

# Deep Learning

## Episode 3

# Deep learning for natural language

Maxim Borisyak, Alexander Panin, Andrey Ustyuzhanin



Yandex  
Data Factory

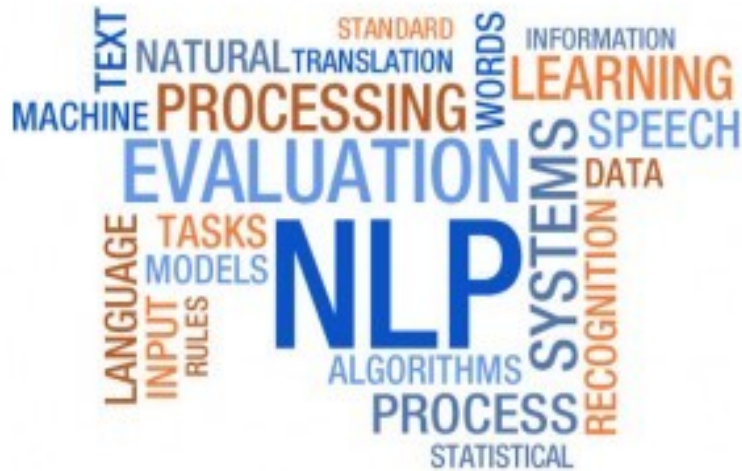
LAMBDA



British Hedgehog  
Preservation Society

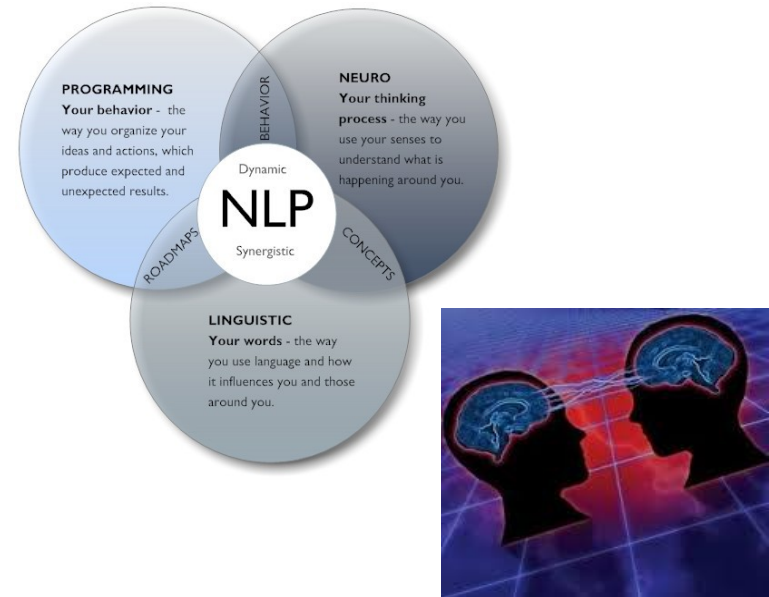


# What is NLP?



NLP

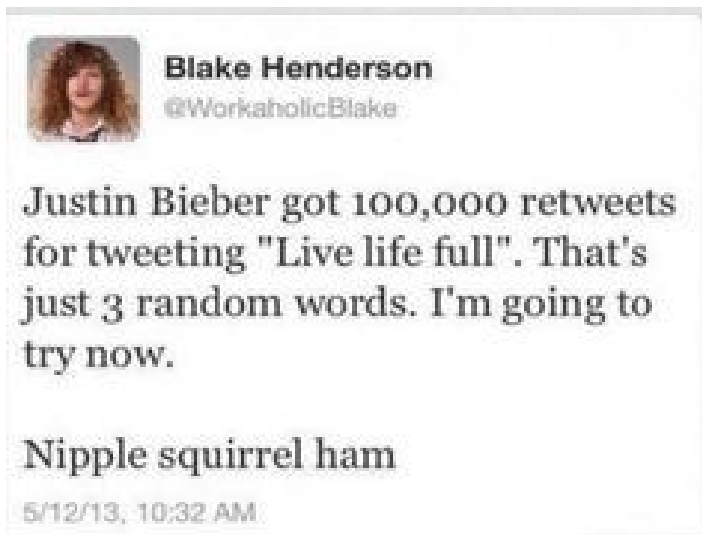
Light side of the force



NLP

Dark side of the force<sub>2</sub>

# Text classification/regression



Other usage:

- Adult content filter (safe search)
- Detect age/gender/interests by search queries
- Convert movie review into “stars”
- Survey public opinion for the new iphone Vs old one (SNA)

...

# Text 101

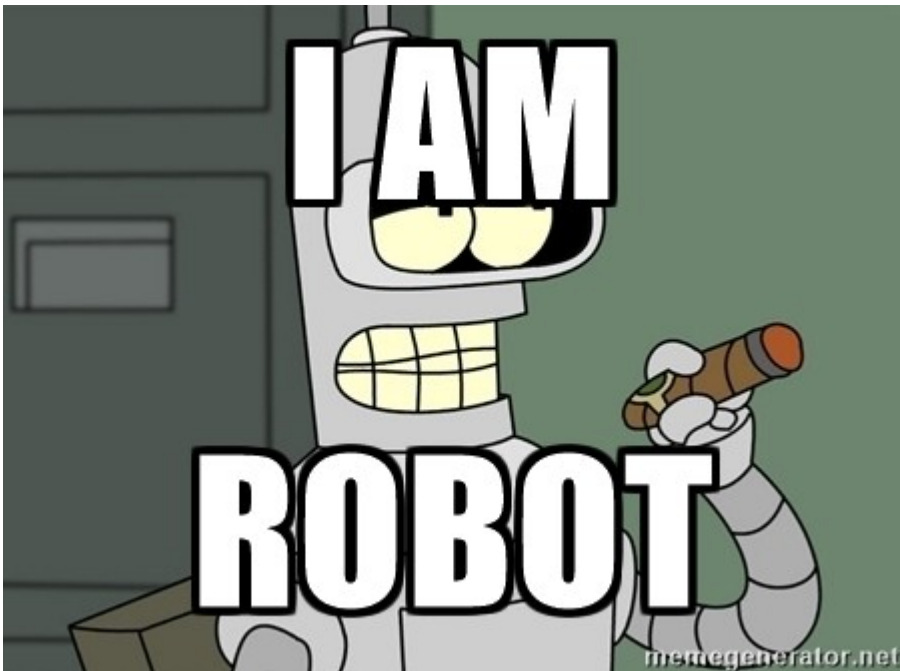
## text

/tɛkst/ 

*noun*

1. a book or other written or printed work, regarded in terms of its content rather than its physical form.  
"a text which explores pain and grief"  
*synonyms:* written work, **book**, **work**, printed work, **narrative**  
"a text which explores pain and grief"
2. the main body of a book or other piece of writing, as distinct from other material such as notes, appendices, and illustrations.  
"the pictures are clear and relate well to the text"  
*synonyms:* words, **wording**; **More**

# Text 101: nlp perspective



## Text:

A sequence of tokens(words).

## Token/word:

A sequence of characters.

## Character:

An atomic element of text.

\_(ツ)\_

# Text 101: tokens

Evolution of the hyaluronan synthase (*has*) operon in *Streptococcus zooepidermicus* and other pathogenic streptococci

↓ Filtering

Evolution of the hyaluronan synthase *has* operon in *Streptococcus zooepidermicus* and other pathogenic streptococci

↓ Tokenization

Evolution

of

the

hyaluronan

synthase

*has*

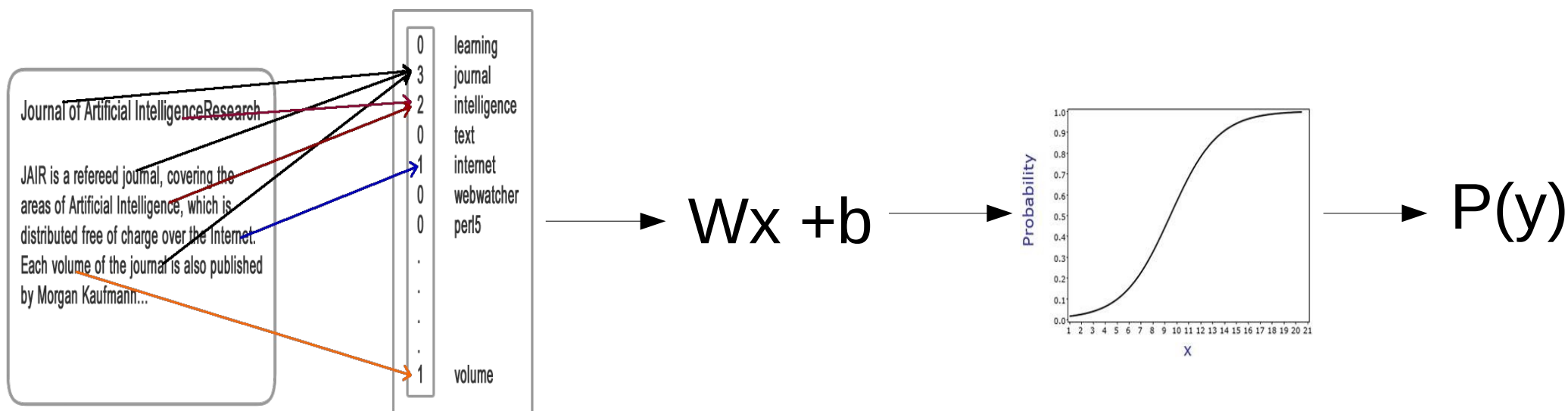
operon

...

