Non-Standard Datenbanken und Data Mining

Deep Learning
Embedding Representations

Prof. Dr. Ralf Möller

Universität zu Lübeck

Institut für Informationssysteme

Übersicht

- Einführung, Klassifikation vs. Regression, parametrisches und nicht-parametrisches überwachtes Lernen
- Netze aus differenzierbaren Modulen („neuronale“ Netze), Support-Vektor-Maschinen
- Häufungsanalysen, Warenkorbanalyse, Empfehlungen
- Statistische Grundlagen: Stichproben, Schätzer, Verteilung, Dichte, kumulative Verteilung, Skalen: Nominal-, Ordinal-, Intervall- und Verhältnisskala, Hypothesentests, Konfidenzintervalle, Reliabilität, Interne Konsistenz, Cronbach Alpha, Trennschärfe
- Bayessche Statistik, Bayessche Netze zur Spezifikation von diskreten Verteilungen, Anfragen, Anfragebeantwortung, Lernverfahren für Bayessche Netze bei vollständigen Daten
- Induktives Lernen: Versionsraum, Informationstheorie, Entscheidungsbäume, Lernen von Regeln
- Ensemble-Methoden, Bagging, Boosting, Random Forests
- Clusterbildung, K-Means, Analyse der Variation (Analysis of Variation, ANOVA), Inter-Cluster-Variation, Intra-Cluster-Variation, F-Statistik, Bonferroni-Korrektur, MANOVA
- Analyse Sozialer Strukturen
- Deep Learning, Einbettungstechniken
- Zusammenfassung

Word-Word Associations in Document Retrieval

Recap bag-of-words approaches

- Client profiles, TF-IDF

Words are not independent of each other

Need to represent some aspects of word semantics

Church, K.W., Hanks, P.: Word association norms mutual information, and lexicography. Comput. Linguist. 1(1), 22–29, **1990**

Point(wise) Mutual Information: PMI

- Measure of association used in information theory and statistics

$$\operatorname{pmi}(x;y) \equiv \log \frac{p(x,y)}{p(x)p(y)} = \log \frac{p(x|y)}{p(x)} = \log \frac{p(y|x)}{p(y)}$$

- Positive PMI: PPMI(x, y) = max(pmi(x, y), 0)
- Quantifies the discrepancy between the probability of their coincidence given their joint distribution and their individual distributions, assuming independence
- Finding collocations and associations between words
- Countings of occurrences and co-occurrences of words in a text corpus can be used to approximate the probabilities p(x) or p(y) and p(x,y) respectively

PMI – Example

word 1	*word 2*	*count word 1*	*count word 2*	count of co-occurrences	PMI
puerto	rico	1938	1311	1159	10.0349081703
hong	kong	2438	2694	2205	9.72831972408
los	angeles	3501	2808	2791	9.56067615065
carbon	dioxide	4265	1353	1032	9.09852946116
prize	laureate	5131	1676	1210	8.85870710982
san	francisco	5237	2477	1779	8.83305176711
nobel	prize	4098	5131	2498	8.68948811416
ice	hockey	5607	3002	1933	8.6555759741
star	trek	8264	1594	1489	8.63974676575
car	driver	5578	2749	1384	8.41470768304
it	the	283891	3293296	3347	-1.72037278119
are	of	234458	1761436	1019	-2.09254205335
this	the	199882	3293296	1211	-2.38612756961
is	of	565679	1761436	1562	-2.54614706831
and	of	1375396	1761436	2949	-2.79911817902
a	and	984442	1375396	1457	-2.92239510038
in	and	1187652	1375396	1537	-3.05660070757
to	and	1025659	1375396	1286	-3.08825363041
to	in	1025659	1187652	1066	-3.12911348956
of	and	1761436	1375396	1190	-3.70663100173

- Counts of pairs of words getting the most and the least PMI scores in the first 50 millions of words in Wikipedia (dump of October 2015)
- Filtering by 1,000 or more co-occurrences.
- The frequency of each count can be obtained by dividing its value by 50,000,952. (Note: natural log is used to calculate the PMI values in this example, instead of log base 2)

[Wikipedia]

PMI – Co-occurrence Matrix

Add-2 Smoothed Count(w,context)

	computer	data	pinch	result	sugar
apricot	2	2	3	2	3
pineapple	2	2	3	2	3
digital	4	3	2	3	2
information	3	8	2	6	2

PPMI(w,context)

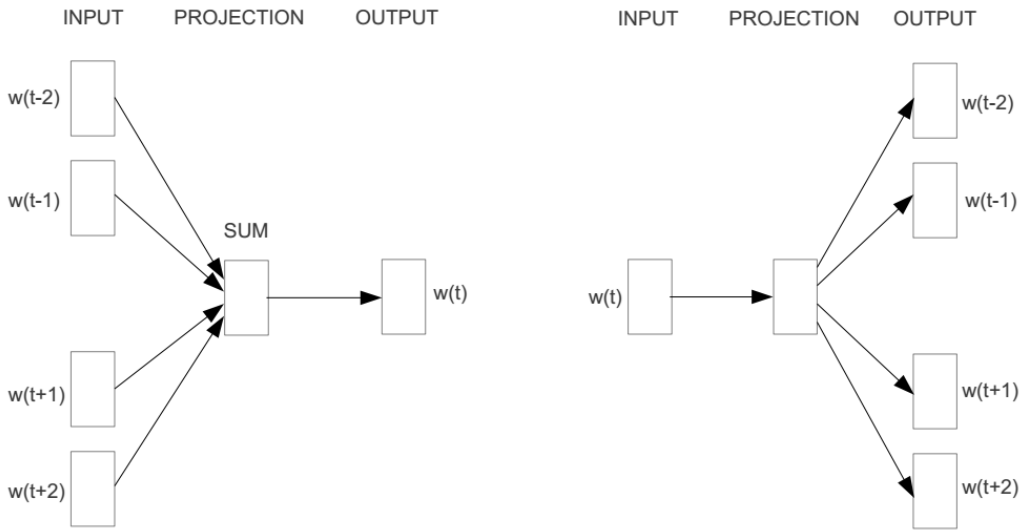
	computer	data	pinch	result	sugar
apricot	-	-	2.25	-	2.25
pineapple	-	-	2.25	-	2.25
digital	1.66	0.00	-	0.00	-
information	0.00	0.57	-	0.47	-

