Word Embeddings and A Very Simple Word Embedding Based Model

June 22, 2019

Block 3, Lecture 2
Applied Data Science
MMCi Term 4, 2019

Matthew Engelhard

-gram[1]

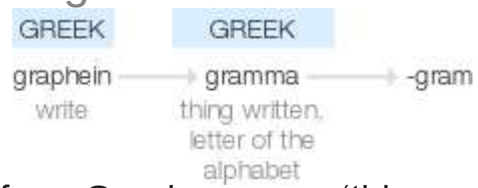
combining form

suffix: **-gram**

1.in nouns denoting something written or recorded (especially in a certain way).

"cryptogram"

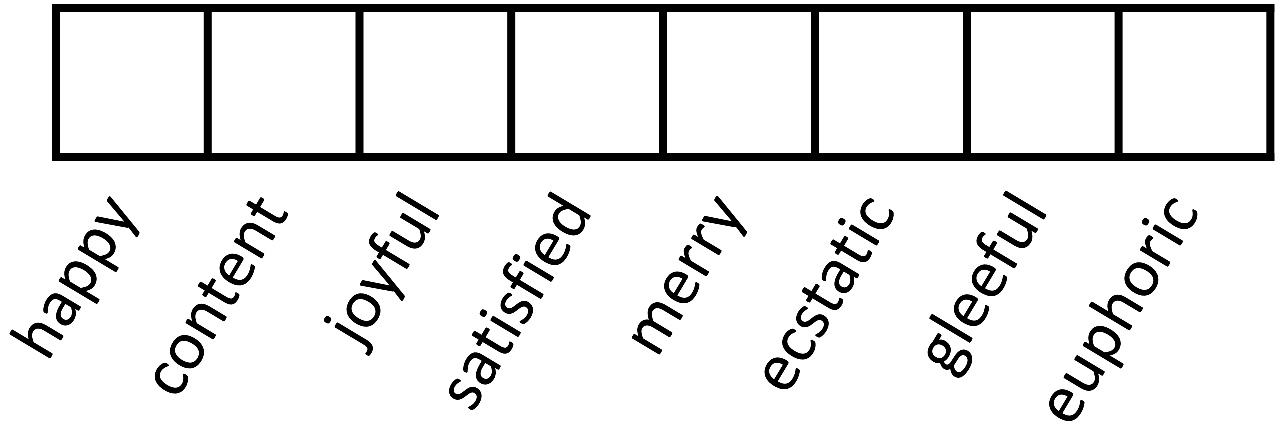
Origin



from Greek *gramma* ‘thing written, letter of the alphabet’, from *graphein* ‘write’.

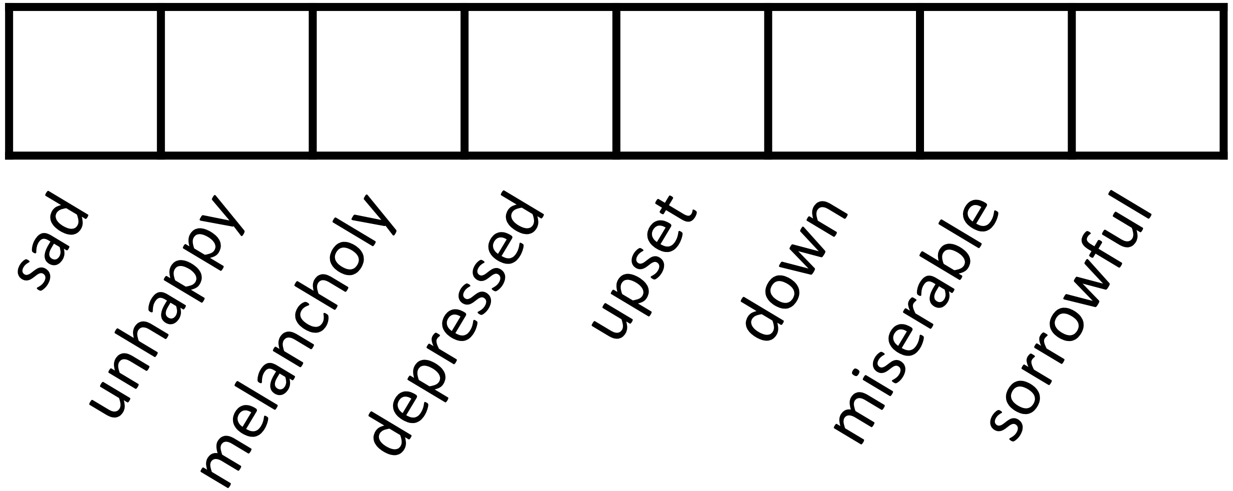
MOTIVATING WORD EMBEDDINGS

Problem: our model counts words, but has no understanding of their meaning



Goal: predict sentiment (positive/negative

Problem: our model counts words, but has no understanding of their meaning



Goal: predict sentiment (positive/negative)

To effectively predict sentiment, it would be helpful to understand which words have similar meaning

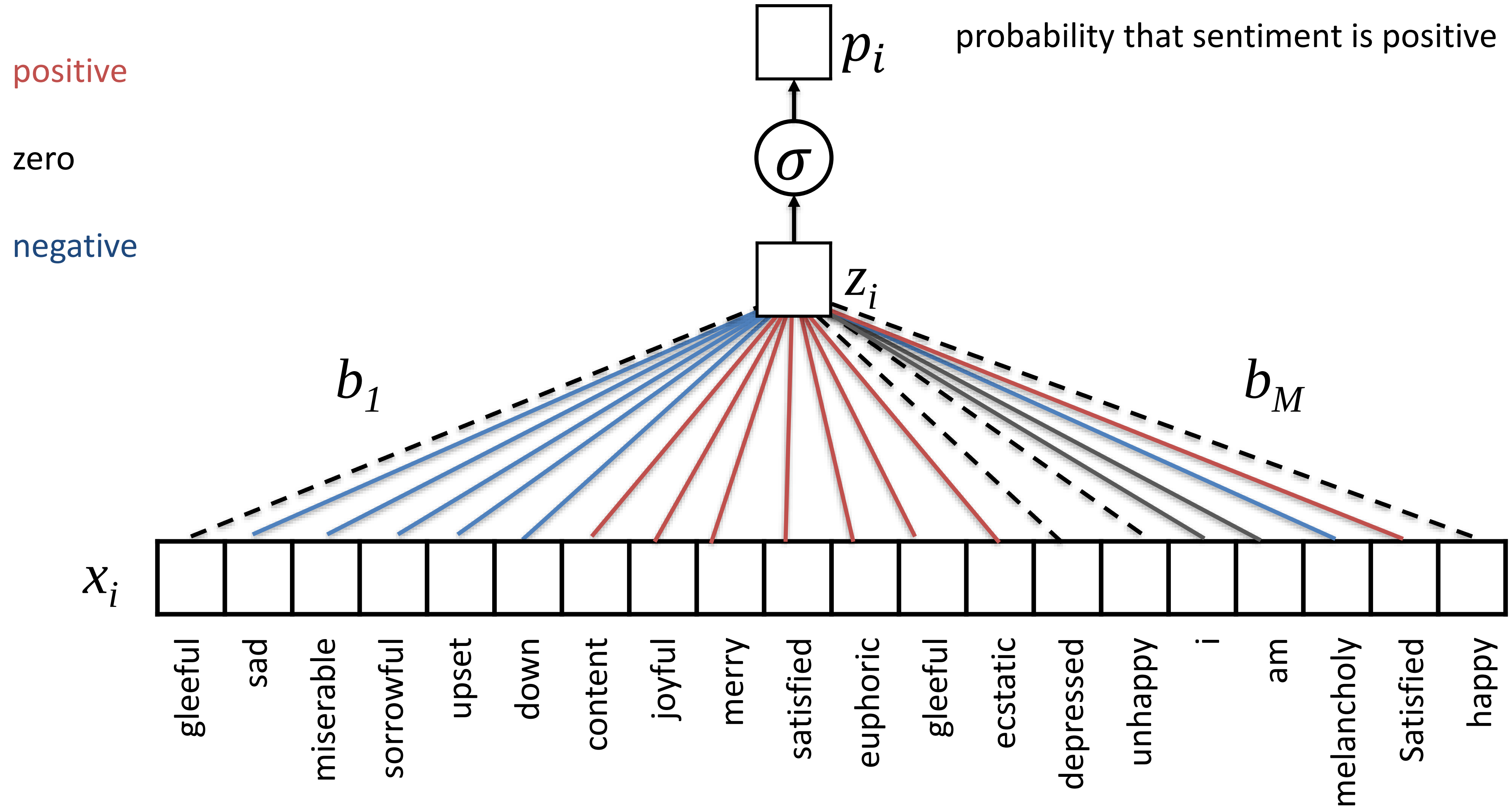
I am sad ☹
I am miserable ☹
I am sorrowful ☹
I am upset ☹
I am down ☹
I am content ☺
I am joyful ☺
I am merry ☺
I am satisfied ☺
I am euphoric ☺

training set

I am depressed ?
I am unhappy ?
I am happy ?
I am gleeful ?

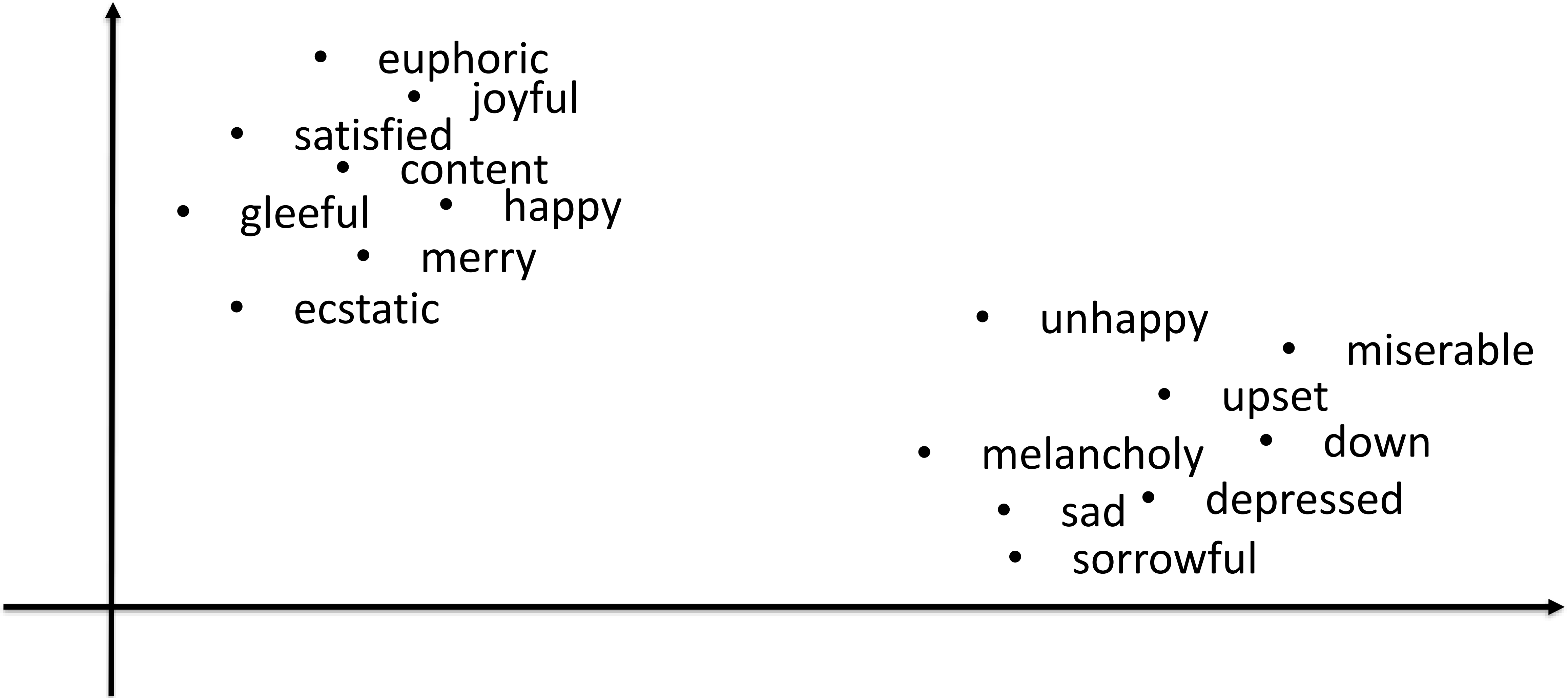
test set

logistic regression: positive / negative sentiment

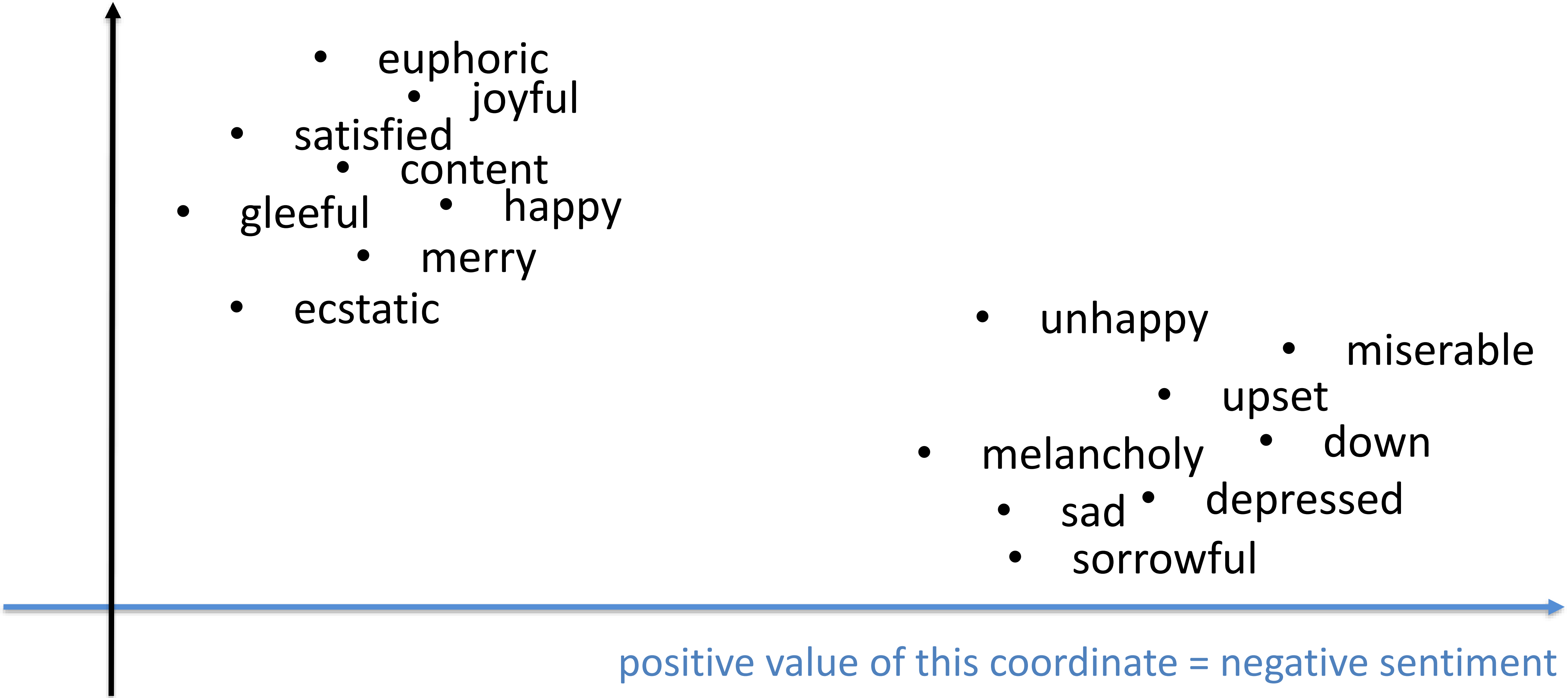


I passed out and Mom said I was shaking

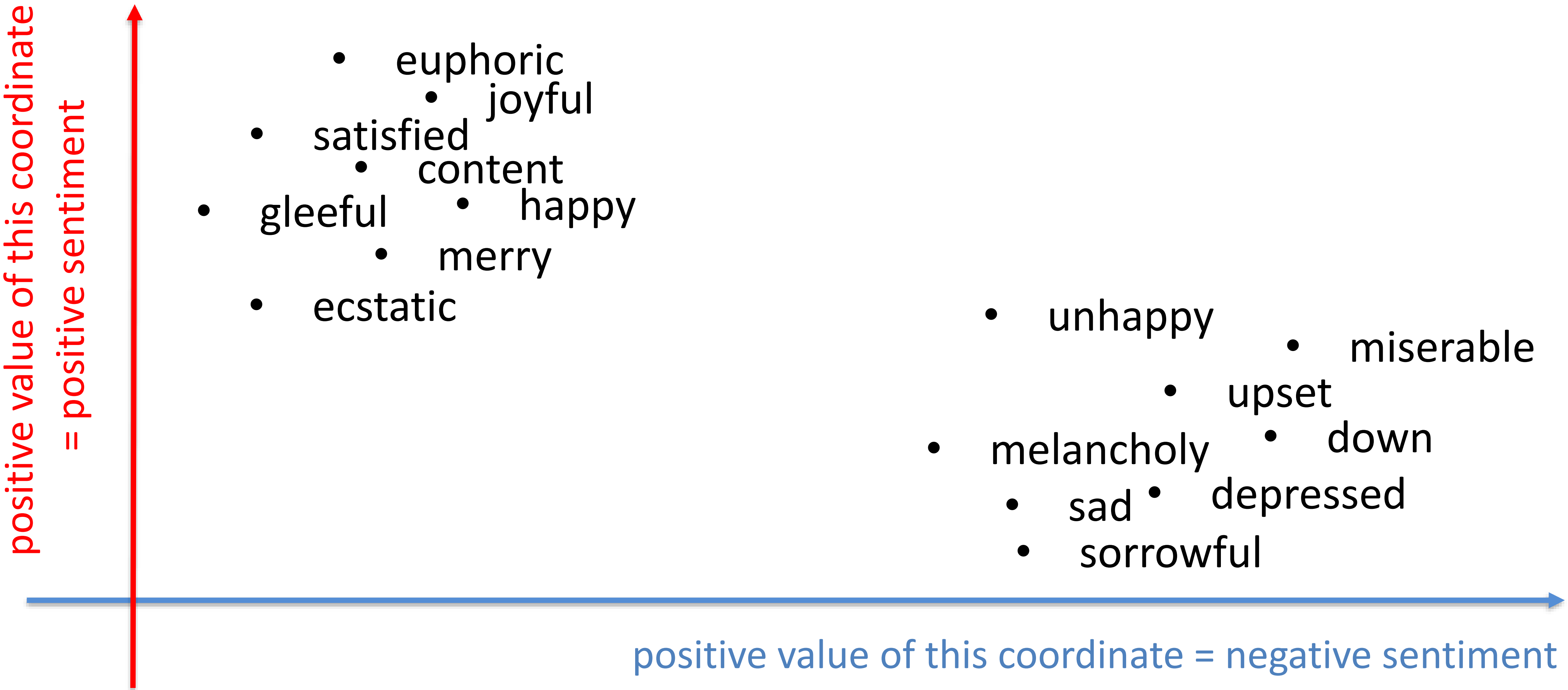
We’d like a representation of words that places similar words close together



We’d like a representation of words that places similar words close together



We’d like a representation of words that places similar words close together



Word Embeddings: Map Each Word in Vocabulary to a Point in Space

- The closer words are in the mapping, the more related (or synonymous) they are
- Question: how might we learn this mapping?

	lawyer 1	attorney 2	penguin 3	apple 4	●	●	●	V
longitude	78.8986	79.0558	135.0000	74.0060				
latitude	35.9940	35.9132	82.8628	40.7128				

vocabulary words

Example Word Geography

Here we show the learned geography of many different vocabulary words

(limited to 2 dimensions)

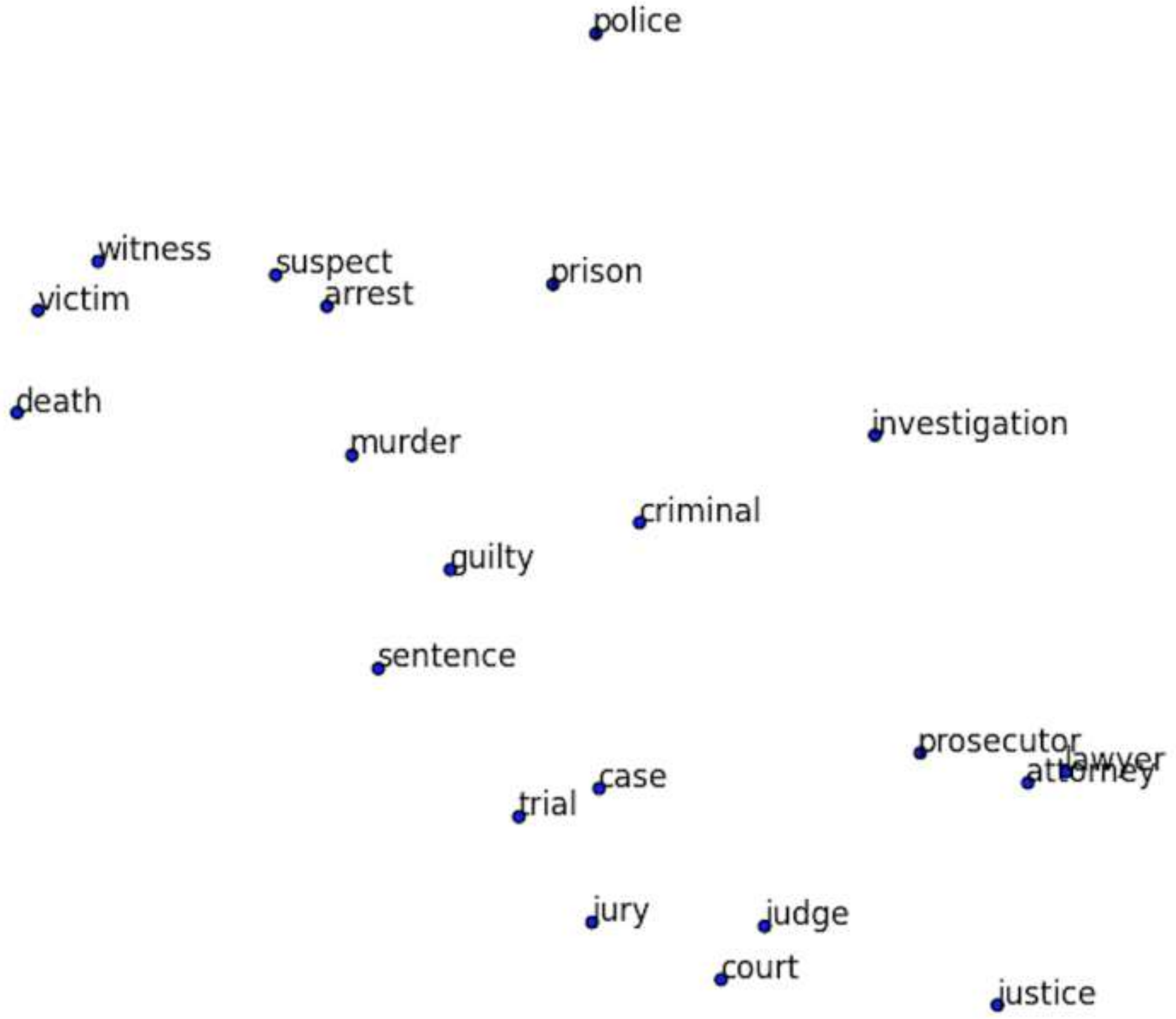
Too many words here to see! Let’s zoom in on a smaller section.