# What is a text 101: bag of words



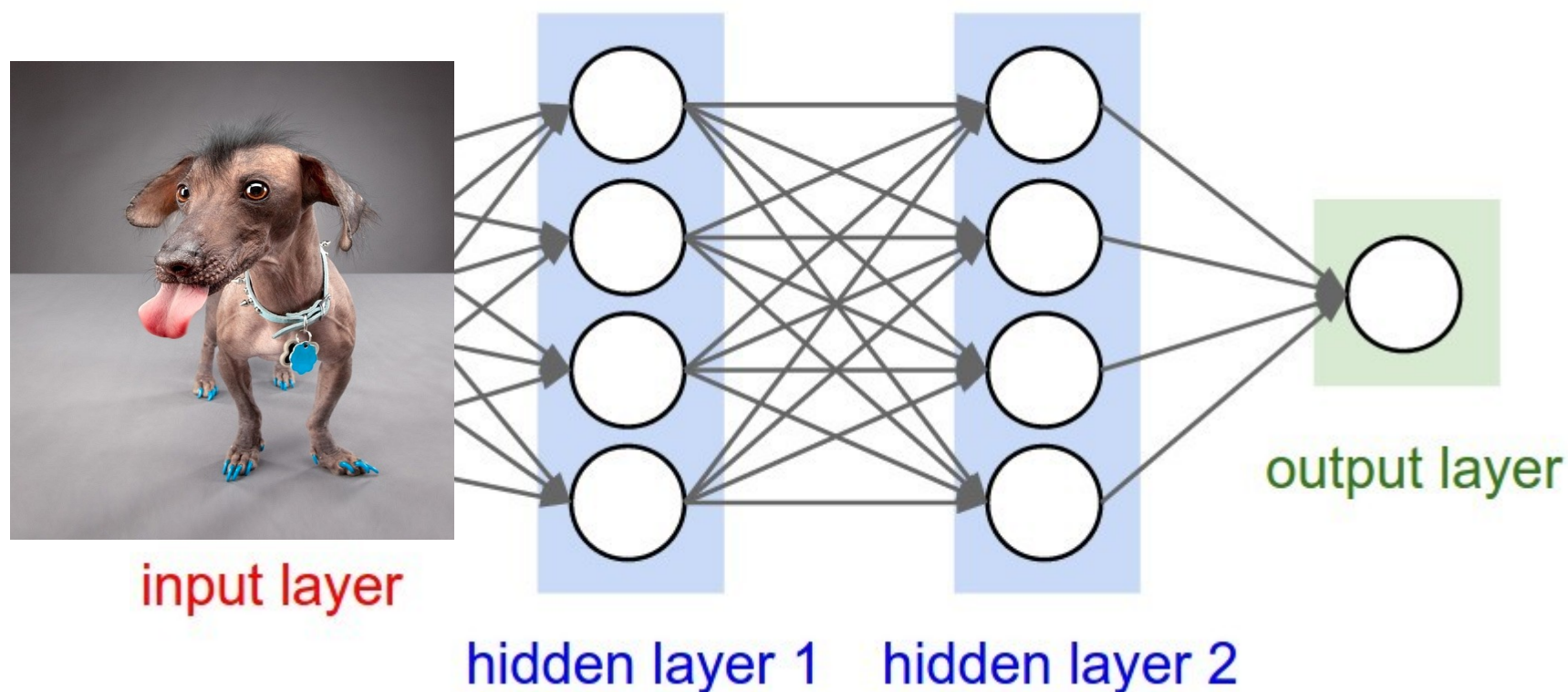
# Text classification: BoW + linear



**Divination: How many features (approx) will such model have?**

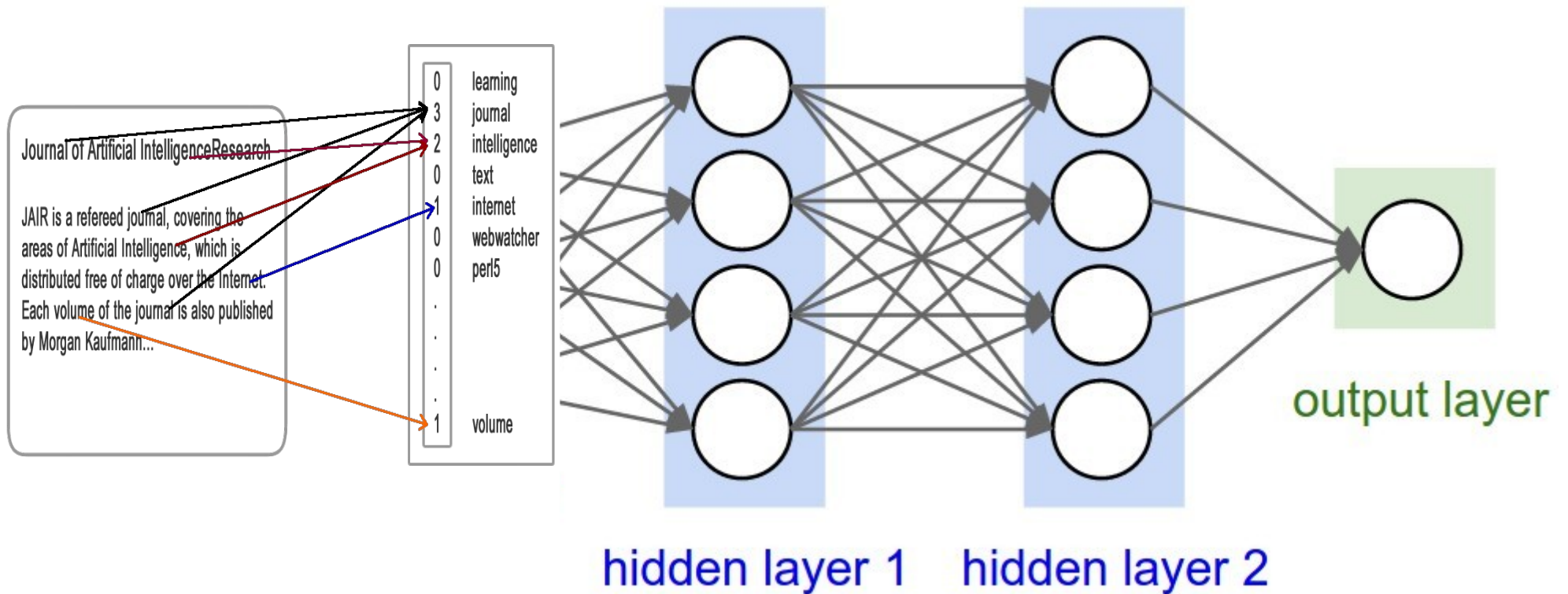


# Text classification: BoW + linear

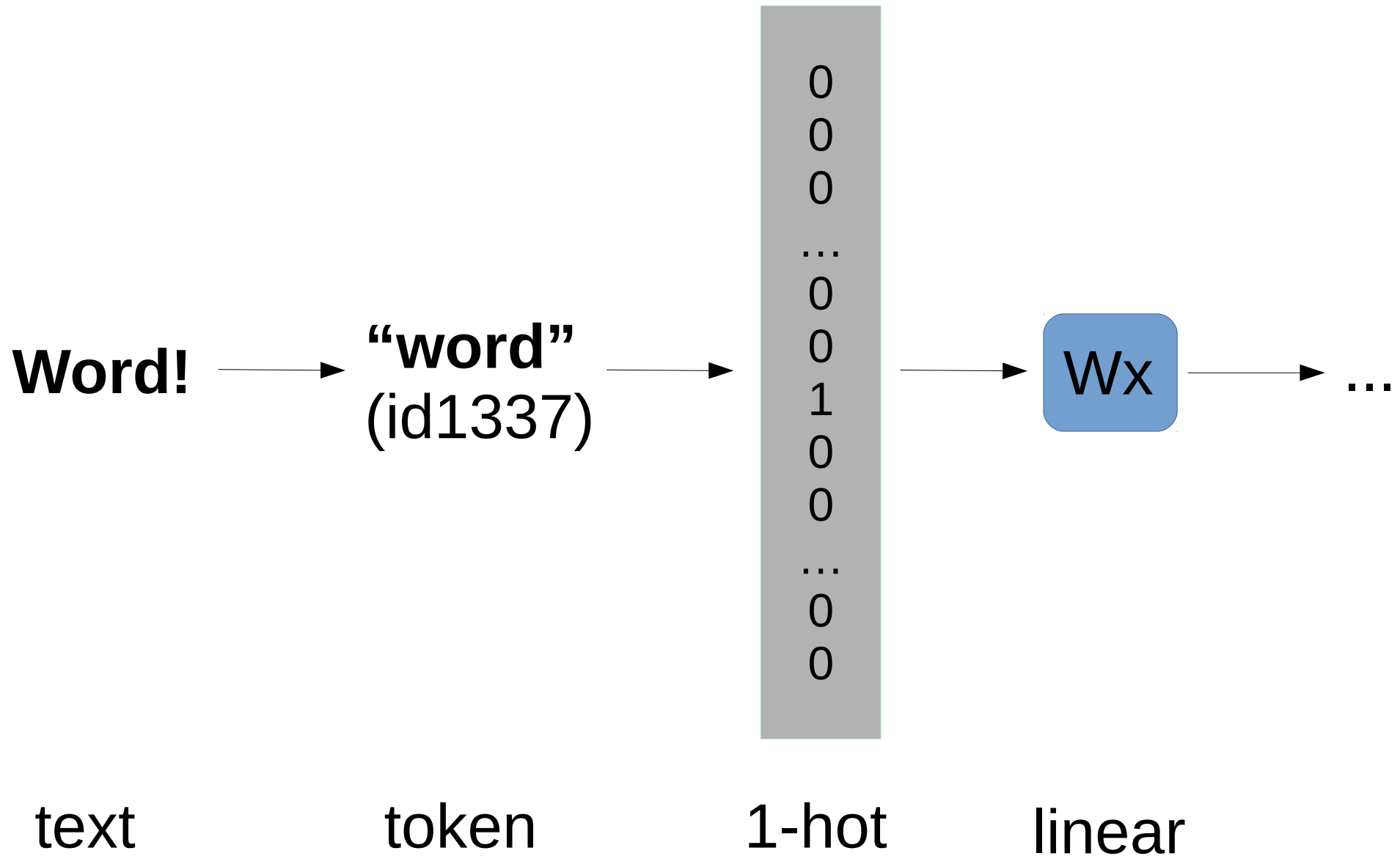


**Oops... wrong slide...**

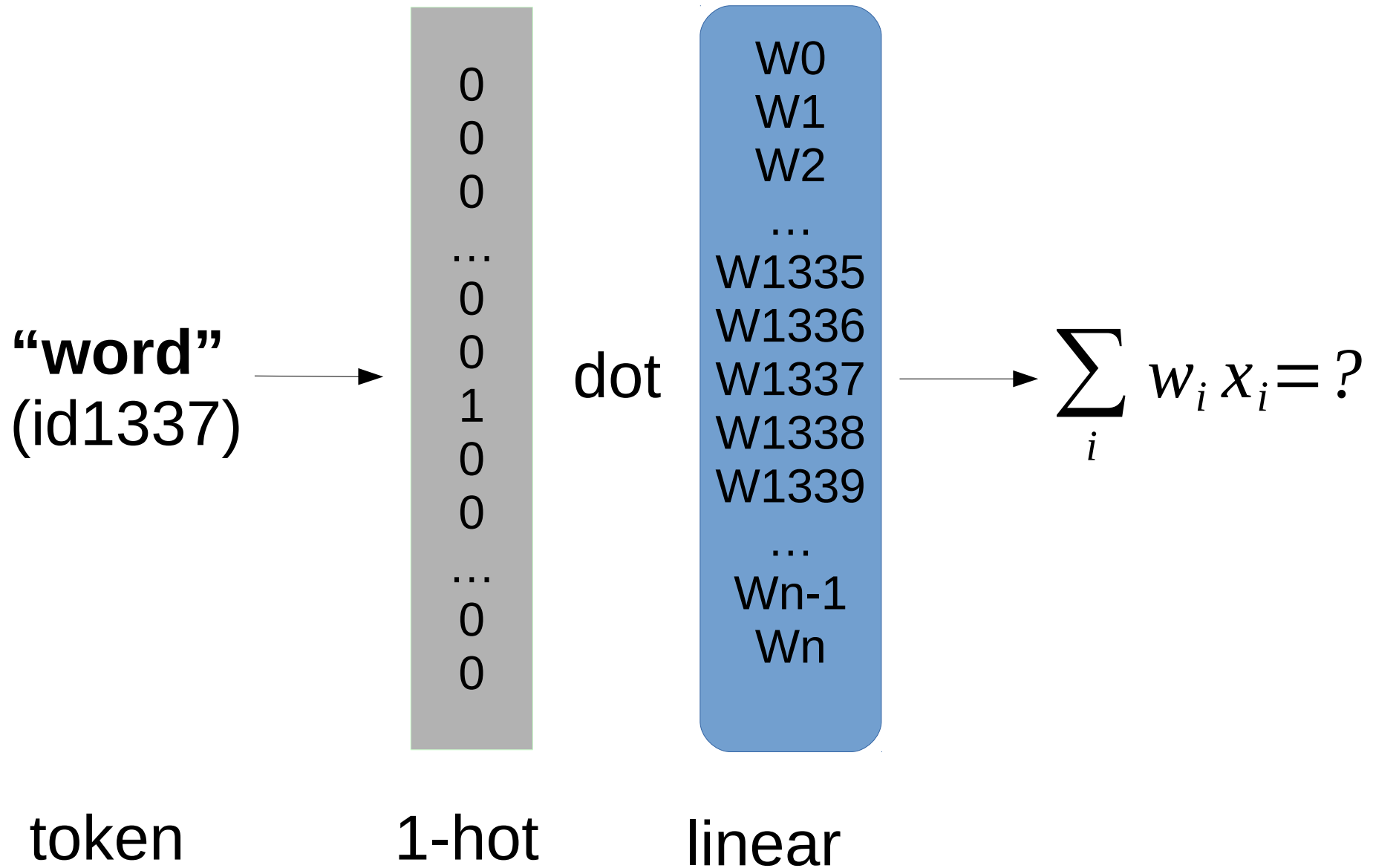
# Text classification: BoW + linear



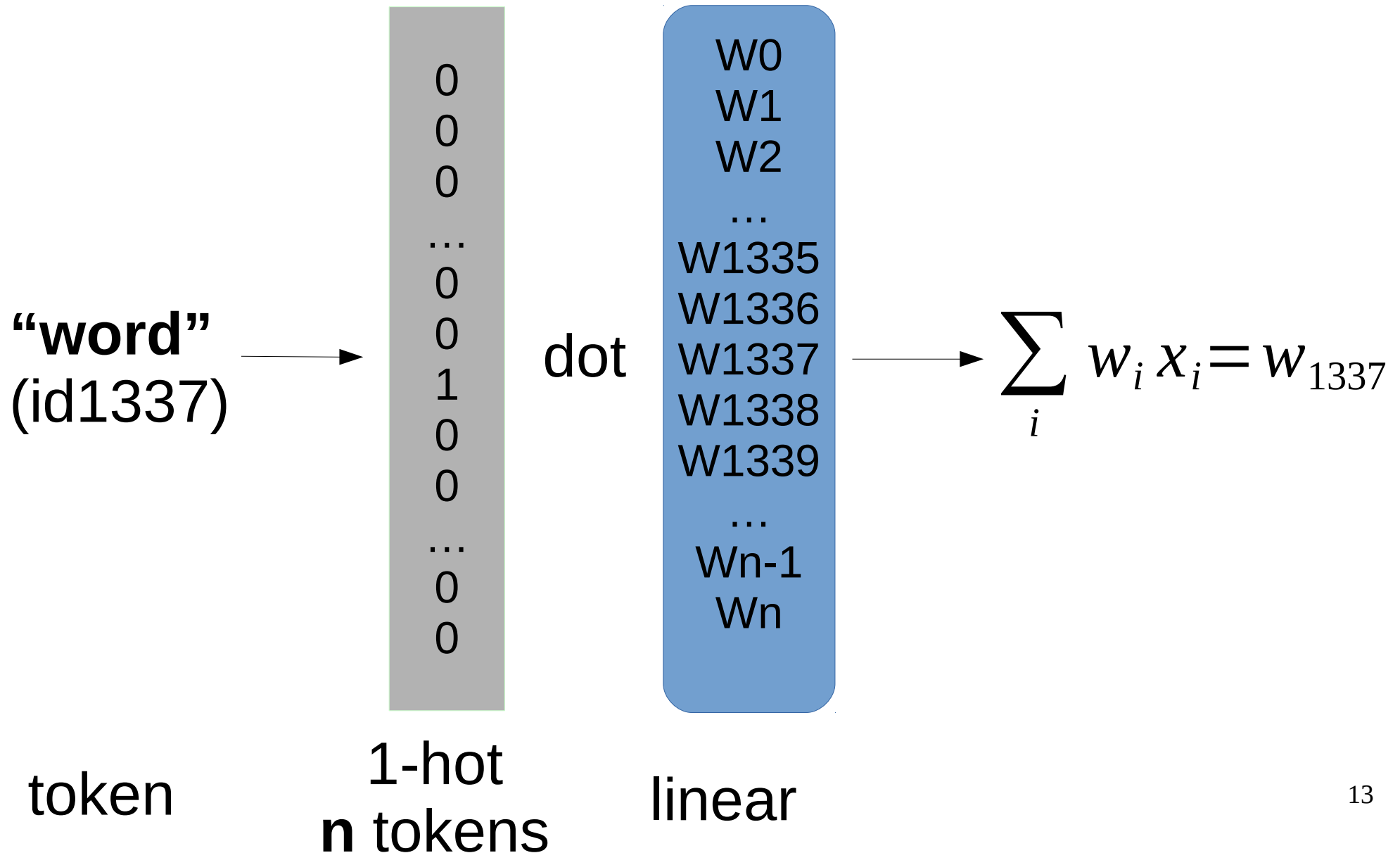
