THE UNIVERSITY OF BRITISH COLUMBIA

Topics in AI (CPSC 532S):
Multimodal Learning with Vision, Language and Sound

Lecture 8: Word2Vec, Language Models and RNNs

Course **Logistics**

- **Assignment 3**
- **Final project group** Goolge form will be out **tomorrow**

Representing a **Word:** One Hot Encoding

Vocabulary		**one-hot** encodings
dog	1	[1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
cat	2	[0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
person	3	[0, 0, 1, 0, 0, 0, 0, 0, 0, 0]
holding	4	[0, 0, 0, 1, 0, 0, 0, 0, 0, 0]
tree	5	[0, 0, 0, 0, 1, 0, 0, 0, 0, 0]
computer	6	[0, 0, 0, 0, 0, 1, 0, 0, 0, 0]
using	7	[0, 0, 0, 0, 0, 0, 1, 0, 0, 0]

*slide from V. Ordonex

Representing **Phrases**: Bag-of-Words

Vocabulary

dog	1
cat	2
person	3
holding	4
tree	5
computer	6
using	7

bag-of-words representation

person holding dog	{3, 4, 1}	[1, 0, 1, 1, 0, 0, 0, 0, 0, 0]
person holding cat	{3, 4, 2}	[1, 1, 0, 1, 0, 0, 0, 0, 0, 0]
person using computer	{3, 7, 6}	[0, 0, 0, 1, 0, 1, 1, 0, 0, 0]

dog cat person holding tree computer using

person using computer person holding cat	{3, 3, 7, 6, 2}	[0, 1, 2, 1, 0, 1, 1, 0, 0, 0]

*slide from V. Ordonex

Distributional Hypothesis [Lenci, 2008]

— At least certain aspects of the meaning of lexical expressions depend on their distributional properties in the linguistic contexts

— The degree of semantic similarity between two linguistic expressions is a function of the similarity of the two linguistic contexts in which they can appear

* Adopted from slides by Louis-Philippe Morency

What is the meaning of “**bardiwac**”?

— He handed her glass of **bardiwac**.

— Beef dishes are made to complement the **bardiwacs**.

— Nigel staggered to his feet, face flushed from too much **bardiwac**.

— Malbec, one of the lesser-known **bardiwac** grapes, responds well to Australia’s sunshine.

— I dined off bread and cheese and this excellent **bardiwac**.

—The drinks were delicious: blood-red **bardiwac** as well as light, sweet Rhenish.

bardic is an alcoholic beverage made from grapes

* Adopted from slides by Louis-Philippe Morency

The **Use Theory** of Meaning

“If you can understand and predict in which context a word will appear in, then you understood the meaning of the word” [Paul Horwich]

Geometric Interpretation: Co-occurrence as feature

— Row vector describes usage of word in a corpus of text

— Can be seen as coordinates o the point in an n-dimensional Euclidian space

	get	see	use	hear	eat	kill
knife	51	20	84	0	3	0
cat	52	58	4	4	6	26
dog	115	83	10	42	33	17
boat	59	39	23	4	0	0
cup	98	14	6	2	1	0
pig	12	17	3	2	9	27
banana	11	2	2	0	18	0

Co-occurrence Matrix

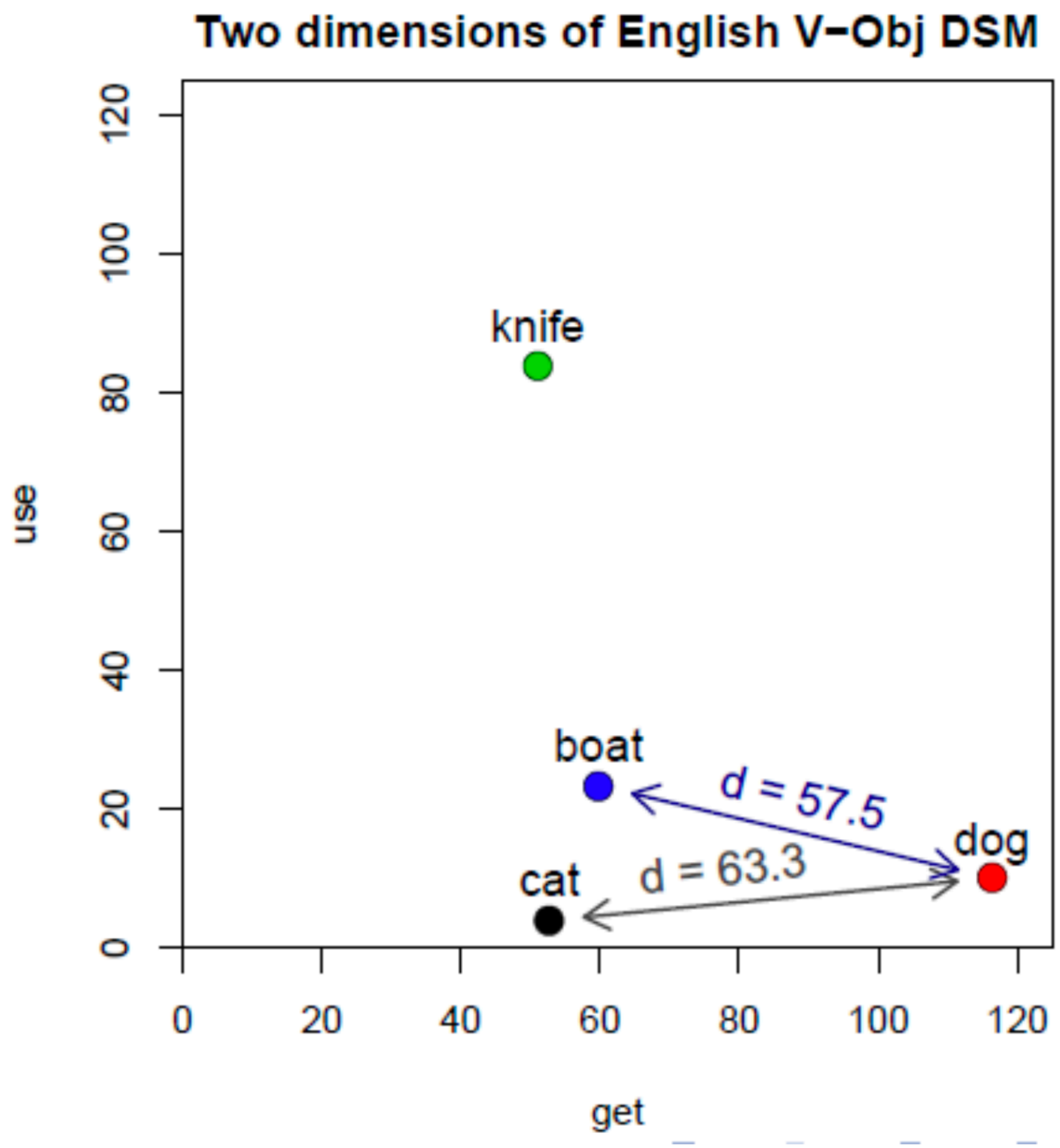
* Slides from Louis-Philippe Morency

Distance and Similarity

— Illustrated in two dimensions

— Similarity = spatial proximity (Euclidian distance)

— Location depends on frequency of noun (dog is 27 times as frequent as ca)



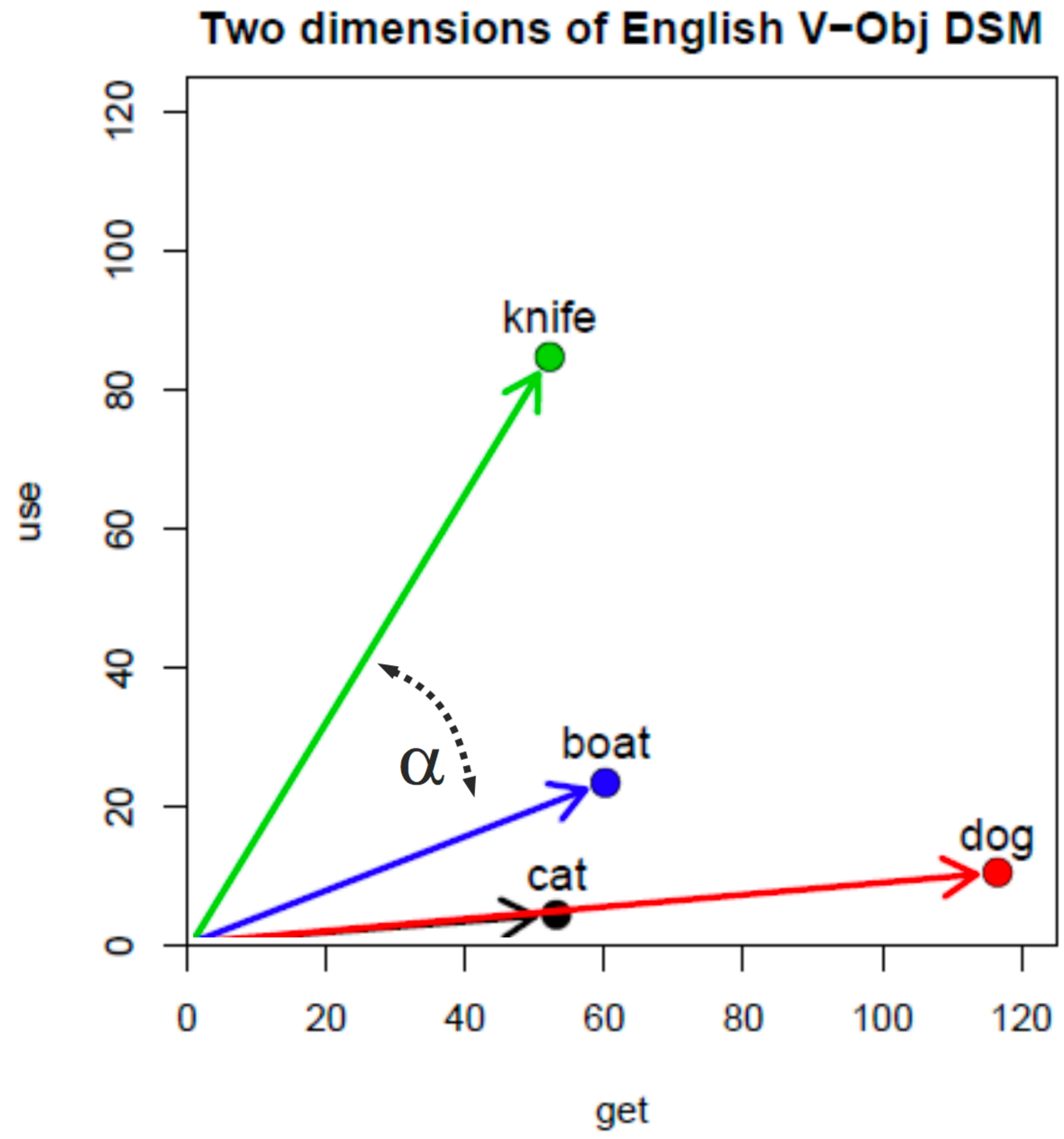
* Slides from Louis-Philippe Morency

Angle and Similarity

— direction is more important than location

— normalize length of vectors

— or use angle as a distance measure



* Slides from Louis-Philippe Morency

Geometric Interpretation: Co-occurrence as feature

— Row vector describes usage of word in a corpus of text

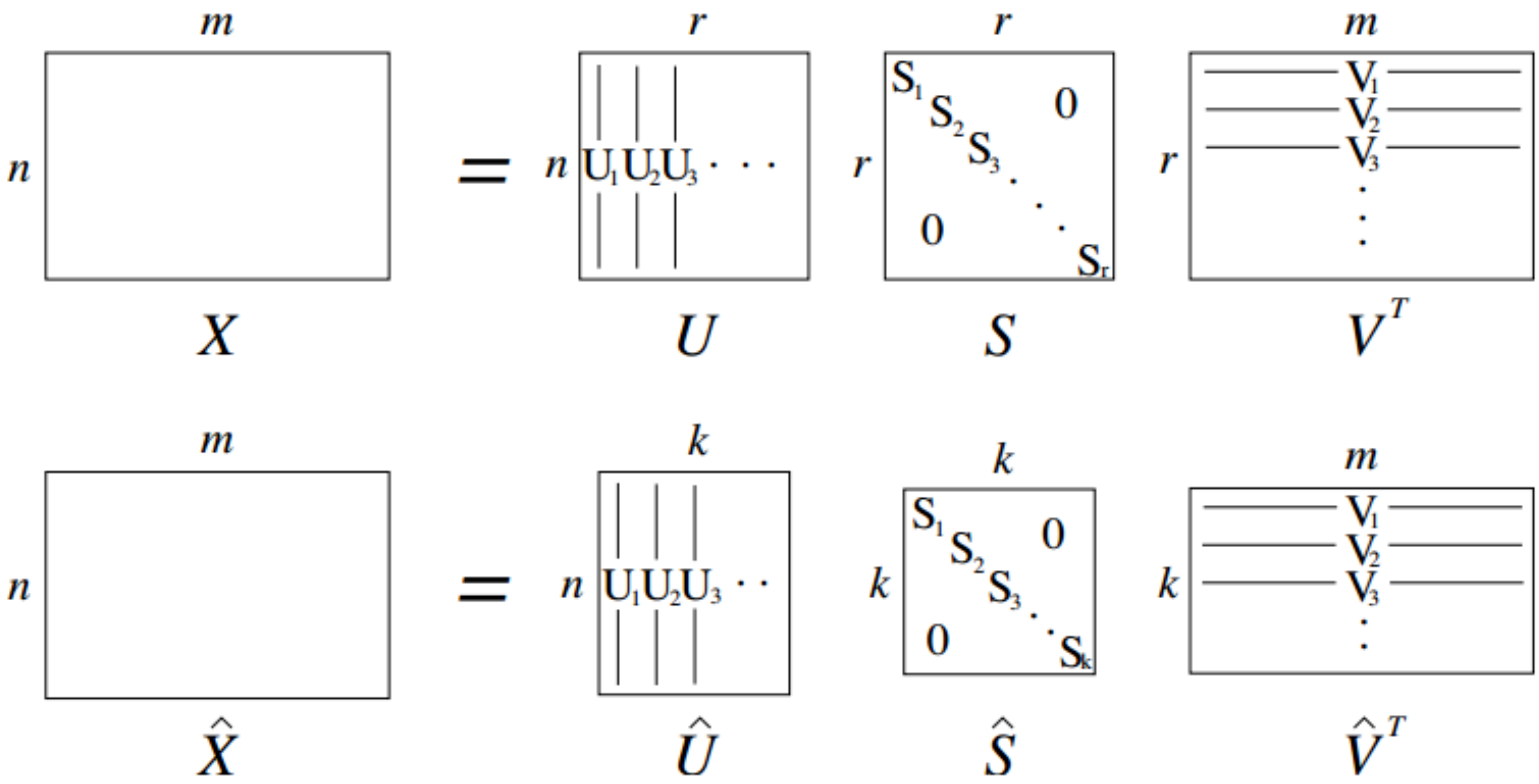
— Can be seen as coordinates of the point in an n-dimensional Euclidian space

Way too high dimensional!

	get	see	use	hear	eat	kill
knife	51	20	84	0	3	0
cat	52	58	4	4	6	26
dog	115	83	10	42	33	17
boat	59	39	23	4	0	0
cup	98	14	6	2	1	0
pig	12	17	3	2	9	27
banana	11	2	2	0	18	0

Co-occurrence Matrix

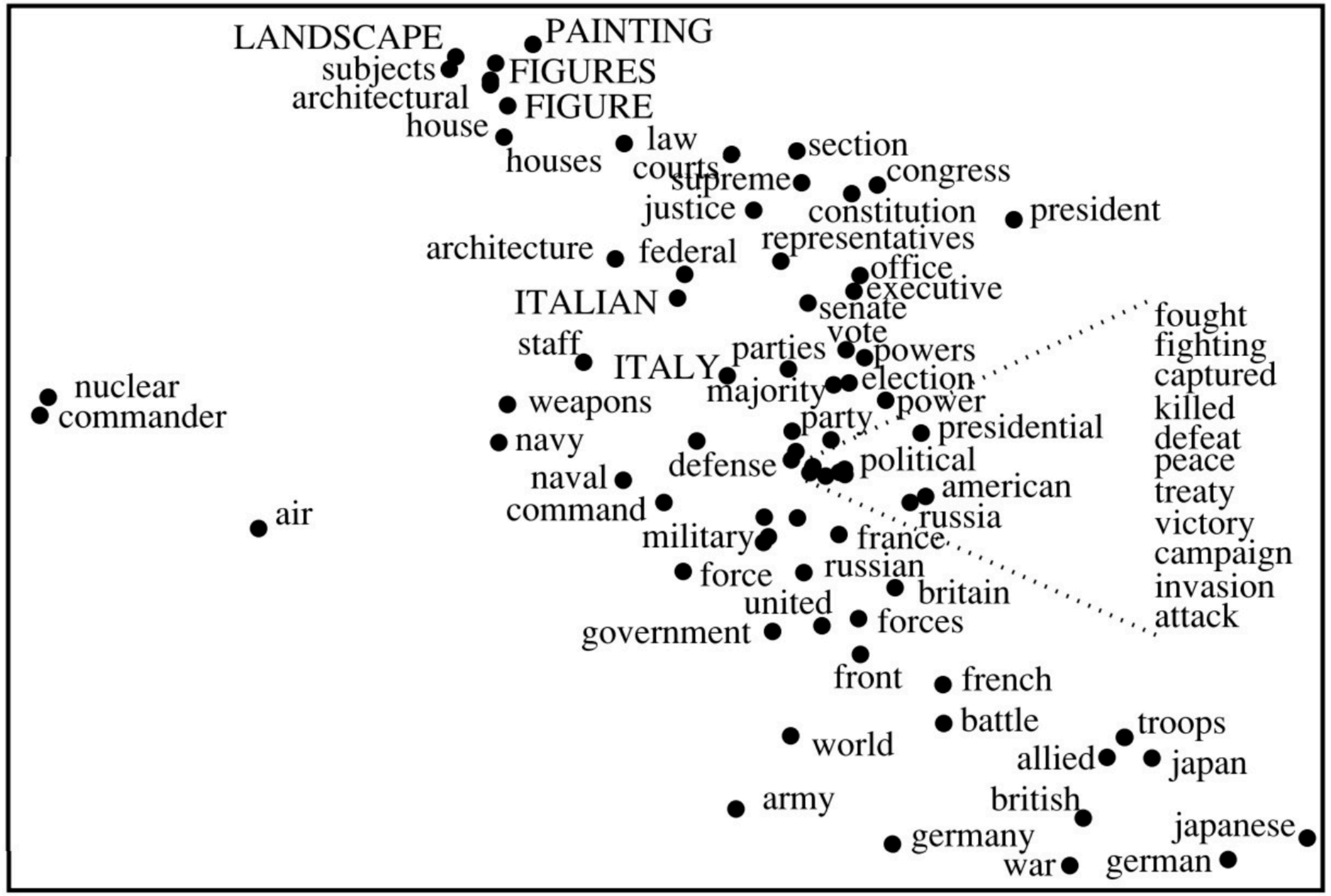
SVD for Dimensionality Reduction



*slide from Vagelis Hristidis

Learned Word Vector Visualization

We can also use other methods, like LLE here:



