

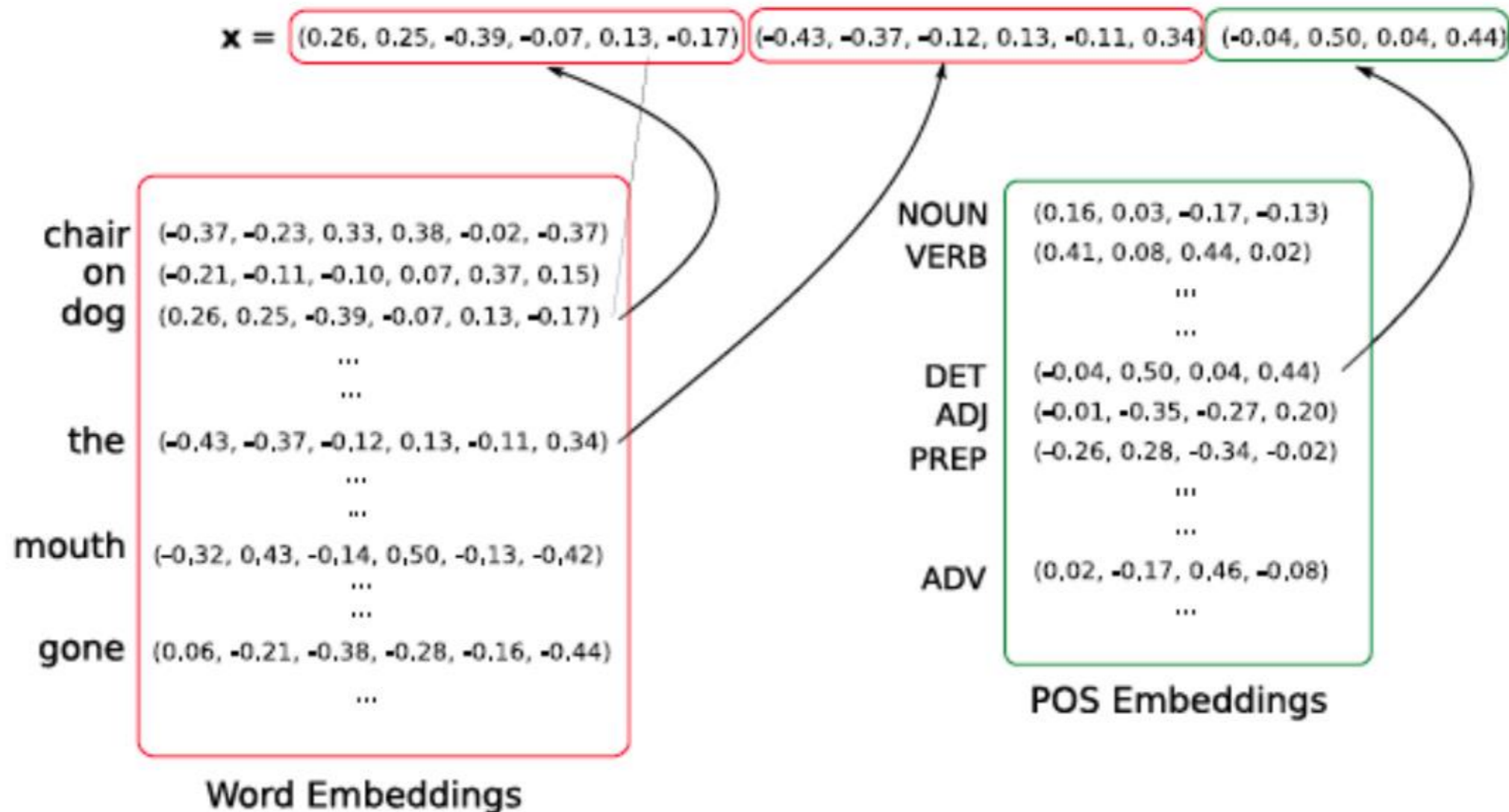
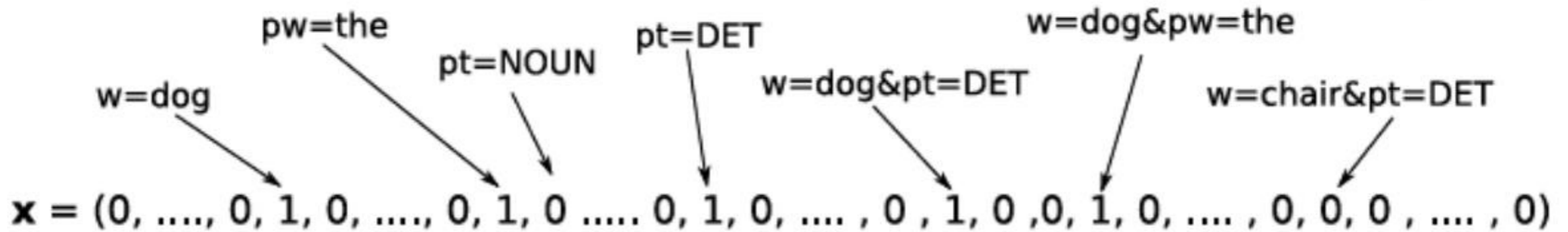
N-gram features Convolutional Networks

Yoav Goldberg

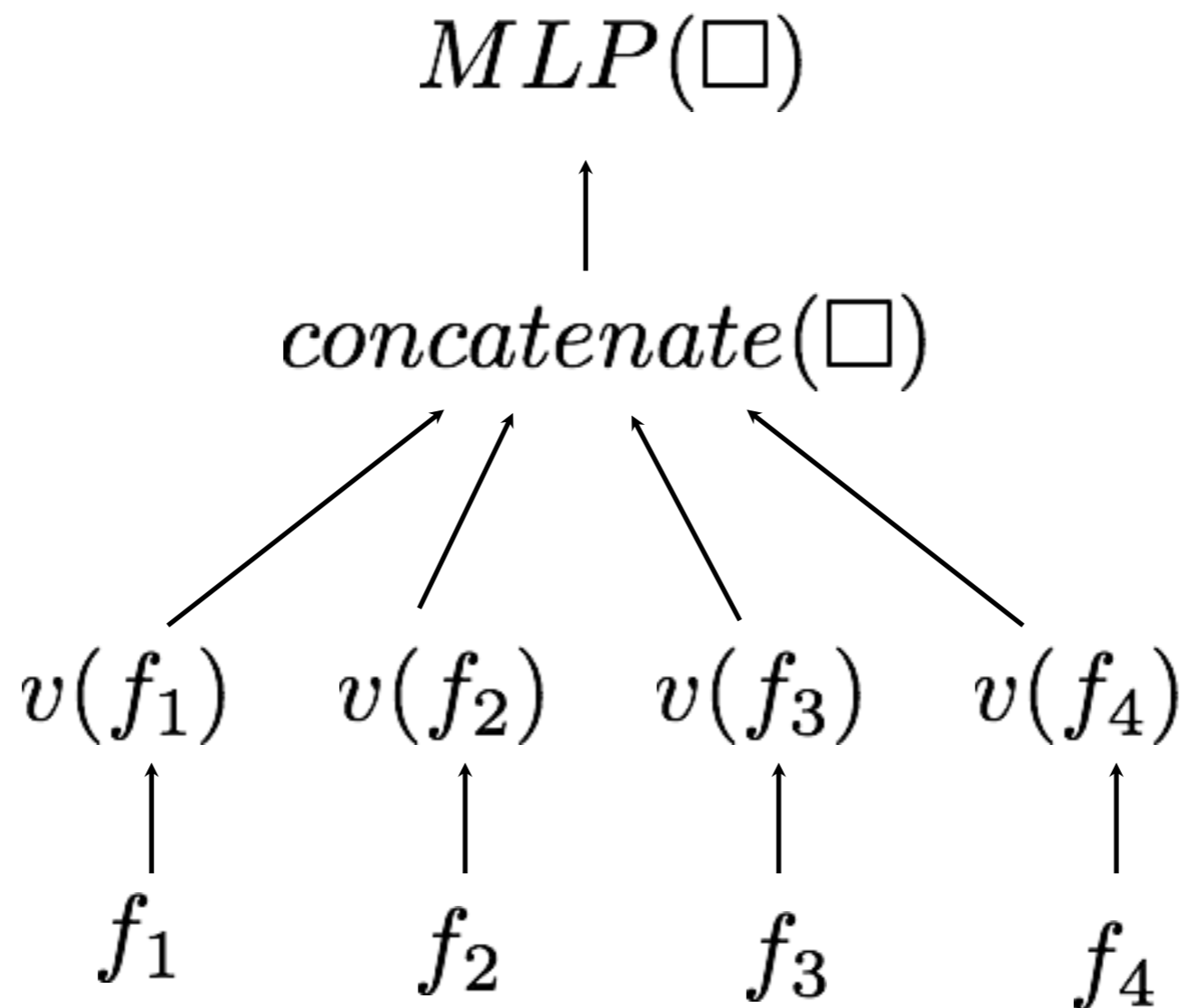
"feature embeddings"

- Each feature is assigned a vector.
- The input is a combination of feature vectors.
- The feature vectors are **parameters of the model** and are trained jointly with the rest of the network.
- **Representation Learning:** similar features will receive similar vectors.

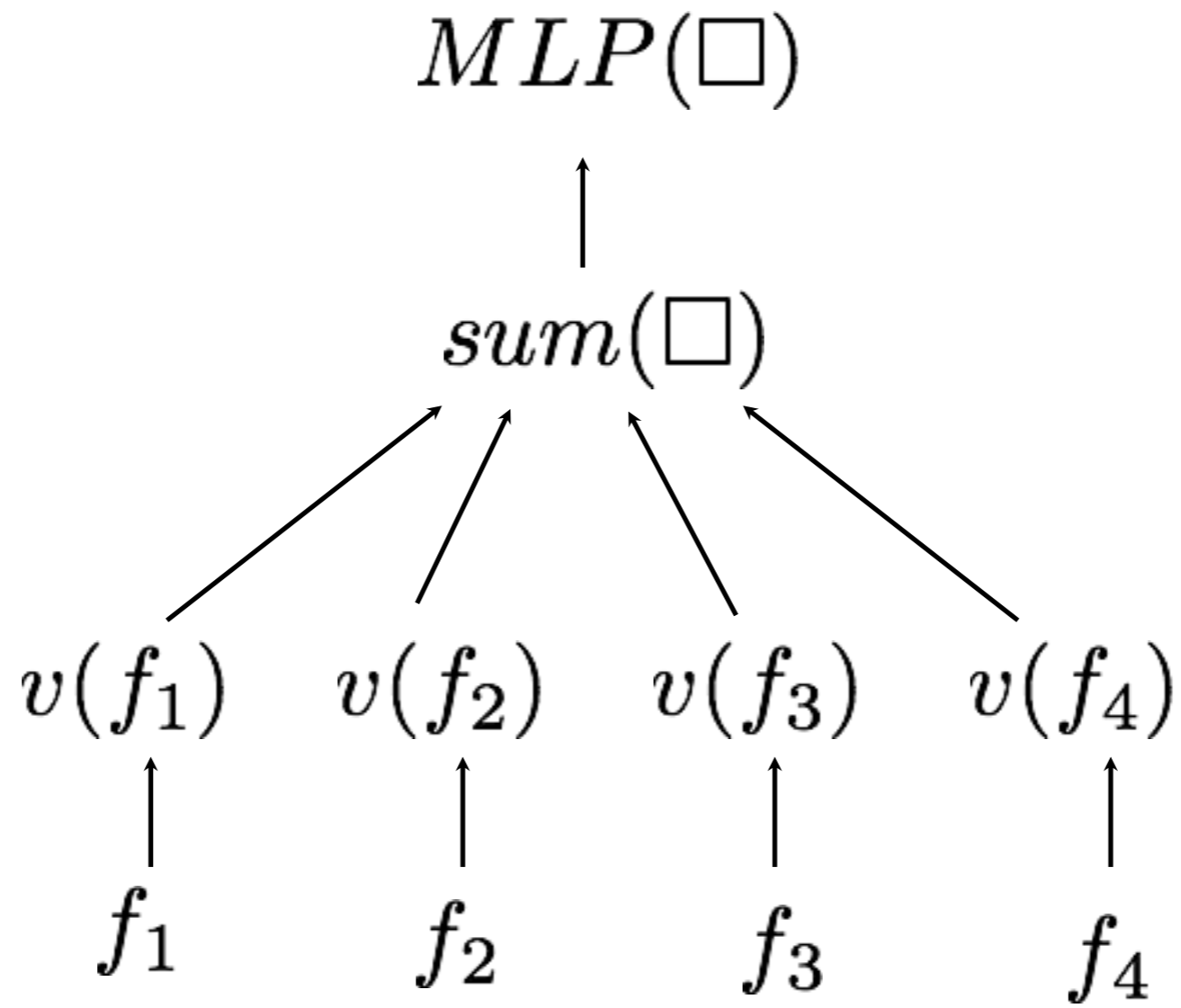
"feature embeddings"