# Sparse vector products



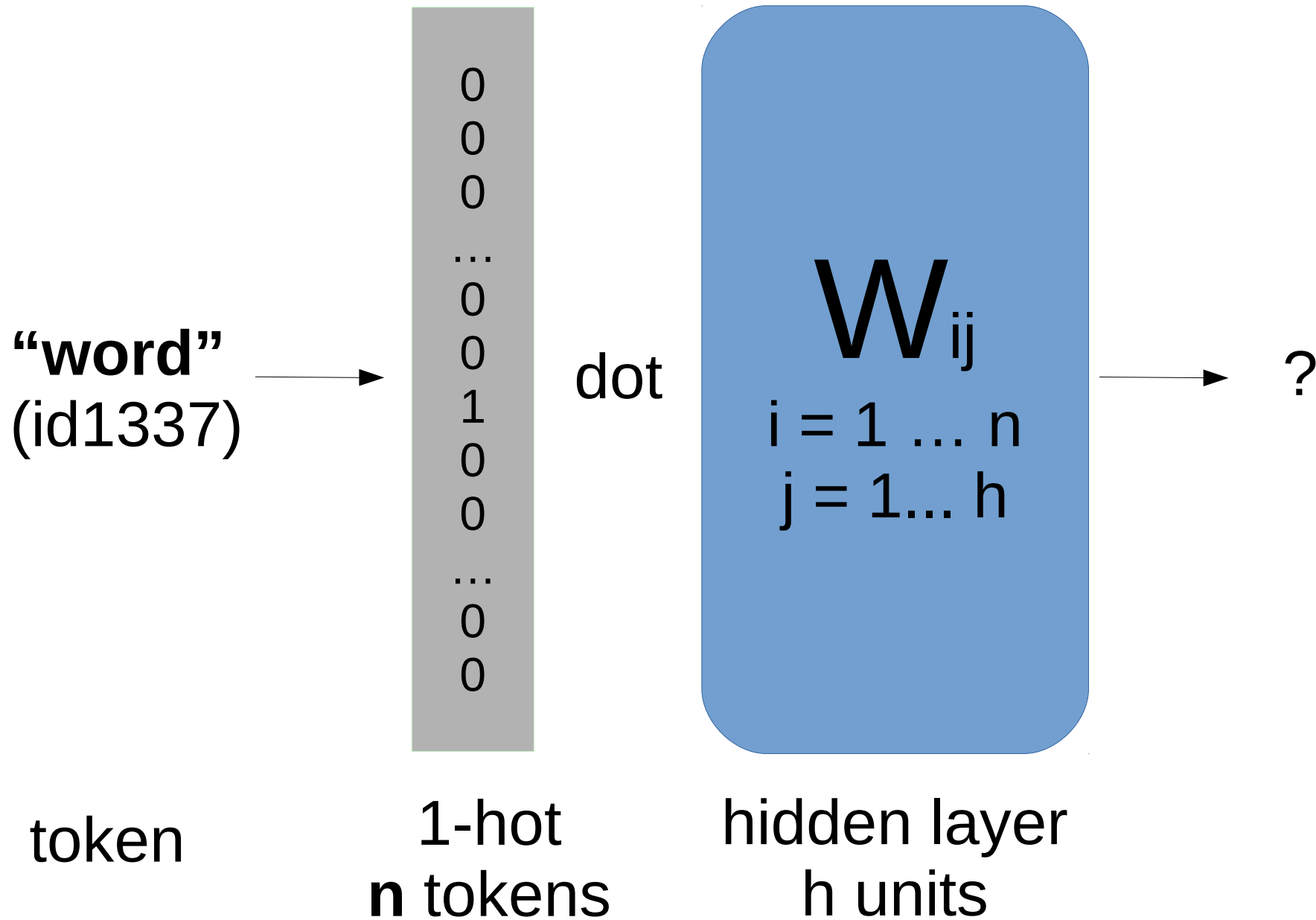
# Sparse vector products



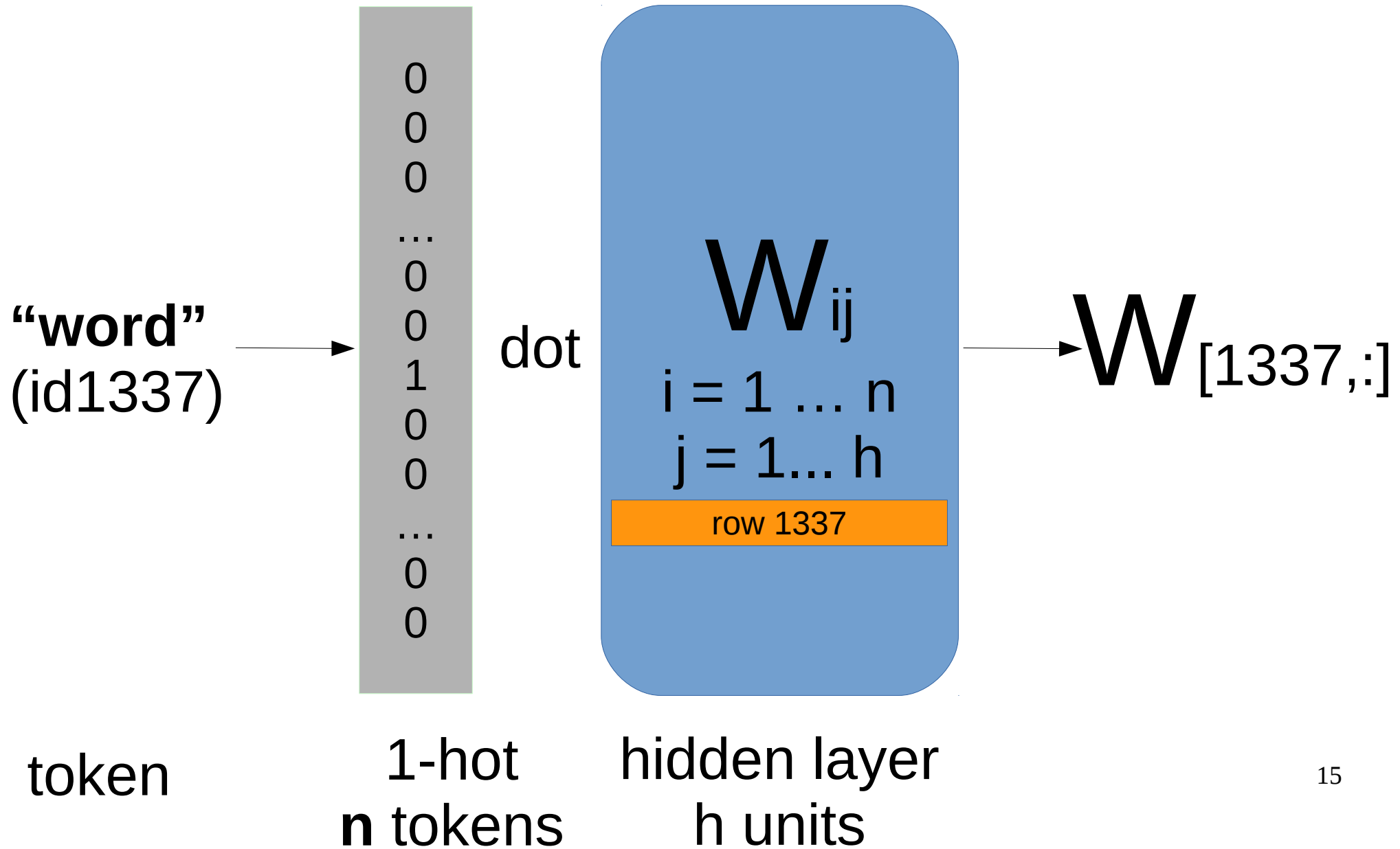
# Sparse vector products



# Sparse vector products

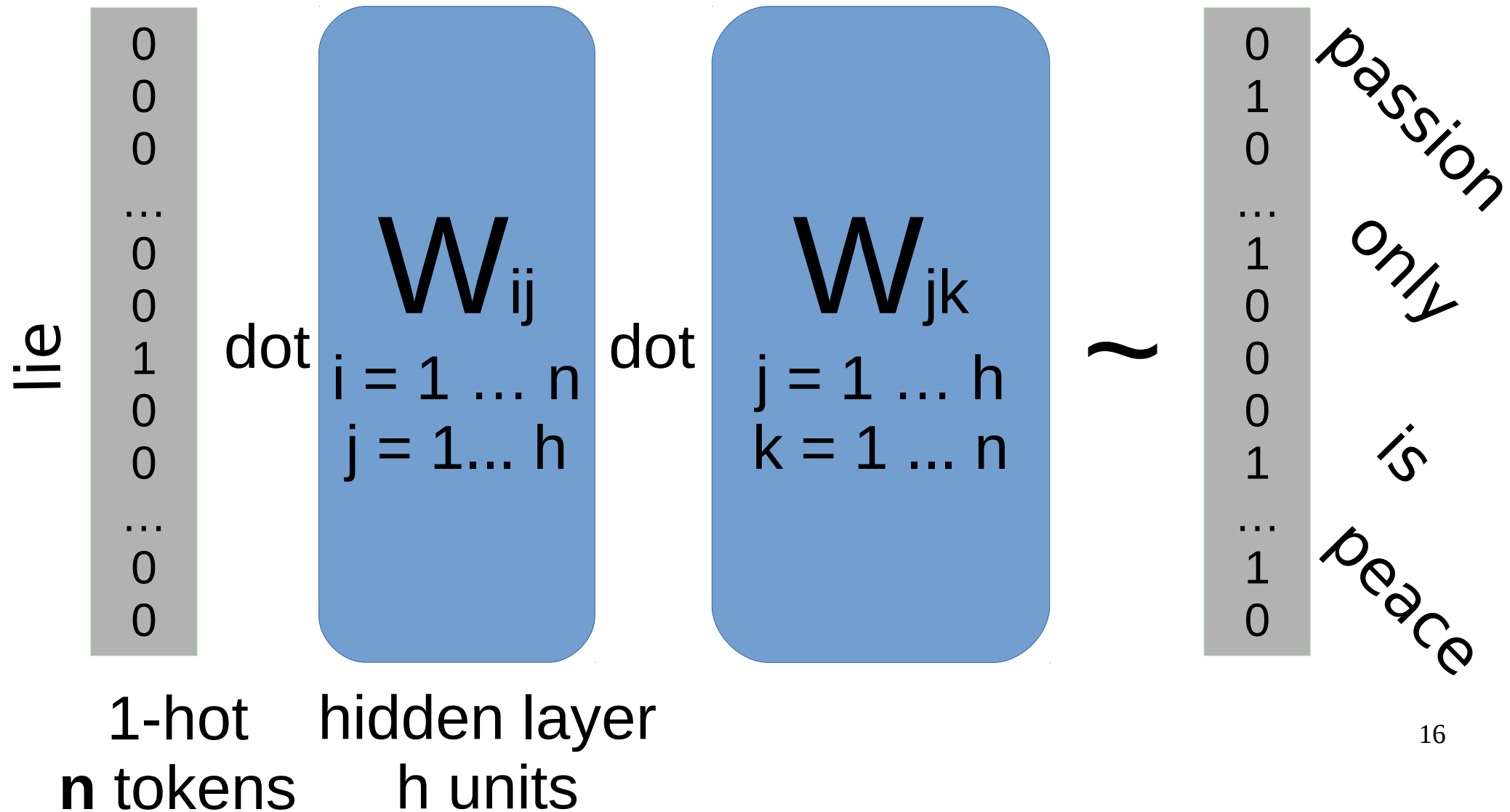


# Embedding



# Embedding: word2vec

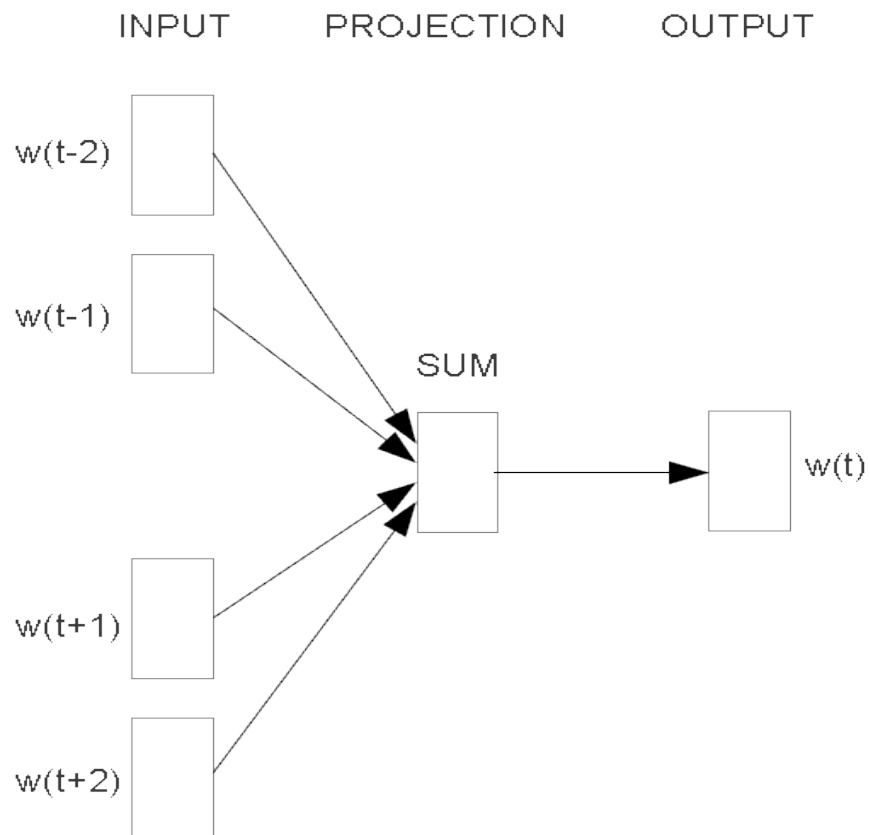
“Peace is a lie, there is only passion”



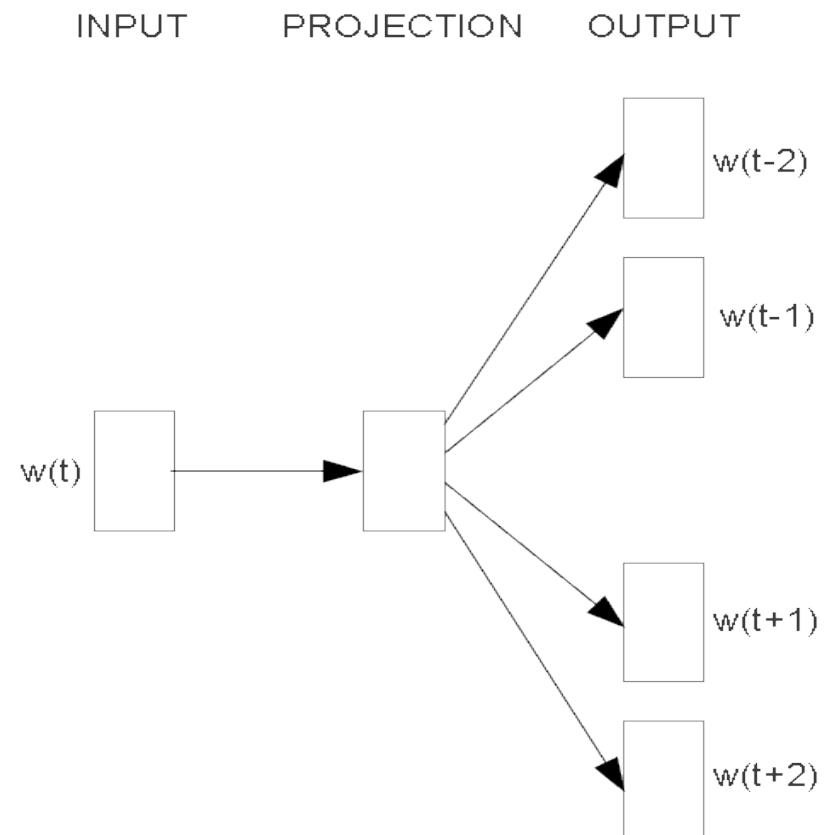


