Structuralism as the **Origin** of **Self-Supervised Learning** and **Word Embeddings**

Piero Molino

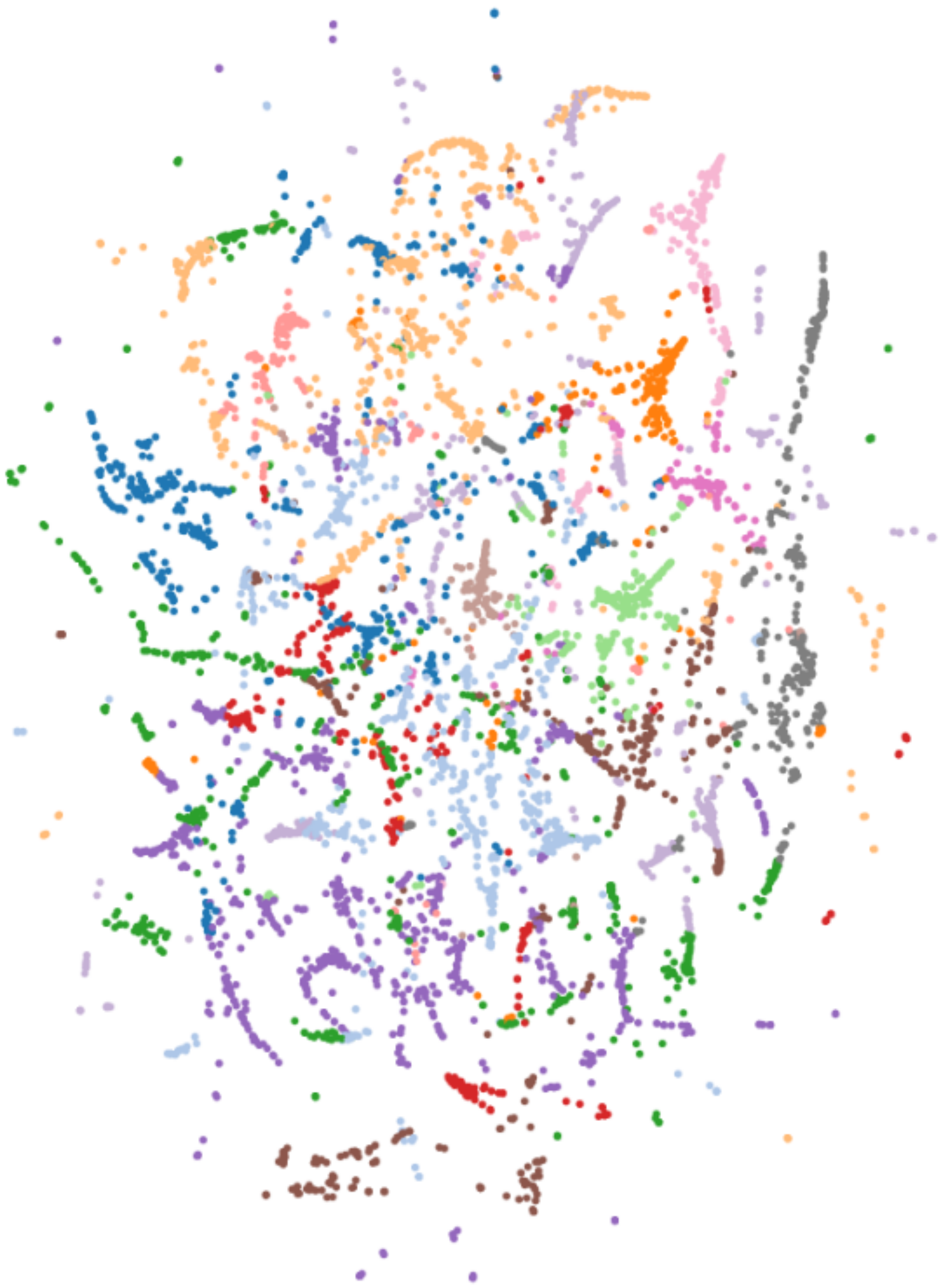
Motivation

Word Embeddings: were a **hot trend** in NLP (Post-word2vec era, 2013-2018)

Self-Supervised Learning: **hot trend** in NLP and ML (Post-ELMo era, 2018+)

Many researchers and practitioner are **oblivious of previous work** in computer science, cognitive science and computational linguistics (Pre-word2vec era: up to 2013)

Delays progress due to reinventing the wheel + <u>many lessons to be learned</u>



Goal

Overview* of the **history** and cultural background to start **building** on **existing knowledge**

Propose a unifying interpretation of **many algorithms**

*__Incomplete personal overview__, a useful **starting point** for exploration

Outline

1. Linguistic background: Structuralism
2. Distributional Semantics
3. Embedding Methods overview
4. Self-Supervised Learning

Terminology

Word Embeddings, Distributed Representations, **Word Vectors**, **Distributional Semantic Models**, Distributional Representations, **Semantic Vector Space**, Word Space, Semantic Space, Geometrical model of Meaning, Context-theoretic models, Corpus-based semantics, Statistical semantics

They all **mean** (almost) the **same thing**

Distributional Semantic Models → Computational Linguistics literature

Word Embeddings → Neural Networks literature

Structuralism

Structuralism

"The belief that phenomena of human life are **not intelligible except** through their **interrelations**.
These relations constitute a **structure**,
and behind local variations in the surface phenomena
there are constant laws of abstract culture"

- Simon Blackburn, Oxford Dictionary of Philosophy, 2008

Origins of Structuralism

Ferdinand de Saussure, *Cours de linguistique générale*, 1916

Published posthumous from notes of his students

Previous ideas close to structuralism:

- Wilhelm von Humboldt, *Über den Dualis*, 1828
- Wilhelm von Humboldt, *Über die Verschiedenheit des menschlichen Sprachbaues*, 1836
- Ferdinand de Saussure, *Mémoire sur le système des primitif voyelles dans les langues indo-européennes*, 1879

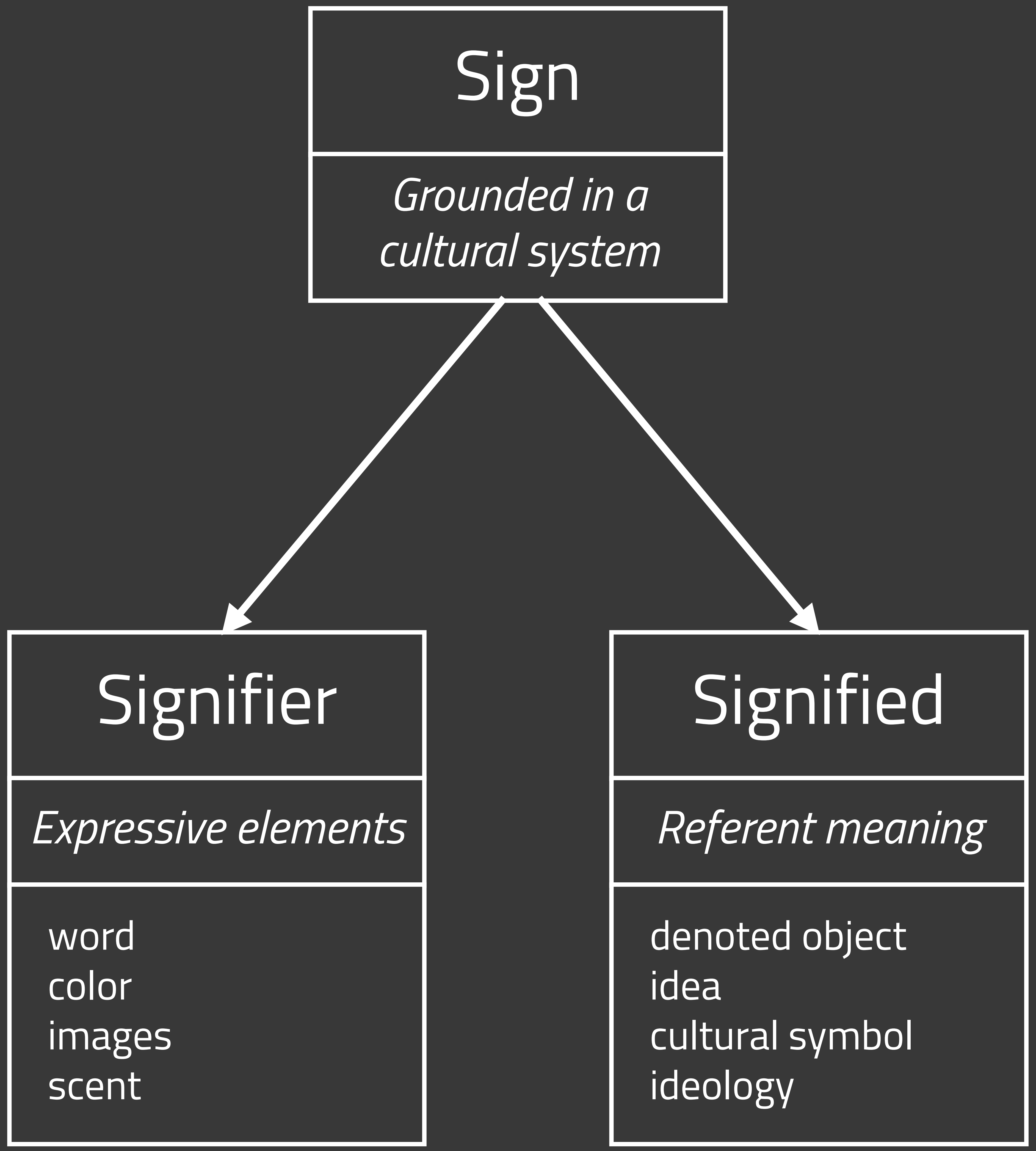
Structuralism and Semiotics

Langue vs *Parole*

Sign, Signifier, Signified

Different languages use **different signifiers** for the **same signified** → the choice of signifiers is **arbitrary**

Meaning of signs is **defined** by their **relationships** and **contrasts** with **other signs**



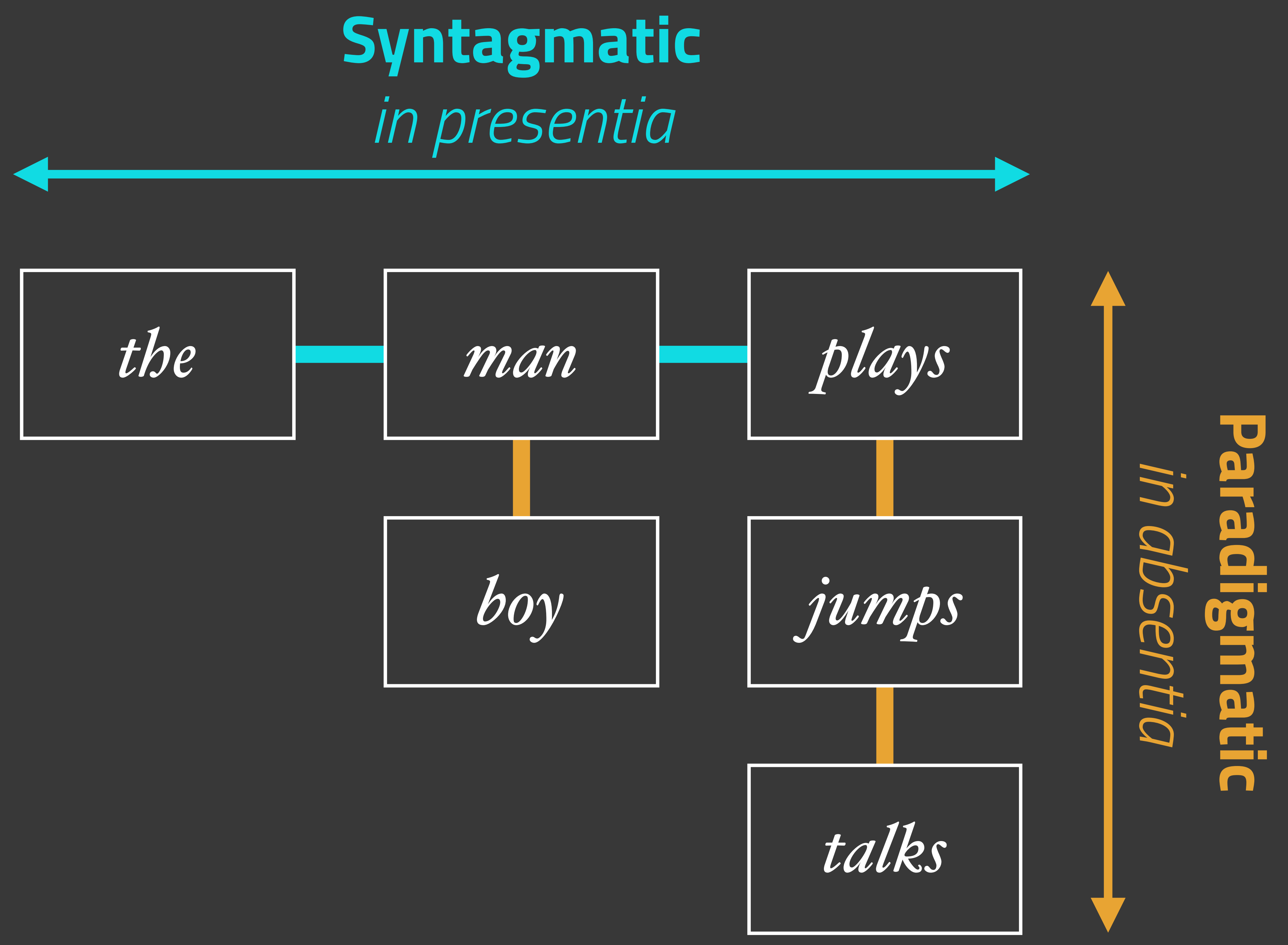
Meaning of signs is **defined** by their **relationships** and **contrasts** with **other signs**

Linguistic relationships

Paradigmatic: relationship between words that can be **substituted** for each other in the same position within a given sentence

Syntagmatic: relationship a word has with other words that **surround** it

Originally de Saussure used the term *'associative'*, the term *'paradigmatic'* was introduced by Louis Hjelmslev, *Principes de grammaire générale*, 1928