Example Word Geography

If we zoom in on a small region of our word map, it's all related words.

Note the similarity of all the words as a whole, but also of the individual neighbors.

"Lawyer" and "attorney" are nearly identical in space!

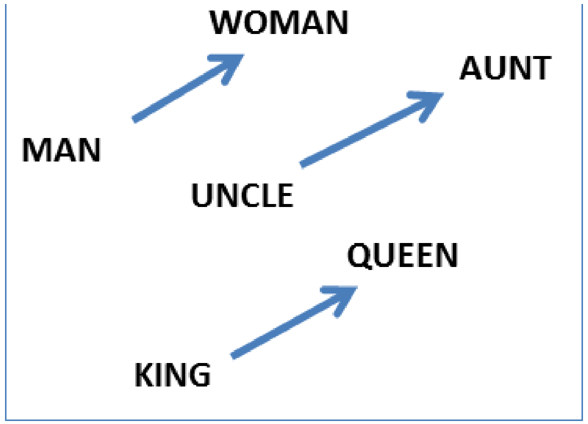


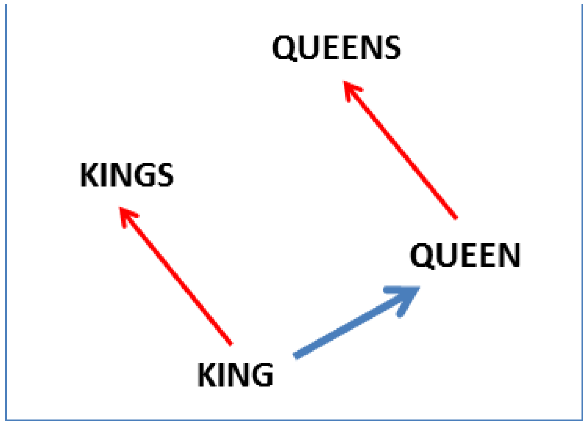
Learn Geographical Relationships

The relationship between words can be maintained, we can do mathematical operations on these word vectors.

Add the same vector distance between man and woman will convert uncle to aunt and king to queen.

Plural relationships are also maintained.





Word to Vector (Word2Vec)

- Map each word to a point in space

- We will use >2D space to allow us to capture many different “dimensions” of meaning

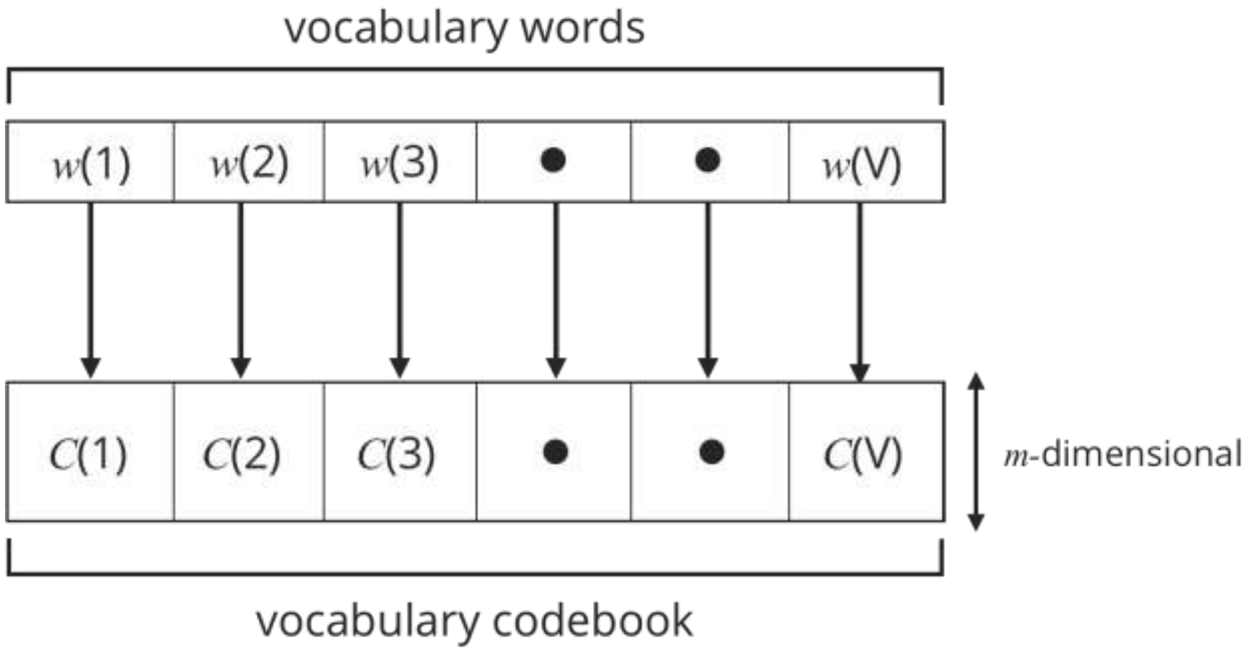
- Do this by creating a lookup table for each word in our *vocabulary* (i.e. all the words we know). In code, the lookup table is implemented as a dictionary.

	lawyer 1	attorney 2	penguin 3	apple 4	•	•	•	V
longitude	78.8986	79.0558	135.0000	74.0060				
latitude	35.9940	35.9132	82.8628	40.7128				

vocabulary words

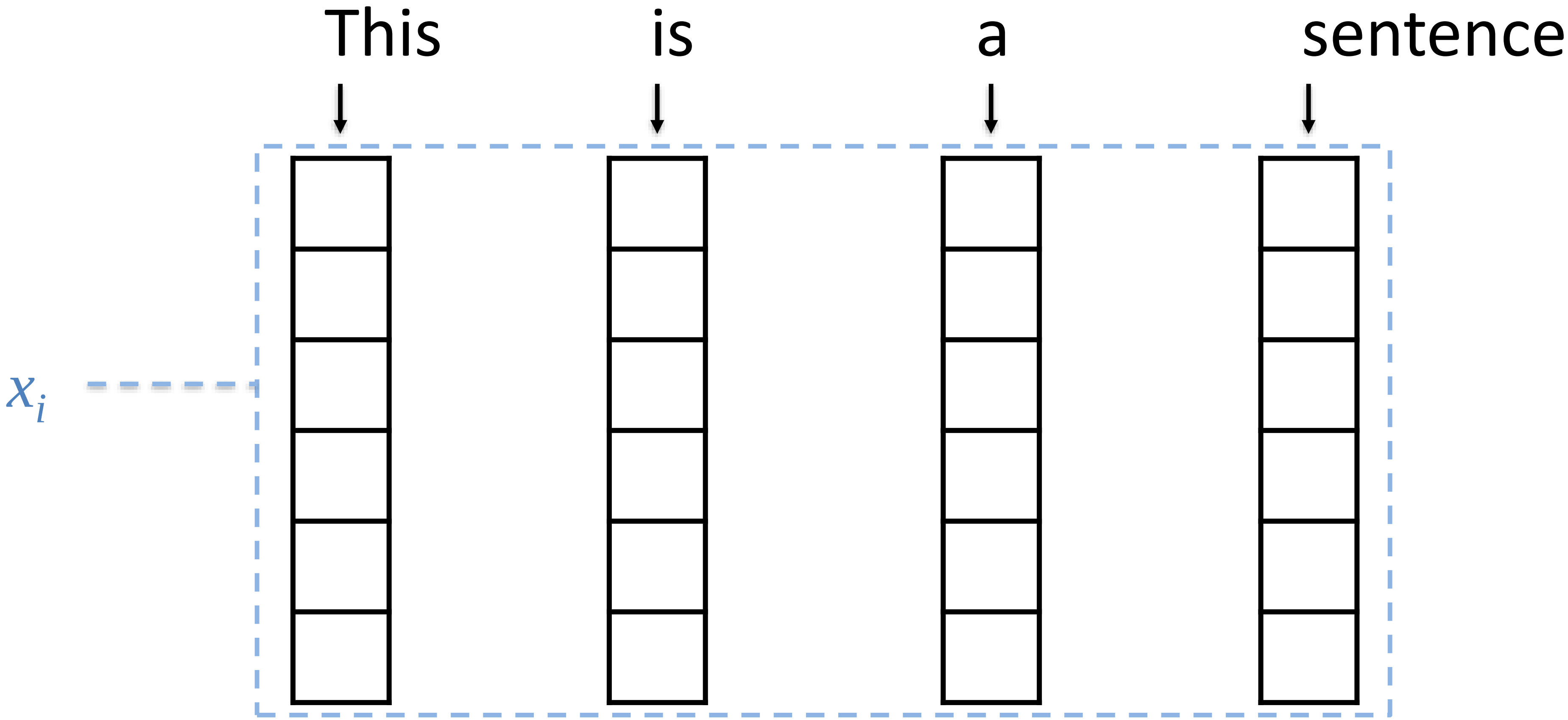
Word to Vector (Word2Vec)

- Enter the word into the dictionary
- Receive a vector, or "embedding" for that word



What happens when we apply this to a sentence?

- Look up words individually to obtain their vectors
- Construct a sequence of vectors



So how do we learn spatial locations for each word?

KEY IDEA: words are *defined* by the context in which they appear

A man strolls down the street

A woman strolls down the street

A child strolls down the street

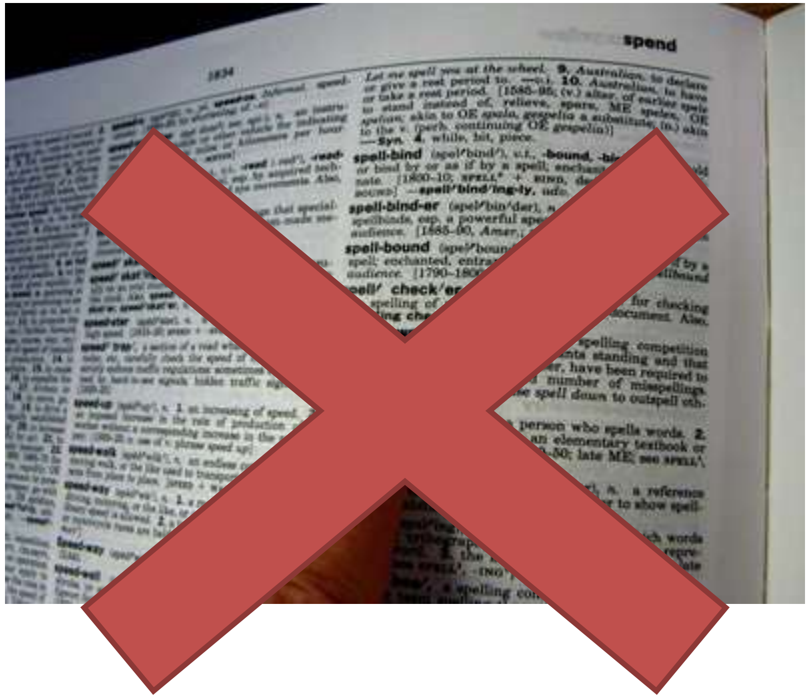
A crocodile strolls down the street

A banana strolls down the street

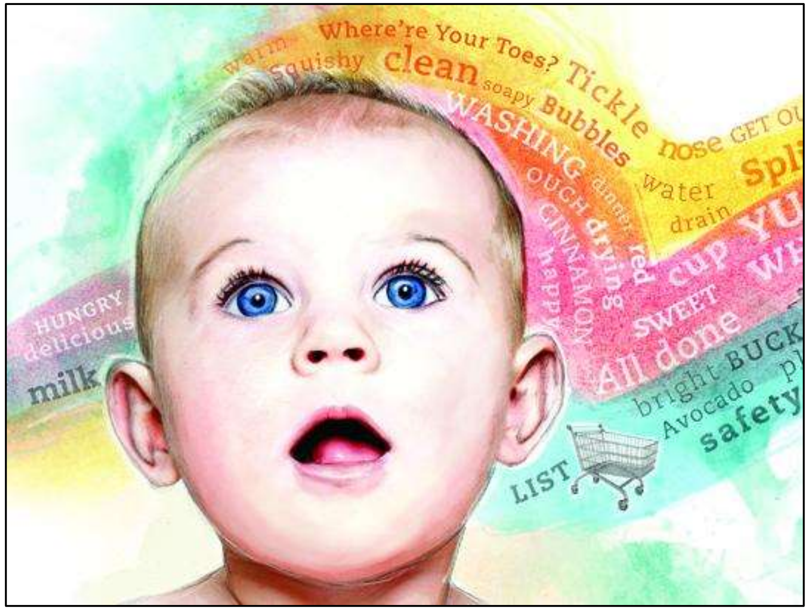
A concept strolls down the street

KEY IDEA: words are *defined* by the <u>context</u> in which they appear

-> if words are always exchangeable, they must have very similar meaning



https://www.parenting.com/activities/baby/teach-baby-to-talk/



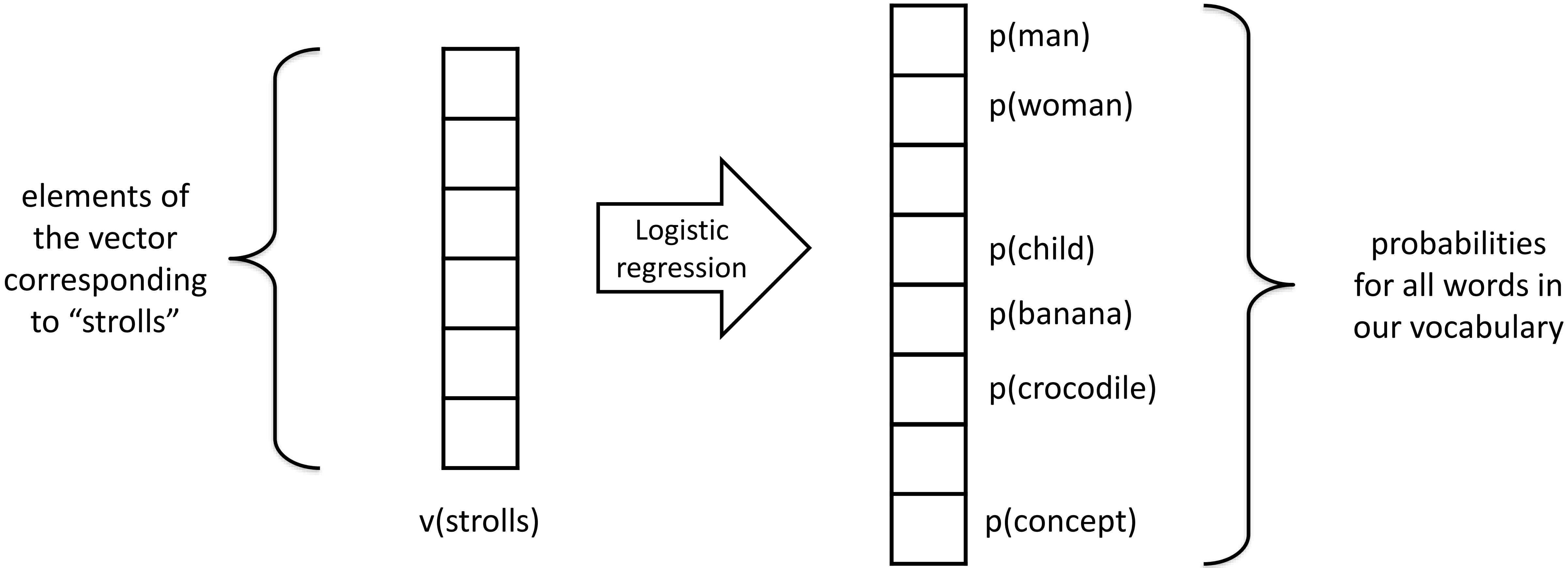
learn word meaning like an adult:
explicit definitions

learn word meaning like an child:
<u>implicit definitions from context</u>

LEARNING WORD EMBEDDINGS

So how do we learn spatial locations for each word?

We want: a vector for each word that allows us to predict its context (i.e. what other words are likely/unlikely to be around it)



Predict Context Words from Input Words

{input word, context word}

{strolls, man}
{strolls, woman}
{swims, crocodile}
{swims, fish}
{flies, bird}
{flies, plane}

We define a context word as one that appears inside a fixed-length window around the input word in our training corpus.

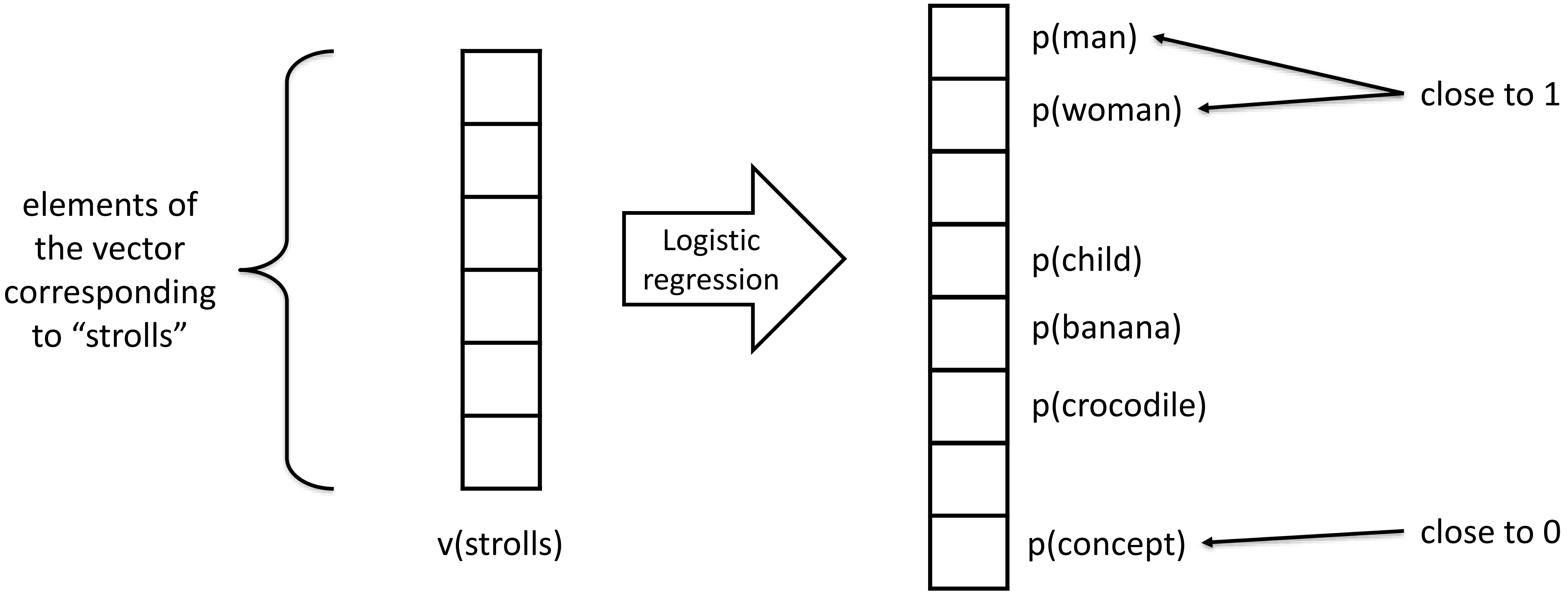
(e.g. Wikipedia)

A man $\boxed{\text{strolls}}$ down the street.

input context

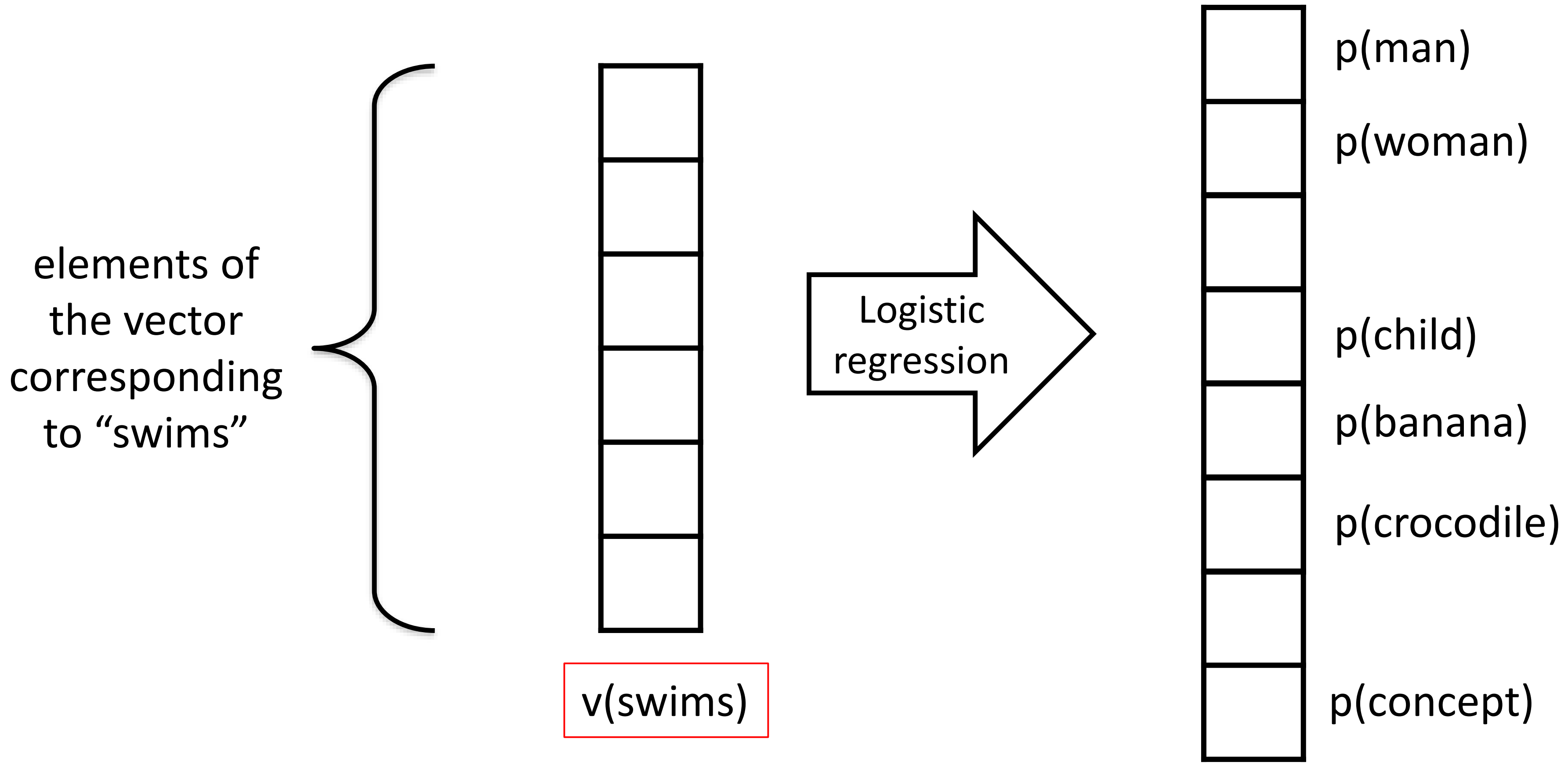
So how do we learn spatial locations for each word?

