Word Embeddings and A Very Simple Word Embedding Based Model

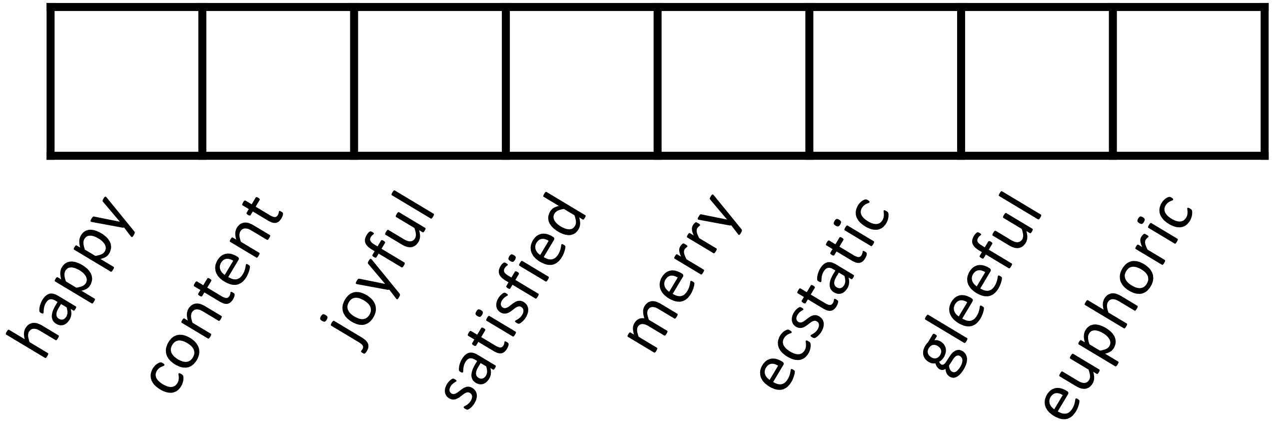
July 10, 2020

Applied Data Science
MMCi Term 4

Matthew Engelhard

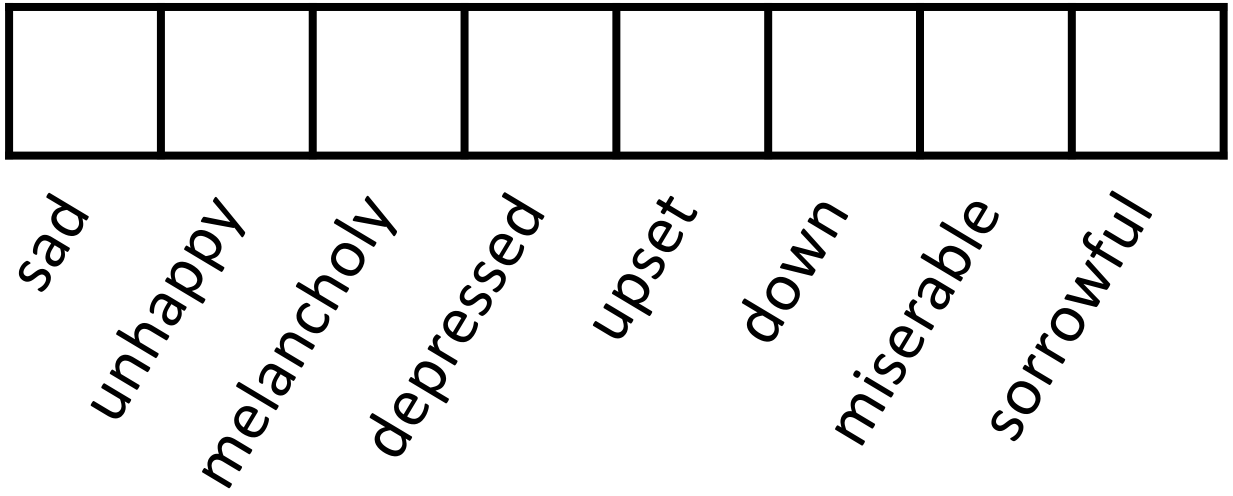
MOTIVATING WORD EMBEDDINGS

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

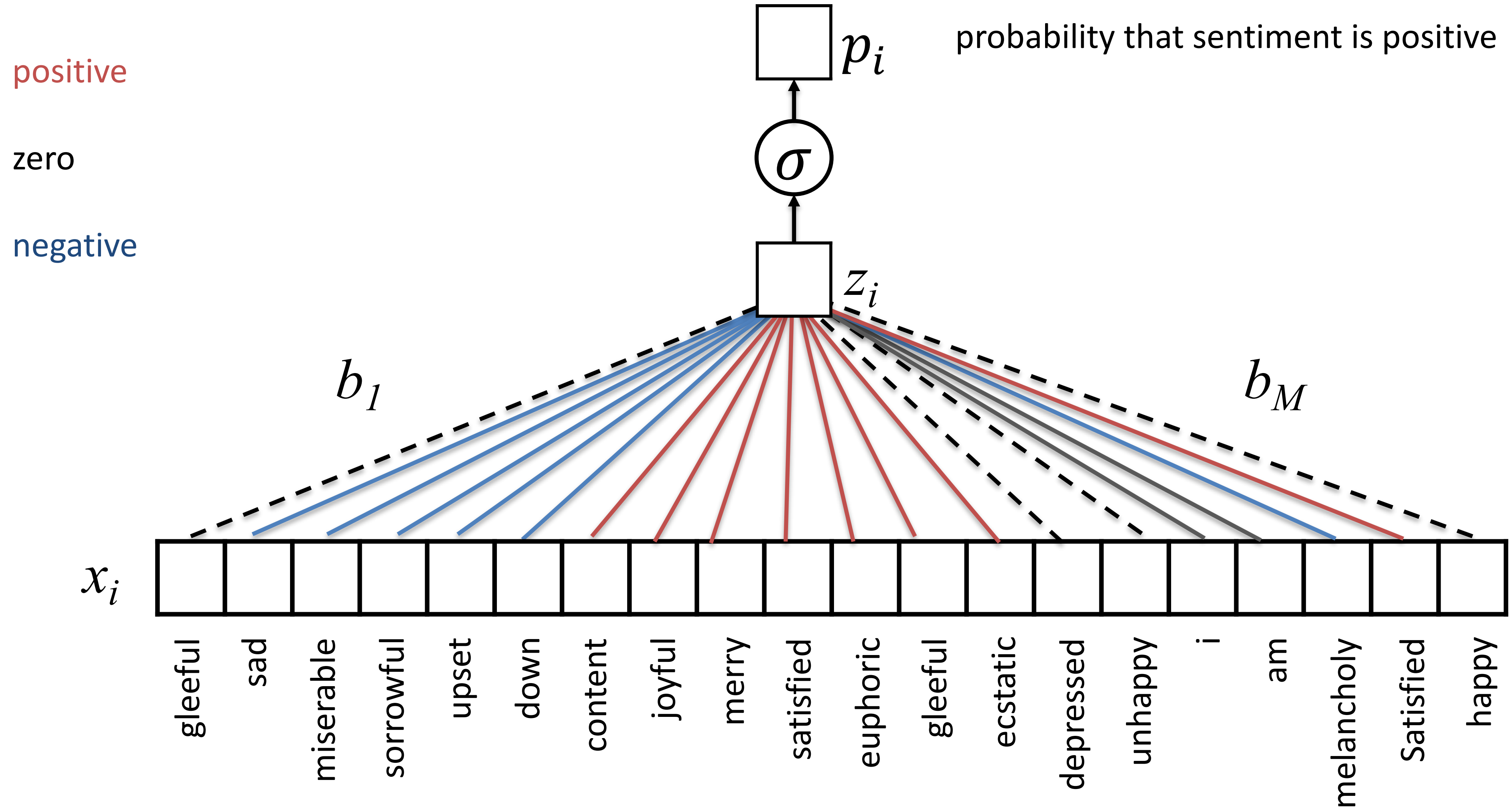


Goal: predict sentiment (positive/negative)

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

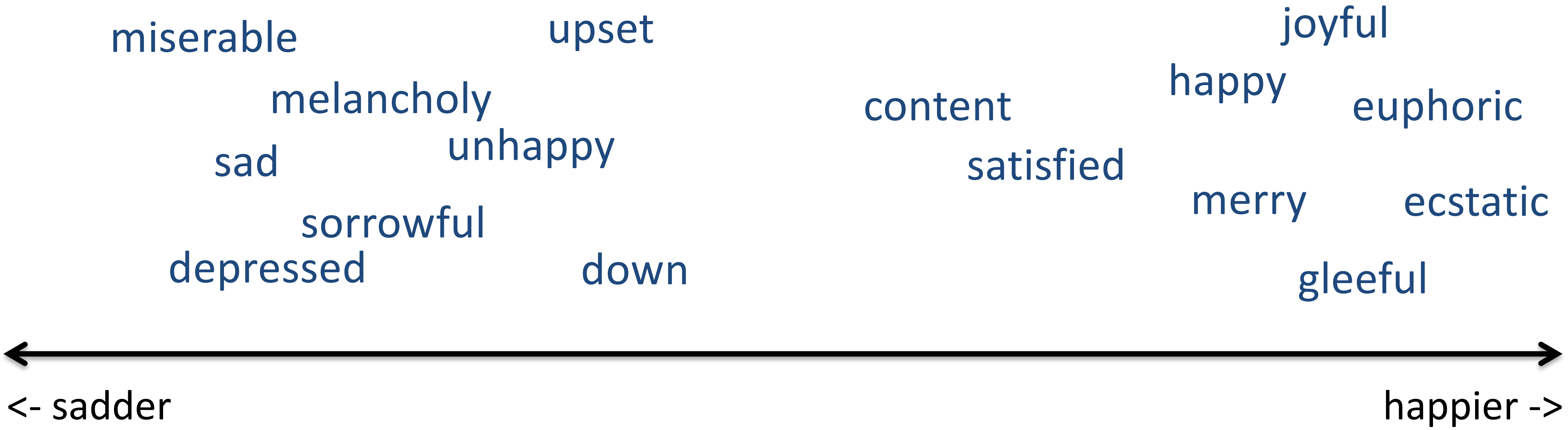
I am sad	🙁	training set
I am miserable	🙁	
I am sorrowful	🙁	
I am upset	🙁	
I am down	🙁	
I am content	🙂	
I am joyful	🙂	
I am merry	🙂	
I am satisfied	🙂	
I am euphoric	🙂	
I am depressed	?	test set
I am unhappy	?	
I am happy	?	
I am gleeful	?	

logistic regression: positive / negative sentiment



I passed out and Mom said I was shaking

We'd like a numeric representation of words that encodes their meaning



Numeric value indicating whether the word is happy or sad

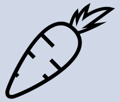
Training a robot to buy groceries

Example from Anand Chowdhury, MMCi 2019

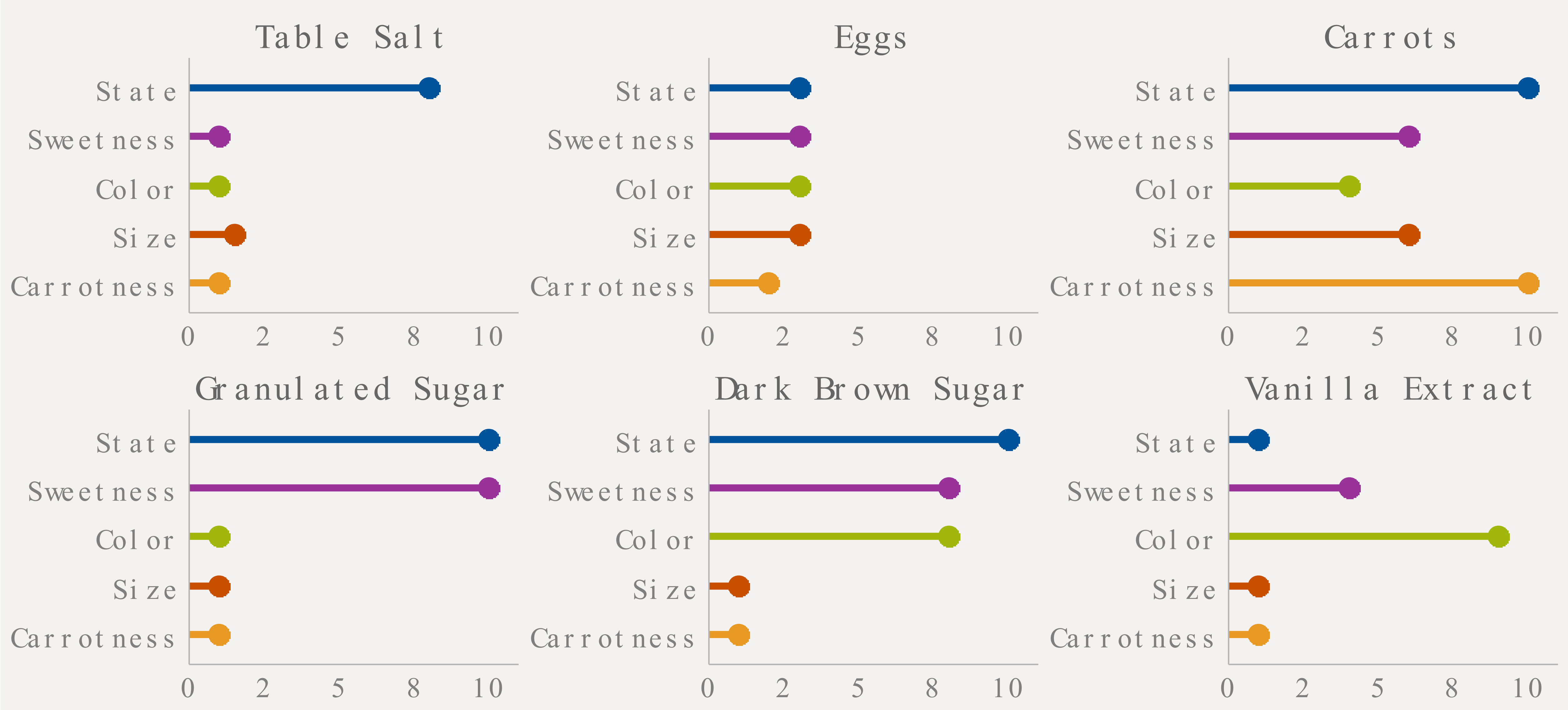
Grocery List

- ❑ granulated sugar
- ❑ vanilla extract
- ❑ dark brown sugar
- ❑ carrots
- ❑ table salt
- ❑ eggs

Characteristics/Dimensions

Dimension	1	10
State	Liquid	Solid
Sweetness	Bland	Sweet
Color	Light	Dark
Size	Small	Large
Carrotness	Not really	

Five dimensions



Make Sense of Items not Seen Before

Item	State	Sweetness	Color	Size	Carrotness
???	0	8	7	6	0
???	0	0	10	6	0
???	8	9	8	3	0
???	0	5	3	4	10

Make Sense of Items not Seen Before

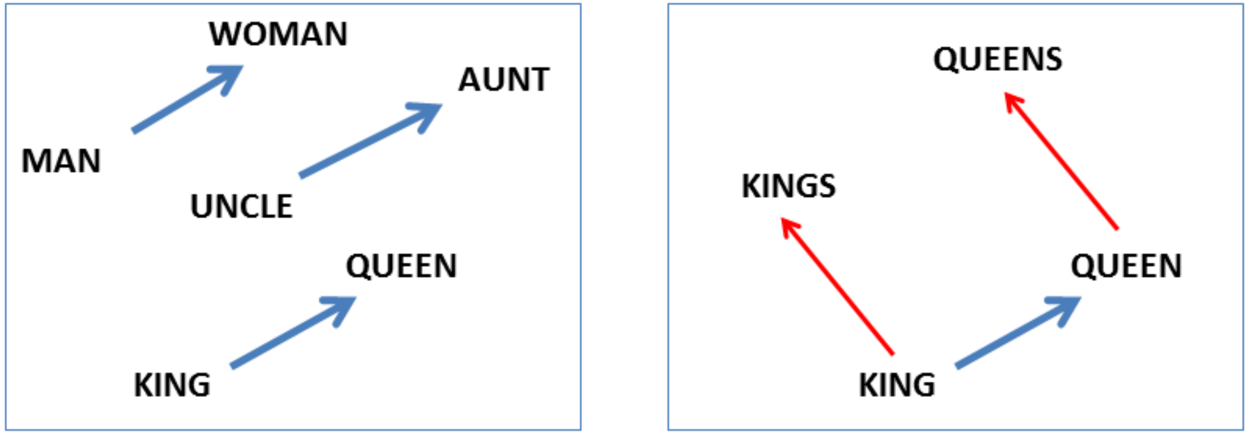
Item	State	Sweetness	Color	Size	Carrotness
Soda / Sweet Tea	0	8	7	6	0
Black Coffee	0	0	10	6	0
Chocolate	8	9	8	3	0
Carrot Juice	0	5	3	4	10

Recipe

Dark Brown Sugar – Granulated Sugar + Carrots

	Item	State	Sweetness	Color	Size	Carrotness
	Dark Brown Sugar	10	8	8	1	1
-	Granulated Sugar	10	10	1	1	1
+	Carrots	10	6	4	6	10
=	???	10	4	11	6	10

Word Embeddings: Assign Each Word in our Vocabulary to a Numeric Vector (of characteristics)

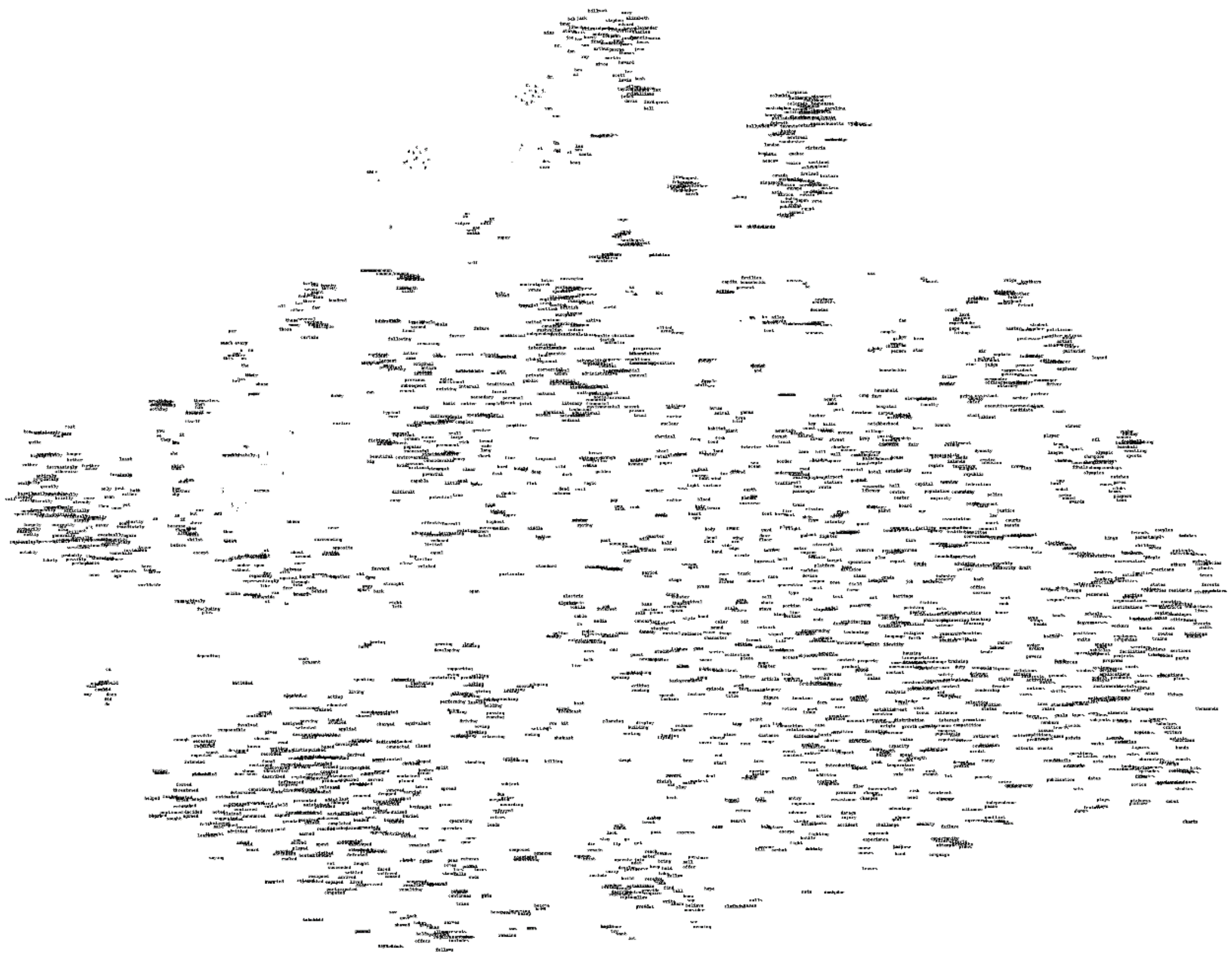


Dimension	1	10
Gender	Male	Female
Class	Commoner	Royalty
Plural	One	Many

Visualizing Word Embeddings

Here we show the learned numeric representations (limited here to 2 dimensions) of many different vocabulary words

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

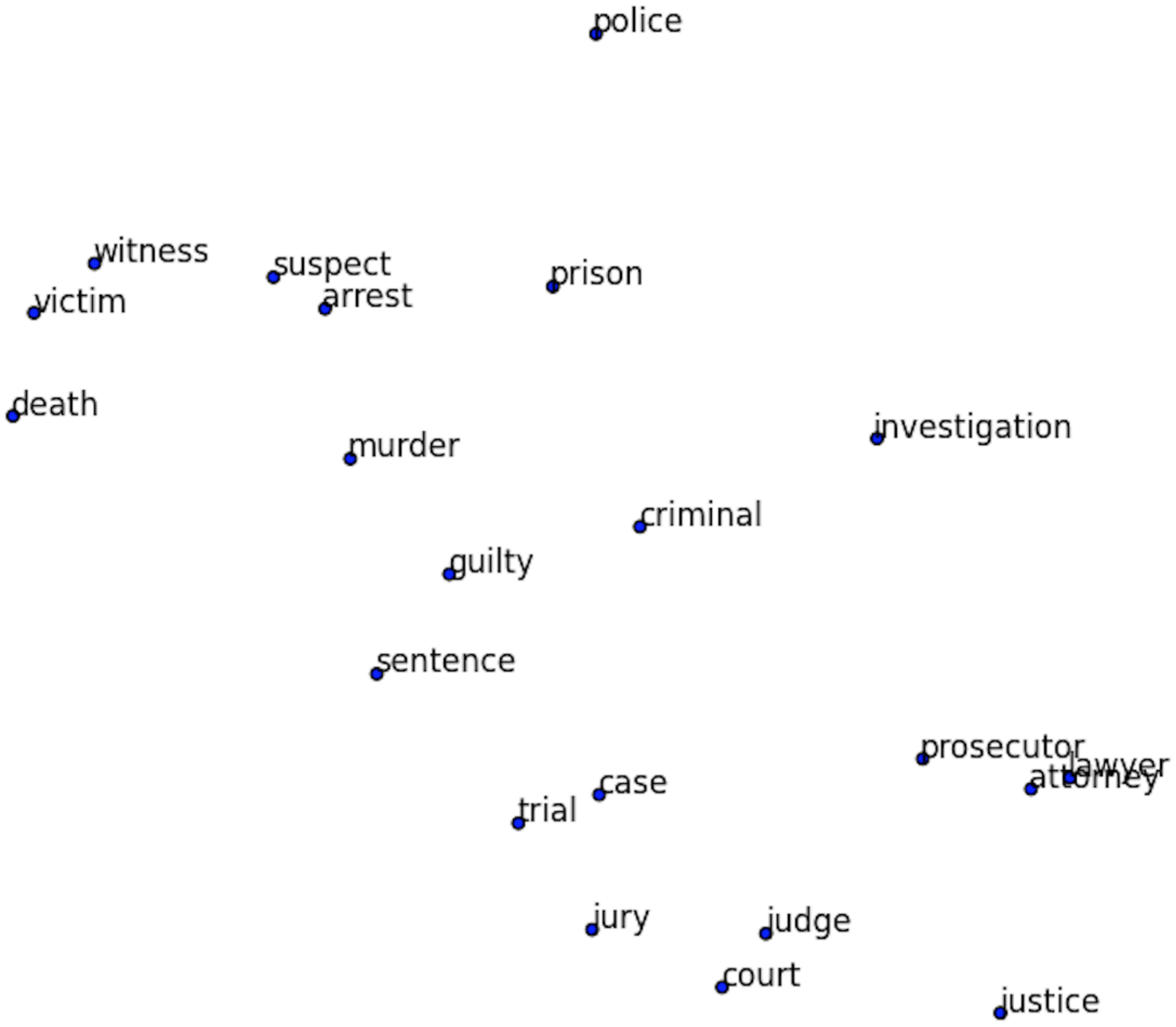


Visualizing Word Embeddings

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 right next to each other – they have almost identical characteristics!