Paradigmatic

Synonymy

Bubbling	*Effervescent*	*Sparkling*

Antonymy

Hot ⟷ *Cold*

Hyponymy

Feline — Hypernym

Tiger *Lion* — Hyponym

Syntagmatic

Collocation

against the **law**

law *enforcement*

become **law**

law *is passed*

Colligation

normal

VERB past	**time**
saved	
spent	
wasted	

sport

ADJECTIVE	**time**
half	
extra	
full	

Distributionalism

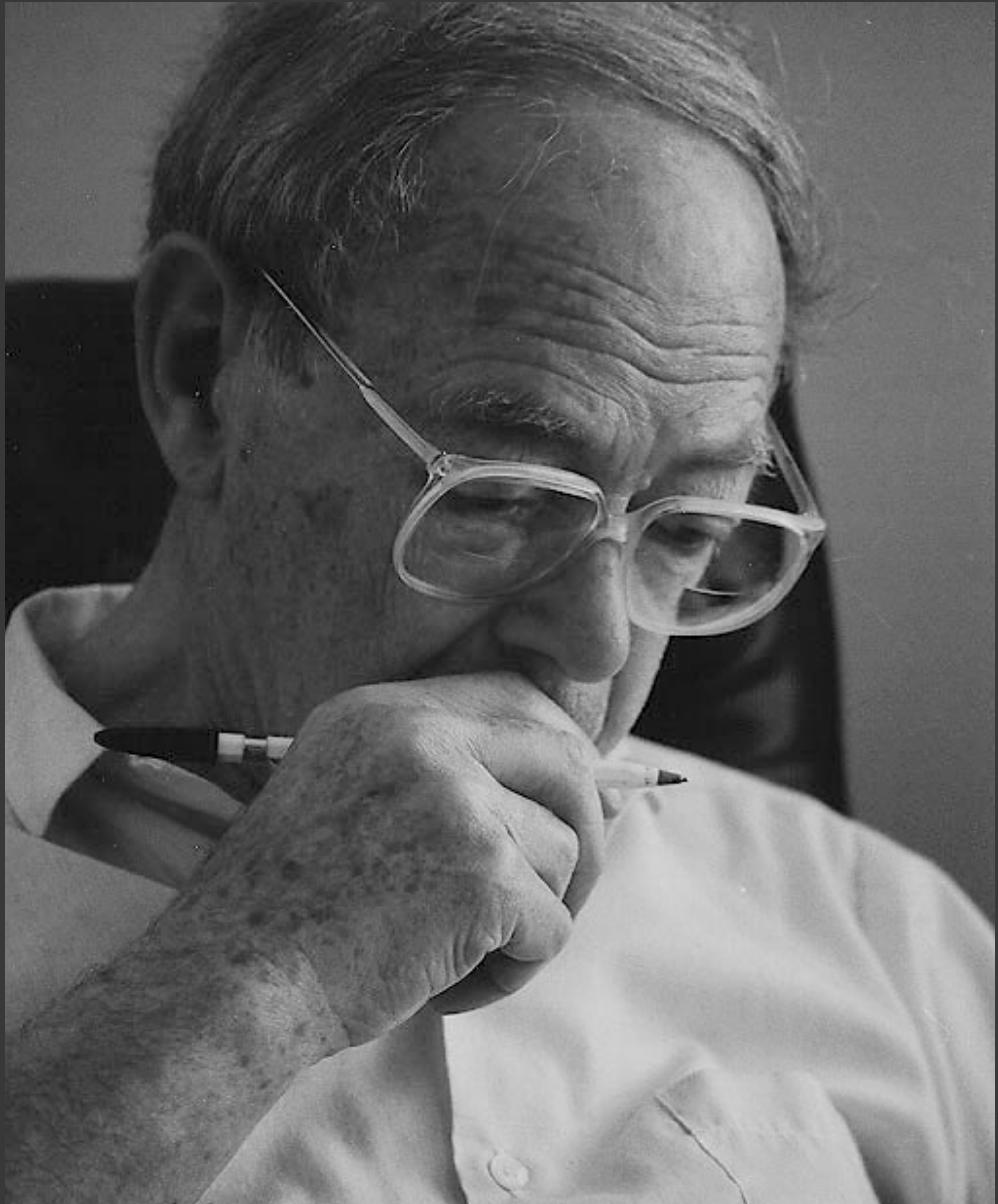
American structuralist branch

Leonard Bloomfield, *Language*, 1933

Zellig Harris. *Methods in Structural Linguistics*, 1951

Zellig Harris, *Distributional Structure*, 1954

Zellig Harris, *Mathematical Structure of Language*, 1968



Philosophy of Language

"The **meaning of a word**
is its **use** in the language"

- Ludwig Wittgenstein, Philosophical Investigation, 1953

Corpus Linguistics

"You shall know a **word** by the **company it keeps**"

- J.R. Firth, Papers in Linguistics, 1957

Other relevant work

Willard Van Orman Quine,
Word and Object, 1960

Margaret Masterman,
The Nature of a Paradigm,
1965

Distributional Semantics

Distributional Hypothesis

The degree of **semantic similarity** between two linguistic expressions *A* and *B* is a function of the **similarity** of the **linguistic contexts** in which *A* and *B* can appear

First formulation by Harris, Charles, Miller, Firth or Wittgenstein?

He filled the **wampimuk**, passed it around and we all drunk some

---

We found a little, hairy **wampimuk** sleeping behind the tree

– McDonald and Ramscar, 2001

He filled the **wampimuk**, passed it around and we all drunk some

---

We found a little, hairy **wampimuk** sleeping behind the tree

– McDonald and Ramscar, 2001

Distributional Semantic Model

1. Represent words through *vectors* recording their **co-occurrence counts** with **context elements** in a **corpus**

2. *(Optionally)* Apply a **re-weighting scheme** to the resulting co-occurrence matrix

3. *(Optionally)* Apply **dimensionality reduction** techniques to the co-occurrence matrix

4. Measure **geometric distance** of word vectors as **proxy** to **semantic similarity** / **relatedness**

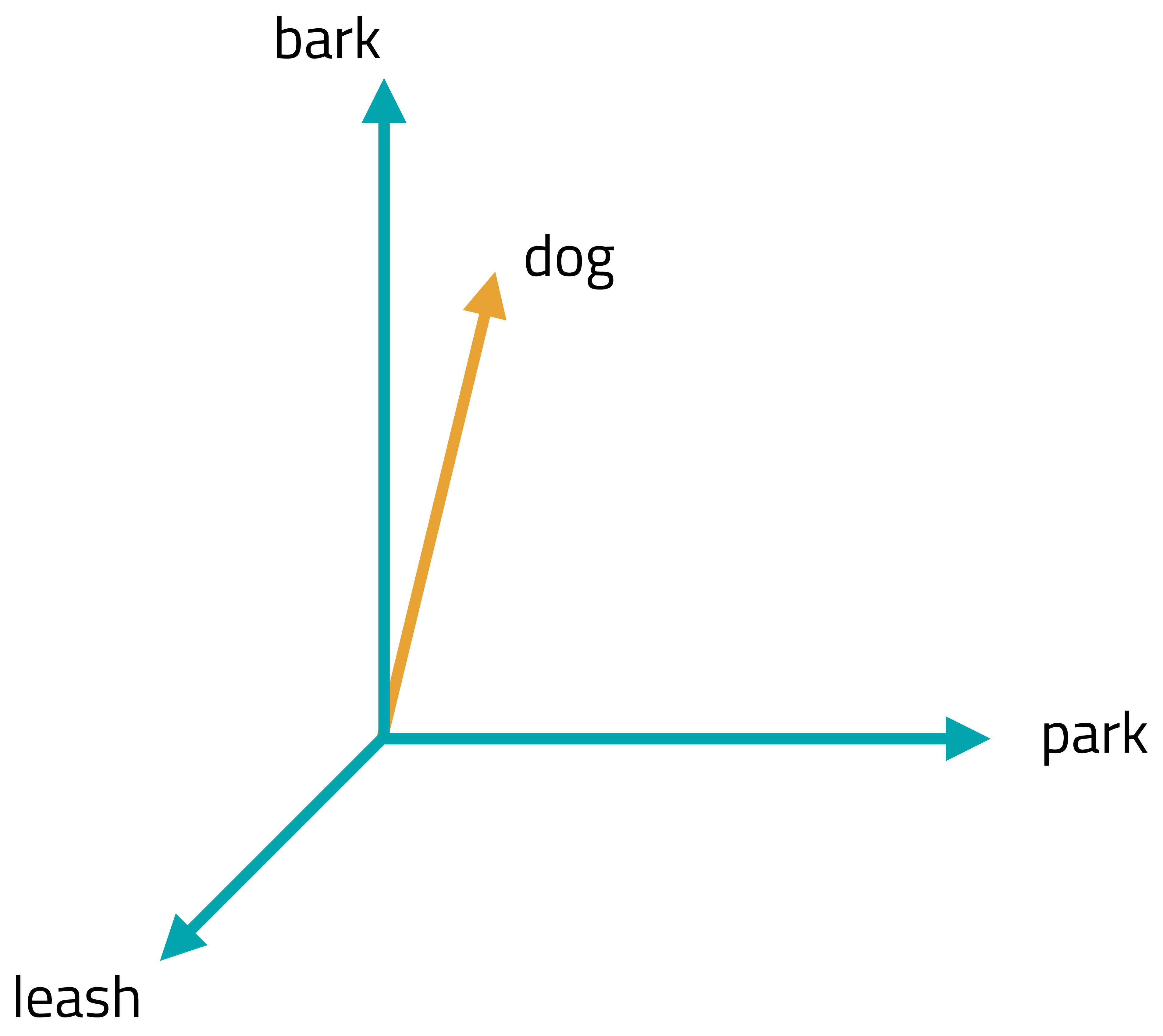
Example

Target: a specific word

Context: noun and verbs in the same sentence

*The **dog** **barked** in the **park**. The **owner** of the **dog** put him on the **leash** since he **barked**.*

word	count
bark	2
park	1
leash	1
owner	1



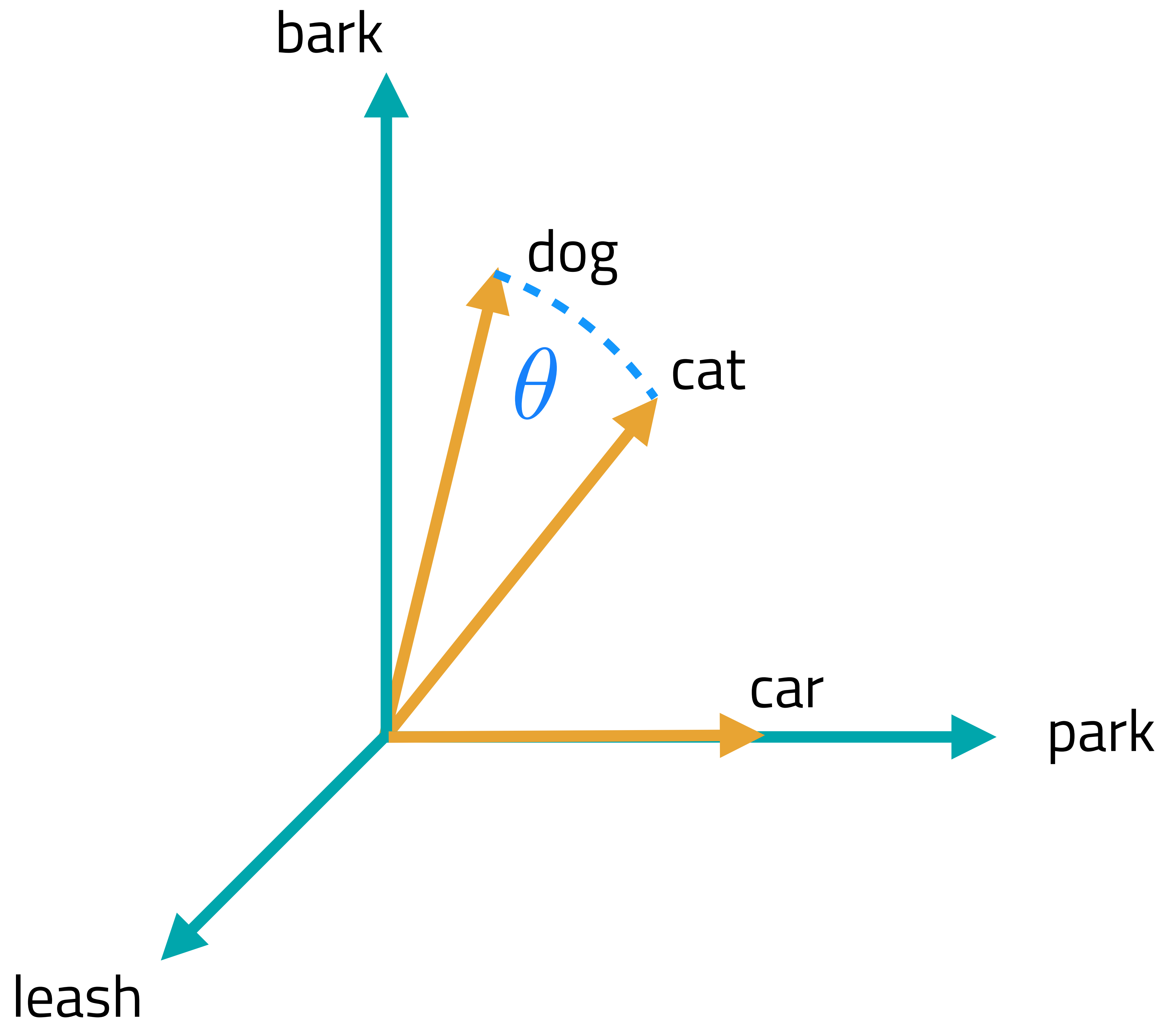
Example

Targets		Contexts					
		leash	walk	run	owner	leg	bark
	dog	3	5	1	5	4	2
	cat	0	3	3	1	5	0
	lion	0	3	2	0	1	0
	light	0	0	1	0	0	0
	dark	1	0	0	2	1	0
	car	0	0	4	3	0	0

Example

Use **cosine similarity** as a measure of **relatedness**

$$\cos\theta = \frac{x \cdot y}{\|x\|\|y\|} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=0}^{n} x_i^2}\sqrt{\sum_{i=0}^{n} y_i^2}}$$



Similarity and Relatedness

Semantic similarity

words sharing salient attributes / features

- synonymy (car / automobile)
- hypernymy (car / vehicle)
- co-hyponymy (car / van / truck)

Semantic relatedness

words semantically associated without being necessarily similar

- function (car / drive)
- meronymy (car / tyre)
- location (car / road)
- attribute (car / fast)

(Budansky and Hirst, 2006)

Context

The **meaning of a word** can be **defined** in terms of its **context** (properties, features)

- Other words in the same document / paragraph / sentence
- Words in the immediate neighbors
- Words along dependency paths
- Linguistic patterns
- Predicate-Argument structures
- Frames
- Hand crafted features

  First attempt in 1960s in Charles Osgood's *semantic differentials*, also used in first connectionist AI approaches in the 1980s

Any process that builds a **structure** on **sentences** can be used as a **source for properties**

Context Examples

Document

DOC1: The silhouette of the **sun**[1] beyond a wide-open bay on the lake; the **sun**[2] still glitters although evening has arrived in Kuhmo. It's midsummer; the living room has its instruments and other objects in each of its corners.

Context Examples

Wide window

DOC1: The silhouette of the **sun**[1] beyond a wide-open bay on the lake; the **sun**[2] still glitters although evening has arrived in Kuhmo. It's midsummer; the living room has its instruments and other objects in each of its corners.

Context Examples

Wide window (content words)

DOC1: [1 The silhouette of the **sun** [1] beyond a [2 wide-open bay on the lake; the **sun** [2] still glitters although evening has arrived in Kuhmo. 1] It's midsummer; the living room has its instruments and other objects in each of its corners. 2]

Context Examples

Small window (content words)

DOC1: [1]The silhouette of the [1]**sun** beyond a wide-open [2]bay on the lake[1]; the [2]**sun** still glitters although evening has arrived[2] in Kuhmo. It’s midsummer; the living room has its instruments and other objects in each of its corners.

Context Examples

PoS coded content lemmas

DOC1: [1]The silhouette/N of the [1]**sun** beyond a wide-open/A [2]bay/N on the lake/N[1]; the [2]**sun** still glitters/V although evening/N has arrive/V[2] in Kuhmo. It’s midsummer; the living room has its instruments and other objects in each of its corners.

Context Examples

PoS coded content lemmas filtered by syntactic path

PPDEP

DOC1: The silhouette/N of the **sun** beyond a wide-open bay on the lake; the

SUBJ

sun still glitters/V although evening has arrived in Kuhmo. It's midsummer;

the living room has its instruments and other objects in each of its corners.