Nonlinear dimensionality reduction by locally linear embedding. Sam Roweis & Lawrence Saul. Science, v.290,2000

[Roweis and Saul, 2000]

Issues with **SVD**

Computational cost for a $d \times n$ matrix is $\mathcal{O}(dn^2)$, where $d < n$

— Makes it not possible for large number of word vocabularies or documents

It is hard to incorporate out of sample (**new**) words or documents

*slide from Vagelis Hristidis

word2vec: Representing the Meaning of Words [Mikolov et al., 2013]

Key idea: Predict surrounding words of every word

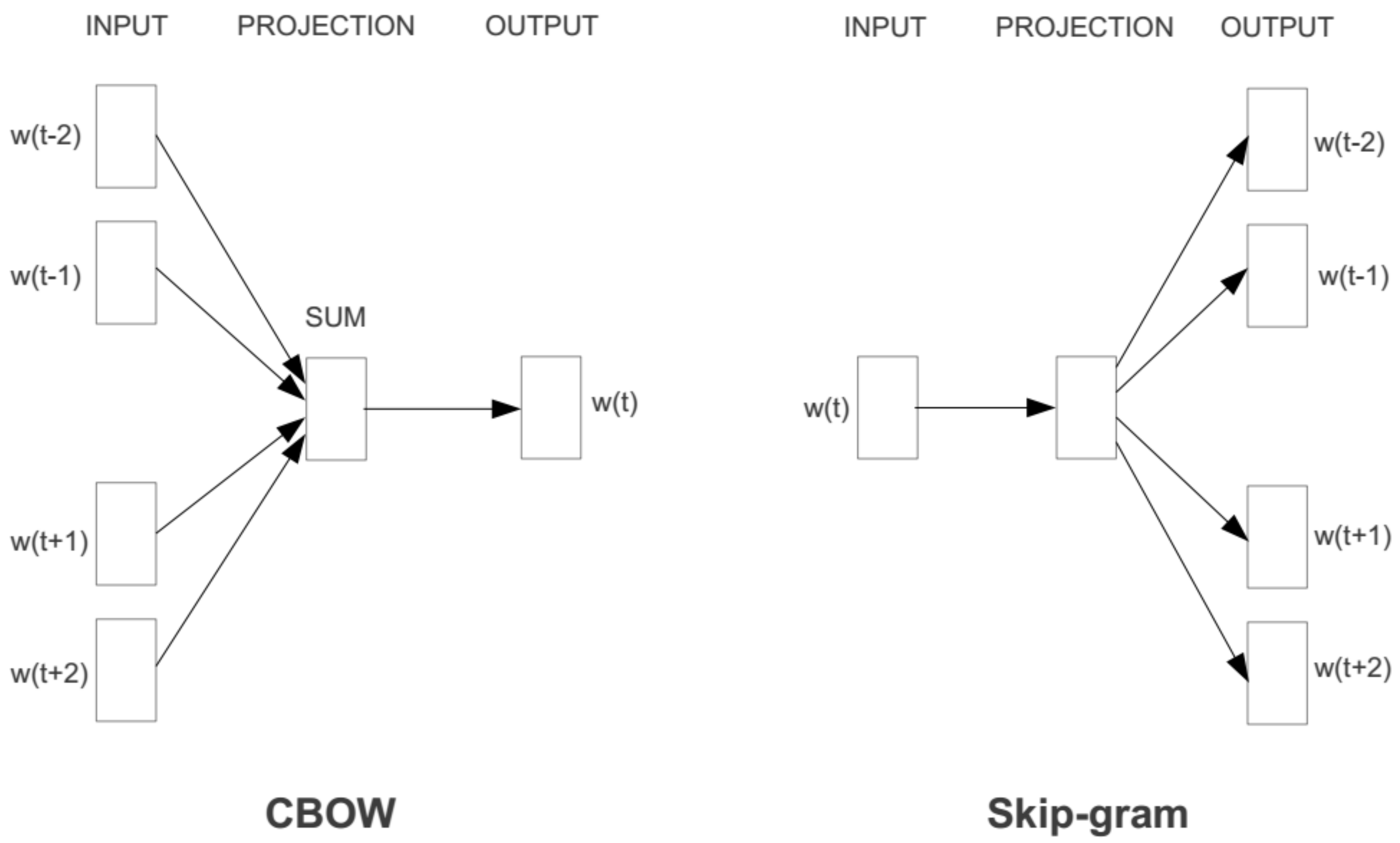
Benefits: Faster and easier to incorporate new document, words, etc.

*slide from Vagelis Hristidis

word2vec: Representing the Meaning of Words [Mikolov et al., 2013]

Key idea: Predict surrounding words of every word

Benefits: Faster and easier to incorporate new document, words, etc.



Continuous Bag of Words (**CBOW**): use context words in a window to predict middle word

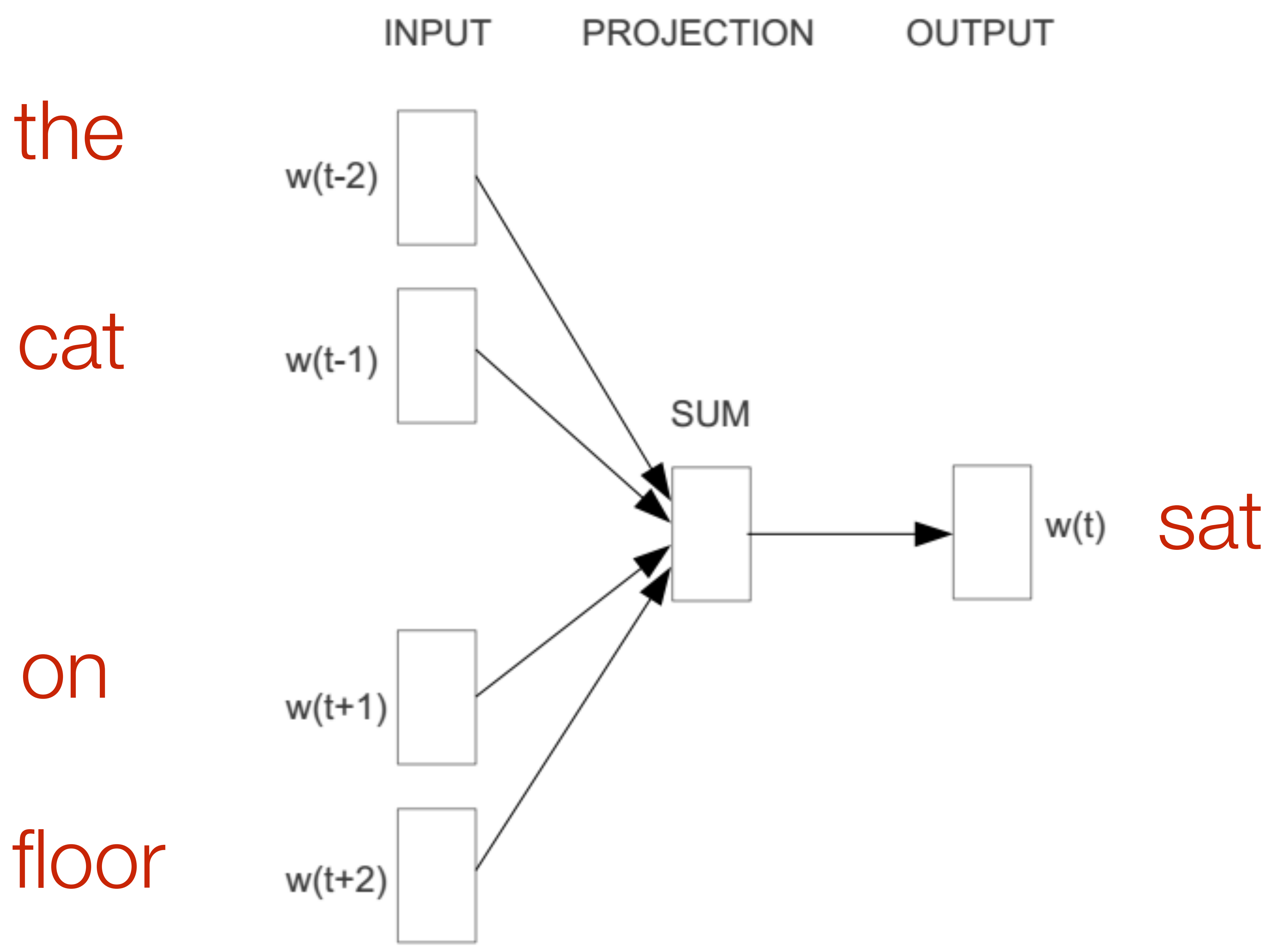
Skip-gram: use the middle word to predict surrounding ones in a window

*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

[Mikolov et al., 2013]

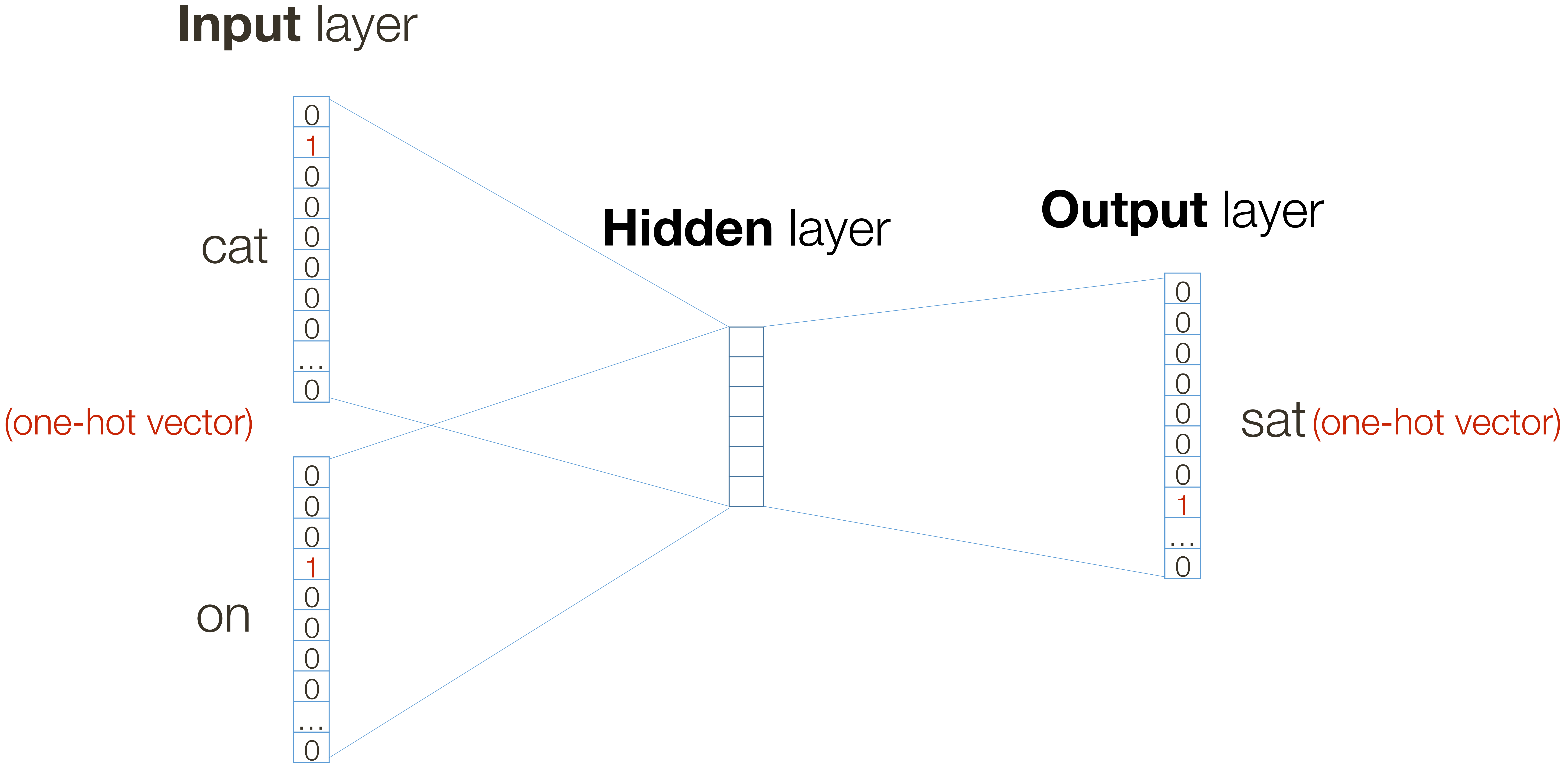
Example: "The cat sat on floor" (window size 2)



*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

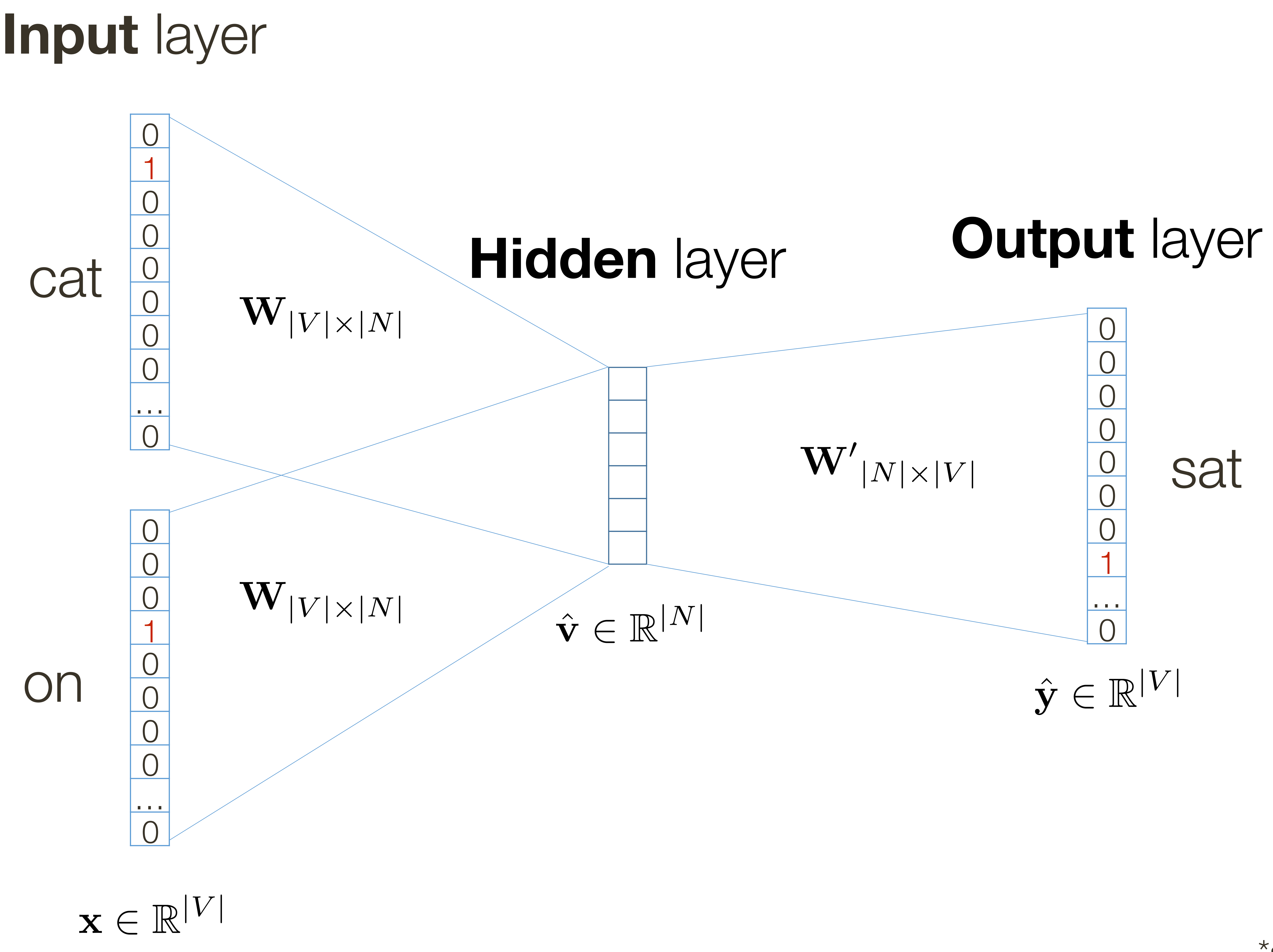
[Mikolov et al., 2013]