Embedding Approaches to Word Semantics

- Represent each word with a low-dimensional vector
- Word similarity = vector similarity
- Key idea: Predict surrounding words of every word

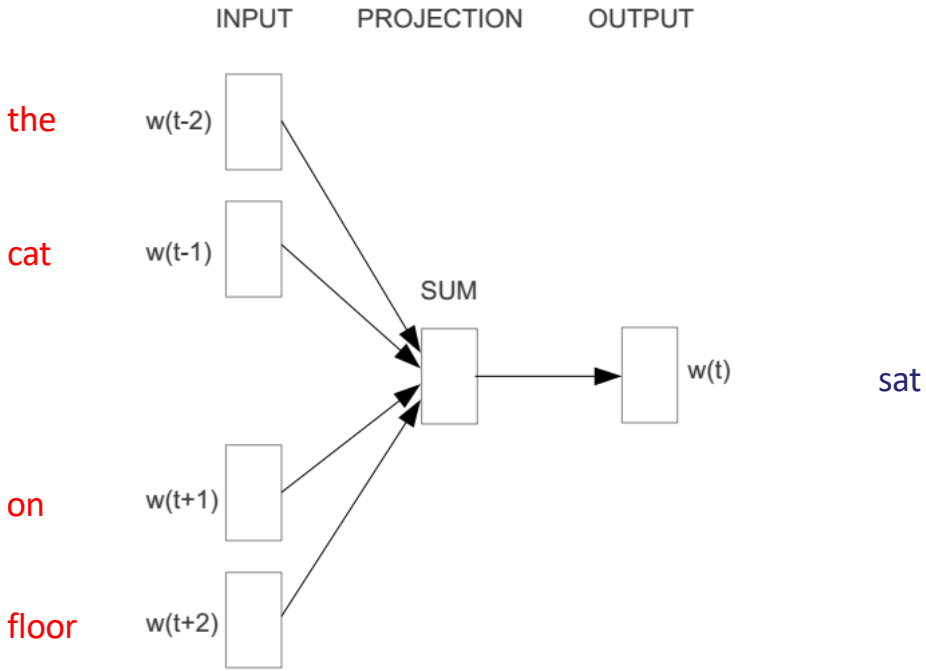
Represent the meaning of **words** – word2vec

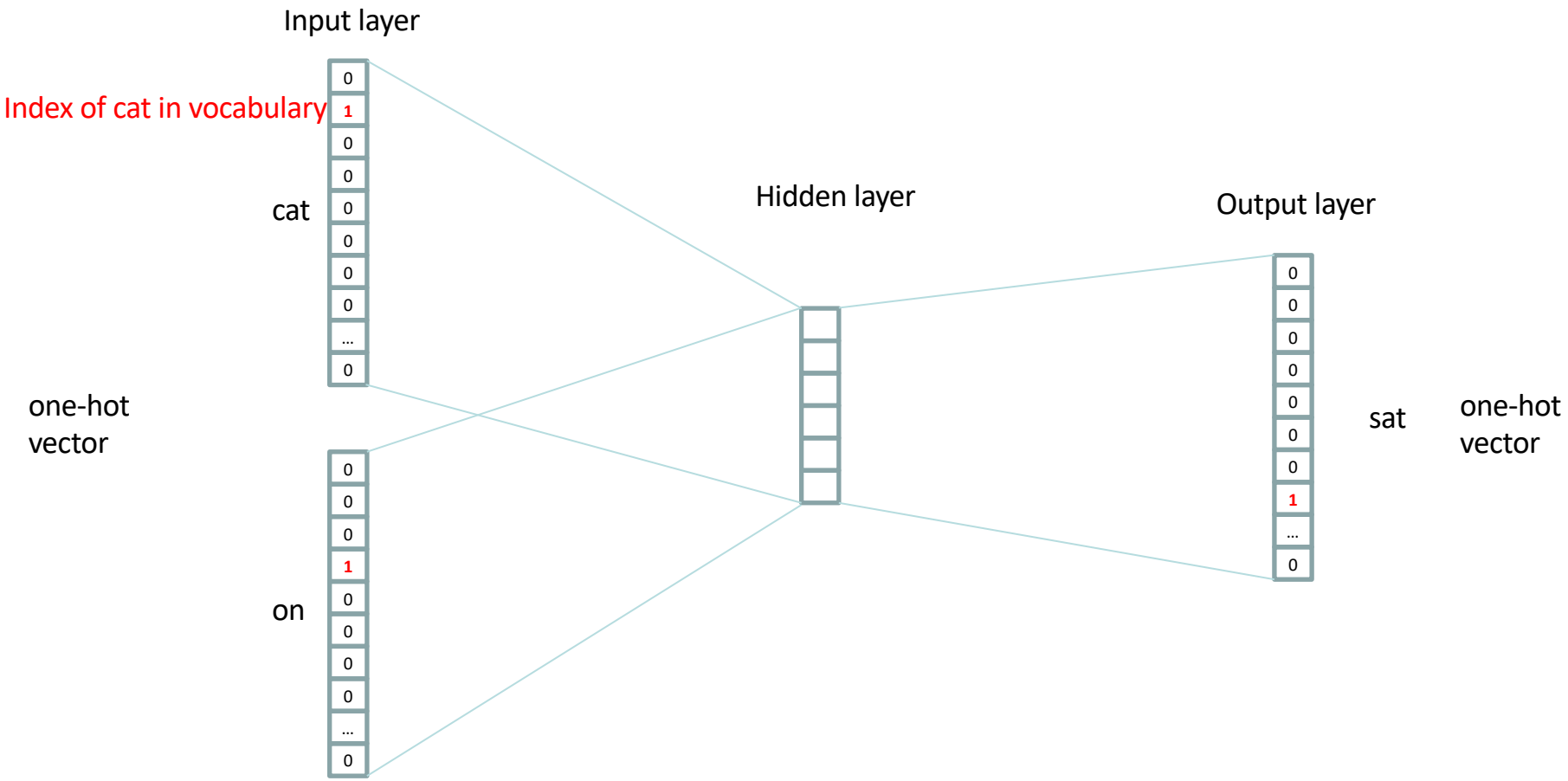
- 2 basic structural models:
  - Continuous Bag of Words (CBOW): use a window of words to predict the middle word
  - Skip-gram (SG): use a word to predict the surrounding ones in window.