Context Examples

Syntactic path coded lemmas

PPDEP

DOC1: The silhouette/N_PPDEP of the **sun** beyond a wide-open bay on the lake;

SUBJ

the **sun** still glitters/V_SUBJ although evening has arrived in Kuhmo. It's midsummer; the living room has its instruments and other objects in each of its corners.

Effect of Context

Neighbors of ***dog*** in *BNC Corpus*

	2-word window	30-word window	
More paradigmatic	cat	<u>kennel</u>	**More syntagmatic**
	horse	puppy	
	fox	pet	
	pet	bitch	
	rabbit	terrier	
	pig	rottweiler	
	animal	canine	
	mongrel	cat	
	sheep	<u>bark</u>	
	pigeon	alsatian	

Effect of Context

Neighbors of ***Turing*** in *Wikipedia*

Co-hyponyms **Paradigmatic**	**Syntactic dependencies**	**5-word window**	**Topically related** **Syntagmatic**
	Pauling	nondeterministic	
	Hotelling	non-deterministic	
	Heting	computability	
	Lessing	deterministic	
	Hamming	finite-state	

Weighting Schemes

So far we used **raw counts**

Several other options for populating the *target* x *context* matrix are available

In most cases **Positive Pointwise Mutual Information** is the best choice

Kiela and Clark, *A systematic study of Semantic Vector Space Parameters*, 2014, is a good review

Scheme	Definition
None	$w_{ij} = f_{ij}$
TF-IDF	$w_{ij} = \log(f_{ij}) \times \log(\frac{N}{n_j})$
TF-ICF	$w_{ij} = \log(f_{ij}) \times \log(\frac{N}{f_j})$
Okapi BM25	$w_{ij} = \frac{f_{ij}}{0.5+1.5\times\frac{f_j}{\frac{f_j}{j}}+f_{ij}} \log \frac{N-n_j+0.5}{f_{ij}+0.5}$
ATC	$w_{ij} = \frac{(0.5+0.5\times\frac{f_{ij}}{max_f})\log(\frac{N}{n_j})}{\sqrt{\sum_{i=1}^{N}[(0.5+0.5\times\frac{f_{ij}}{max_f})\log(\frac{N}{n_j})]^2}}$
LTU	$w_{ij} = \frac{(\log(f_{ij})+1.0)\log(\frac{N}{n_j})}{0.8+0.2\times f_j\times\frac{j}{f_j}}$
MI	$w_{ij} = \log \frac{P(t_{ij}\|c_j)}{P(t_{ij})P(c_j)}$
PosMI	$\max(0, \mathrm{MI})$
T-Test	$w_{ij} = \frac{P(t_{ij}\|c_j)-P(t_{ij})P(c_j)}{\sqrt{P(t_{ij})P(c_j)}}$
χ^2	see (Curran, 2004, p. 83)
Lin98a	$w_{ij} = \frac{f_{ij}\times f}{f_i\times f_j}$
Lin98b	$w_{ij} = -1 \times \log \frac{n_j}{N}$
Gref94	$w_{ij} = \frac{\log f_{ij}+1}{\log n_j+1}$

Similarity Measures

So far we used **cosine similarity**

Several other options for computing similarity are available

In most cases **Correlation** is the best choice (cosine similarity of vectors normalized by their mean)

Kiela and Clark, *A systematic study of Semantic Vector Space Parameters*, 2014, is a good review

Measure	Definition
Euclidean	$\frac{1}{1+\sqrt{\sum_{i=1}^{n}(u_i-v_i)^2}}$
Cityblock	$\frac{1}{1+\sum_{i=1}^{n}\lvert u_i-v_i\rvert}$
Chebyshev	$\frac{1}{1+\max_i \lvert u_i-v_i\rvert}$
Cosine	$\frac{u\cdot v}{\lvert u\rvert\lvert v\rvert}$
Correlation	$\frac{(u-\mu_u)\cdot(v-\mu_v)}{\lvert u\rvert\lvert v\rvert}$
Dice	$\frac{2\sum_{i=0}^{n} min(u_i,v_i)}{\sum_{i=0}^{n} u_i+v_i}$
Jaccard	$\frac{u\cdot v}{\sum_{i=0}^{n} u_i+v_i}$
Jaccard2	$\frac{\sum_{i=0}^{n} min(u_i,v_i)}{\sum_{i=0}^{n} max(u_i,v_i)}$
Lin	$\frac{\sum_{i=0}^{n} u_i+v_i}{\lvert u\rvert+\lvert v\rvert}$
Tanimoto	$\frac{u\cdot v}{\lvert u\rvert+\lvert v\rvert-u\cdot v}$
Jensen-Shannon Div	$1-\frac{\frac{1}{2}(D(u\|\frac{u+v}{2})+D(v\|\frac{u+v}{2}))}{\sqrt{2\log 2}}$
α-skew	$1-\frac{D(u\|\alpha v+(1-\alpha)u)}{\sqrt{2\log 2}}$

Evaluation

Intrinsic

- evaluate **word pairs similarities** → compare with **similarity judgments** given by humans *(WordSim, MEN, Mechanical Turk, SImLex)*
- evaluate on **analogy tasks** *'Paris is to France as Tokyo is to* ***x****' (MSR analogy, Google analogy)*

Extrinsic

- use the vectors in a **downstream task** *(classification, translation, ...)* and evaluate the **final performance** on the task

Best parameters configuration?

(context, similarity measure, weighting, ...)

Depends on the task!

Methods overview

Methods

Semantic Differential *(Osgood at al. 1957)*

Semantic features *(Smith at al. 1974)*

Mechanisms of sentence processing assigning roles to constituents *(McLelland and Kawamoto 1986)*

Learning Distributed Representations of Concepts *(Hinton et al. 1986)*

Forming Global Representations with Extended Back-Propagation [FGREP] *(Mikkulainen and Dyer 1987)*

Sparse Distributed Memory [SDM] *(Kanerva 1988)*

Latent Semantic Analysis [LSA] *(Deerwester et al.1988-1990)*

Hyperspace Analogue to Language [HAL] *(Lund and Burgess 1995)*

Probabilistic Latent Semantic Analysis [pLSA] *(Hoffman et al. 1999)*

Random Indexing *(Kanerva et al. 2000)*

Latent Dirichlet Allocation [LDA] *(Blei et al. 2003)*

A neural probabilistic language model *(Bengio et al. 2003)*

Infomap *(Widdows et al. 2004)*

Correlated Occurrence Analogue to Lexical Semantic [COALS] *(Rohde et al. 2006)*

Dependency Vecotrs *(Padó and Lapata 2007)*

Explicit Semantic Analysis *(Gabrilovich and Markovich 2007)*

Distributional Memory *(Baroni and Lenci 2009)*

Non-Negative Matrix Factorization [NNMF] *(Van de Cruys et al. 2010) originally: (Paatero and Tapper 1994)*

JoBimText *(Biemann and Riedl 2013)*

word2vec [SGNS and CBOW] *(Mikolov et al. 2013)*

vLBL and ivLBL *(Mnih and Kavukcuoglu 2013)*

Hellinger PCA (HPCA) *(Lebret and Collobert 2014)*

Global Vectors [GloVe] *(Pennington et al. 2014)*

Infinite Dimensional Word Embeddings *(Nalisnick and Ravi 2015)*

Gaussian Embeddings *(Vilnis and McCallum 2015)*

Diachronic Word Embeddings *(Hamilton et al. 2016)*

WordRank *(Ji et al. 2016)*

Exponential Family Embeddings *(Rudolph et al. 2016)*

Multimodal Word Distributions *(Athiwaratkun and Wilson 2017)*

...

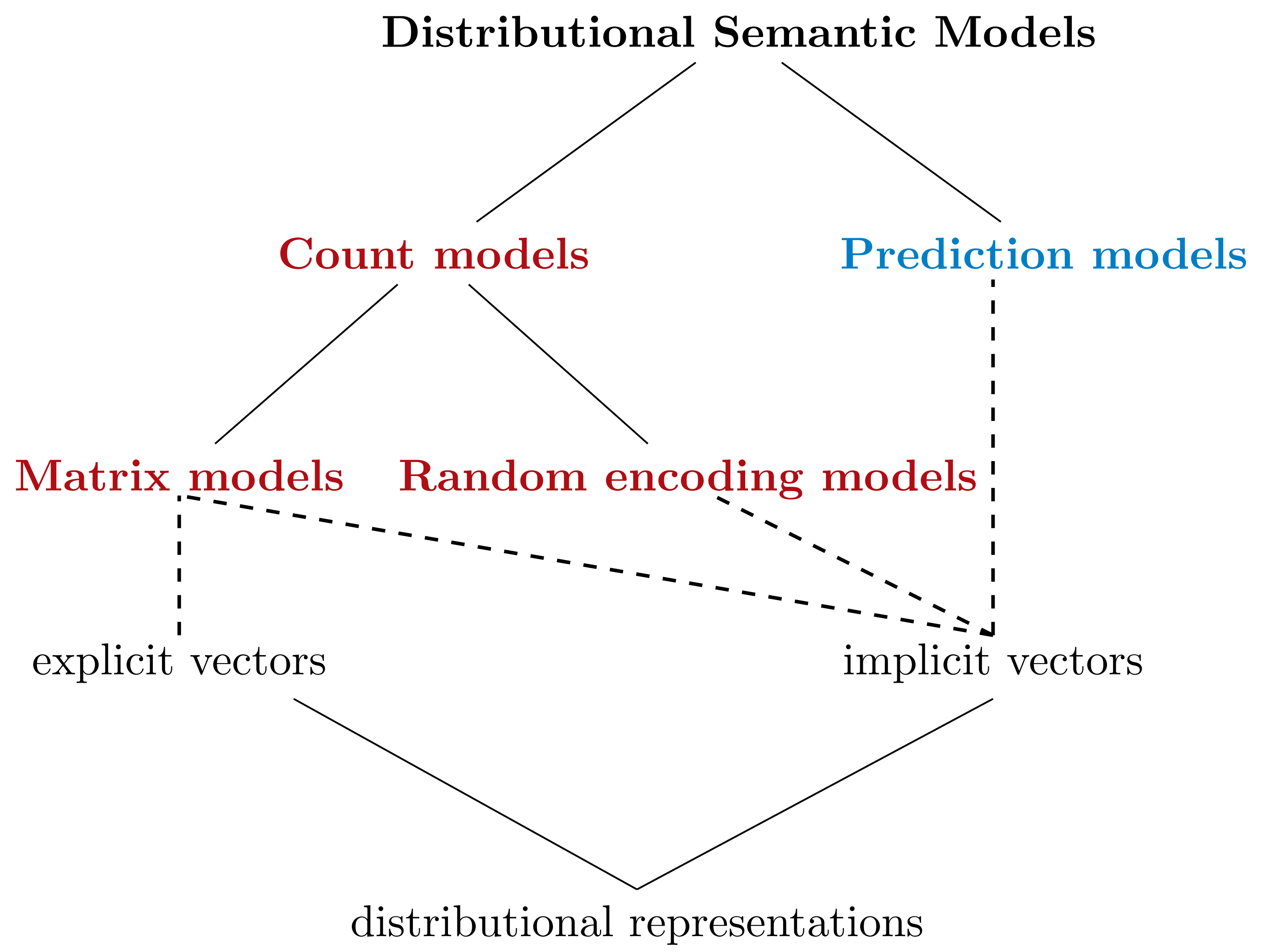
Explicit vs Implicit

Explicit vectors: **big sparse** vectors with **interpretable** dimensions

Implicit vectors: **small dense** vectors with **latent** dimensions

Count vs **Prediction**

Alessandro Lenci, *Distributional models of word meaning*, 2017



Hyperspace Analogue to Language [HAL]

Target: a specific word

Context: window of ten words

Weighting: (10 - distance from target) for each occurrence

Similarity: euclidean distance

Dimensionality reduction: sort **contexts** (columns of the matrix) by variance and keep top 200

the dog barked at the cat

weight $_{\text{dog barked}}$ = 10 *(no gap)*

weight $_{\text{dog cat}}$ = 7 *(3 words gap)*

	c_2	c_7	...	c_3	c_5	...	...	c_6
w_1	54	23	...	8	4	...	...	1
w_1	21	82	...	10	6	...	...	0
...	...	...	...	...	...	...	...	...
w_n	32	47	...	9	3	...	...	1
variance	30	25	...	5	3	...	...	0,5

top 200 keep

201+ discard

Hyperspace Analogue to Language

Advantages

- Simple
- Fast $O(n)$

Disadvantages

- No higher order interactions (only direct co-occurrence)

Latent Semantic Analysis [LSA]

Target: a specific word

Context: document id

Weighting: tf-idf *(term frequency - inverse document frequency), but can use others*

Similarity: cosine

Dimensionality reduction: Singular Value Decomposition (SVD)

TF **IDF**

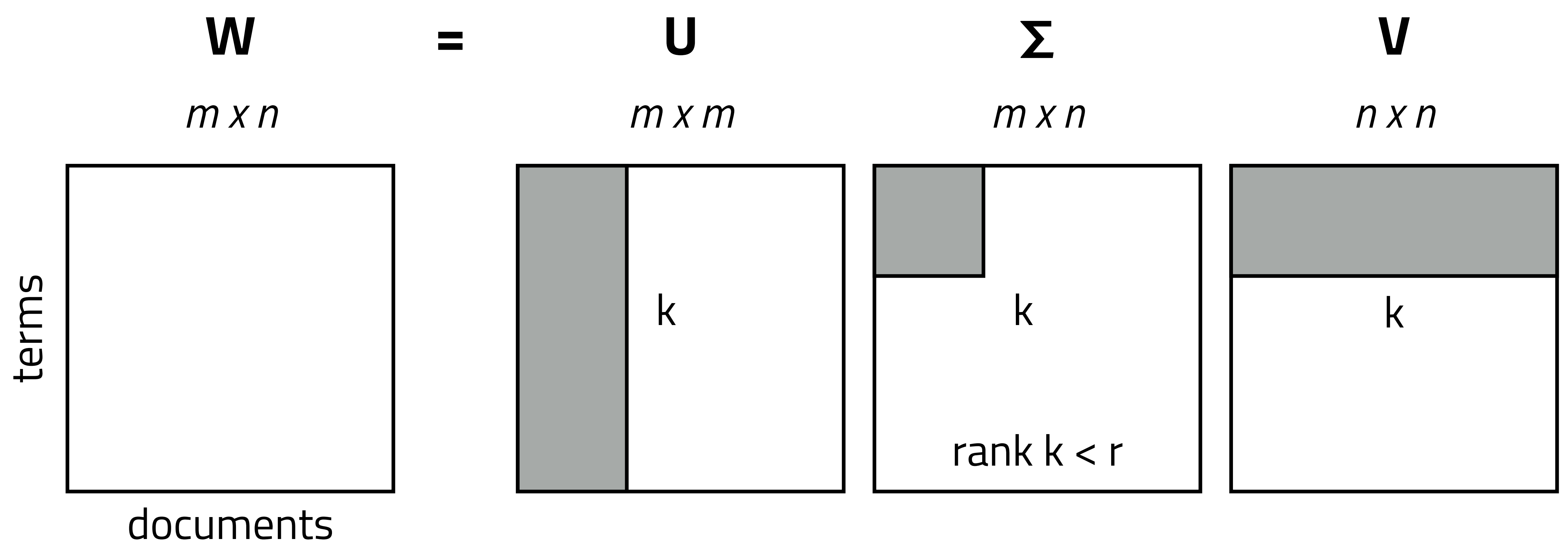
$$weight_{ij} = \log(f_{ij}) \cdot \log(\frac{N}{n_j})$$

frequency of word *j* in document *i*

total documents over documents containing word j

Intuition: the more frequency in the document, the better. The less frequent in the corpus, the better

