

Learning word embeddings via singular value decomposition

Marco Kuhlmann

Department of Computer and Information Science

Co-occurrence matrix

	cheese	bread	goat	sheep
cheese	14	7	5	1
bread	7	12	0	0
goat	5	0	8	12
sheep	1	0	12	2

word vector for *cheese*

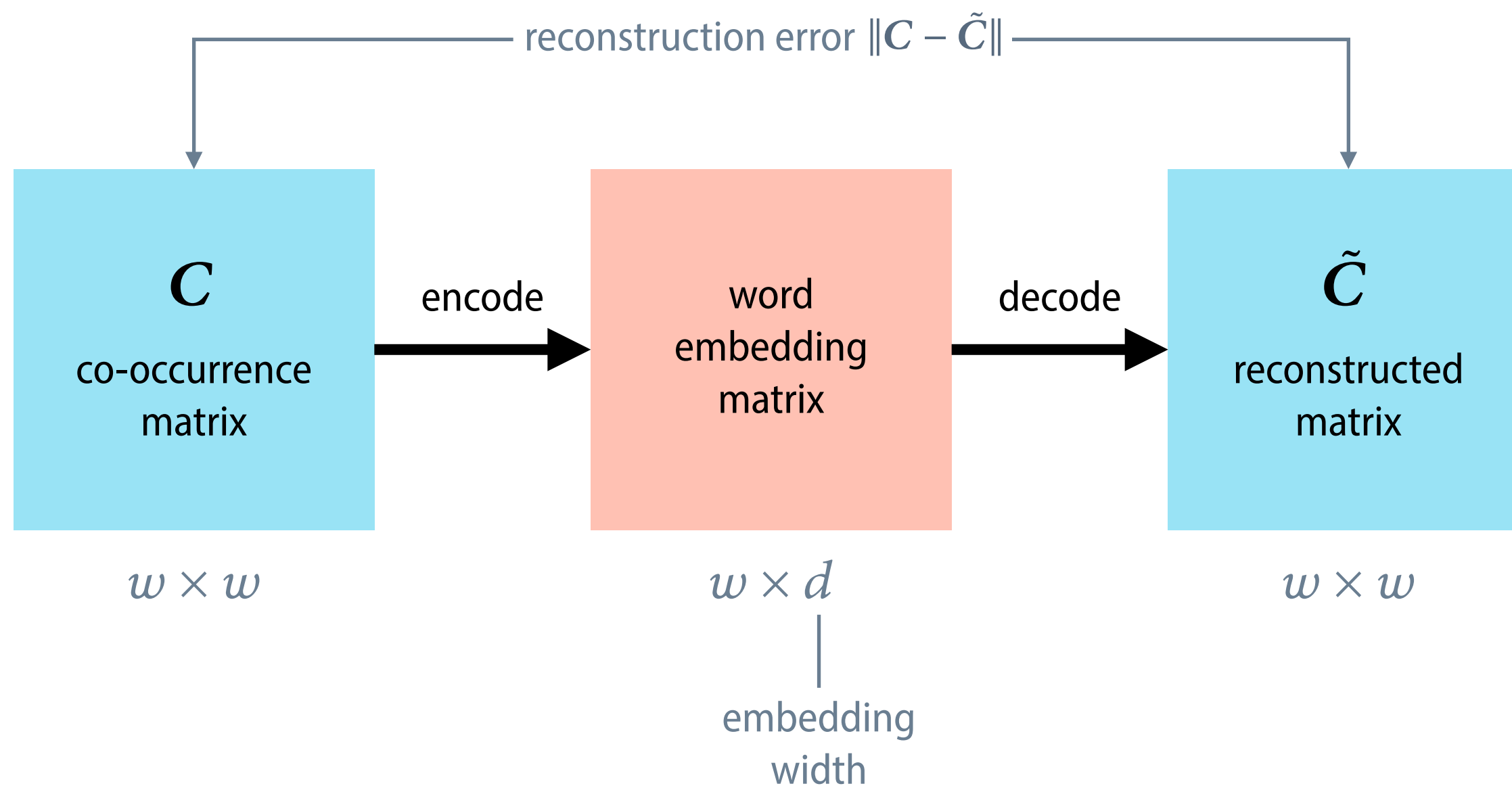
Word embeddings via dimensionality reduction