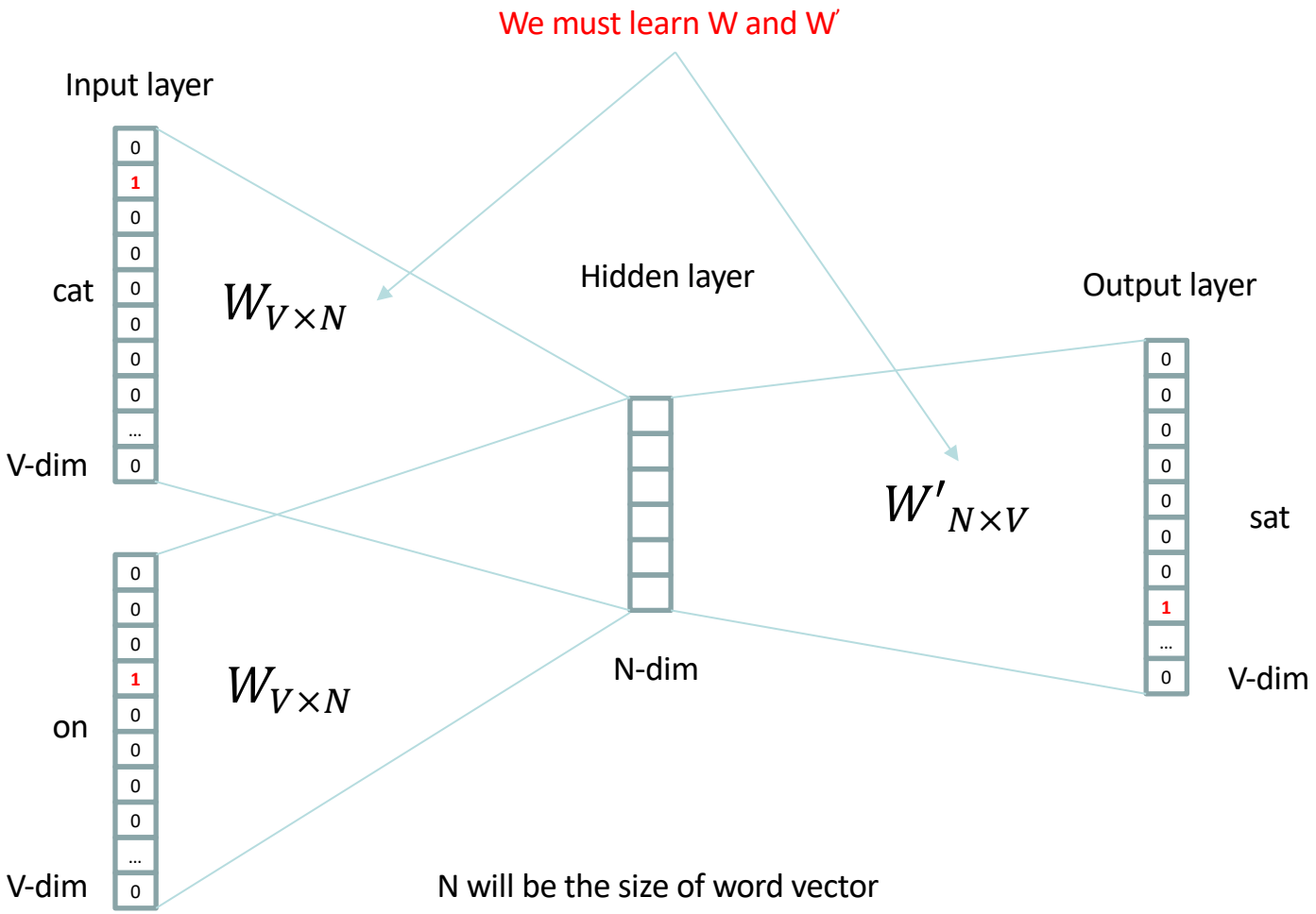


Word2vec – Continuous Bag of Word

- E.g. “The cat <sat> on floor”
  - Window size = 2

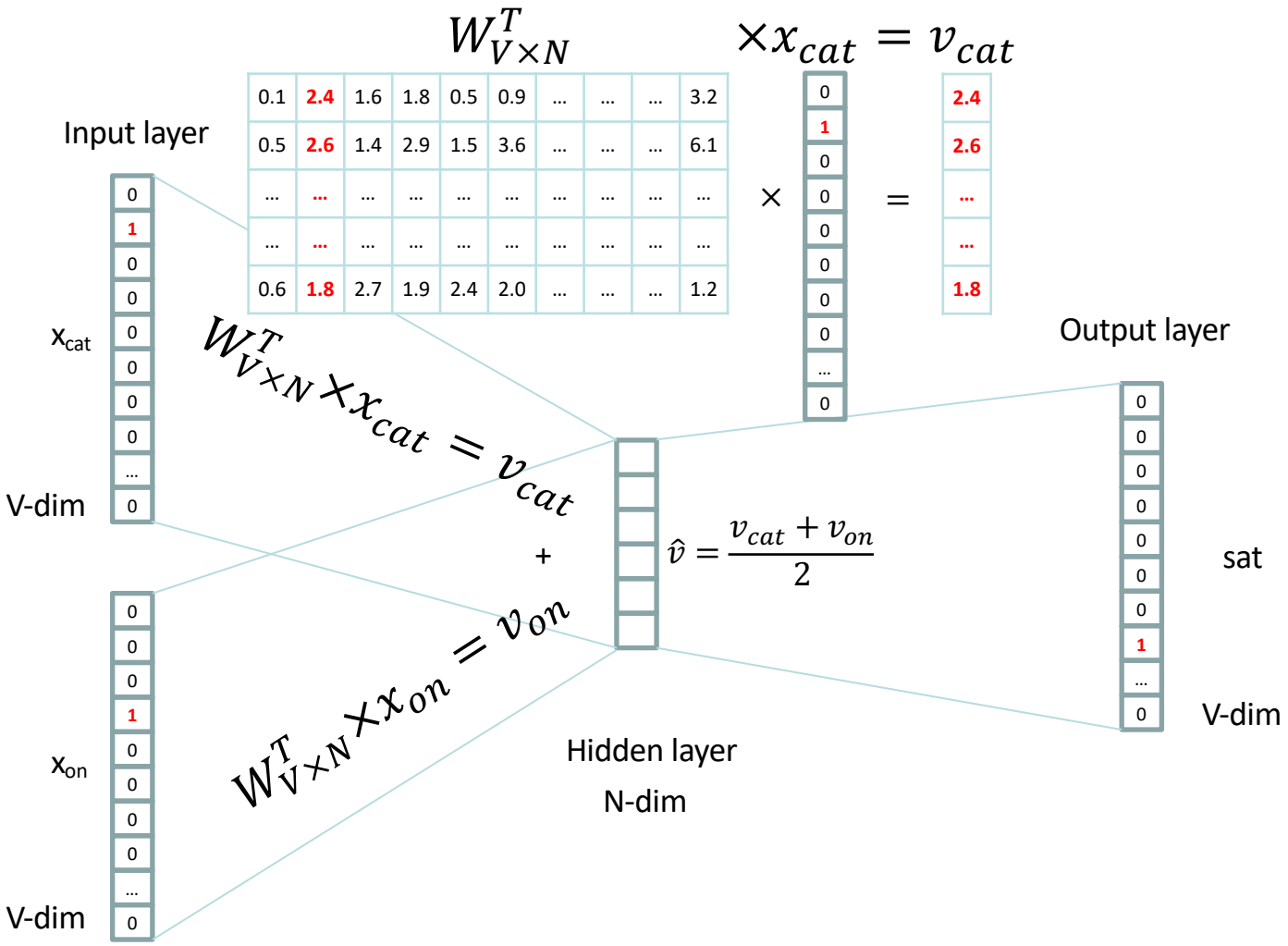