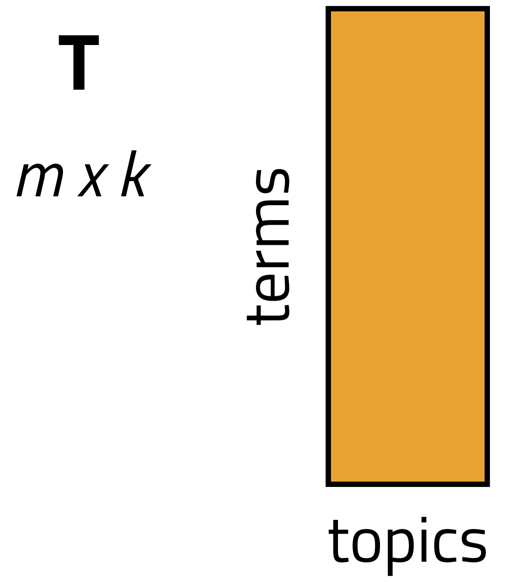
SVD in a nutshell



$$W = U_k \Sigma_k V_k^\top$$

Intuition
keep top *k* singular values as they contain most of the variance
k can be interpreted as the number of topics

Target matrix

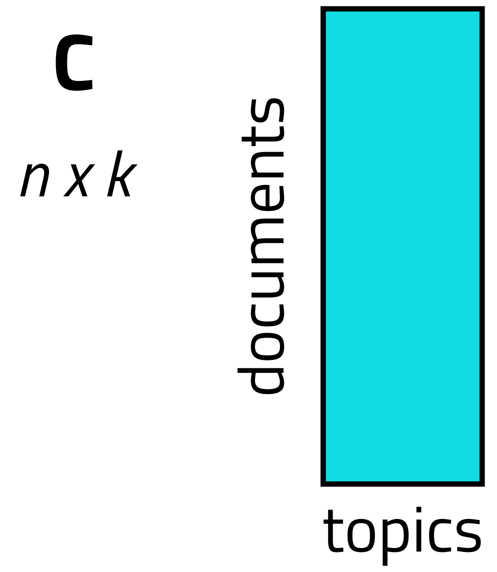


$$T^{SVD} = U_k \Sigma_k$$

Trick from *(Levy at al. 2015)*: throw **Σ** away for better performance

$$T^{SVD} = U_k$$

Context matrix



$$C^{SVD} = V_k^\top$$

Latent Semantic Analysis

Advantages

- Reduced dimension *k* can be interpreted as topics
- Reducing the number of columns unveils higher order interactions

Disadvantages

- Static → can't easily add new documents, words and topics
- SVD is one time operation, without intermediate results
- Expensive in terms of memory and computation $O(k^2m)$

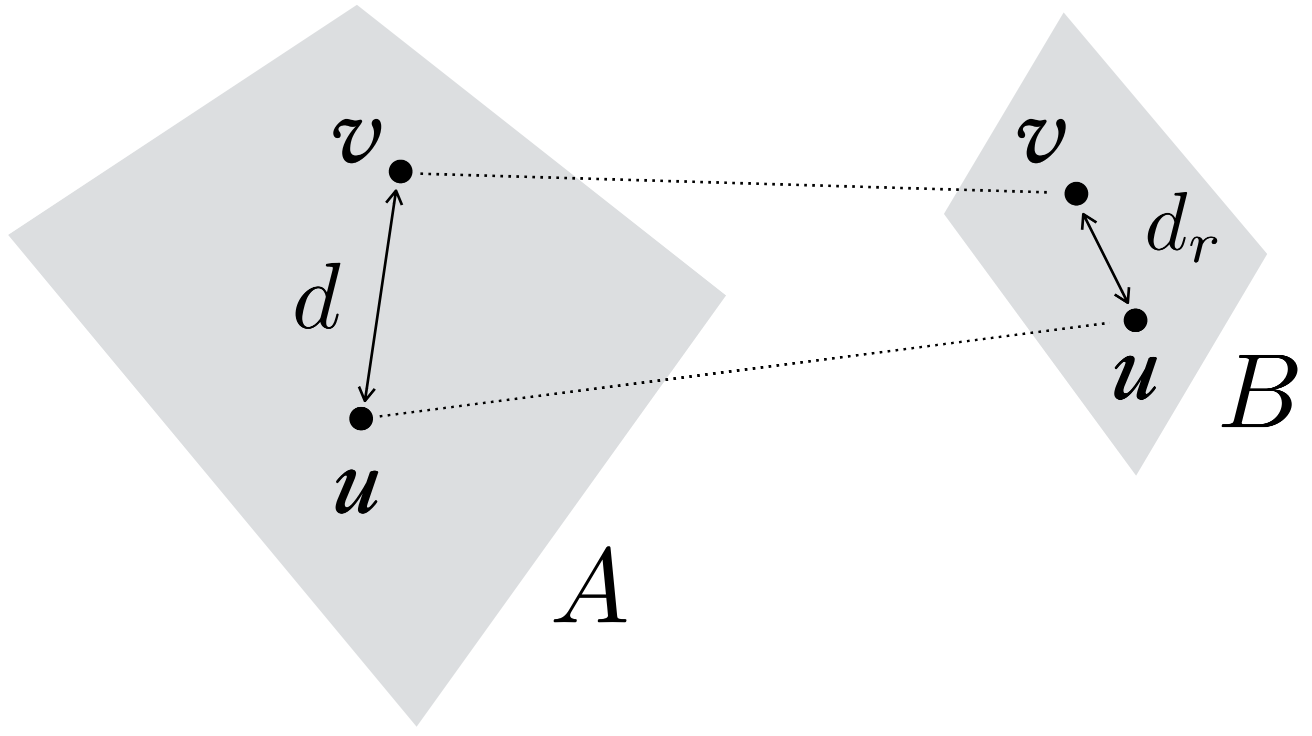
Random Indexing [RI]

Locality-sensitive hashing method that **approximates** the **distance** between points

Generates ***random matrix R*** and projects the ***original matrix A*** to it to obtain a ***reduced matrix B***

Reduced space B preserves the *euclidean distance* between points in ***original space A*** *(Johnson-Lindenstrauss lemma)*

$$B^{n,k} \approx A^{n,m} R^{m,k} \quad k \ll m$$



$$(1-\epsilon)d_r(v,u) \leq d(v,u) \leq (1+\epsilon)d_r(v,u)$$

Random Indexing [RI]

Algorithm

- For every word in the corpus create a **sparse random context** vector with values in *{-1, 0, 1}*
- **Target** vectors are the **sum** of the **context** vectors of the words they co-occur with multiplied by the **frequency** of the co-occurrence

Dataset

I drink beer
You drink a glass of beer

Context Vectors

I	1	0	0	0	0	-1	0
drink	0	0	1	0	0	0	0
beer	0	1	0	0	0	0	0
you	0	-1	0	0	0	0	1
glass	-1	0	0	0	1	0	0

Target Vectors

$$tv_{beer} = 1cv_i + 2cv_{drink} + 1cv_{you} + 1cv_{glass}$$

beer	0	-1	2	0	1	-1	1

Random Indexing

Advantages

- Fast *O(n)*
- Incremental → can add new words any time, just create a new **context** vector

Disadvantages

- In many intrinsic tasks doesn't perform as well as other methods
- Stochasticity in the process → random distortion
- Negative similarity scores

Explicit Semantic Analysis [ESA]

Target: a specific word

Context: Wikipedia article

Assumption: Wikipedia articles are **explicit topics**

Weighting: tf-idf

Similarity: cosine

Dimensionality Reduction: discard *too short* articles and articles with only few other articles *linking* to them

	Mouse [Rodent]	Mouse [computing]	Mickey Mouse	Button	Janson Button	Drag and Drop
mouse	**0,95**	**0,89**	**0,81**	0,50	0,01	0,60
button	0,10	**0,81**	0,20	**0,95**	**0,89**	**0,70**

mouse button	0,50	**0,85**	0,50	**0,72**	0,45	**0,65**

average of 2 vectors → emerges disambiguated meaning

	cat	leopard	jaguar	car	animal	button
Panther	**0,83**	**0,72**	**0,65**	0,3	**0,92**	0,01

Explicit Semantic Analysis

Advantages

- Simple
- Fast *O(n)*
- Interpretable

Disadvantages

- The assumption doesn't always hold
- Doesn't perform as good as other methods
- Vectors are really high dimensional, although quite sparse

JoBimText

Generic holing **@** operation

Apply it to any tuple to obtain **targets** (jo) and **contexts** (bim)

Weighting: custom measure similar to Lin

Similarity: Lexicographer Mutual Information *(PMI x Frequency)* (Kilgarriff et al. 2004)

Input tuple
(nsubj, gave, I)
(det, book, a)
(dobj, gave, book)
(det girl, the)
(prep_to, gave, girl)

target	context
I	(nsubj, gave, @)
gave	(nsubj, @, I)
a	(det, book, @)
book	(det, @, a)
…	…
girl	(prep_to, gave, @)
gave	(prep_to, @, girl)

Input tuple
(I, gave, a, book)
(gave, a, book, to)
(a, book, to, the)
(book, to, the, girl)

target	context
I	(@, gave, a, book)
gave	(I, @, a, book)
a	(I, gave, @, book)
book	(I, gave, a, @)
…	…
the	(book, to, @, girl)
girl	(book, to, the, @)

JoBimText

Advantages

- Generic preprocessing operation deals with many context representations and types of data
- Deals with complex contexts (example: several steps in a tree)

Disadvantages

- No dimensionality reduction → vectors are high dimensional
- No uncovering of higher order relations
- MapReduce implementation only effective on clusters

word2vec

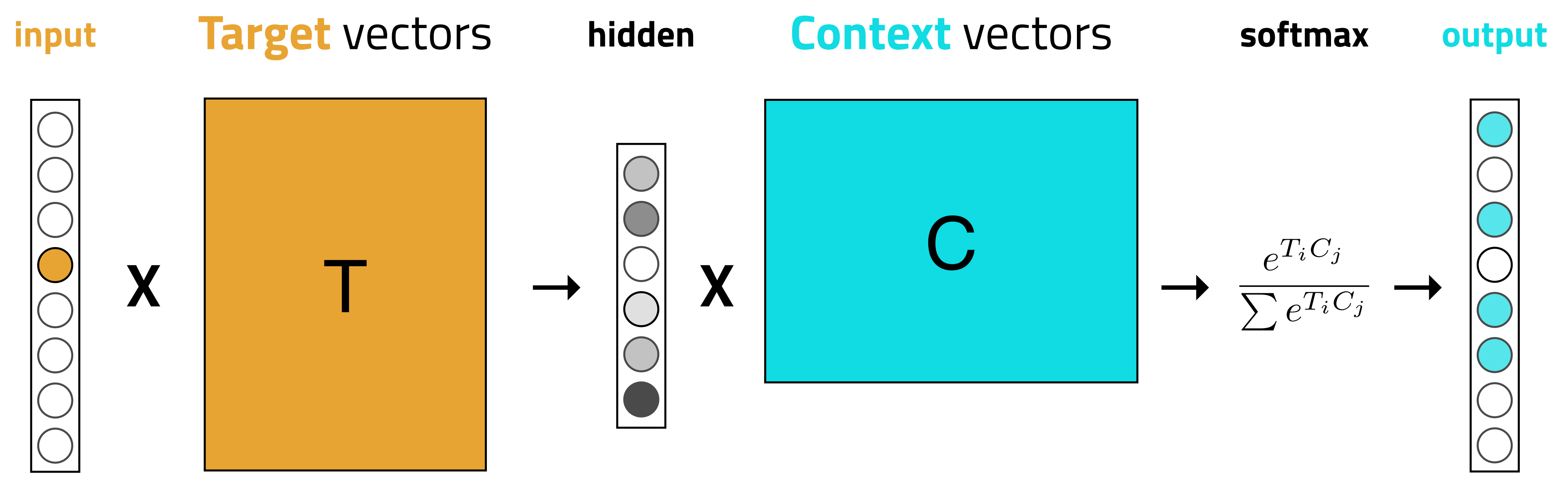
Skip Gram with Negative Sampling (SGNS)

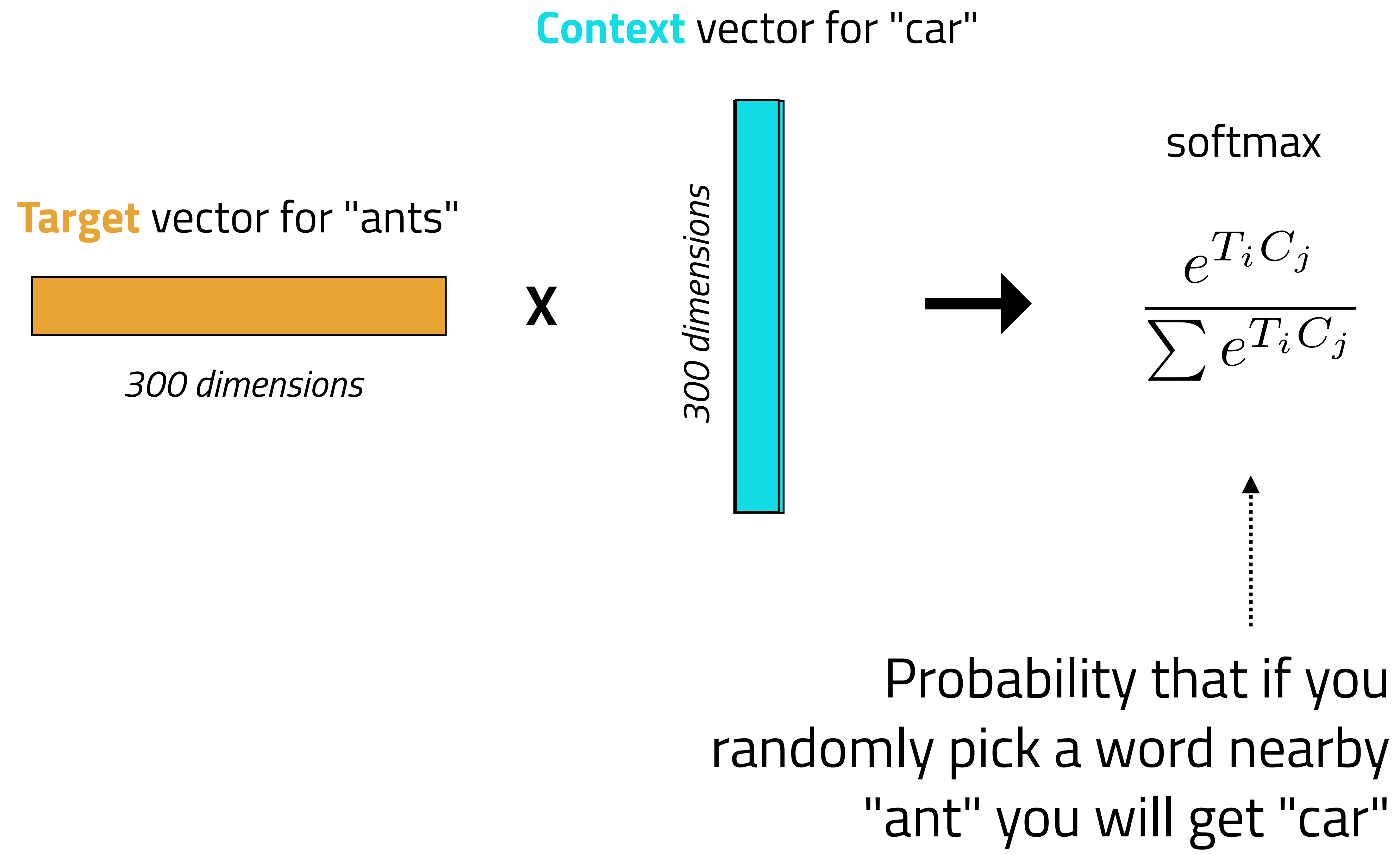
Target: a specific word

Context: window of *n* words

Vectors are obtained training the model to predict the **context** given a **target**