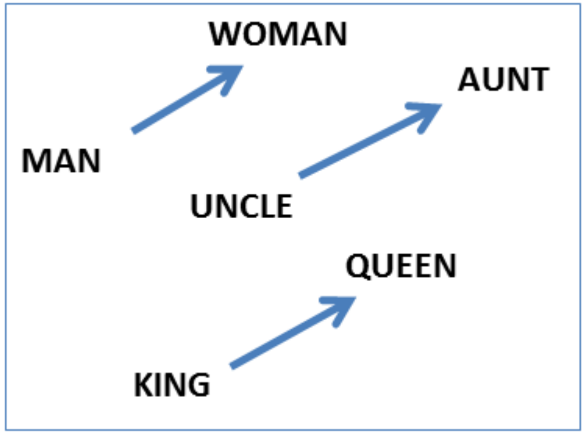


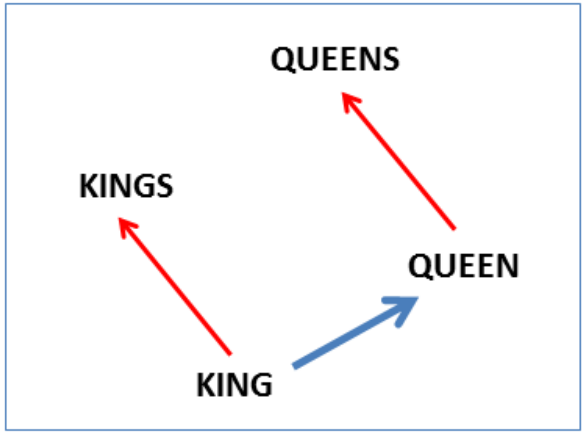
Word Recipes

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.

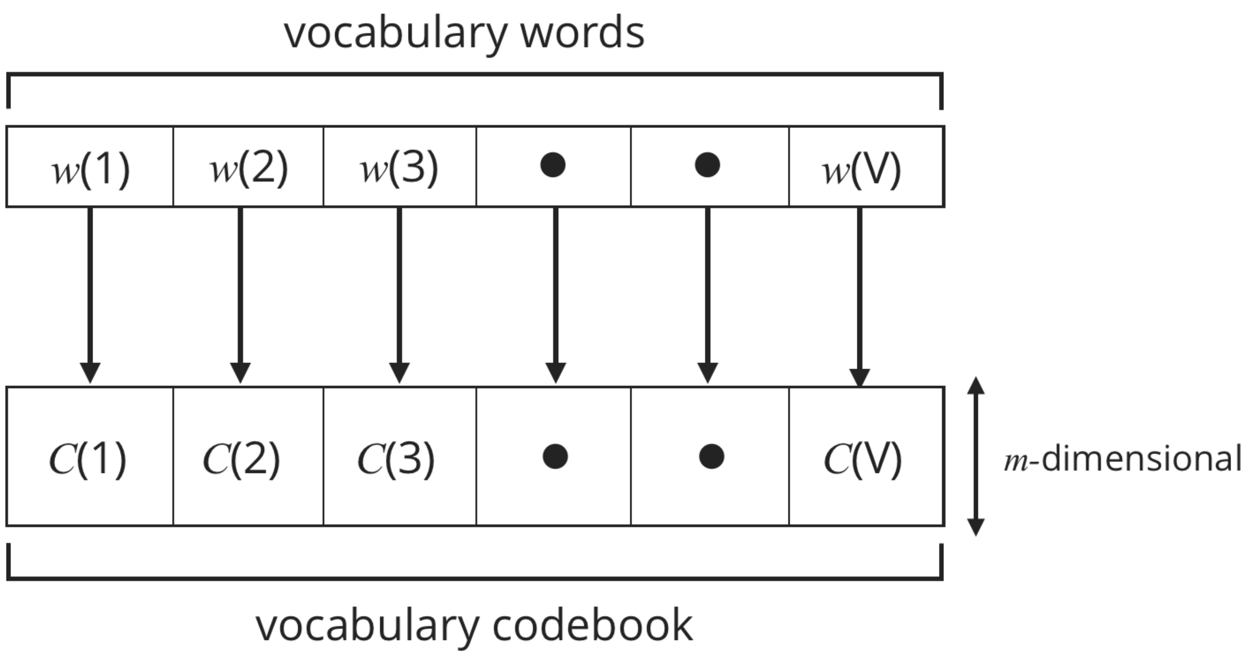
Plural relationships are also maintained.





Word to Vector (Word2Vec)

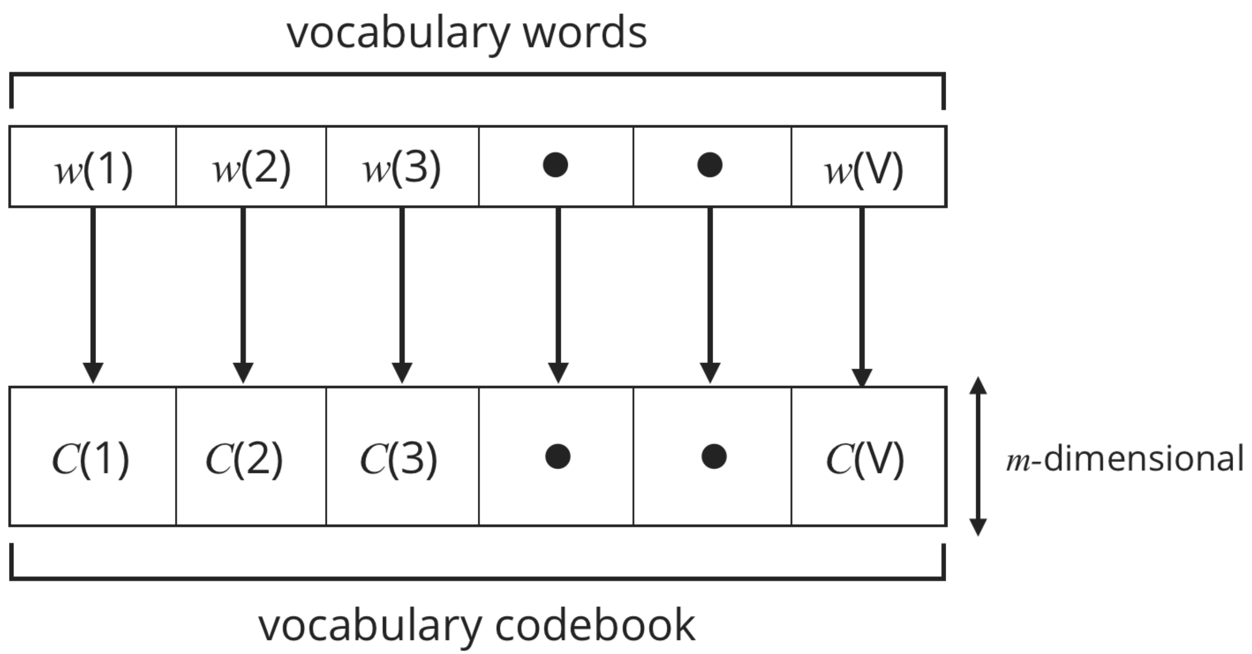
- Assigns words to numeric vectors
- Many characteristics/dimensions (~300) to capture complexity of meaning
- Do this by creating a lookup table for each word in our *vocabulary* (i.e. all the words we know). In code, the lookup table is implemented as a dictionary.



Word to Vector (Word2Vec)

- Enter the word into the dictionary

- Receive the vector, or “embedding”, for that word



Note: we can also do this with categorical variables!

- Locations (city/state)
- Dx and procedure codes
- Medical concepts

- *What attributes could be used to encode the meaning of medical concepts?*

Proceedings — AMIA Joint Summits on Translational Science

AMIA Jt Summits Transl Sci Proc. 2016; 2016: 41–50. PMCID: PMC5001761
Published online 2016 Jul 20. PMID: 27570647

Learning Low-Dimensional Representations of Medical Concepts

Youngduck Choi,[1] Chill Yi-I Chiu, MS,[1] and David Sontag, PhD[1]

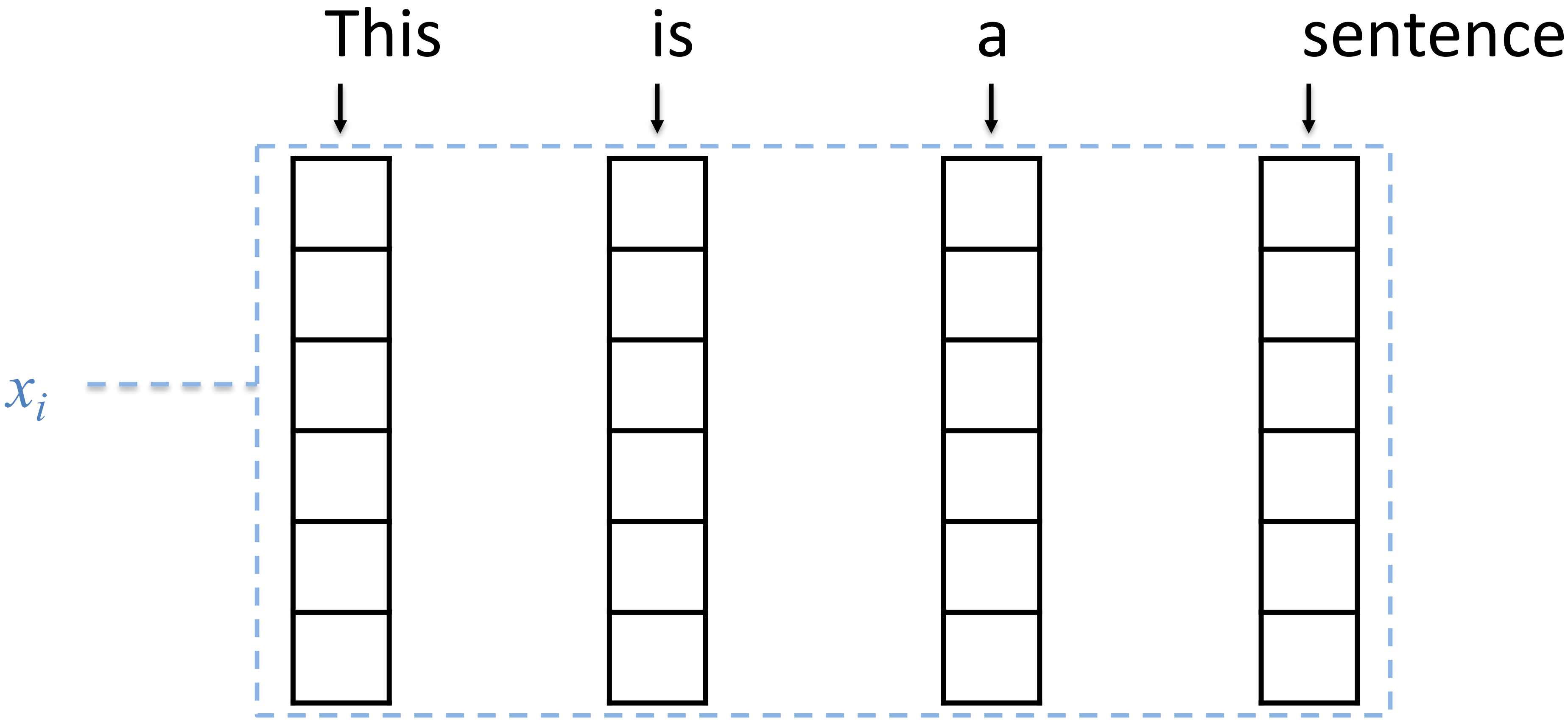
Abstract

Go to:

We show how to learn low-dimensional representations (embeddings) of a wide range of concepts in medicine, including diseases (e.g., ICD9 codes), medications, procedures, and laboratory tests. We expect that these embeddings will be useful across medical informatics for tasks such as cohort selection and patient summarization. These embeddings are learned using a technique called neural language modeling from the natural language processing community. However, rather than learning the embeddings solely from text, we show how to learn the embeddings from claims data, which is widely available both to providers and to payers. We also show that with a simple algorithmic adjustment, it is possible to learn medical concept embeddings in a privacy preserving manner from co-occurrence counts derived from clinical narratives. Finally, we establish a methodological framework, arising from standard medical ontologies such as UMLS, NDF-RT, and CCS, to further investigate the embeddings and precisely characterize their quantitative properties.

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

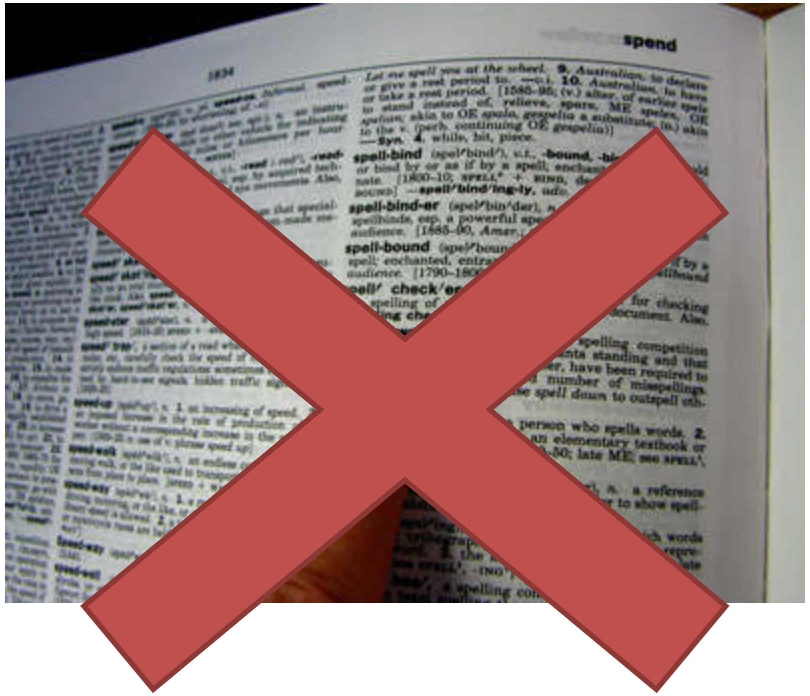
A crocodile strolls down the street

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/

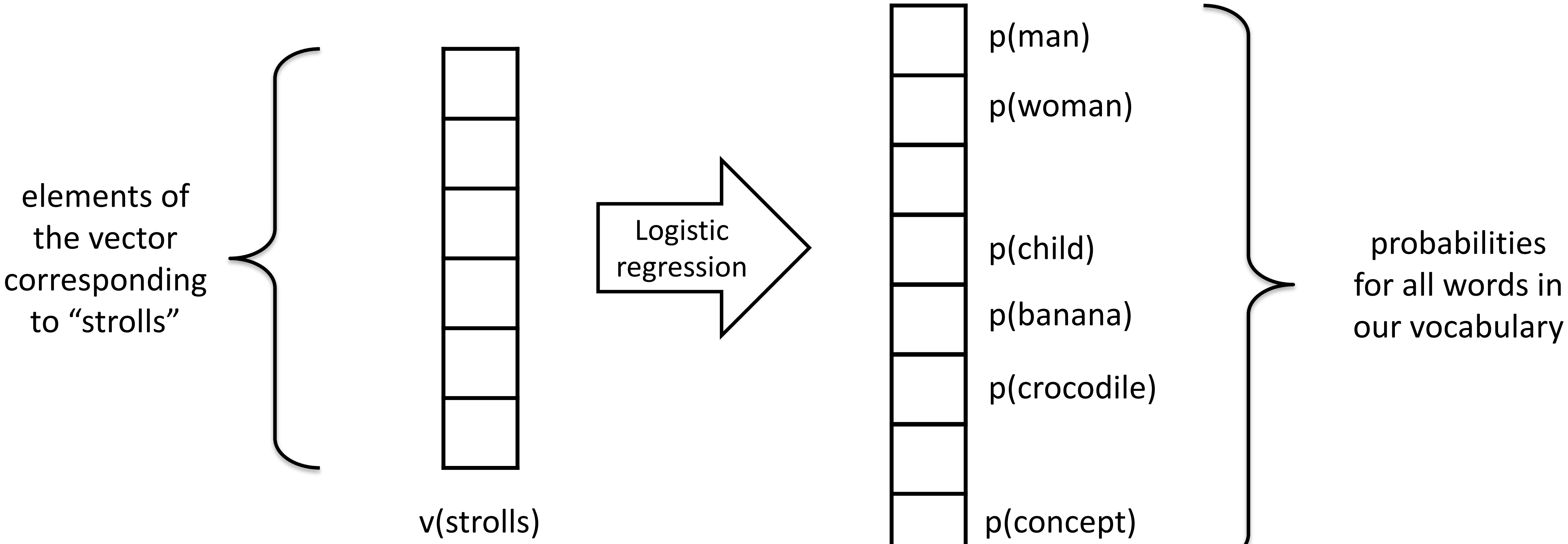
learn word meaning like an adult:
explicit definitions

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.

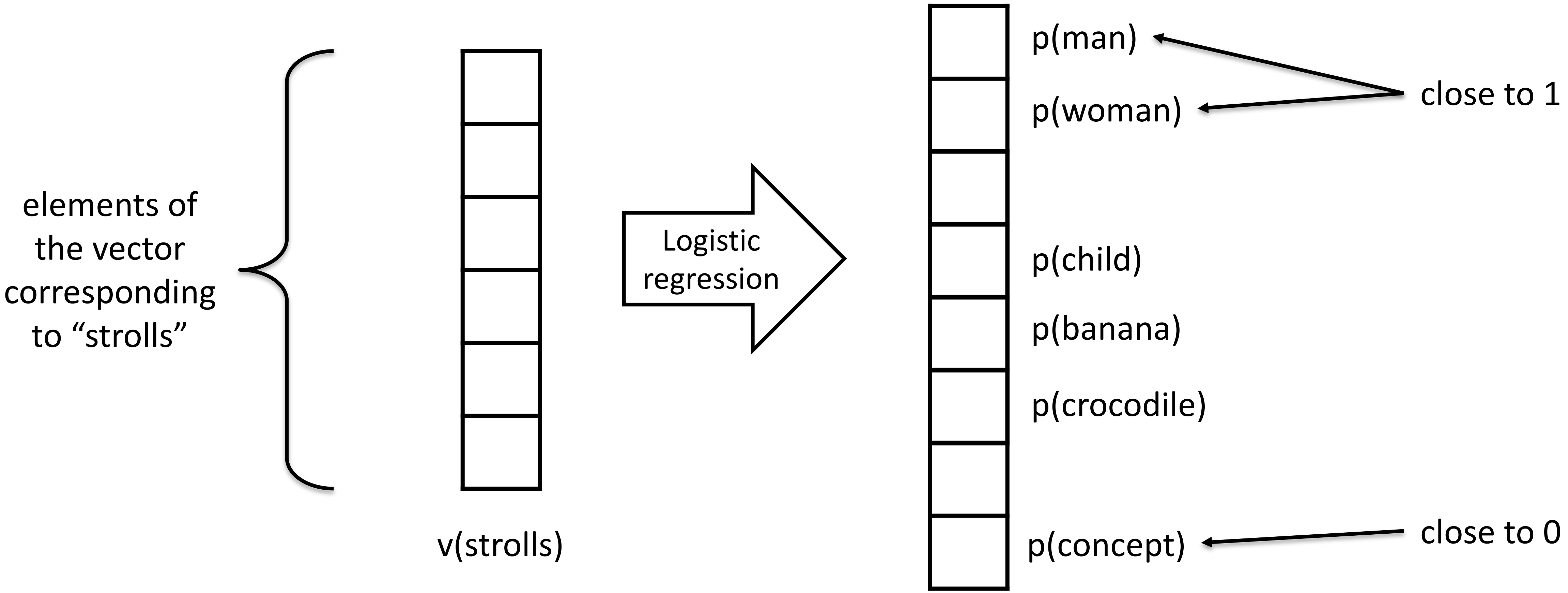
(e.g. Wikipedia)

A man strolls down the street.

input context

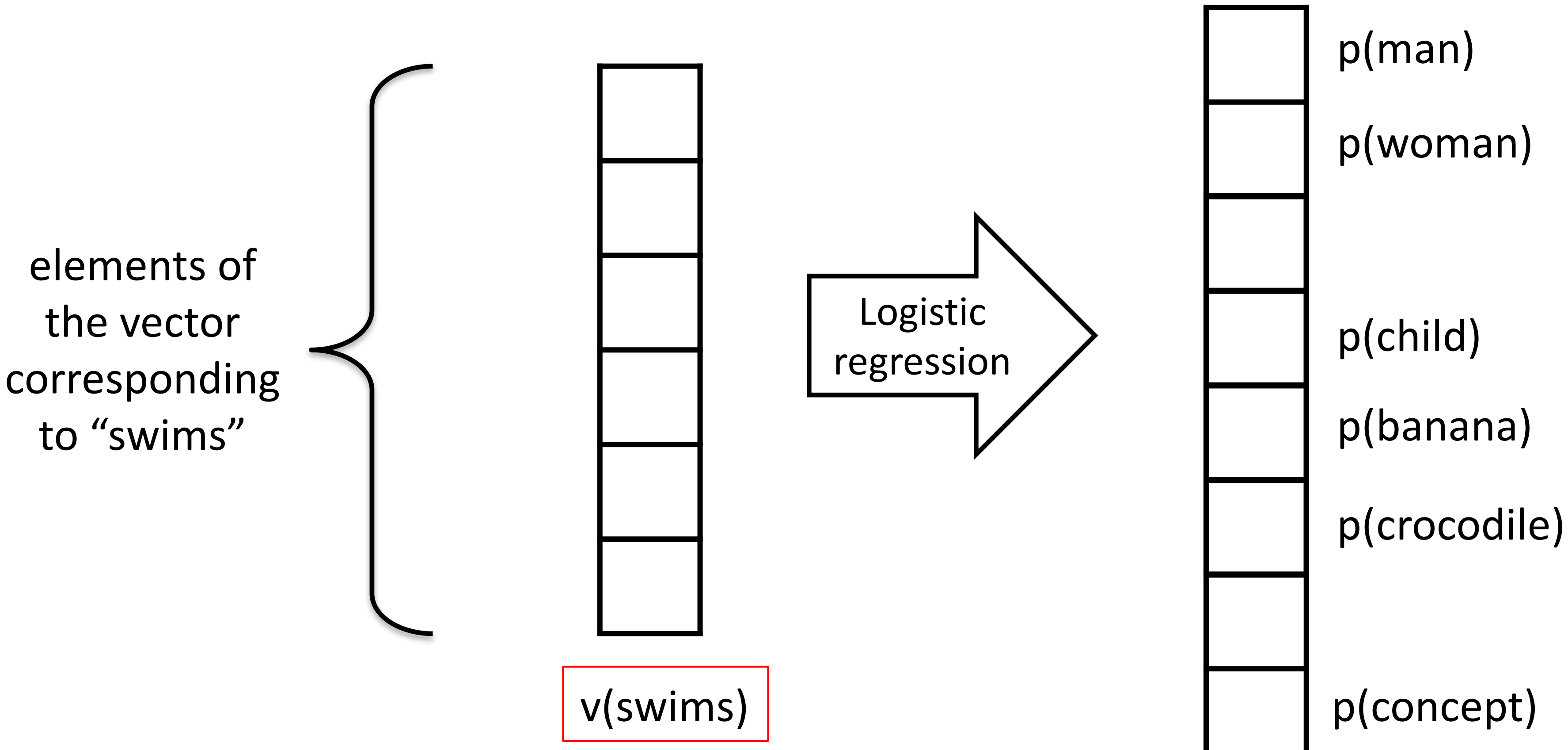
So how do we learn spatial locations for each word?