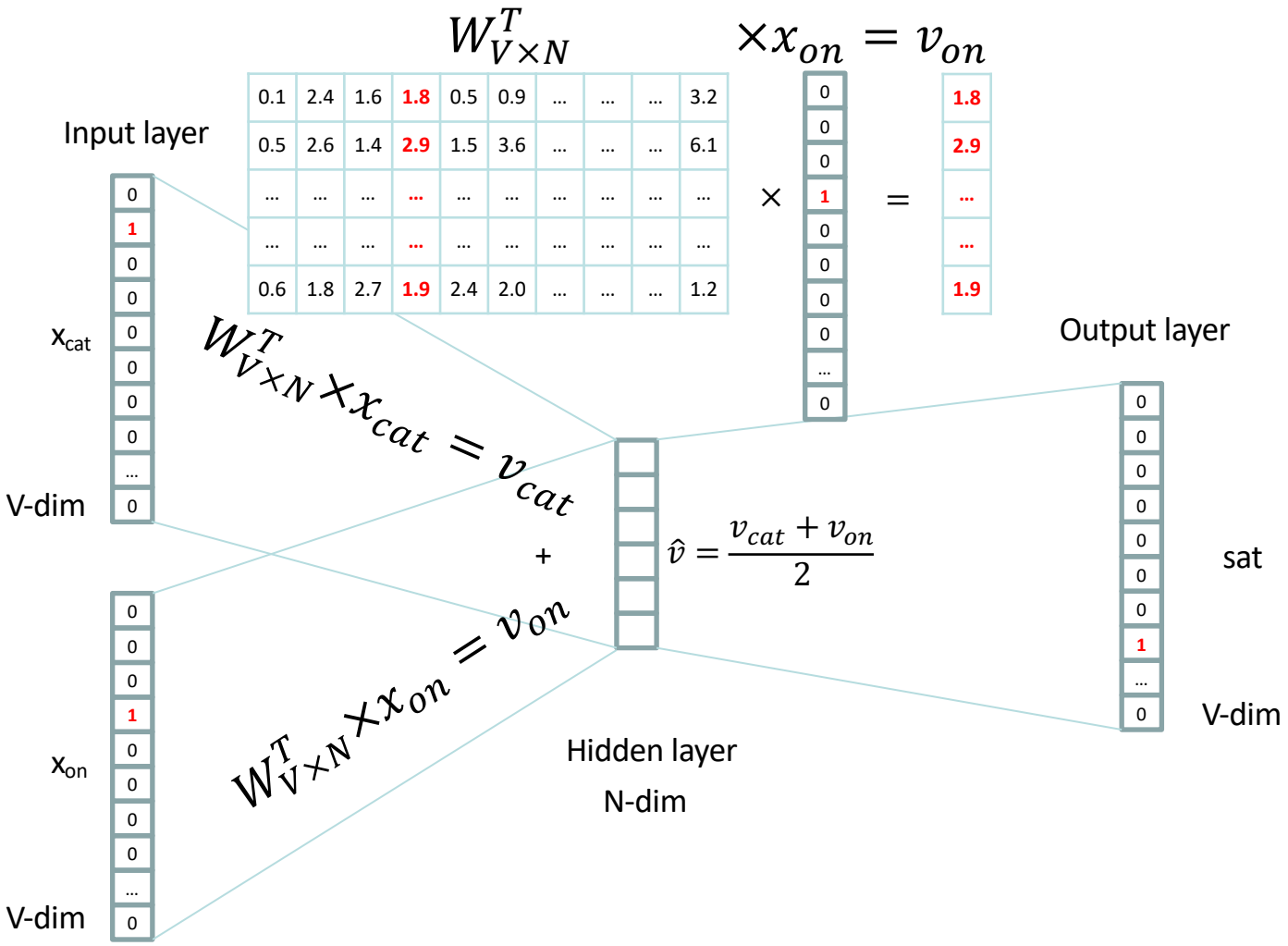

Input layer
Index of cat in vocabulary
cat
one-hot vector
on
Hidden layer
Output layer
sat
one-hot vector