"feature embeddings"



"feature embeddings"



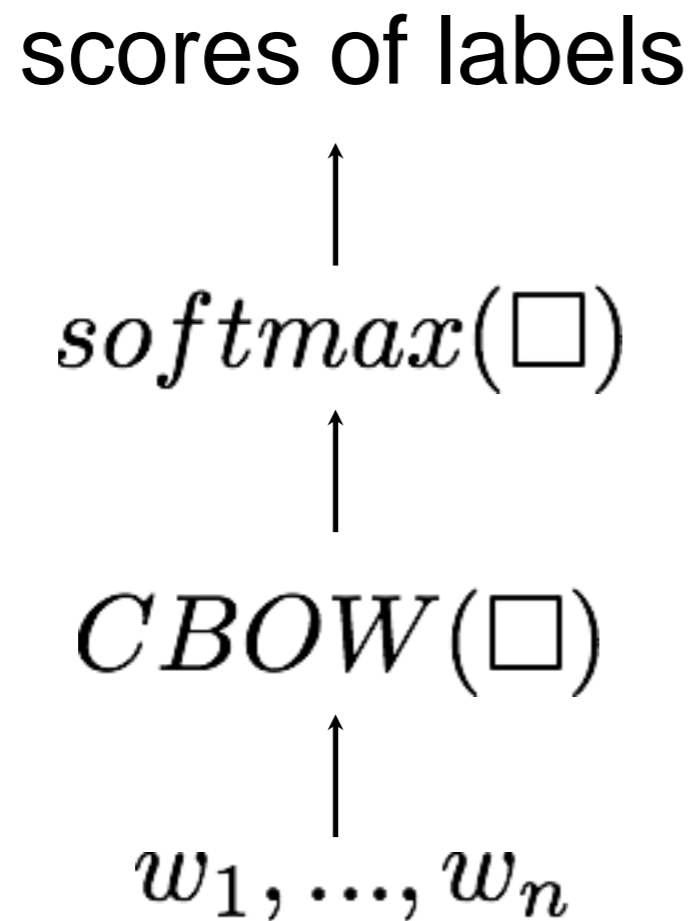
Continuous Bag of Words (CBOW)

$$CBOW(f_1, \dots, f_k) = \frac{1}{k} \sum_{i=1}^k v(f_i)$$

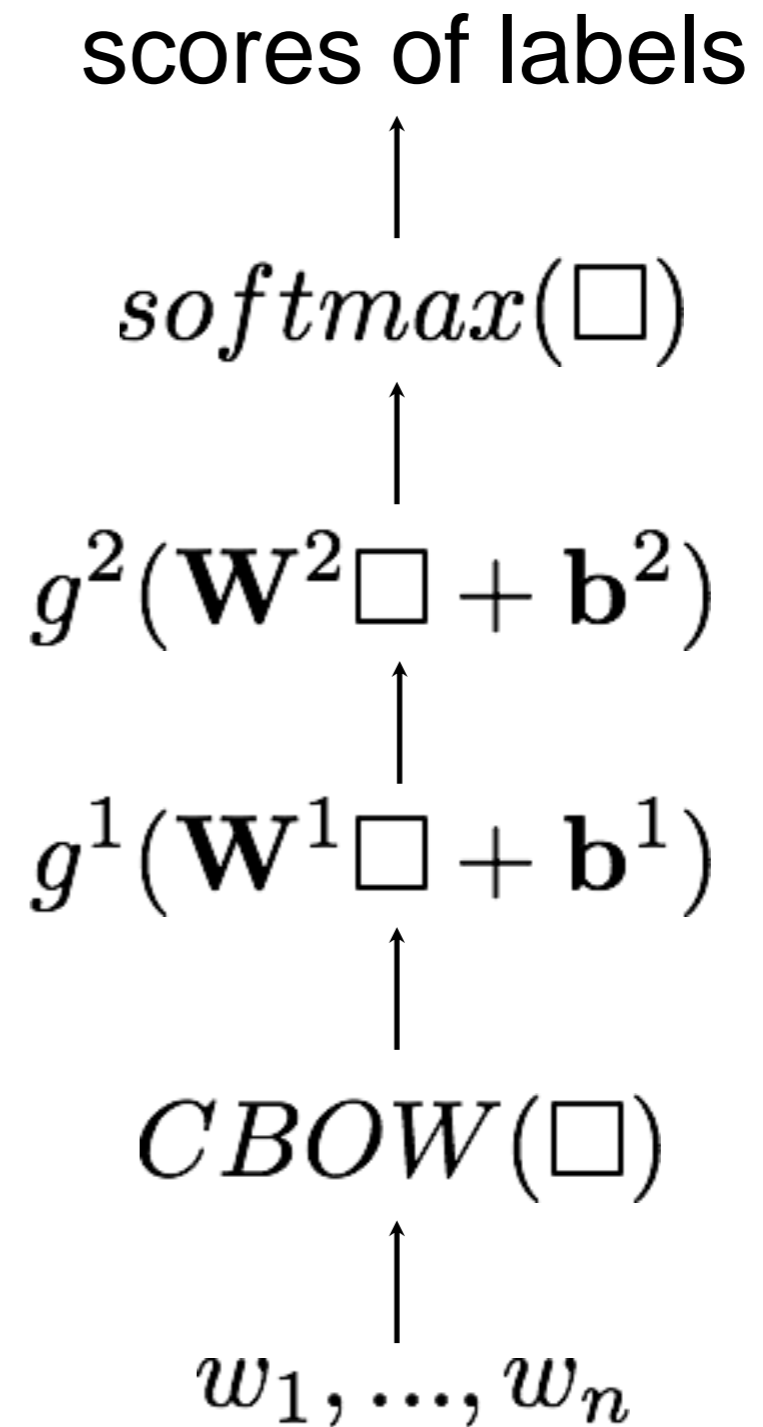
- a popular choice in document classification.
- can assign a different weight to each feature:

$$WCBOW(f_1, \dots, f_k) = \frac{1}{\sum_{i=1}^k a_i} \sum_{i=1}^k a_i v(f_i)$$

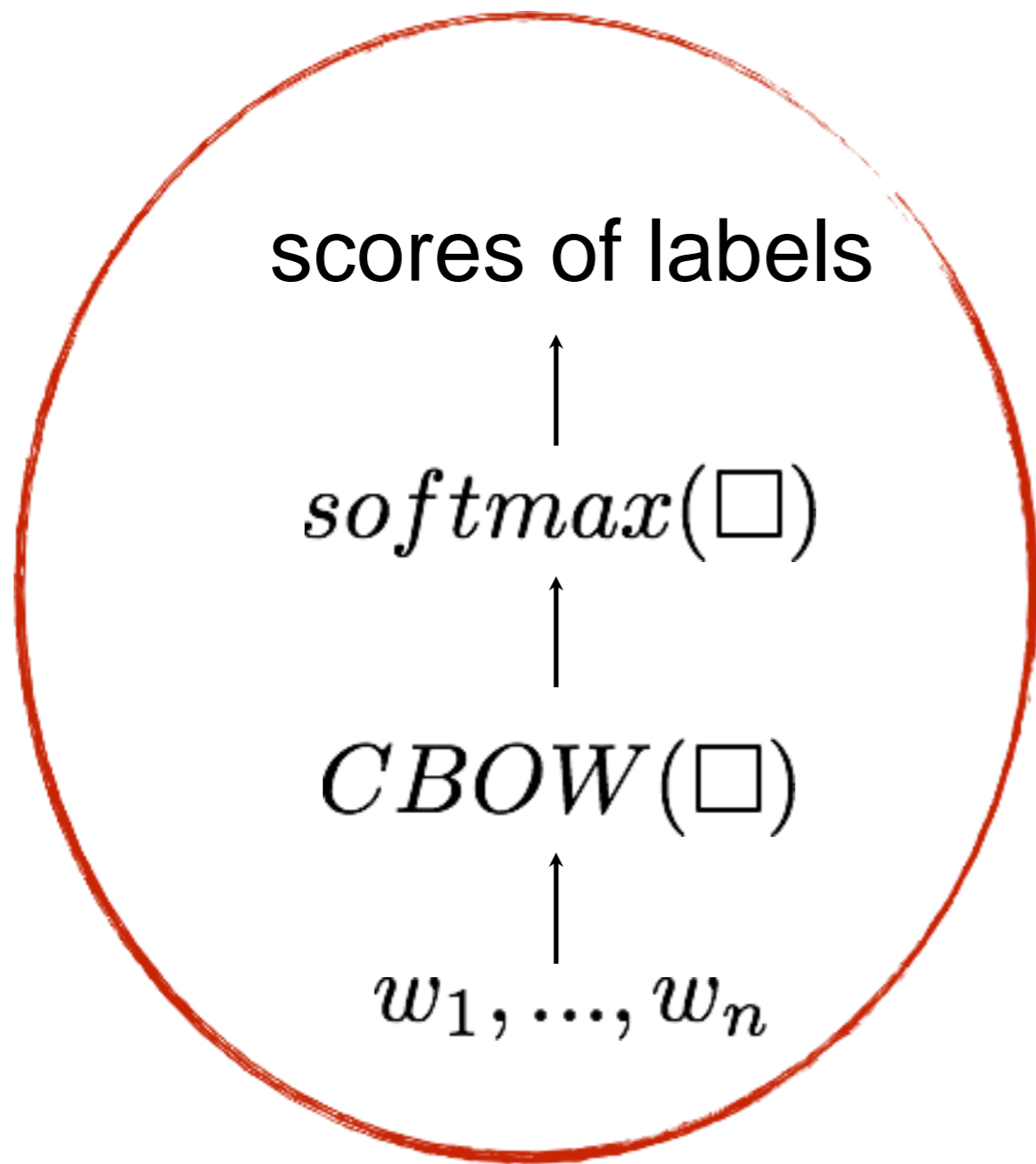
Text Classification with CBOW



"neural bag of words"



"deep averaging network"



If each feature is bigram,
works great.

Moving to unigrams, large drop.

Unigrams + MLP --> better
but not like bigrams.

"neural bag of words"

Importance of Ngrams

- While we can ignore global order in many cases...
- ... local ordering is still often very important.
- Local sub-sequences encode useful structures.

Importance of Ngrams

- While we can ignore global order in many cases...
- ... local ordering is still often very important.
- Local sub-sequences encode useful structures.

(so why not just assign a vector to each ngram?)

ConvNets

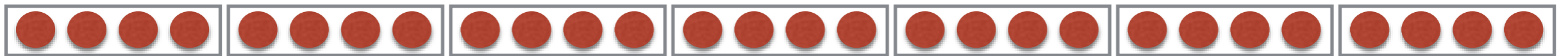
special architecture for local predictors

ConvNets

- CBOW allows encoding arbitrary length sequences, but loses all order information.
- Some local order (i.e. bigrams, trigrams) is informative. Yet, we do not care about exact position in the sequence. (think "good" vs. "not good")
- ConvNets (in language) allow to identify informative local predictors.
- Works by moving a shared function (feature extractor) over a sliding window, then pooling results.