# Embedding: word2vec

**the *distributional hypothesis* : similar context = similar meaning**



**CBOW**



**Skip-gram**

# Embedding: word2vec

## **Side effect: synonyms**

“nice”  $\sim$  “beautiful”

“hard”  $\sim$  “difficult”

## **Side effect: word algebra**

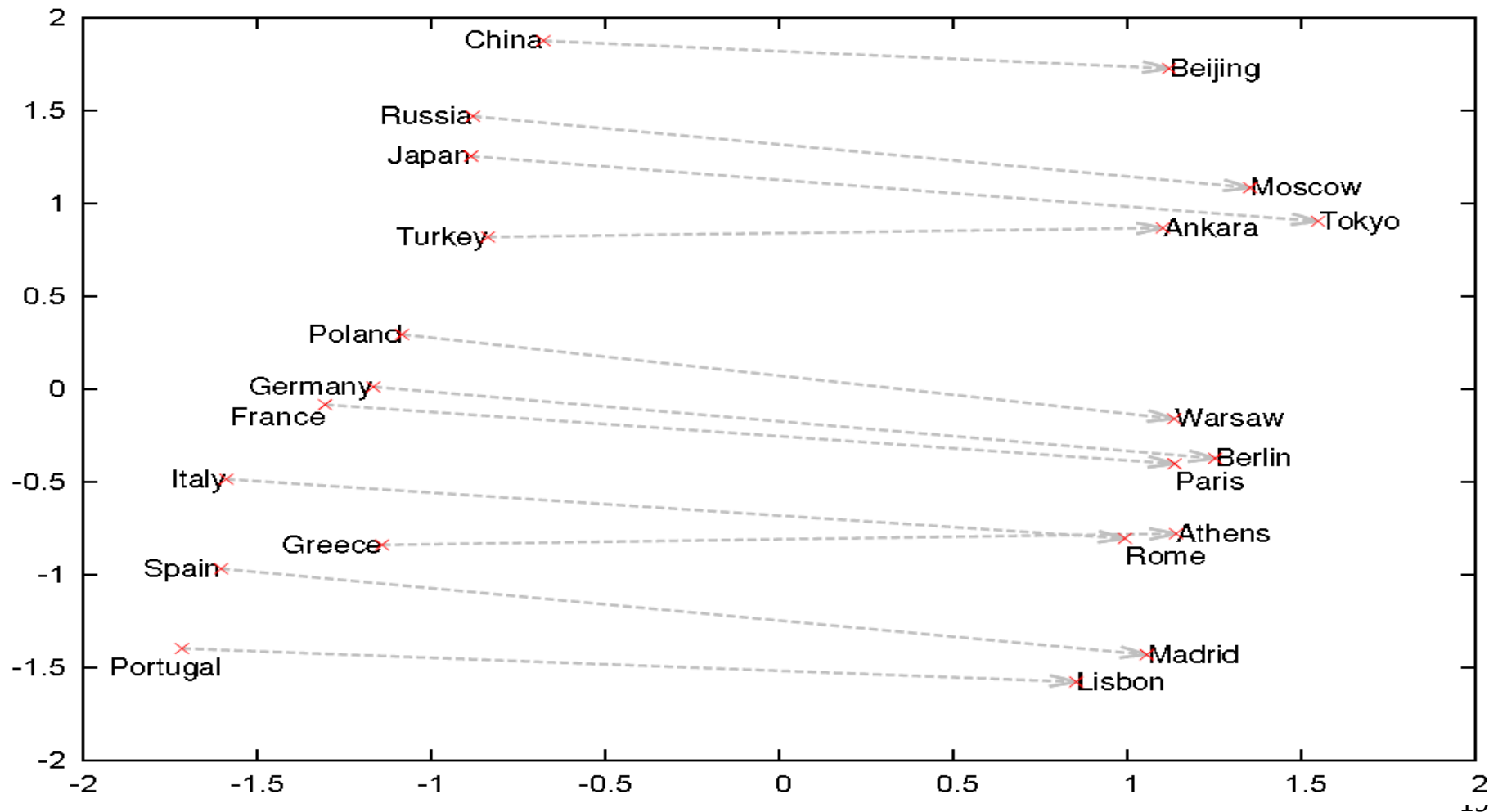
“king” - “man” + “queen”  $\sim$  “queen”

“moscow” - “russia” + “france”  $\sim$  “paris”