We want: a vector for each word that allows us to predict its context (i.e. what other words are likely/unlikely to be around it)



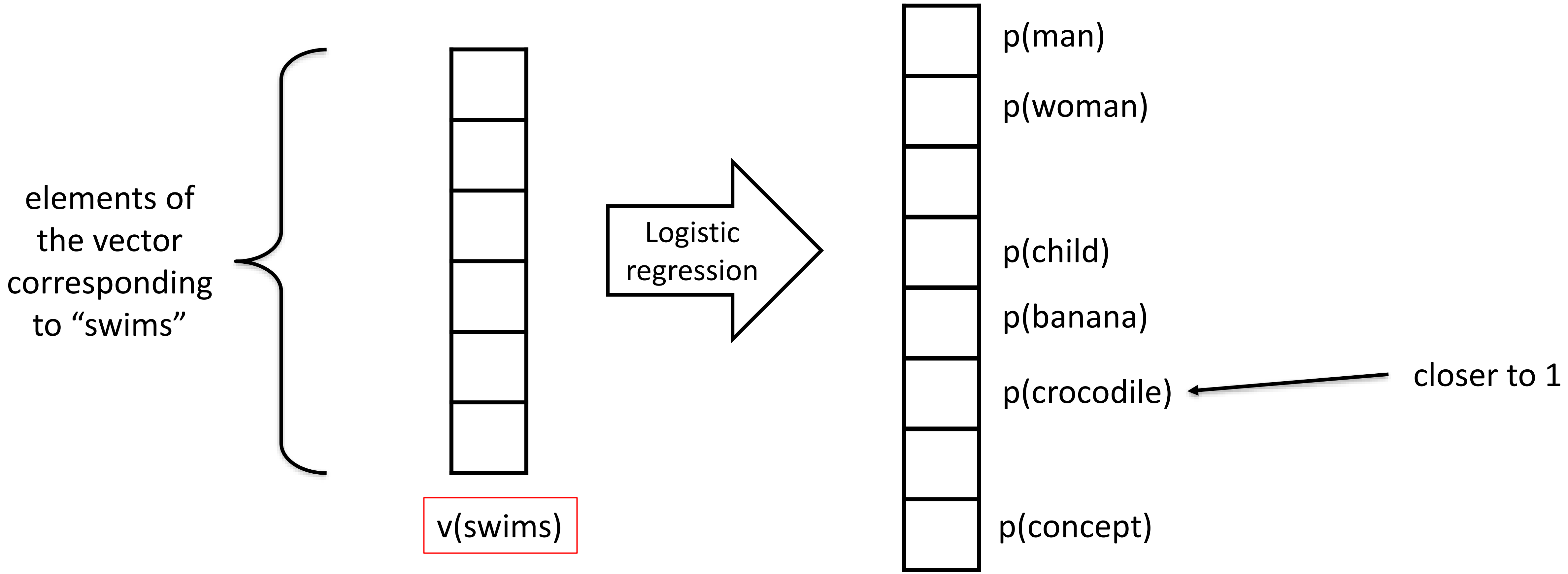
So how do we learn spatial locations for each word?

<u>We want</u>: a vector for each word that allows us to predict its context (i.e. what other words are likely/unlikely to be around it)



So how do we learn spatial locations for each word?

<u>We want</u>: a vector for each word that allows us to predict its context (i.e. what other words are likely/unlikely to be around it)

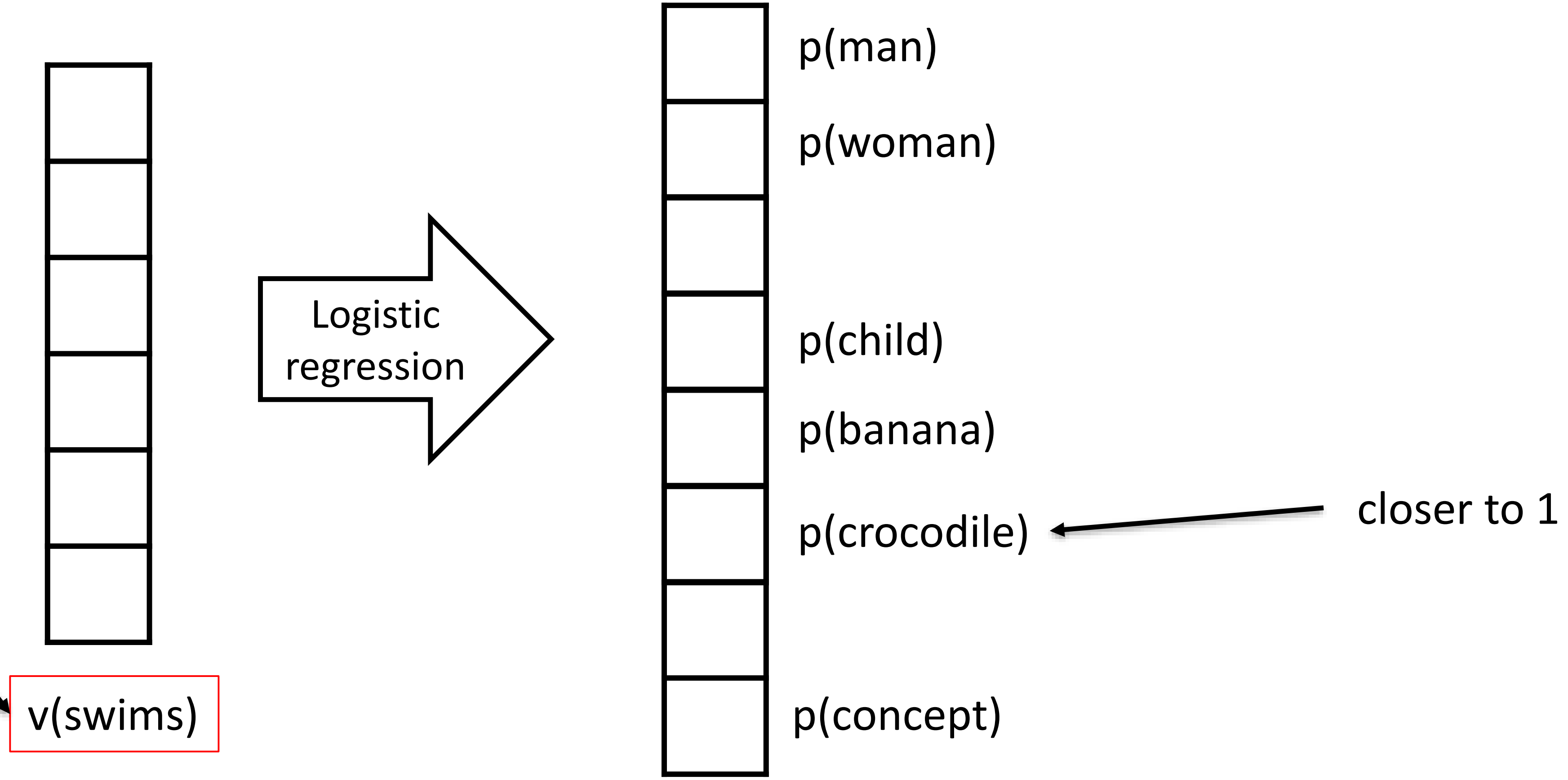


So how do we learn spatial locations for each word?

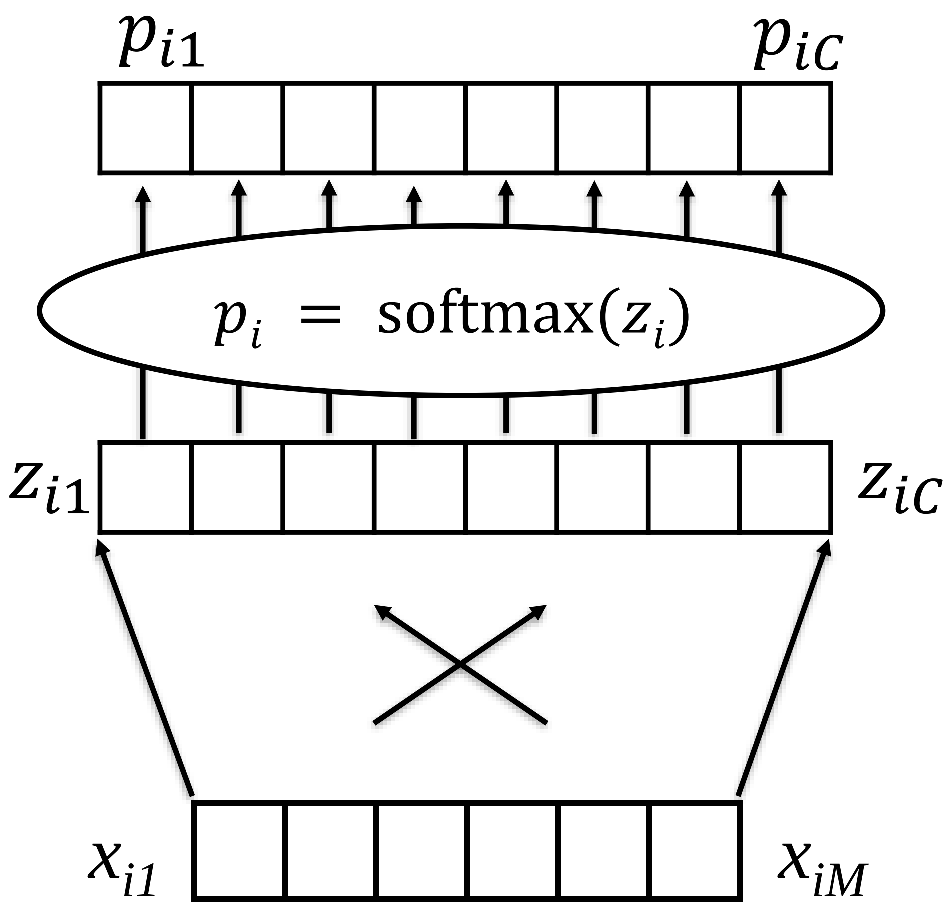
We want: a vector for each word that allows us to predict its context (i.e. what other words are likely/unlikely to be around it)

These vectors are what we want to *learn*

We could build this model if we had the vectors; less clear how to get them

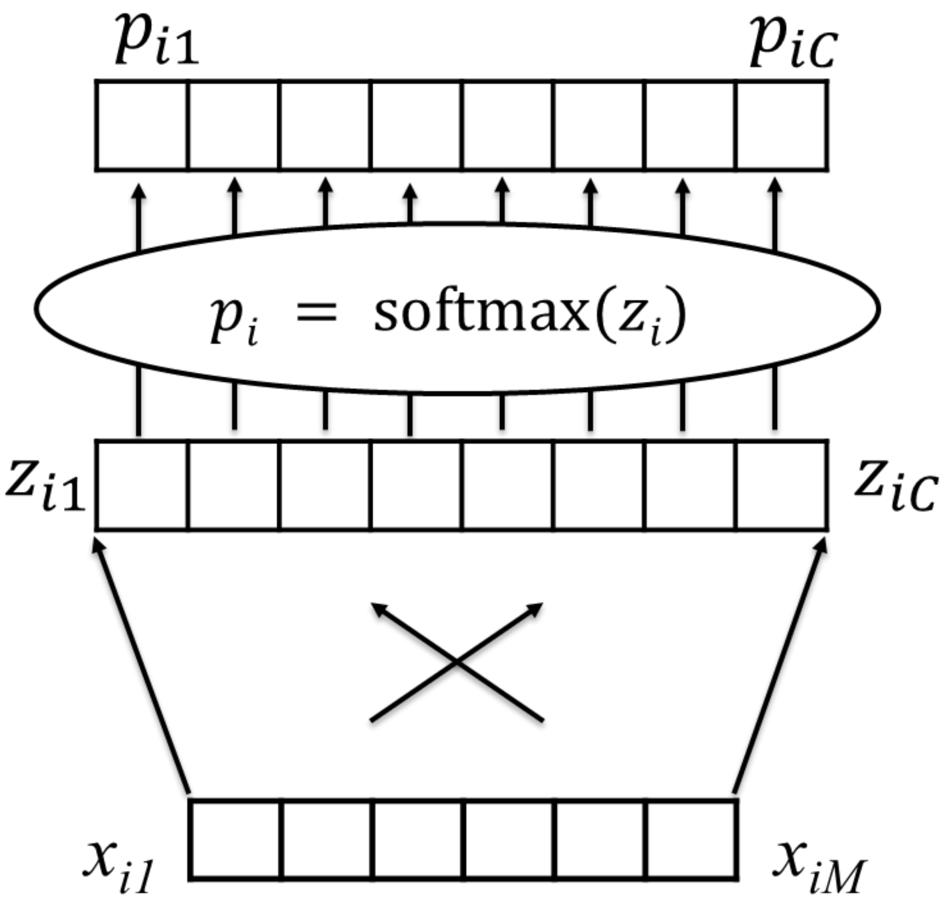


Recall: Multi-Class Logistic Regression

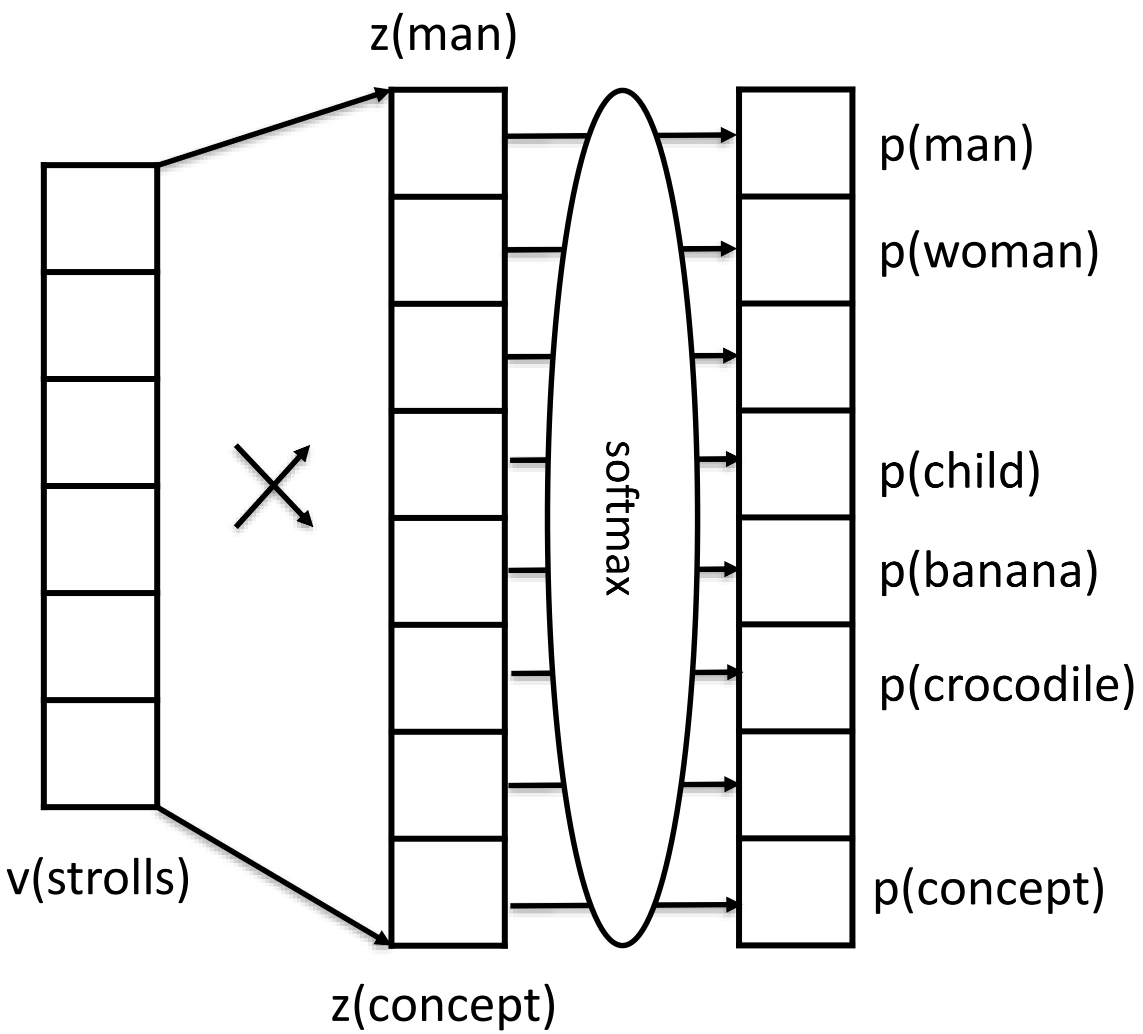


$$p_{ij} = \frac{e^{z_{ij}}}{\sum_{c=1}^{C} e^{z_{ic}}}$$

Recall: Multi-Class Logistic Regression



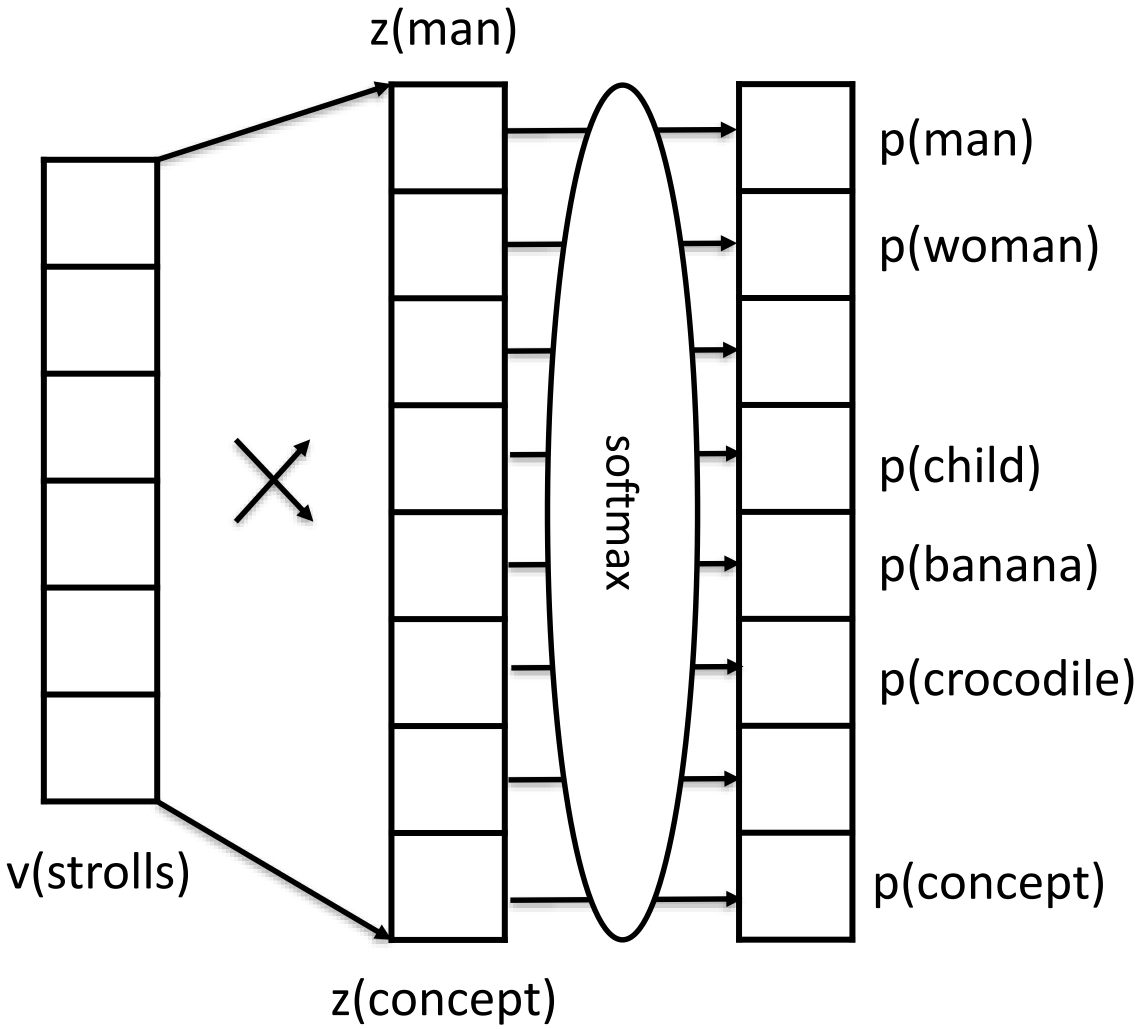
Recall: Multi-Class Logistic Regression



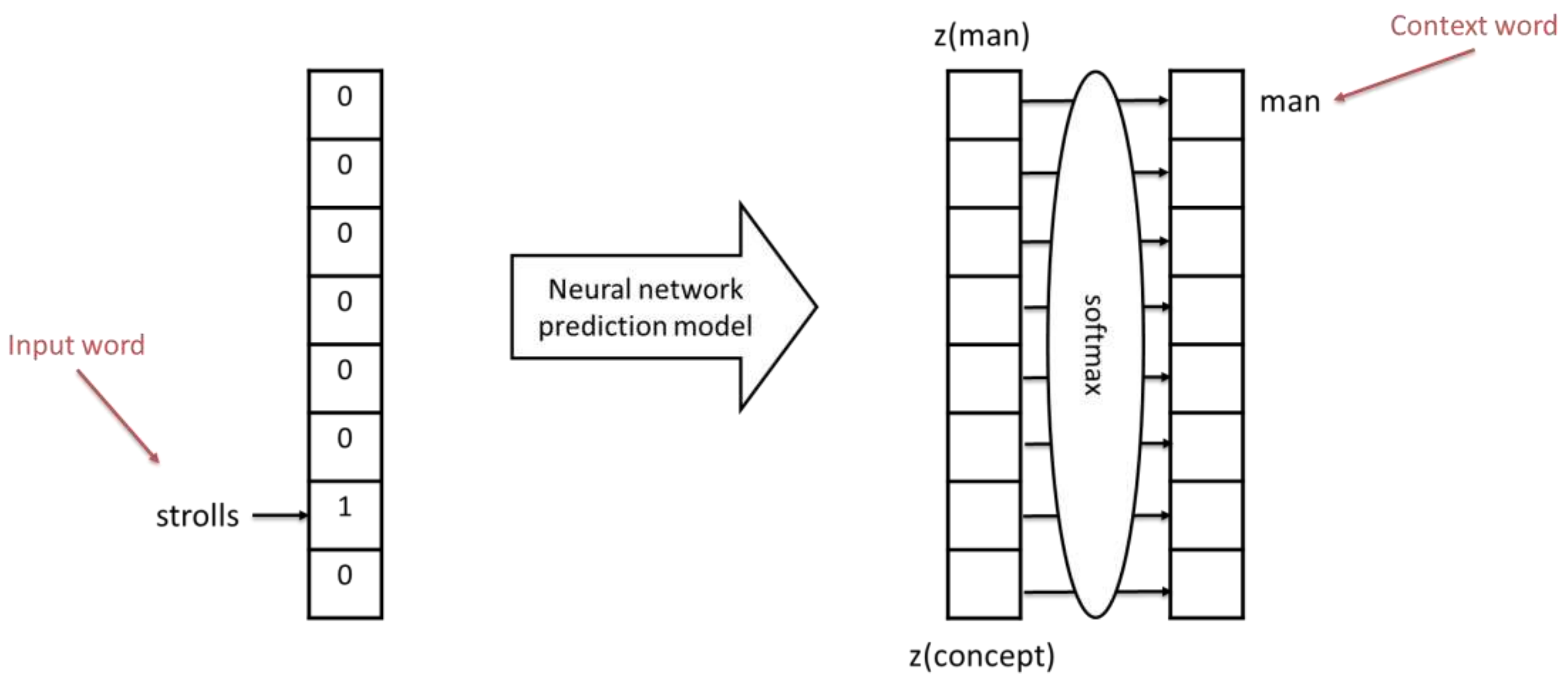
We want: word vectors that allow us to predict their likely context

But again, how do we *learn* these vectors?