ConvNets

- ConvNets have huge success in computer vision.
- It allows invariance to object position.
- It allows composing large predictors from small.



the actual service was not very good



dot



the

actual

service

was

not

very

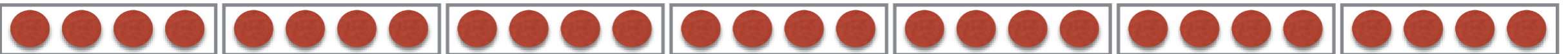
good



||



dot



the

actual

service

was

not

very

good

the actual



||



dot



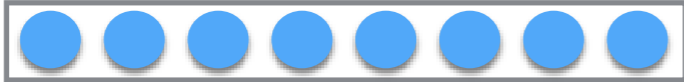
the actual service was not very good

the actual

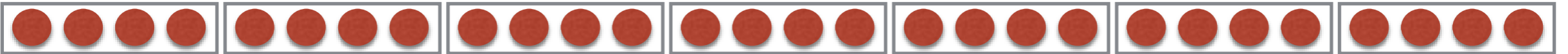
actual service



||



dot



the

actual

service

was

not

very

good

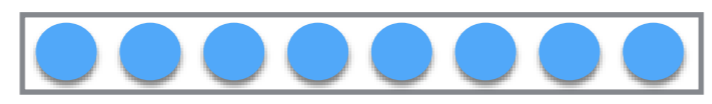
the actual

actual service

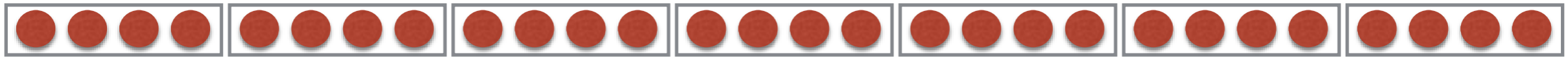
service was



||



dot



the

actual

service

was

not

very

good

the actual

actual service

service was

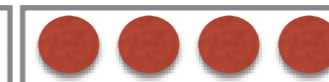
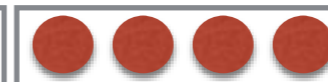
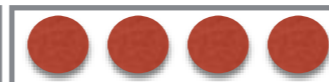
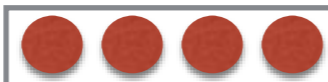
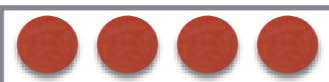
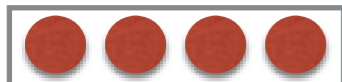
was not



||



dot



the

actual

service

was

not

very

good

the actual

actual service

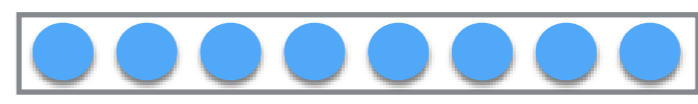
service was

was not

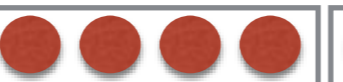
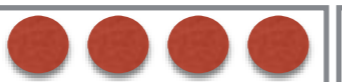
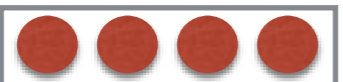
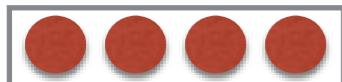
not very