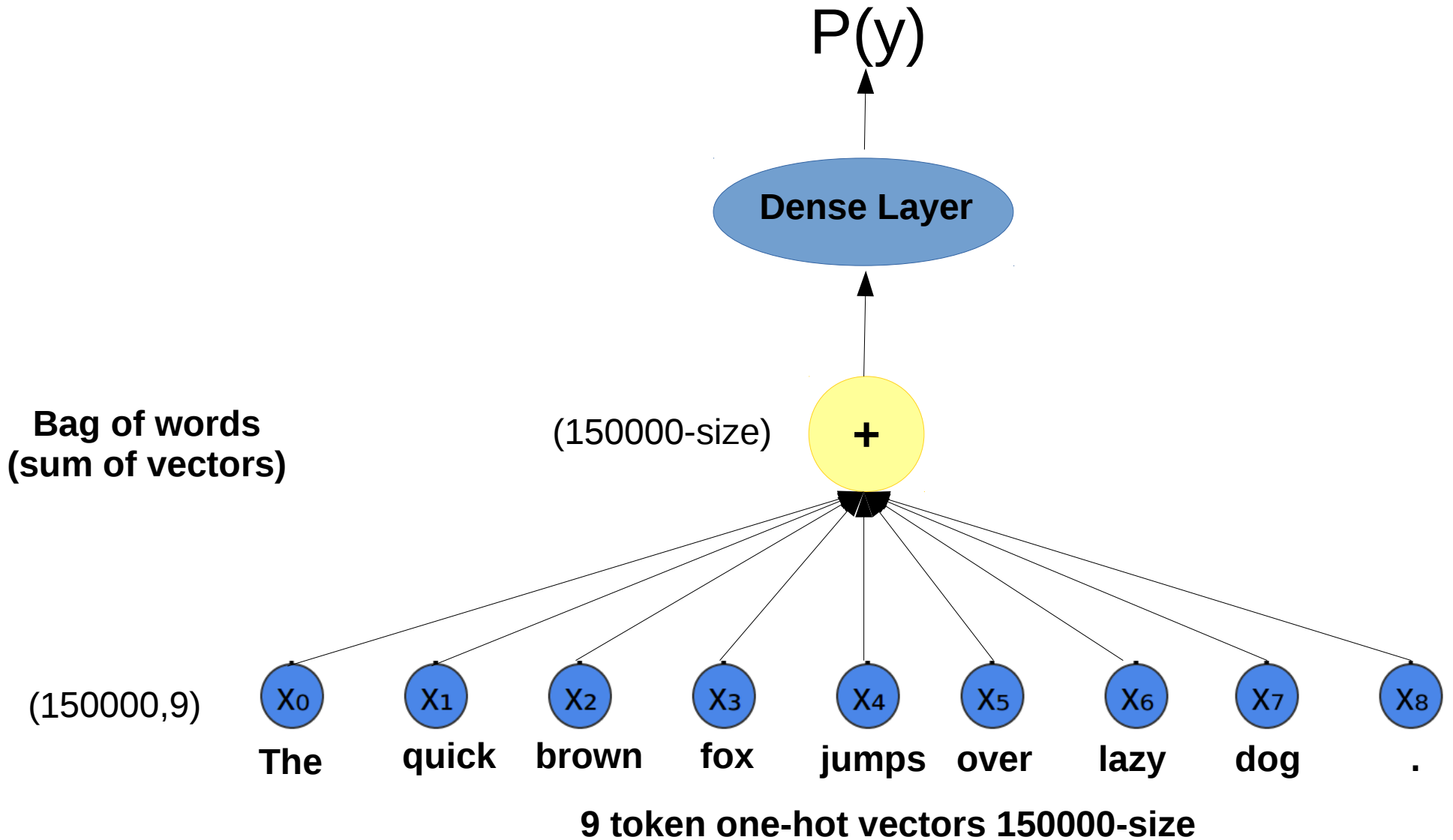
# Embedding: word2vec

## Side effect: word algebra

Country and Capital Vectors Projected by PCA

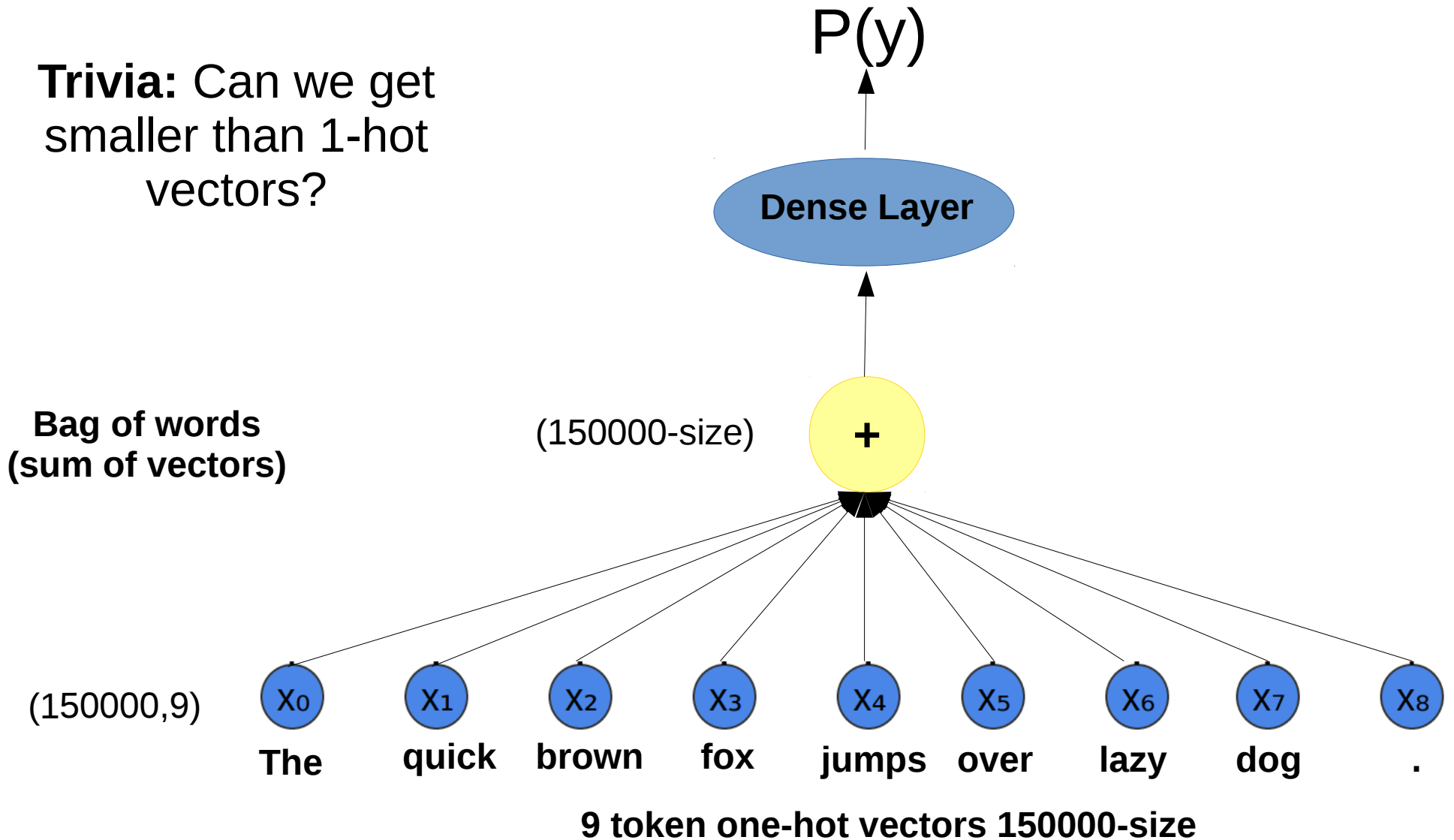


# Text NN



# Text NN

**Trivia:** Can we get smaller than 1-hot vectors?



# Text NN

## Option 1:

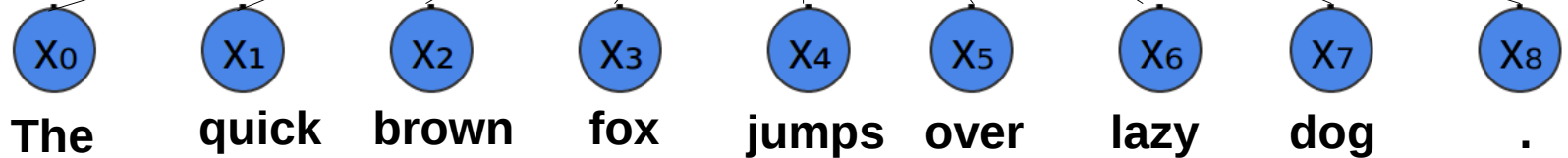
Use pre-trained embedding  
(e.g. glove on wikipedia)

## Option 2:

Train embedding  
(regular backprop)

**Bag of words**  
(sum of vectors)

(100,9)



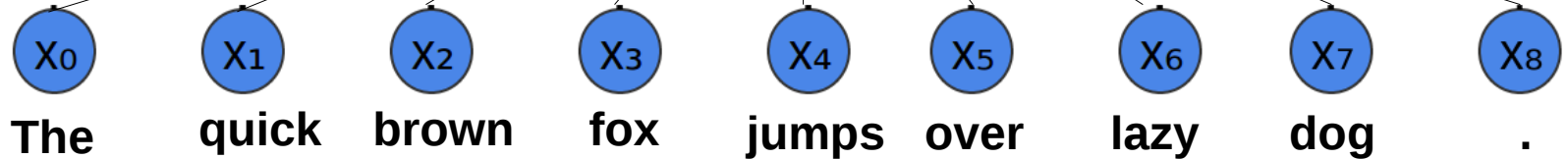
**9 token embedding vectors 100-size**

# Text NN

**Best of both worlds:**  
Start with pre-trained  
embedding and train it

**Bag of words  
(sum of vectors)**

(100,9)



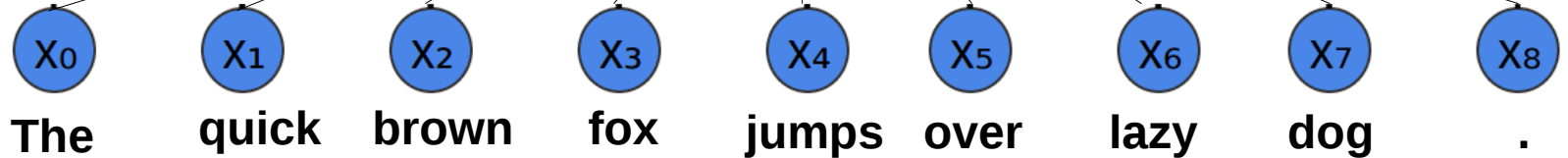
**9 token embedding vectors 100-size**

# Text NN

**Trivia:** How to change the bag of words operation to compute not “how many such words are there” but “is there at least one such word”

Bag of words  
(sum of vectors)

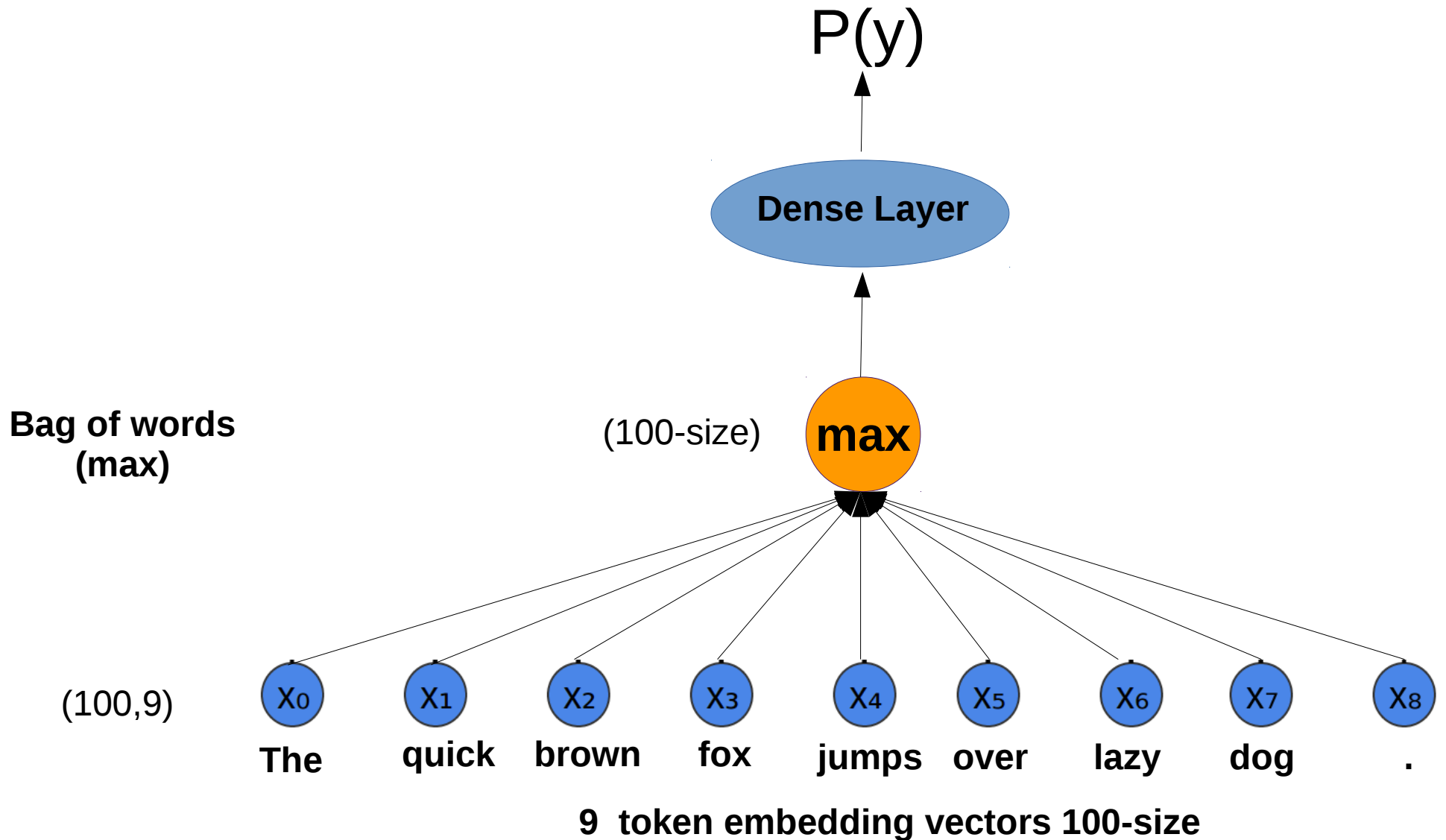
(100,9)