Let's take a step back: we'll focus on understanding how we can predict context words based on input words



Predicting context words based on input words



Input words and context words are one-hot encoded
(similar to bag of words representation)

Predicting context words based on input words

Training Data:

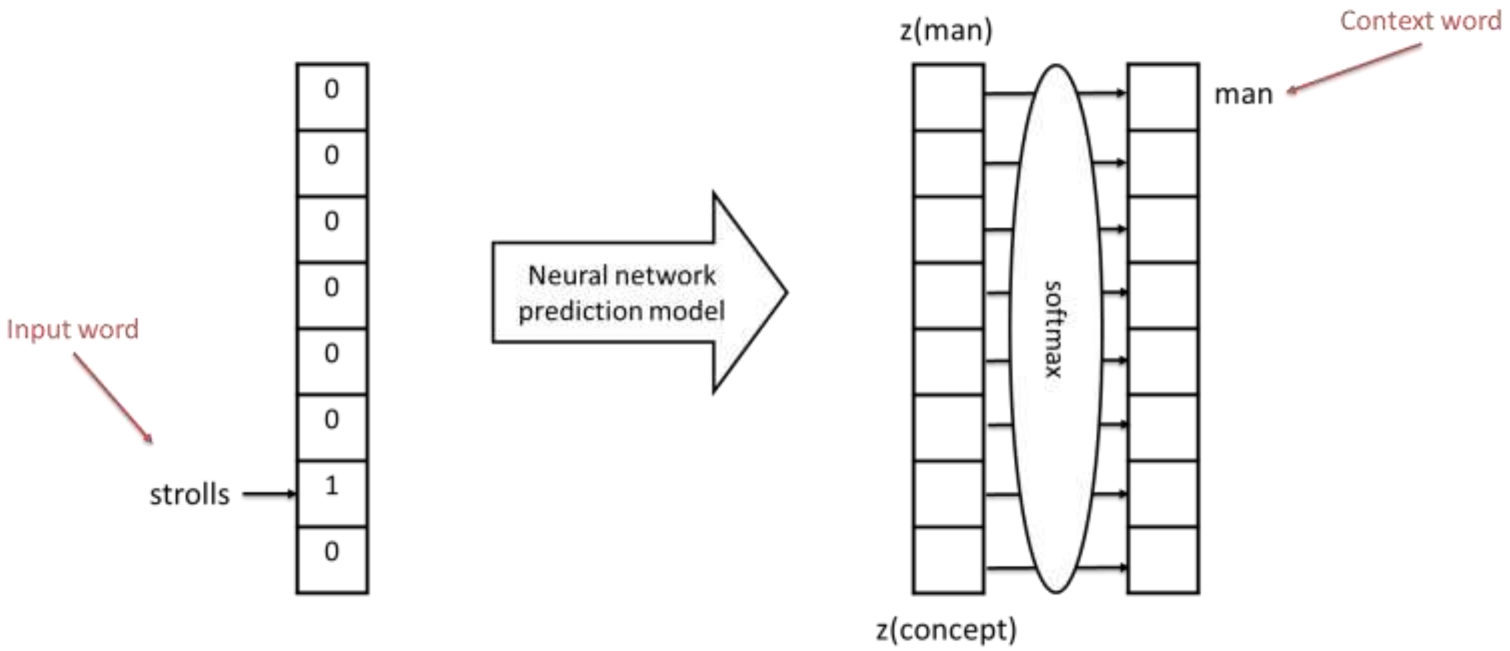
HUGE number of pairs of the following form:

{input word, context word}

e.g. from Wikipedia

Examples:

{strolls, man}
{strolls, woman}
{swims, crocodile}
{swims, fish}
{flies, bird}
{flies, plane}



Predicting context words based on input words

Training Data:

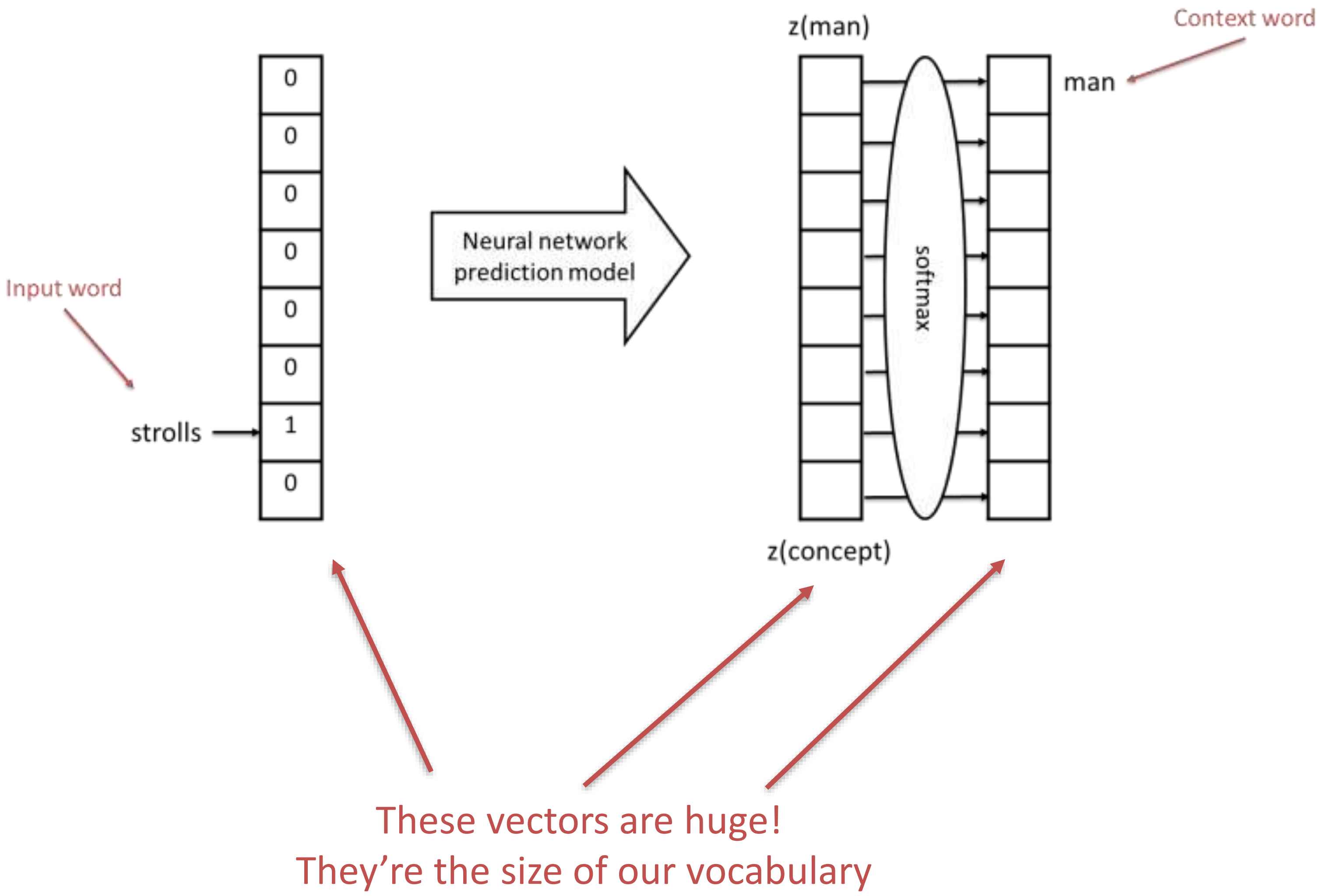
HUGE number of pairs of the following form:

{input word, context word}

e.g. from Wikipedia

Examples:

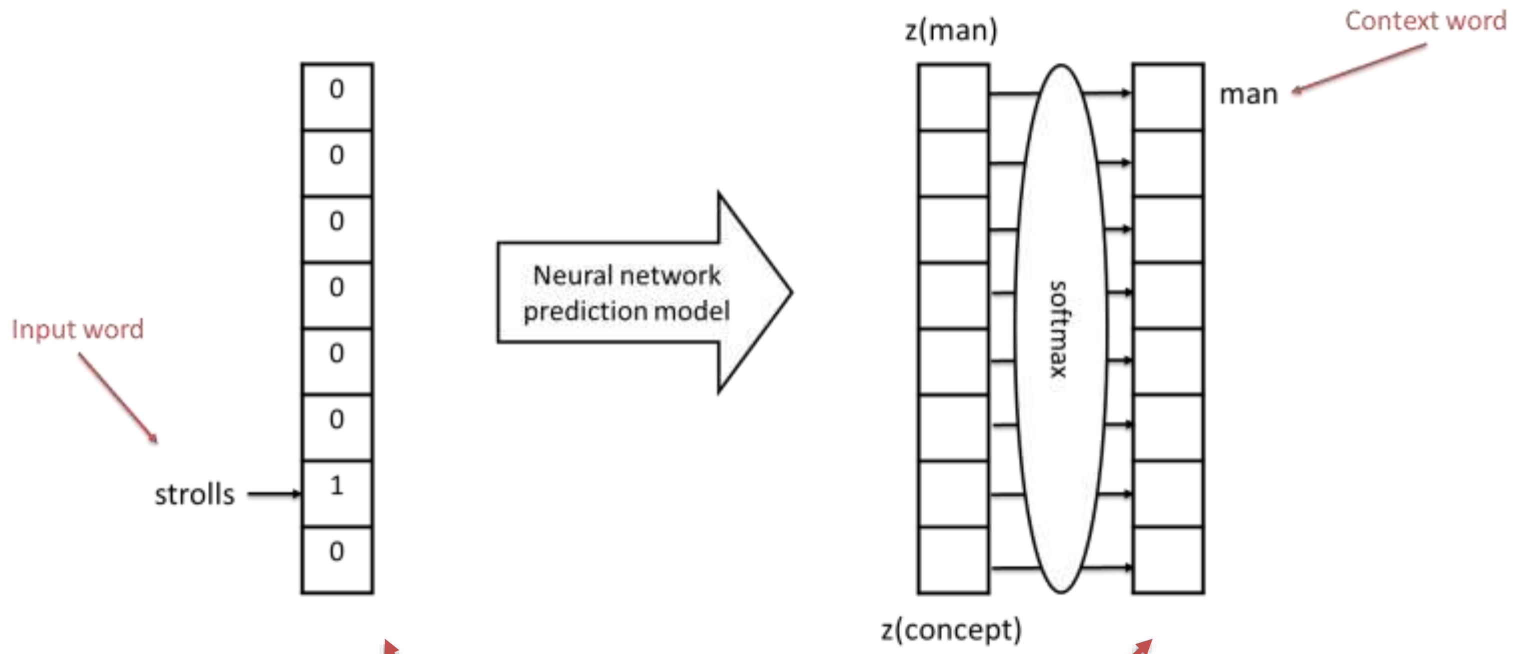
{strolls, man}
{strolls, woman}
{swims, crocodile}
{swims, fish}
{flies, bird}
{flies, plane}



What’s the simplest model we can possibly use?

First idea:

Directly connect our input to the log-odds layer



These vectors are huge!
They’re the size of our vocabulary

What’s the simplest model we can possibly use?

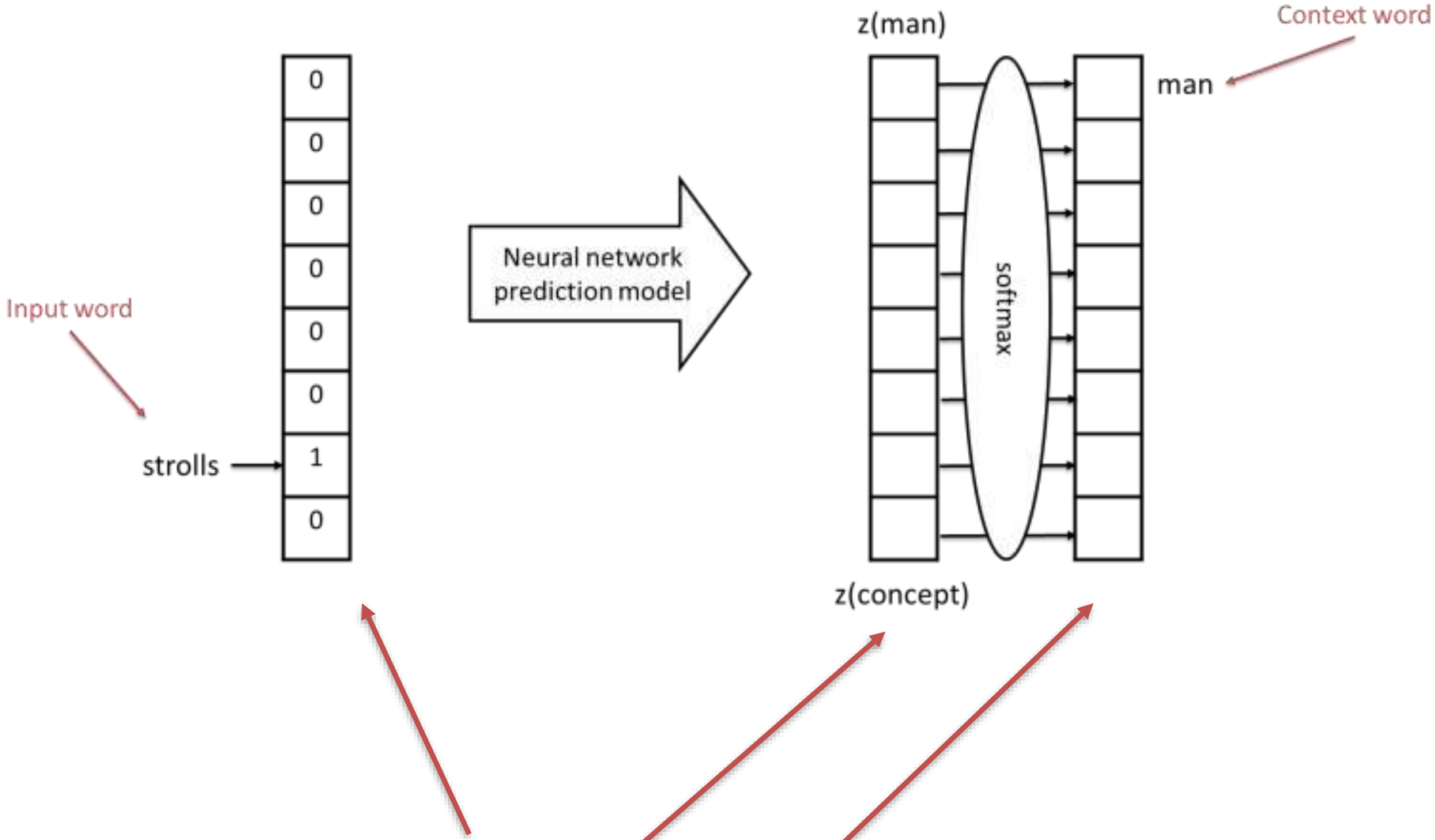
First idea:

Direc conne ur input
to the   -odd ayer

How man nnections?

V x V

Where is ou
vocab ary size
(approx. 6 billion)



These vectors are huge!
They’re the size of our vocabulary

What's the next simplest?

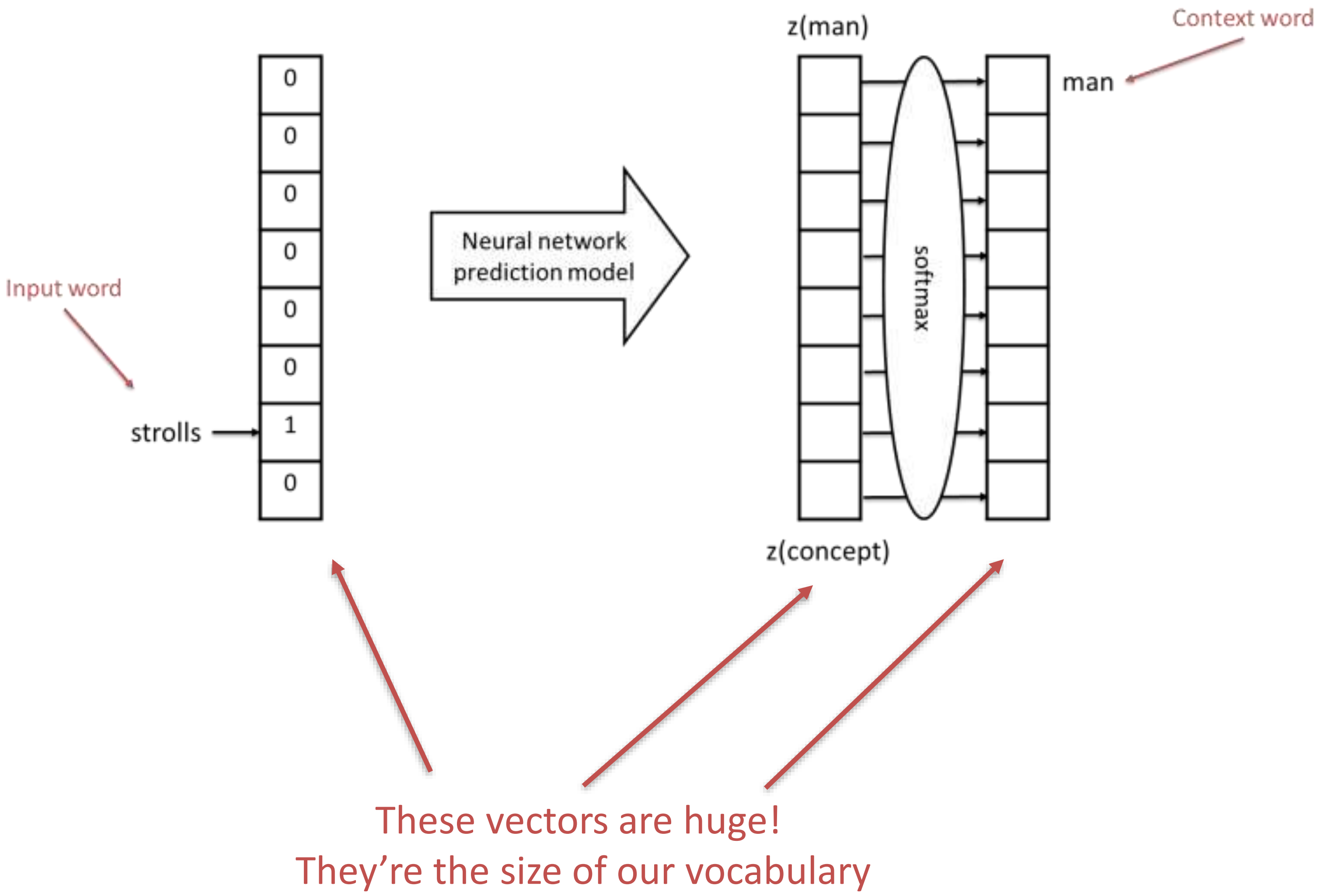
How about a single hidden layer?

How many connections?