||



dot



the

actual

service

was

not

very

good

the actual

actual service

service was

was not

not very

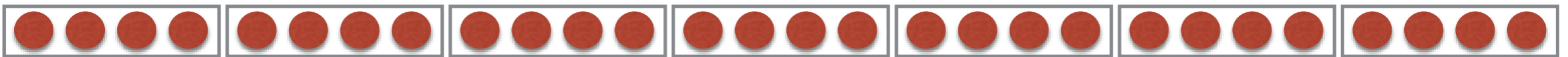
very good



||



dot



the

actual

service

was

not

very

good

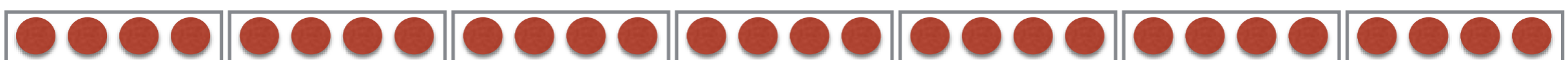
the actual



||

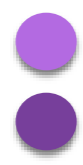


dot

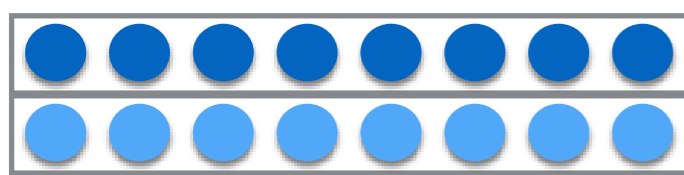


the actual service was not very good

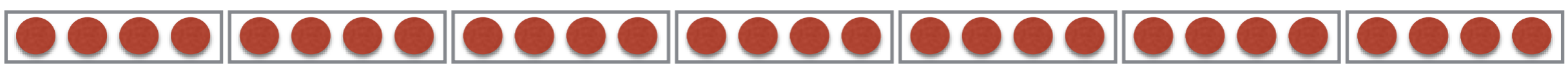
the actual



||



dot

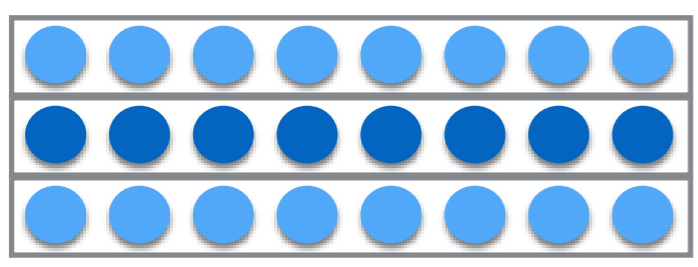


the actual service was not very good

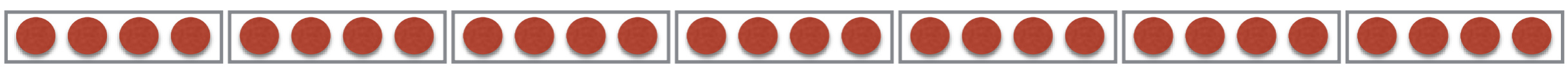
the actual



||



dot



the actual service was not very good

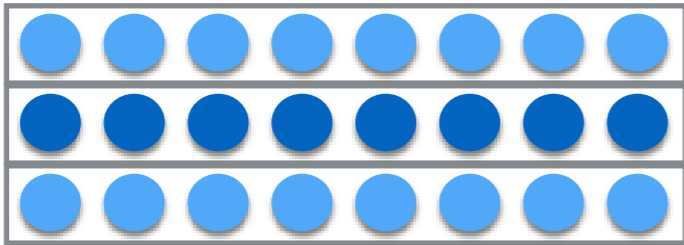
the actual



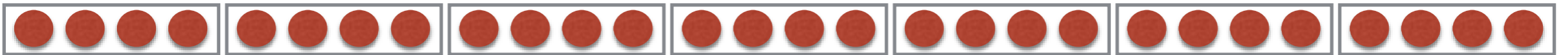
actual service



||



dot



the

actual

service

was

not

very

good

the actual



actual service



service was



was not



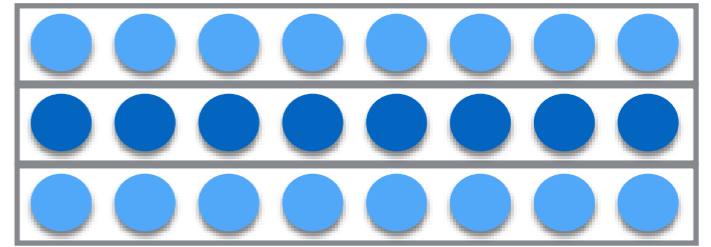
not very



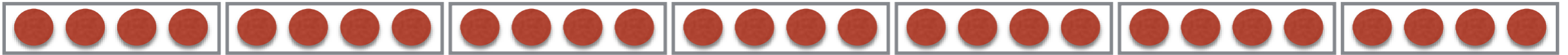
very good



||



dot



the

actual

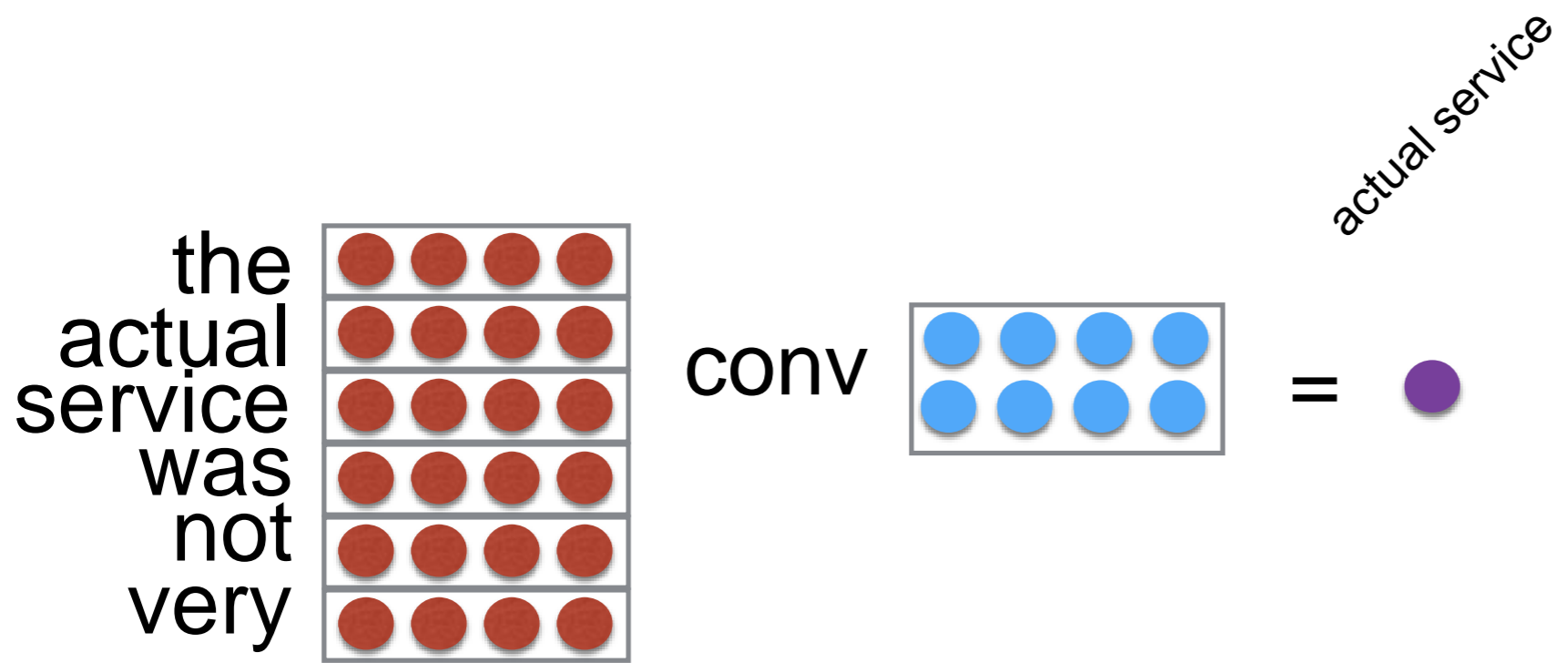
service

was

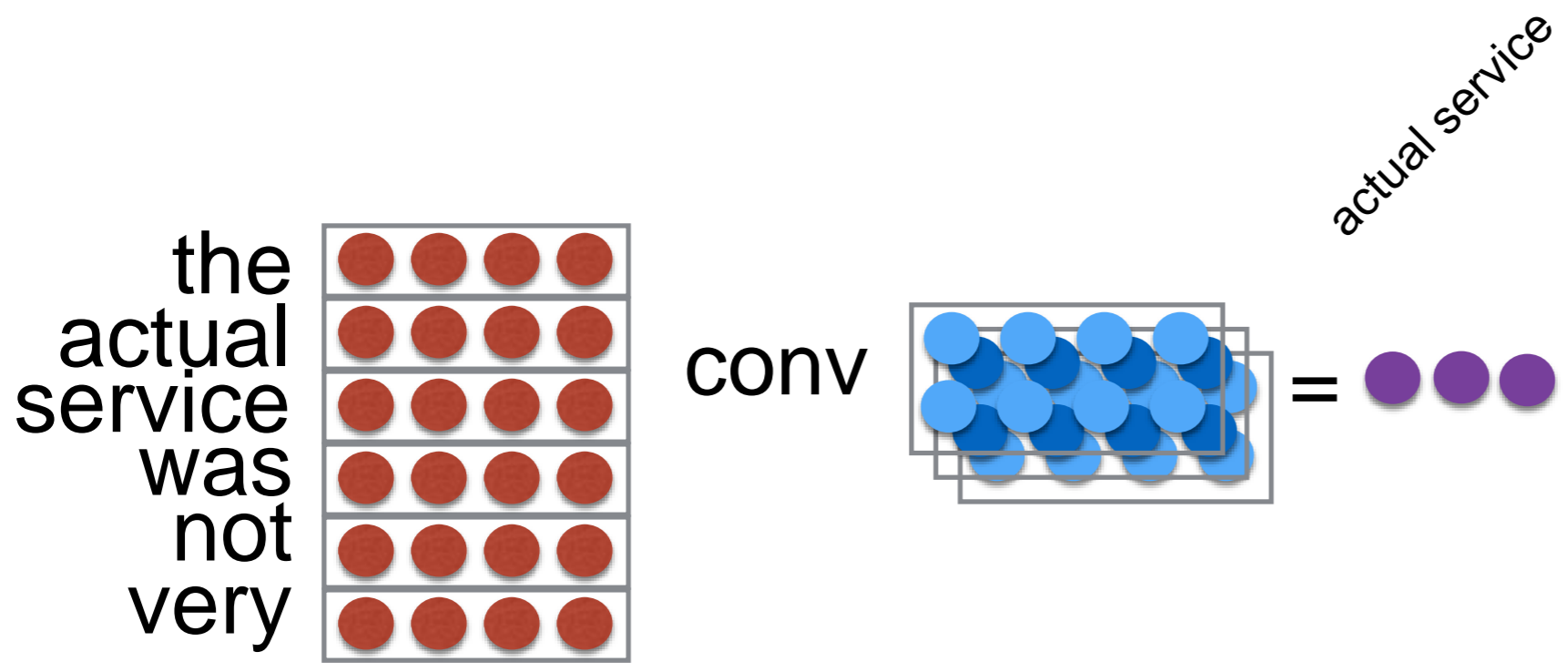
not

very

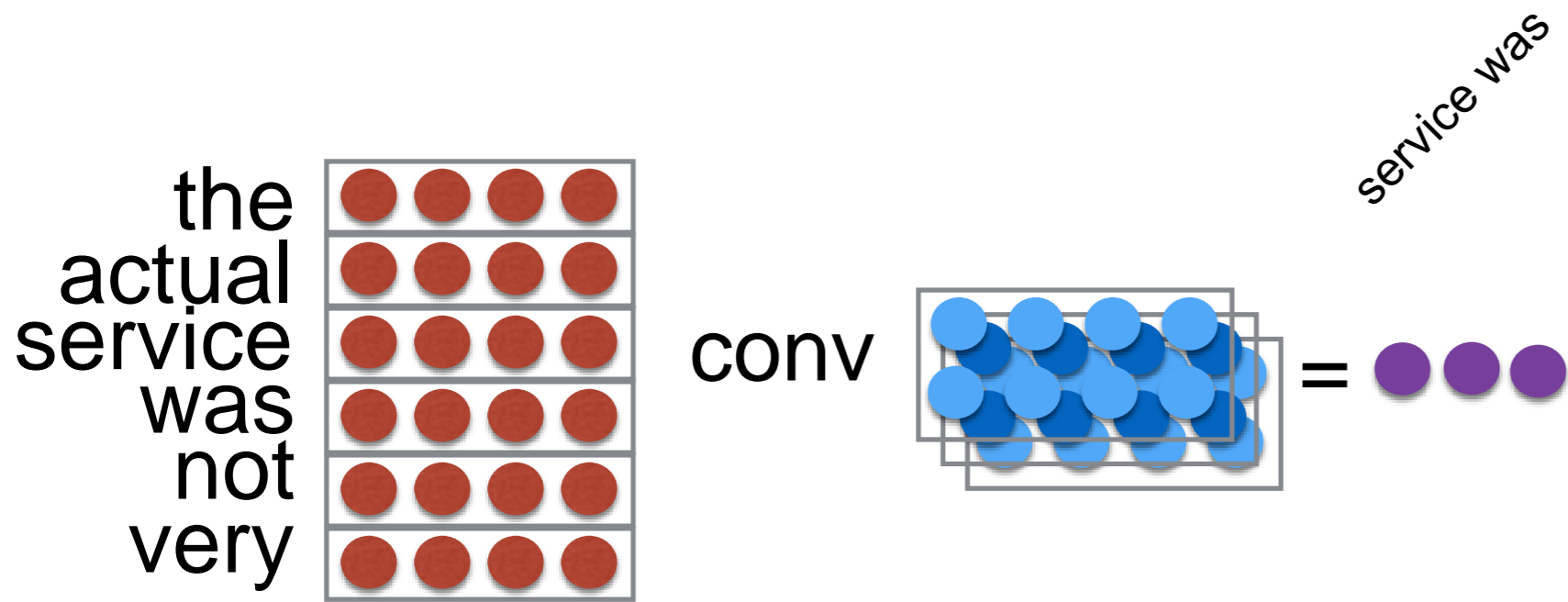
good



(another way to represent text convolutions)



(another way to represent text convolutions)



(another way to represent text convolutions)

the actual



actual service



service was



was not



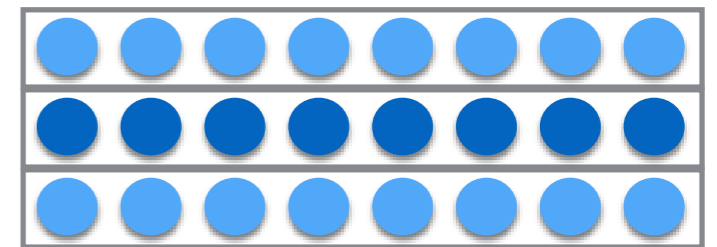
not very



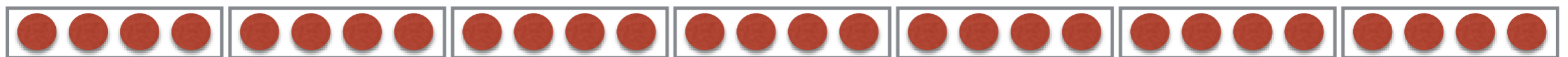
very good



||



dot



the

actual

service

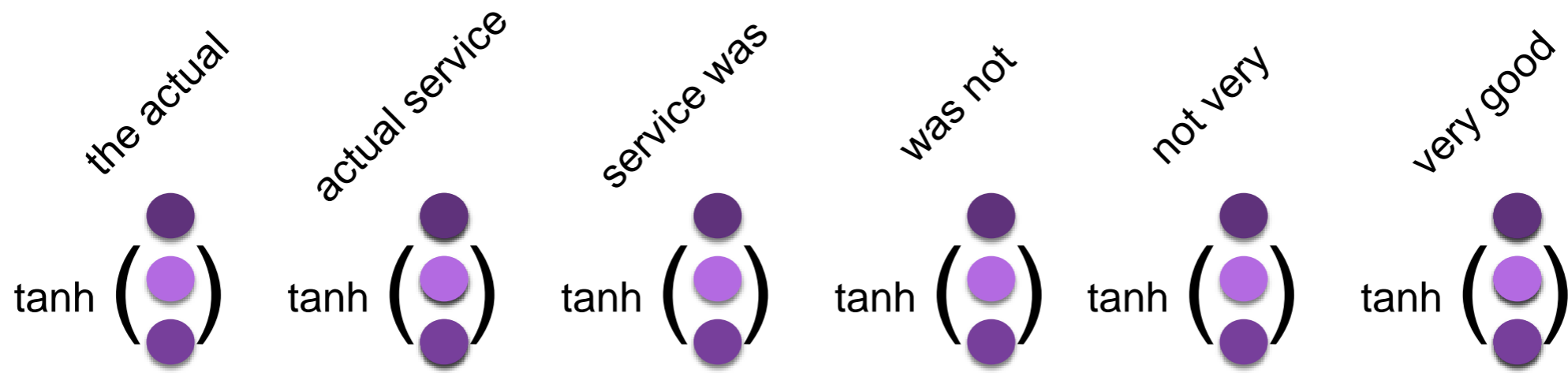
was

not

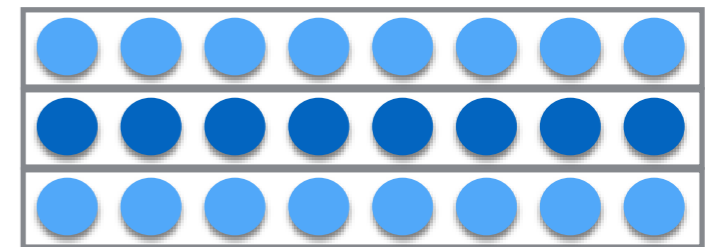
very

good

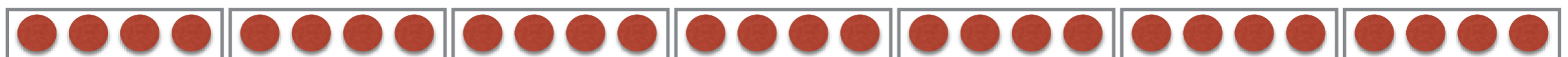
**(we'll focus on the 1-d view here,
but remember they are equivalent)**



||



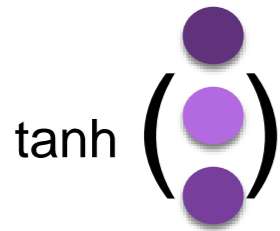
dot



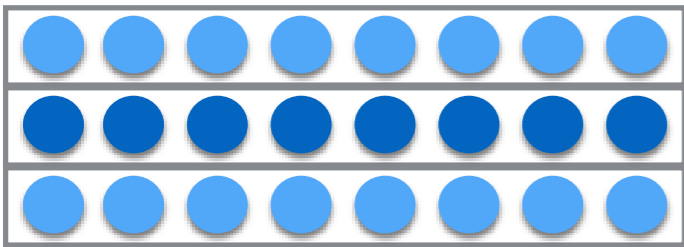
the actual service was not very good

(usually also add non linearity)

the actual



||



dot



the

actual

service

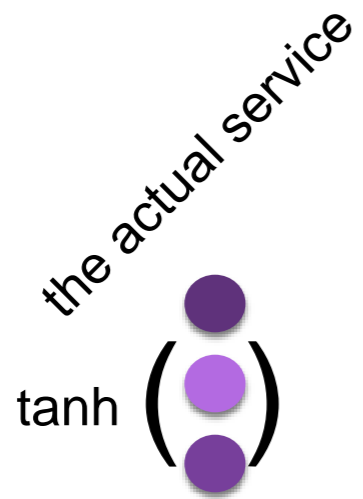
was

not

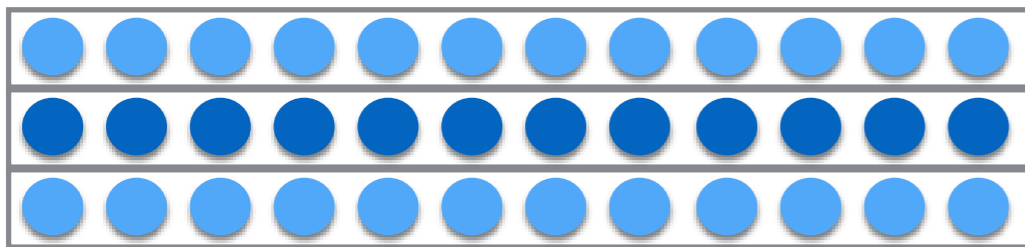
very

good

(can have larger filters)



||



dot



the actual service was not very good

(can have larger filters)

the actual



actual service



service was



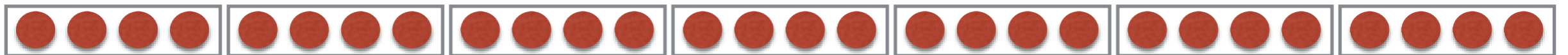
was not



not very



very good



the

actual

service

was

not

very

good

we have the ngram vectors. now what?

the actual



+

actual service



+

service was



+

was not



+

not very

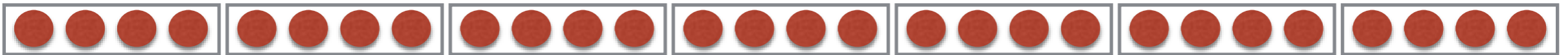


+

very good



=



the

actual

service

was

not

very

good

can do "pooling"