*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

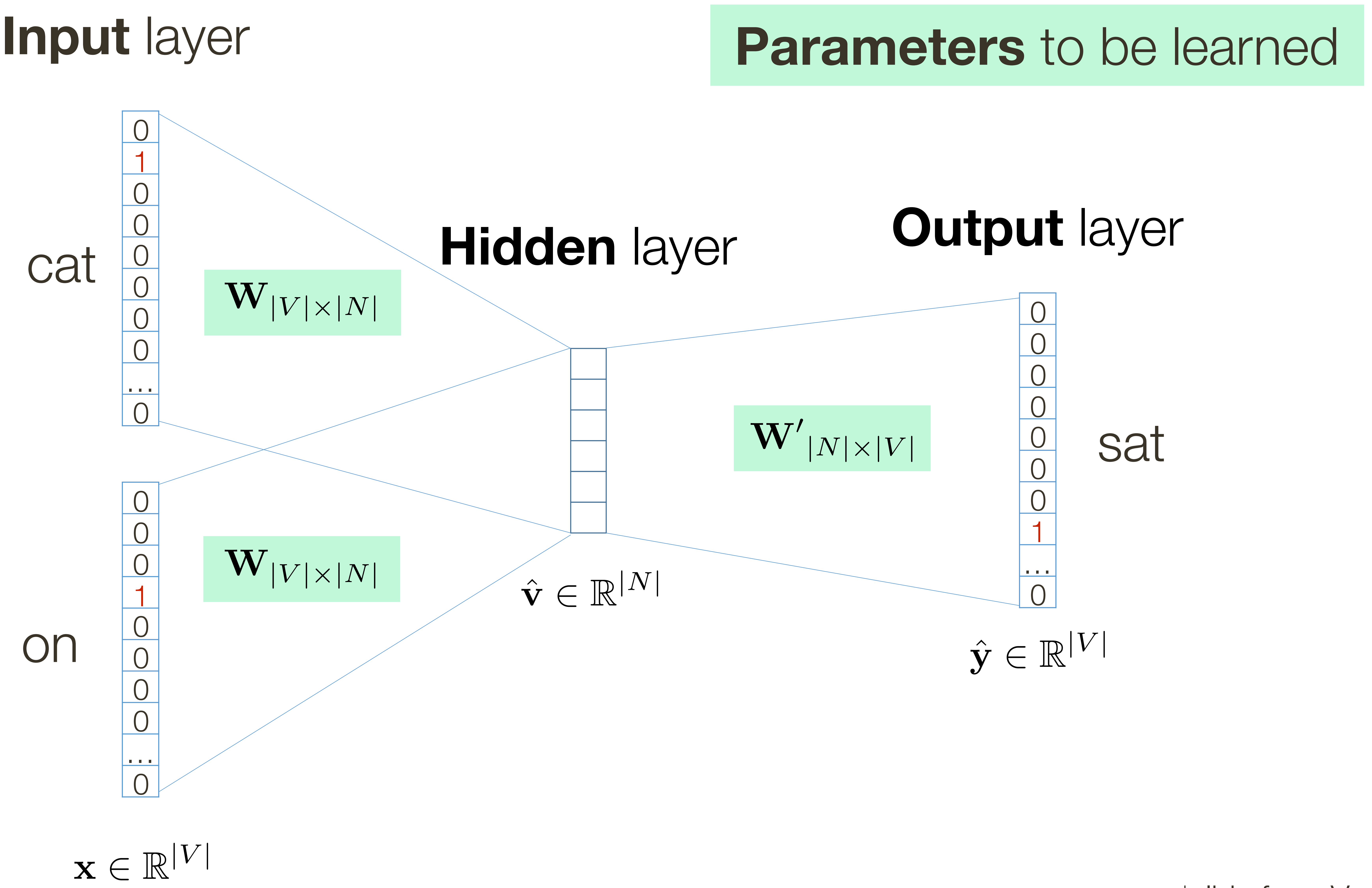
[Mikolov et al., 2013]



*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

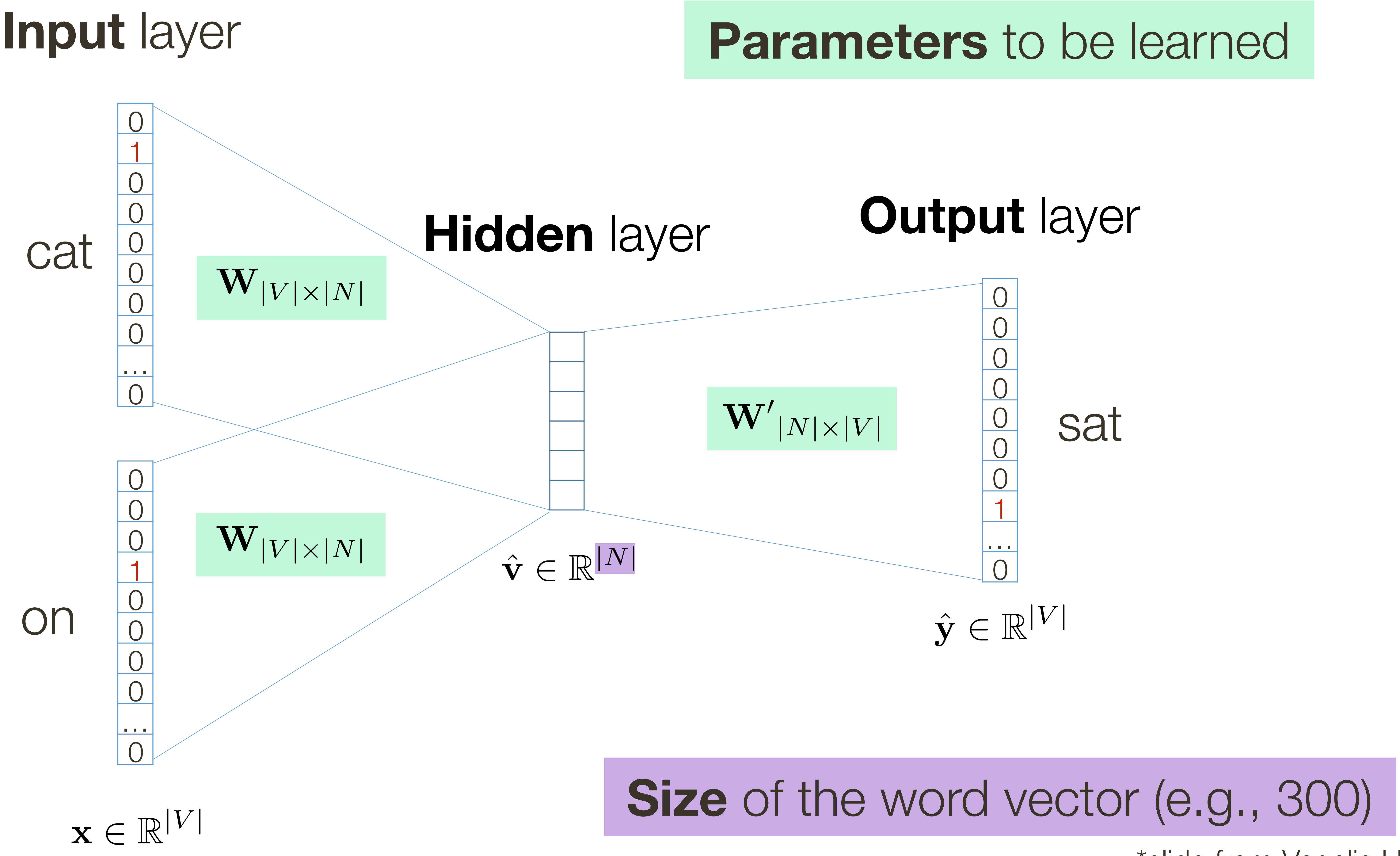
[Mikolov et al., 2013]



*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

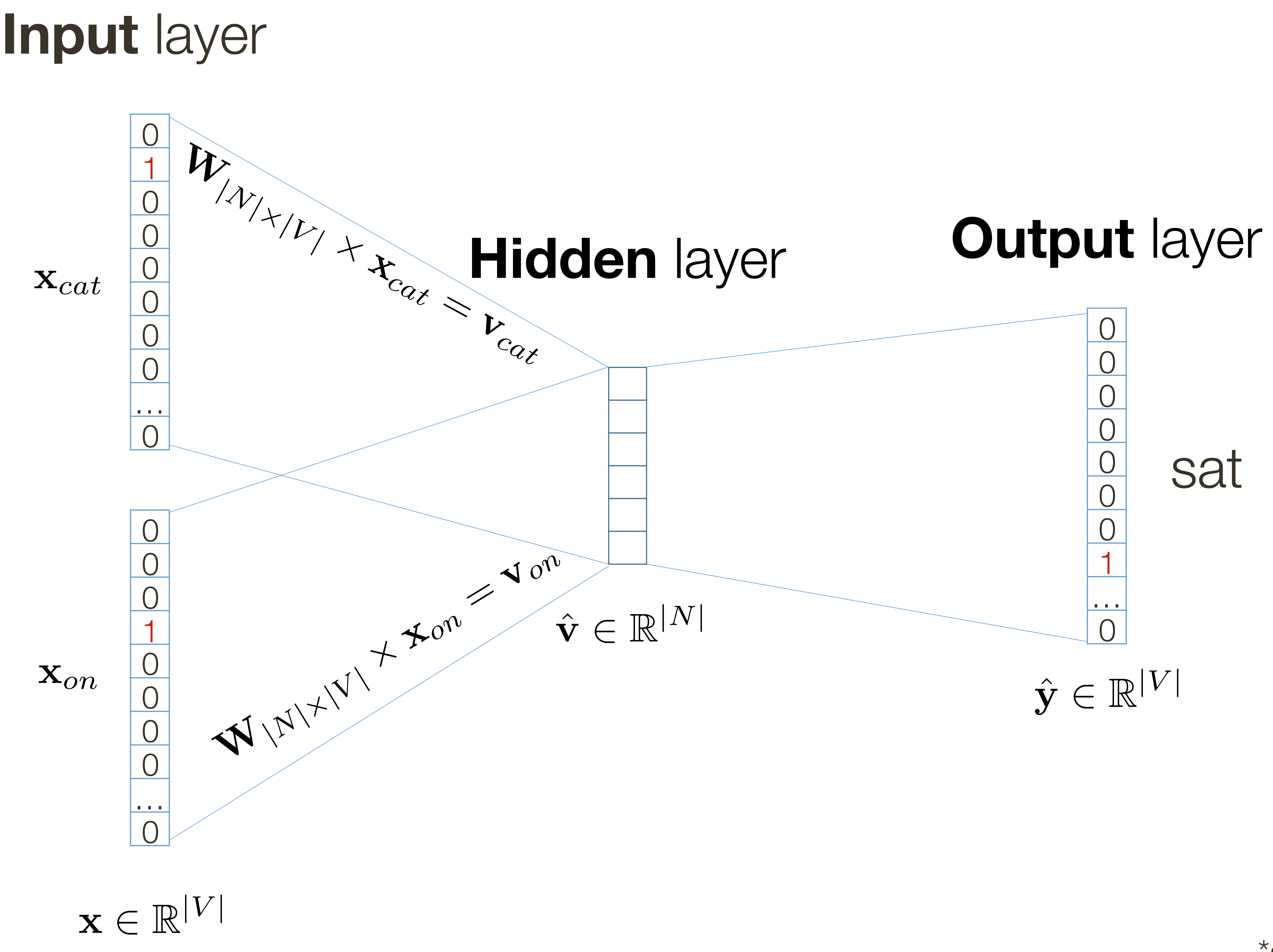
[Mikolov et al., 2013]



*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

[Mikolov et al., 2013]

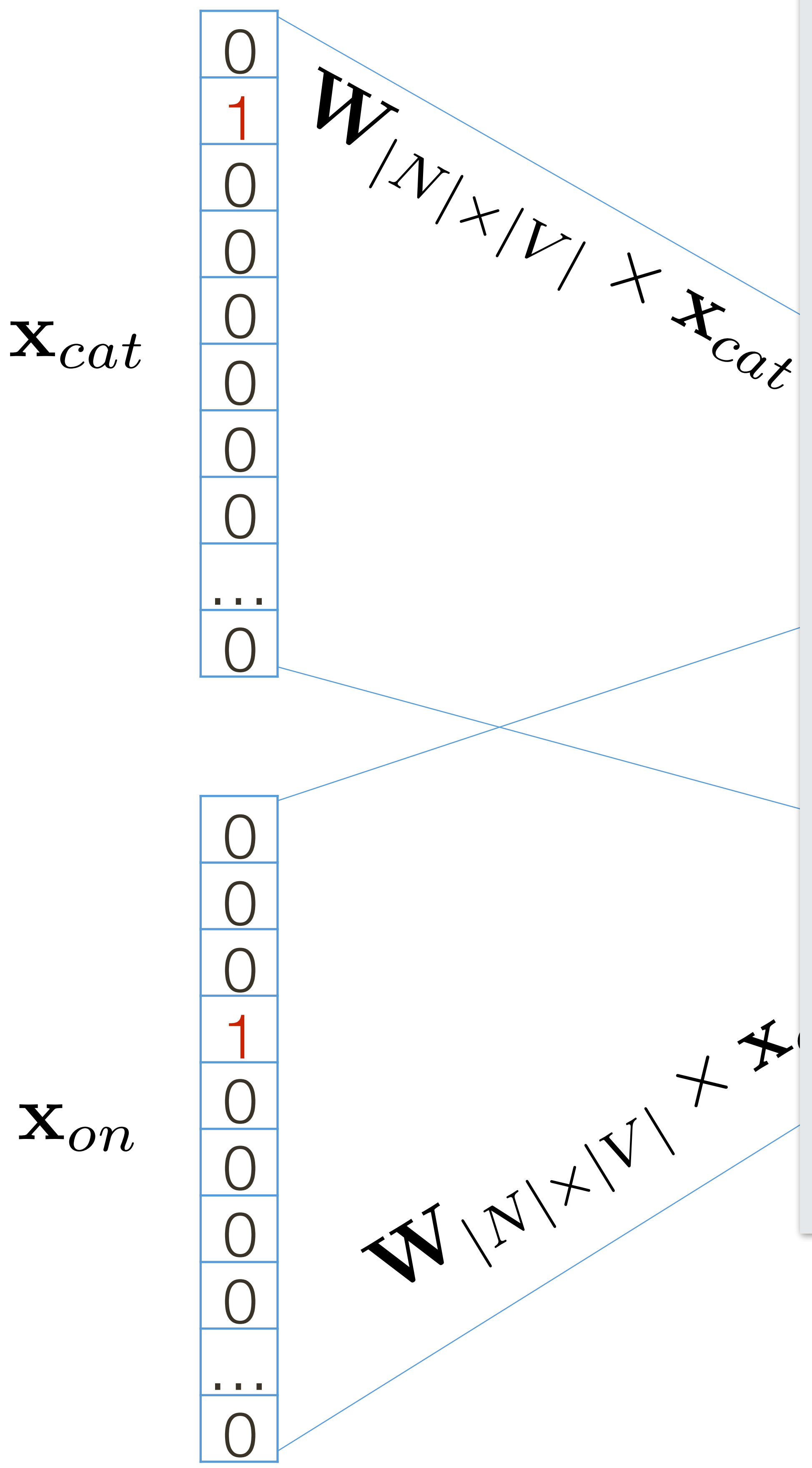


*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

[Mikolov et al., 2013]

Input layer



$\mathbf{x} \in \mathbb{R}^{|V|}$

$$\mathbf{W}^T_{|V|\times|N|} \times \mathbf{x}_{cat} = \mathbf{v}_{cat}$$

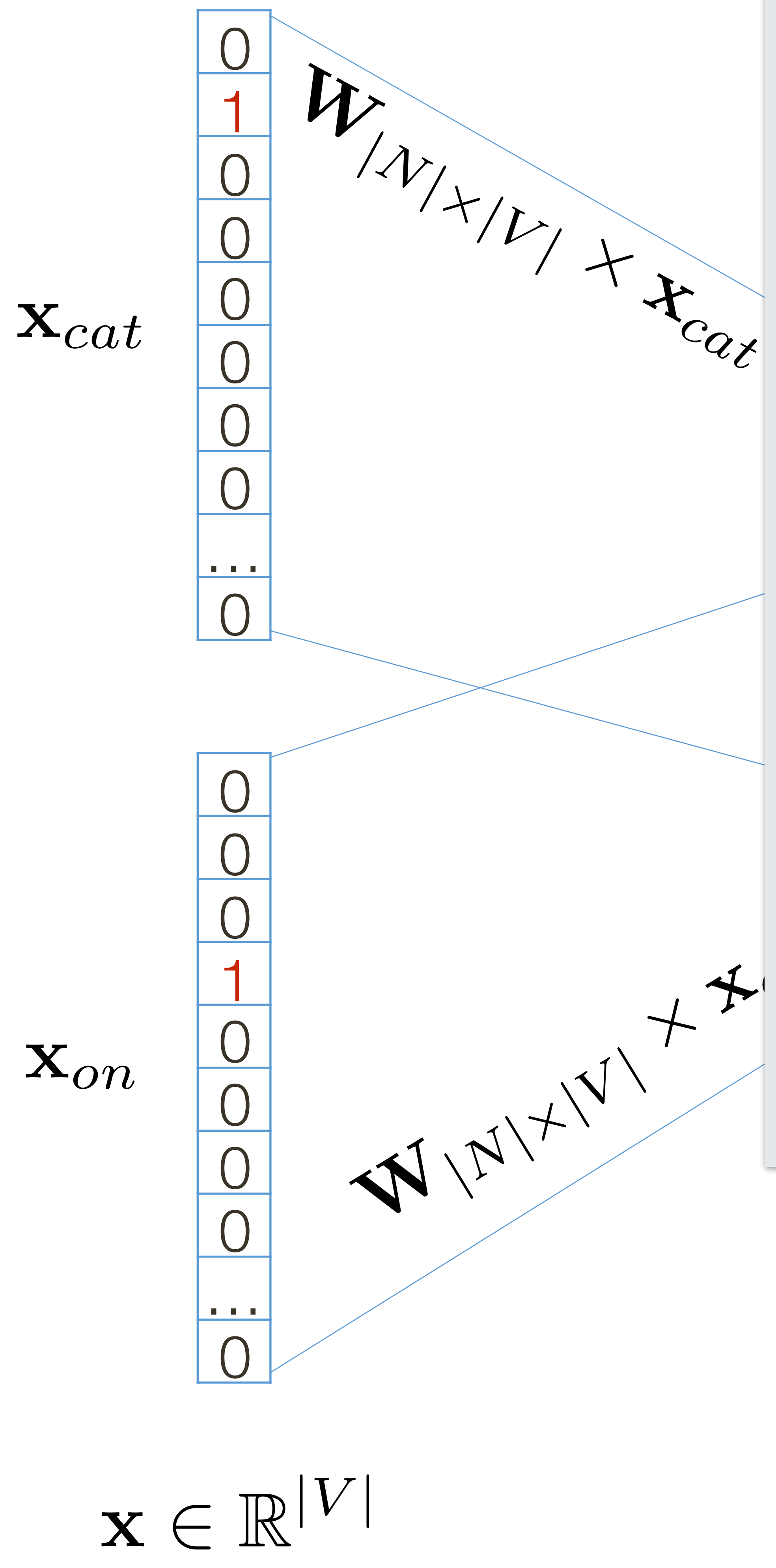
$$\begin{bmatrix} 0.1 & 2.4 & 1.6 & 1.8 & 0.5 & 0.9 & \dots & \dots & \dots & 3.2 \\ 0.5 & 2.6 & 1.4 & 2.9 & 1.5 & 3.6 & \dots & \dots & \dots & 6.1 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0.6 & 1.8 & 2.7 & 1.9 & 2.4 & 2.0 & \dots & \dots & \dots & 1.2 \end{bmatrix} \times \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \mathbf{2.4} \\ \mathbf{2.6} \\ \dots \\ \dots \\ \mathbf{1.8} \end{bmatrix}$$