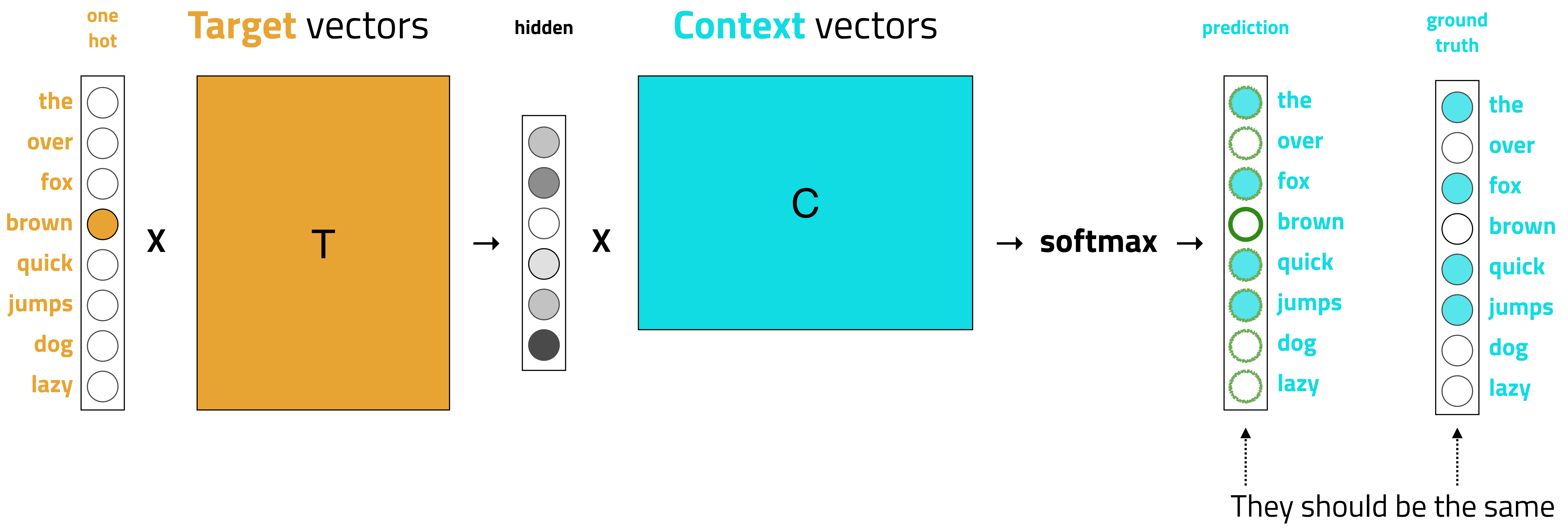
The **error** of the **prediction** is **back-propagated** and the **vectors updated**





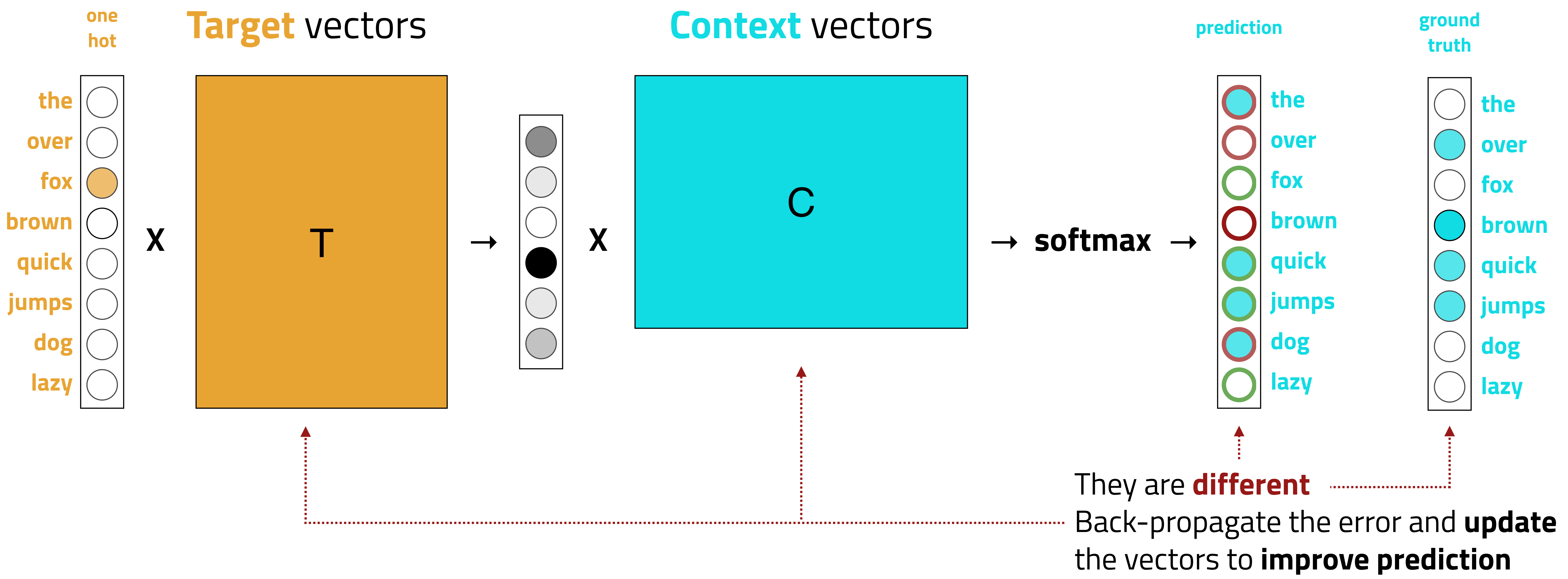
Example

The quick brown fox jumps over the lazy dog



Example

The quick brown fox jumps over the lazy dog



Example Negative Sampling

Calculating the full softmax is **expensive** because of **large vocabulary**

The quick brown fox jumps over the lazy dog

1. Create pairs of **target** and **context** words and predict the probability of them co-occurring to be **1**

   (fox, quick) → **1**
   (fox, brown) → **1**
   (fox, jumps) → **1**
   (fox, over) → **1**

2. Sample **false context** words from their unigram distribution and predict the probability of them co-occurring with **true target** word to be **0**

(fox, quick) → **1**	(fox, the) → **0**
(fox, brown) → **1**	(fox, lazy) → **0**
(fox, jumps) → **1**	(fox, dog) → **0**
(fox, over) → **1**	(fox, the) → **0**

Negative Sampling Loss (NCE)

$$\log \sigma(\boxed{T_i} \cdot \boxed{C_j}) + \sum^{n} \mathop{\mathbb{E}}_{k \sim P(w)} \log \sigma(-\boxed{T_k} \cdot \boxed{C_j})$$

target

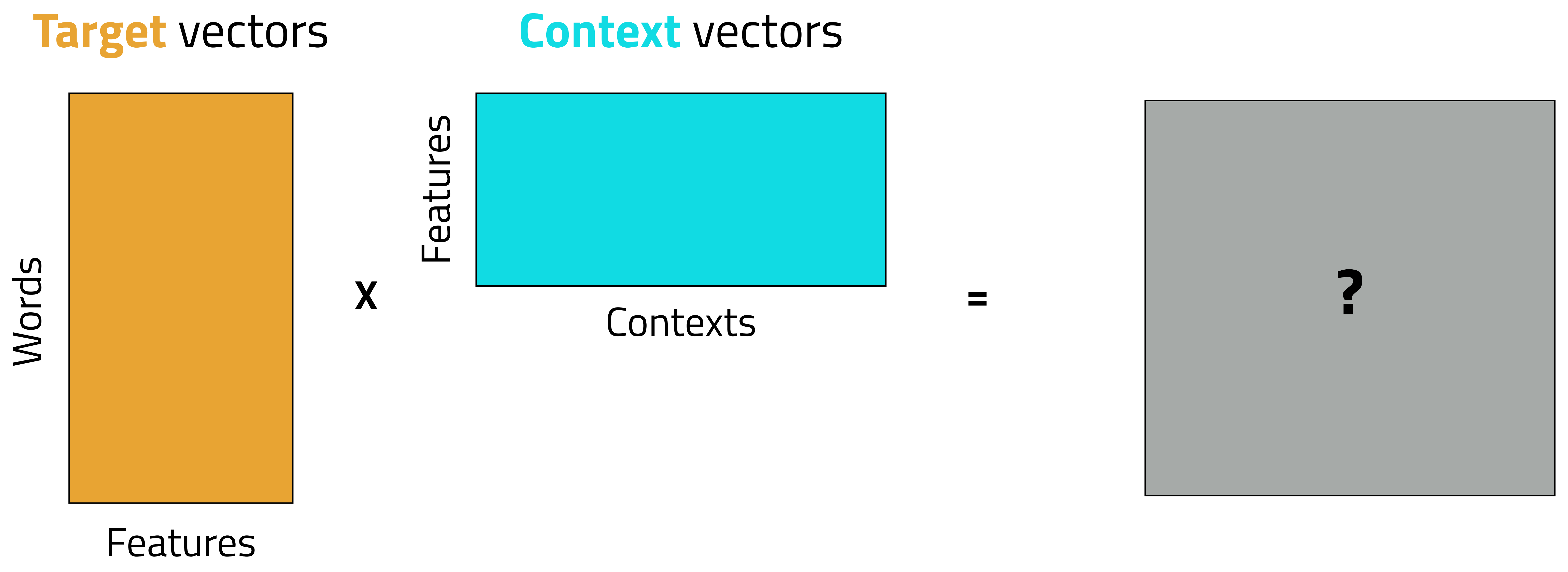
context

Number of negative samples

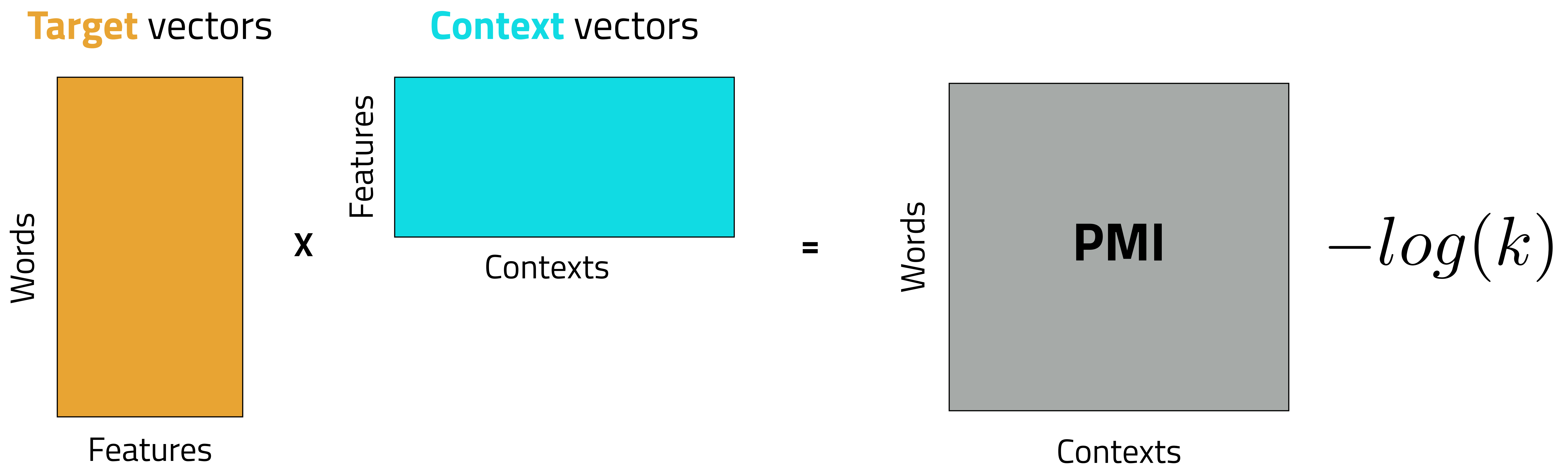
Sample from the distribution of words

Vector of the negative sample

SGNS as matrix factorization



SGNS as matrix factorization



Neural Word Embedding as Implicit Matrix Factorization (Levy and Goldberg 2014)

word2vec

Advantages

- Iterative way for factorizing a matrix
- Fast *O(nm)*, great implementations
- Several parameters to improve performance (negative samples, subsampling of frequent words, ...)
- Default parameters can go a long way

Disadvantages

- Inflexible definition of context
- Doesn't use dataset statistics in a smart way
- Columns are hard to interpret as topics

Are neural word embeddings better than classic DSMs?

Yes

With vanilla parameters

Baroni et al., *Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors*, 2014

No

With optimal parameters

Levy et al., *Improving Distributional Similarity with Lessons Learned from Word Embeddings*, 2015

Maybe

Trained on 1 billion+ words

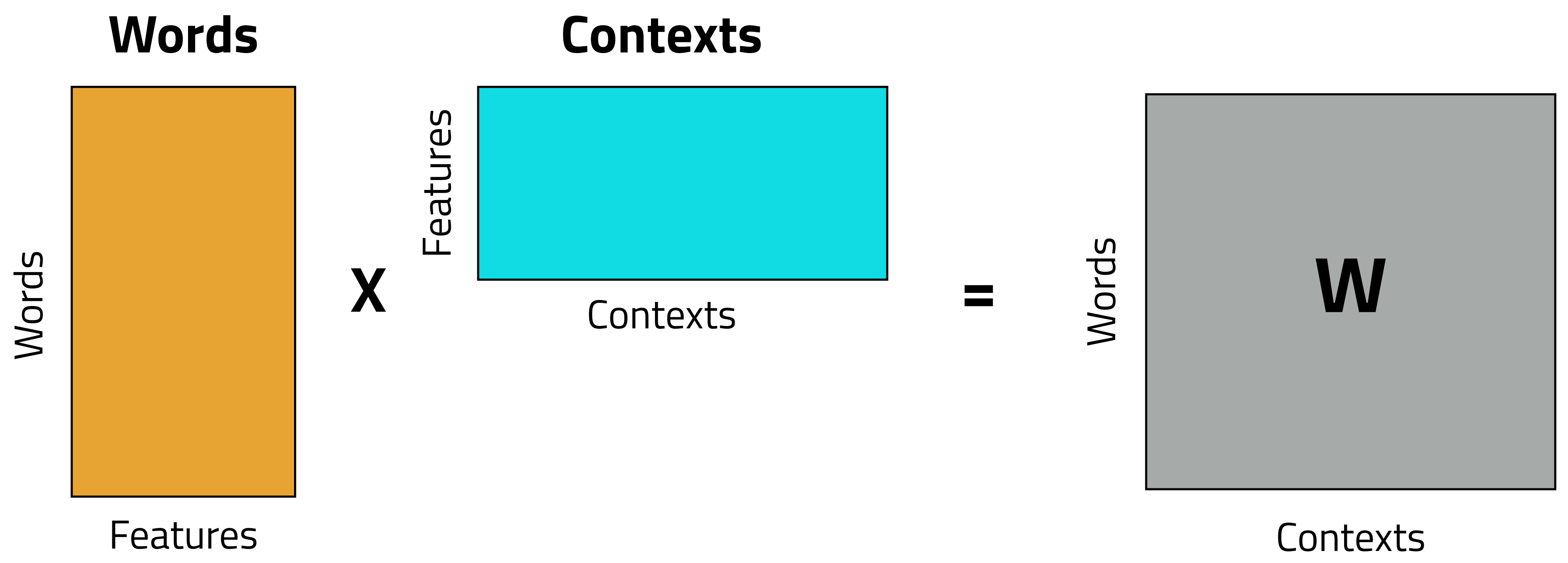
Sahlgren and Lenci, *The Effects of Data Size and Frequency Range on Distributional Semantic Models* , 2016

GloVe

Explicit factorization of **target** x **contexts** matrix

Precomputes the matrix (unlike SGNS)

Uses **directly** the **statistics** of the dataset (frequencies of co-occurrences)



$$J = \sum_{i,j} f(W_{ij})(w_i^\top \tilde{w}_j - \log W_{ij})^2$$

frequency of word *i* in context *j*

target

context

like SGNS

GloVe

Advantages

- Better use of dataset statistics
- Converges to good solutions with less data
- Simple to apply on different contexts

Disadvantages

- Recent comparisons show that on many tasks it doesn't perform as well as LSA or SGNS

Self-Supervised Learning

Compositionality

So far we represented words as vectors, how to represent **sentences**?