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)



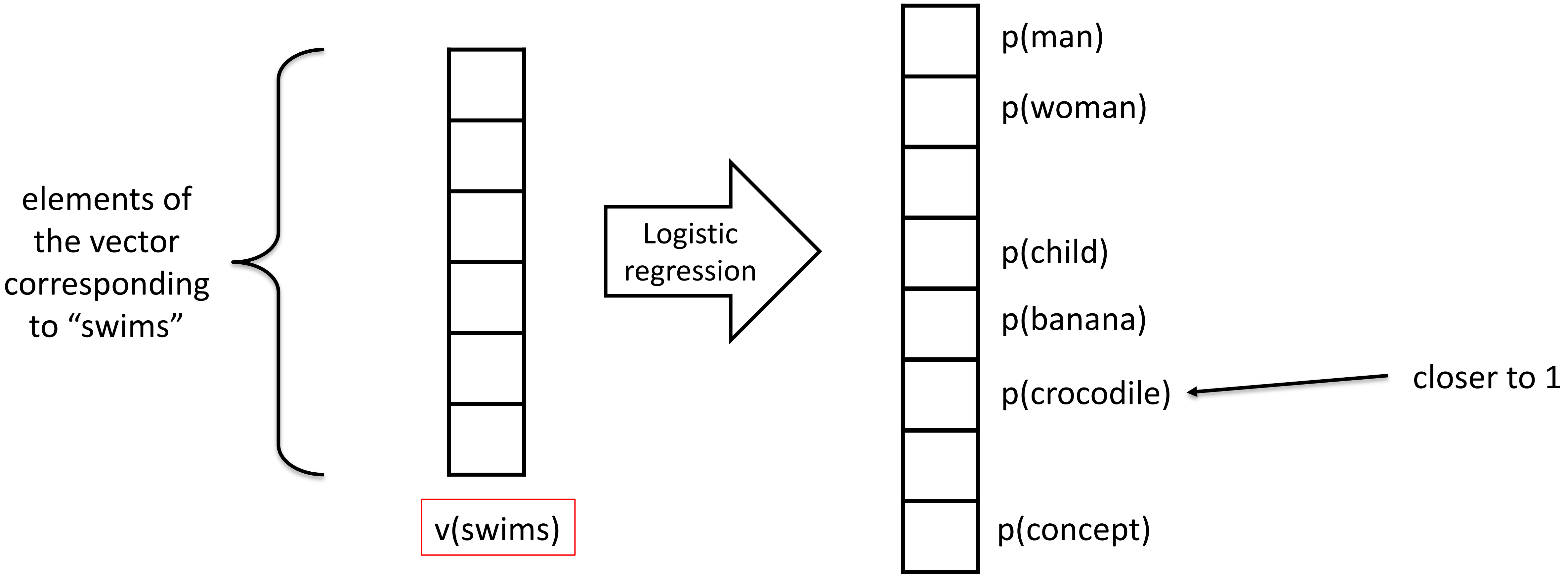
So how do we learn spatial locations for each word?

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?

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)

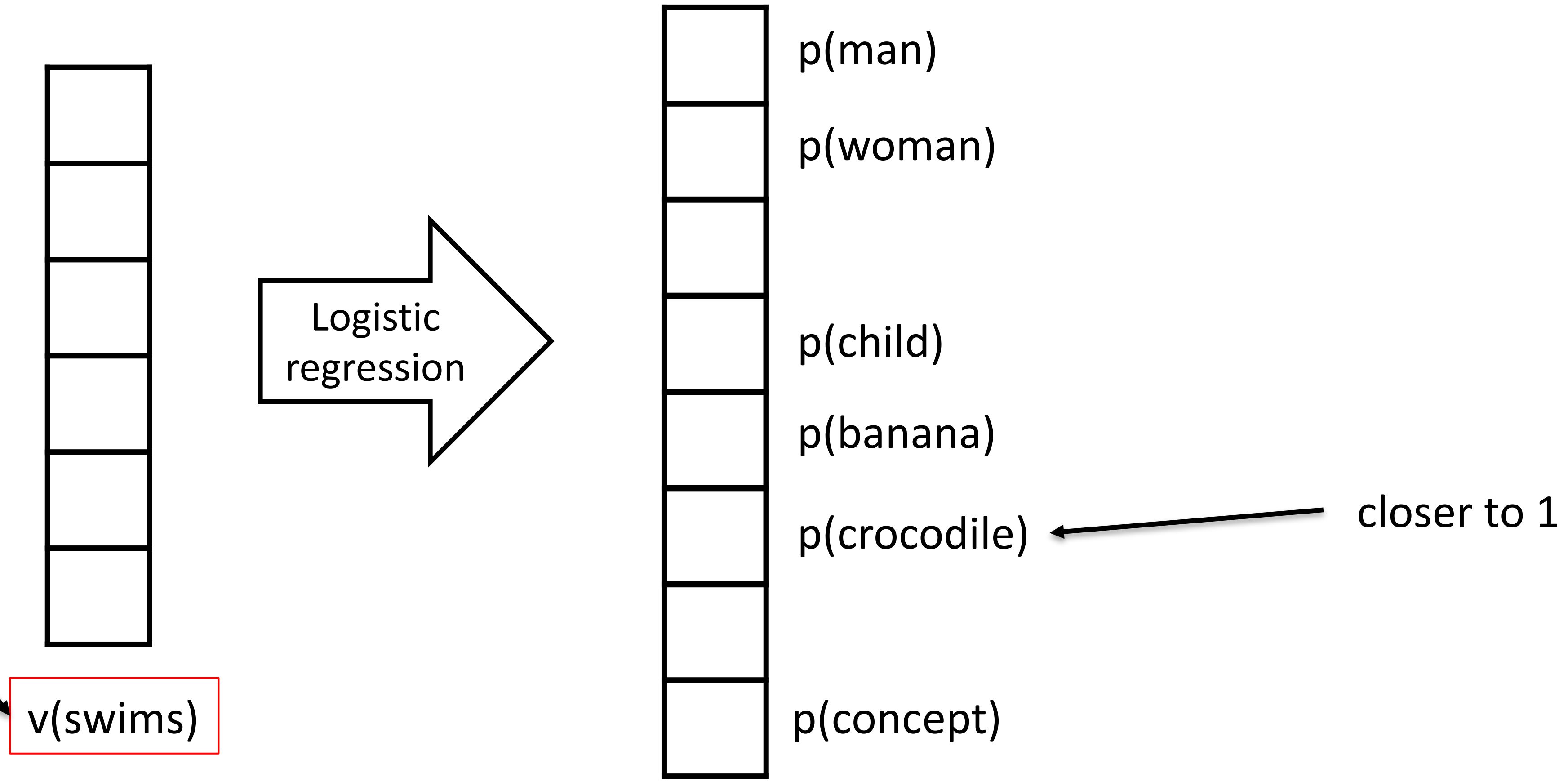


So how do we learn spatial locations for each word?

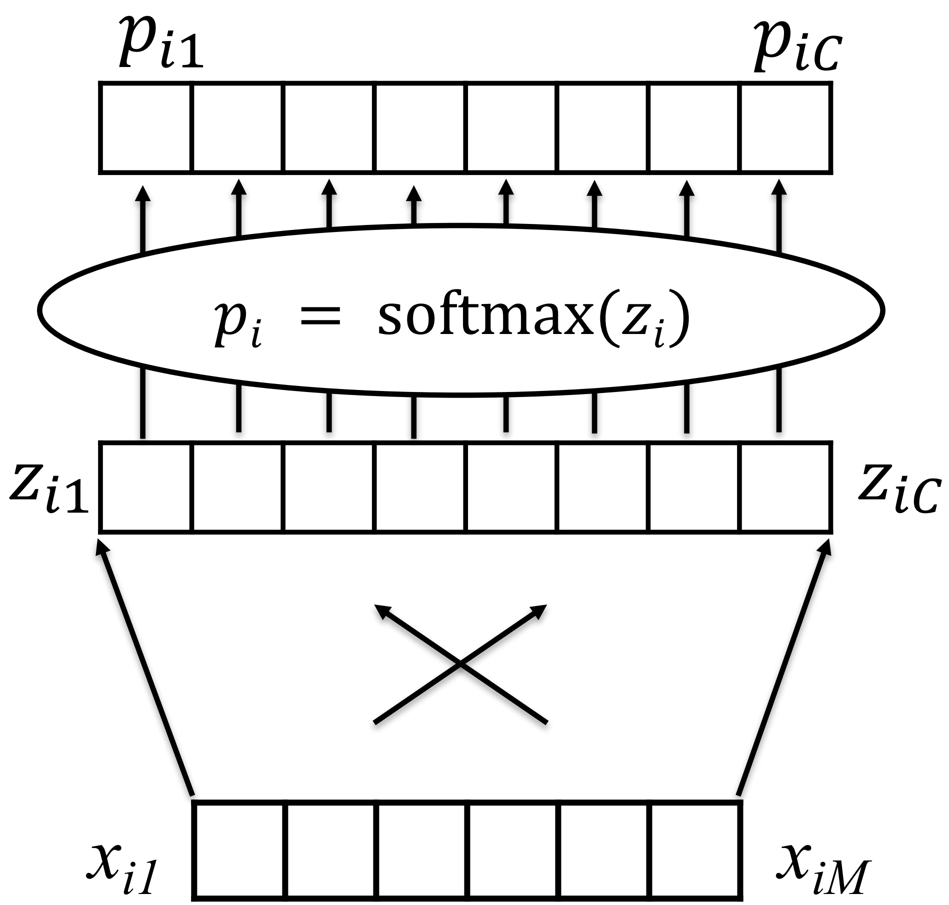
We want: a vector for each word that allows us to predict its context (i.e. what other words are likely/unlikely to be around it)

These vectors are what we want to *learn*

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

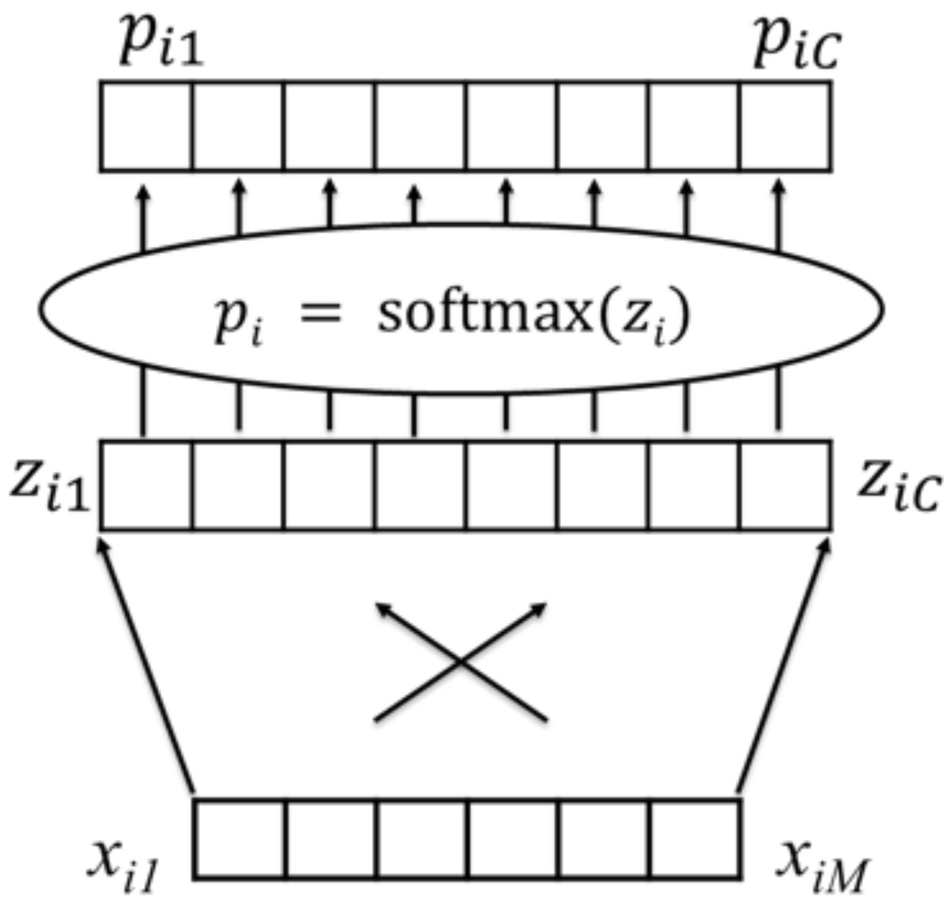


Recall: Multi-Class Logistic Regression

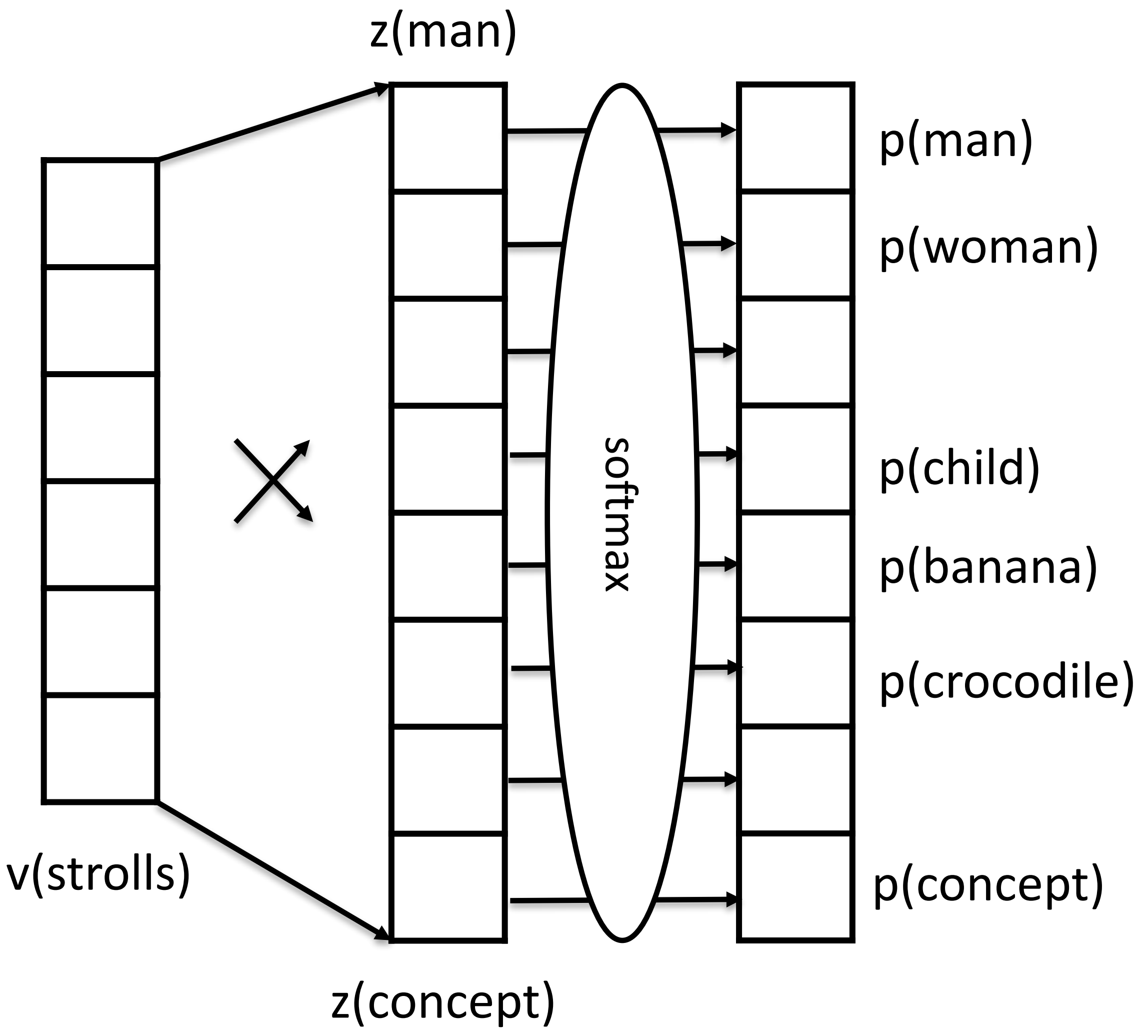


$$p_{ij} = \frac{e^{z_{ij}}}{\sum_{c=1}^{C} e^{z_{ic}}}$$

Recall: Multi-Class Logistic Regression



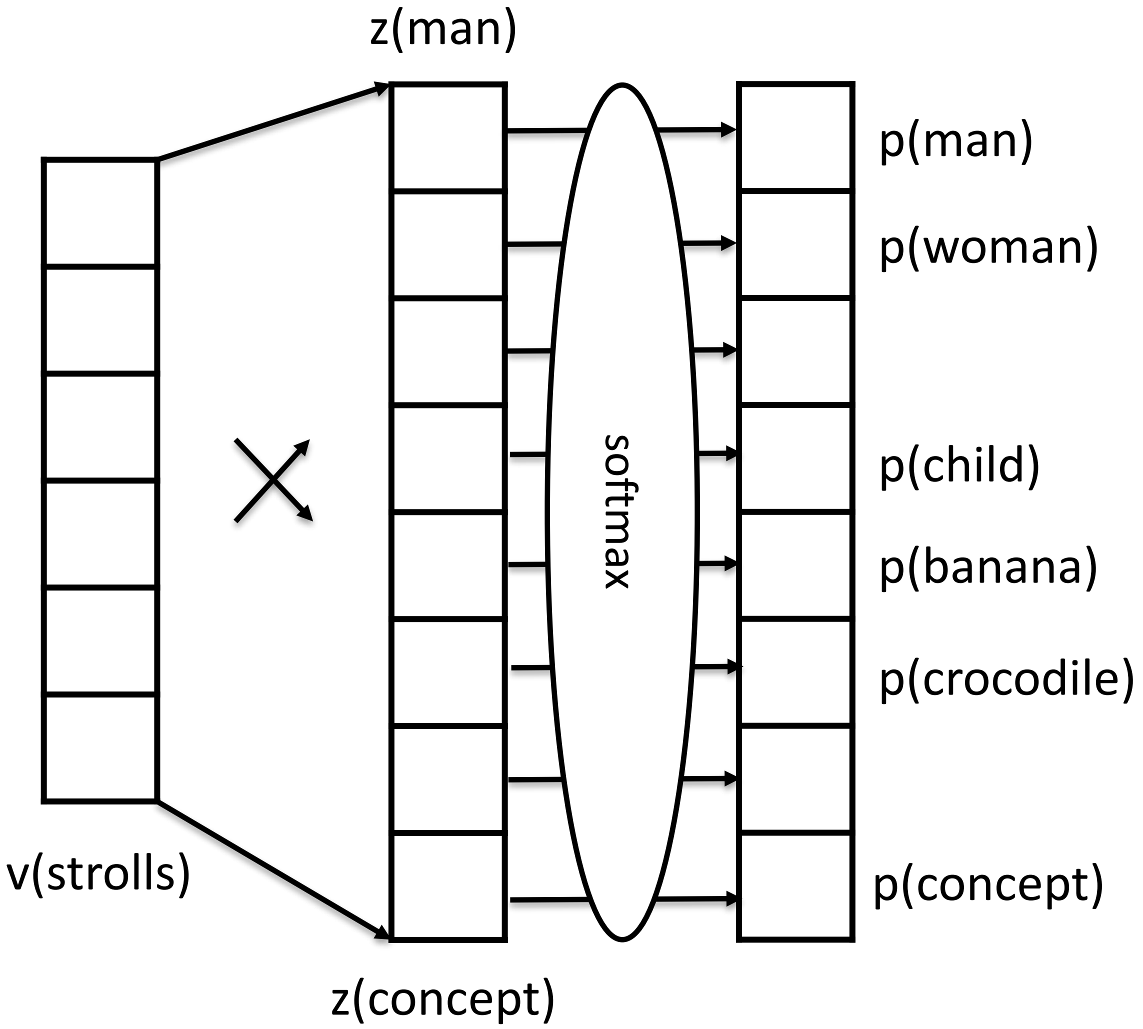
Recall: Multi-Class Logistic Regression



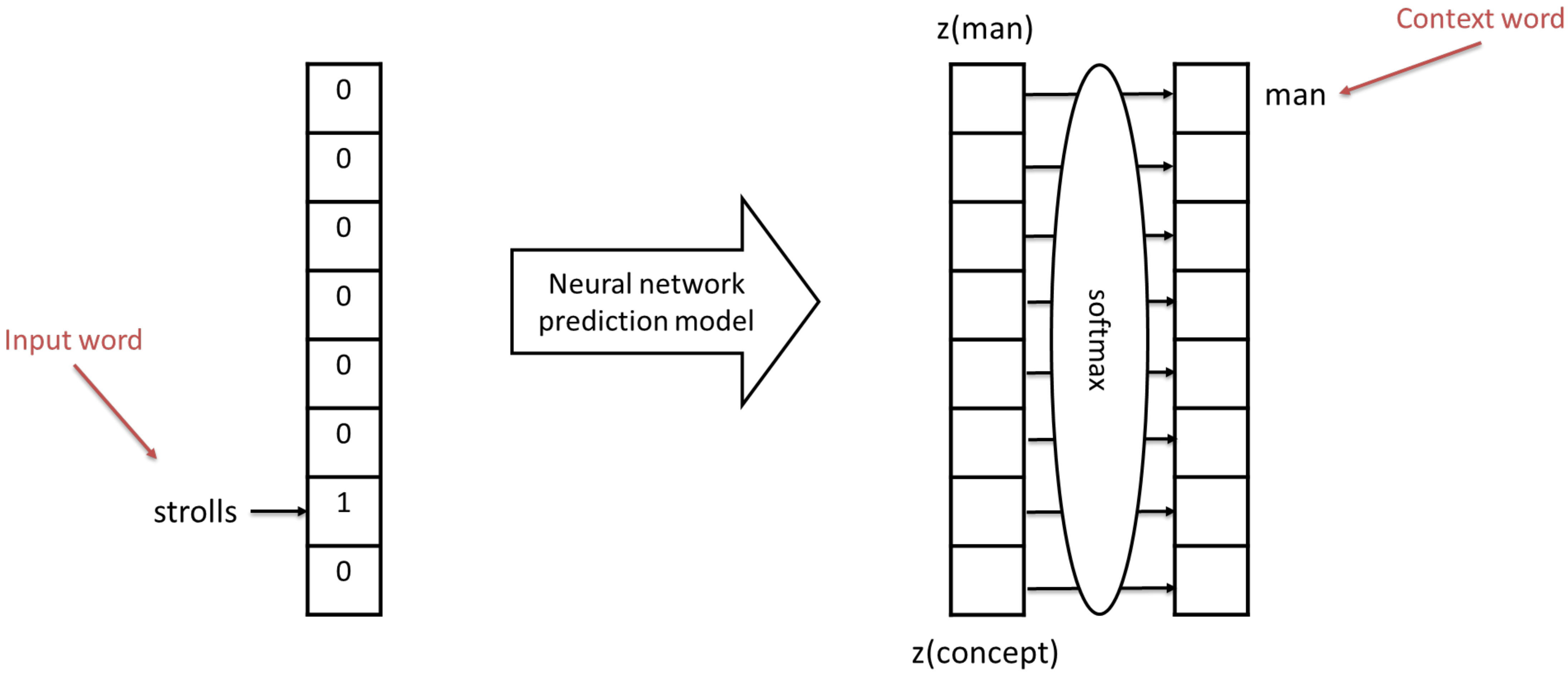
We want: word vectors that allow us to predict their likely context