9 token embedding vectors 100-size



# Text NN



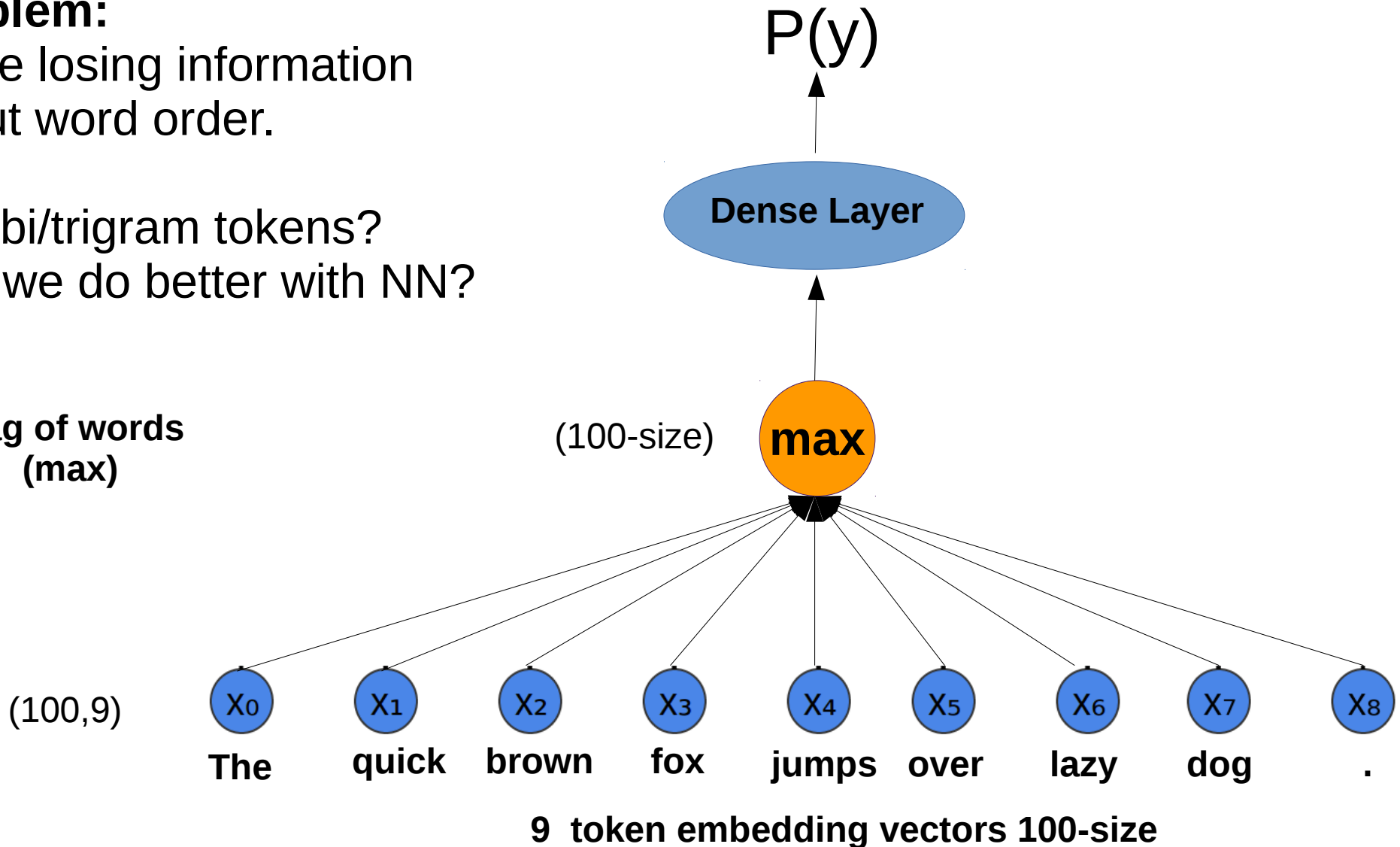
# Text NN

## Problem:

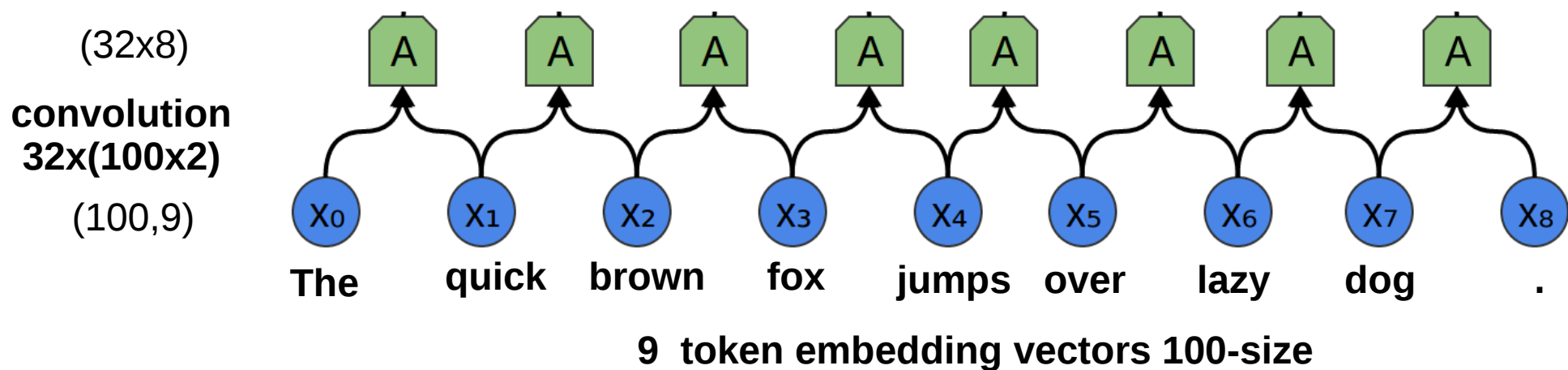
We're losing information about word order.

Use bi/trigram tokens?  
Can we do better with NN?

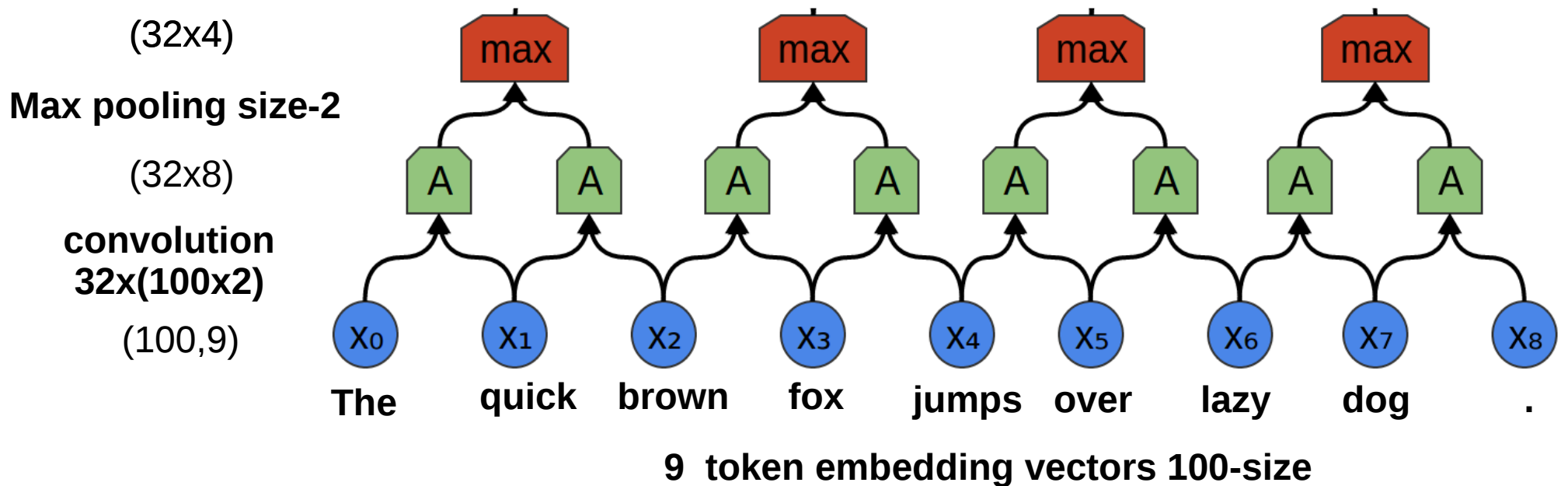
Bag of words  
(max)