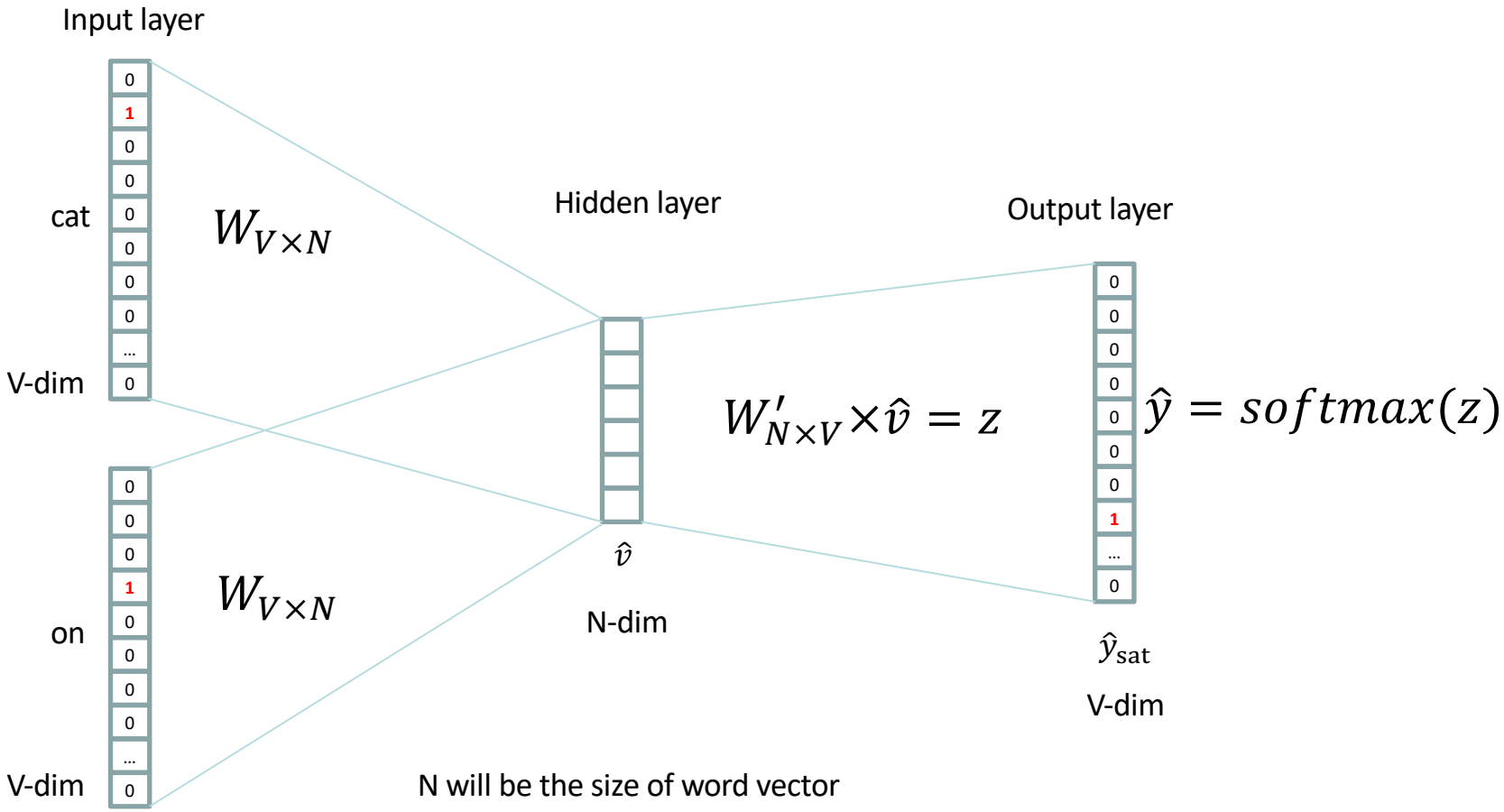

We must learn W and W'
Input layer
cat
V-dim
on
V-dim
$W_{V\times N}$
$W_{V\times N}$
Hidden layer
N-dim
$W'_{N\times V}$
Output layer
sat
V-dim
N will be the size of word vector

Deep Learning

- Hidden layer represents feature space
  - Making explicit features in the data…
  - … that are relevant for a certain task
- Determine features automatically
  - Learning suitable mappings into feature space
- Deep learning also known as representation learning


$W^T_{V\times N}$
$\times x_{cat} = v_{cat}$
Input layer
0.1 2.4 1.6 1.8 0.5 0.9 ... ... 3.2
0.5 2.6 1.4 2.9 1.5 3.6 ... ... 6.1
... ...
... ...
0.6 1.8 2.7 1.9 2.4 2.0 ... ... 1.2
×
=
2.4
2.6
...
...
1.8
Output layer
x_{cat}
V-dim
$W^T_{V\times N} x_{cat} = v_{cat}$
+
$W^T_{V\times N} x_{on} = v_{on}$
$\hat{v} = \frac{v_{cat} + v_{on}}{2}$
sat
x_{on}
V-dim
Hidden layer
N-dim