Can't use the co-occurrences of sentences in their context as **sentences** are **sparse**, most of them occur once

Should represent their meaning **combining** word representations

*The **meaning of an utterance** is a **function** of the **meaning of its parts** and their **composition rules*** - Gottlob Frege, *Über Sinn und Bedeutung*, 1892

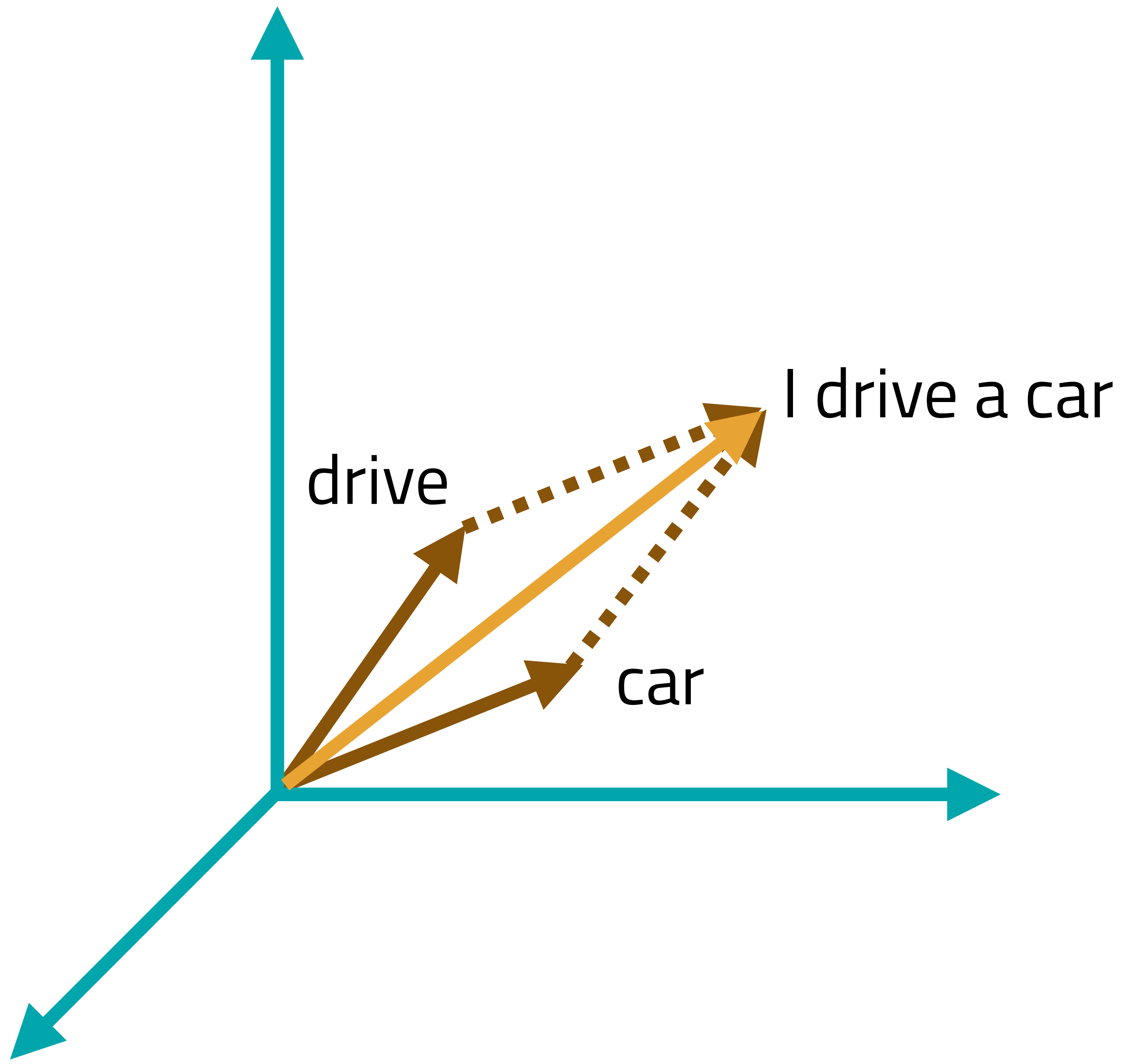
Composition operators

Simple solution, just **sum** the vectors of the words in a sentence!

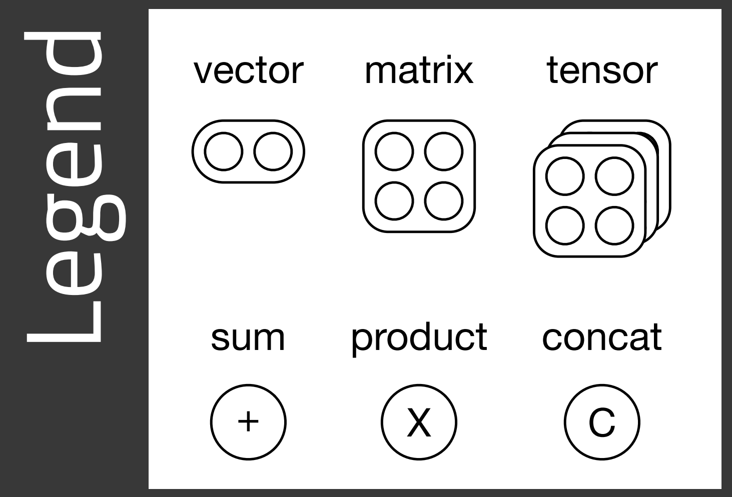
Other **operators**: product, weighted sum, convolution, ... (Mitchell and Lapata, 2008)

It's hard to perform better than the simple **sum**

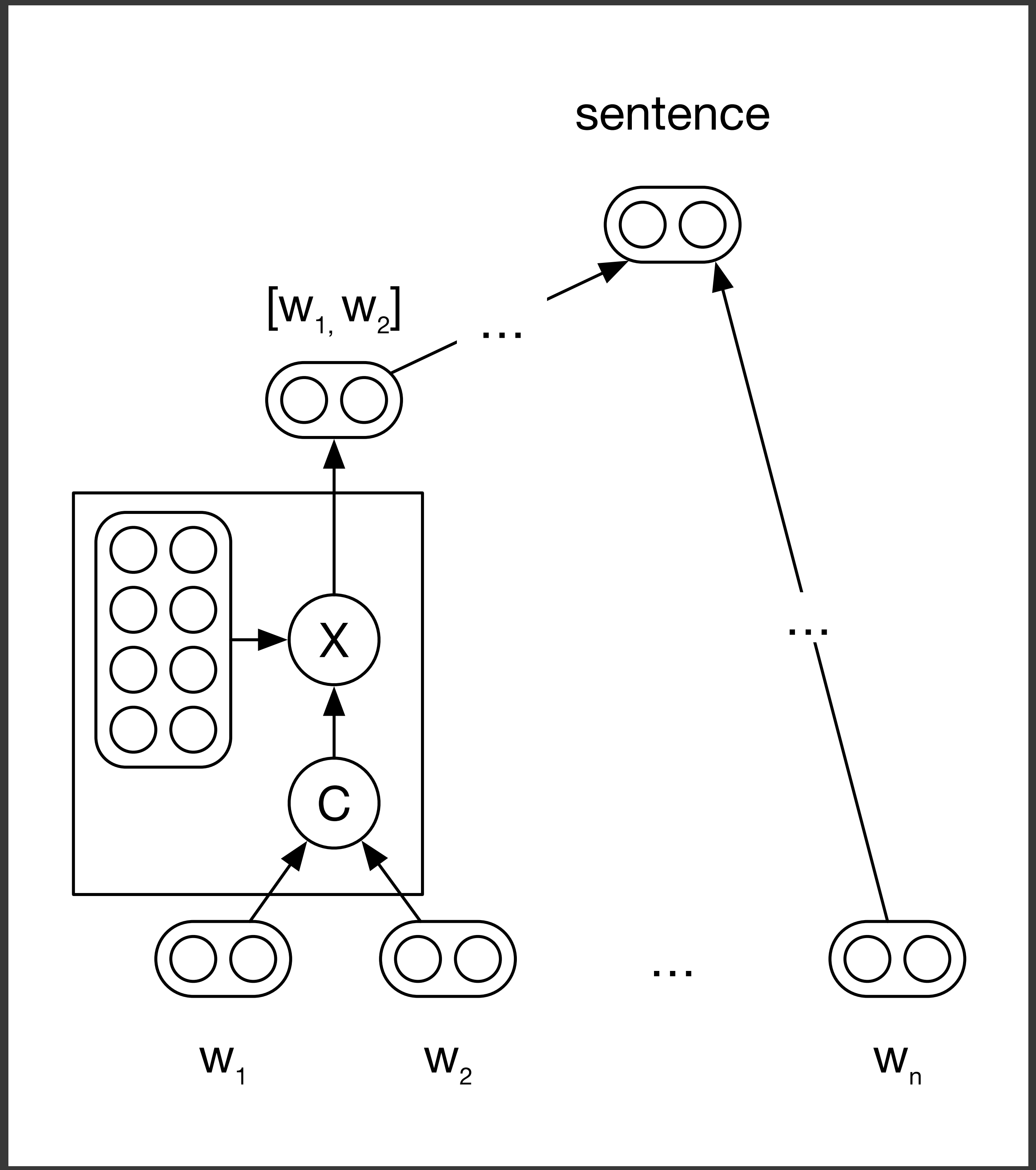
Sum can't be the real answer as it's **commutative** → doesn't consider **word order**