- The row vectors of co-occurrence matrices are high-dimensional, but we want word embeddings to be low-dimensional.

hundreds instead of hundreds of thousands of dimensions

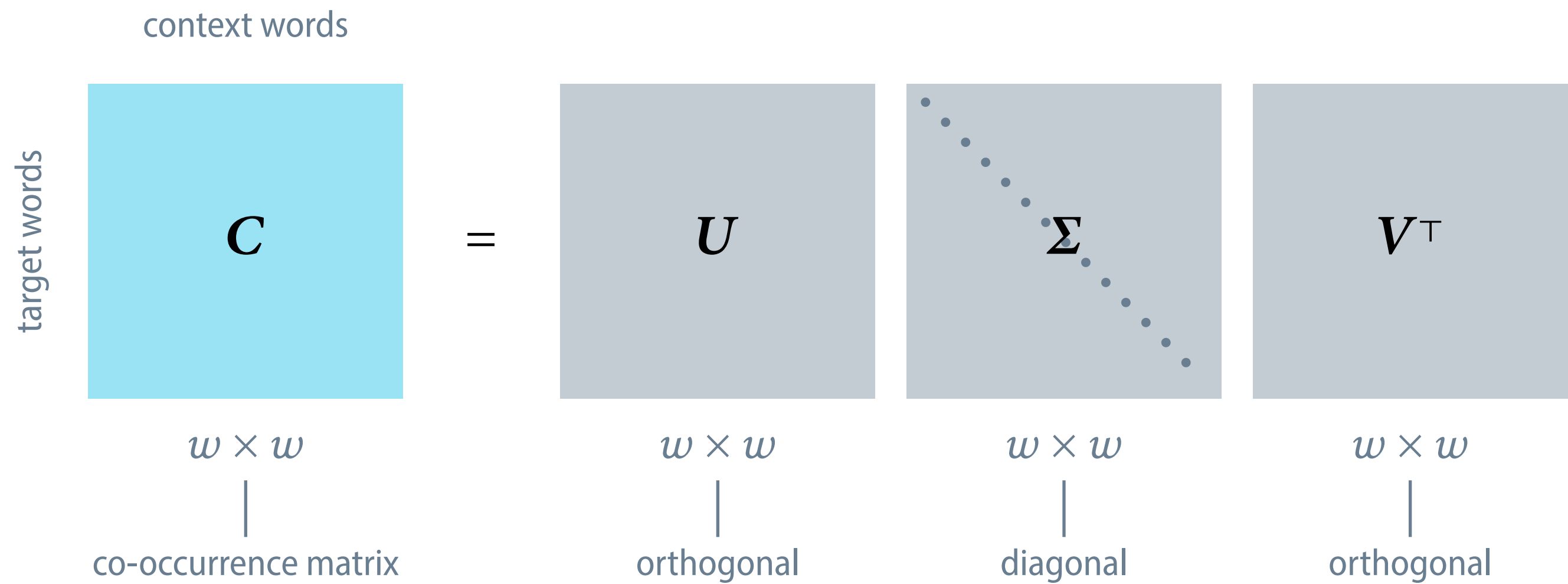
- One idea is to use dimensionality reduction and find a lower-dimensional representation of the co-occurrence matrix.