"Pooling"

Combine K vectors into a single vector

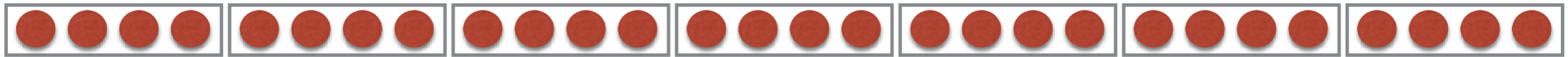
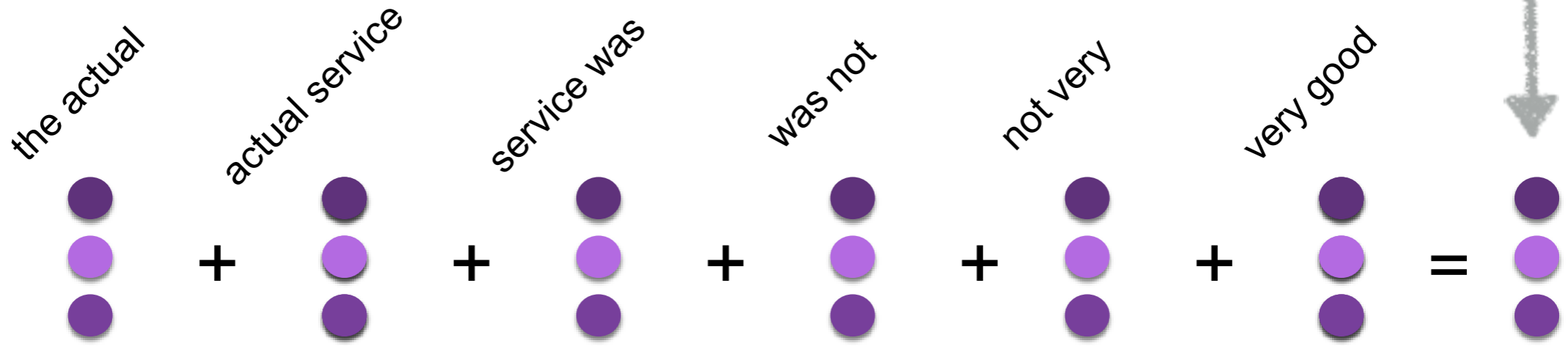
"Pooling"

Combine K vectors into a single vector

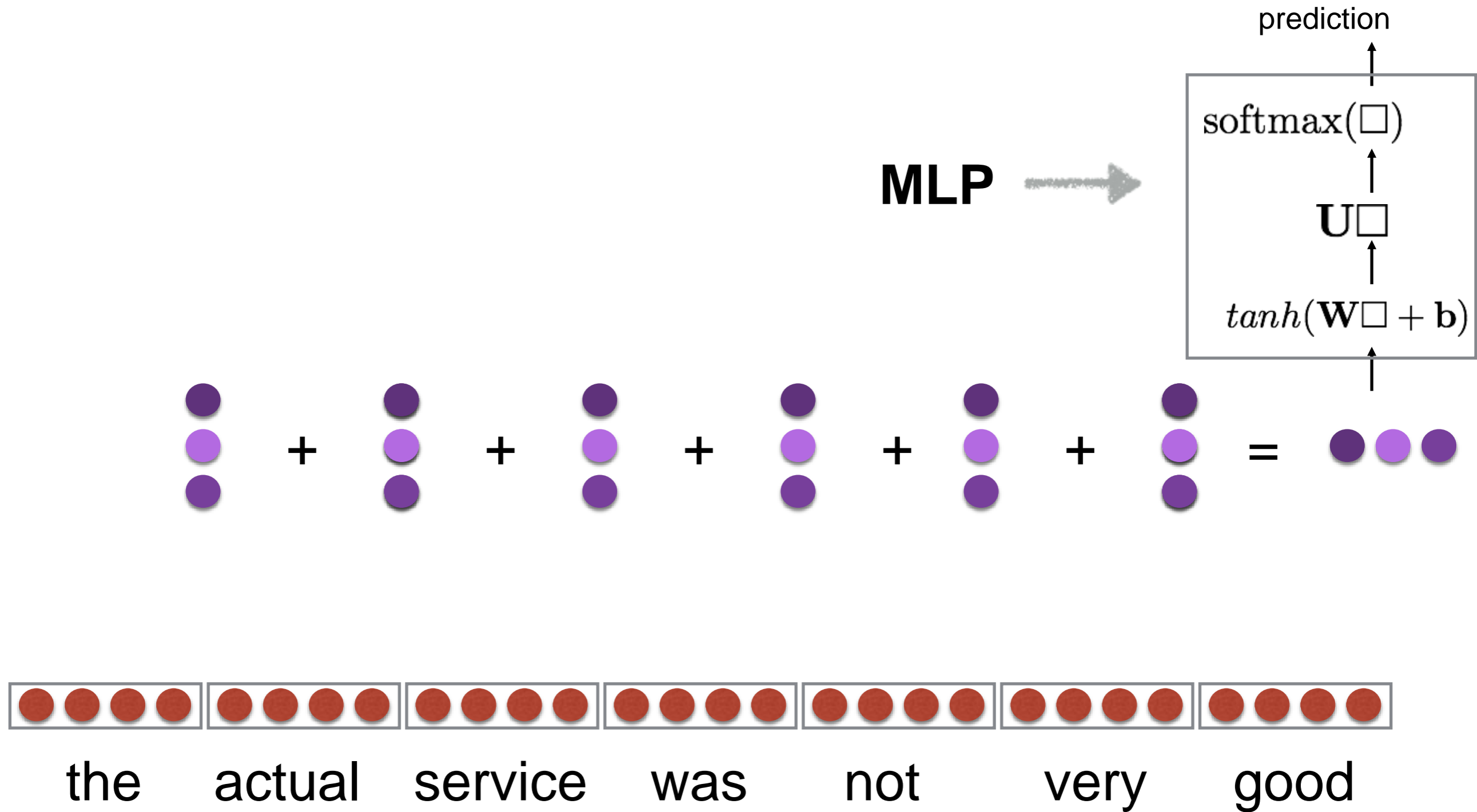
**This vector is a summary of the K vectors,
and can be used for prediction.**

average pooling

average vector

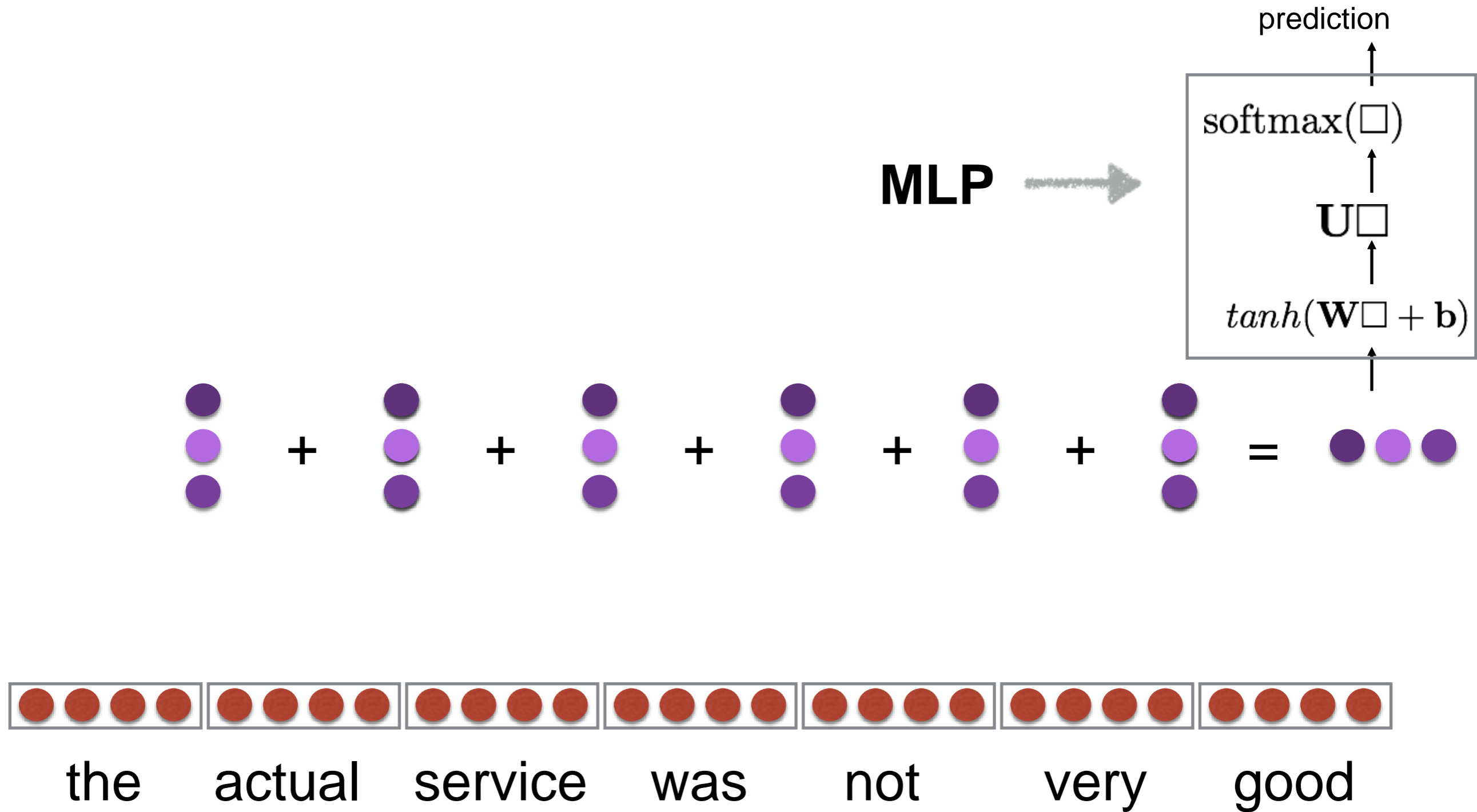


the actual service was not very good



train end-to-end for some task

(train the MLP, the filter matrix, and the embeddings together)



train end-to-end for some task

(train the MLP, the filter matrix, and the embeddings together)
the vectors learn to capture what's important

we have the ngram vectors. now what?

Can look at the differences between terms.

microsoft <i>office</i> software		car <i>body</i> shop	
Free <i>office</i> 2000	0.550	car <i>body</i> kits	0.698
download <i>office</i> excel	0.541	auto <i>body</i> repair	0.578
word <i>office</i> online	0.502	auto <i>body</i> parts	0.555
apartment <i>office</i> hours	0.331	wave <i>body</i> language	0.301
massachusetts <i>office</i> location	0.293	calculate <i>body</i> fat	0.220
international <i>office</i> berkeley	0.274	forcefield <i>body</i> armour	0.165

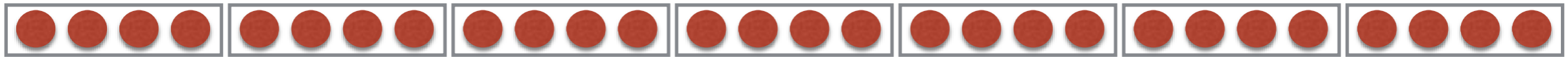
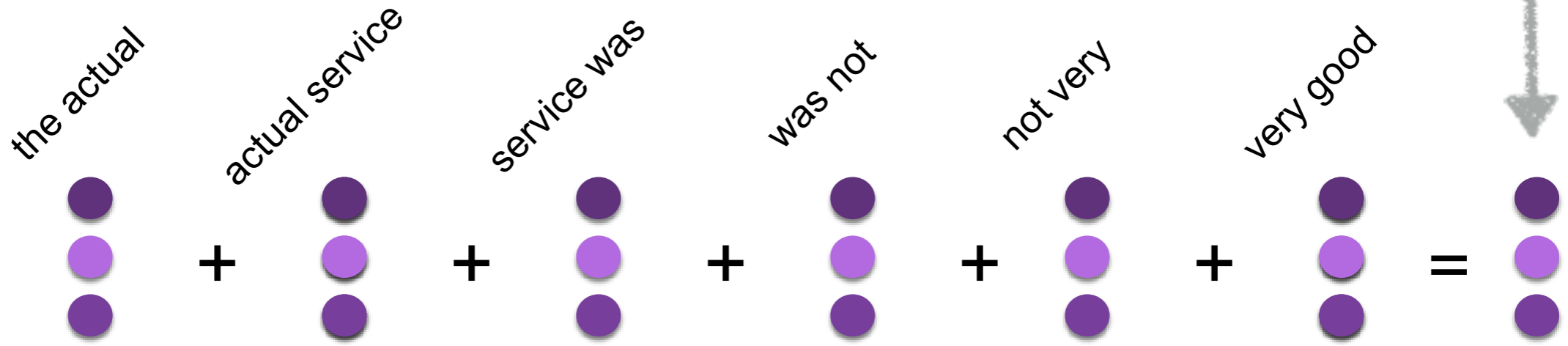
Table 2: Sample word n-grams and the cosine similarities between the learned word-n-gram feature vectors of “*office*” and “*body*” in different contexts after the CLSM is trained.