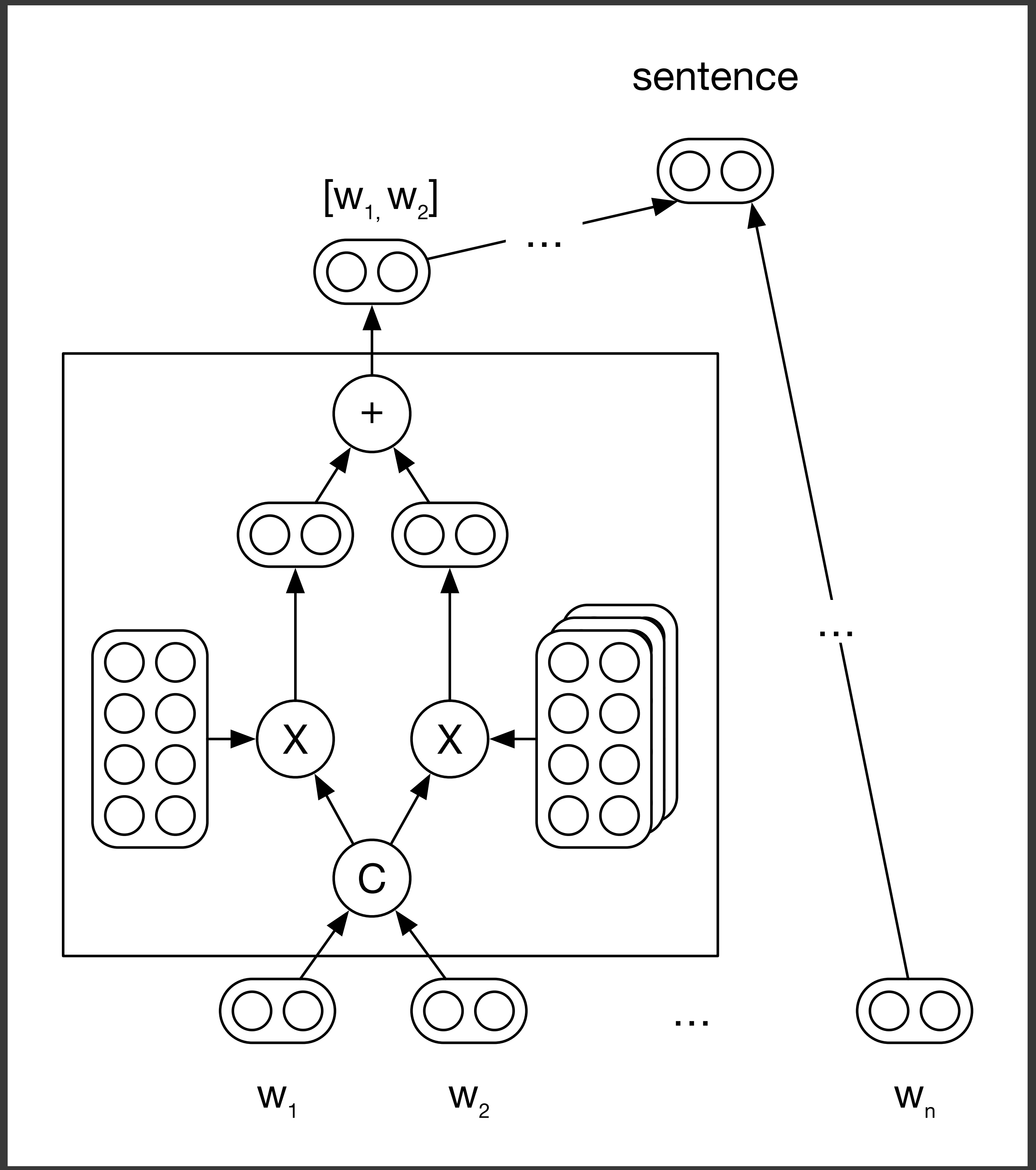
Learn to compose



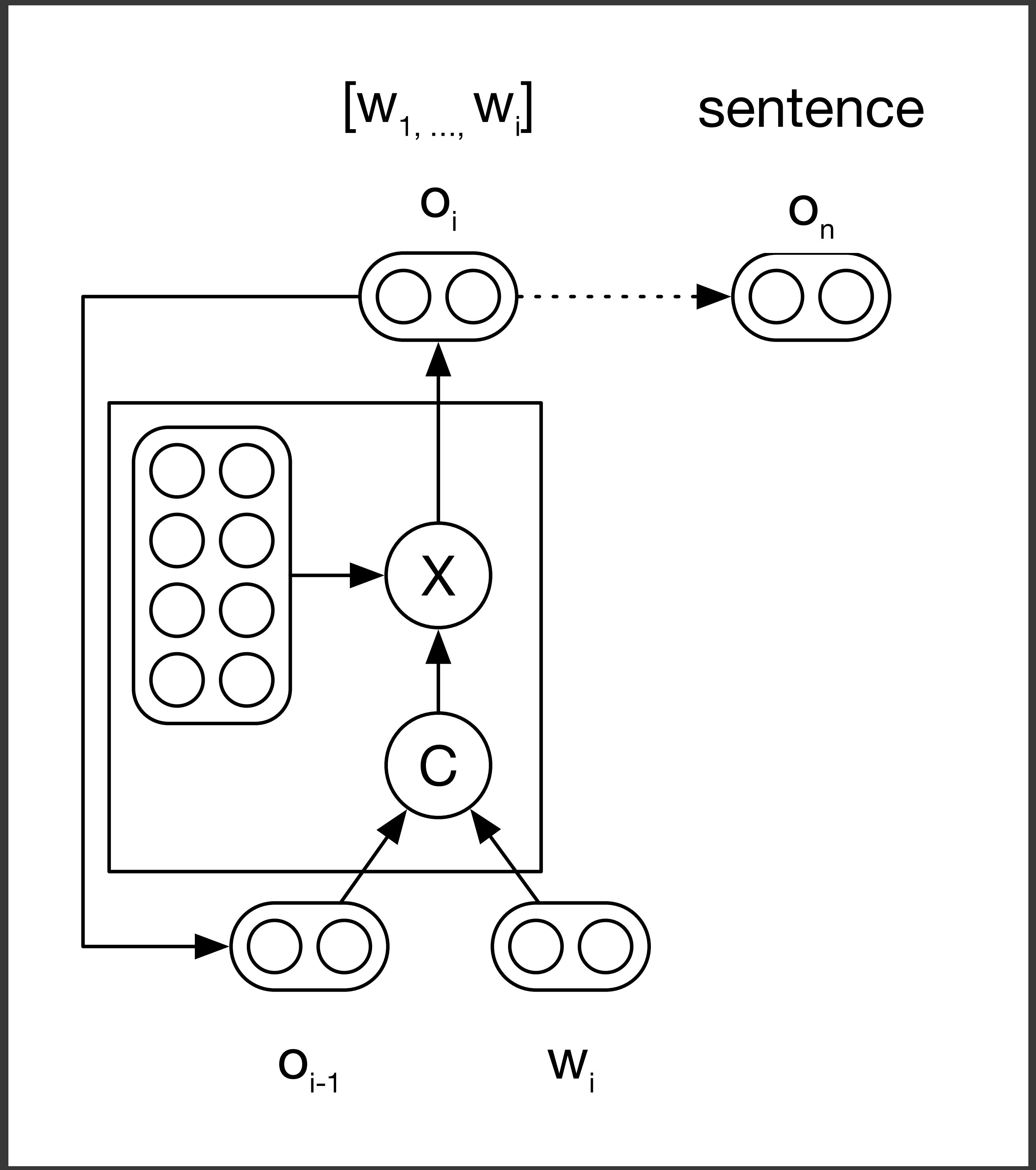
Recursive Matrix Vector Network (Socher at al. 2012)



Recursive Neural Tensor Network (Socher et al. 2013)



Recurrent Neural Network (Elman 1990) and others



Language Modeling

The quick brown fox jumps over the lazy dog

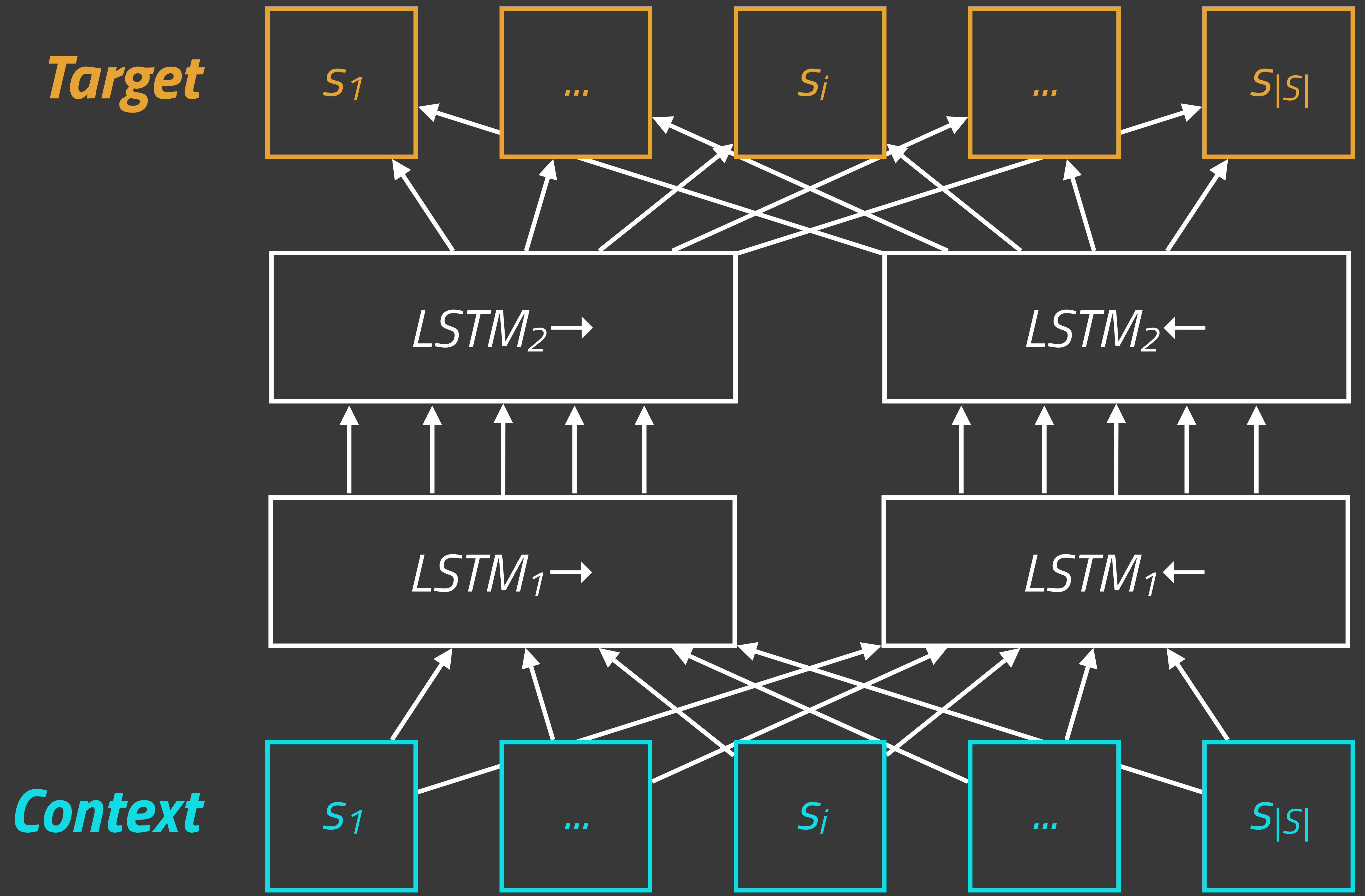
$$P(s) = \prod_{i}^{|s|} P(s_i | s_1, \ldots, s_{i-1})$$

Bidirectional Language Modeling

The quick brown fox jumps over the lazy dog

$$P(s) = \prod_{i}^{|s|} P(s_i | s_1, \ldots, s_{i-1}) P(s_i | s_{i+1}, \ldots, s_{|S|})$$

Bidirectional Language Modeling (ELMo)



Deep Contextualized Word Representations (Clark et al. 2017)

Masked Language Modeling (BERT)

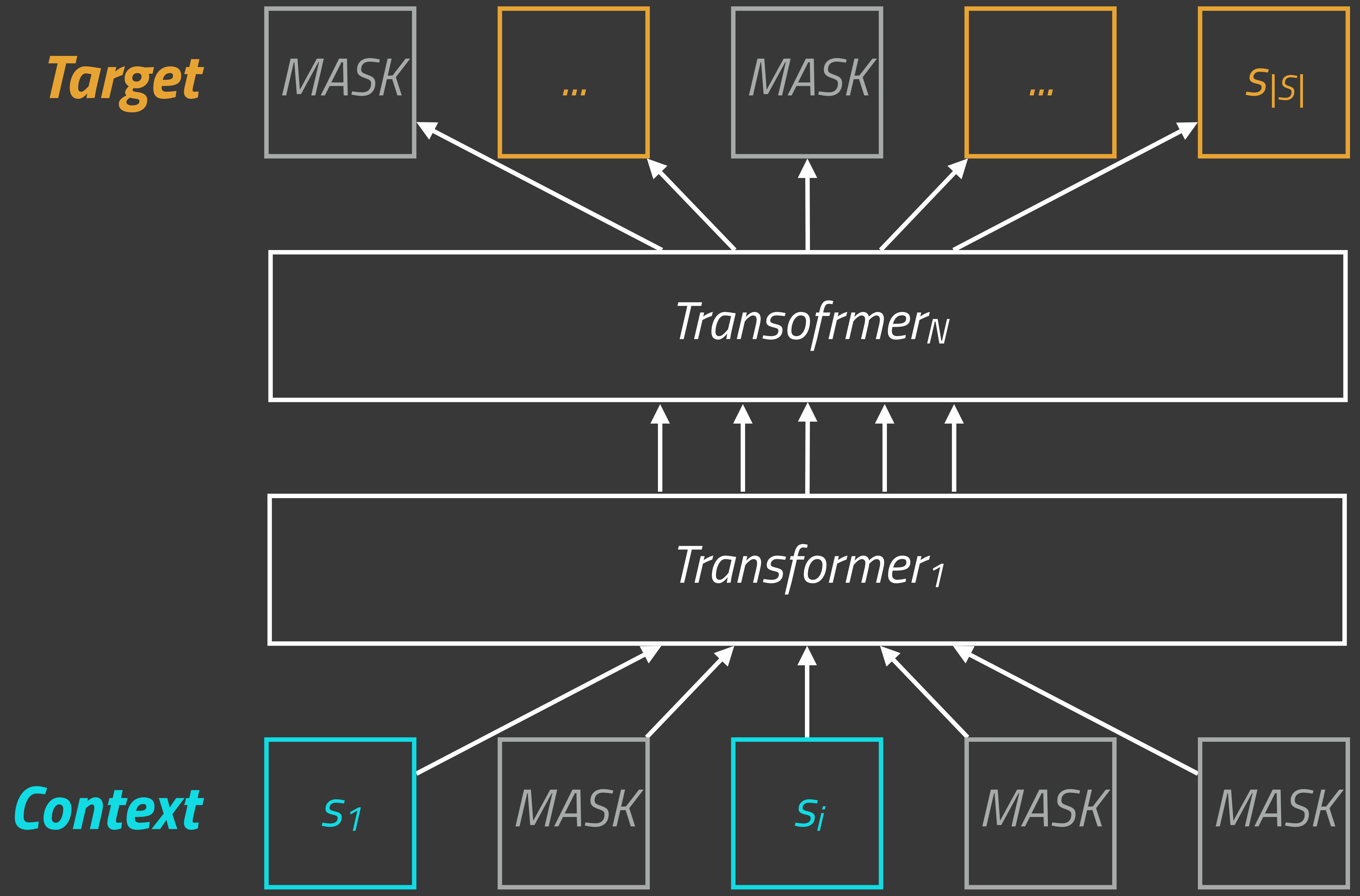
Sample 1: *The quick brown fox jumps over the lazy dog*

Sample 2: *The quick brown fox jumps over the lazy dog*

Sample 3: *The quick brown fox jumps over the lazy dog*

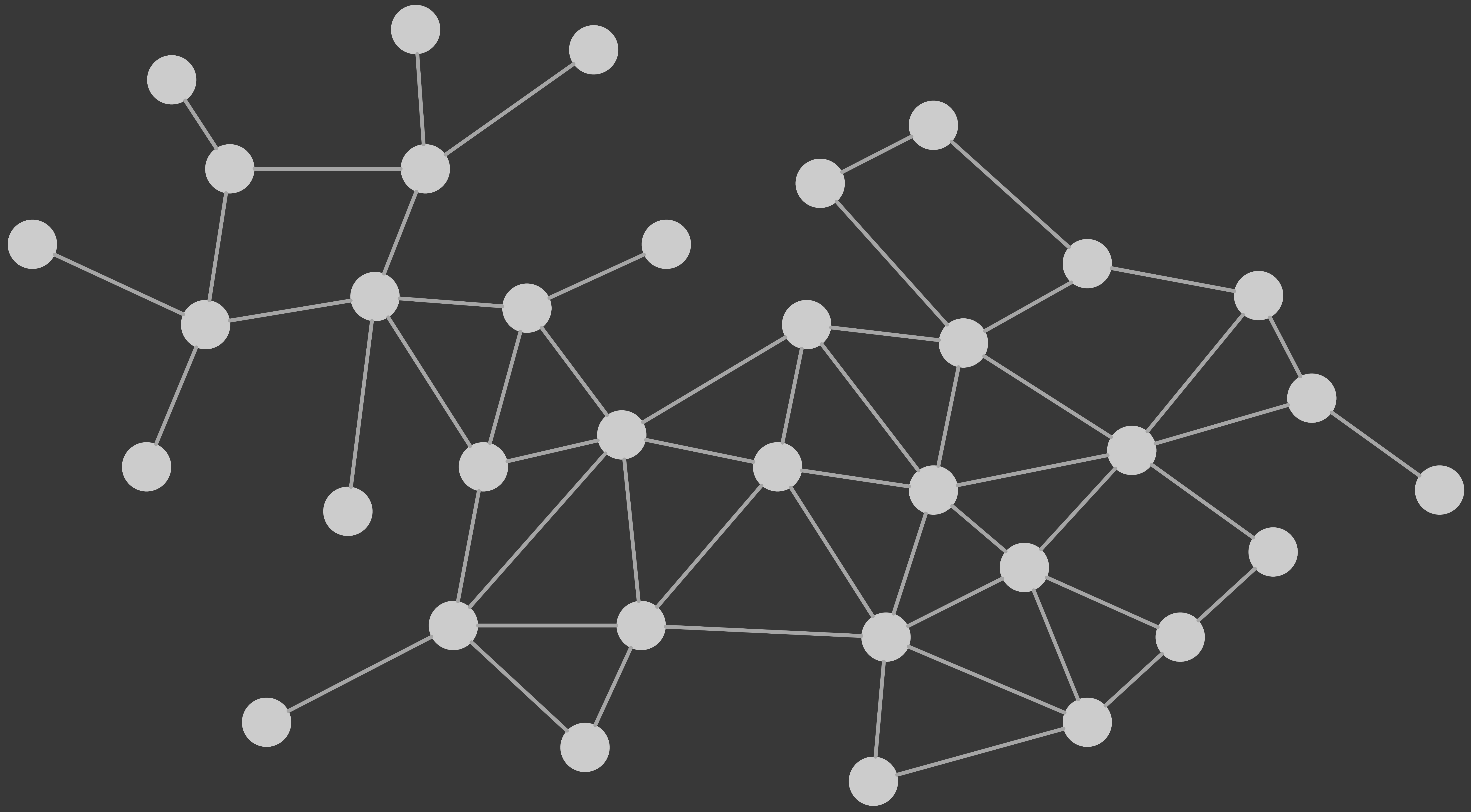
Sample 4: *The quick brown fox jumps over the lazy dog*