Word embeddings via dimensionality reduction



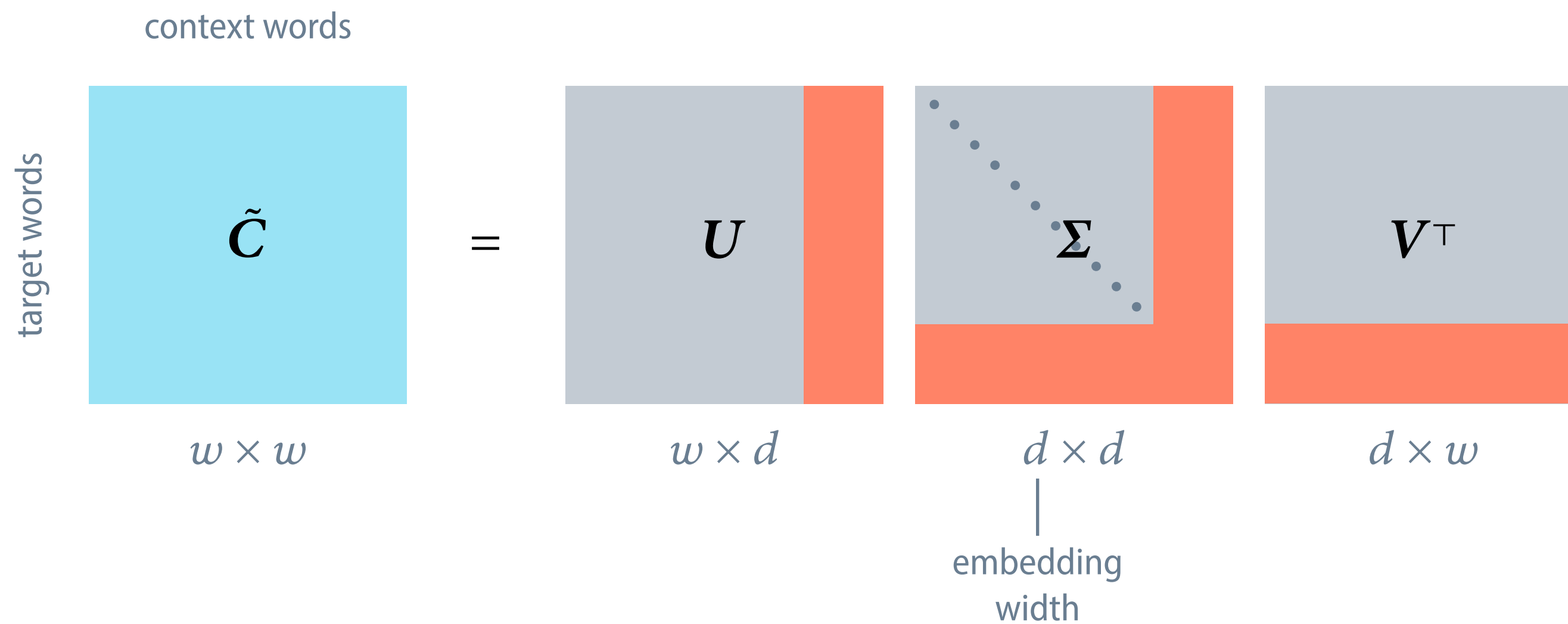
Singular value decomposition

Eisenstein § 14.3



Truncated singular value decomposition

Eisenstein § 14.3



Deerwester et al. (1990)

original co-occurrence matrix

14	7	5	1
7	12	0	0
5	0	8	12
1	0	12	2

U

-0.678	0.293	0.124	0.662
-0.445	0.562	-0.044	-0.696
-0.491	-0.591	-0.628	-0.124
-0.318	-0.499	0.767	-0.248

Σ

22.676	0.000	0.000	0.000
0.000	15.644	0.000	0.000
0.000	0.000	7.655	0.000
0.000	0.000	0.000	5.335

V^T

-0.678	-0.445	-0.491	-0.318
0.293	0.562	-0.591	-0.499
-0.124	0.044	0.628	-0.767
0.662	-0.696	-0.124	-0.248

original co-occurrence matrix

14	7	5	1
7	12	0	0
5	0	8	12
1	0	12	2

U

-0.678	0.293	0.124	0.662
-0.445	0.562	-0.044	-0.696
-0.491	-0.591	-0.628	-0.124
-0.318	-0.499	0.767	-0.248

Σ

22.676	0.000	0.000	0.000
0.000	15.644	0.000	0.000
0.000	0.000	7.655	0.000
0.000	0.000	0.000	5.335

V^T

-0.678	-0.445	-0.491	-0.318
0.293	0.562	-0.591	-0.499
-0.124	0.044	0.628	-0.767
0.662	-0.696	-0.124	-0.248

reconstruction $d = 3$

11.659	9.459	5.439	1.878
9.459	9.418	-0.461	-0.922
5.439	-0.461	7.918	11.835
1.878	-0.922	11.835	1.671

reconstruction error: 1.779

original co-occurrence matrix

14	7	5	1
7	12	0	0
5	0	8	12
1	0	12	2

U

-0.678	0.293	0.124	0.662
-0.445	0.562	-0.044	-0.696
-0.491	-0.591	-0.628	-0.124
-0.318	-0.499	0.767	-0.248