$W^T_{V\times N}$
$\times x_{on} = v_{on}$
Input layer
0.1 2.4 1.6 1.8 0.5 0.9 ... ... 3.2
0.5 2.6 1.4 2.9 1.5 3.6 ... ... 6.1
... ...
... ...
0.6 1.8 2.7 1.9 2.4 2.0 ... ... 1.2
×
=
1.8
2.9
...
...
1.9
x_{cat}
V-dim
$W^T_{V\times N} x_{cat} = v_{cat}$
+
$W^T_{V\times N} x_{on} = v_{on}$
$\hat{v} = \frac{v_{cat} + v_{on}}{2}$
Hidden layer
N-dim
Output layer
sat
V-dim
x_{on}
V-dim


Input layer
cat
V-dim
on
V-dim
$W_{V\times N}$
$W_{V\times N}$
Hidden layer
$\hat{v}$
N-dim
$W'_{N\times V}\times\hat{v}=z$
Output layer
$\hat{y}=softmax(z)$
$\hat{y}_{\text{sat}}$
V-dim
N will be the size of word vector

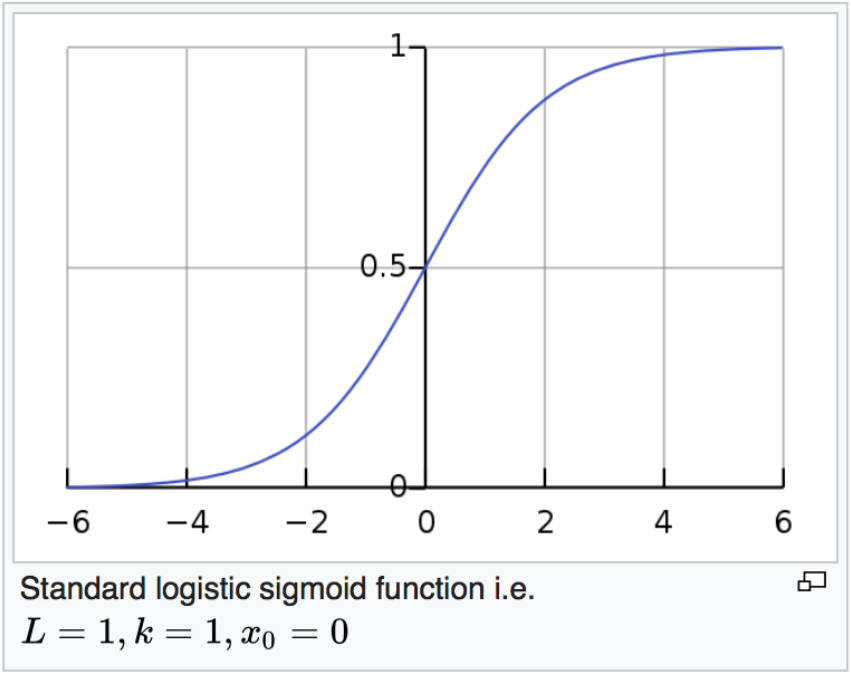
Logistic function

A **logistic function** or **logistic curve** is a common "S" shape (sigmoid curve), with equation:

$$f(x) = \frac{L}{1 + e^{-k(x-x_0)}}$$

where

- e = the natural logarithm base (also known as Euler's number),
- x_0 = the x-value of the sigmoid's midpoint,
- L = the curve's maximum value, and
- k = the steepness of the curve.[1]



Standard logistic sigmoid function i.e. $L = 1, k = 1, x_0 = 0$

[Wikipedia]

softmax(**z**)

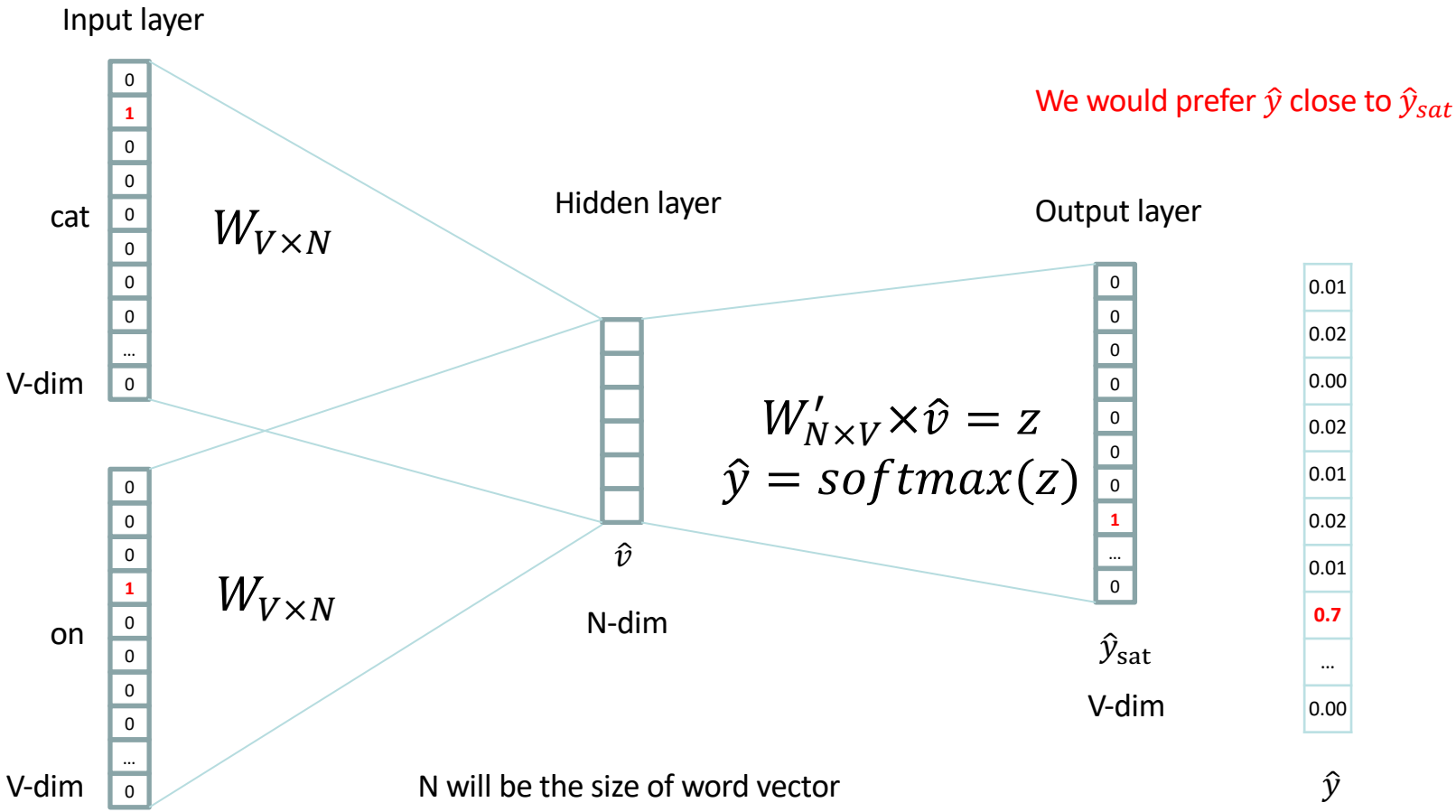
The **softmax function**, or **normalized exponential function**, is a generalization of the logistic function that "squashes" a K-dimensional vector $\mathbf{z}$ of arbitrary real values to a K-dimensional vector $\sigma(\mathbf{z})$ of real values in the range [0, 1] that add up to 1. The function is given by