A Latent Semantic Model with Convolutional-Pooling Structure for Information Retrieval

Yelong Shen Microsoft Research Redmond, WA, USA yeshen@microsoft.com	Xiaodong He Microsoft Research Redmond, WA, USA xiahe@microsoft.com	Jianfeng Gao Microsoft Research Redmond, WA, USA jfgao@microsoft.com	Li Deng Microsoft Research Redmond, WA, USA deng@microsoft.com	Grégoire Mesnil University of Montréal Montréal, Canada gregoire.mesnil@umontreal.ca
---	--	---	---	---

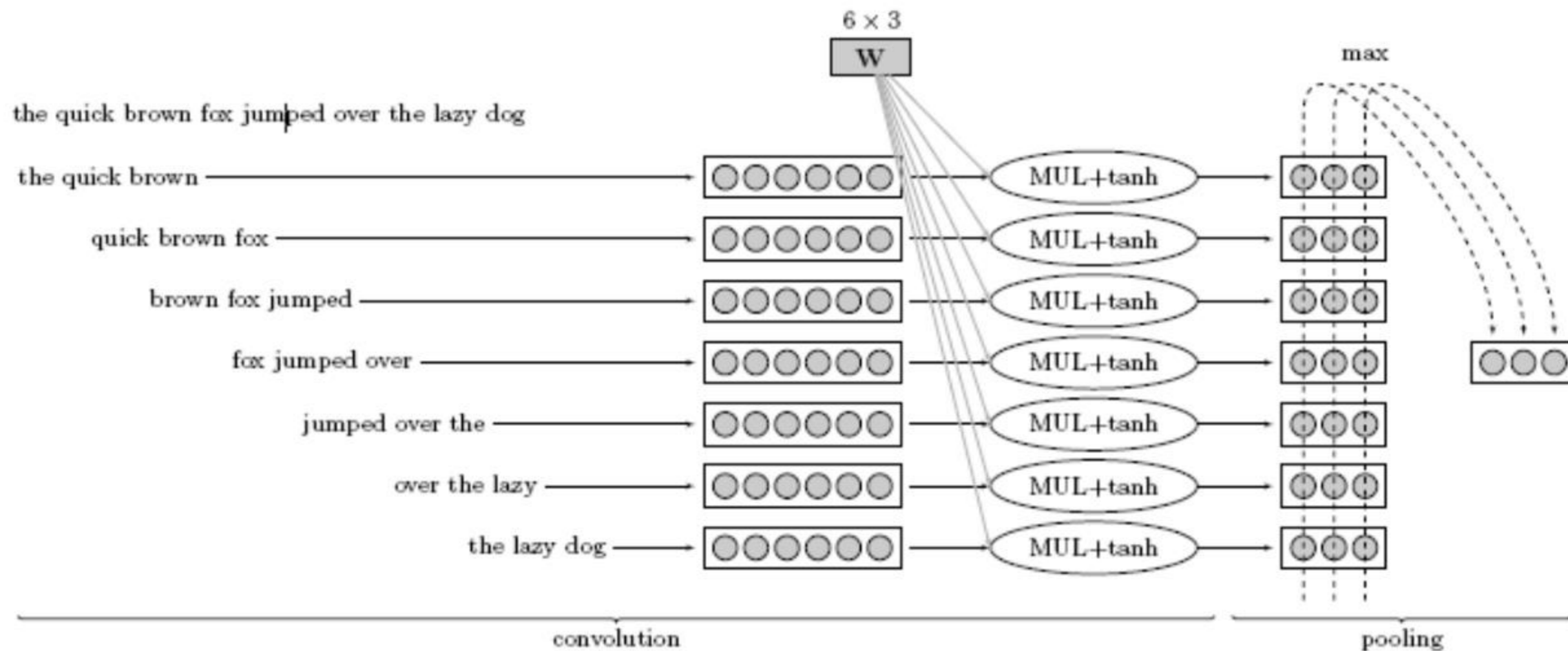
average pooling

average vector



the actual service was not very good

Another way to draw this:



one benefit of max-pooling: it's "**interpretable**"

we can know where each element
in the summary vector came from

Examples of resulting "summaries"

microsoft **office excel** could allow remote **code execution**

welcome to the **apartment office**

online **body fat** percentage **calculator**

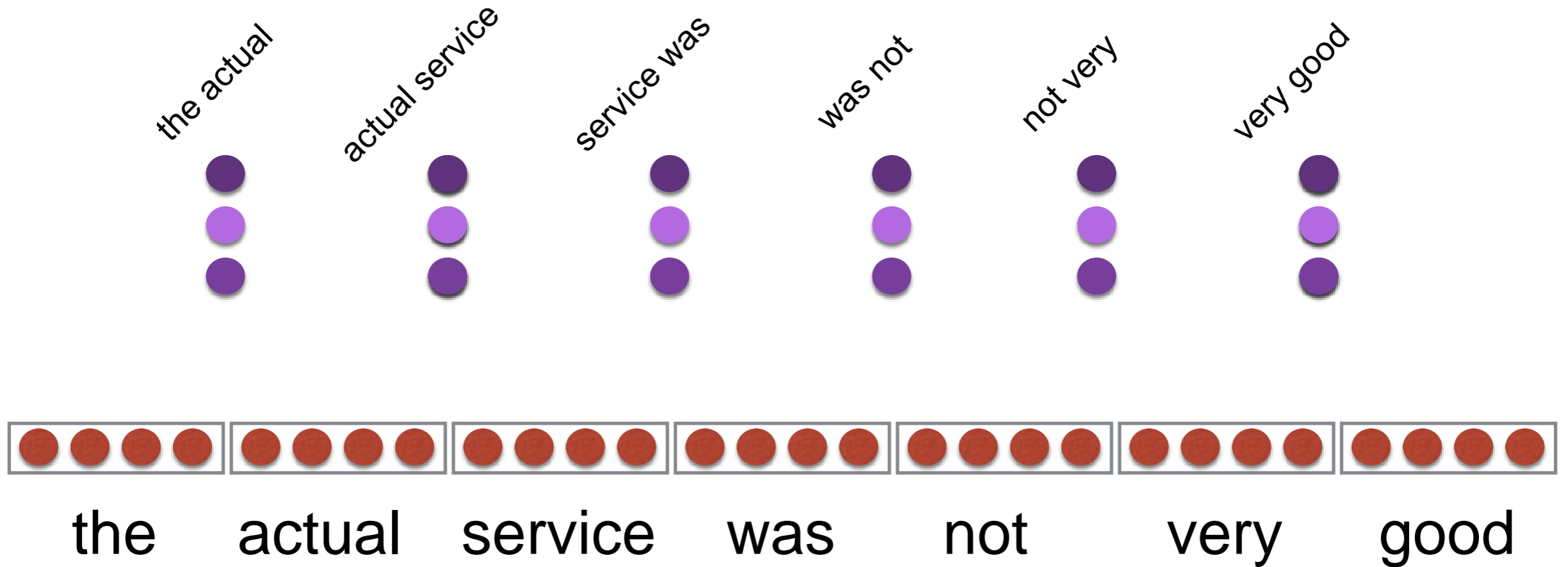
online **auto body** repair **estimates**

vitamin a the **health** benefits given by **carrots**

calcium supplements and **vitamin d** discussion stop **sarcoidosis**

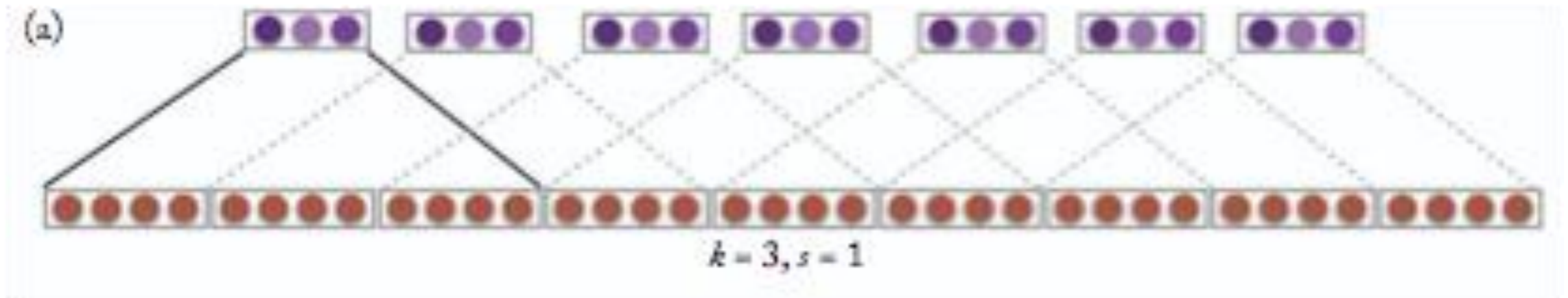
Table 3: Sample document titles. We examine the five most active neurons at the max-pooling layer and highlight the words in **bold** who win at these five neurons in the *max* operation. Note that, the feature of a word is extracted from that word together with the context words around it, but only the center word is highlighted in bold.