Σ

22.676	0.000	0.000	0.000
0.000	15.644	0.000	0.000
0.000	0.000	7.655	0.000
0.000	0.000	0.000	5.335

V^T

-0.678	-0.445	-0.491	-0.318
0.293	0.562	-0.591	-0.499
-0.124	0.044	0.628	-0.767
0.662	-0.696	-0.124	-0.248

reconstruction $d = 3$

11.659	9.459	5.439	1.878
9.459	9.418	-0.461	-0.922
5.439	-0.461	7.918	11.835
1.878	-0.922	11.835	1.671

reconstruction error: 1.779

reconstruction $d = 2$

11.776	9.417	4.845	2.605
9.417	9.432	-0.249	-1.181
4.845	-0.249	10.934	8.148
2.605	-1.181	8.148	6.178

reconstruction error: 5.442

original co-occurrence matrix

14	7	5	1
7	12	0	0
5	0	8	12
1	0	12	2

U

-0.678	0.293	0.124	0.662
-0.445	0.562	-0.044	-0.696
-0.491	-0.591	-0.628	-0.124
-0.318	-0.499	0.767	-0.248

Σ

22.676	0.000	0.000	0.000
0.000	15.644	0.000	0.000
0.000	0.000	7.655	0.000
0.000	0.000	0.000	5.335

V^T

-0.678	-0.445	-0.491	-0.318
0.293	0.562	-0.591	-0.499
-0.124	0.044	0.628	-0.767
0.662	-0.696	-0.124	-0.248

reconstruction d = 3

11.659	9.459	5.439	1.878
9.459	9.418	-0.461	-0.922
5.439	-0.461	7.918	11.835
1.878	-0.922	11.835	1.671

reconstruction error: 1.779

reconstruction d = 2

11.776	9.417	4.845	2.605
9.417	9.432	-0.249	-1.181
4.845	-0.249	10.934	8.148
2.605	-1.181	8.148	6.178

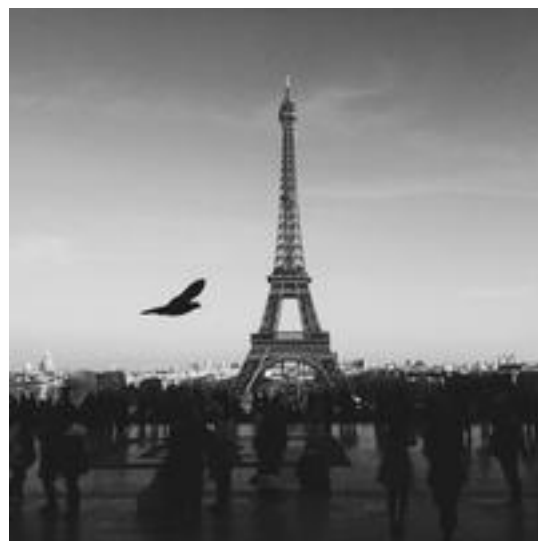
reconstruction error: 5.442

reconstruction d = 1

10.436	6.843	7.552	4.888
6.843	4.486	4.951	3.205
7.552	4.951	5.465	3.537
4.888	3.205	3.537	2.289

reconstruction error: 20.737

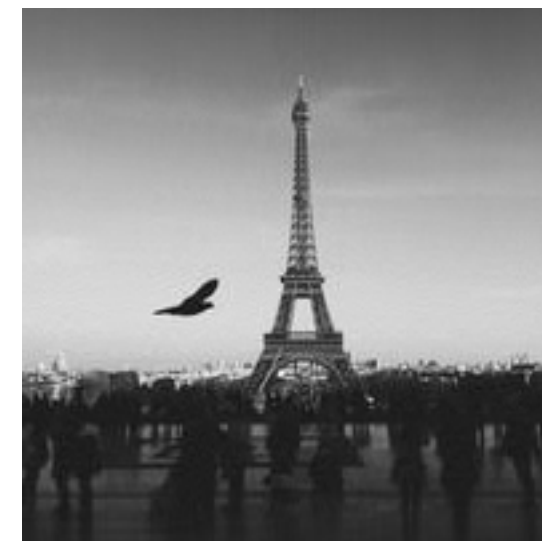
Truncated singular value decomposition



$d = 200$



$d = 100$



$d = 50$



$d = 20$



$d = 10$



$d = 5$

Reading off word embeddings

- We would like to quantify the similarity of word embeddings. The main operation involved in this is the dot product.
cosine similarity = dot product of length-normalised vectors
- Conveniently, the dot products between the row vectors of the reconstructed matrix are equal to the dot products between the row vectors of the lower-dimensional matrix $U\Sigma$.
- We could read off word embeddings from that matrix; but empirically it works better to dismiss Σ and only use U .