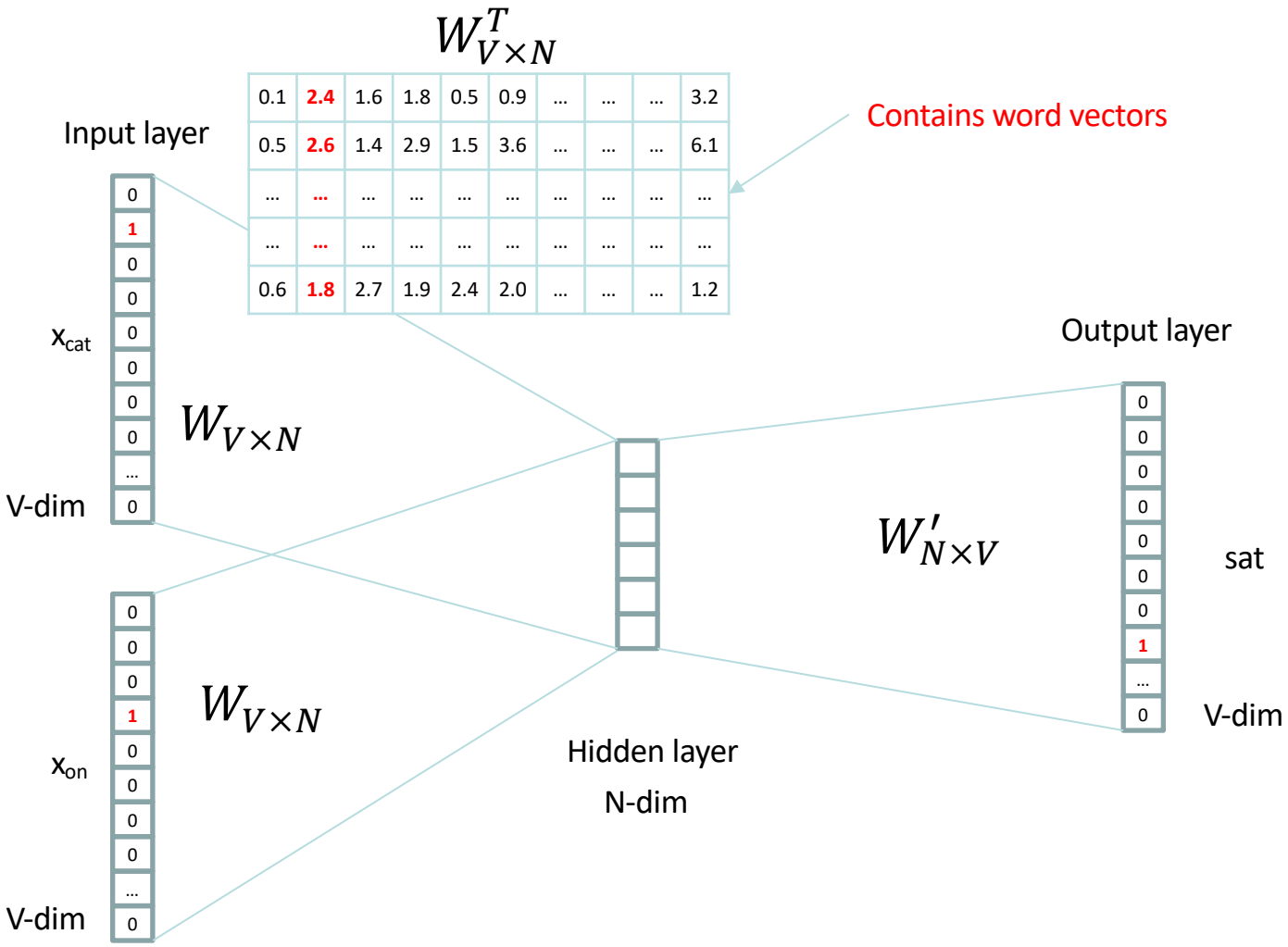
$$\sigma : \mathbb{R}^K \to [0,1]^K$$

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \quad \text{for } j = 1, \ldots, K.$$

In probability theory, the output of the softmax function can be used to represent a categorical distribution – that is, a probability distribution over K different possible outcomes.