*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

[Mikolov et al., 2013]

Input layer



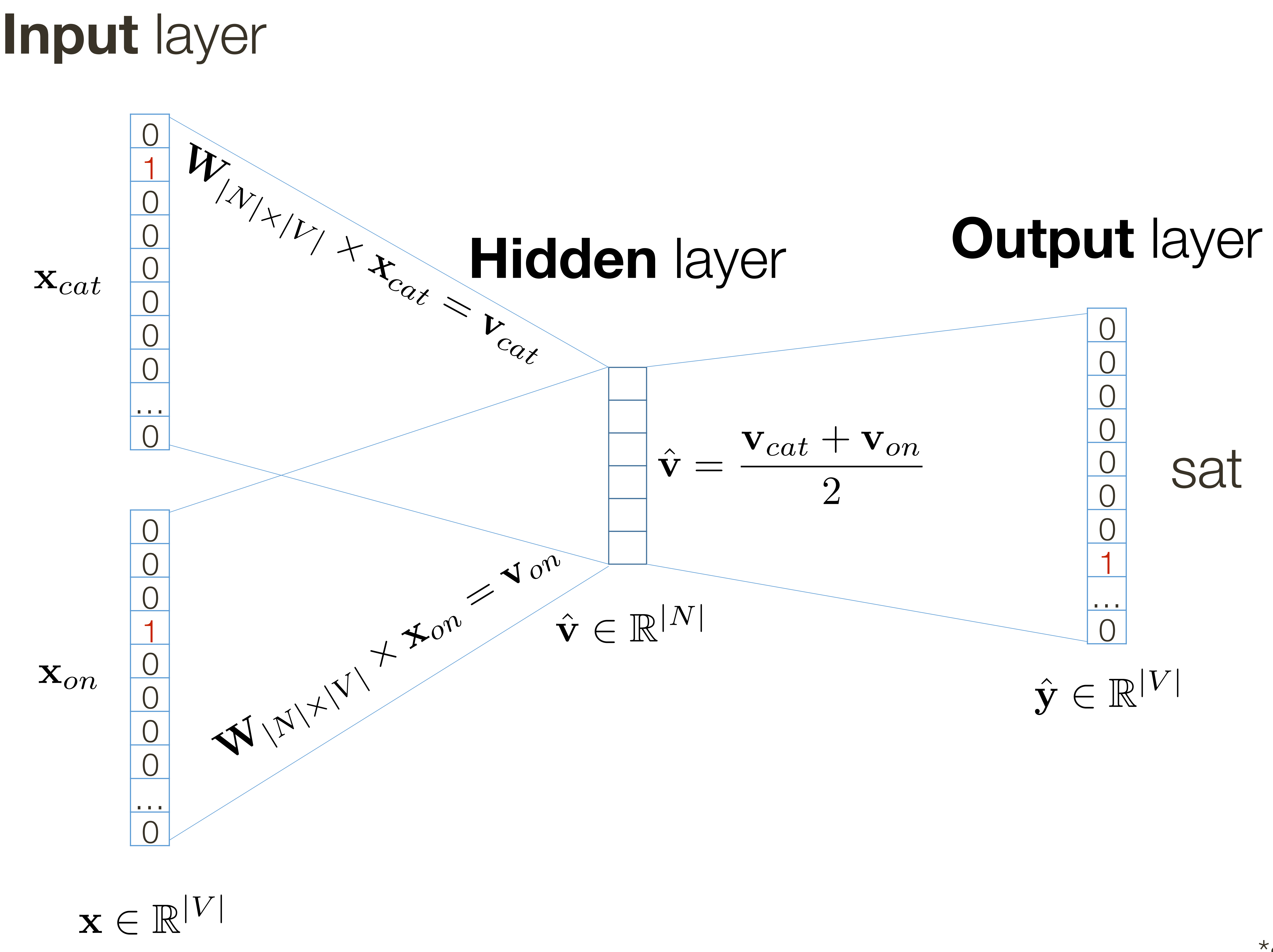
$$\mathbf{W}^T_{|V|\times|N|} \times \mathbf{x}_{on} = \mathbf{v}_{on}$$

$$\begin{bmatrix} 0.1 & 2.4 & 1.6 & 1.8 & 0.5 & 0.9 & \ldots & \ldots & \ldots & 3.2 \\ 0.5 & 2.6 & 1.4 & 2.9 & 1.5 & 3.6 & \ldots & \ldots & \ldots & 6.1 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0.6 & 1.8 & 2.7 & 1.9 & 2.4 & 2.0 & \ldots & \ldots & \ldots & 1.2 \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \mathbf{1.8} \\ \mathbf{2.9} \\ \ldots \\ \ldots \\ \mathbf{1.9} \end{bmatrix}$$

*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

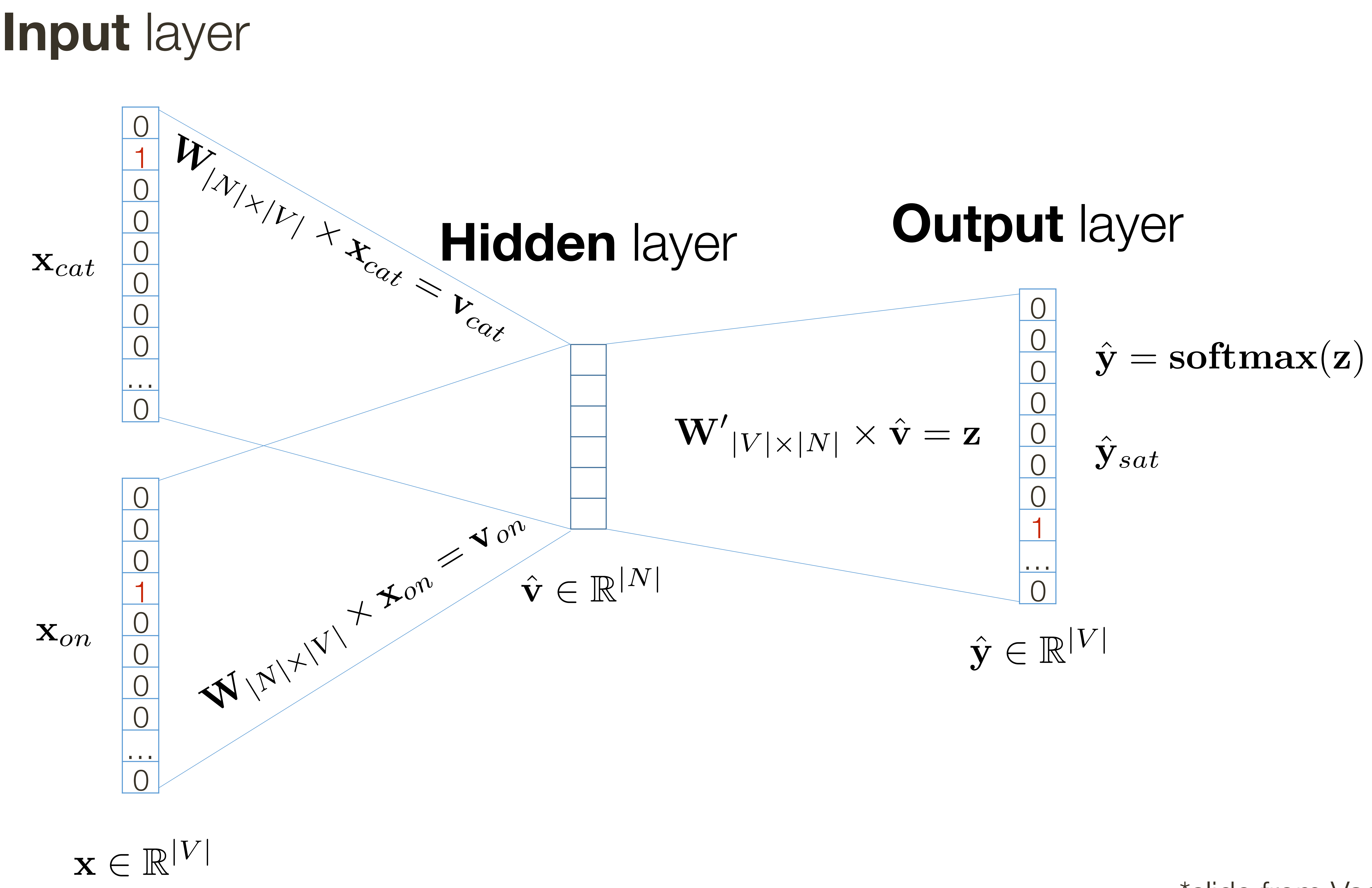
[Mikolov et al., 2013]



*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

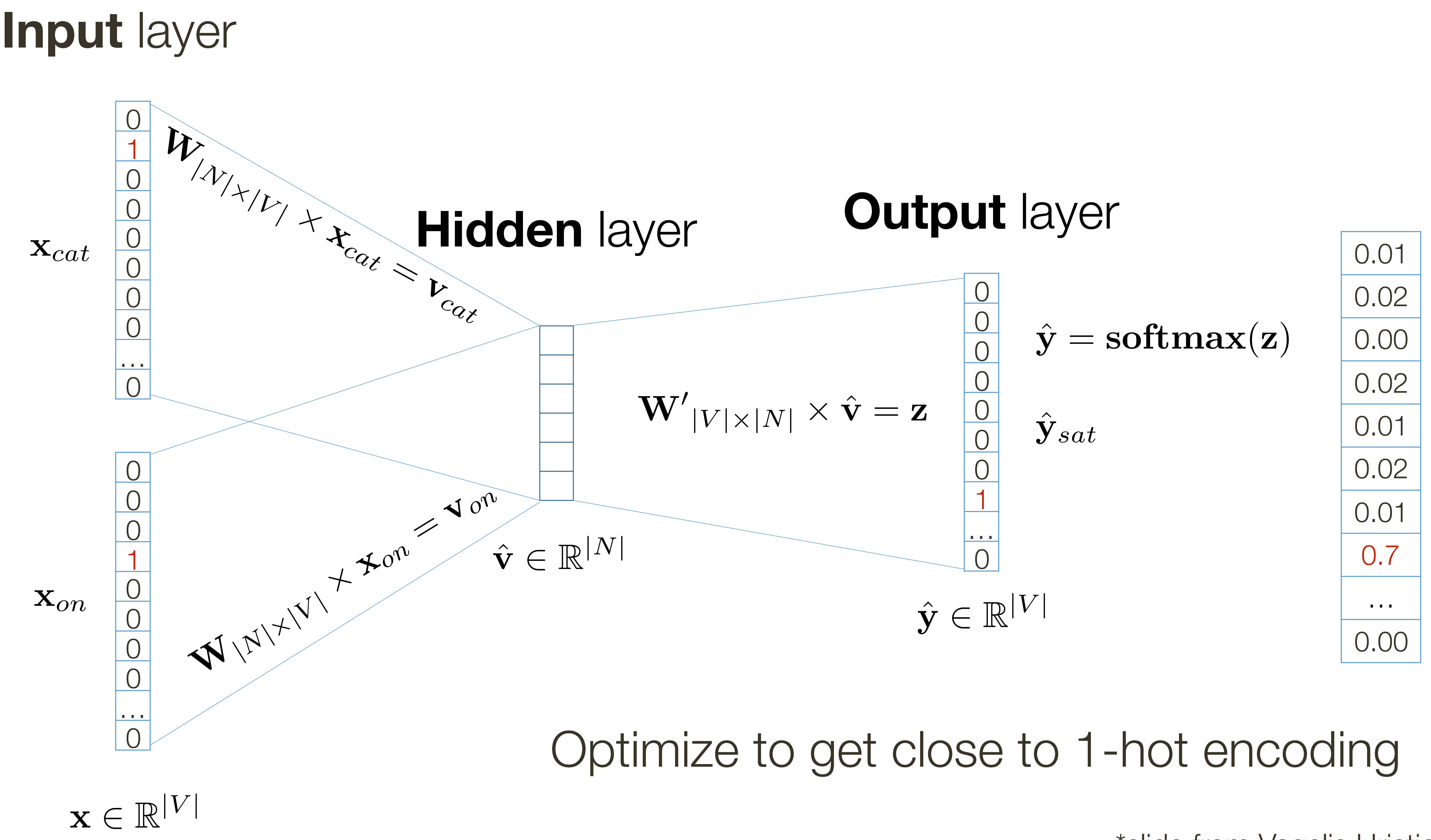
[Mikolov et al., 2013]



*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

[Mikolov et al., 2013]

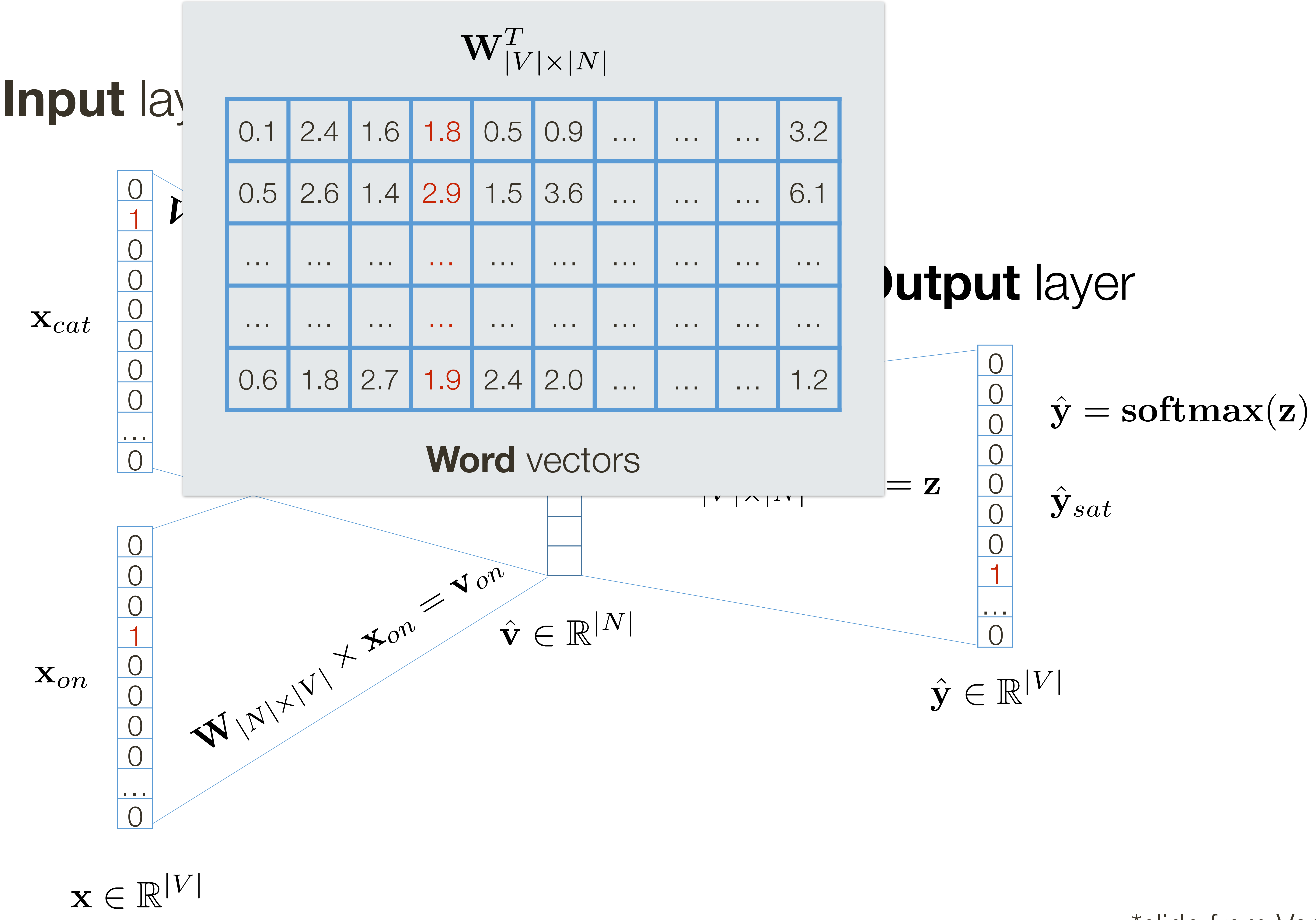


Optimize to get close to 1-hot encoding

*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

[Mikolov et al., 2013]