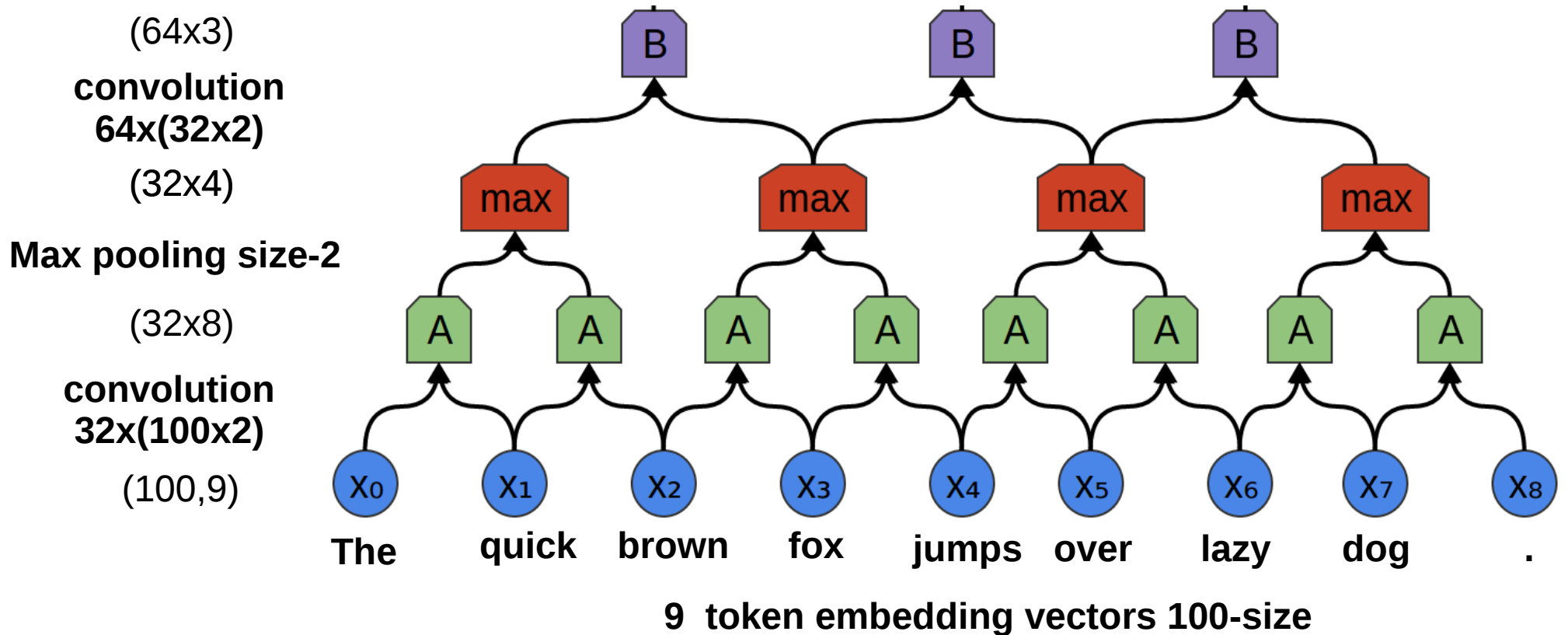
# Text CNN



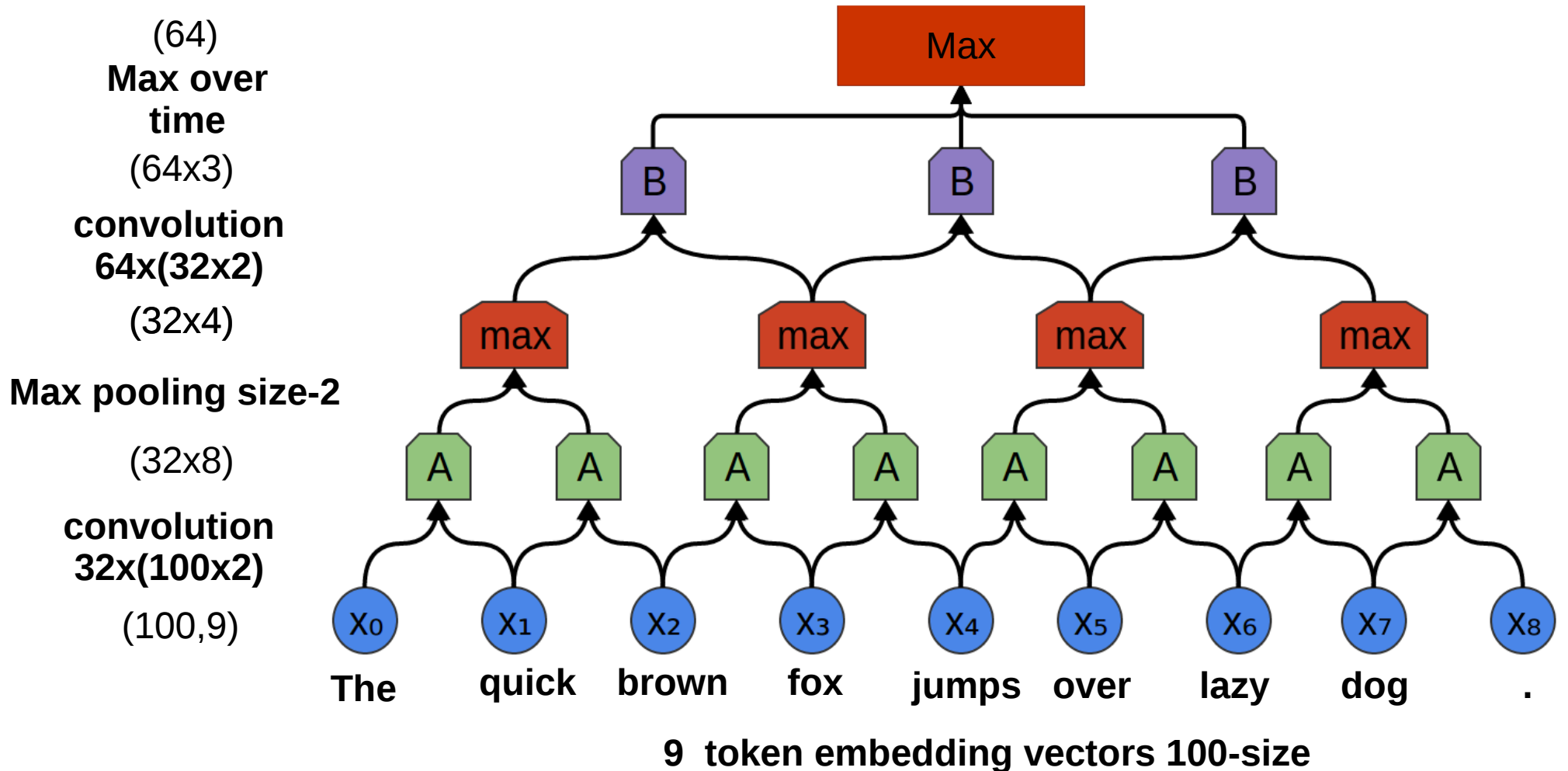
# Text CNN



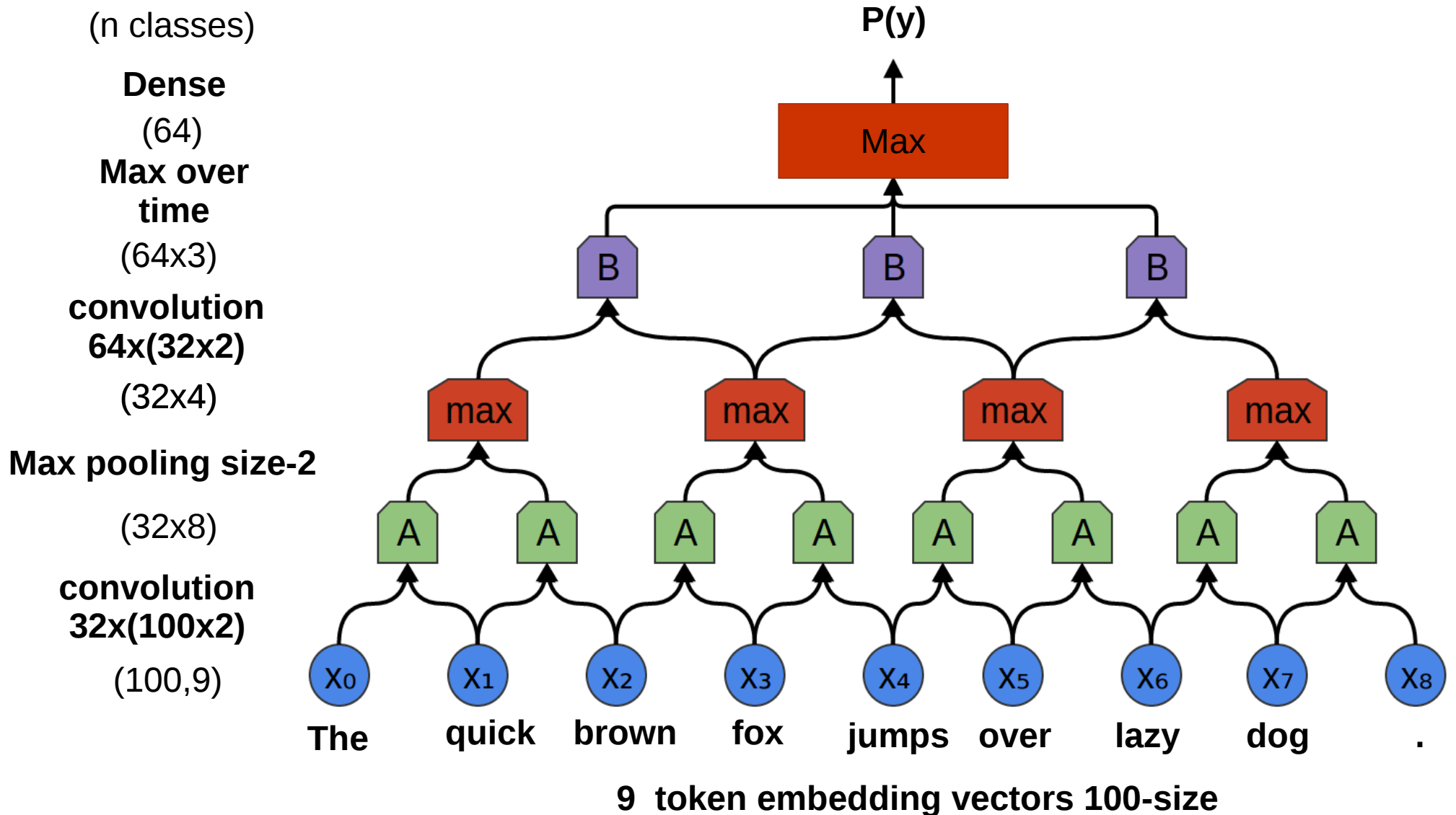
# Text CNN



# Text CNN

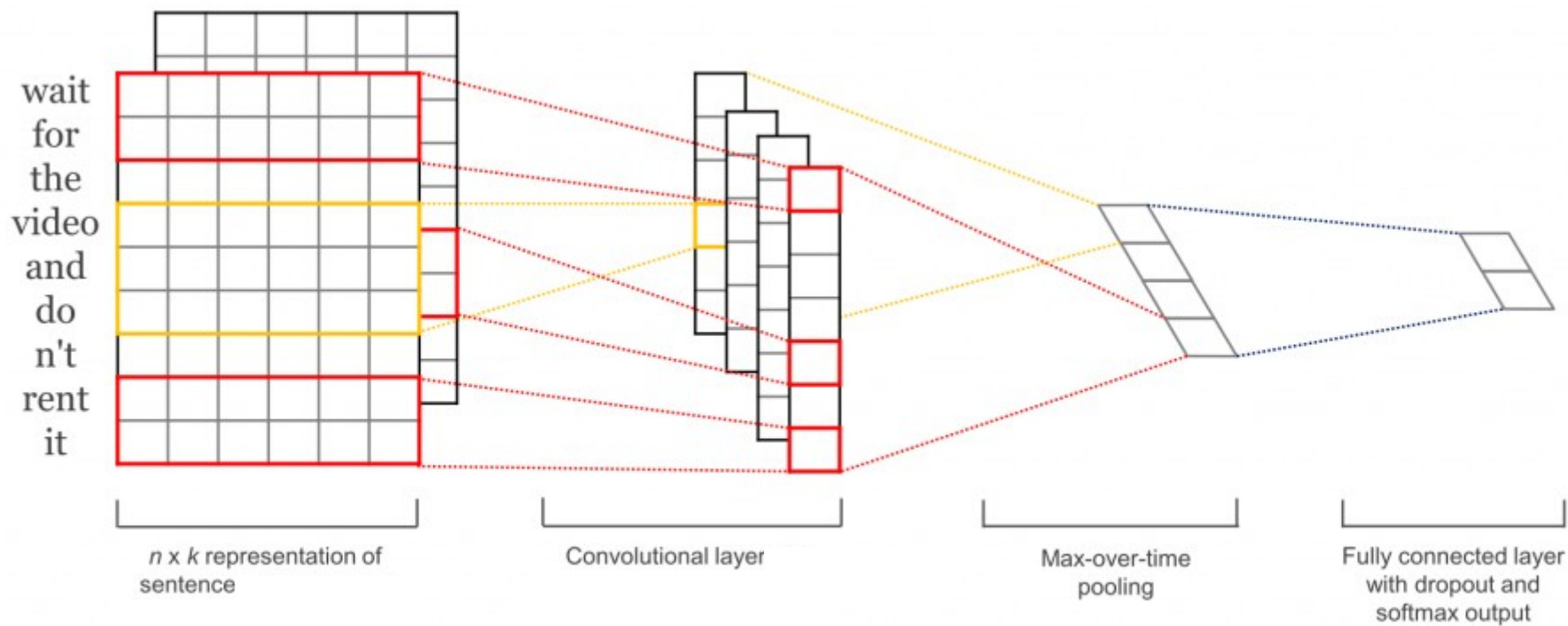


# Text CNN



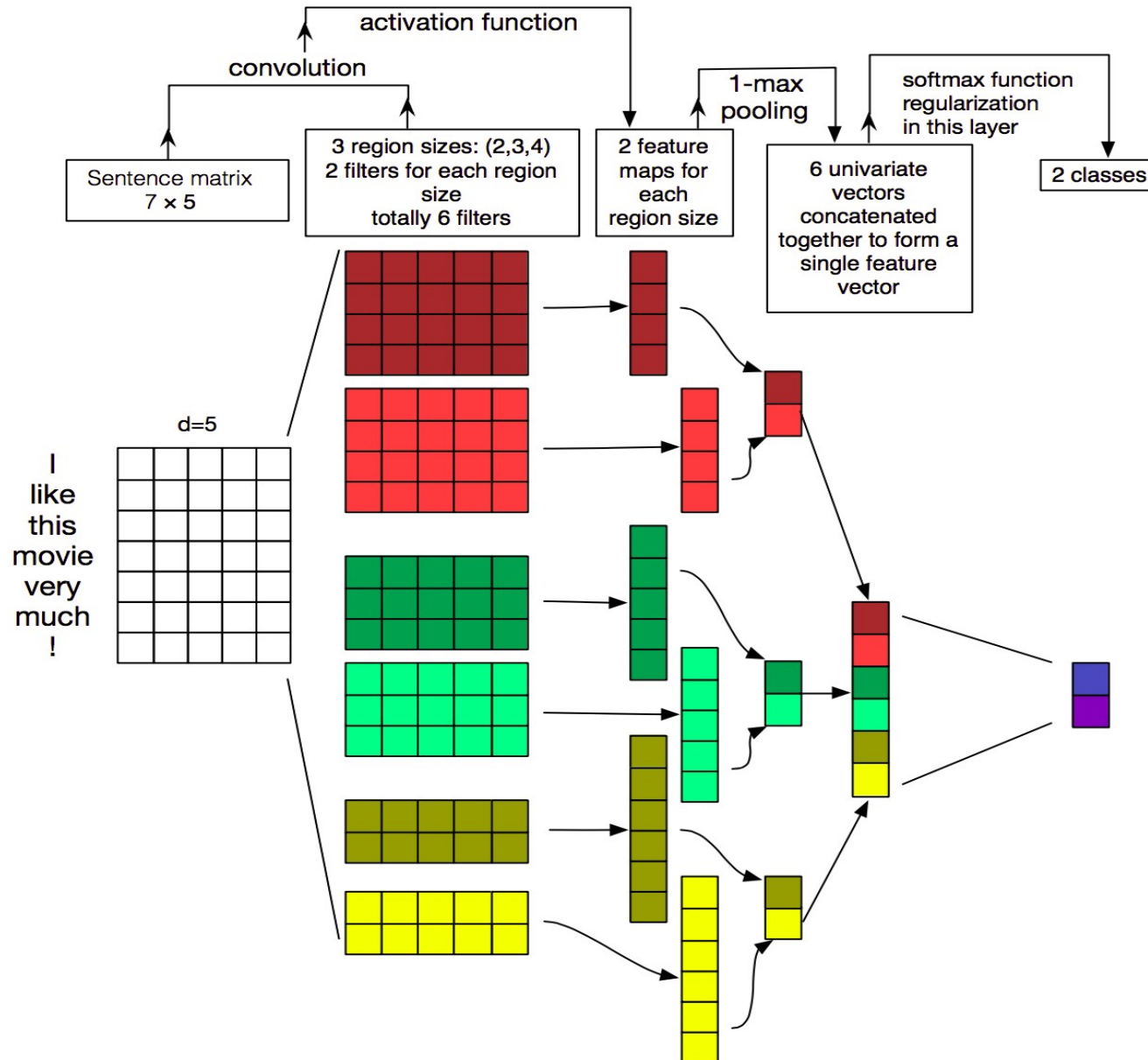
# Text CNN

1-hot/emb





# Text CNN: monsters



# Text CNN: monsters

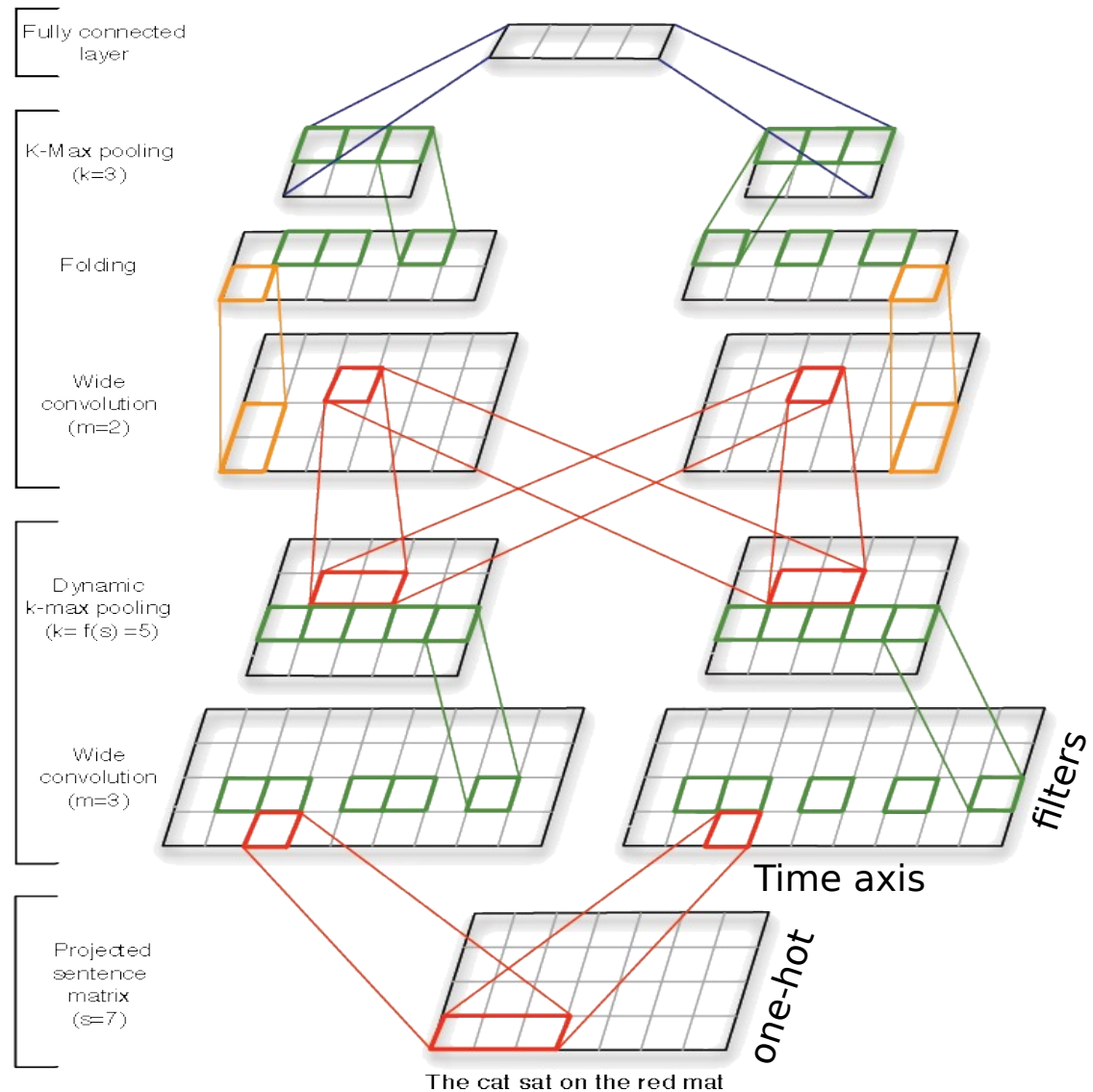
Dense layer

“Folding” = sum pooling along embedding direction

“K-max pooling” = take not 1 but k highest activations in their original order.  
E.g. (0,1,3,2,0,1,4,1) → (3,2,4)

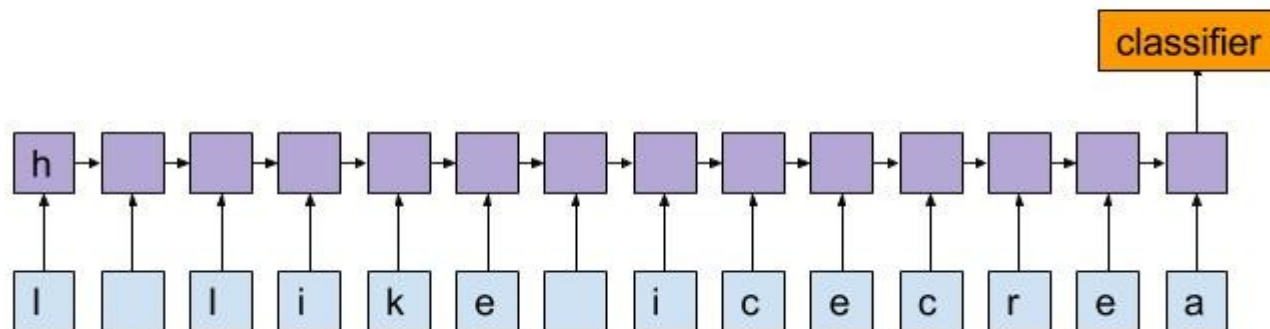
Wide convolution is convolution with full padding

Each word encoded with a vector (e.g. “one-hot”).



Other NLP problems  
(coming soon)

# Language model

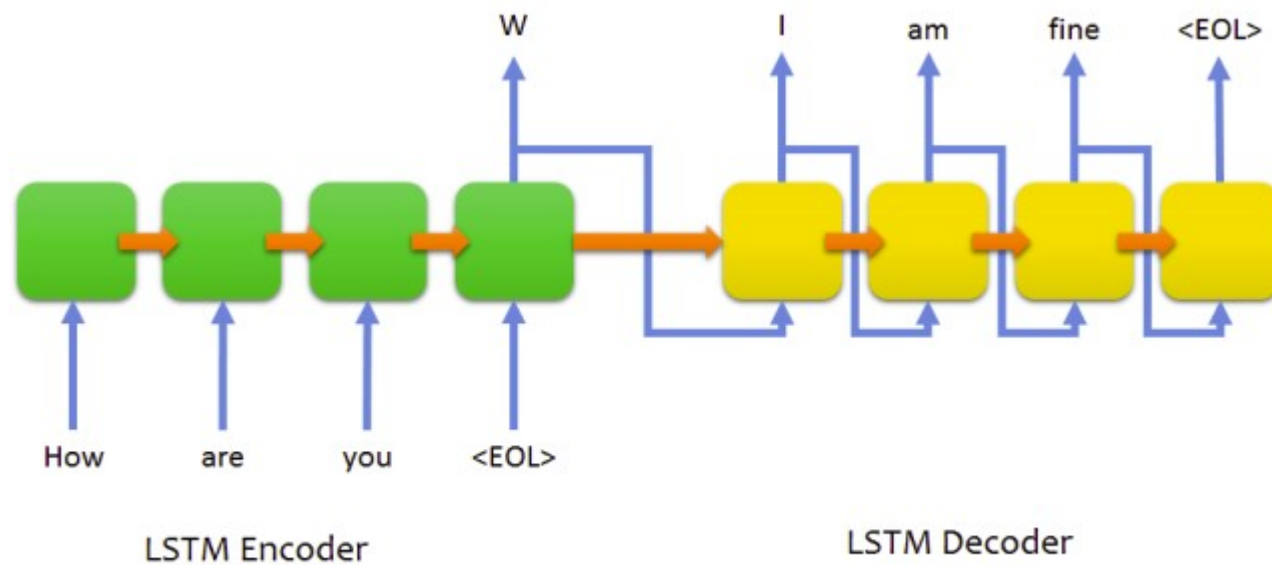


# POS tagging

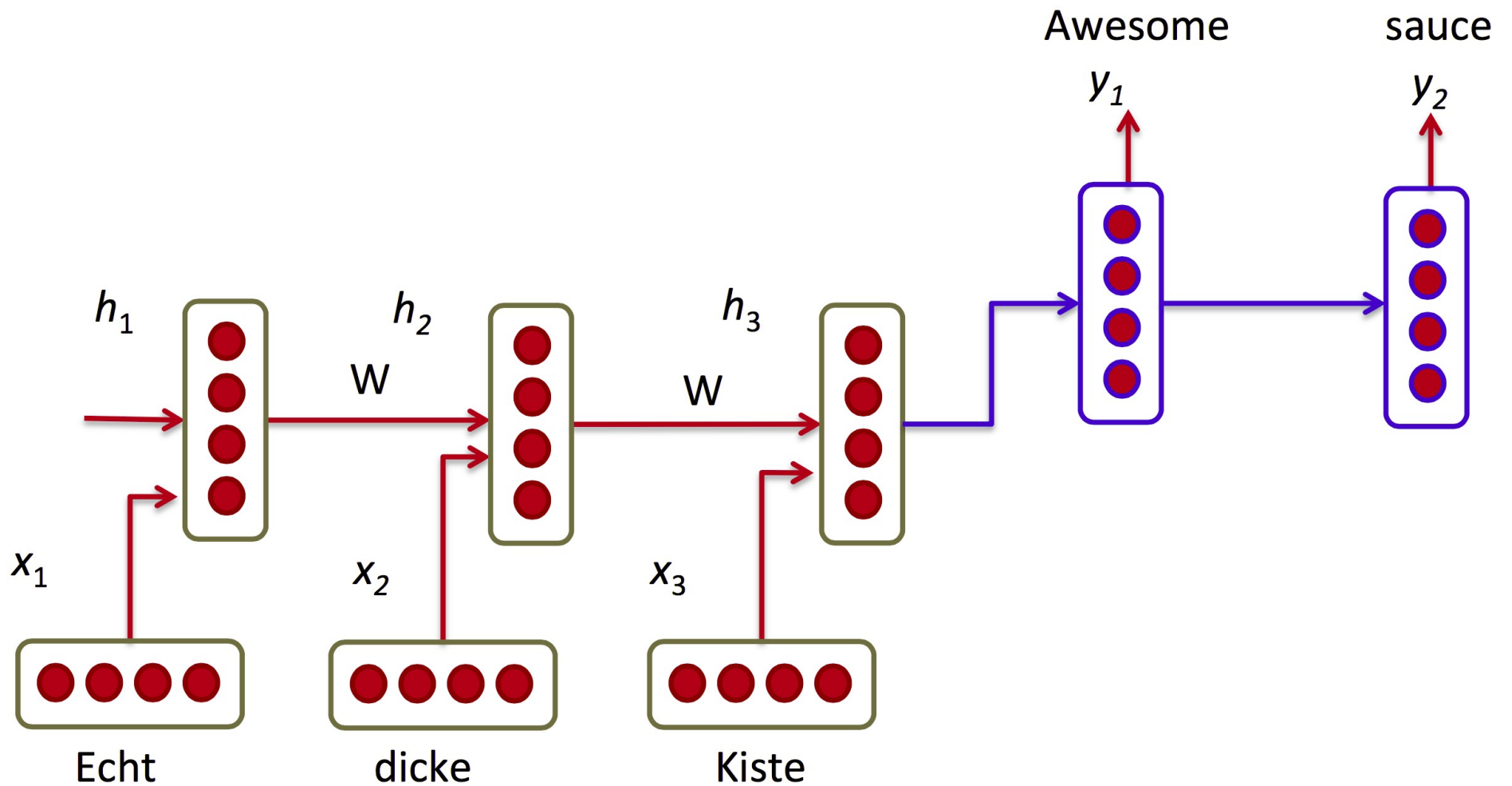
DT/ The JJ/ quick JJ/ brown NN/ fox NNS/ jumps IN/ over DT/ the JJ/ lazy NN/ dog

- -RRB- - Right bracket
- CD - Cardinal number
- EX - Existential there
- IN - Preposition
- JJR - Comparative adjective
- LS - List Item Marker
- NN - Singular noun
- NNP - Proper singular noun
- PDT - Predeterminer
- PRP - Personal pronoun
- RB - Adverb
- RBS - Superlative Adverb
- SYM - Symbol
- UH - Interjection
- VBD - Verb, past tense
- VBN - Verb, past participle
- VBZ - Verb, 3rd ps. sing. present
- WP - wh-pronoun
- WRB - wh-adverb
- CC - Coordinating conjunction
- DT - Determiner
- FW - Foreign word
- JJ - Adjective
- JJS - Superlative adjective
- MD - Modal
- NNS - Plural noun
- NNPS - Proper plural noun
- POS - Possessive ending
- PP\$ - Possessive pronoun
- RBR - Comparative adverb
- RP - Particle
- TO - to
- VB - Verb, base form
- VBG - Verb, gerund/present participle
- VBP - Verb, non 3rd ps. sing. present
- WDT - wh-determiner
- WP\$ - Possessive wh-pronoun

# Conversation model



# Machine translation



# Nuff

Neural network won't write itself without you!  
(except it will <https://arxiv.org/abs/1511.06279> )