[Wikipedia]


Input layer
cat
V-dim
on
V-dim
$W_{V\times N}$
$W_{V\times N}$
Hidden layer
$\hat{v}$
N-dim
$W'_{N\times V}\times\hat{v}=z$
$\hat{y}=softmax(z)$
Output layer
$\hat{y}_{sat}$
V-dim
We would prefer $\hat{y}$ close to $\hat{y}_{sat}$
$\hat{y}$
N will be the size of word vector



Consider either W or W' as the word's representation.

Word Analogies

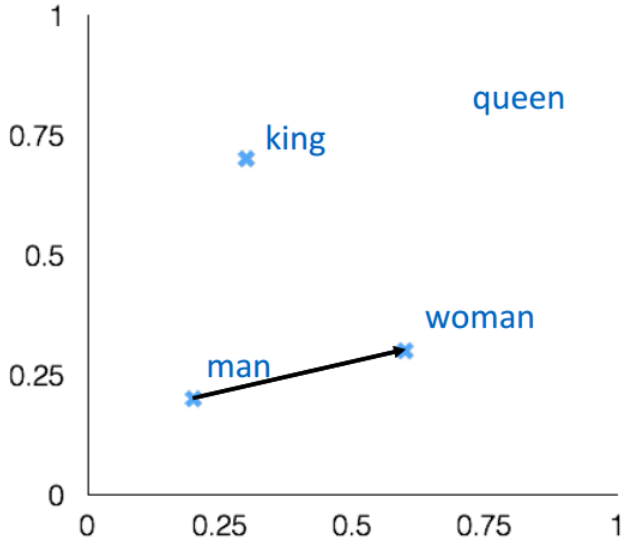
Test for linear relationships, examined by Mikolov et al. (2014)

a:b :: c:?

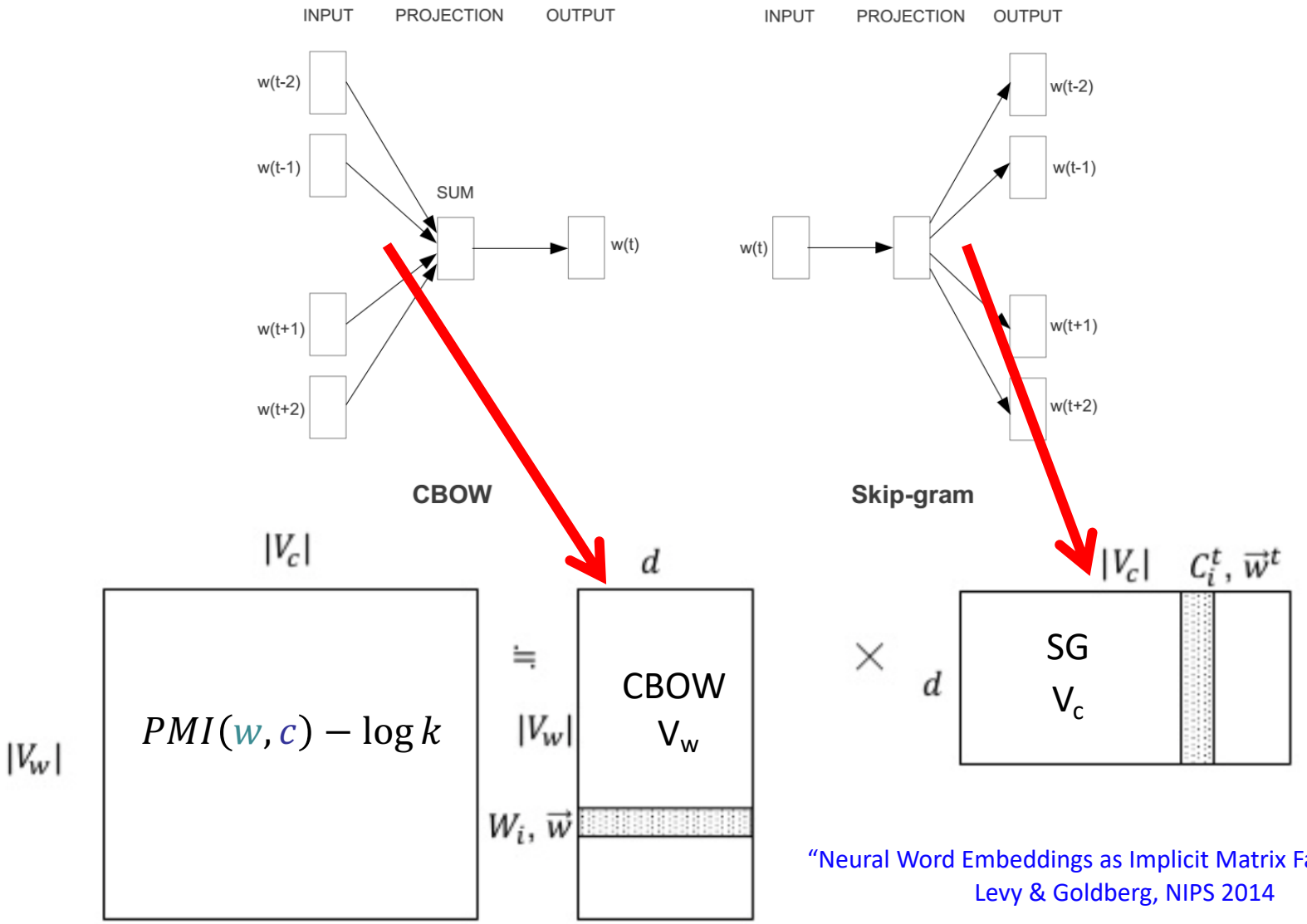
$$d = \arg\max_x \frac{(w_b - w_a + w_c)^T w_x}{||w_b - w_a + w_c|| ||w_x||}$$

man:woman :: king:?

+	king	[0.30 0.70]
-	man	[0.20 0.20]
+	woman	[0.60 0.30]
	queen	[0.70 0.80]



The Picture: CBOW and Skip-Gram (SG)



"Neural Word Embeddings as Implicit Matrix Factorization"
Levy & Goldberg, NIPS 2014

Deep Learning