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

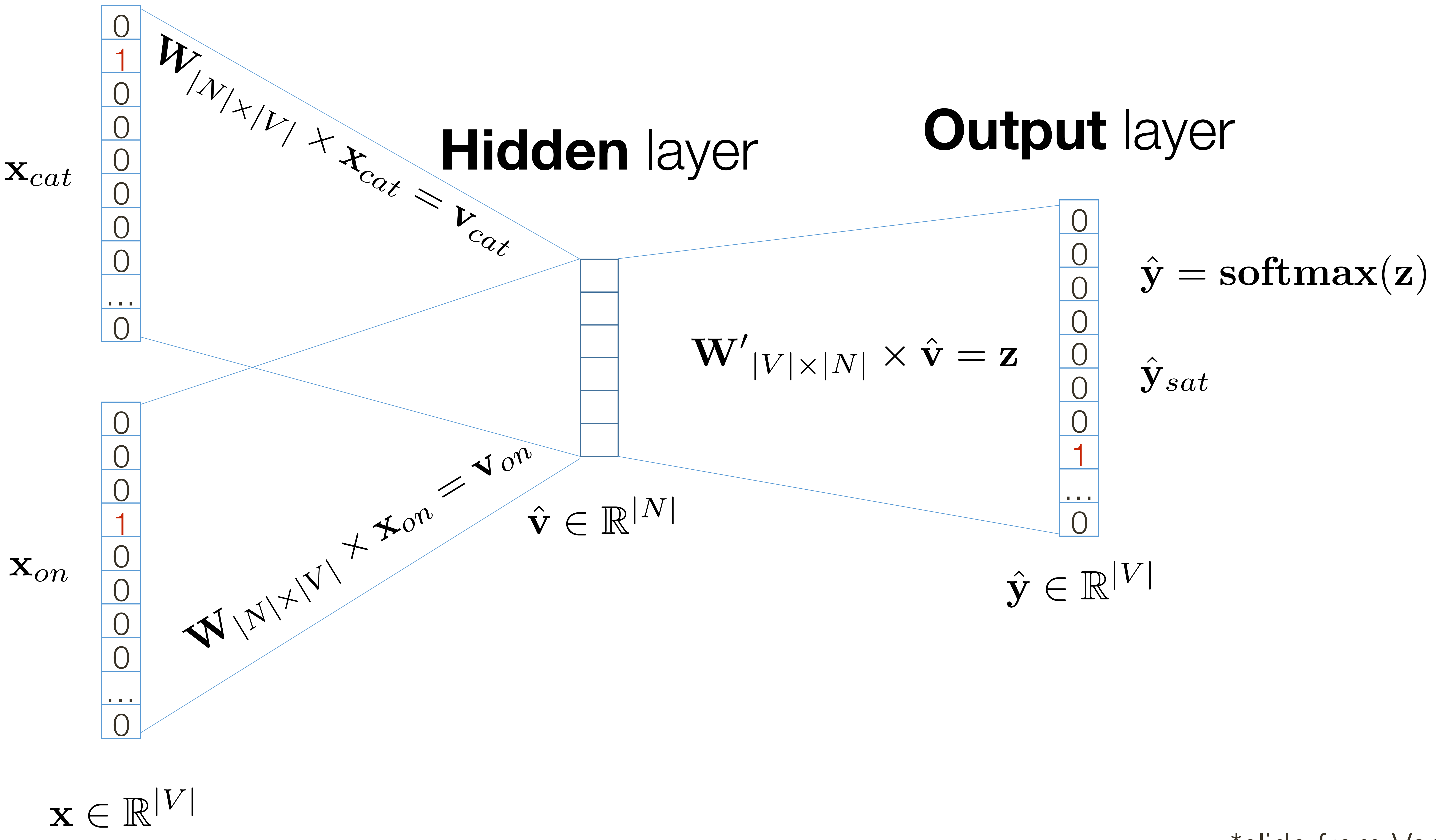
Word vectors

*slide from Vagelis Hristidis

CBOW: Interesting Observation

[Mikolov et al., 2013]

There are two representations for same word!



*slide from Vagelis Hristidis

CBOW: Interesting Observation

[Mikolov et al., 2013]

Another way to look at it: Maximize similarity between context word representation and the word representation itself

$$p(w|c) = \frac{\exp\left[\left(\sum_c \mathbf{W}\mathbf{x}_c\right)^T \left(\mathbf{W}\mathbf{x}_w\right)\right]}{\sum_i^{|V|} \exp\left[\left(\mathbf{W}\mathbf{x}_i\right)^T \left(\mathbf{W}\mathbf{x}_w\right)\right]}$$

CBOW: Interesting Observation

[Mikolov et al., 2013]

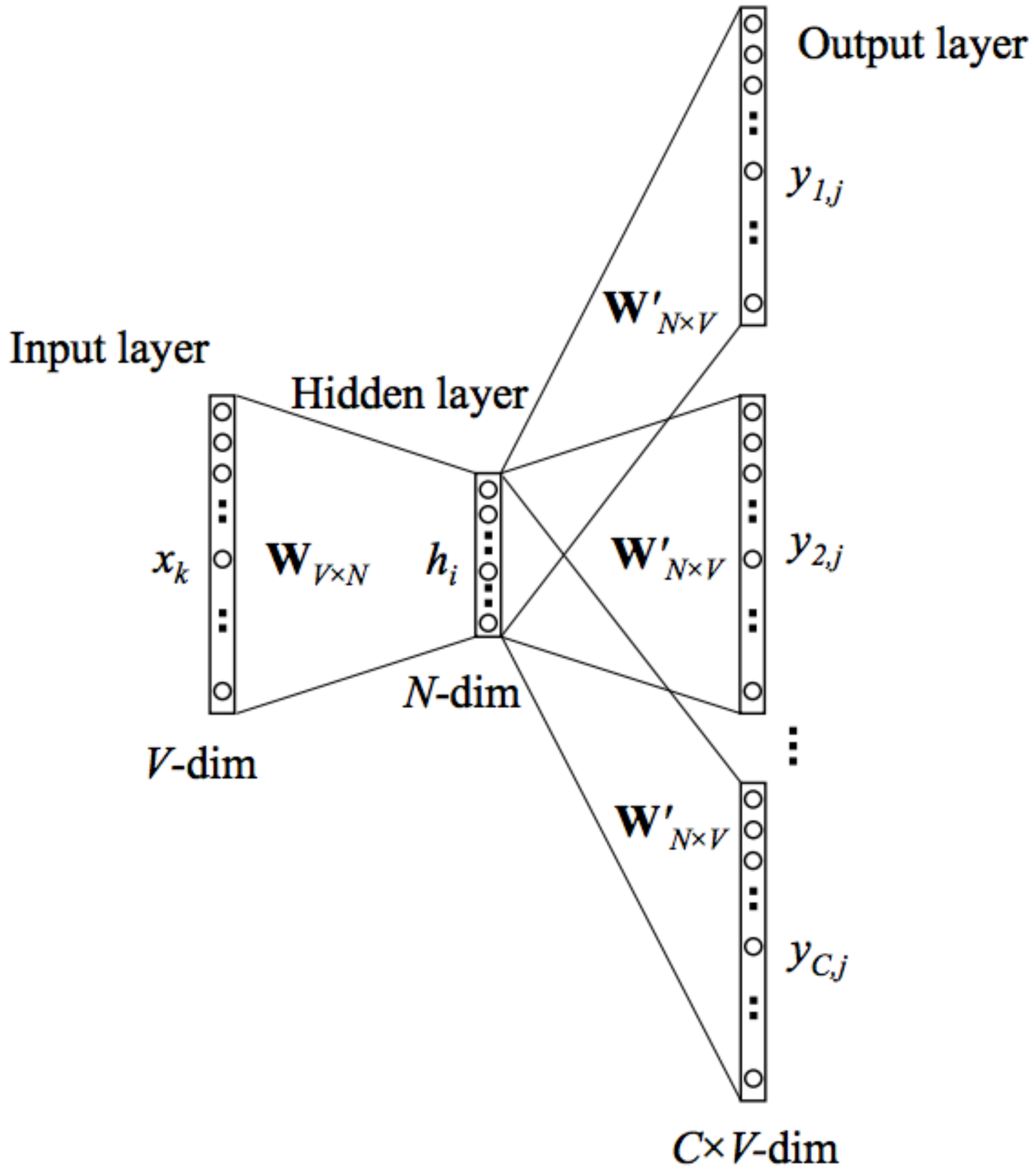
Another way to look at it: Maximize similarity between context word representation and the word representation itself

$$J(\mathbf{W}) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{-m\leq j\leq m;j\neq 0}\log p(w_{t+j}|w_t)$$

$$p(w_{t+j}|w_t) = \frac{\exp(\mathbf{w}_{t+j}^T\mathbf{w}_t)}{\sum_{i=1}^{|V|}\exp(\mathbf{w}_i^T\mathbf{w}_t)}$$

Skip-Gram Model

[Mikolov et al., 2013]



Comparison

[Mikolov et al., 2013]

– **CBOW** is not great for rare words and typically needs less data to train

– **Skip-gram** better for rate words and needs more data to train the model

Model	Vector Dimensionality	Training words	Accuracy [%]		
			Semantic	Syntactic	Total
Collobert-Weston NNLM	50	660M	9.3	12.3	11.0
Turian NNLM	50	37M	1.4	2.6	2.1
Turian NNLM	200	37M	1.4	2.2	1.8
Mnih NNLM	50	37M	1.8	9.1	5.8
Mnih NNLM	100	37M	3.3	13.2	8.8
Mikolov RNNLM	80	320M	4.9	18.4	12.7
Mikolov RNNLM	640	320M	8.6	36.5	24.6
Huang NNLM	50	990M	13.3	11.6	12.3
Our NNLM	20	6B	12.9	26.4	20.3
Our NNLM	50	6B	27.9	55.8	43.2
Our NNLM	100	6B	34.2	**64.5**	50.8
CBOW	300	783M	15.5	53.1	36.1
Skip-gram	300	783M	**50.0**	55.9	**53.3**

Interesting Results: **Word Analogies**

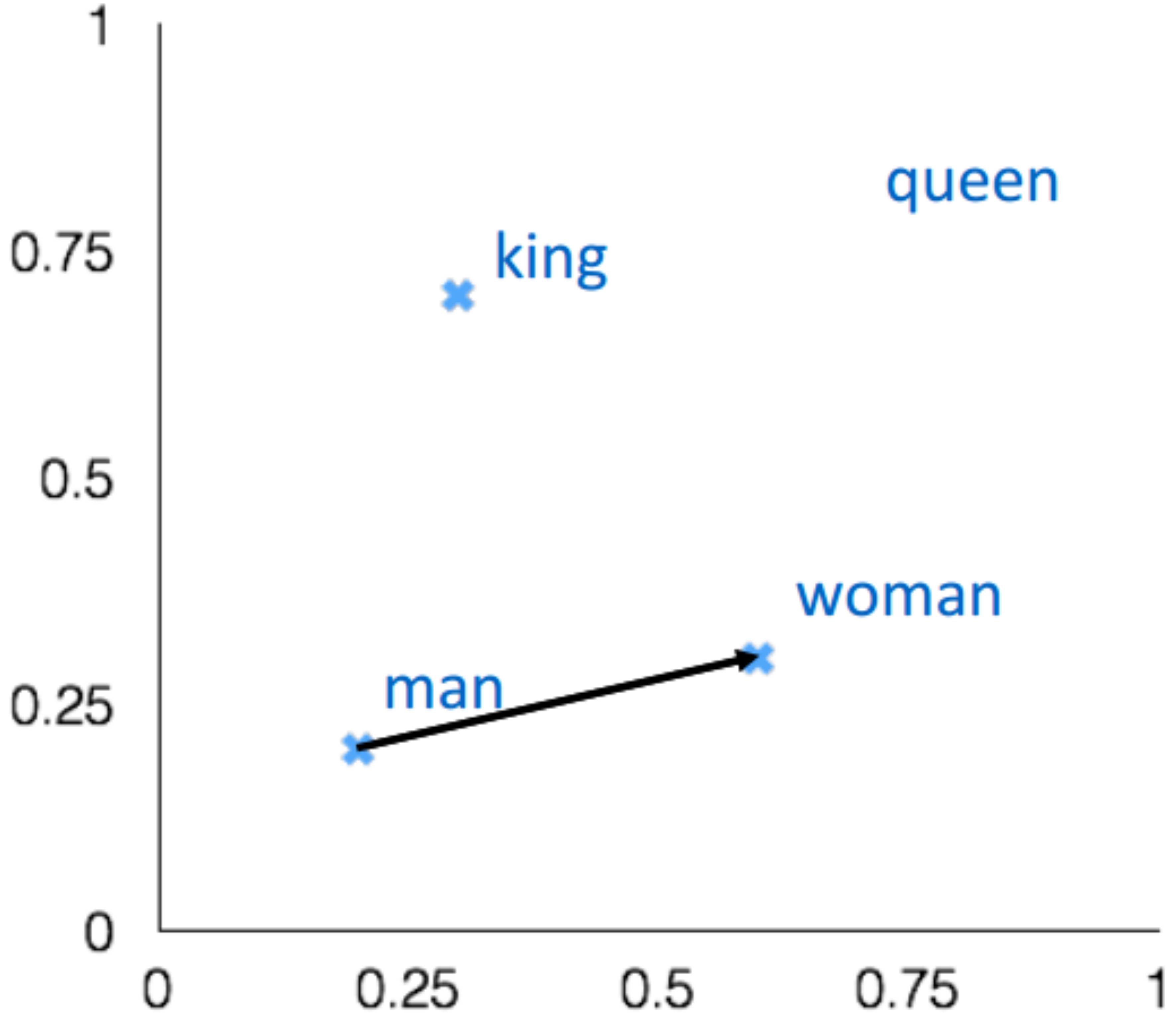
Test for linear relationships, examined by Mikolov et al. (2014)

a:b :: c:?

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

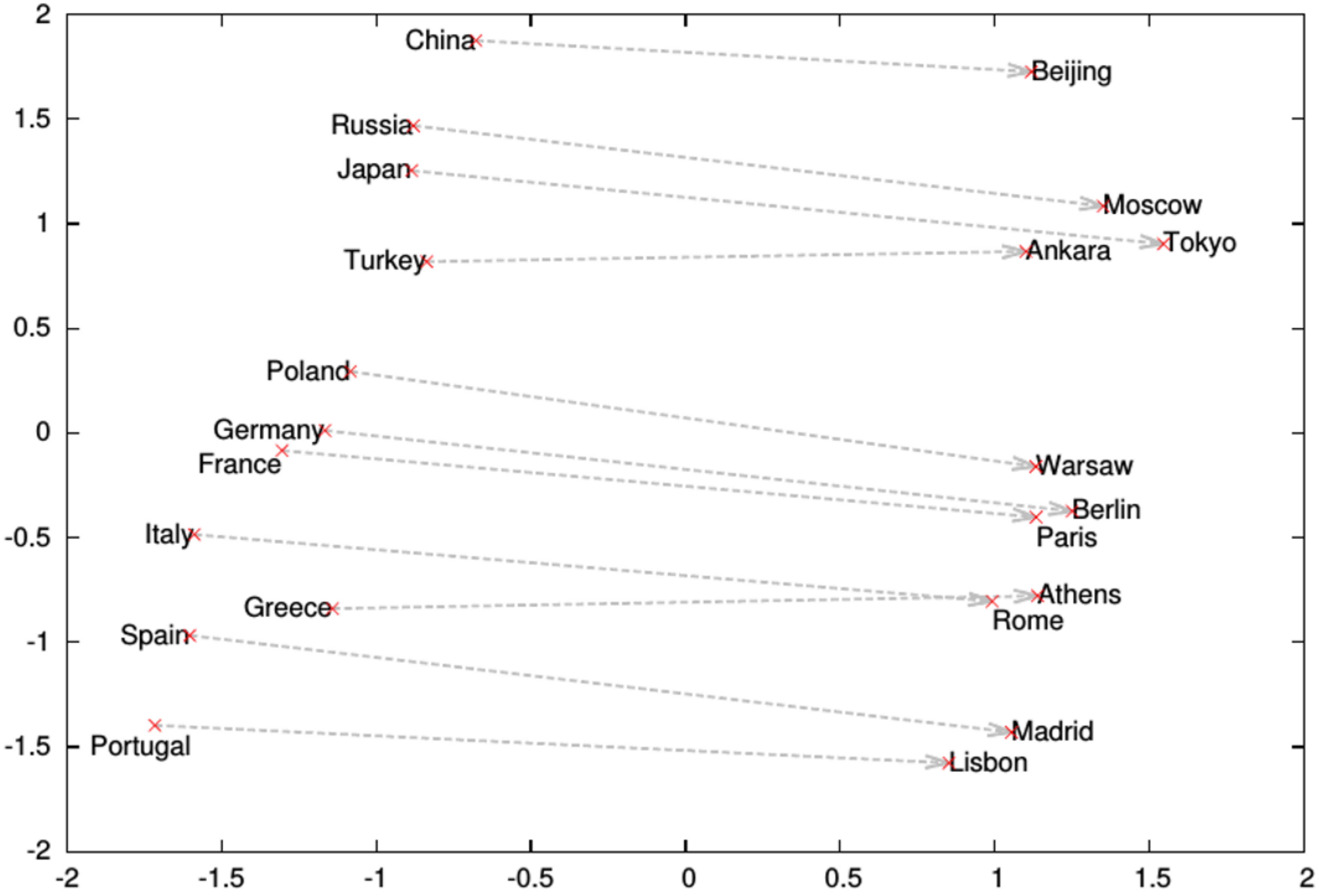
man:woman :: king:?