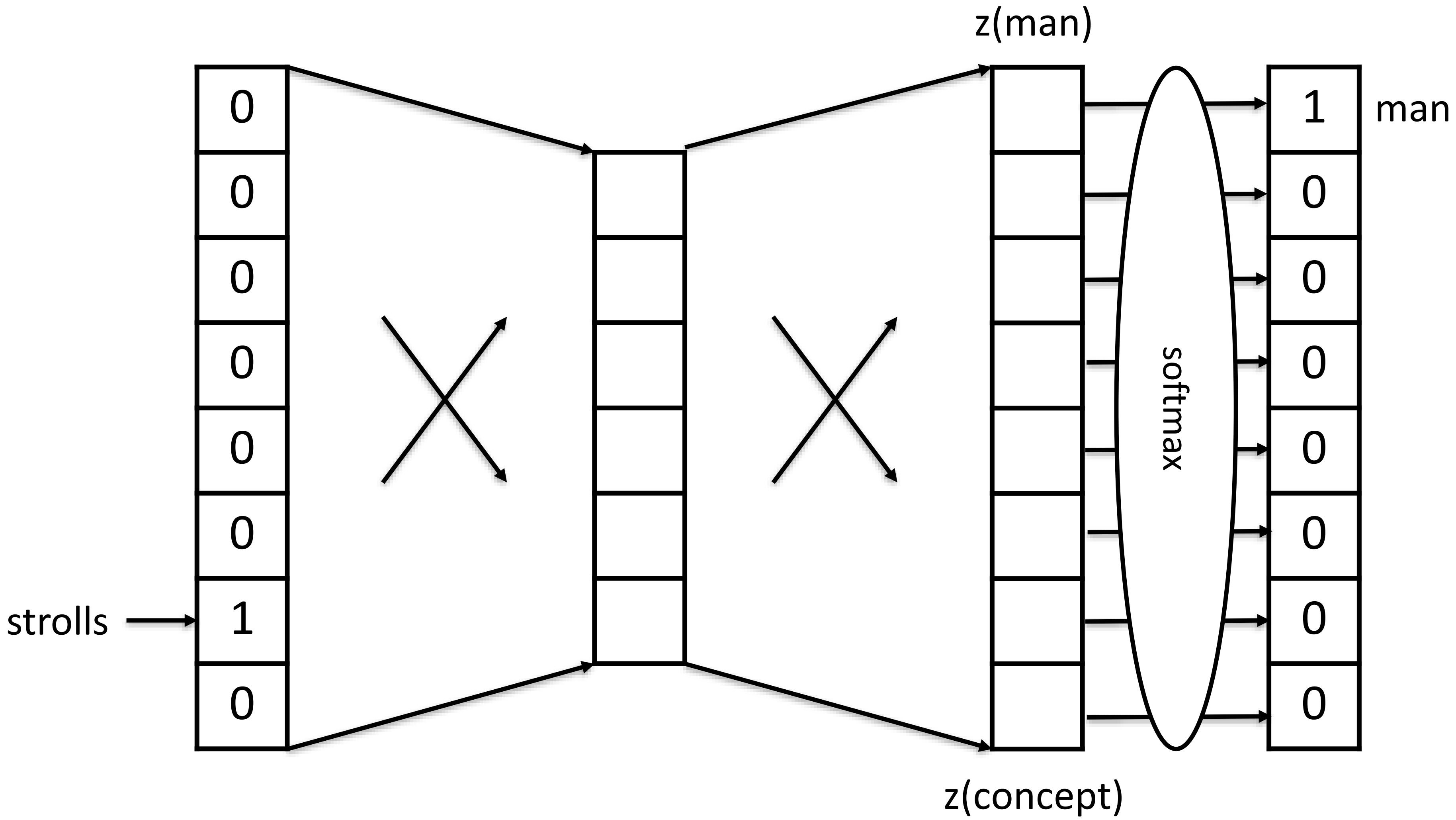
V x H x 2

Where V is our vocabulary size (approx. 6 billion)

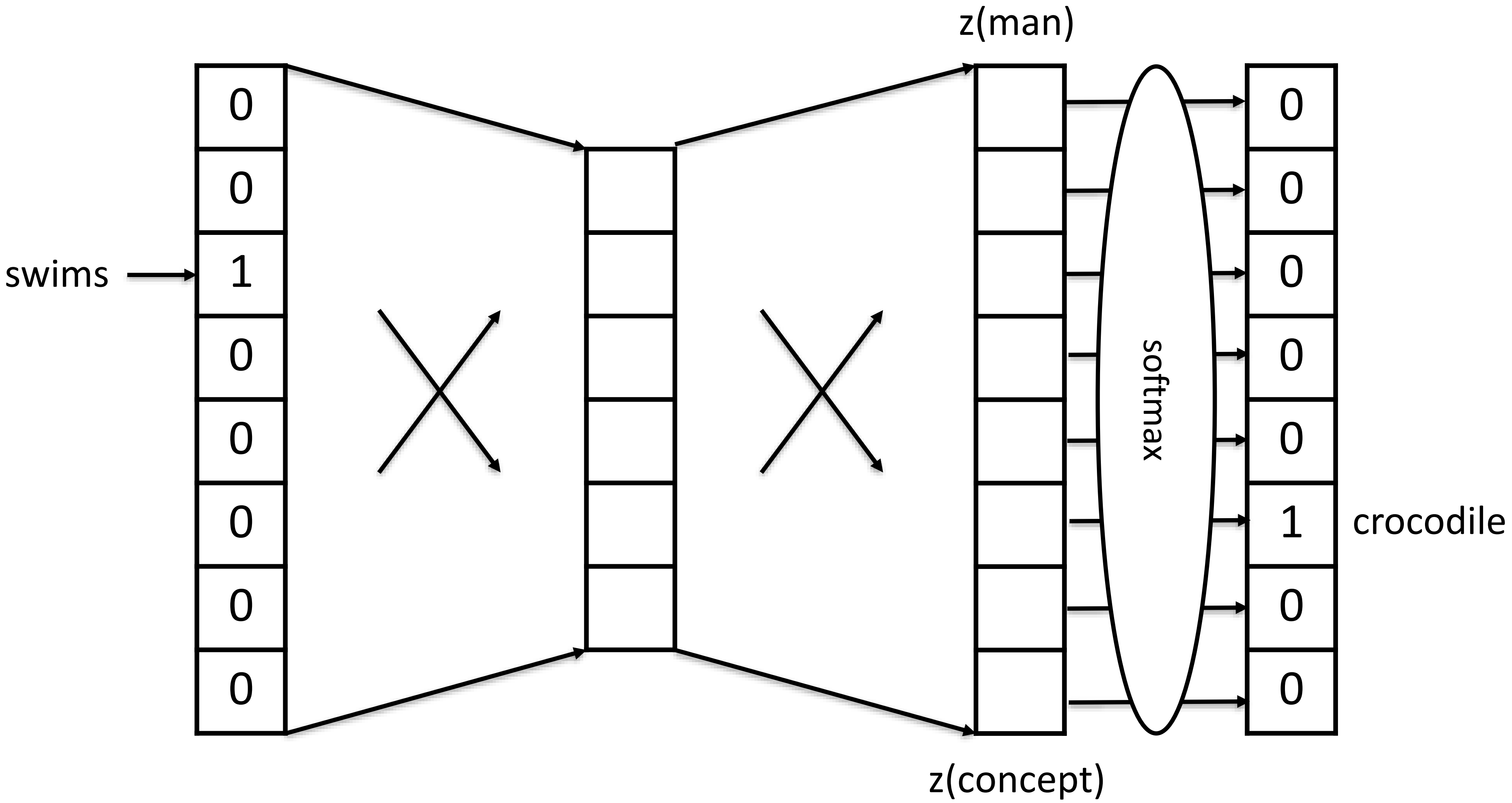
And H is our hidden layer size ($\ll V$)



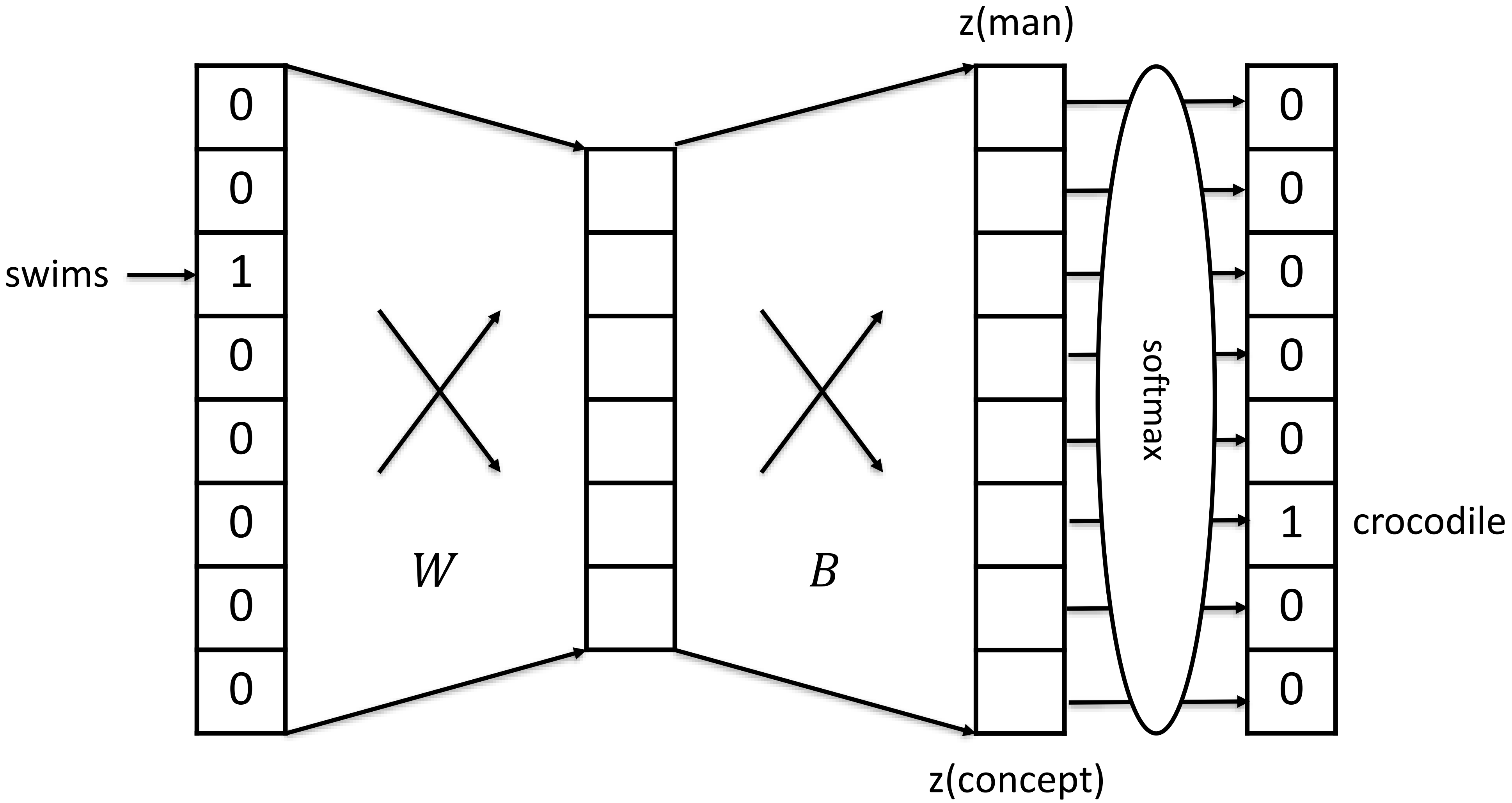
OK, let’s try it: use a single hidden layer



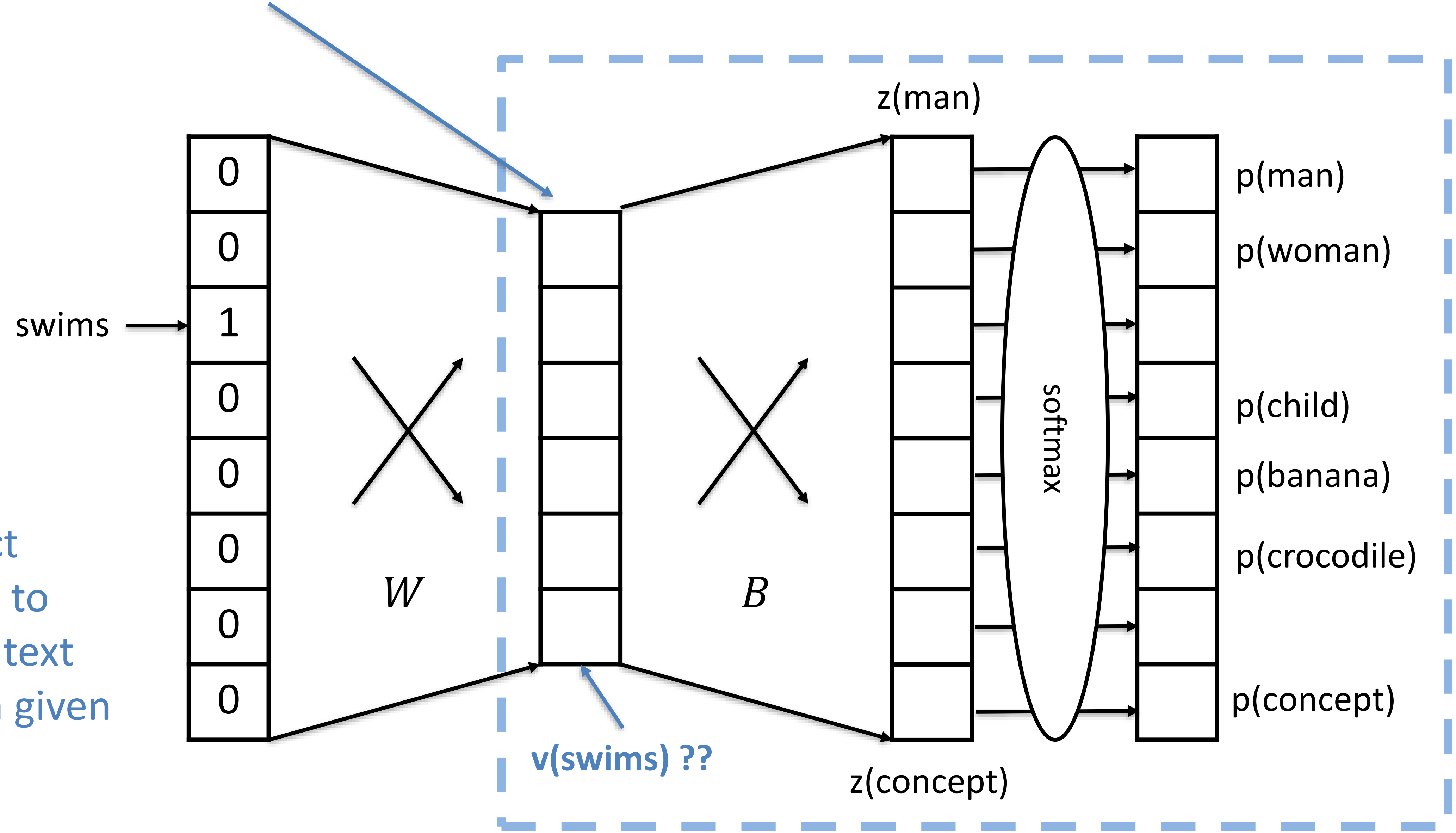
Use mini-batches of training examples; minimize cross-entropy loss



Learn our parameters: Weight Matrices W and B

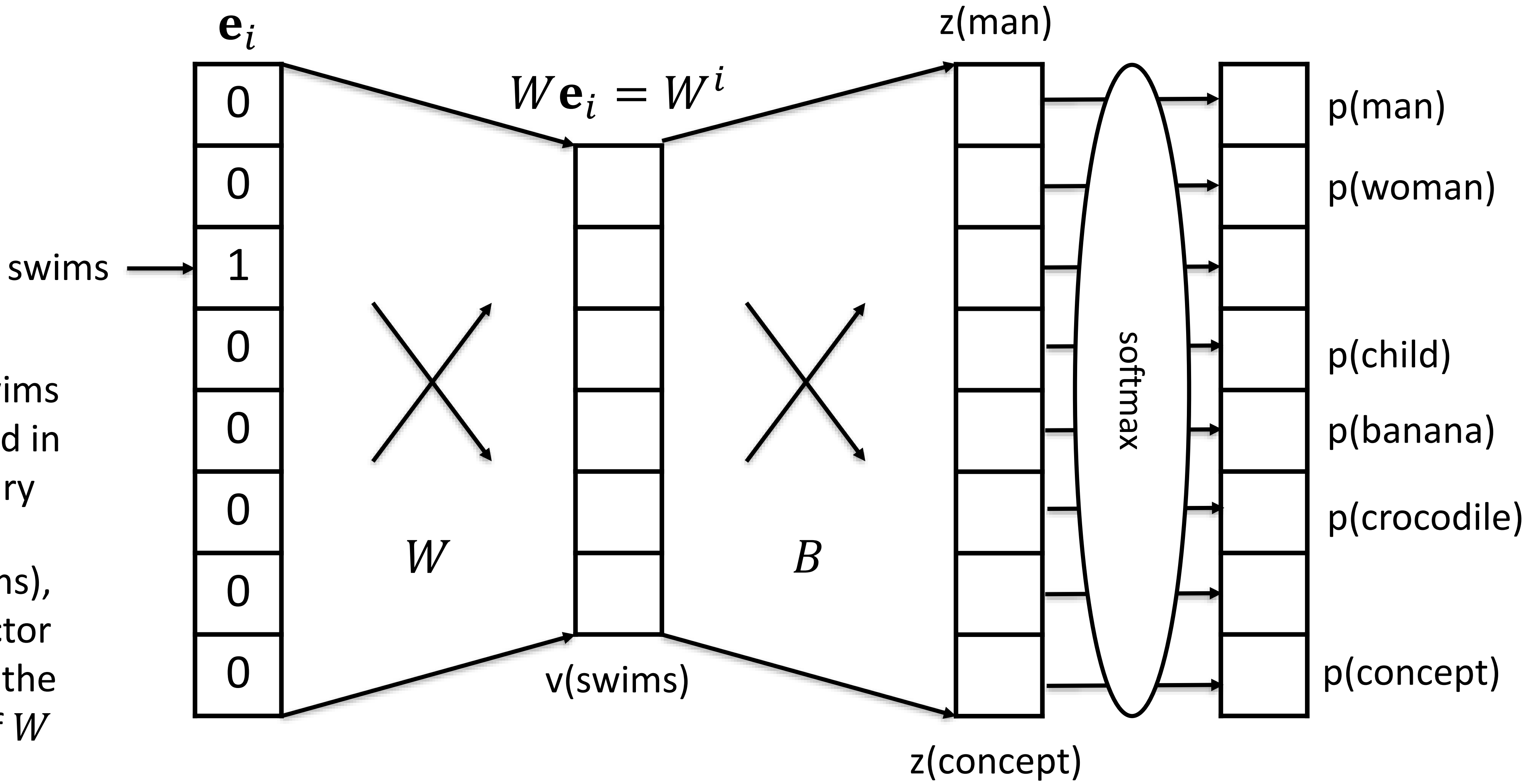


Isn't **this** the vector we were looking for?



- it's compact
- It allows us to predict context words for a given input word

Let's take a closer look at W



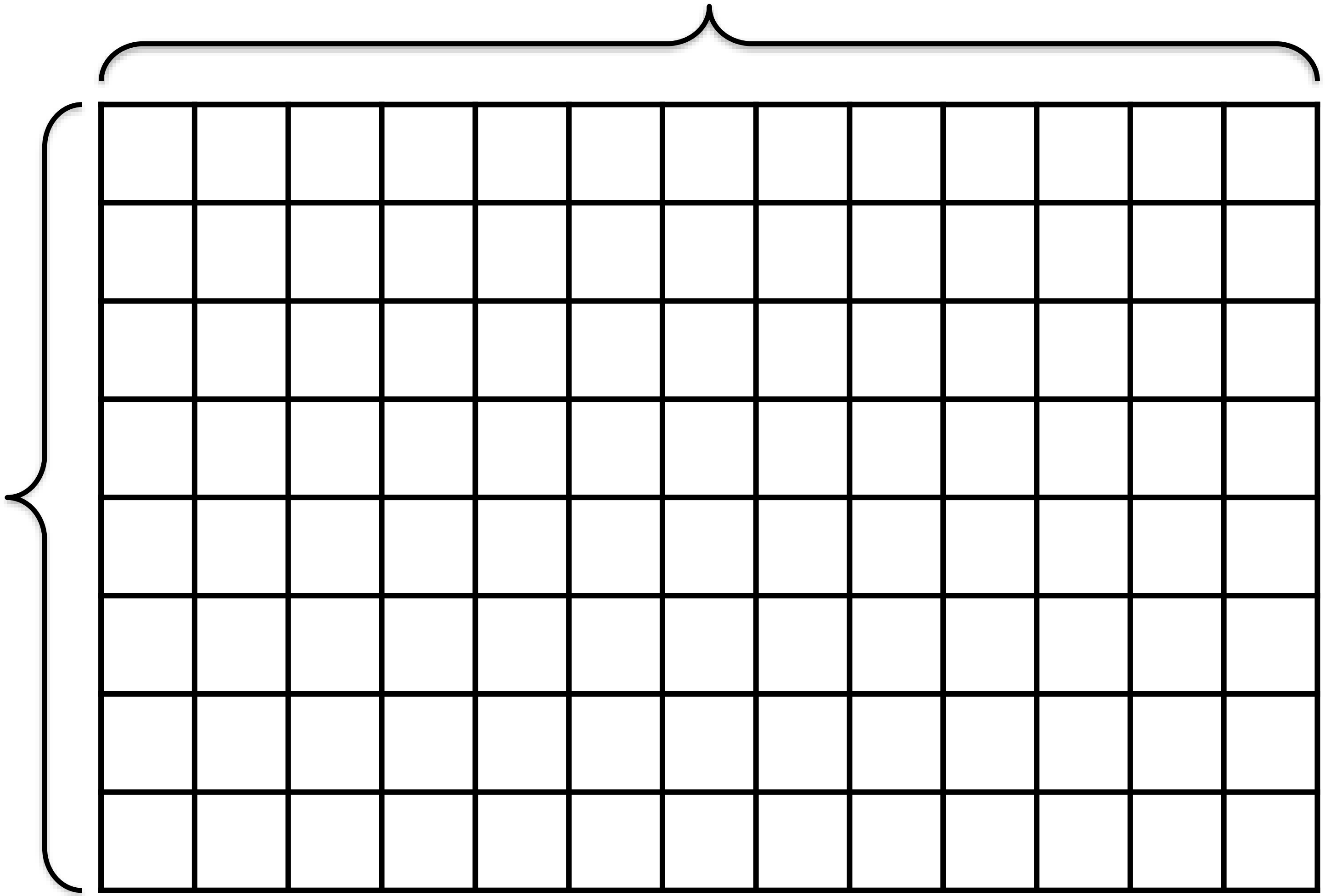
- suppose swims is the i^{th} word in our vocabulary

