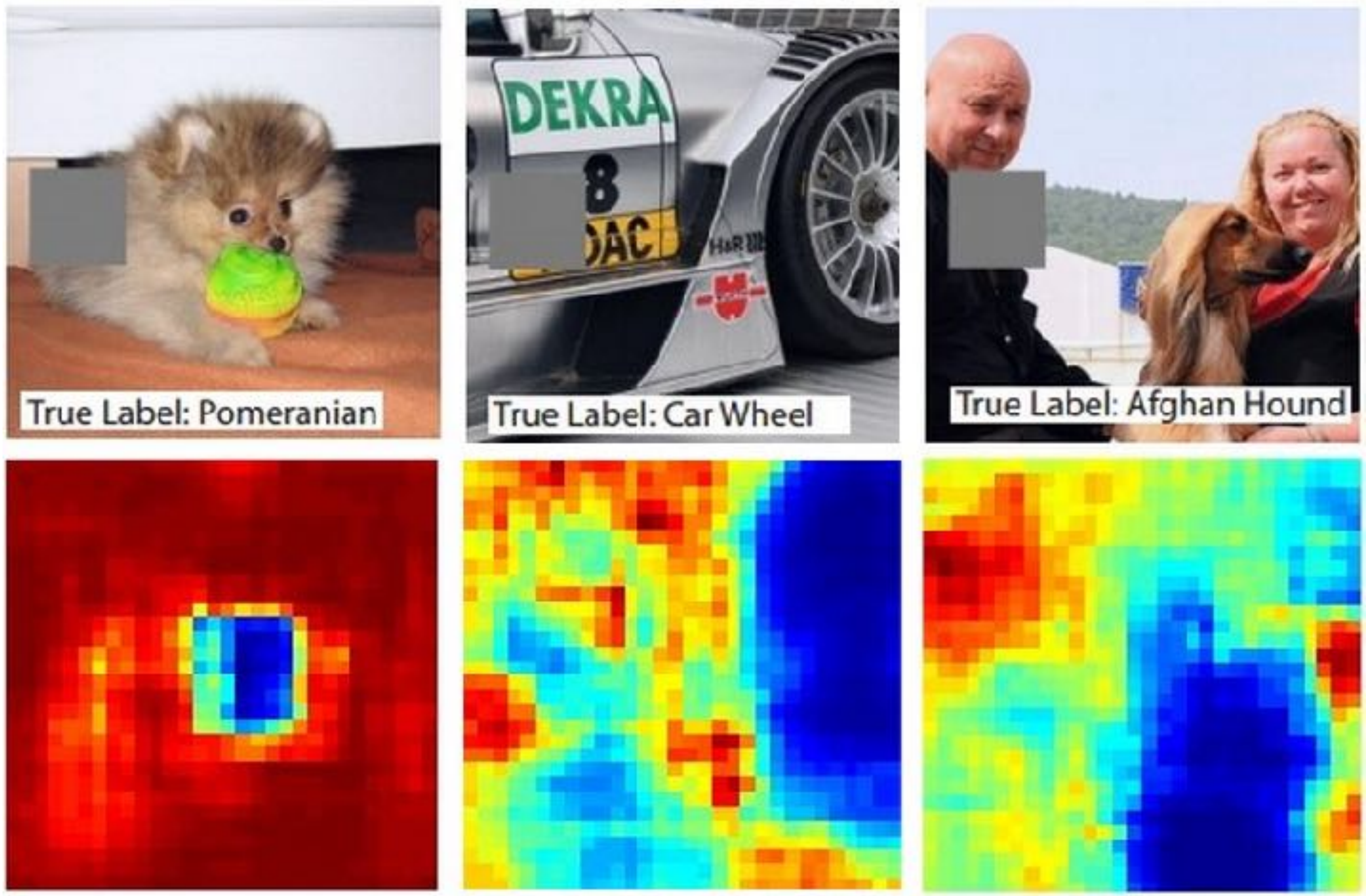
- Do dimension reduction on feature vectors



Example: Sutskever+ 2014

Occlusion

- Blank out one part at a time (in NLP, word?), and measure the difference from the final representation/prediction



Example: Karpathy 2016

Let’s Try It!

`cnn-activation.py`

Questions?