Strides



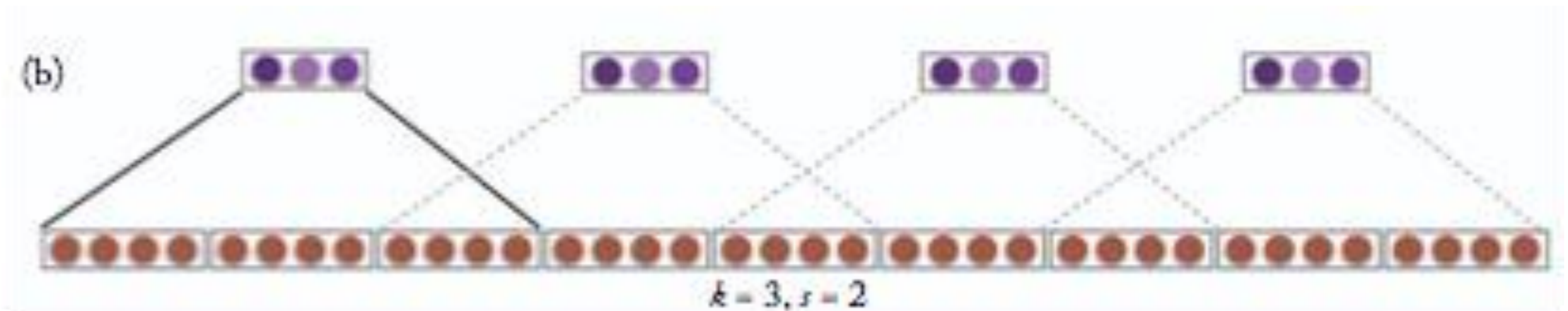
strides = how much you move

Strides



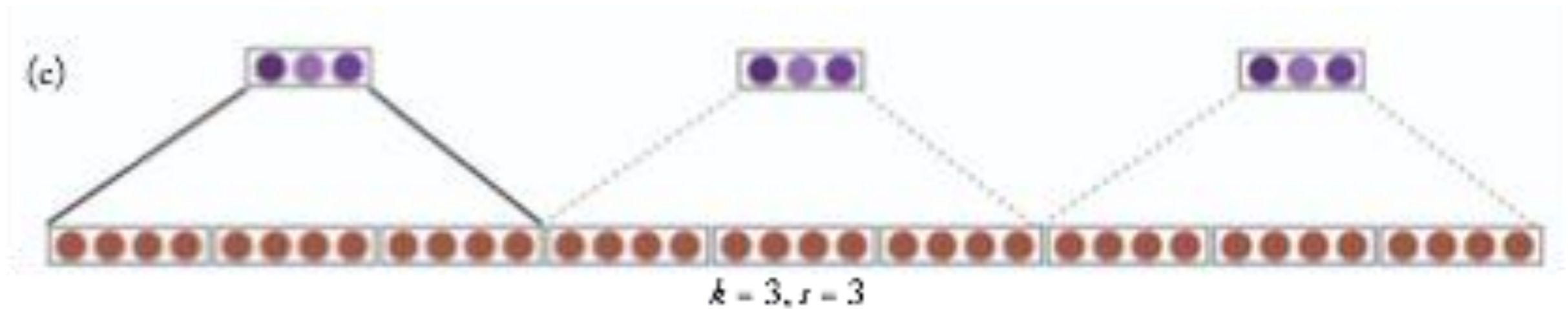
$k = 3, \text{ stride} = 1$

Strides



$k = 3, \text{ stride} = 2$

Strides



$k = 3, \text{ stride} = 3$

Hierarchy

Hierarchy



the actual



actual service



service was



was not



not very



very good



the

actual

service

was

not

very

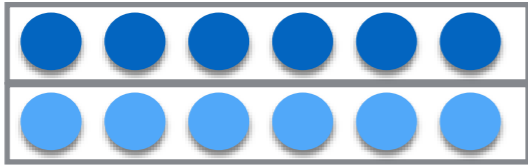
good

can have hierarchy

the actual service



||



dot



the actual



actual service



service was



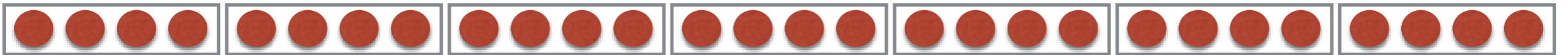
was not



not very



very good



the

actual

service

was

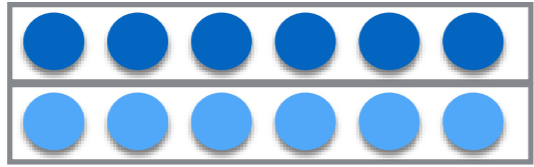
not

very

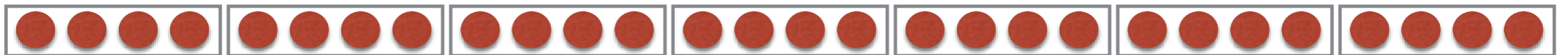
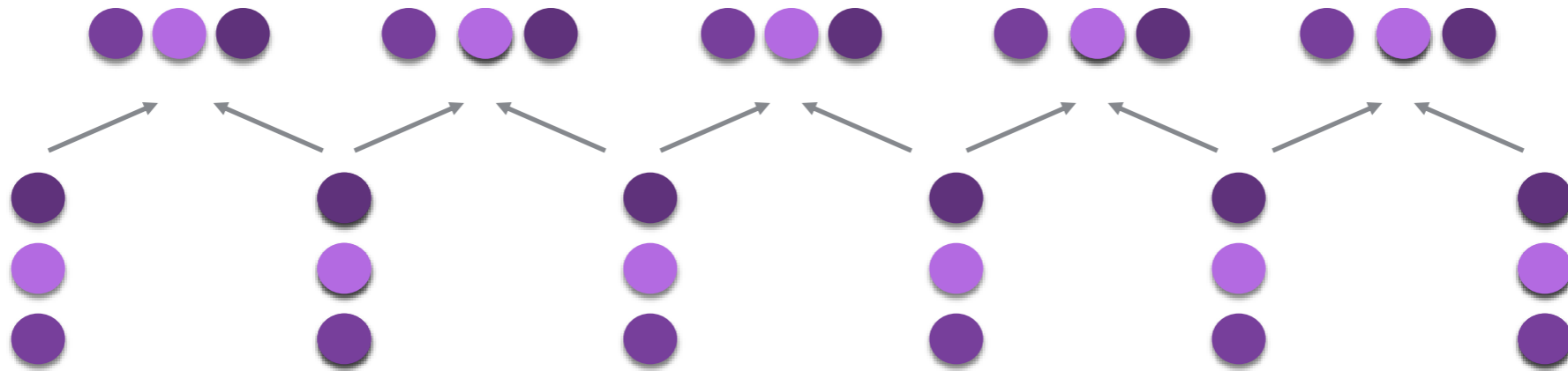
good

can have hierarchy

II



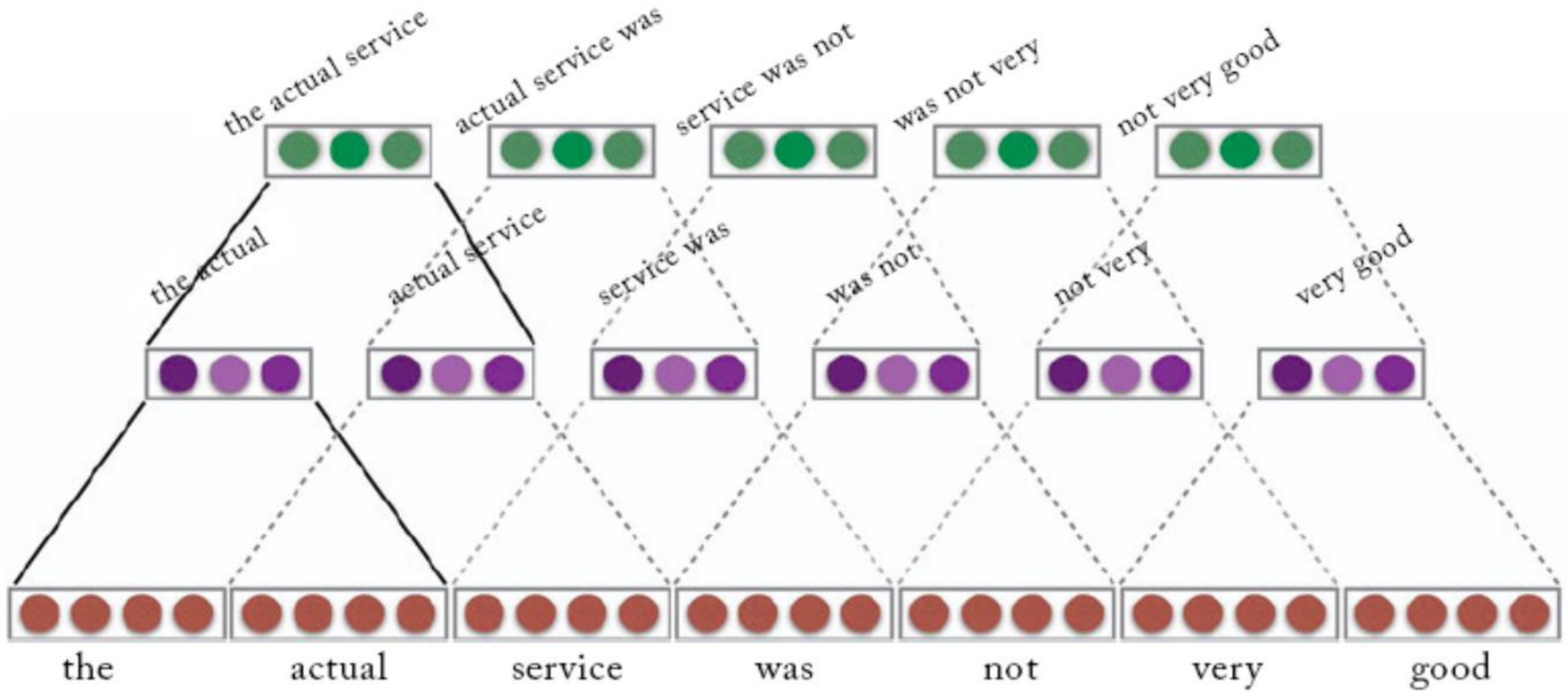
dot



the actual service was not very good

(can combine: **pooling + hierarchy**)