- then v(swims), the word vector for swims, is the i^{th} column of W

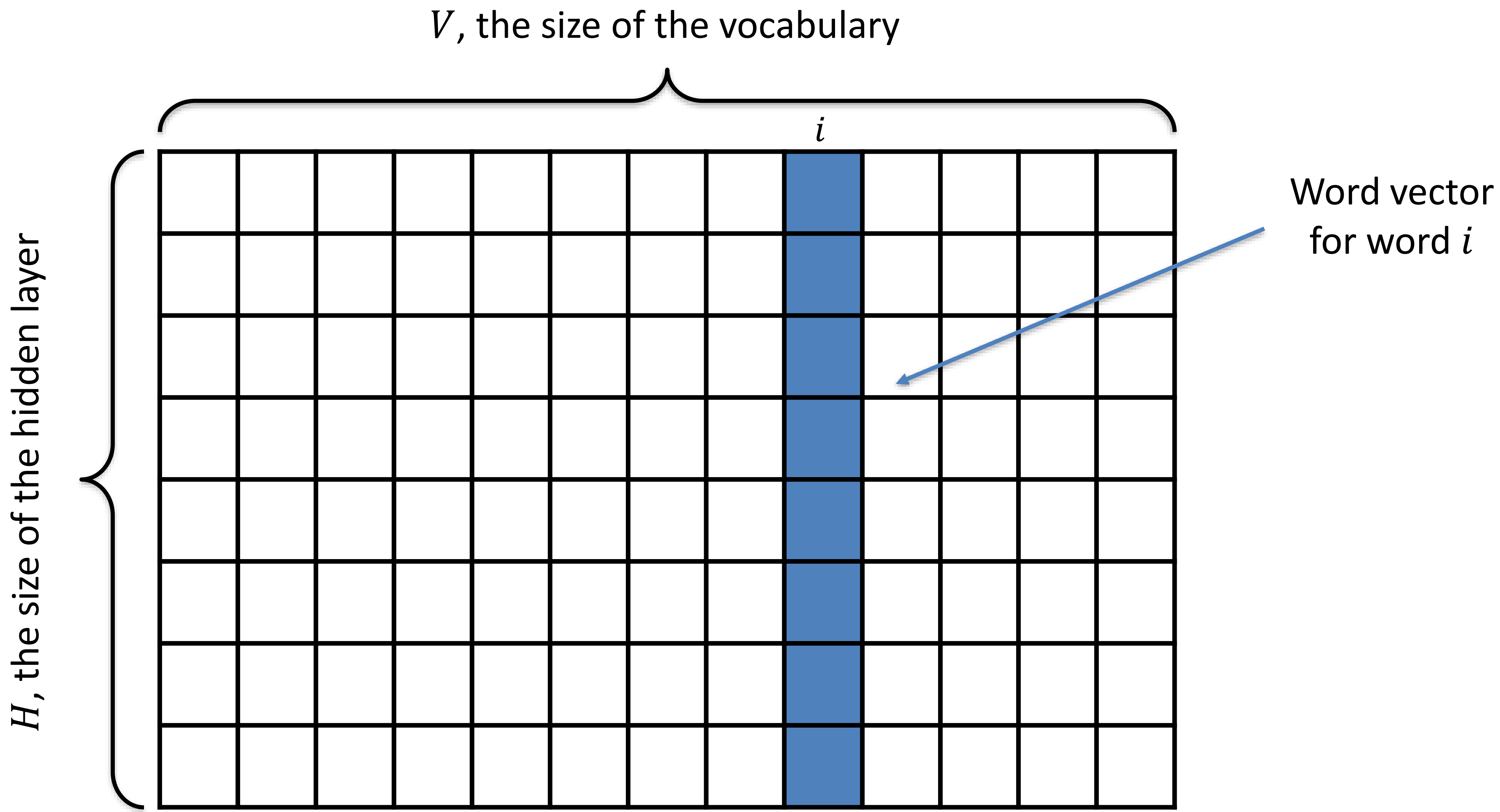
Let’s take a closer look at W

V, the size of the vocabulary

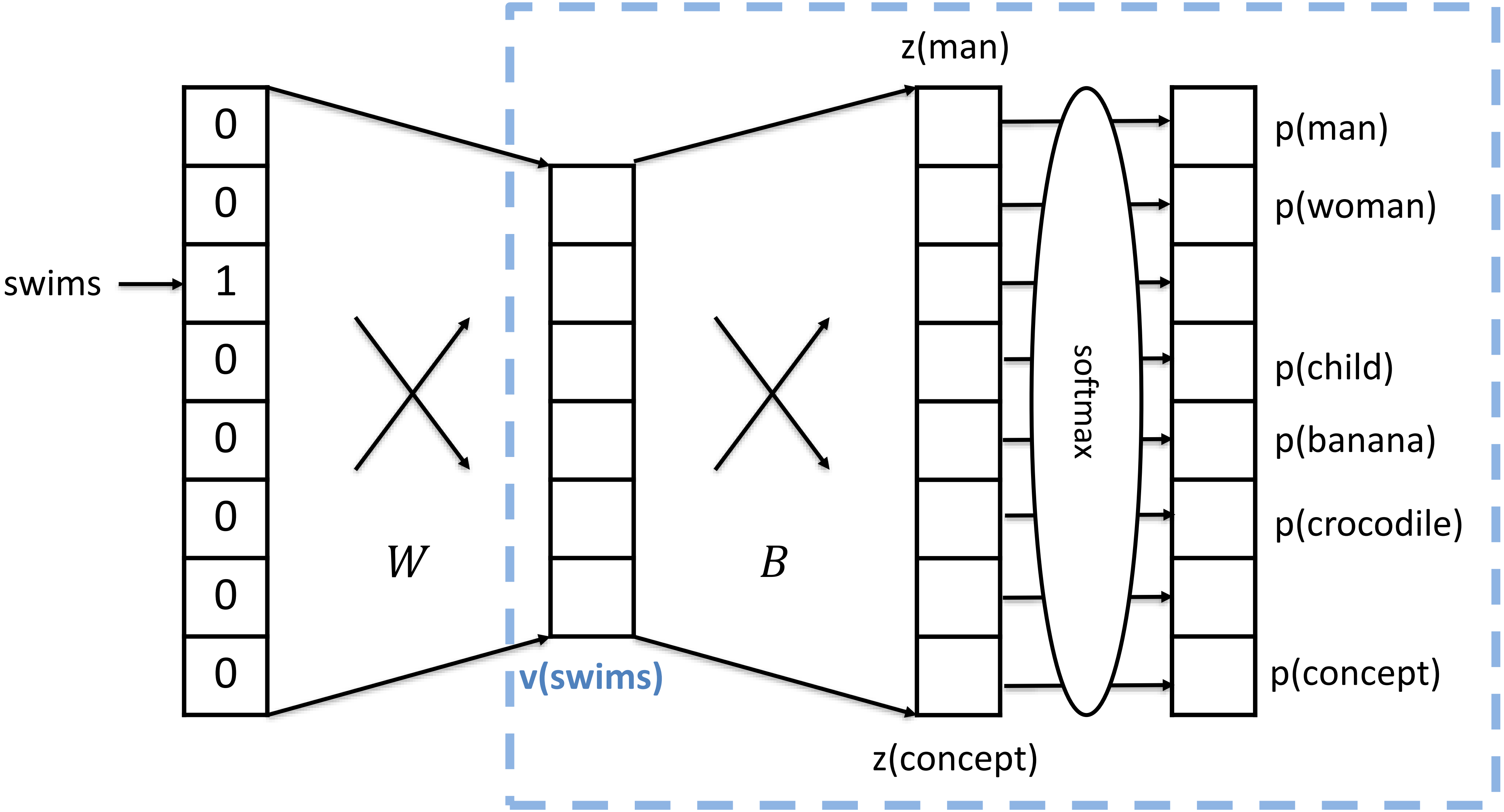
H, the size of the hidden layer



Let's take a closer look at W



We now have a distributed representation of word ***meaning*** based on ***context***



Important Takeaways:

- We are learning a vector representation for each word based on the contexts in which it appears

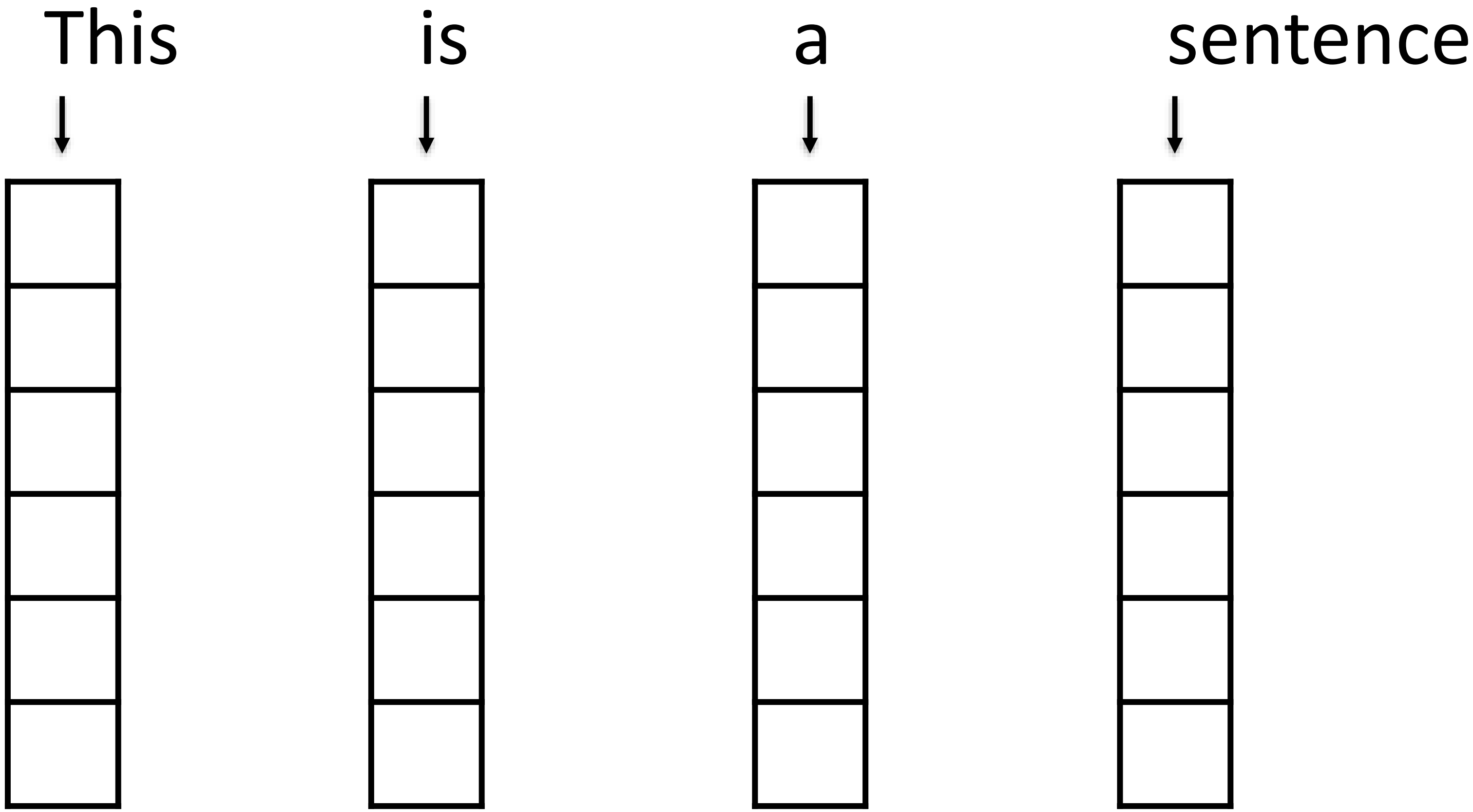
- training data: large number of pairs of nearby words from a large corpus

- These vectors give us much more flexibility when modeling: makes text sequences like other sequences

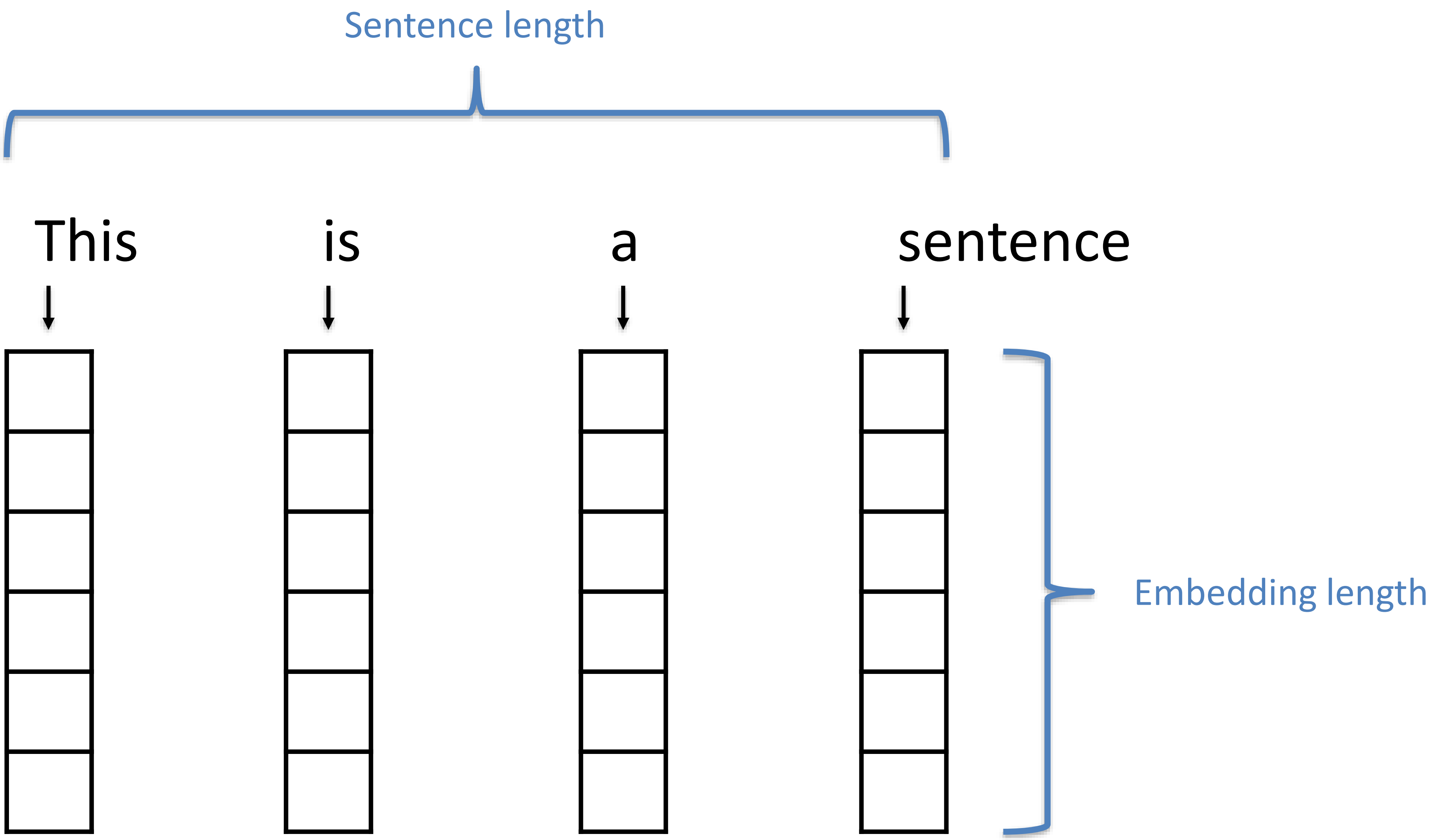
A VERY SIMPLE WORD EMBEDDING-BASED MODEL

VSWEM Step 1: Convert sentence to vectors

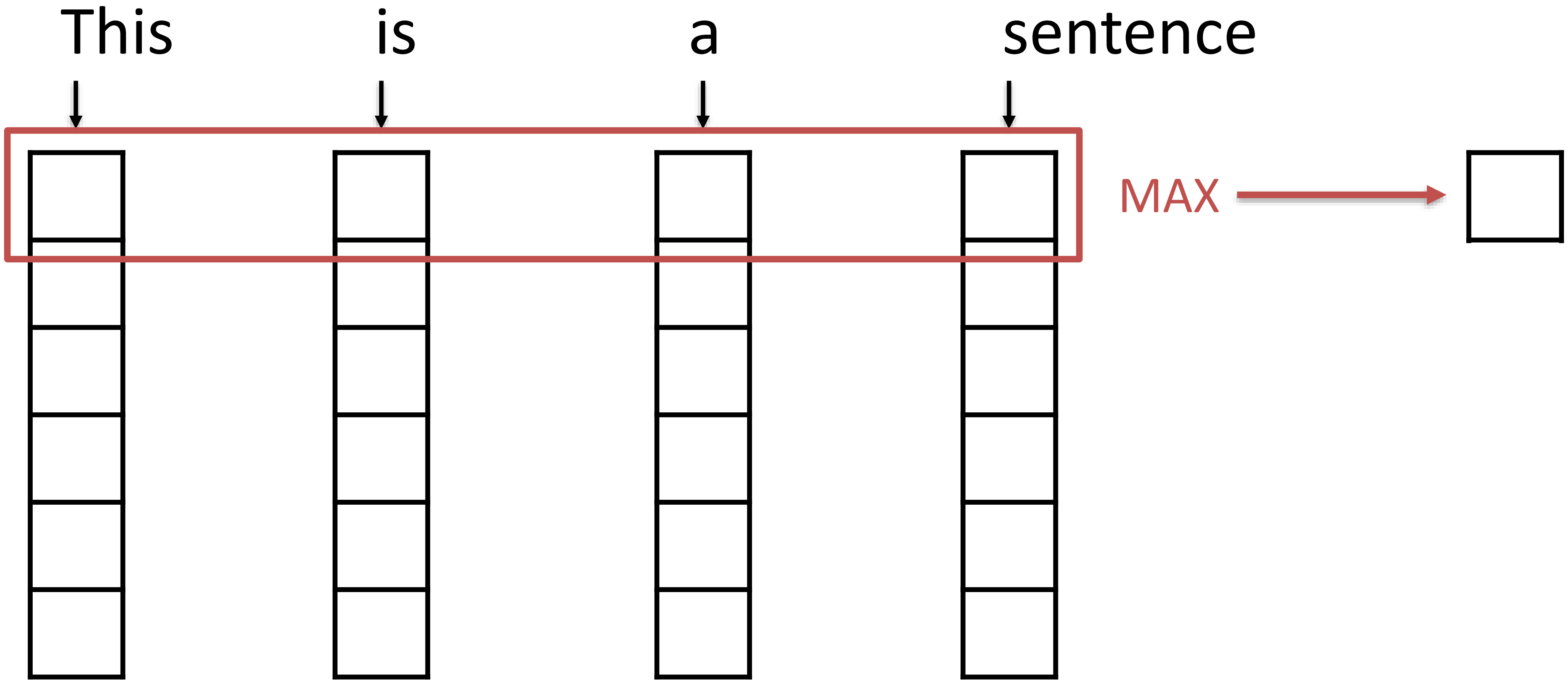
- Look up words individually to obtain their vectors
- Construct a <u>sequence</u> of vectors



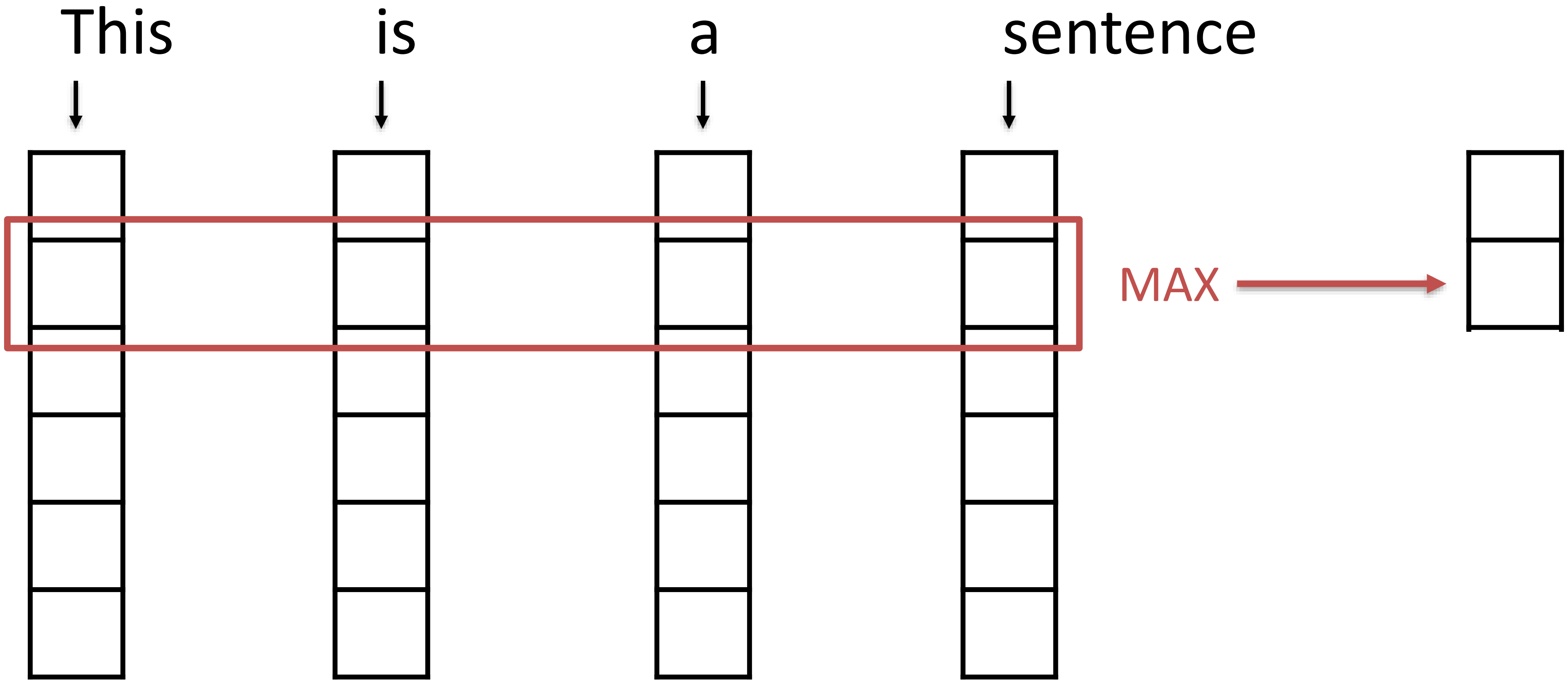
VSWEM Step 1: Convert sentence to vectors