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



Interesting Results: **Word Analogies**

[Mikolov et al., 2013]



Language Models

Model the **probability of a sentence**; ideally be able to sample plausible sentences

Why is this useful?

$$\underset{wordsequence}{\arg\max}\ P(wordsequence \mid acoustics) =$$

$$\underset{wordsequence}{\arg\max}\ \frac{P(acoustics \mid wordsequence) \times \boxed{P(wordsequence)}}{P(acoustics)}$$

$$\underset{wordsequence}{\arg\max}\ P(acoustics \mid wordsequence) \times \boxed{P(wordsequence)}$$

Simple **Language Models**: N-Grams

Given a word sequence: $w_{1:n} = [w_1, w_2, ..., w_n]$

We want to estimate $p(w_{1:n})$

Using **Chain Rule** of probabilities:

$$p(w_{1:n}) = p(w_1)p(w_2|w_1)p(w_3|w_1, w_2) \cdots p(w_n|w_{1:n-1})$$

Bi-gram Approximation:

$$p(w_{1:n}) = \prod_{k=1}^{n} p(w_k|w_{k-1})$$

N-gram Approximation:

$$p(w_{1:n}) = \prod_{k=1}^{n} p(w_k|w_{k-N+1:k-1})$$

* Slides from Louis-Philippe Morency

Estimating **Probabilities**

N-gram conditional probabilities can be estimated based on raw concurrence counts in the observed sequences

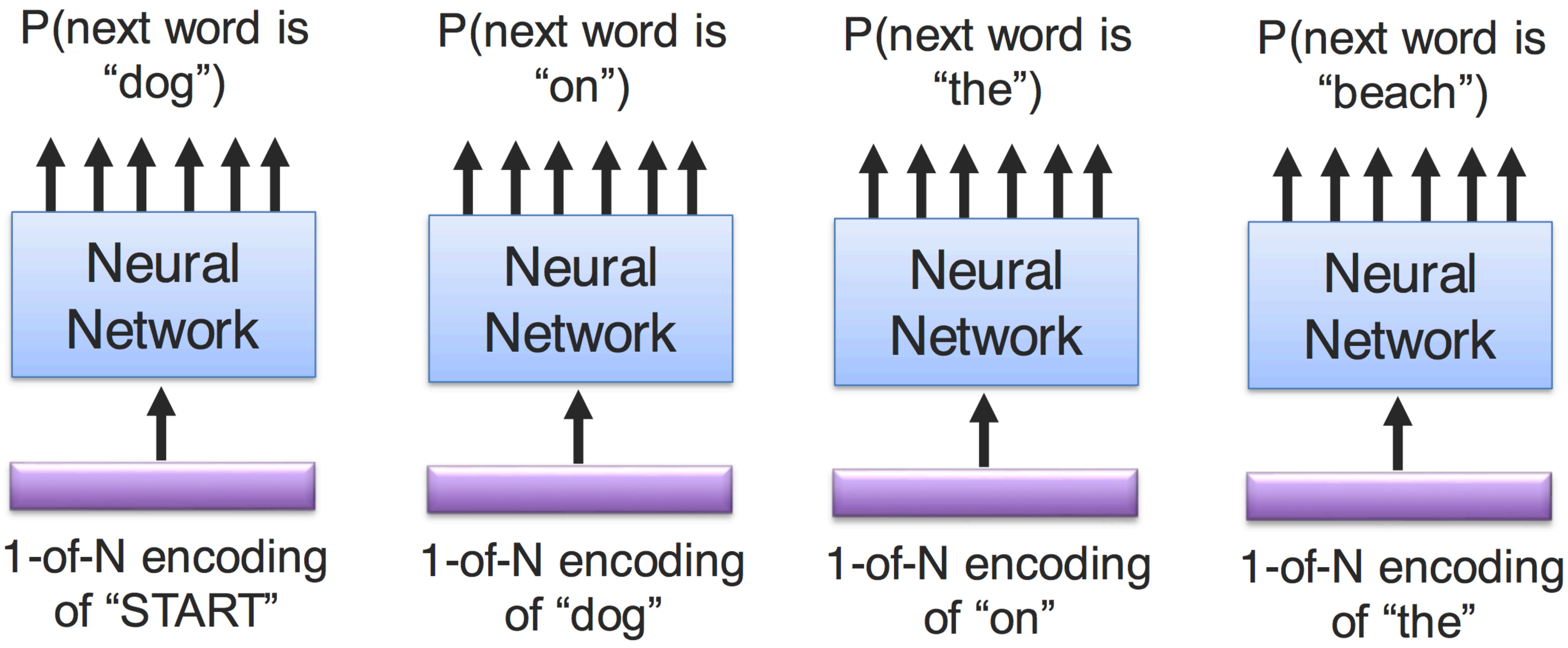
Bi-gram:

$$p(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})}$$

N-gram:

$$p(w_n|w_{n-N-1:n-1}) = \frac{C(w_{n-N-1:n-1}w_n)}{C(w_{n-N-1:n-1})}$$

Neural-based Unigram Language Mode

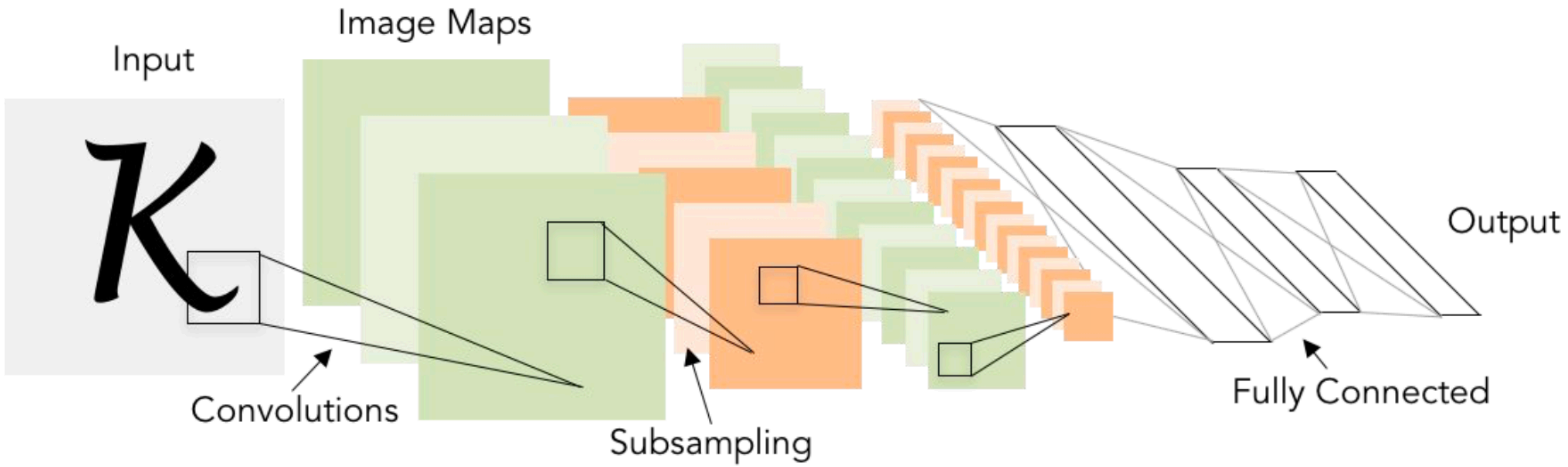


Problem: Does not model sequential information (too local)

We need sequence modeling!

* Slides from Louis-Philippe Morency

Sequence Modeling



Why Model **Sequences**?

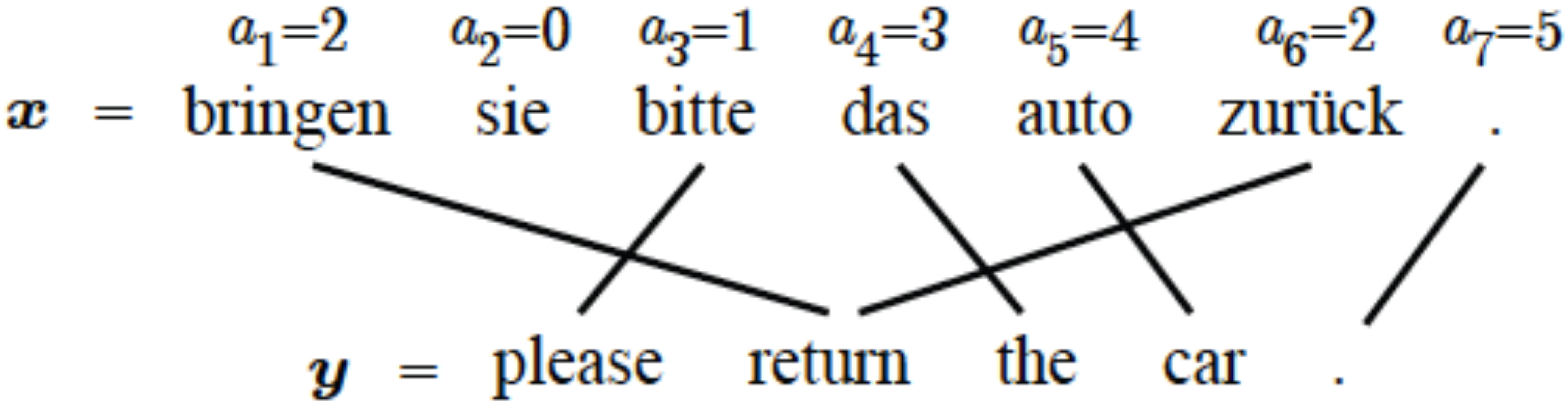
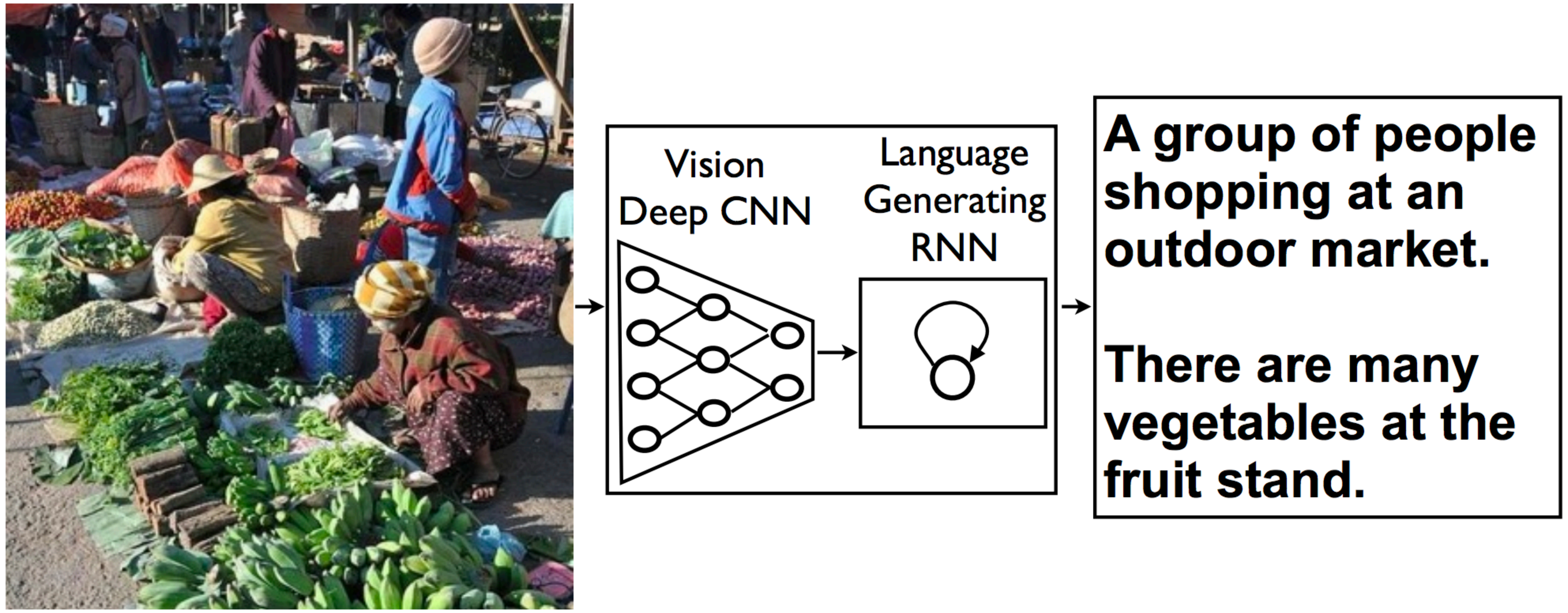


Image Credit: Alex Graves and Kevin Gimpel

* slide from Dhruv Batra

Multi-modal tasks

[Vinyals *et al.*, 2015]



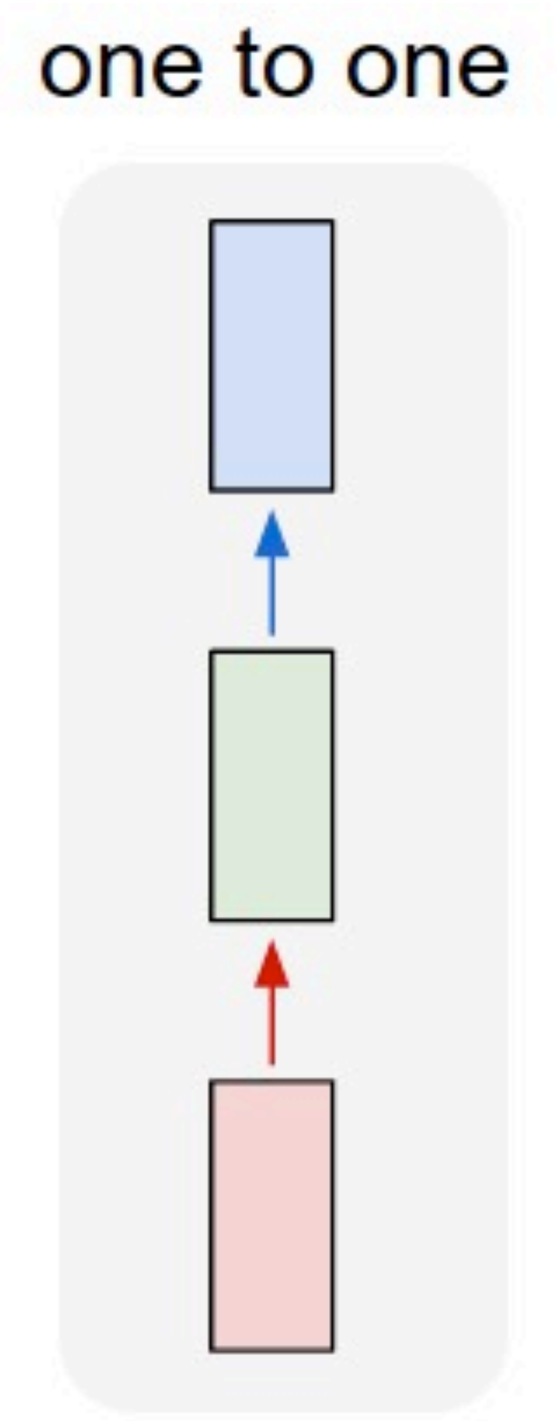
Sequences where you don't expect them …

Classify images by taking a series of "glimpses"

[Gregor et al., ICML 2015]

[Mnih et al., ICLR 2015]

Sequences in Inputs or Outputs?



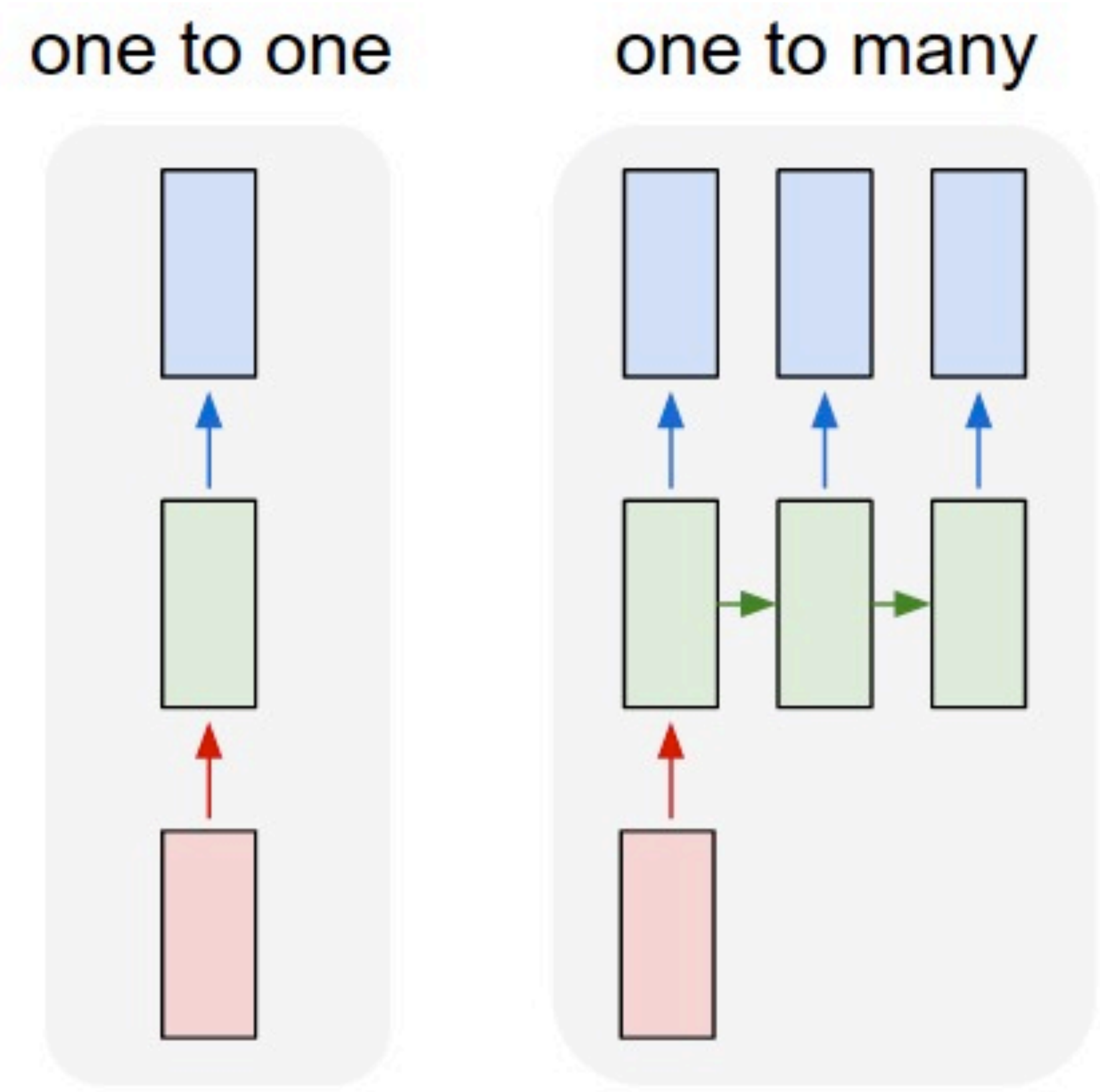
Input: No sequence

Output: No seq.