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:

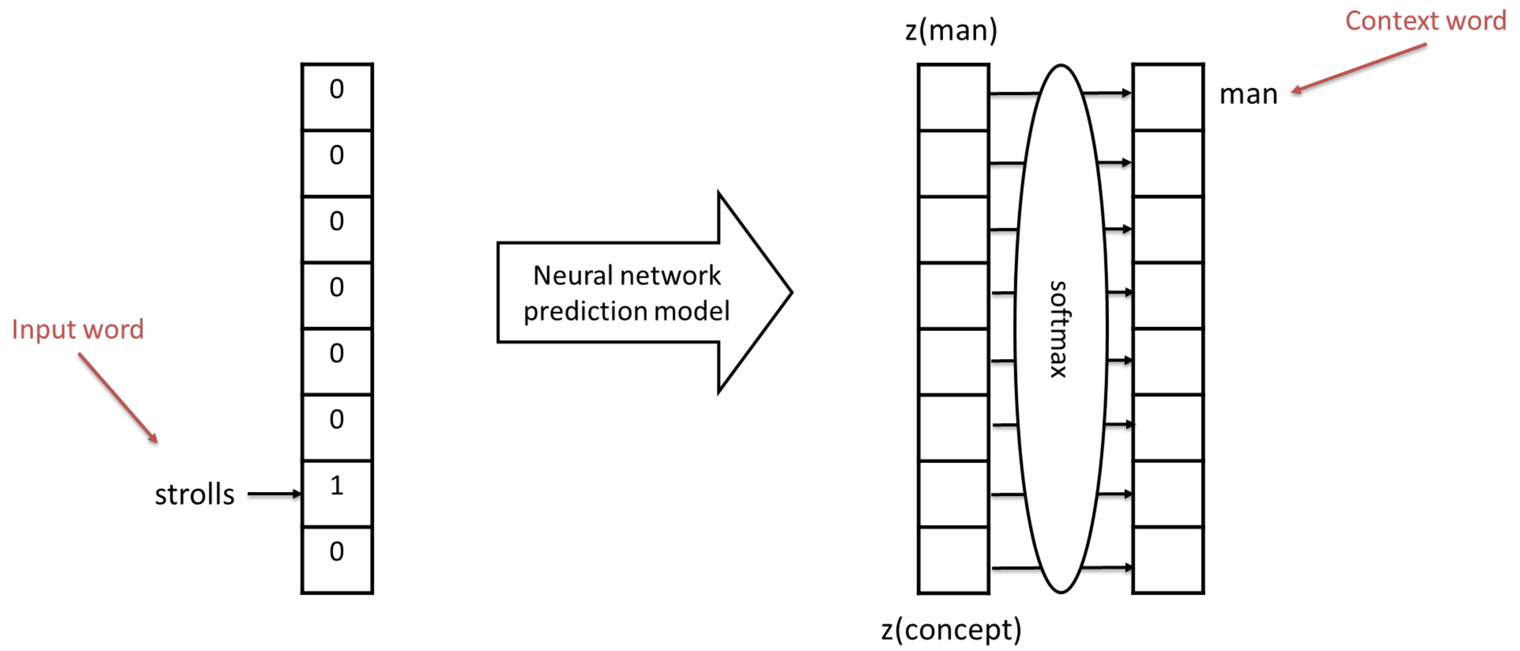
HUGE number of pairs of the following form:

{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:

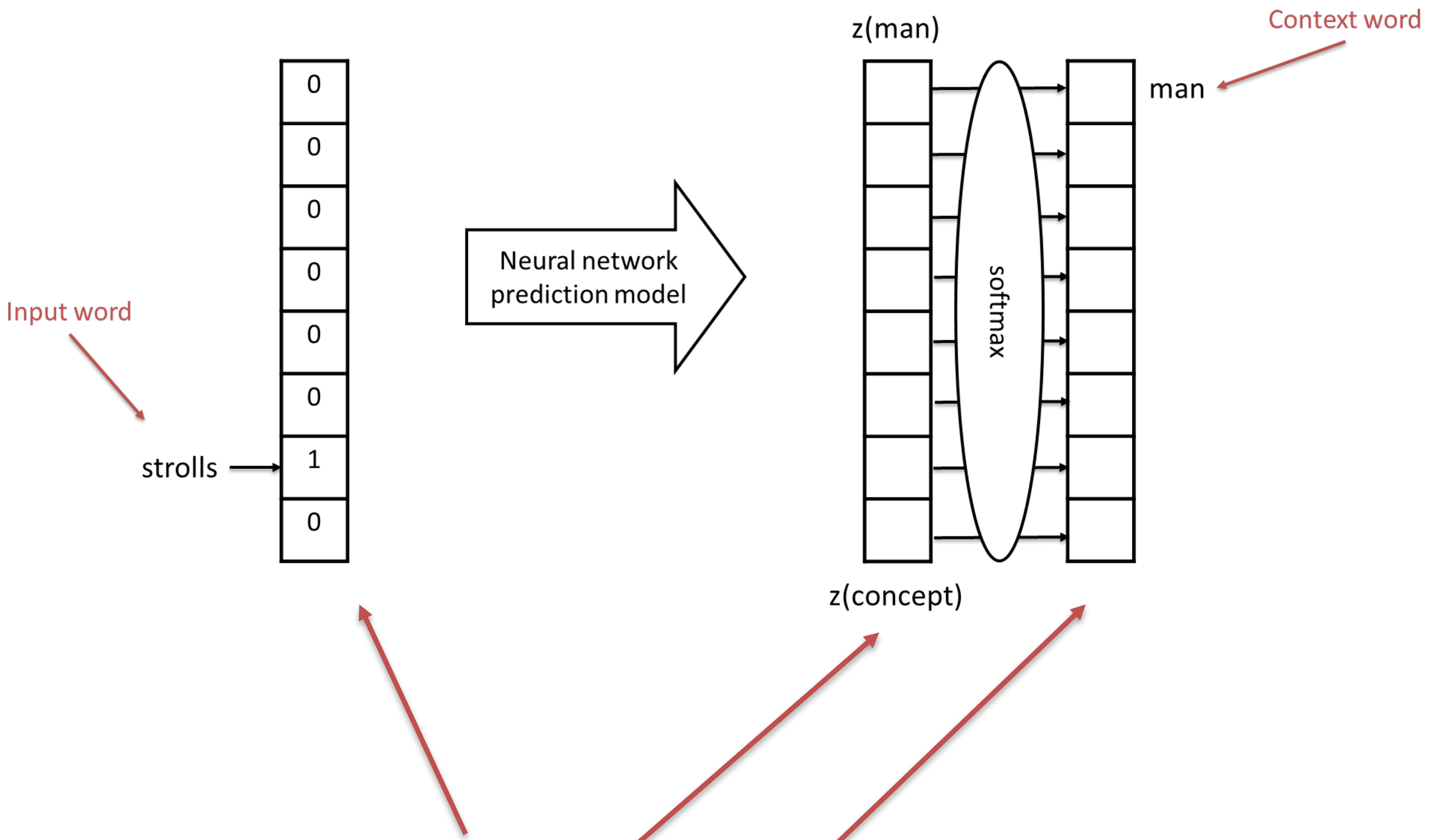
HUGE number of pairs of the following form:

{input word, context word}

e.g. from Wikipedia

Examples:

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

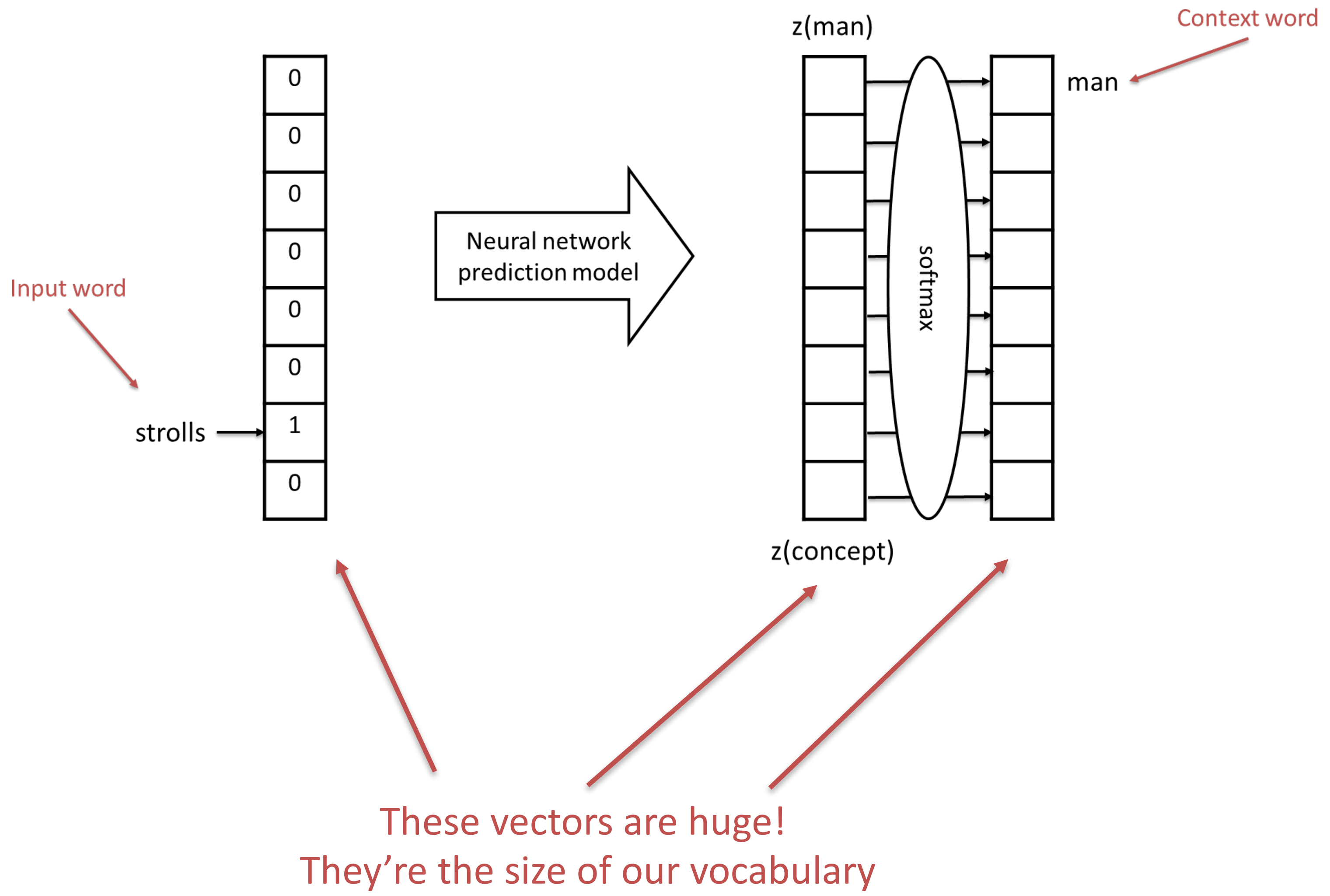


These vectors are huge!
They're the size of our vocabulary

What's the simplest model we can possibly use?

First idea:

Directly connect our input to the log-odds layer



What's the simplest model we can possibly use?

~~First idea:~~

~~Directly connect our input to the log-odds layer~~

~~How many connections?~~

~~V x V~~

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

First idea:

Directly connect our input to the log-odds layer

How many connections?

V x V

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

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

What's the next simplest?

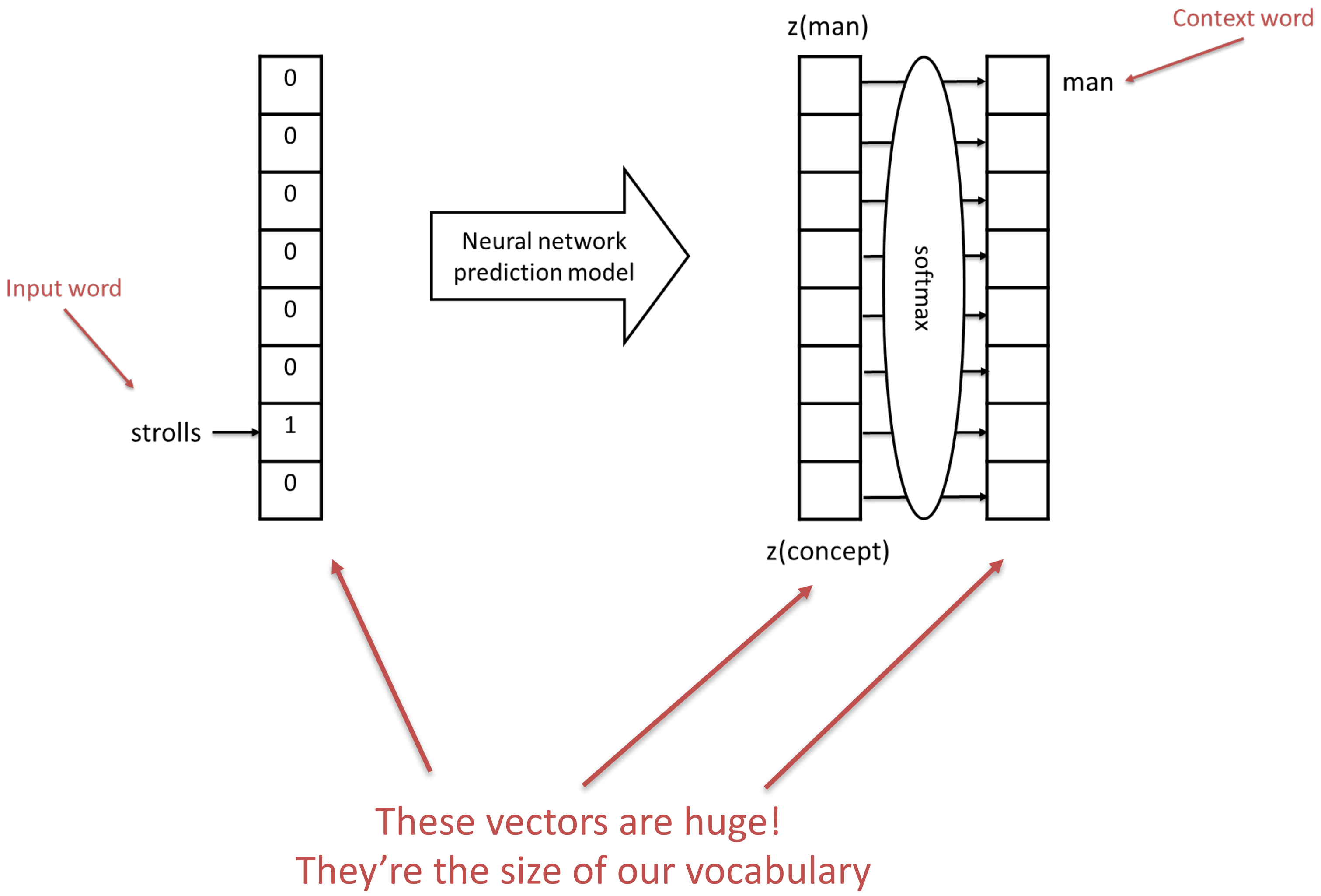
How about a single hidden layer?

How many connections?

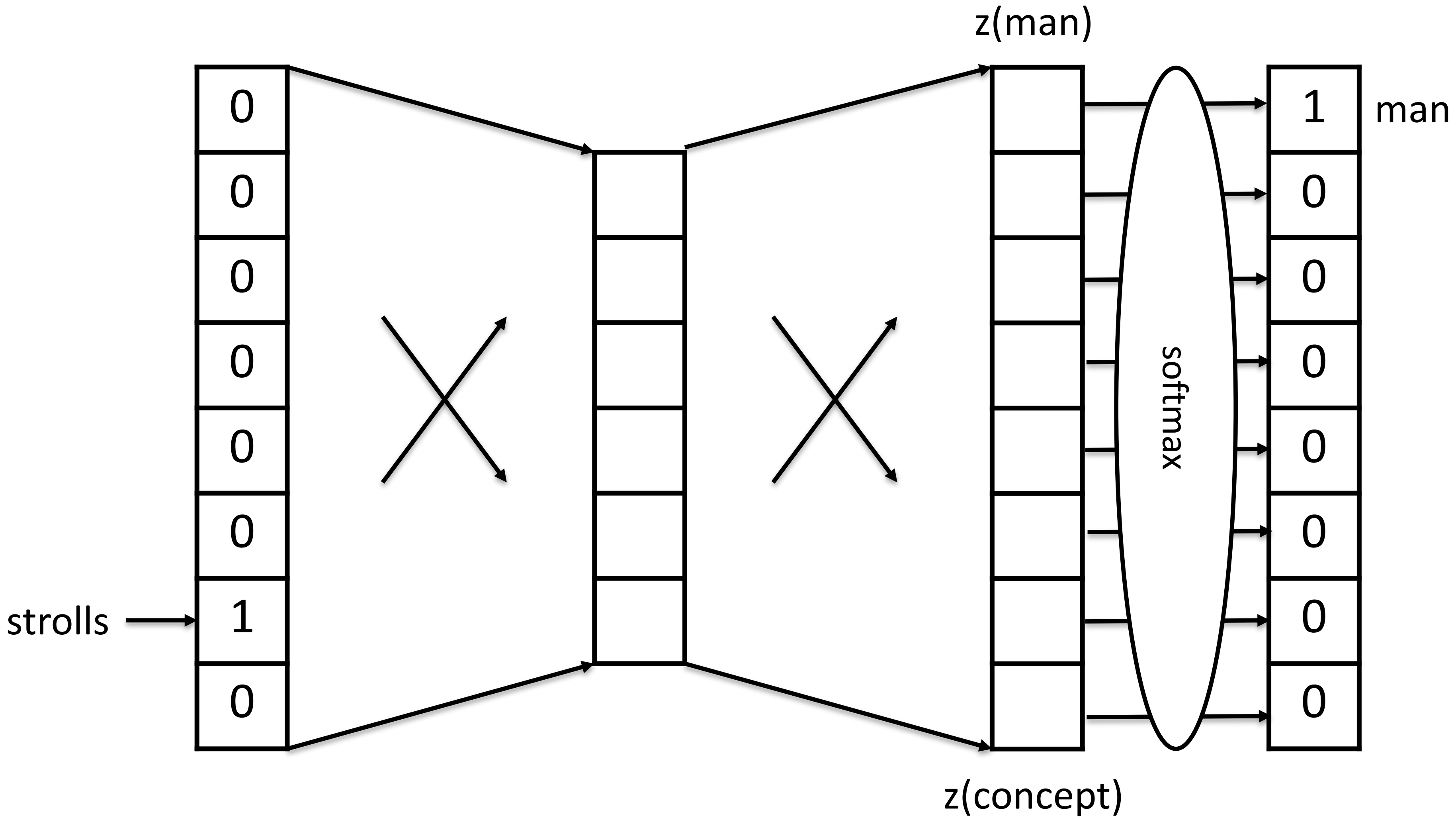
V x H x 2

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

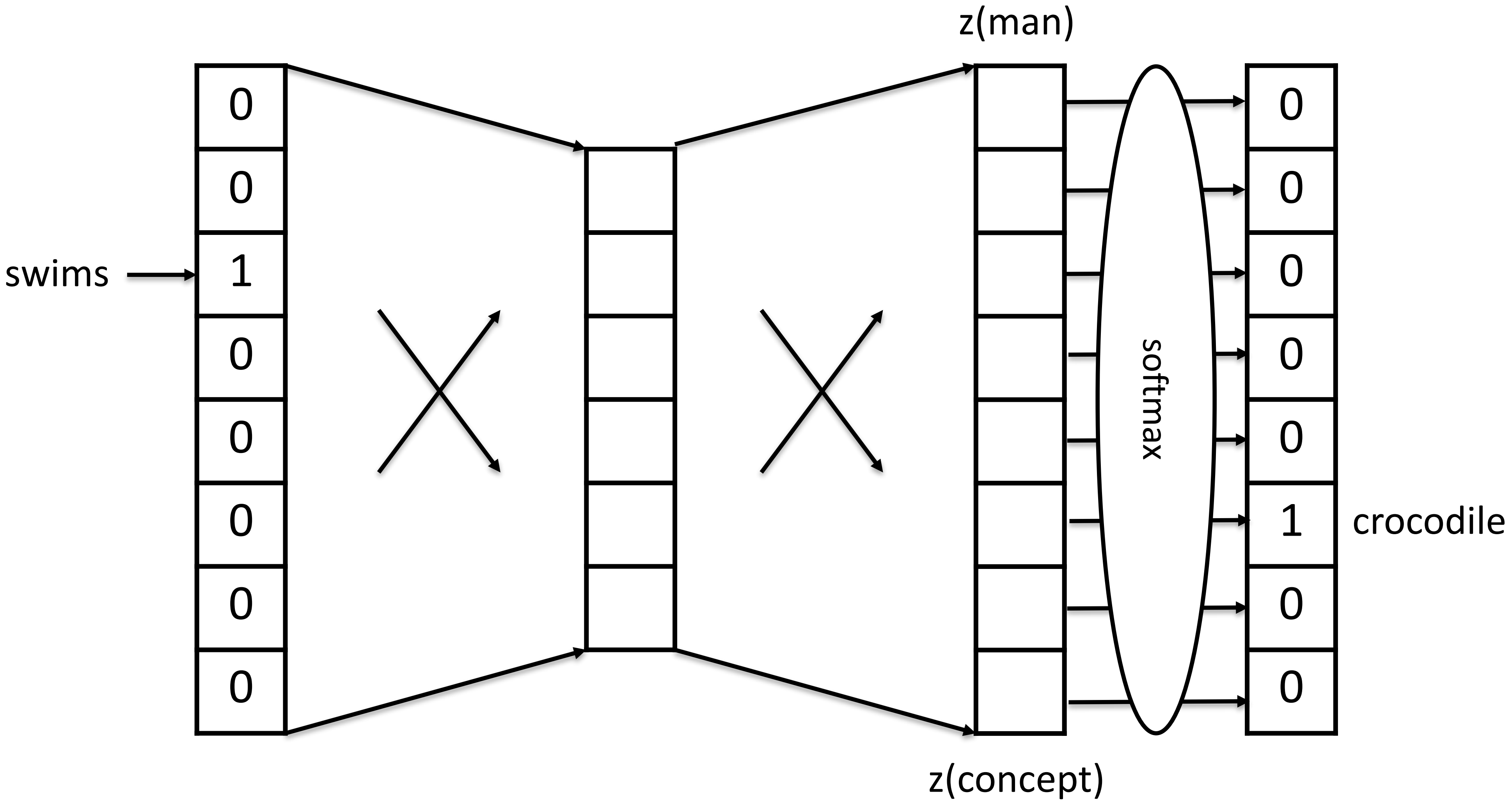
And H is our hidden layer size ($\ll V$)