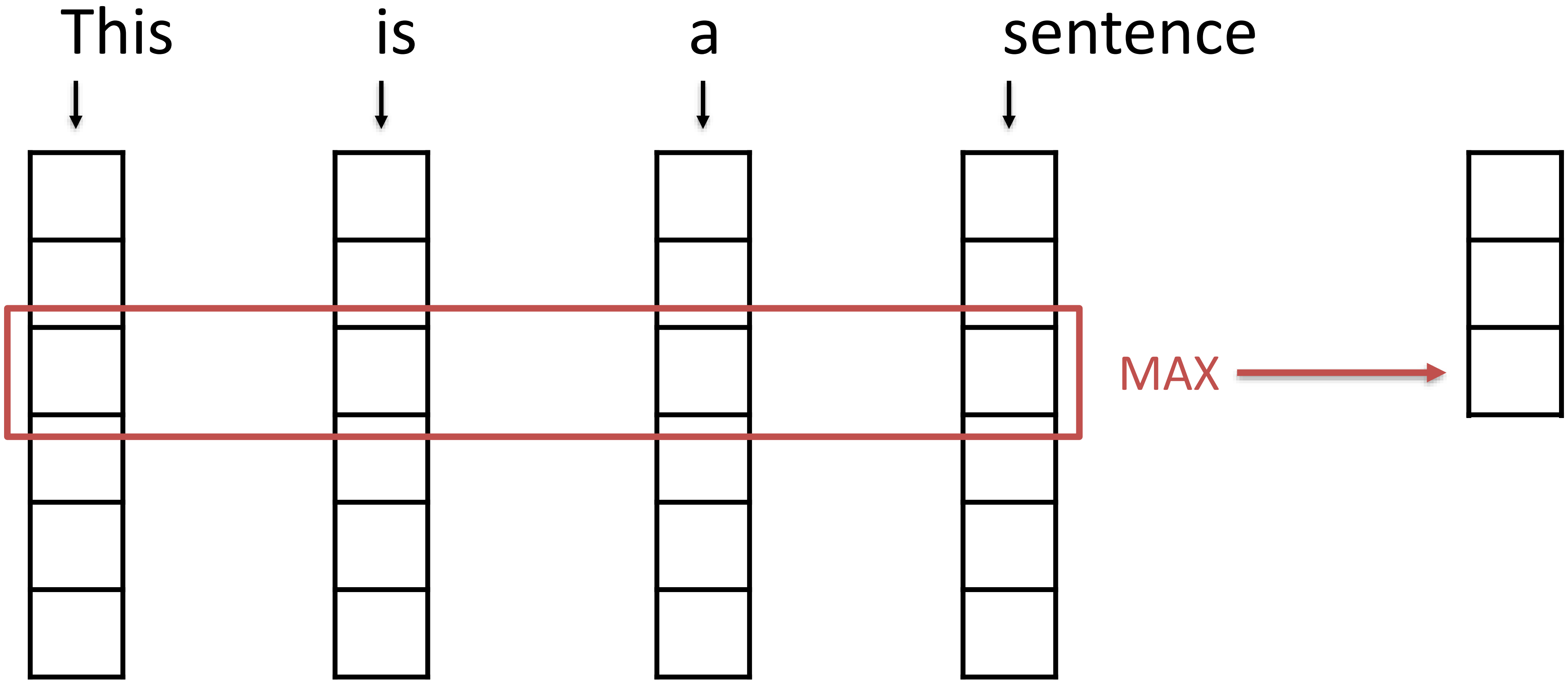
VSWEM Step 2: Take the MAX over the sentence for each embedding dimension



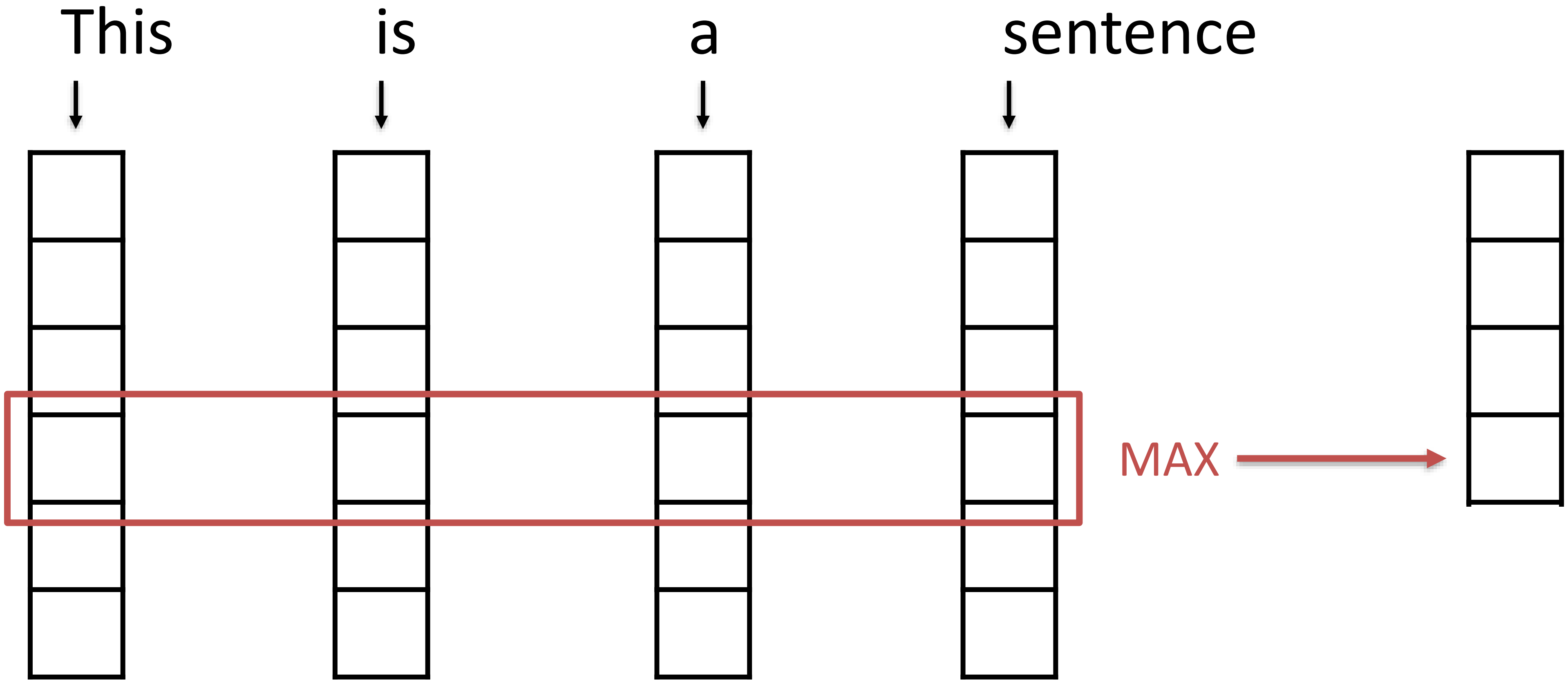
VSWEM Step 2: Take the MAX over the sentence for each embedding dimension



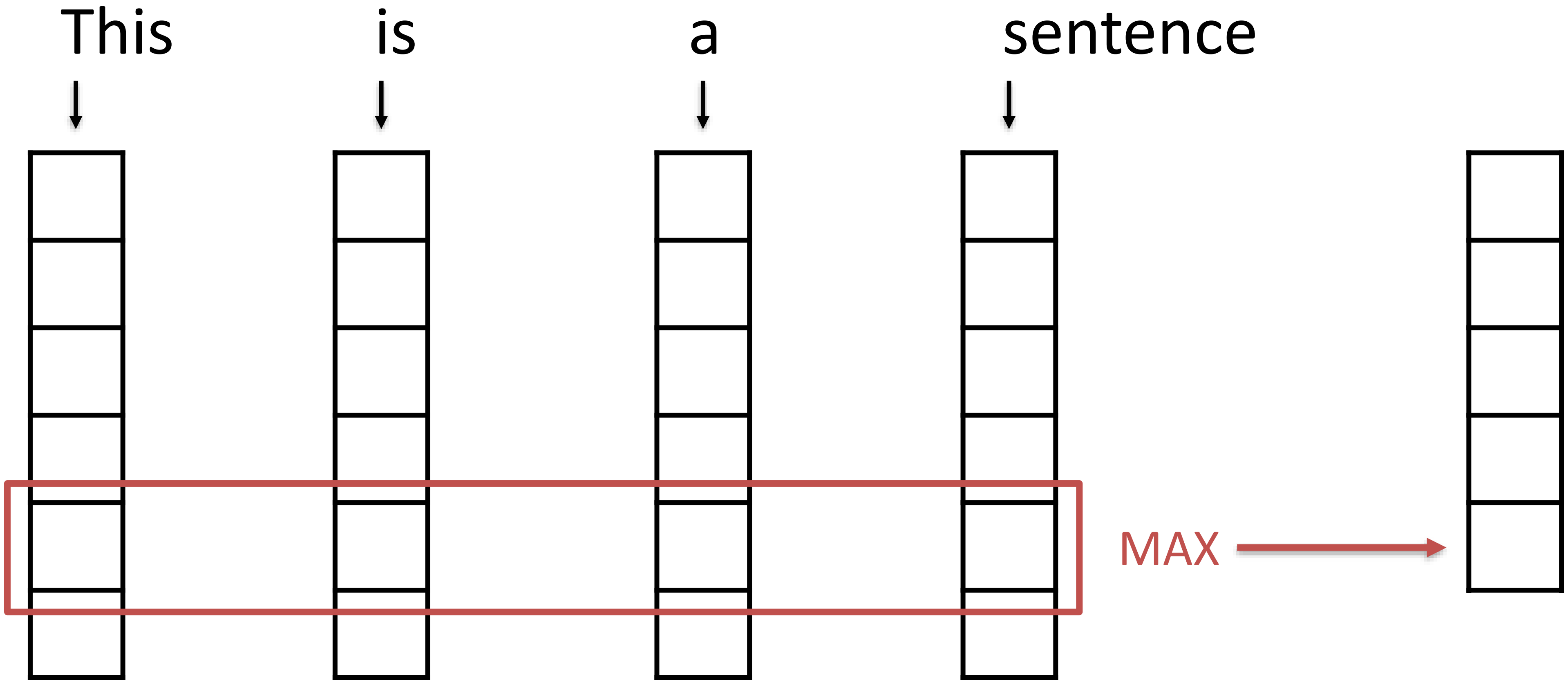
VSWEM Step 2: Take the MAX over the sentence for each embedding dimension



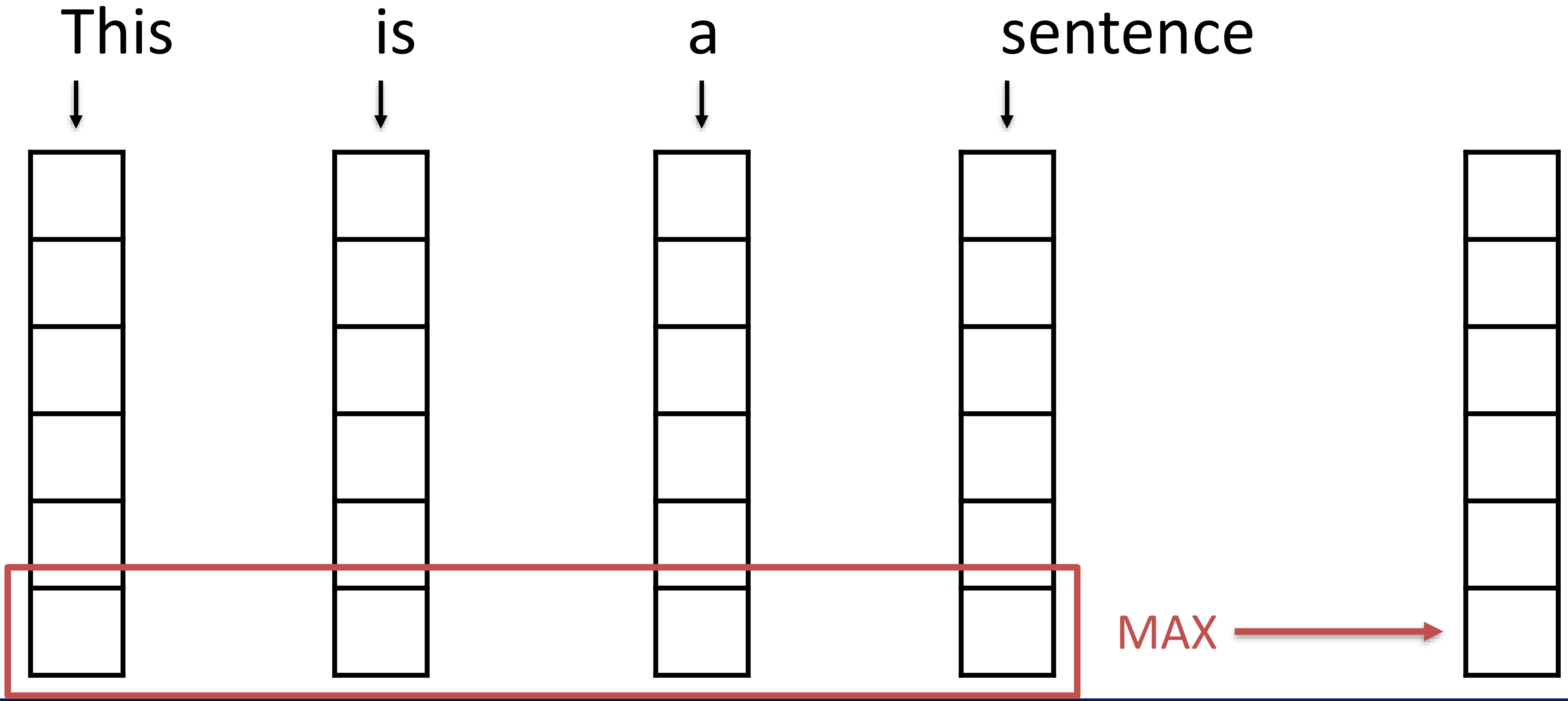
VSWEM Step 2: Take the MAX over the sentence for each embedding dimension



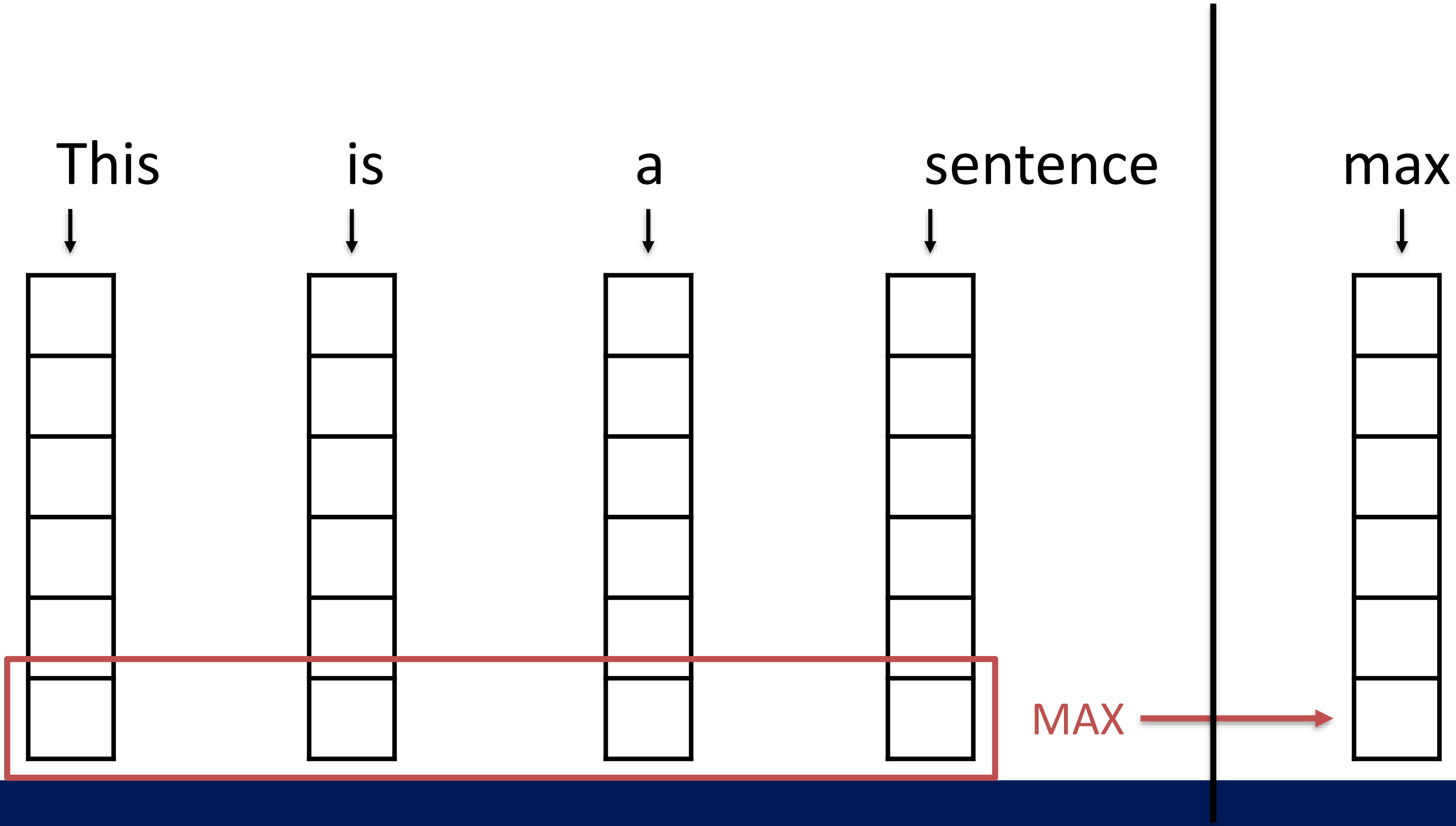
VSWEM Step 2: Take the MAX over the sentence for each embedding dimension



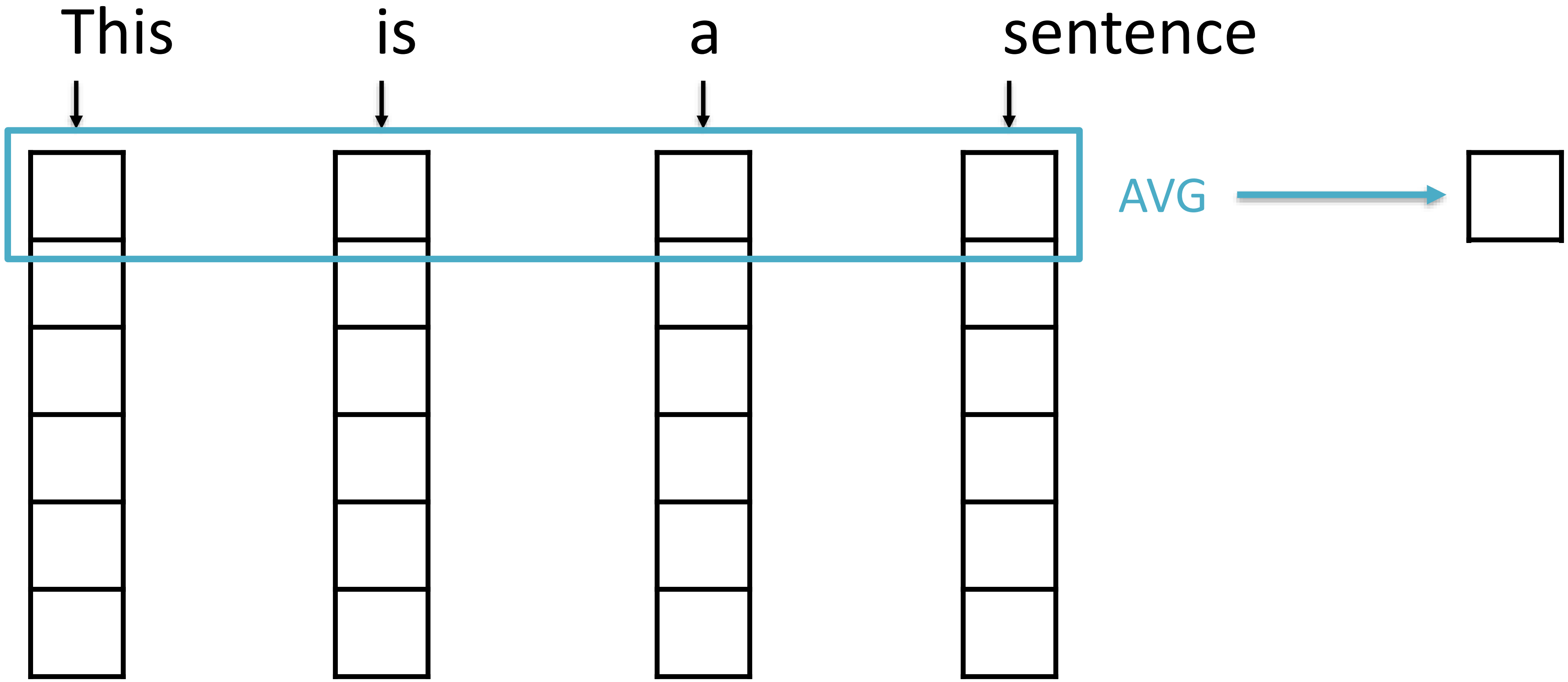
VSWEM Step 2: Take the MAX over the sentence for each embedding dimension