Limitations of the singular value decomposition

- Computing the singular value decomposition is expensive; as a rule of thumb, the running time is in $\Theta(w^2d)$.

In practice, randomized algorithms are used.

- The method is not incremental – the decomposition needs to be recomputed from scratch whenever new data arrives.

Pointwise mutual information

- Raw counts give too much weight to function words such as *the*, *she*, *has*, and too little weight to *cheese*, *bread*, *sheep*.
- We measure the associative strength between a target word w and a context c using **pointwise mutual information**:

$$\text{PMI}(w, c) = \log \frac{P(w, c)}{P(w)P(c)}$$

Computing PMI on a co-occurrence matrix

	cheese	bread	goat	sheep
cheese	14	7	5	1
bread	7	12	0	0
goat	5	0	8	12
sheep	1	0	12	2

Computing PMI on a co-occurrence matrix

$$P(\textit{cheese}) = \frac{27}{86}$$

	cheese	bread	goat	sheep
cheese	14	7	5	1
bread	7	12	0	0
goat	5	0	8	12
sheep	1	0	12	2

27

Σ 86

Computing PMI on a co-occurrence matrix

$$P(\textit{bread}) = \frac{19}{86}$$

	cheese	bread	goat	sheep
cheese	14	7	5	1
bread	7	12	0	0
goat	5	0	8	12
sheep	1	0	12	2

19

Σ 86

Computing PMI on a co-occurrence matrix

$$P(\textit{cheese}, \textit{bread}) = \frac{14}{86}$$

	cheese	bread	goat	sheep
cheese	14	7	5	1
bread	7	12	0	0
goat	5	0	8	12
sheep	1	0	12	2

Computing PMI on a co-occurrence matrix

	cheese	bread	goat	sheep	
cheese	$\frac{14/86}{27/86 \cdot 27/86}$	$\frac{7/86}{27/86 \cdot 19/86}$	$\frac{5/86}{27/86 \cdot 25/86}$	$\frac{1/86}{27/86 \cdot 15/86}$	27
bread	$\frac{7/86}{19/86 \cdot 27/86}$	$\frac{12/86}{19/86 \cdot 19/86}$	$\frac{0/86}{19/86 \cdot 25/86}$	$\frac{0/86}{19/86 \cdot 15/86}$	19
goat	$\frac{5/86}{25/86 \cdot 27/86}$	$\frac{0/86}{25/86 \cdot 19/86}$	$\frac{8/86}{25/86 \cdot 25/86}$	$\frac{12/86}{25/86 \cdot 15/86}$	25
sheep	$\frac{1/86}{15/86 \cdot 27/86}$	$\frac{0/86}{15/86 \cdot 19/86}$	$\frac{12/86}{15/86 \cdot 25/86}$	$\frac{2/86}{15/86 \cdot 15/86}$	15
	27	19	25	15	$\Sigma 86$

Computing PMI on a co-occurrence matrix

	cheese	bread	goat	sheep
cheese	log 1.65	log 1.17	log 0.64	log 0.21
bread	log 1.17	log 2.86	log 0.00	log 0.00
goat	log 0.64	log 0.00	log 1.10	log 2.75
sheep	log 0.21	log 0.00	log 2.75	log 0.76

Positive pointwise mutual information (PPMI)

- Because the rows of co-occurrence matrices are sparse, many PMI values will be $\log 0 = -\infty$.
- A common approach is to use **positive pointwise mutual information (PPMI)**, and replace all negative values with zero:

$$\text{PPMI}(w, c) = \max(\text{PMI}(w, c), 0)$$

- A shortcoming of this approach is its bias towards infrequent events; for these, the PPMI value will be very high.