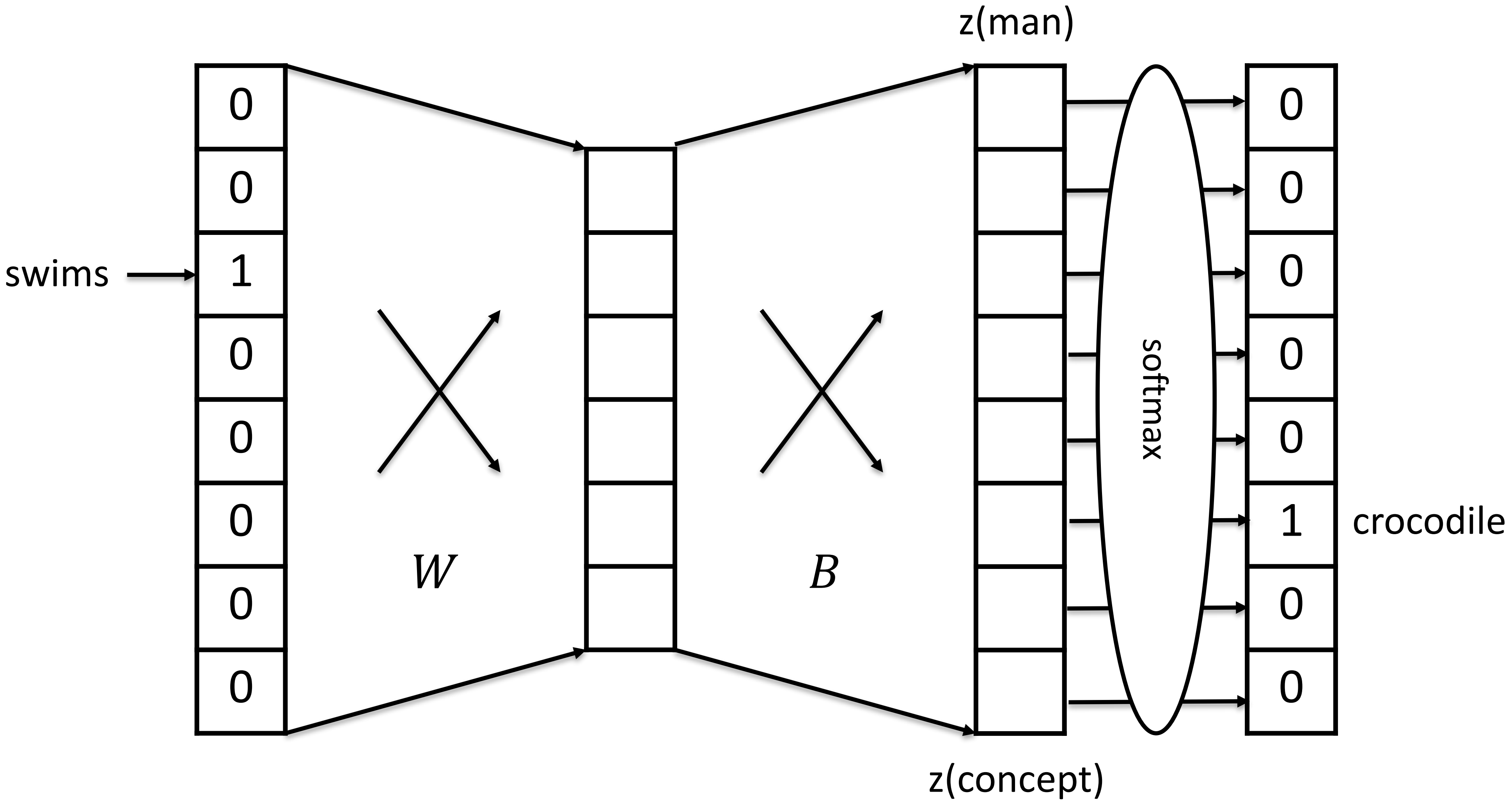
OK, let's try it: use a single hidden layer



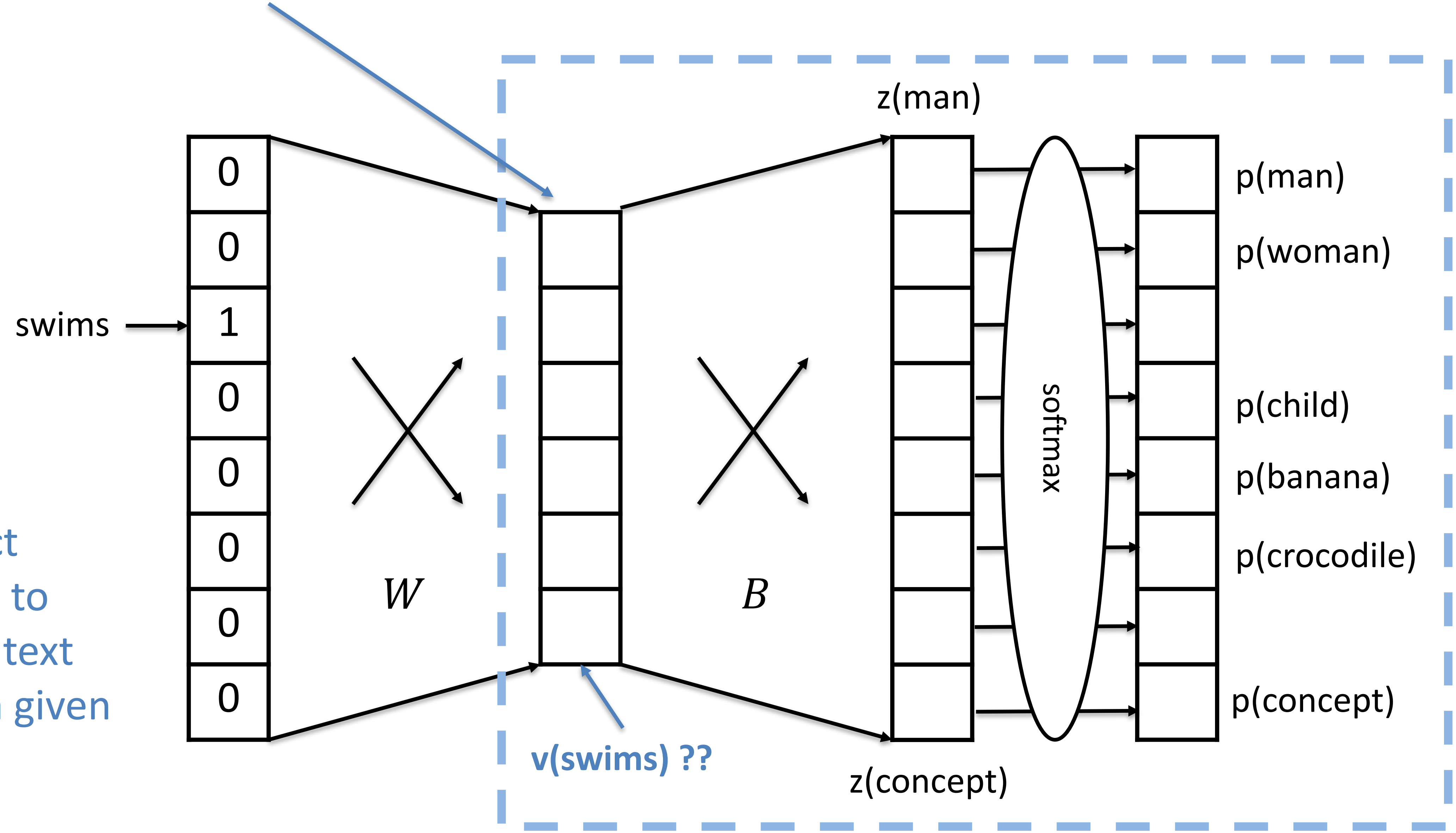
Use mini-batches of training examples; minimize cross-entropy loss



Learn our parameters: Weight Matrices W and B

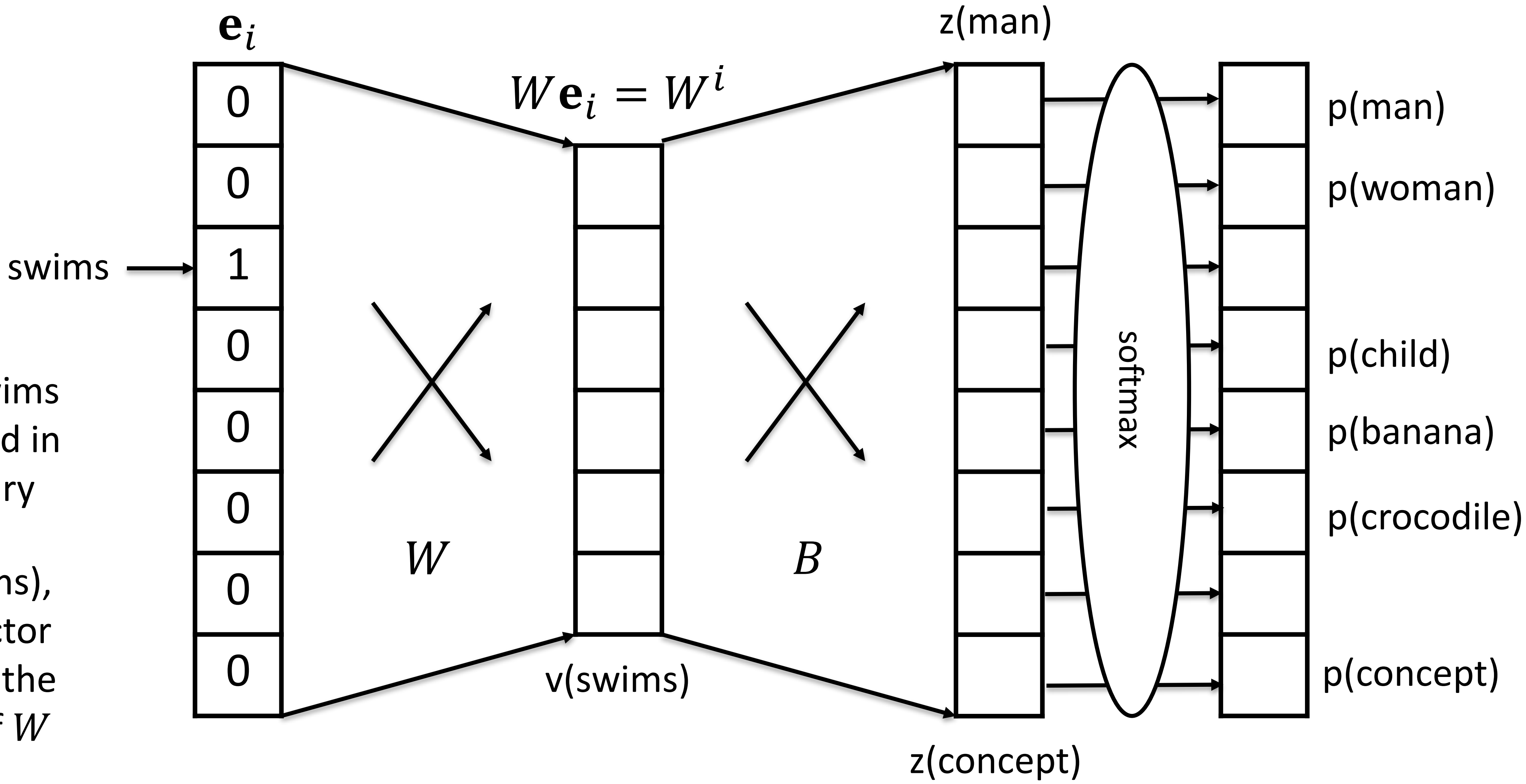


Isn't **this** the vector we were looking for?



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

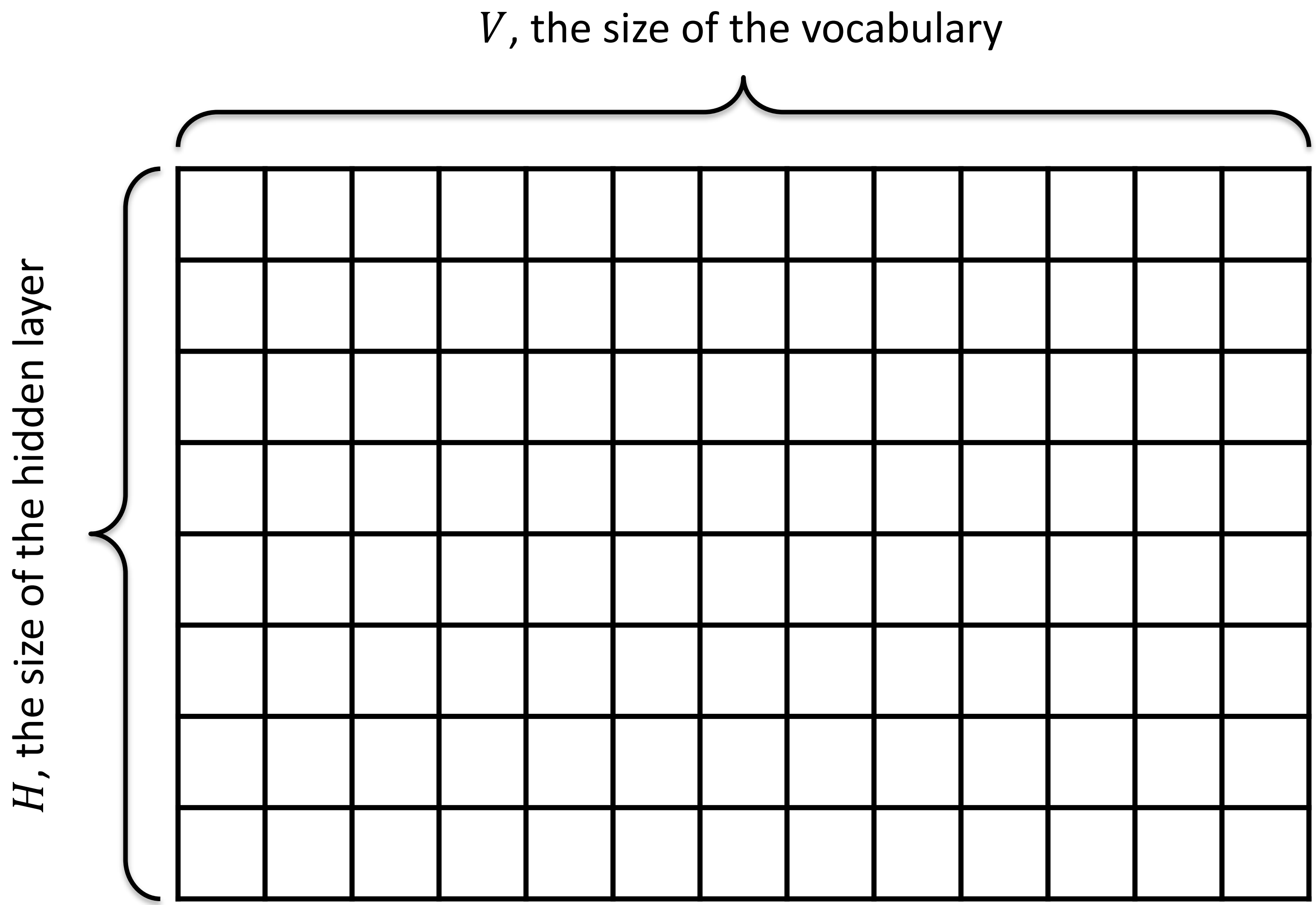
Let's take a closer look at W



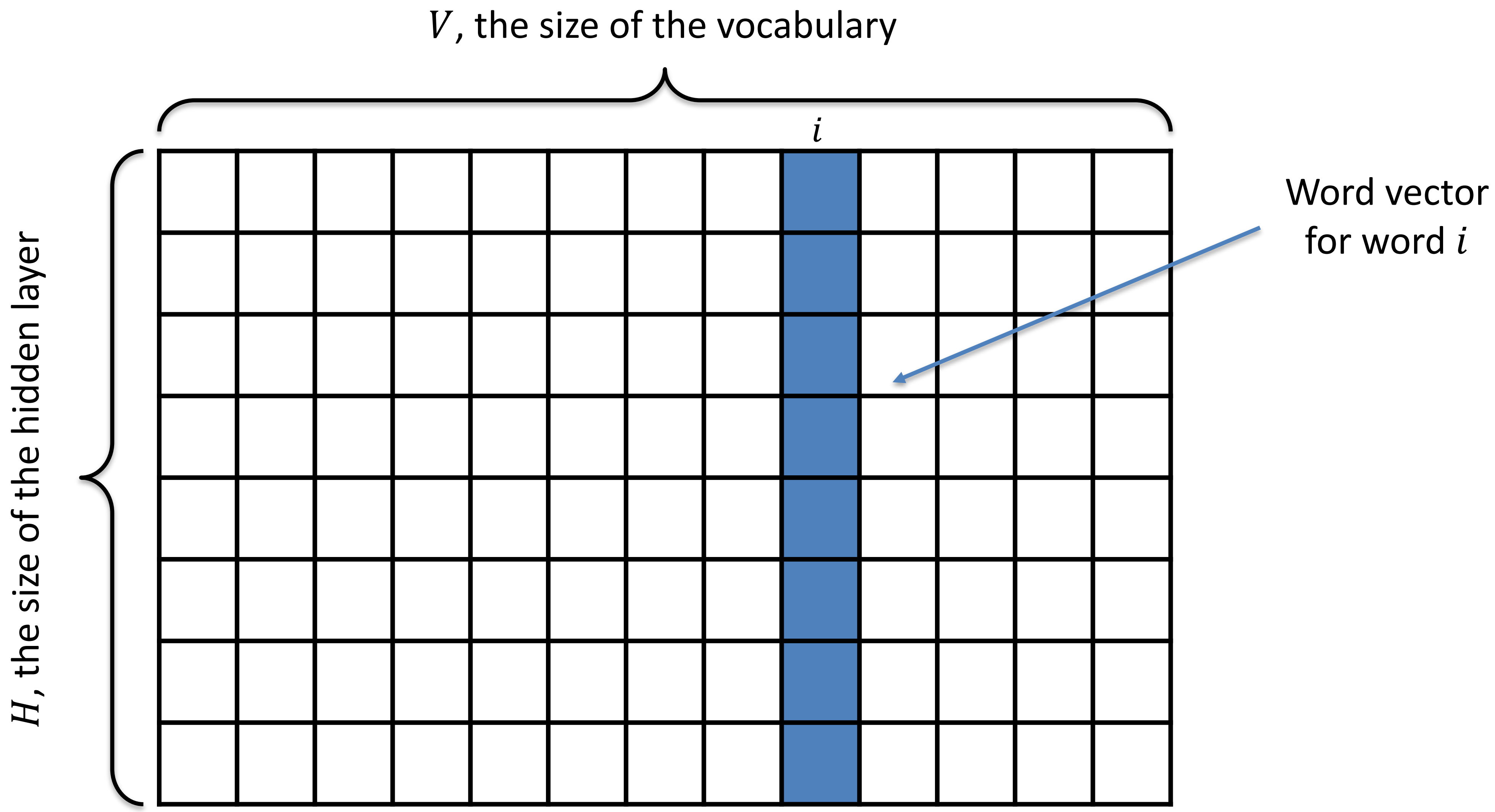
- suppose swims is the i^{th} word in our vocabulary

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

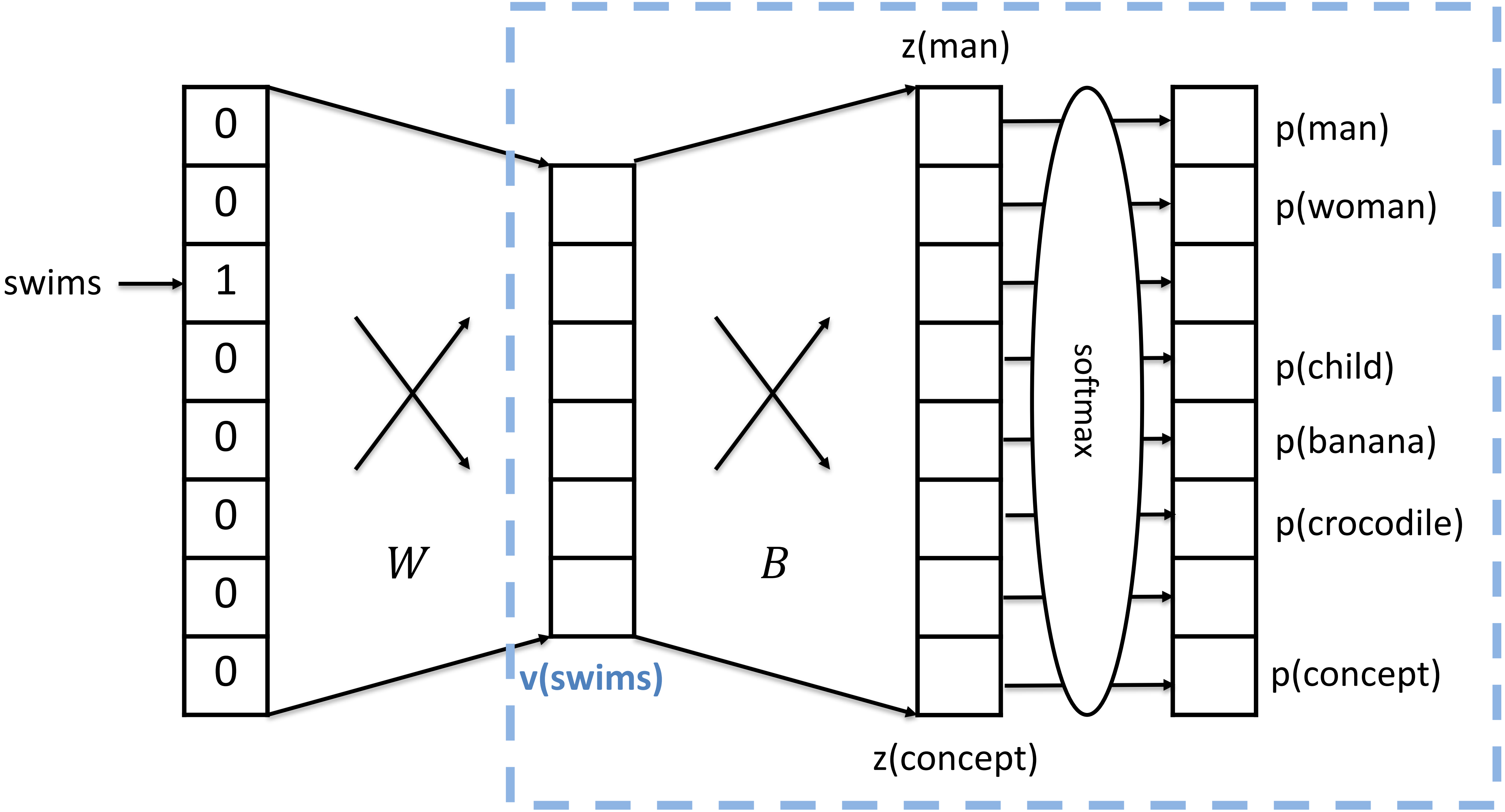
Let’s take a closer look at W



Let's take a closer look at W



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



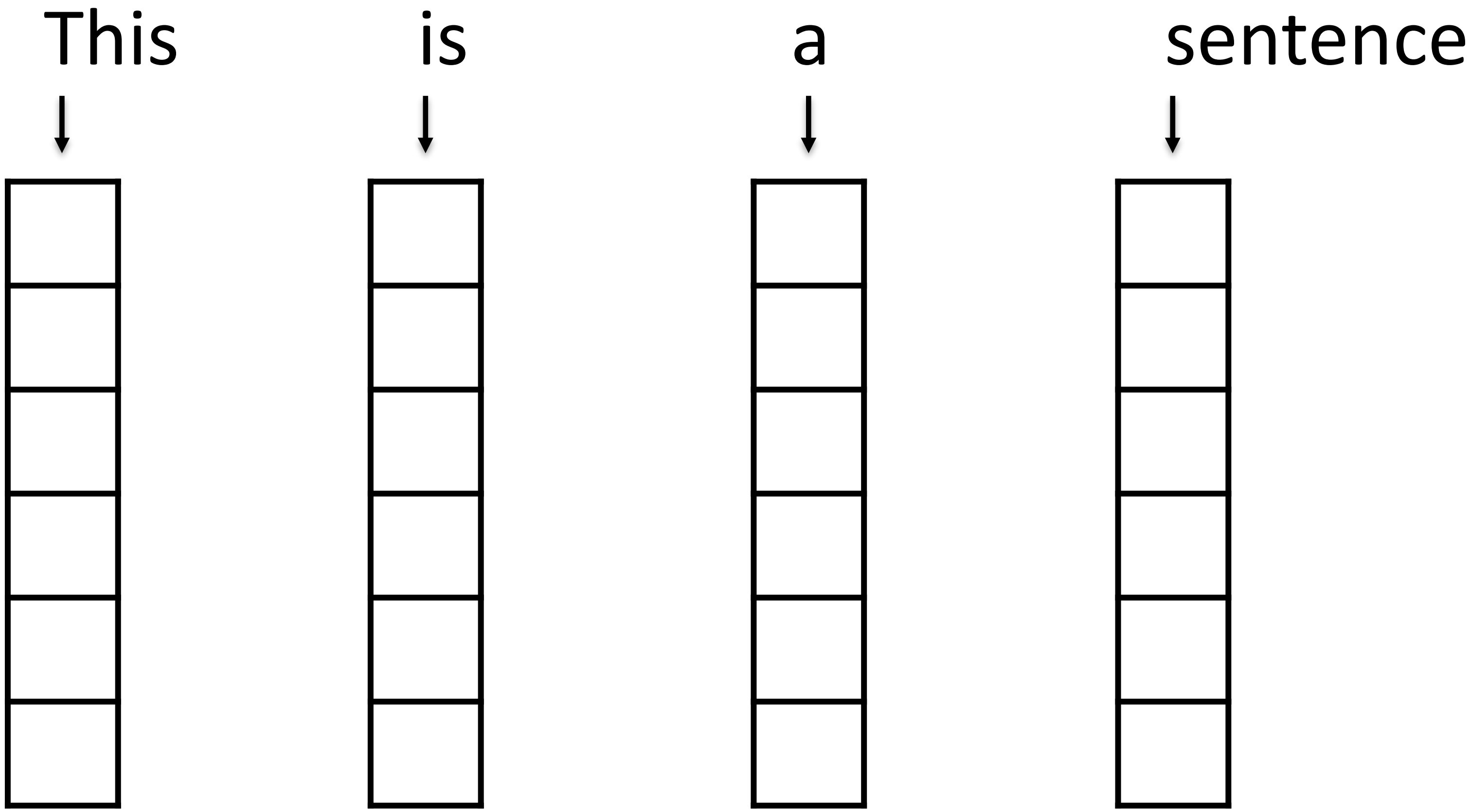
Important Takeaways:

- 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

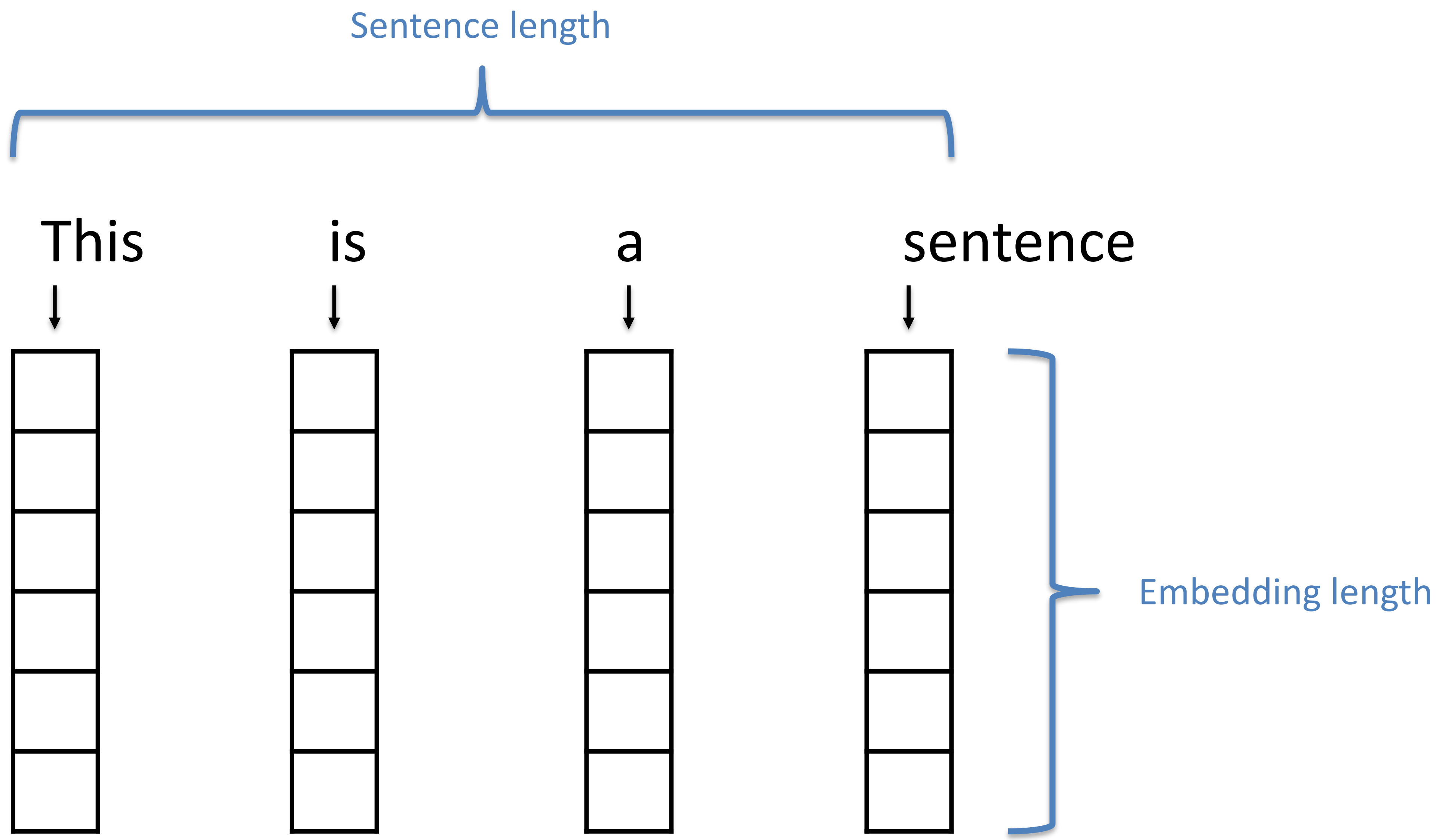
A VERY SIMPLE WORD EMBEDDING-BASED MODEL

VSWEM Step 1: Convert sentence to vectors

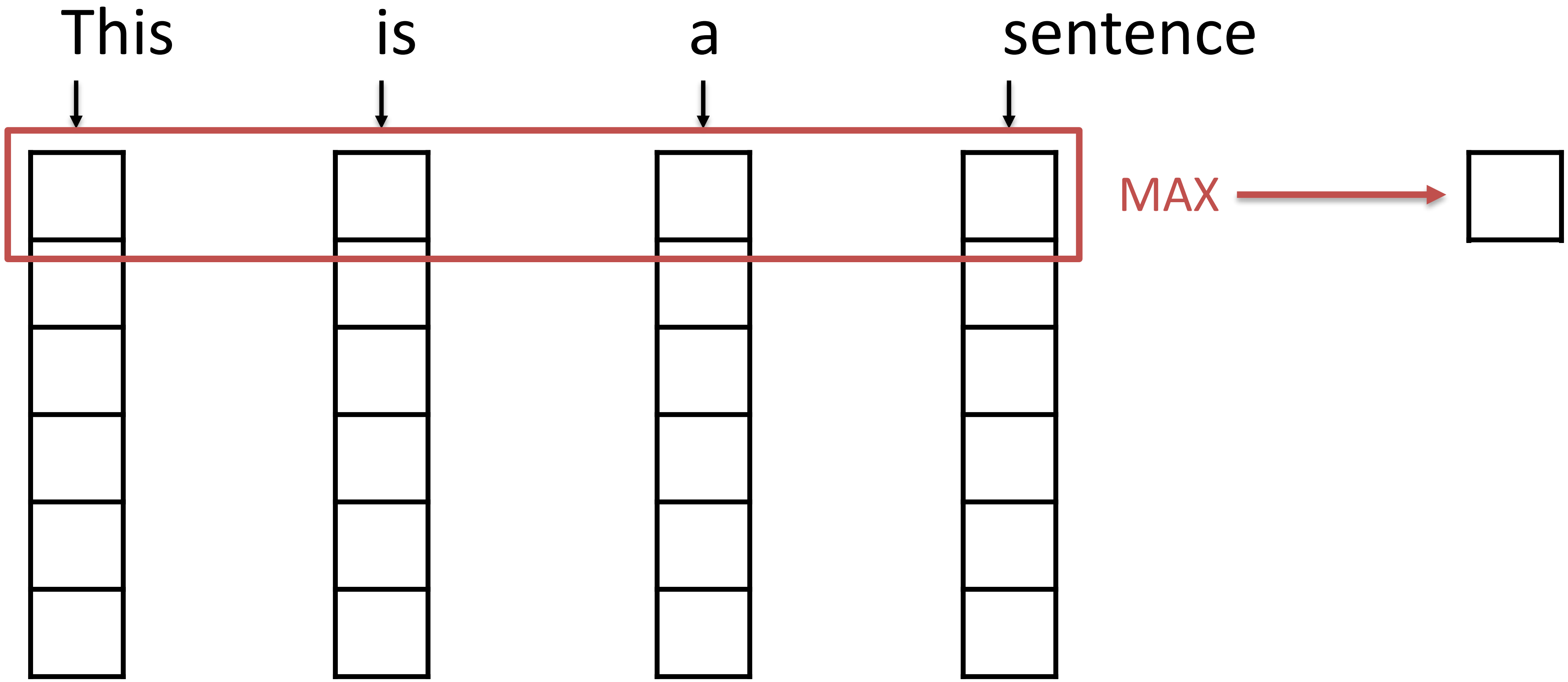
- Look up words individually to obtain their vectors
- Construct a sequence of vectors



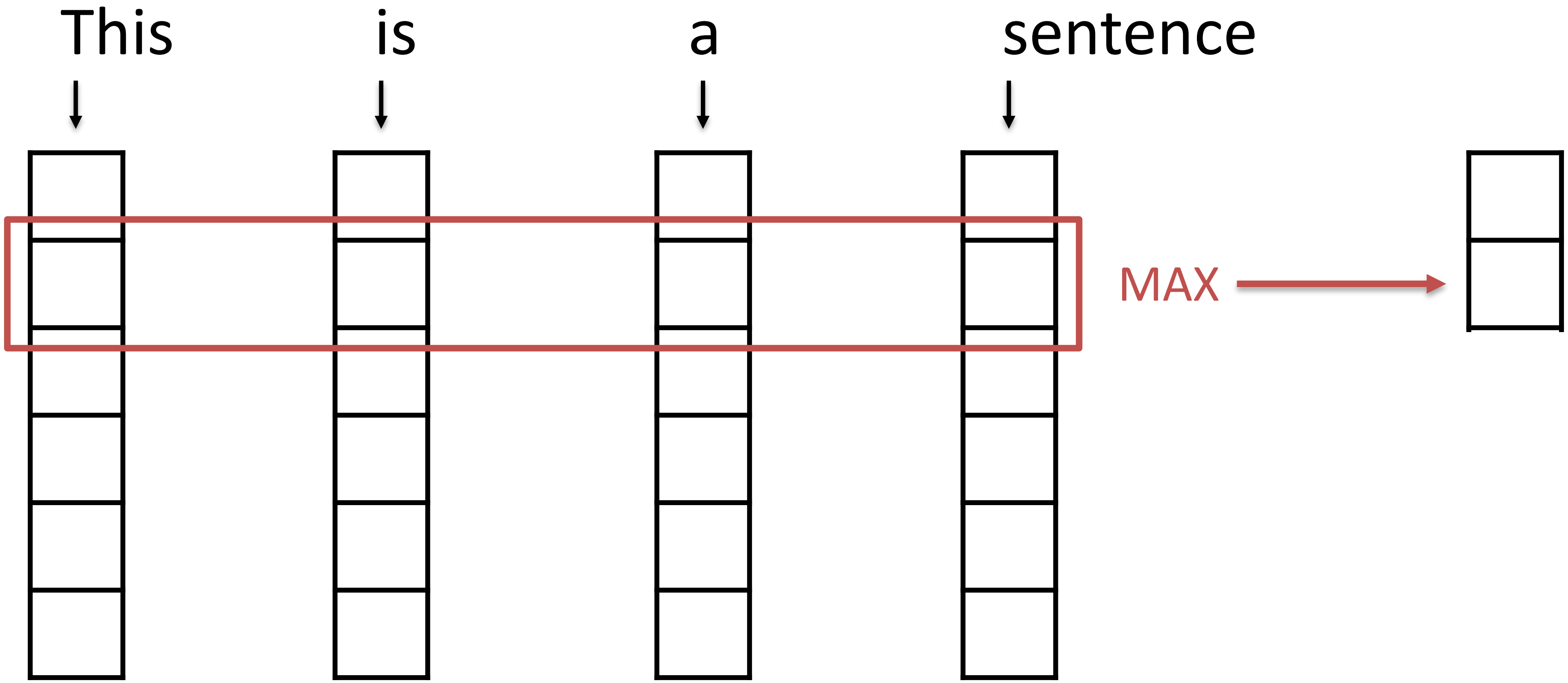
VSWEM Step 1: Convert sentence to vectors



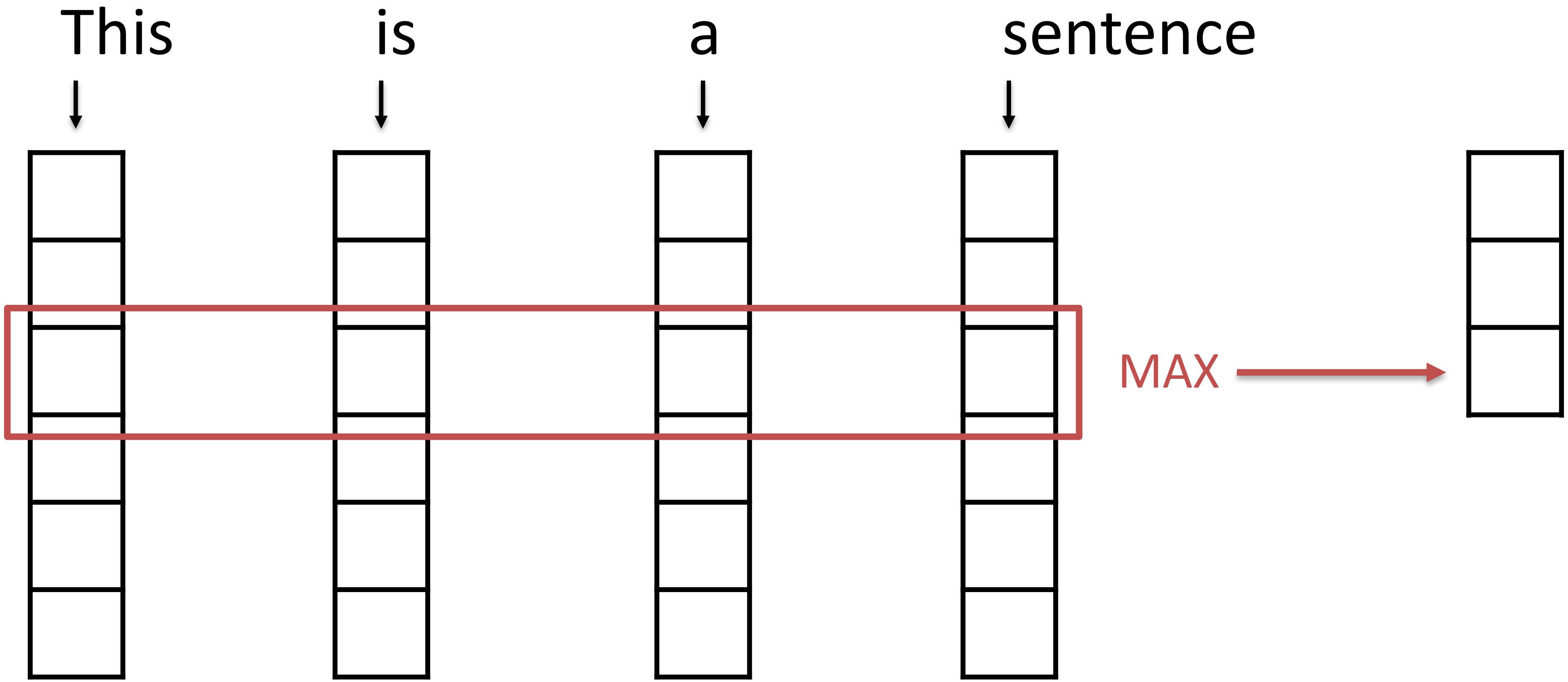
VSWEM Step 2: Take the MAX over the sentence for each embedding dimension



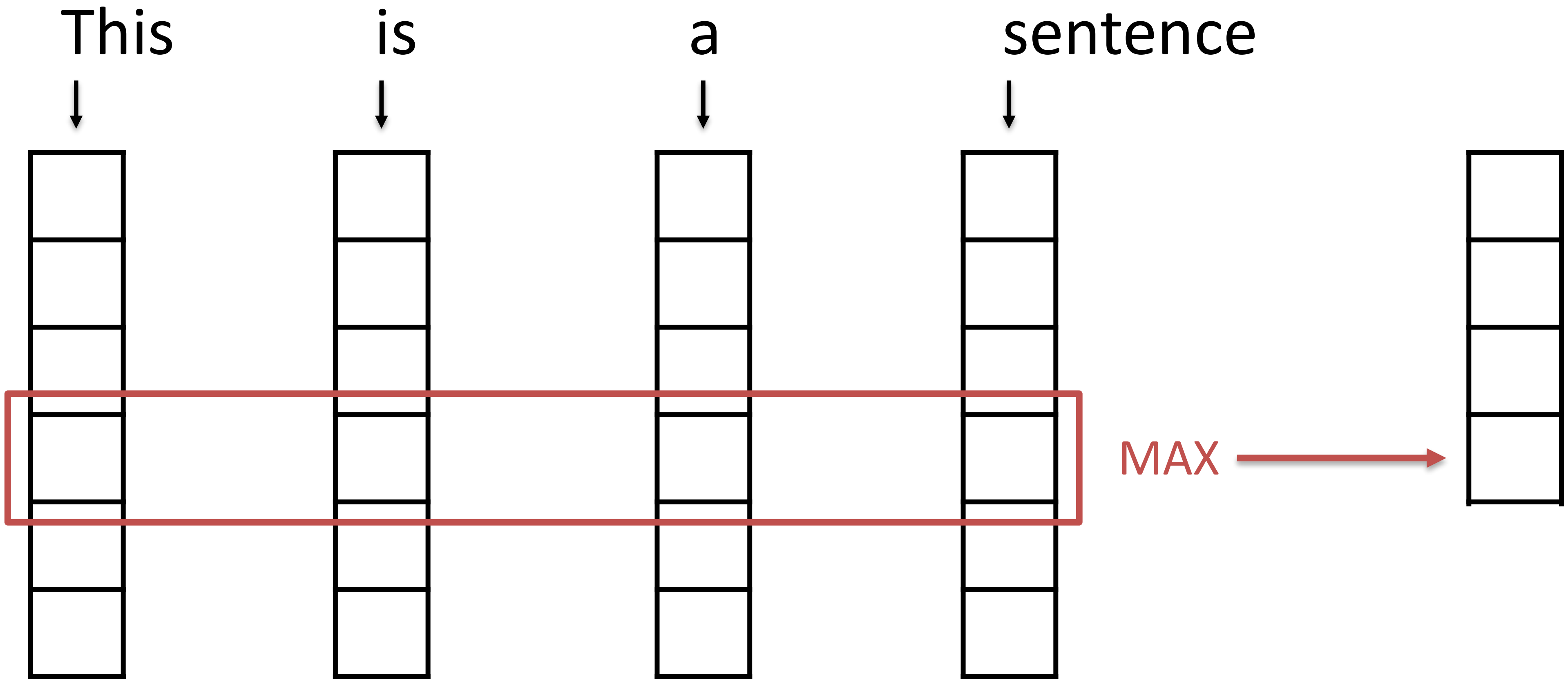
VSWEM Step 2: Take the MAX over the sentence for each embedding dimension



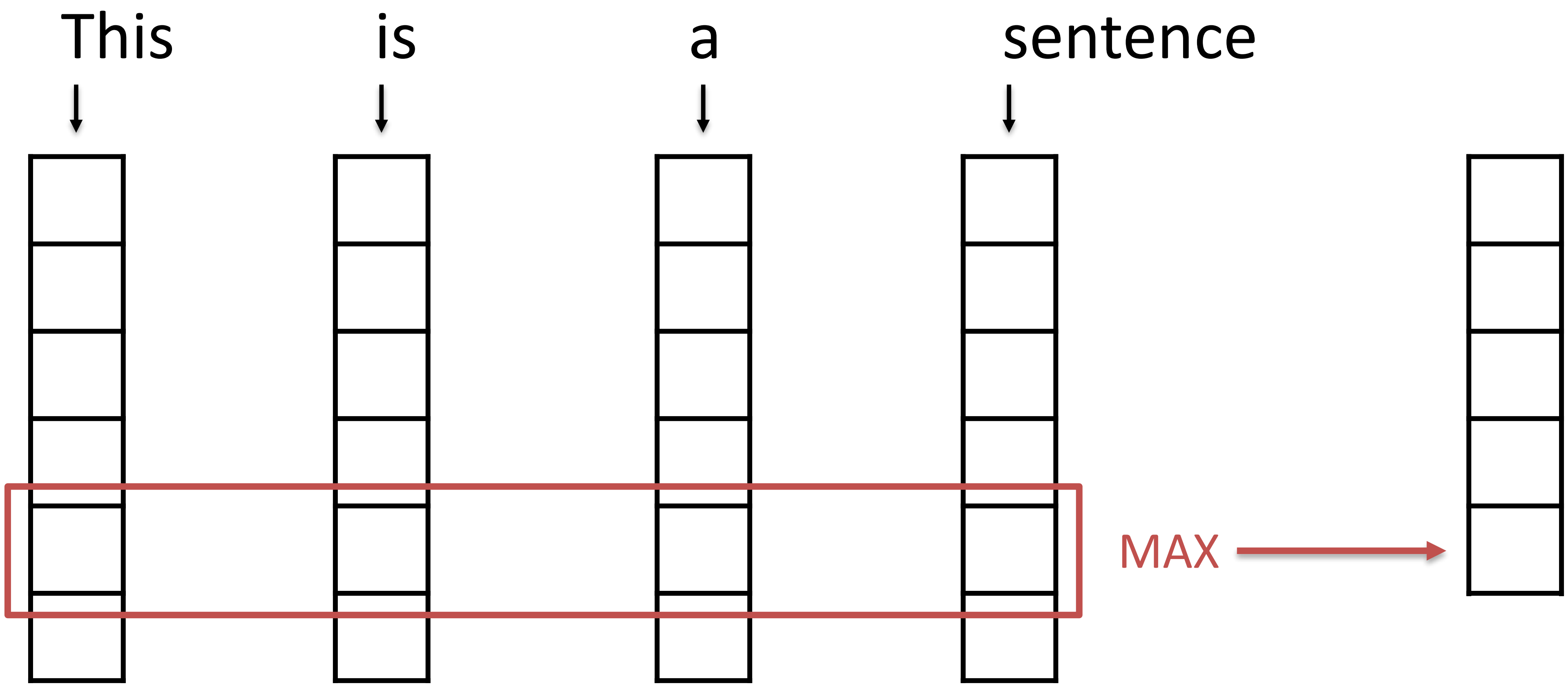
VSWEM Step 2: Take the MAX over the sentence for each embedding dimension