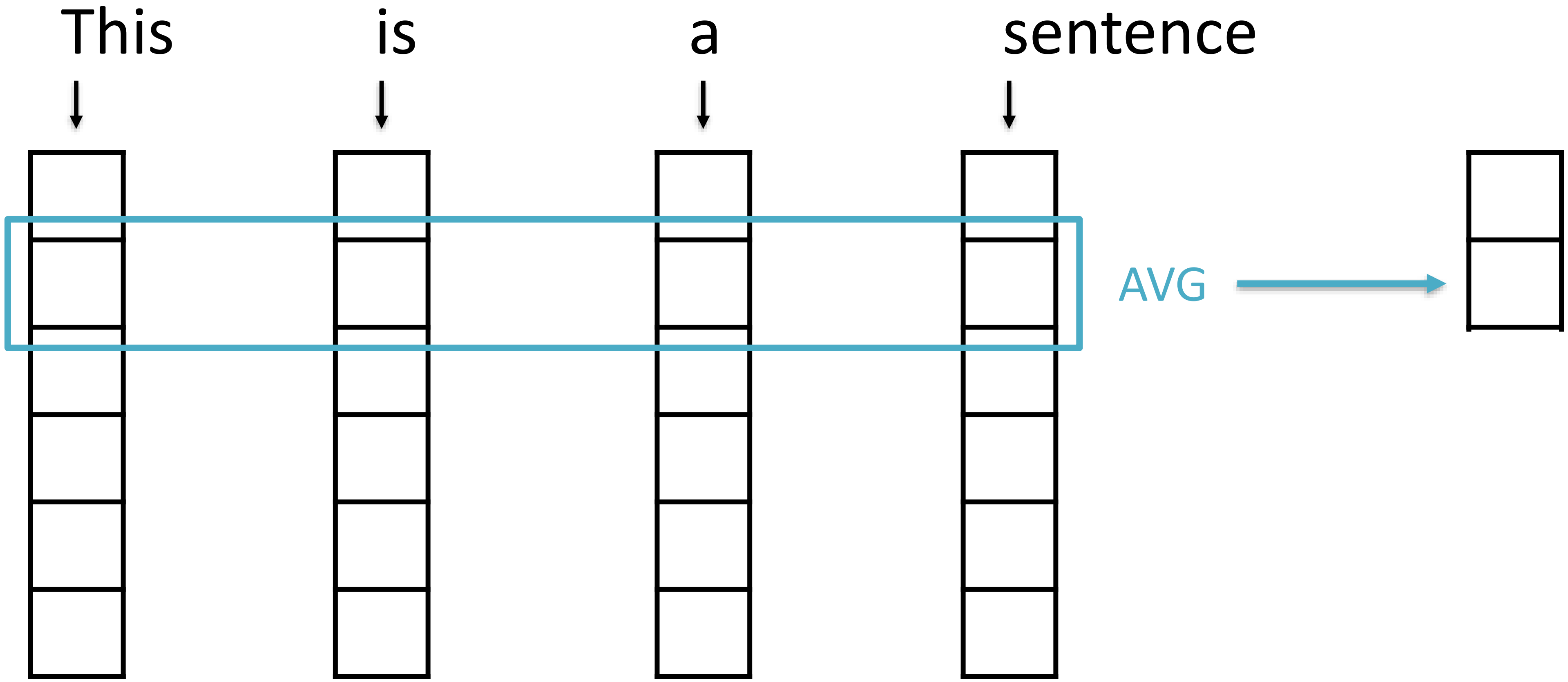
VSWEM Step 2: Take the MAX over the sentence for each embedding dimension



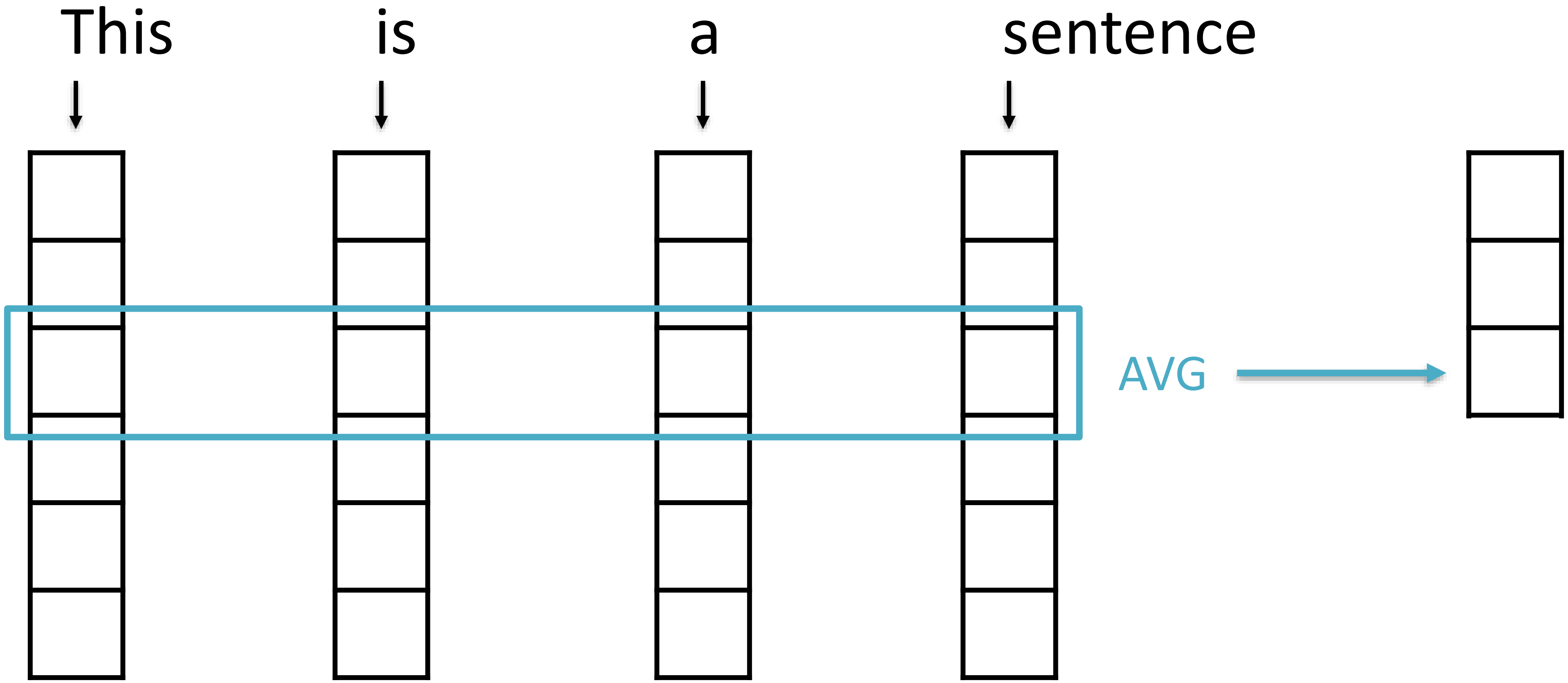
VSWEM Step 3: Take the AVERAGE over the sentence for each embedding dimension



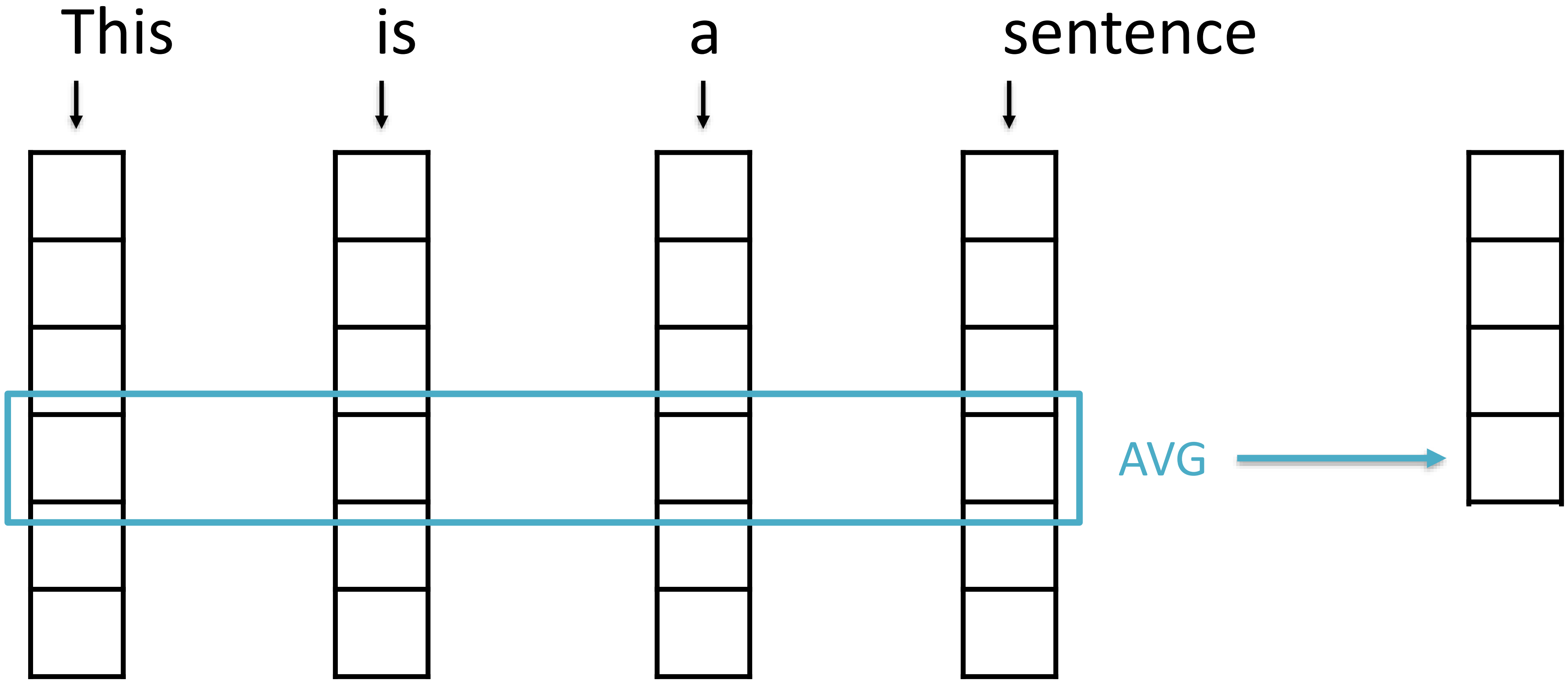
VSWEM Step 3: Take the AVERAGE over the sentence for each embedding dimension



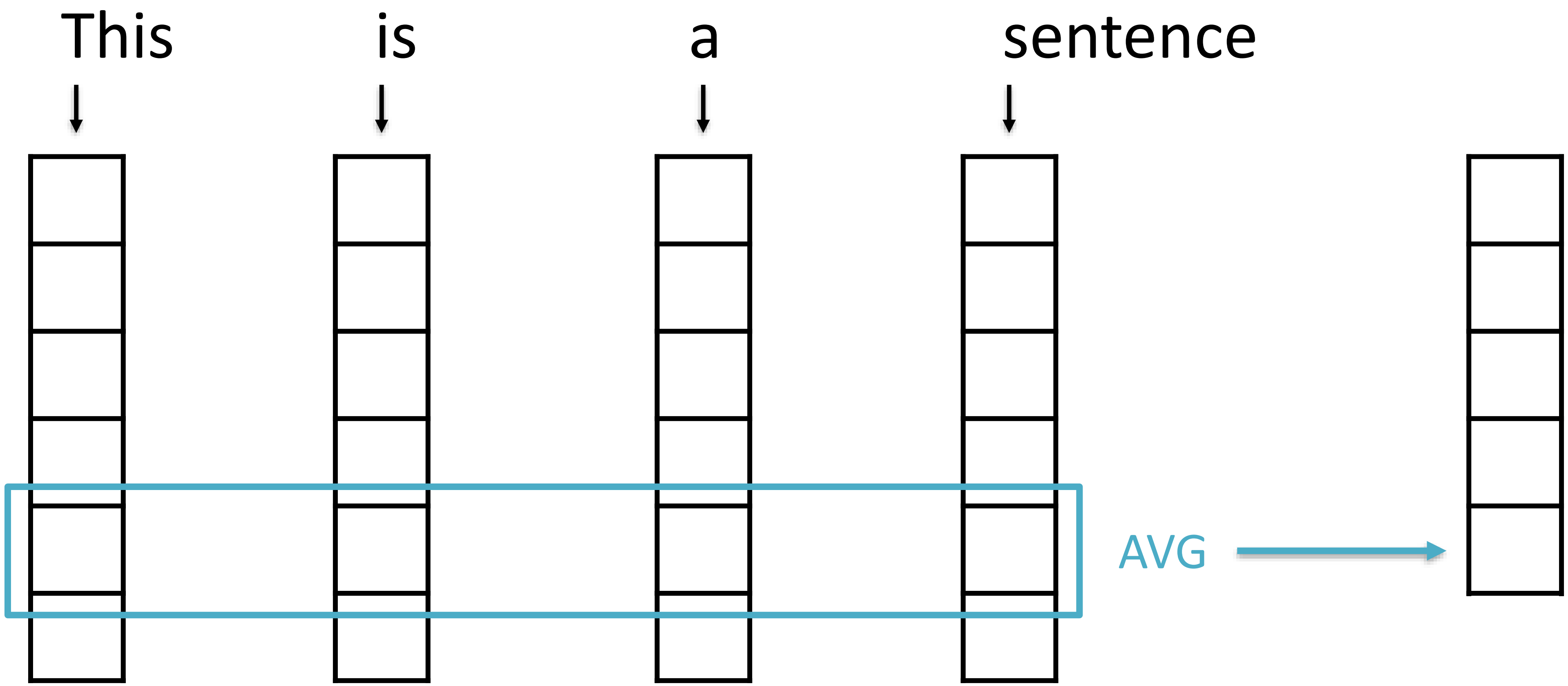
VSWEM Step 3: Take the AVERAGE over the sentence for each embedding dimension



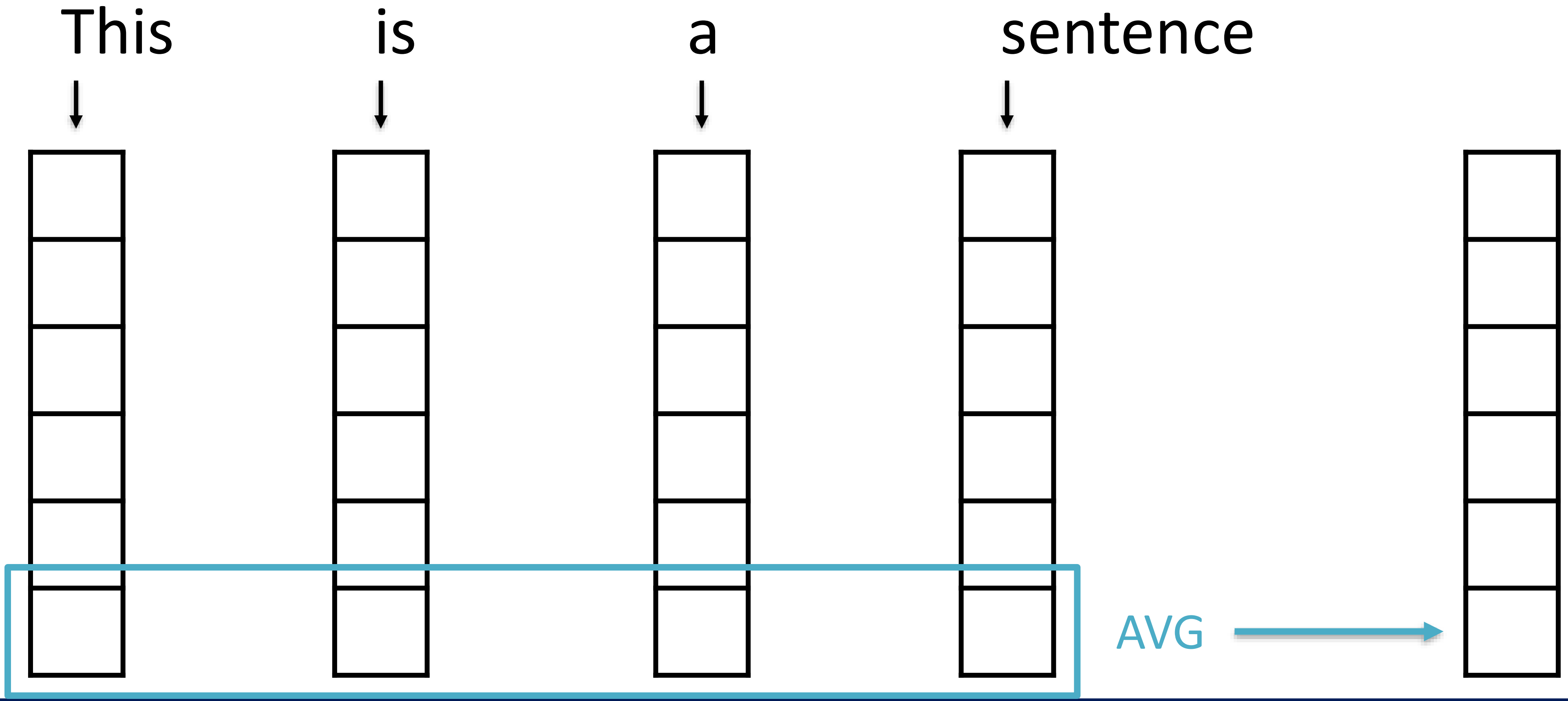
VSWEM Step 3: Take the AVERAGE over the sentence for each embedding dimension



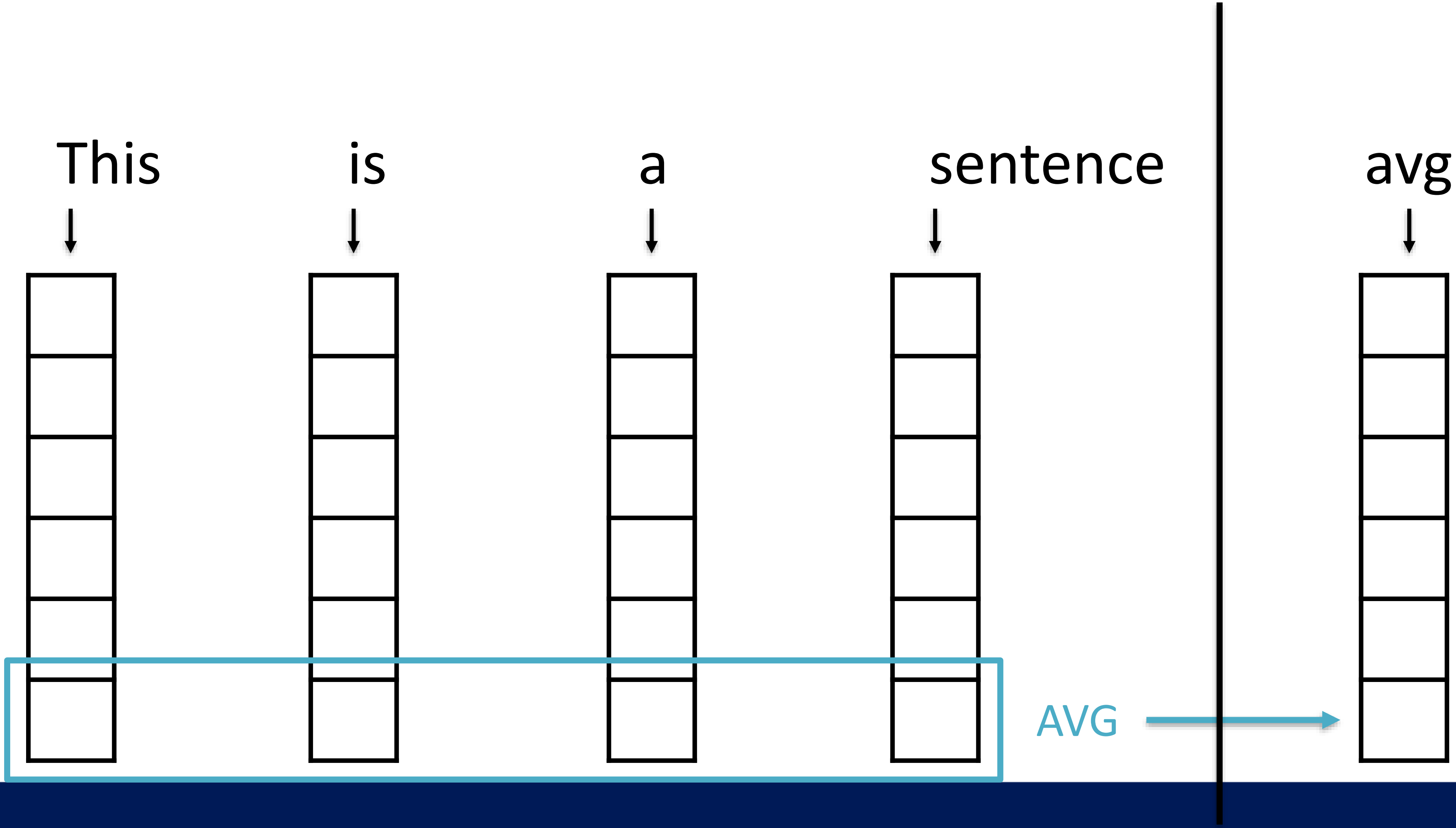
VSWEM Step 3: Take the AVERAGE over the sentence for each embedding dimension



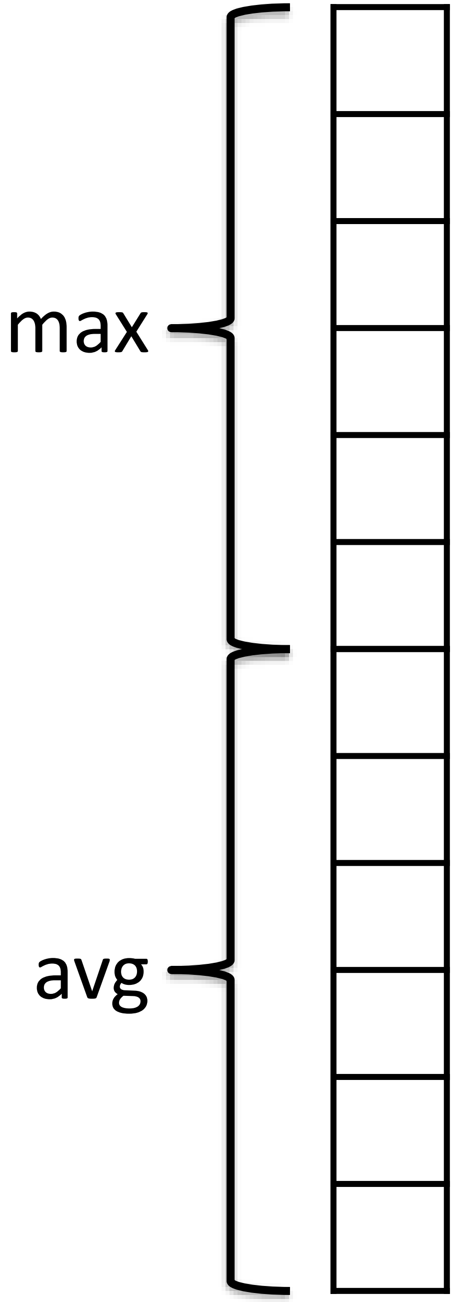
VSWEM Step 3: Take the AVERAGE over the sentence for each embedding dimension



VSWEM Step 3: Take the AVERAGE over the sentence for each embedding dimension



VSWEM Step 4: Concatenate MAX and AVG



Sentence i

This is a sentence

x_i

<u>Question</u>:

What information are we *losing* when we do this?

Bag of Words and VSWEM for medical abstract classification

COMPUTATIONAL EXERCISE 3