Hierarchy



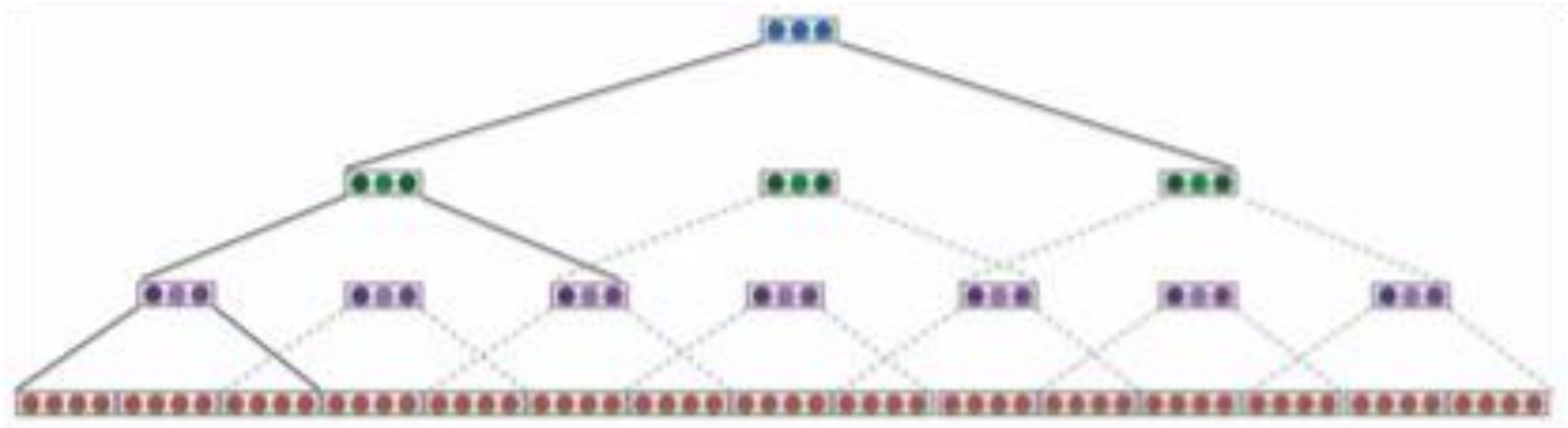
2-layer hierarchical conv with $k=2$

Dilated Convolutions

we want to cover more of the sequence

idea: strides + hierarchy

Dilated Convolutions



dilated convolution, $k=3$

idea: strides + hierarchy

ConvNets Summary

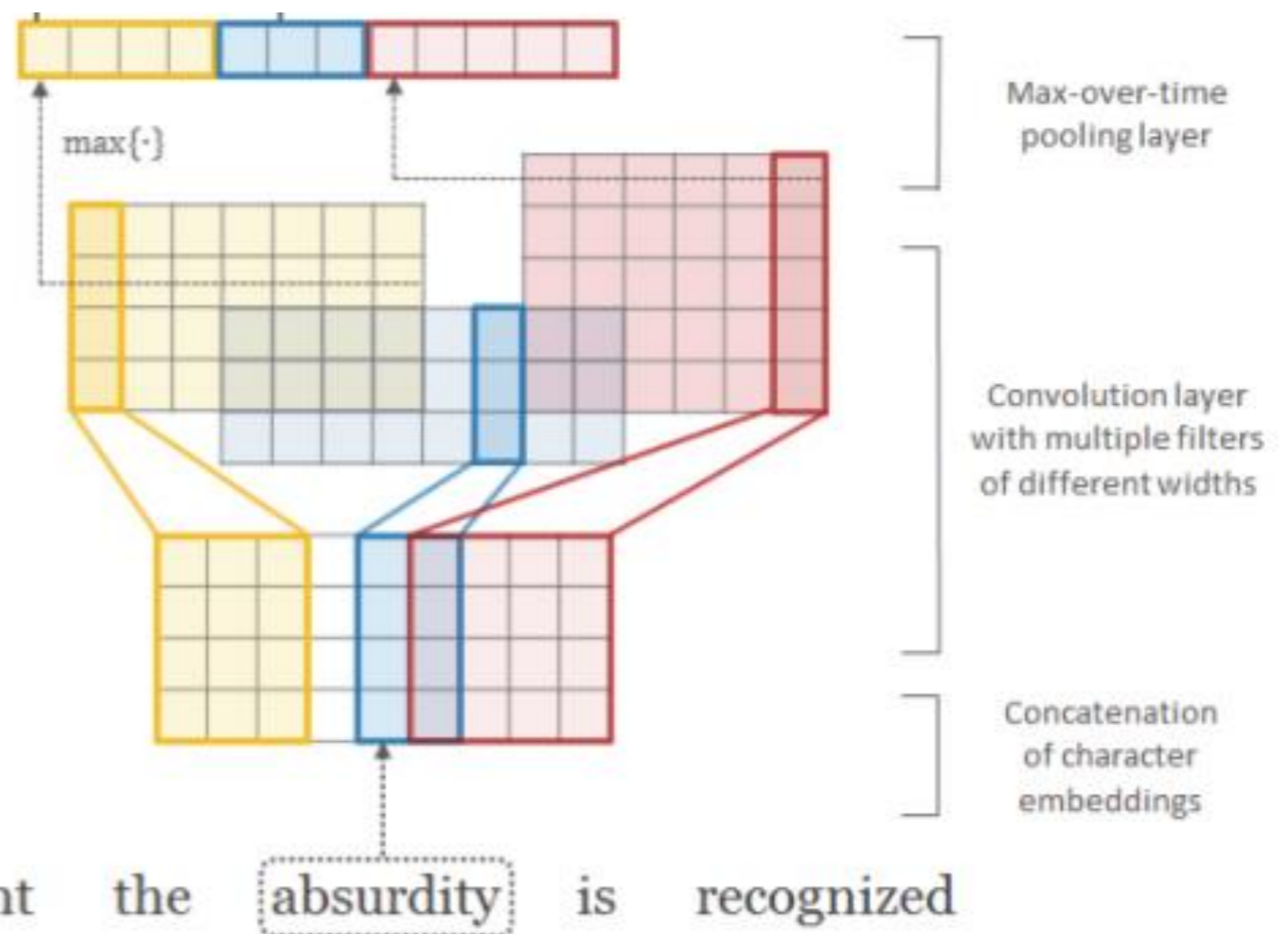
- Shared matrix used as feature detector.
- Extracts interesting ngrams.
- Pool ngrams to get fixed length representation.
- Max-pooling works well.
 - Max vs. Average pooling.
- Use hierarchy / dilation to expand coverage.
- Train end-to-end.

Character CNNs

- Fix the input OOV problem
 - Input: some insight in word shapes (xxxxing, xxxxly)
 - Output: can't ever output a word not in vocabulary
- Idea
 - Instead (or in addition of) word embedding
 - Use word = CNN over character sequences

Char CNN for Words

- Varied filter sizes
- Word embedding



Character-Aware Neural Language Models

Yoon Kim
School of Engineering
and Applied Sciences
Harvard University
yoonkim@seas.harvard.edu

Yacine Jernite
Courant Institute
of Mathematical Sciences
New York University
jernite@cs.nyu.edu

David Sontag
Courant Institute
of Mathematical Sciences
New York University
dsontag@cs.nyu.edu

Alexander M. Rush
School of Engineering
and Applied Sciences
Harvard University
srush@seas.harvard.edu

- Can't differentiate between words w similar spellings
- Solution: add small correction $[e_w = \text{CNN}(\text{chars}_w) + M \cdot \text{corr}_w]$

Alternative: Hashing Trick

- ConvNet is an architecture for finding good ngrams.
- But if we know ngrams are important, why not just have ngram embeddings (ngram vectors)?
- --> for large vocabulary, not scalable.

Can't represent all ngrams, don't know which are important.

Alternative: Hashing Trick

- **Problem:** our ngram vocabulary size is 10^9
- **Solution:** use smaller space via hashing, allow feature clashes.

Hashing Trick

- We have $> 10^9$ different ngrams.
- We can afford $\sim 10^6$ different embeddings.
- Map each ngram to a number in $[0, 10^6]$
- Use the corresponding embedding vector.
- Clashes will happen, but it will probably be ok.
- Even safer: map each ngram to two numbers using two different hash functions, sum the vectors.

Hashing Trick vs ConvNets

- What are the benefits of using bag of ngrams?
- What are the benefits of using ConvNet (ngram detector)?
- Does it matter if the vocabulary size is small or large?

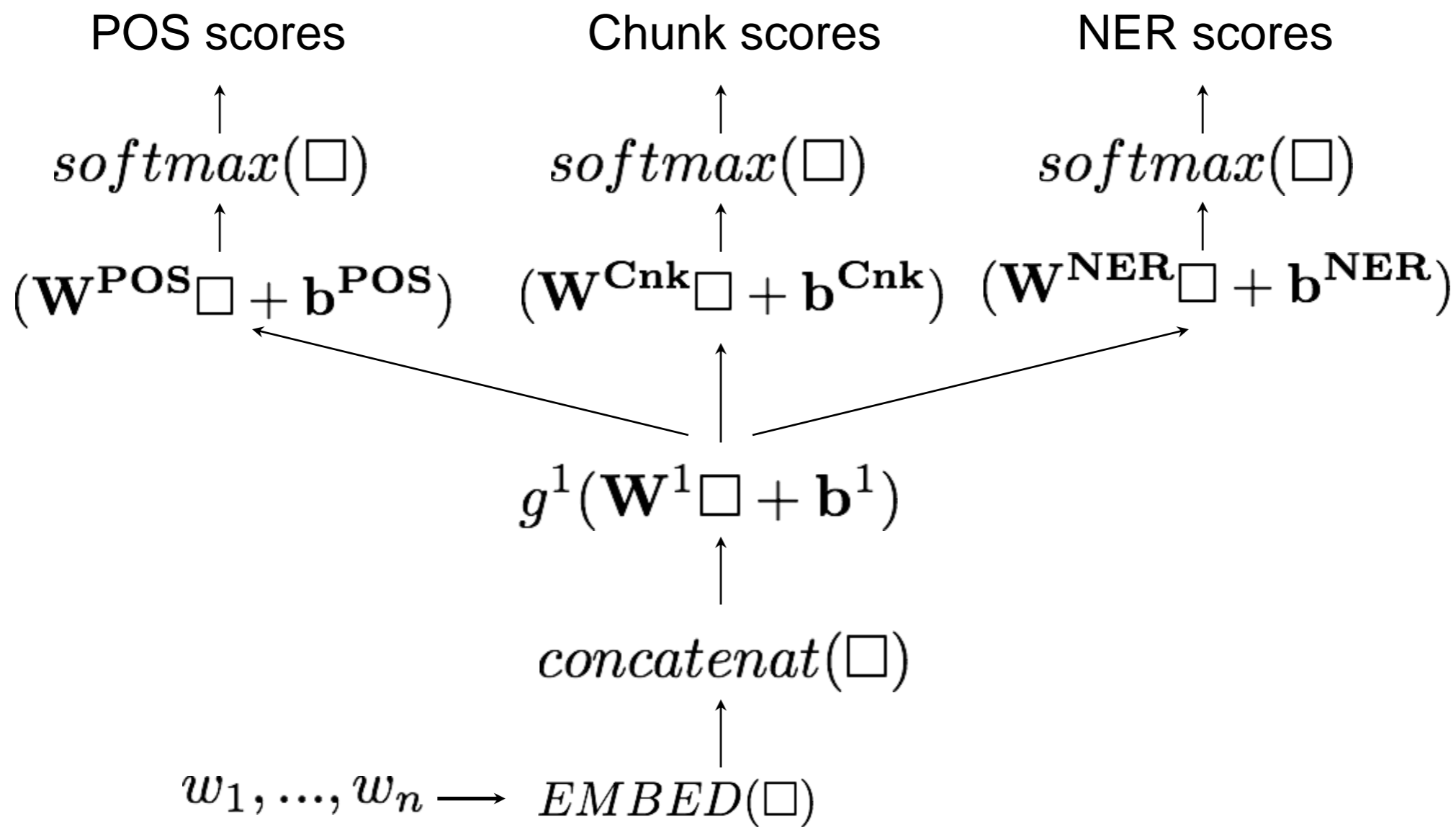
(discuss)

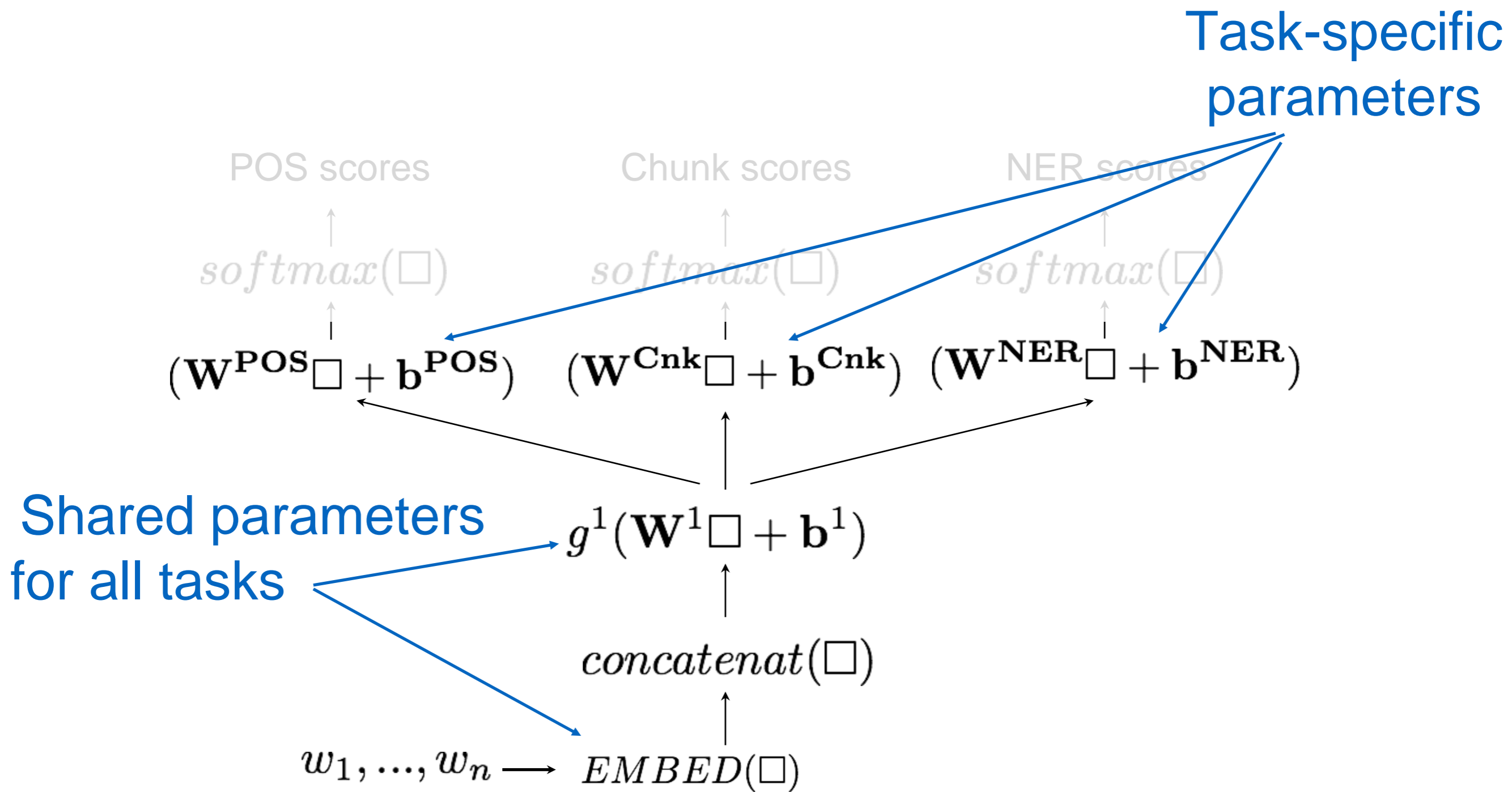
Multi-task Learning

(time permitting)

The pitch

- Different NLP prediction tasks have shared structures.
- Hints for predicting A may help to predict B.
- Instead of training a network to do one thing, train it to do several things.
- YOU ARE ALL WINNERS





Multi-Task Learning