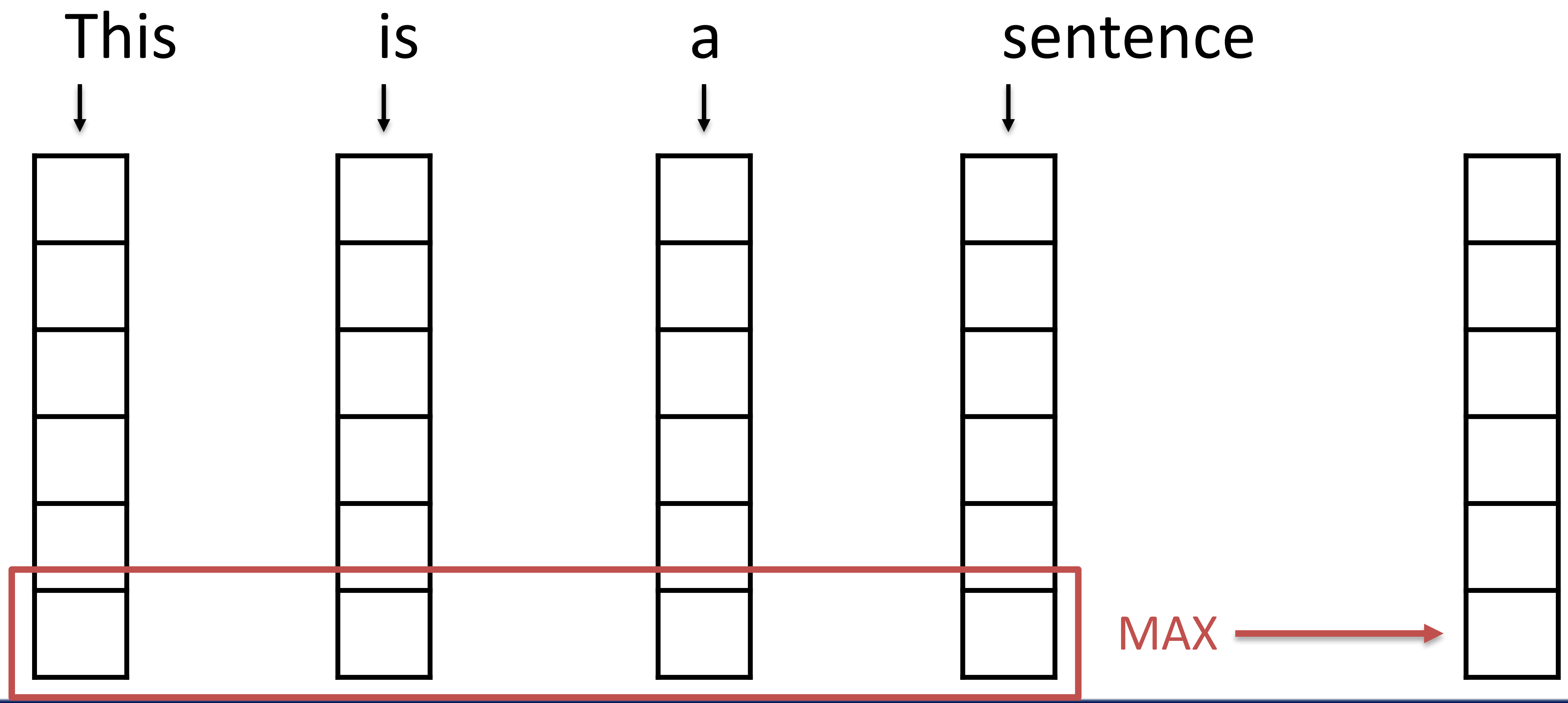
VSWEM Step 2: Take the MAX over the sentence for each embedding dimension



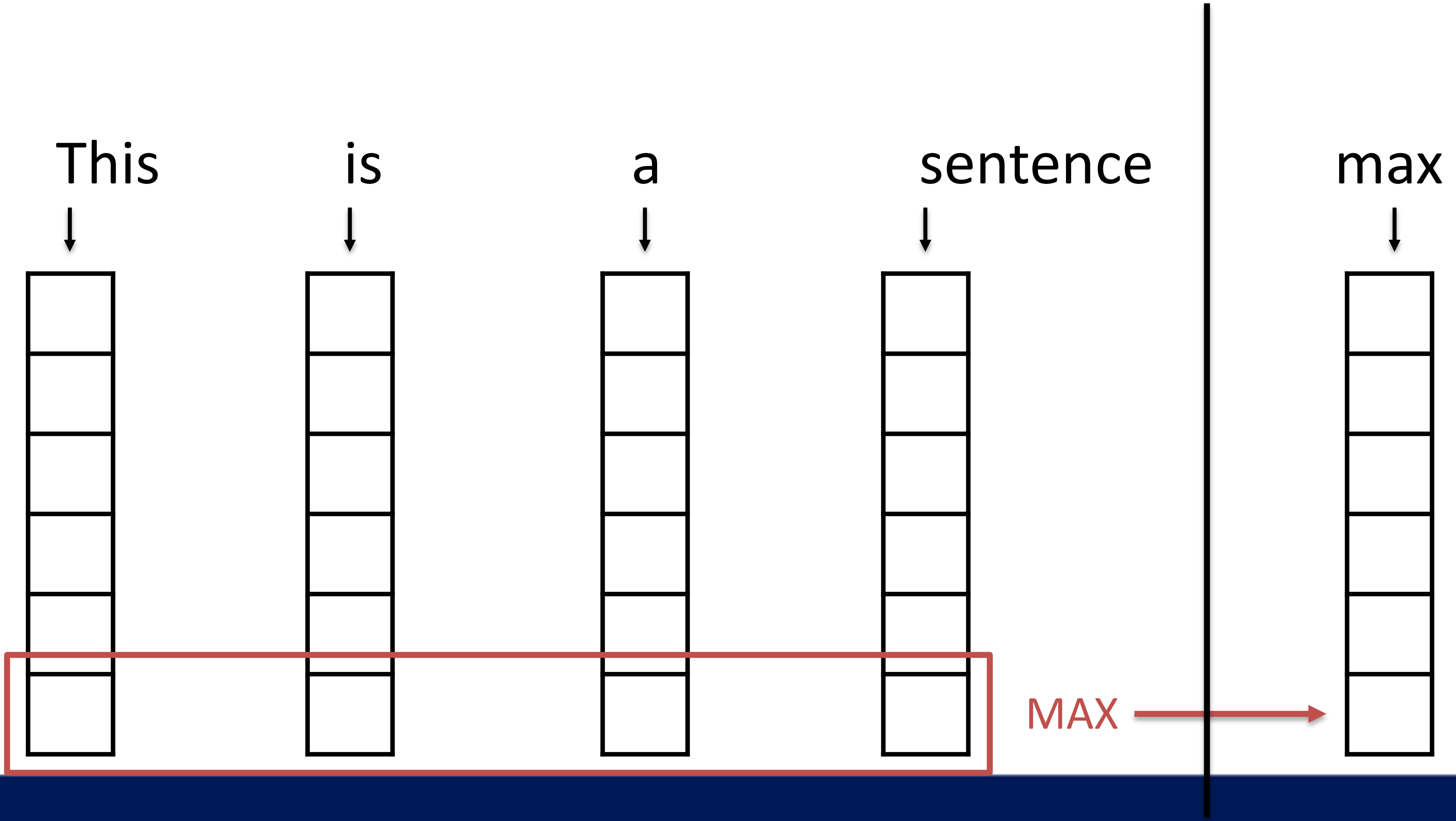
VSWEM Step 2: Take the MAX over the sentence for each embedding dimension



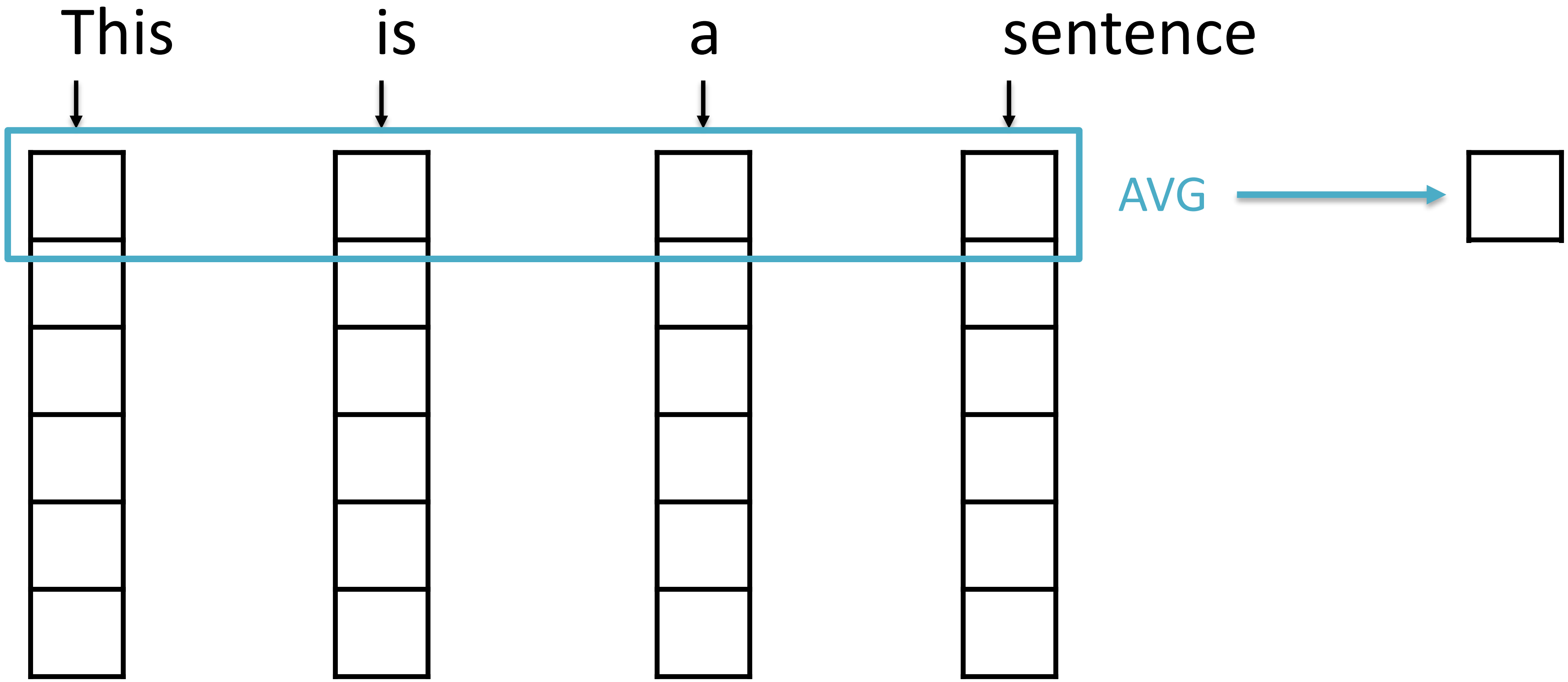
VSWEM Step 2: Take the MAX over the sentence for each embedding dimension



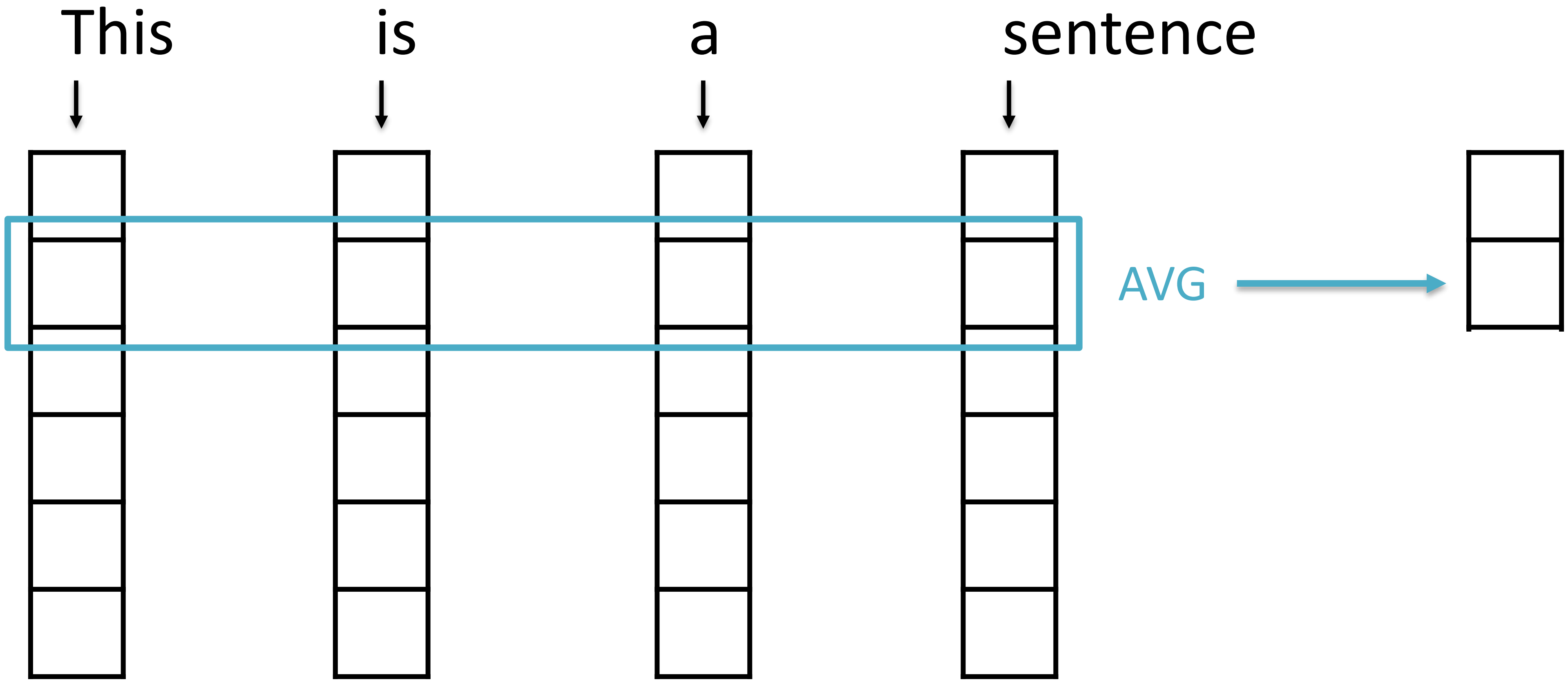
VSWEM Step 2: Take the MAX over the sentence for each embedding dimension



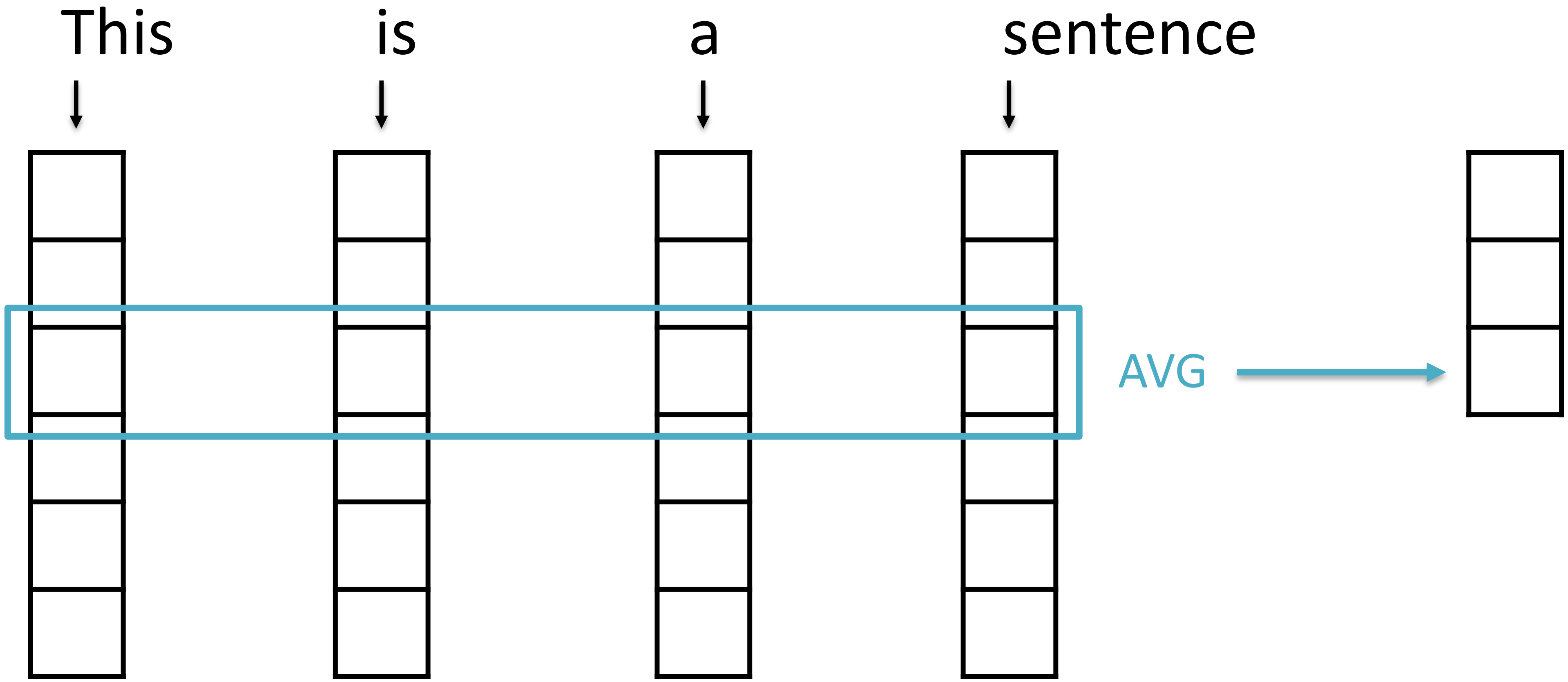
VSWEM Step 3: Take the AVERAGE over the sentence for each embedding dimension



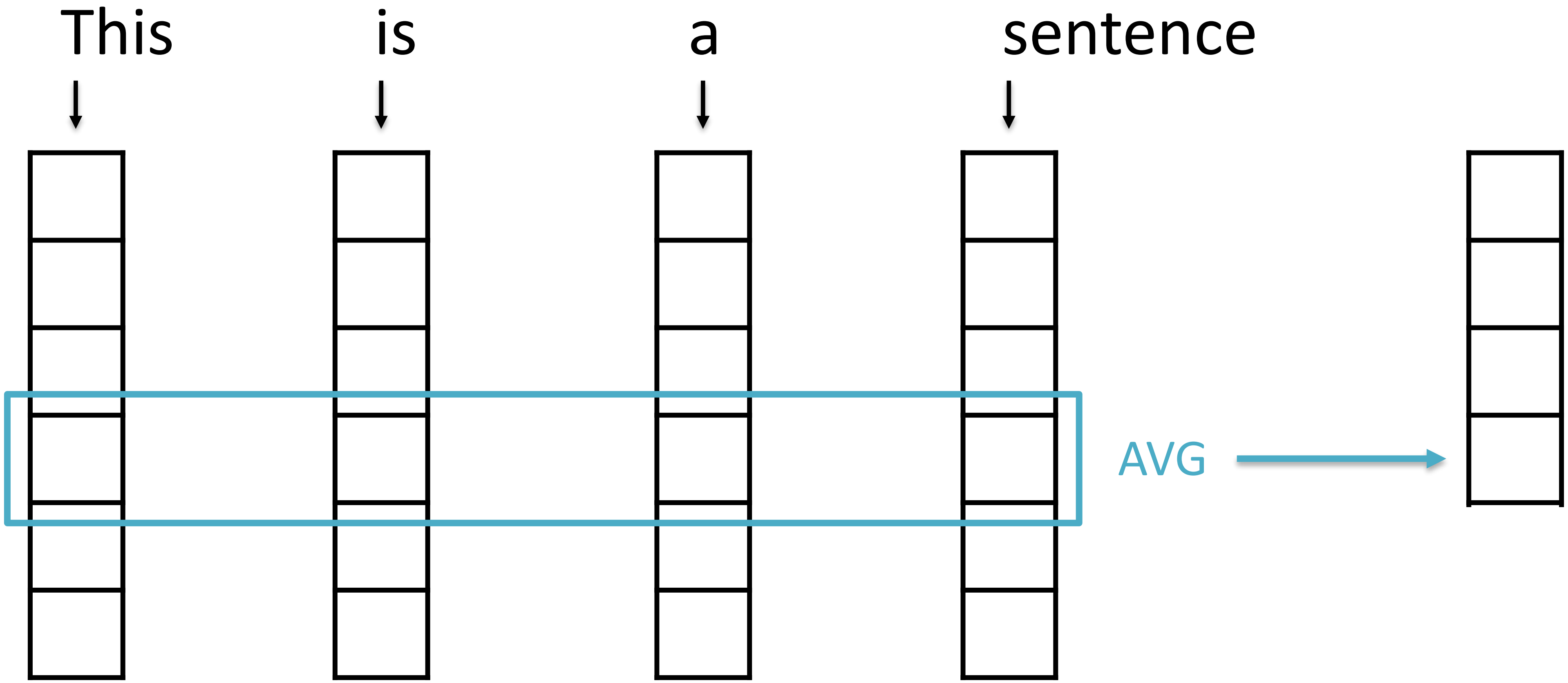
VSWEM Step 3: Take the AVERAGE over the sentence for each embedding dimension