Masked Language Modeling (BERT)

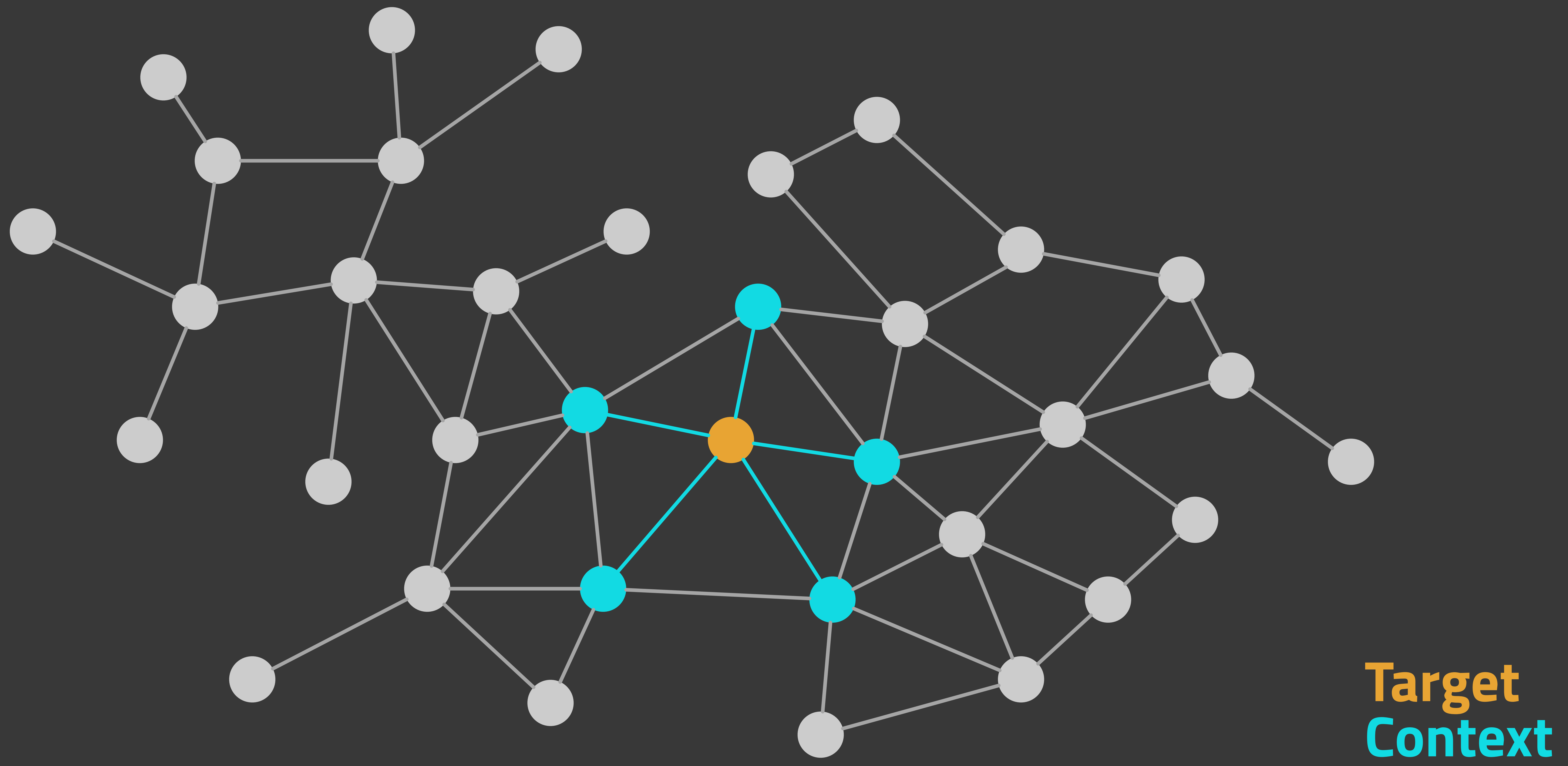


BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Delvin at al. 2017)

Embeddings for Graphs



Embeddings for Graphs



Graph Representation Learning

DeepWalk (Perozzi et al. 2014)

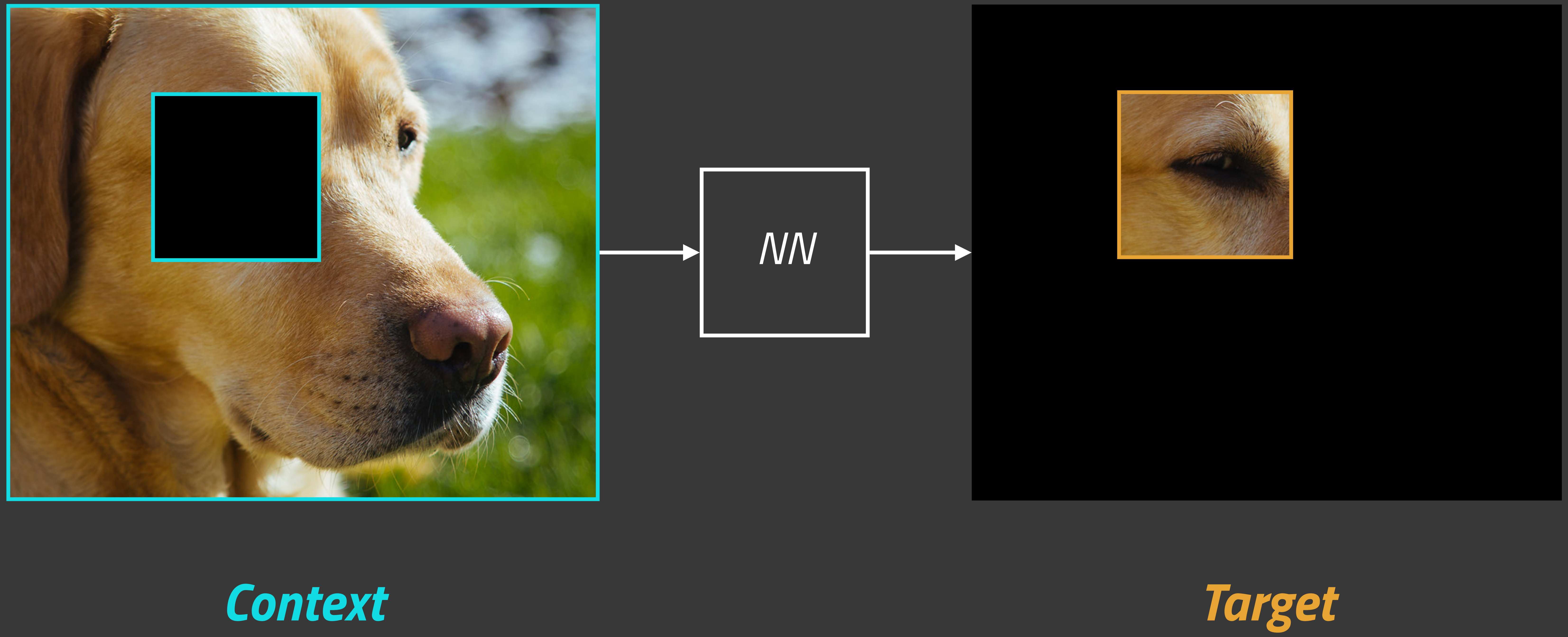
node2vec (Grover and Leskovec 2016)

GraphSage (Hamilton at el. 2017) for supervised tasks, but same principle

Many papers of *Knowledge Graphs* (Nickel et al. 2015 for a review)

Computer Vision

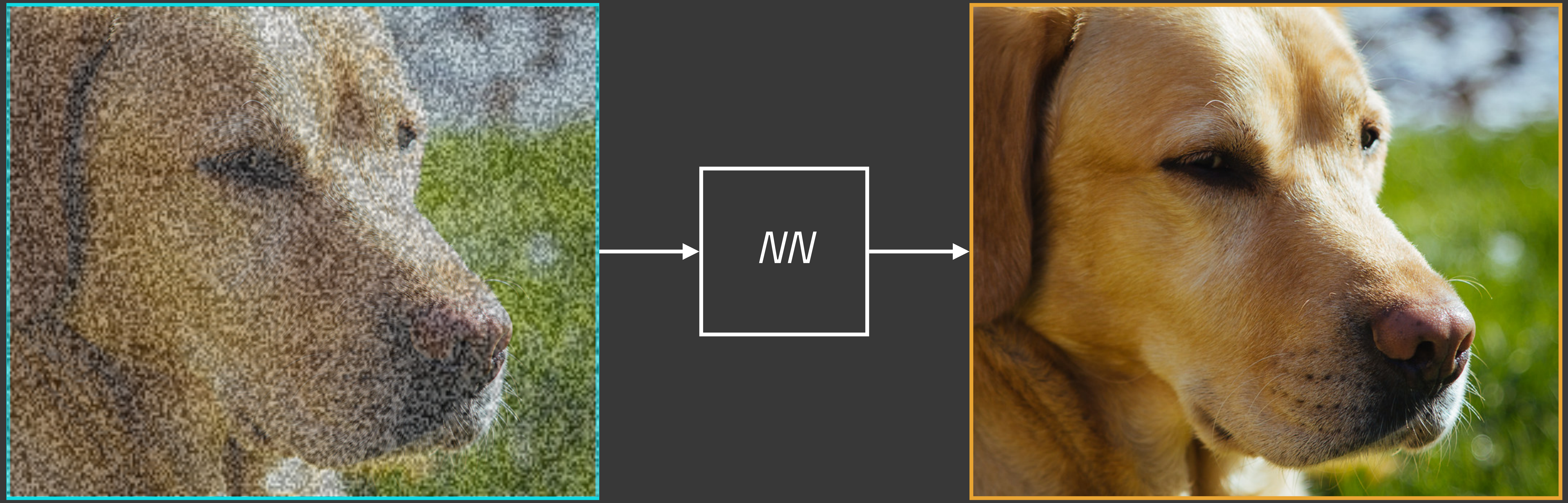
Inpainting



Context Encoders: Feature Learning by Inpainting (Pathak at al. 2016)

Computer Vision

Denoising Autoencoders

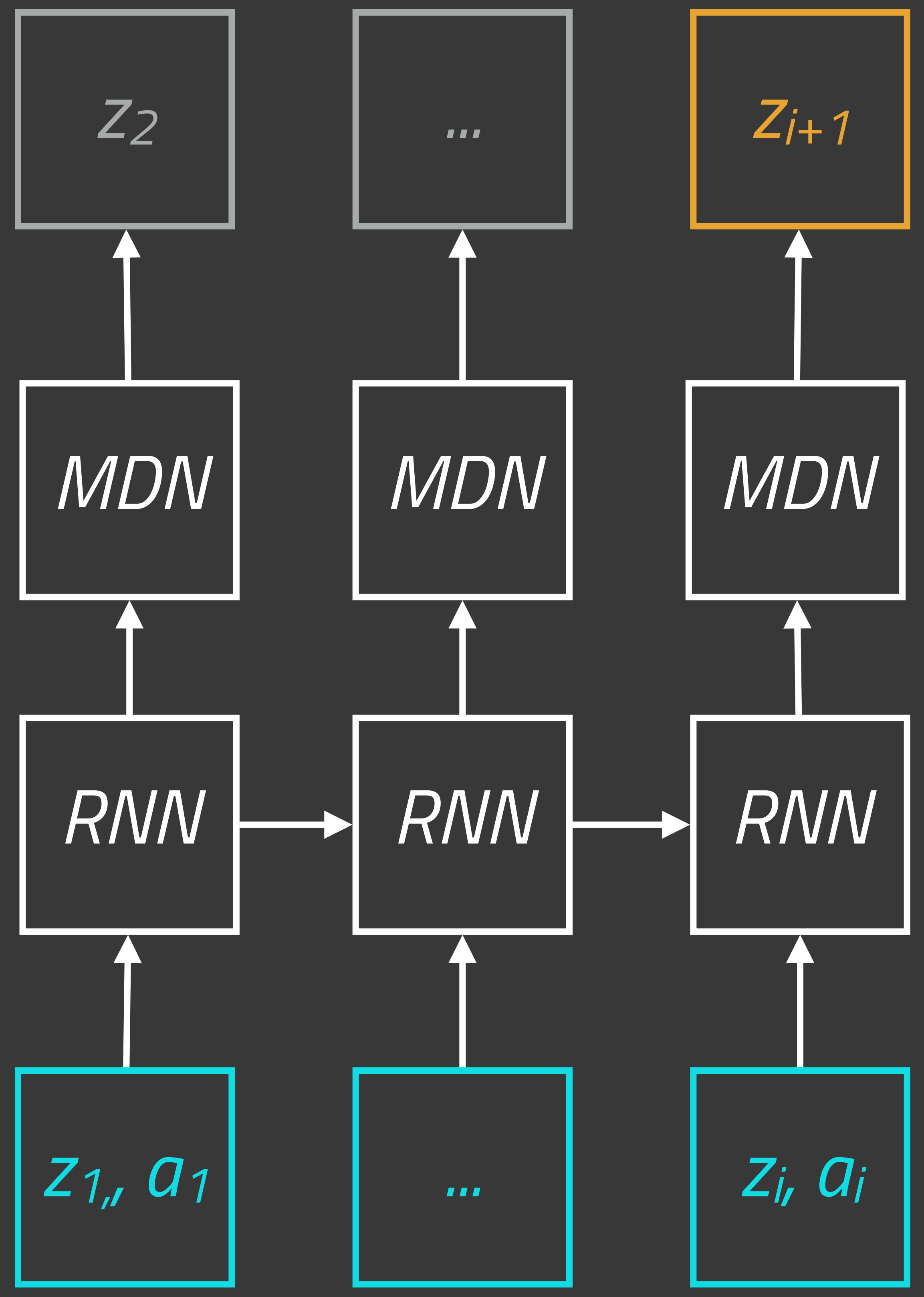


Context ***Target***

Noise is a kind of *'soft mask'*

Extracting and Composing Robust Features with Denoising Autoencoders (Vincent at al. 2008)

Reinforcement Learning



Target: how the world will look like in the futere if I take action a_i

Context: how the world looks like in the past unitl now

World Models (Ha and Schmidhuber 2008)

Making the World Differentiable: On Using Self-Supervised Fully Recurrent Neural Networks for Dynamic Reinforcement Learning and Planning in Non-Stationary Environments (Schmidhuber 1990)

Video

T*racking moving objects* (Wang and Gupta 2015)

Frame order validation (Misra et al. 2016)

Video colorization (Vodrick at el. 2018)

Conclusions

Structuralism can be a **unifying interpretation** for **why** many learning algorithms work

Many aspects of reality can be modeled in terms of **targets** and **contexts**

Context is king

Self-Supervised Learning is *'just'* **applied structuralism**

Thanks

Contacts
piero.molino@gmail.com
http://w4nderlu.st

Influenced this talk:

Magnus Sahlgren

Alfio Gliozzo

Marco Baroni

Alessandro Lenci

Yoav Goldberg

Andre Freitas

Pierpaolo Basile

Aurélie Herbelot

Arianna Betti