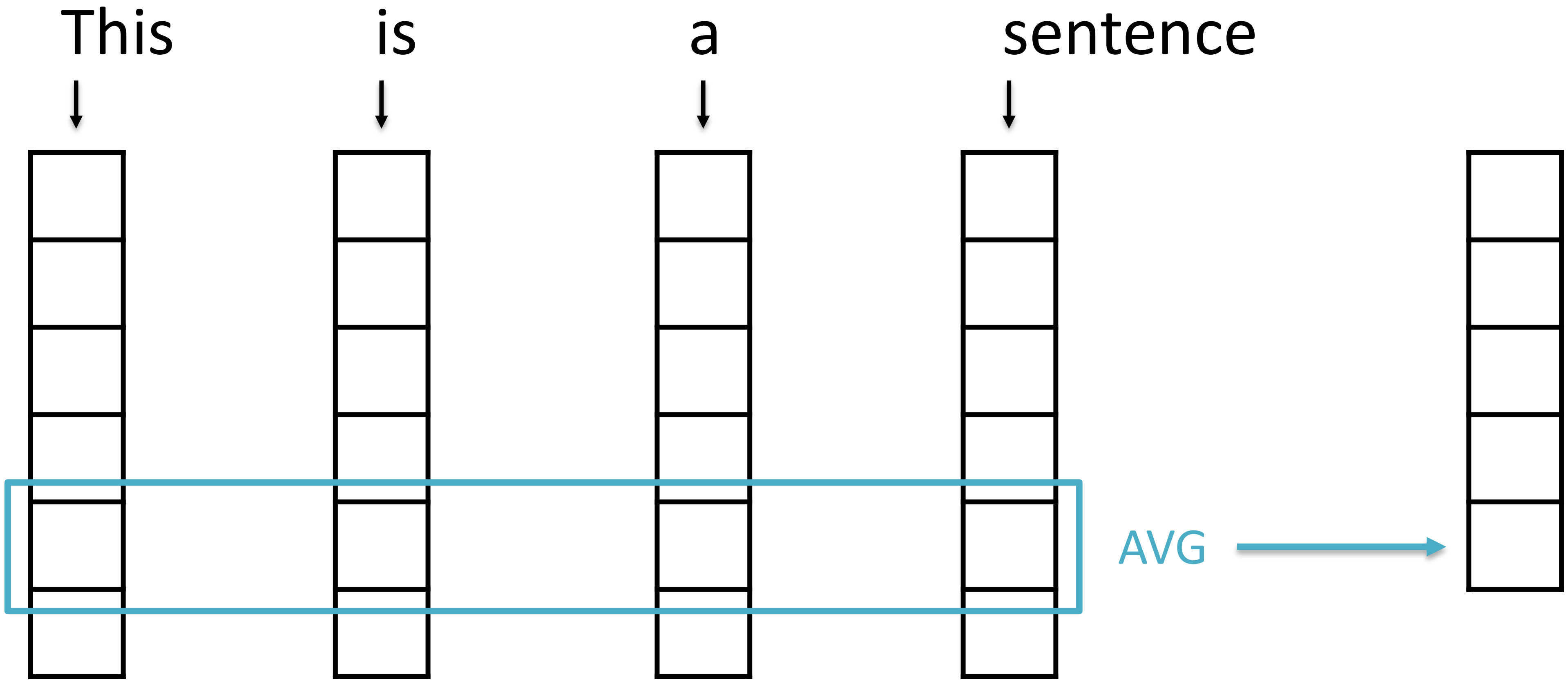
VSWEM Step 3: Take the AVERAGE over the sentence for each embedding dimension



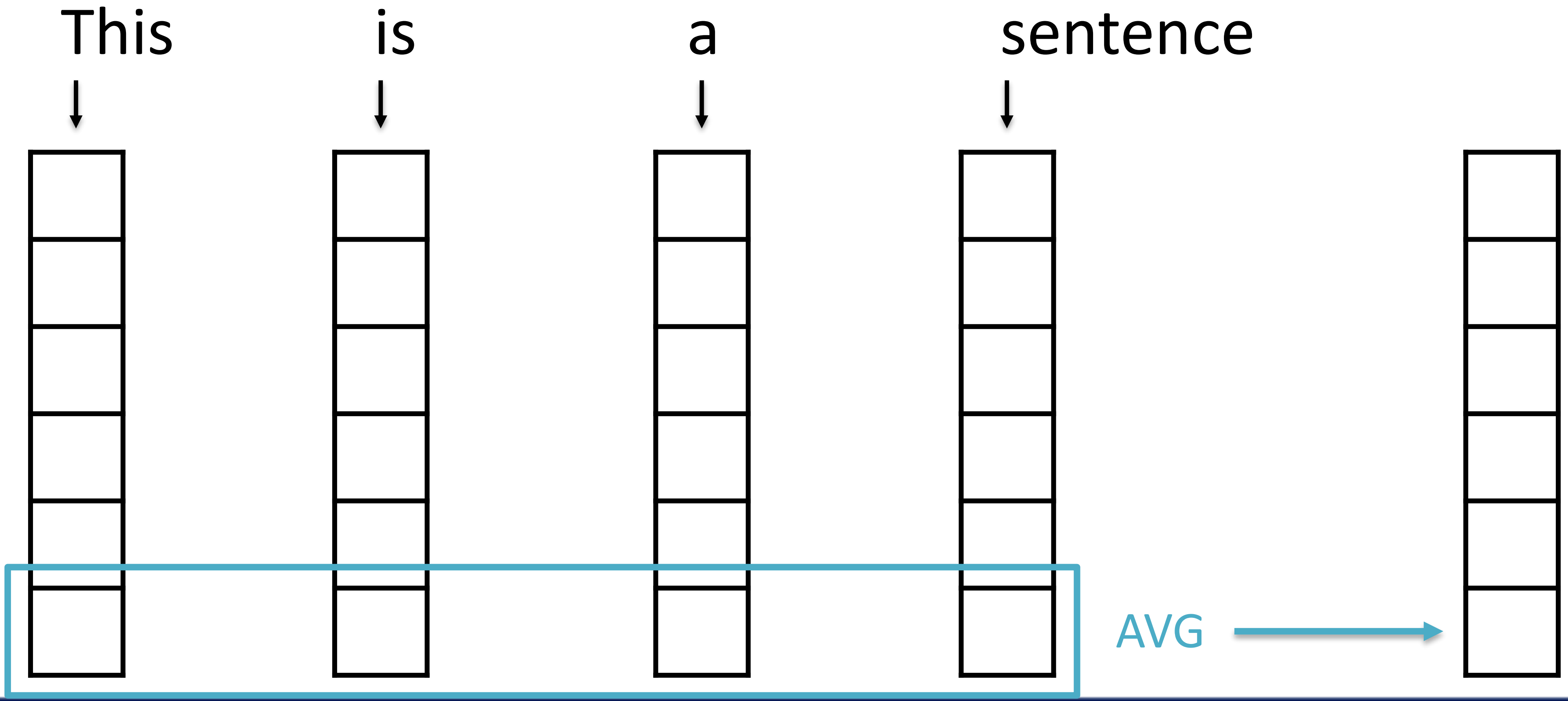
VSWEM Step 3: Take the AVERAGE over the sentence for each embedding dimension



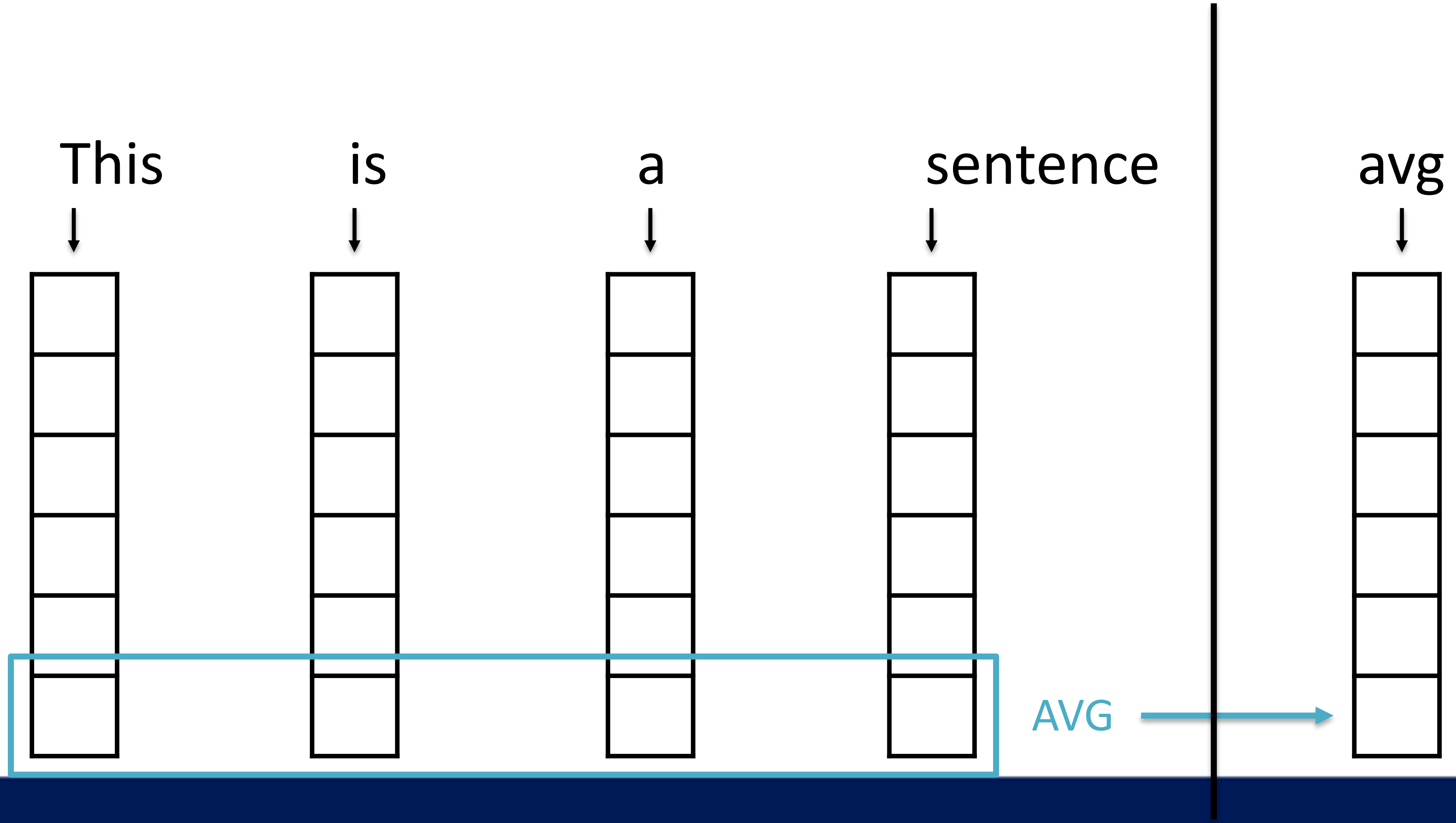
VSWEM Step 3: Take the AVERAGE over the sentence for each embedding dimension



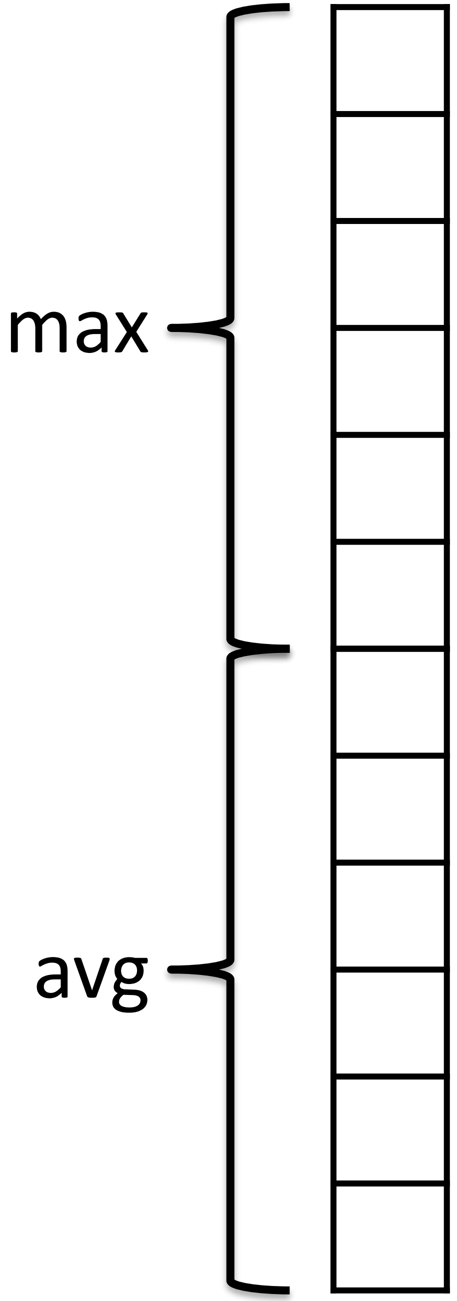
VSWEM Step 3: Take the AVERAGE over the sentence for each embedding dimension



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



VSWEM Step 4: Concatenate MAX and AVG



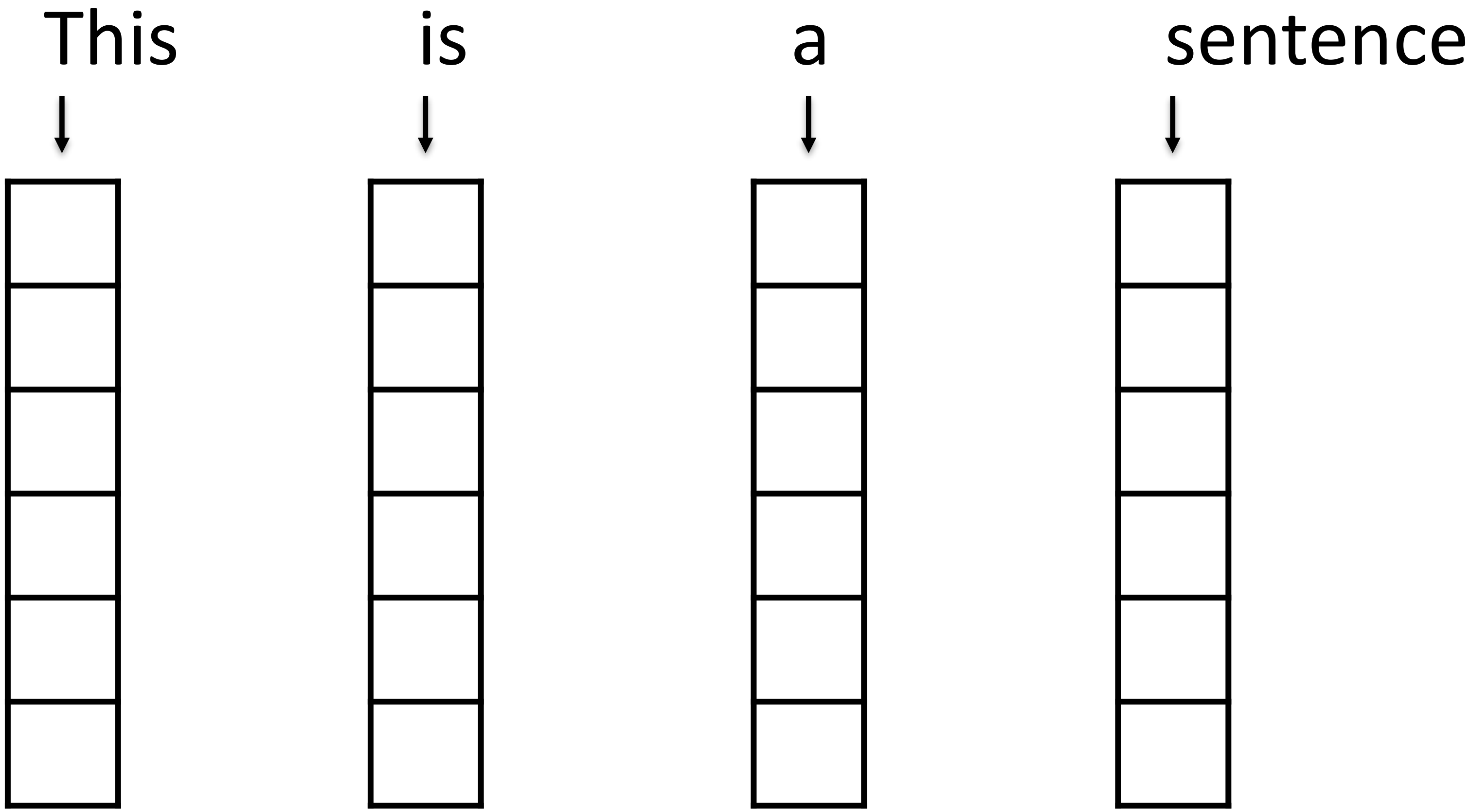
Sentence i → This is a sentence

x_i

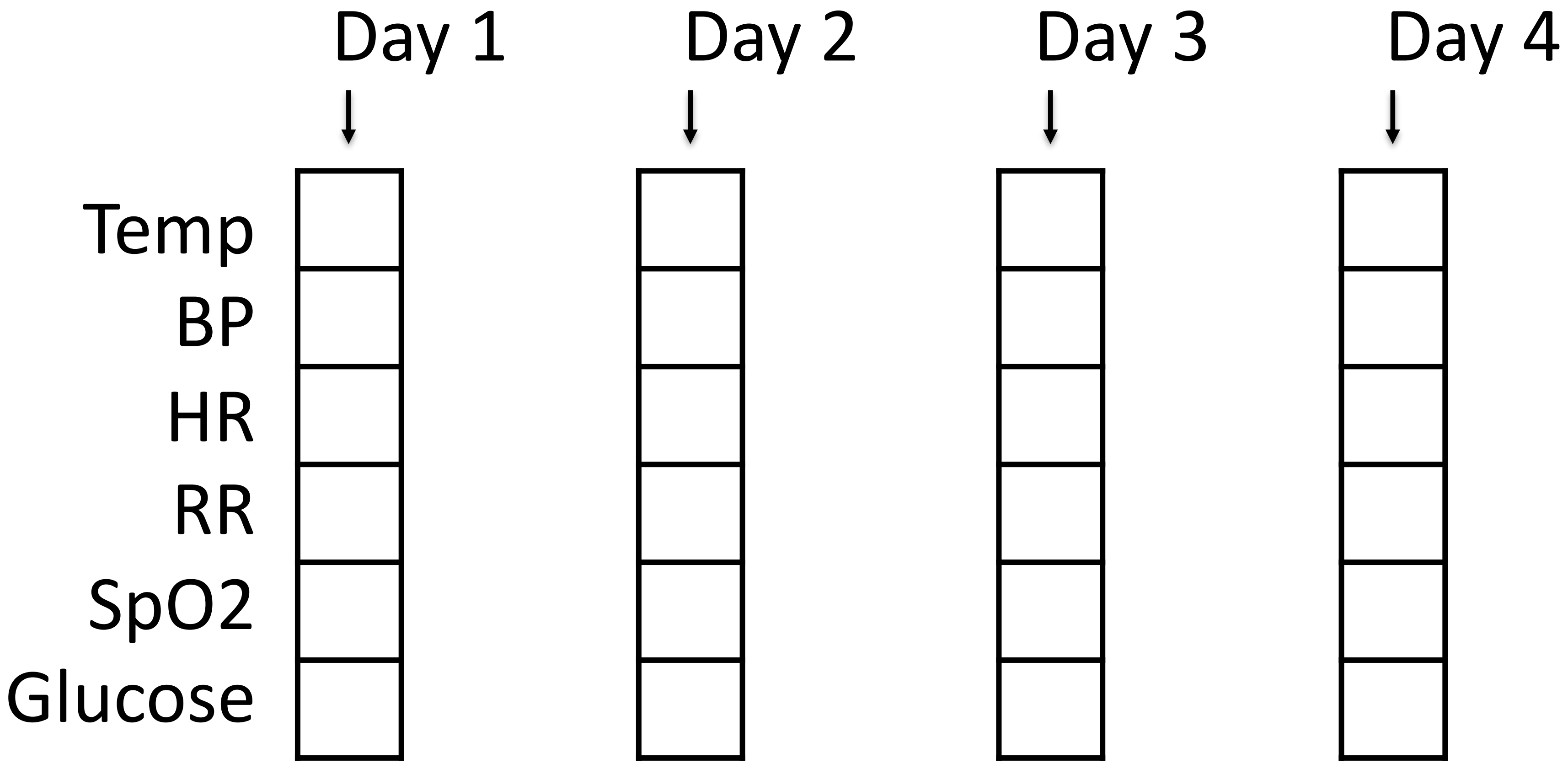
Question:

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

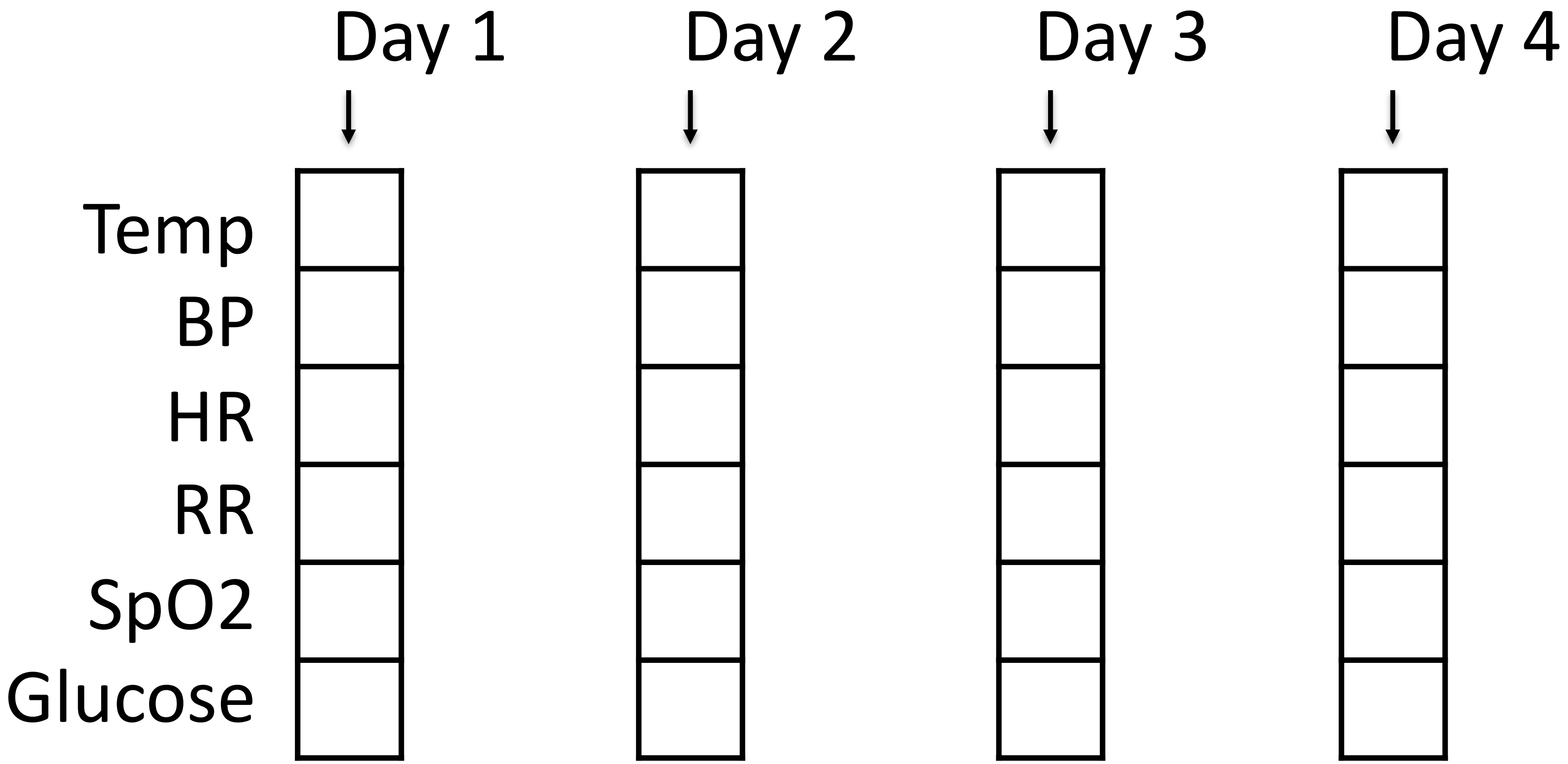
A sequence of word vectors…



…now looks just like a sequence of measurements



In this case, too, we can get a single numeric vector for our predictive models by taking a max and average (or any other summary statistics we'd like)



But when we do this, we lose information about measurement order (or word order).

- Next time, we'll talk about ways to overcome this limitation