Example:
"standard"
classification /
regression problems

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Sequences in Inputs or Outputs?

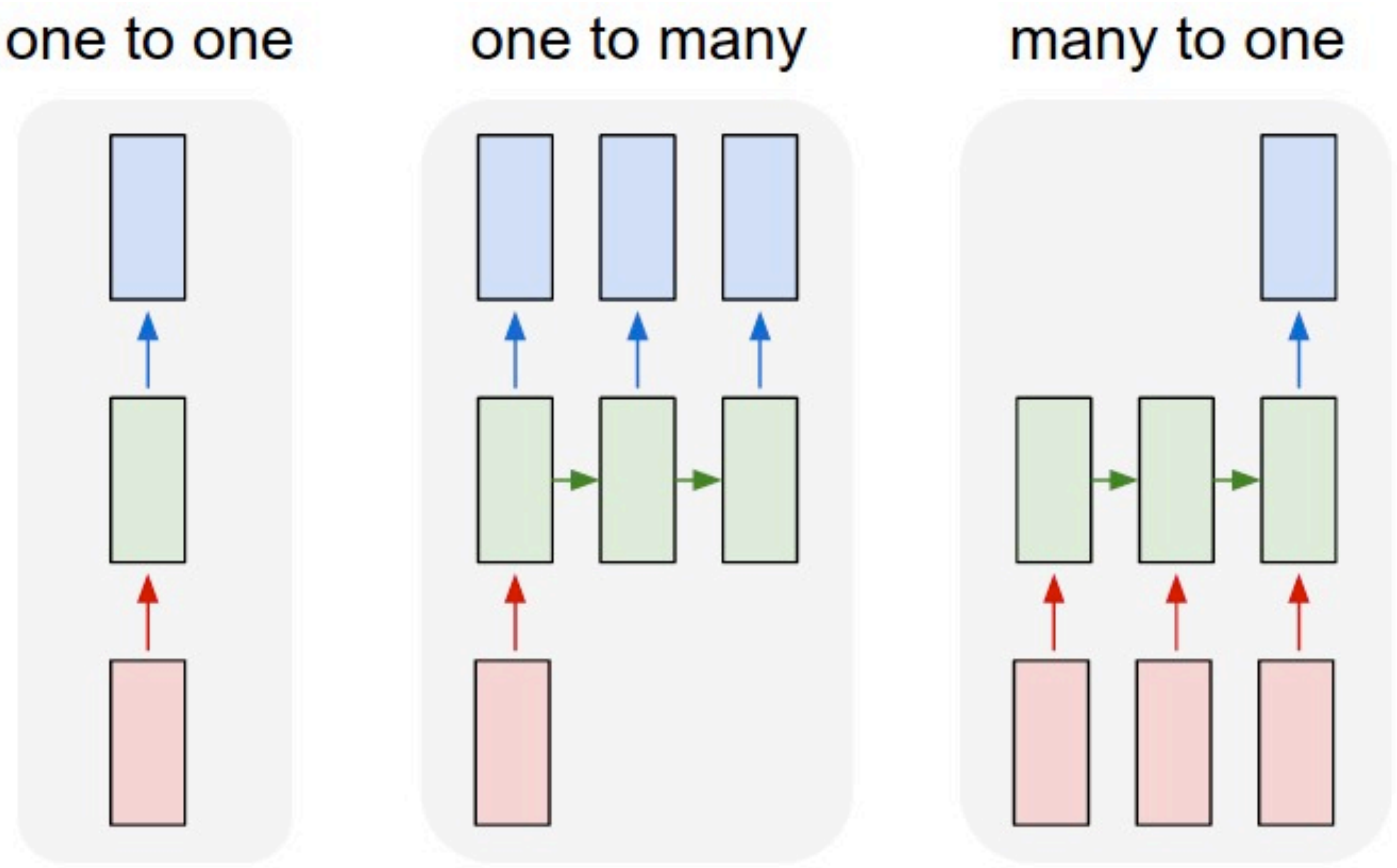


Input: No sequence
Output: No seq.
Example: "standard" classification / regression problems

Input: No sequence
Output: Sequence
Example: Im2Caption

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Sequences in Inputs or Outputs?



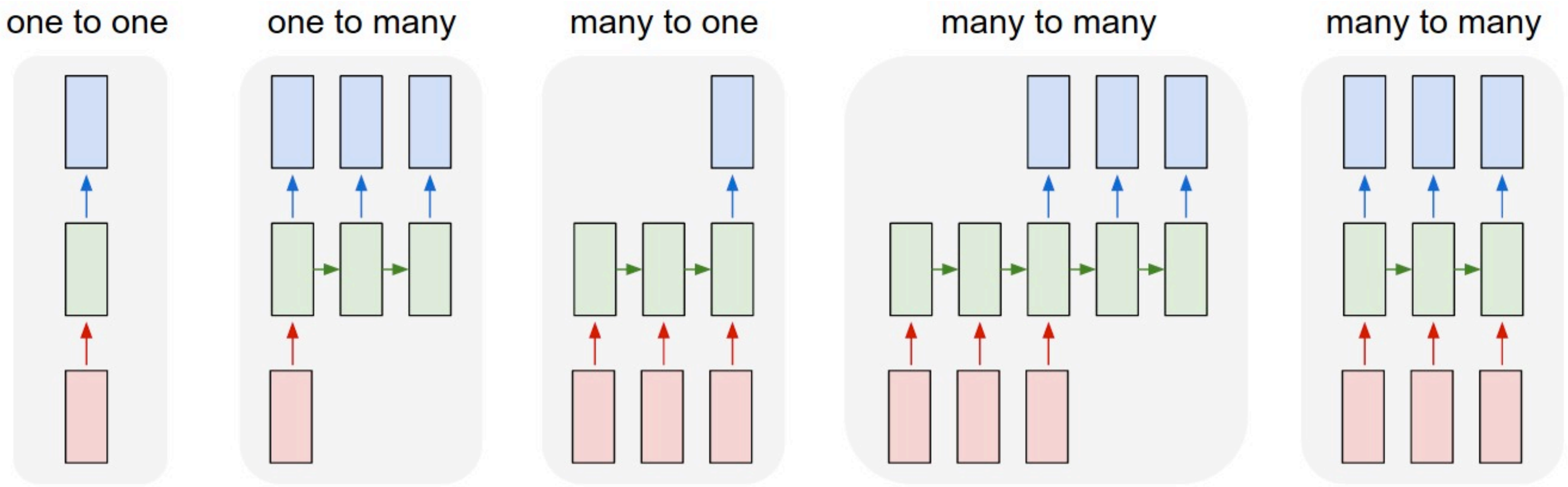
Input: No sequence
Output: No seq.
Example: "standard" classification / regression problems

Input: No sequence
Output: Sequence
Example: Im2Caption

Input: Sequence
Output: No seq.
Example: sentence classification, multiple-choice question answering

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Sequences in Inputs or Outputs?



Input: No sequence
Output: No seq.
Example: "standard" classification / regression problems

Input: No sequence
Output: Sequence
Example: Im2Caption

Input: Sequence
Output: No seq.
Example: sentence classification, multiple-choice question answering

Input: Sequence
Output: Sequence
Example: machine translation, video captioning, open-ended question answering, video question answering

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Key Conceptual Ideas

Parameter Sharing

— in computational graphs = adding gradients

“**Unrolling**”

— in computational graphs with parameter sharing

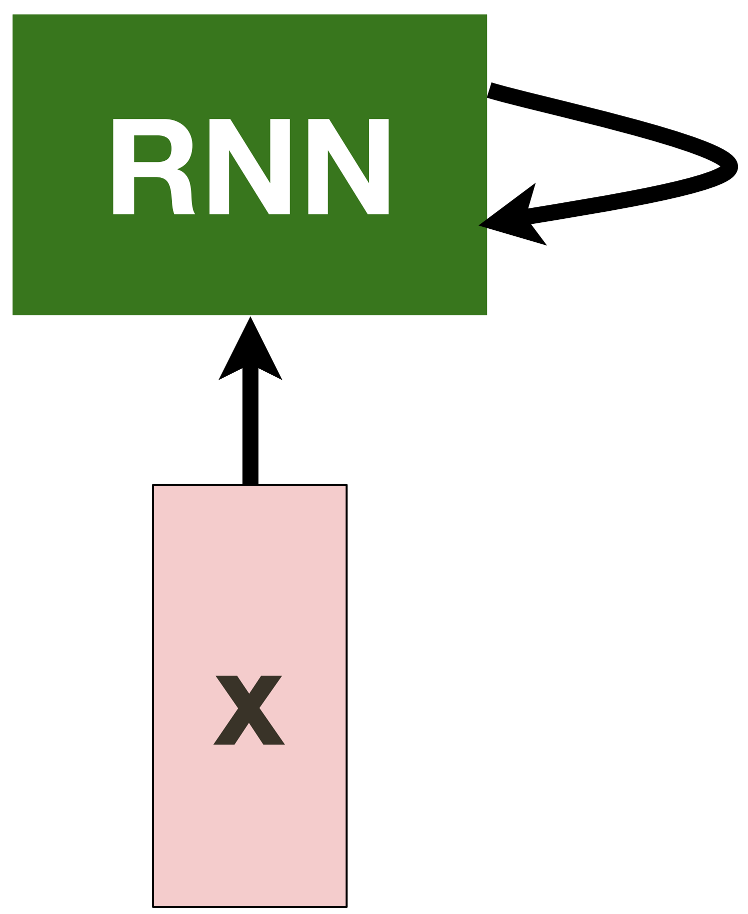
Parameter Sharing + “Unrolling”

— Allows modeling **arbitrary length sequences**!

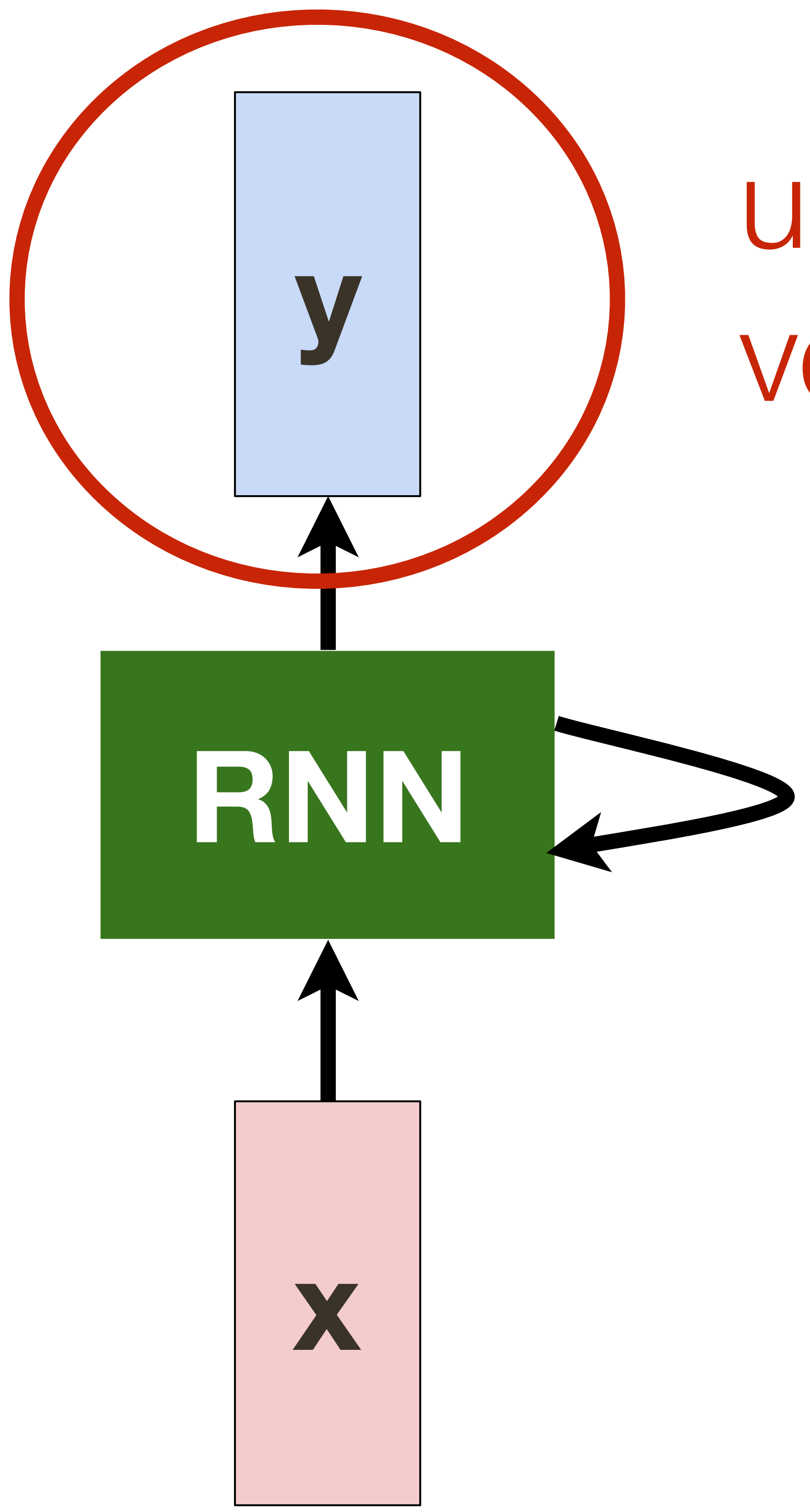
— Keeps number of parameters in check

* slide from Dhruv Batra

Recurrent Neural Network



Recurrent Neural Network



usually want to predict a vector at some time steps

Recurrent Neural Network

We can process a sequence of vectors **x** by applying a **recurrence formula** at every time step:

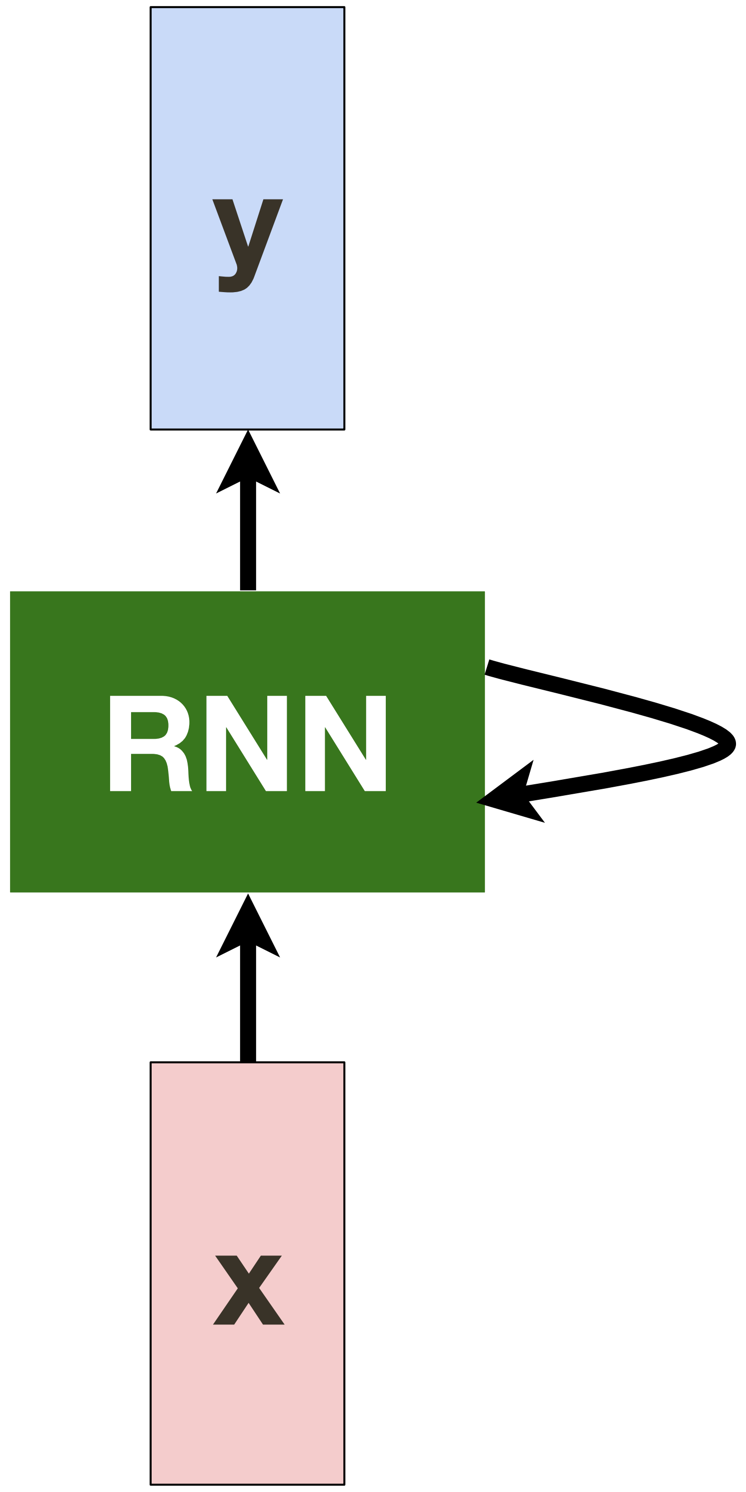
new **state**

old **state**

$$h_t = f_W(h_{t-1}, x_t)$$

some **function** with parameters W

input vector at some time step



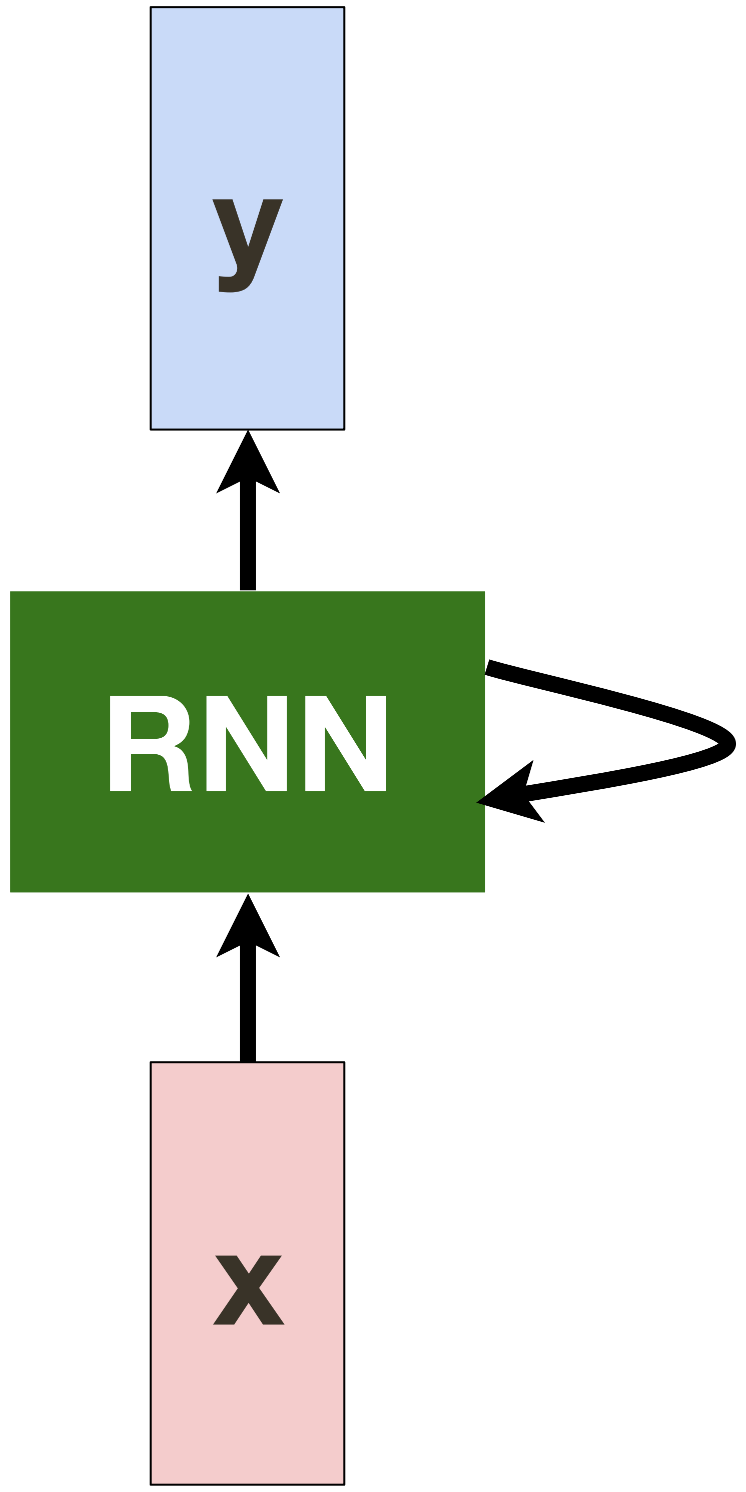
* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Recurrent Neural Network

We can process a sequence of vectors **x** by applying a **recurrence formula** at every time step:

$$h_t = f_W(h_{t-1}, x_t)$$

Note: the same function and the same set of parameters are used at every time step



* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

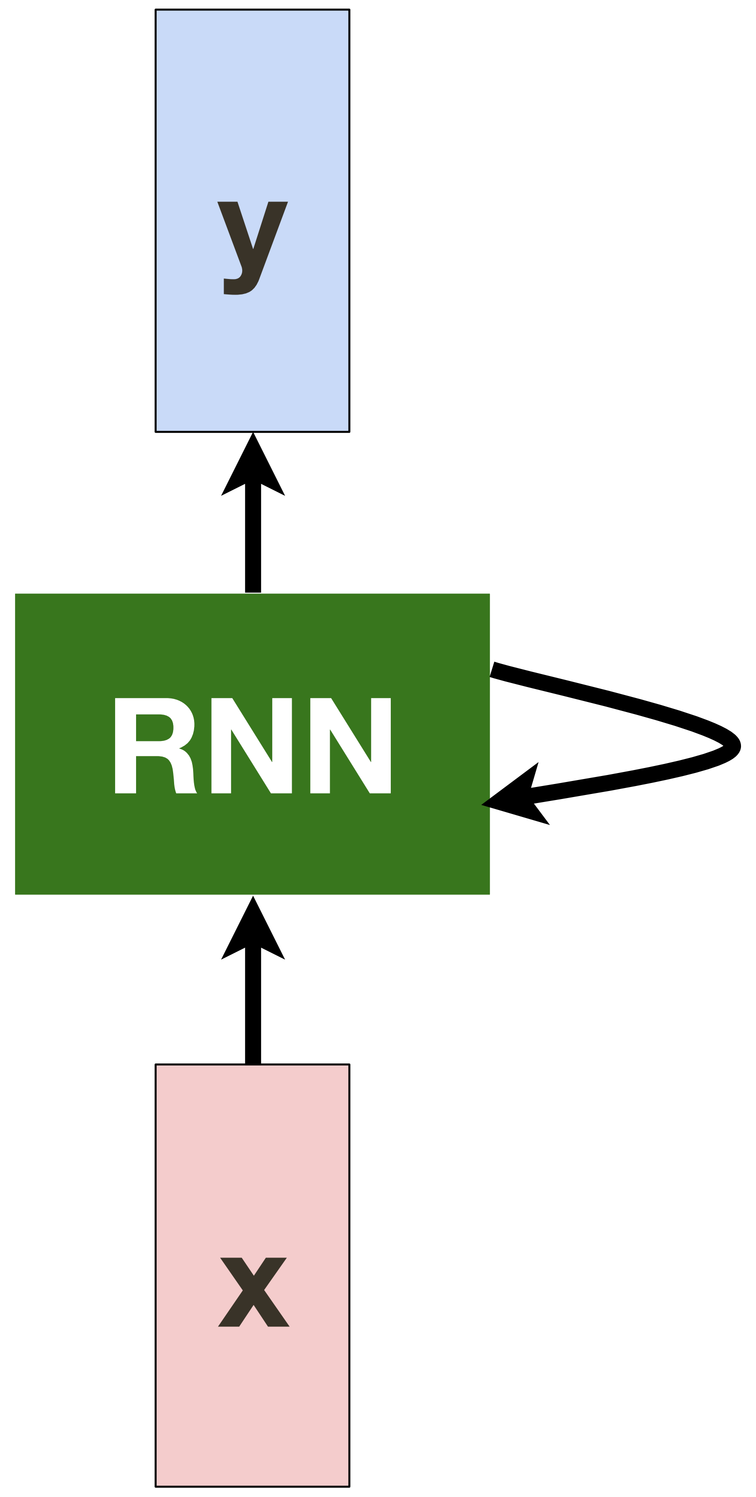
(Vanilla) **Recurrent** Neural Network

$$y_t = W_{hy} h_t + b_y$$

$$h_t = f_W(h_{t-1}, x_t)$$

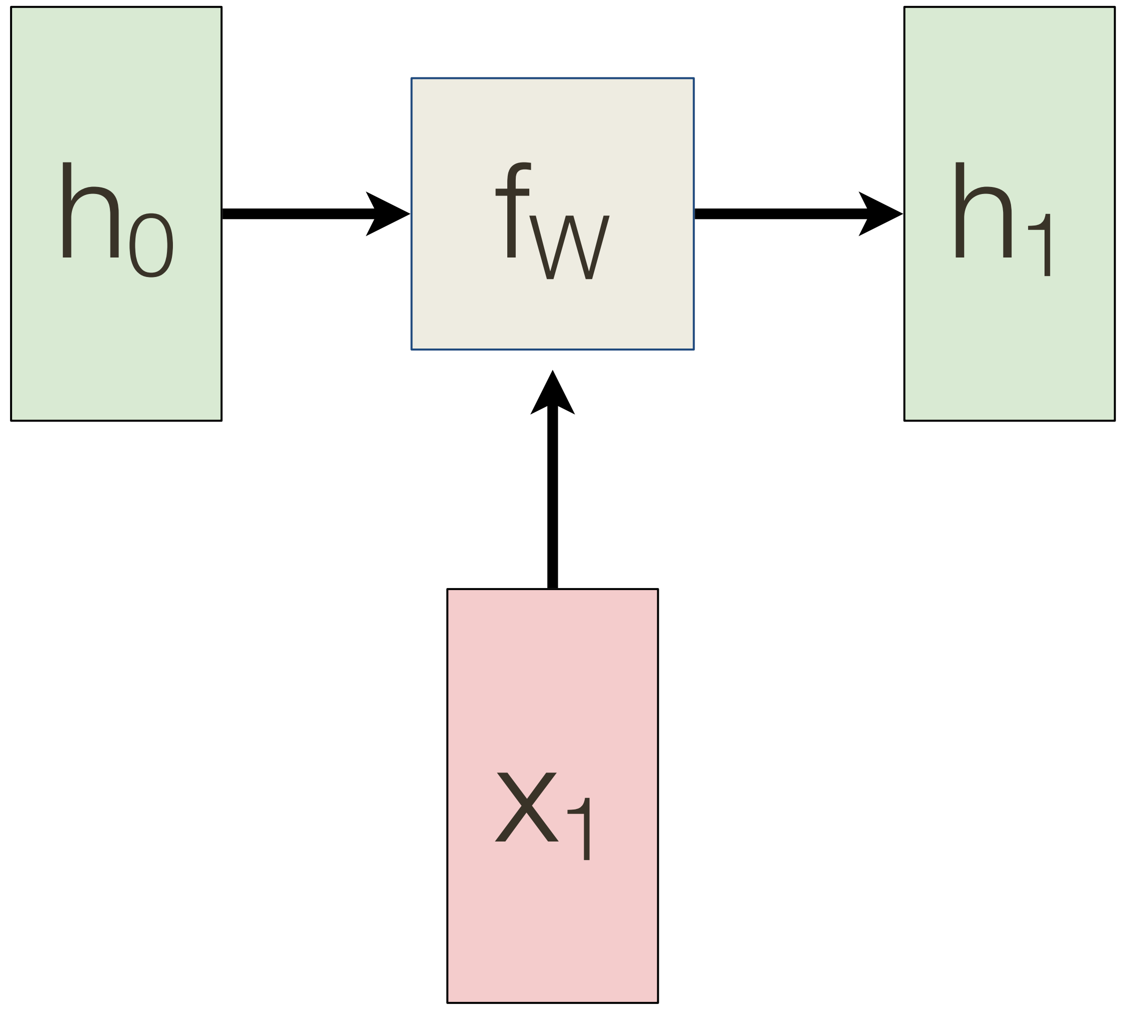
$$\downarrow$$

$$h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t + b_h)$$



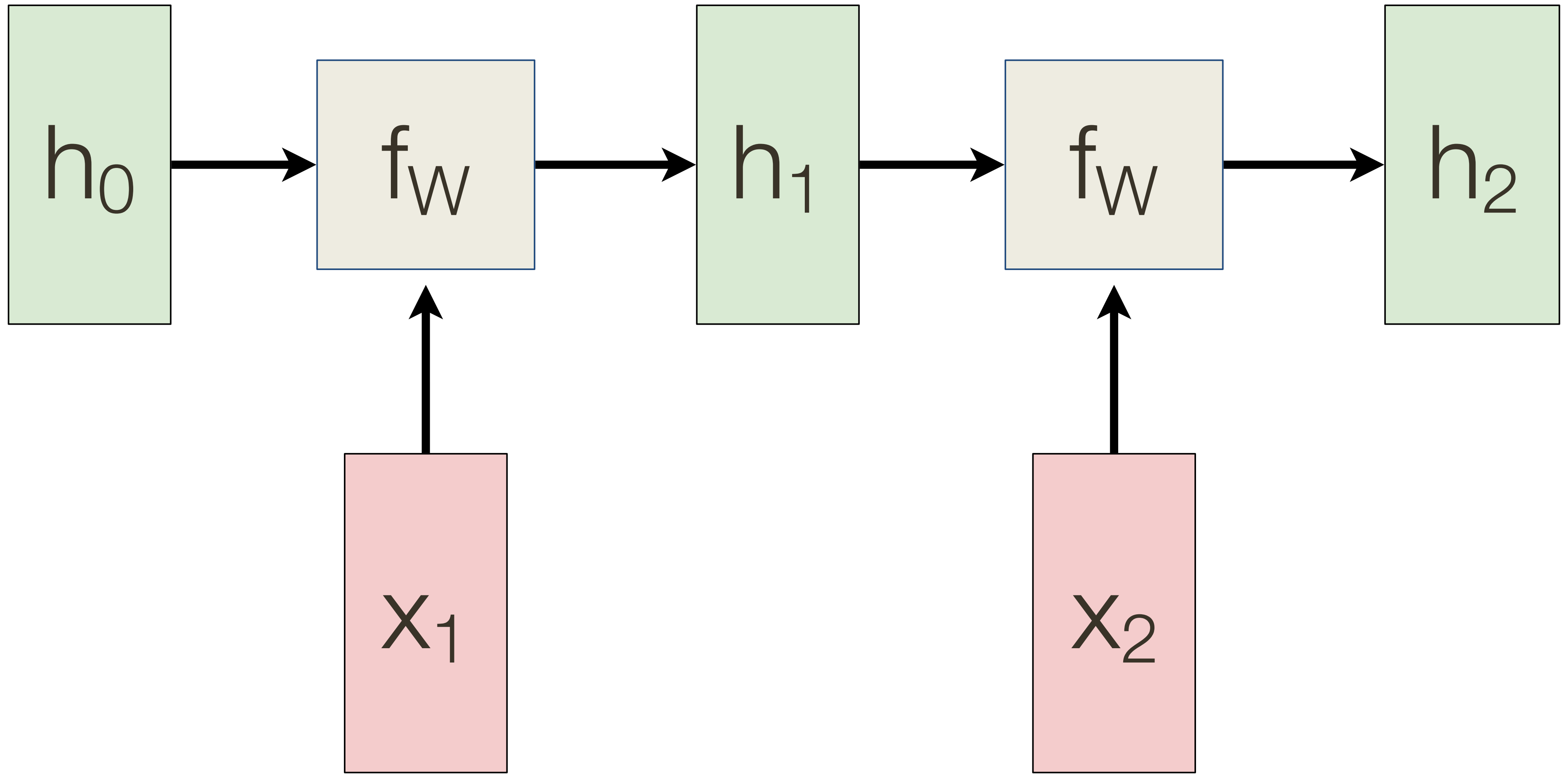
* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

RNN **Computational Graph**



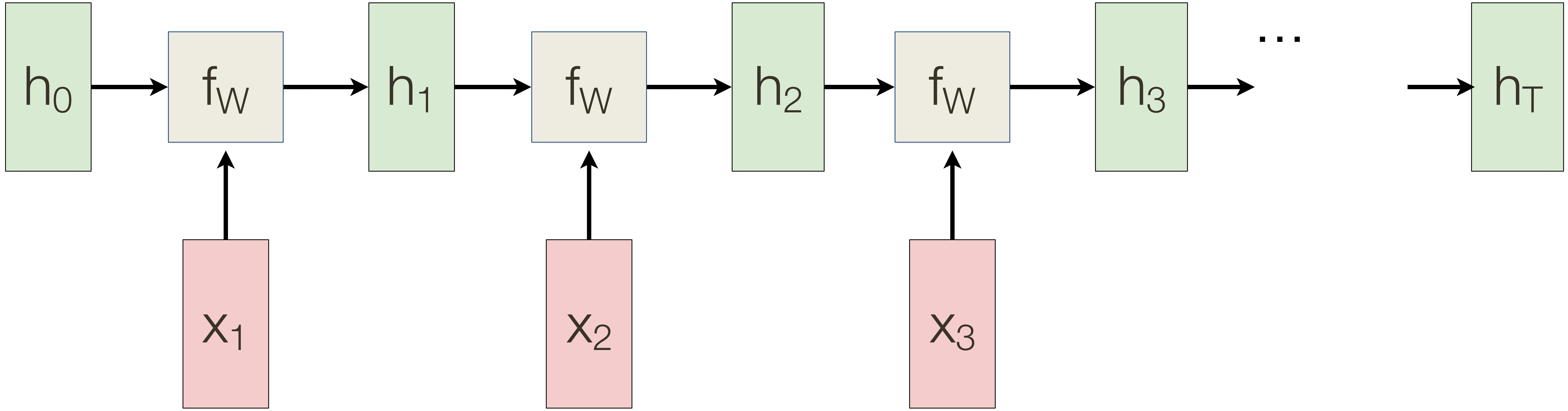
* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

RNN **Computational Graph**



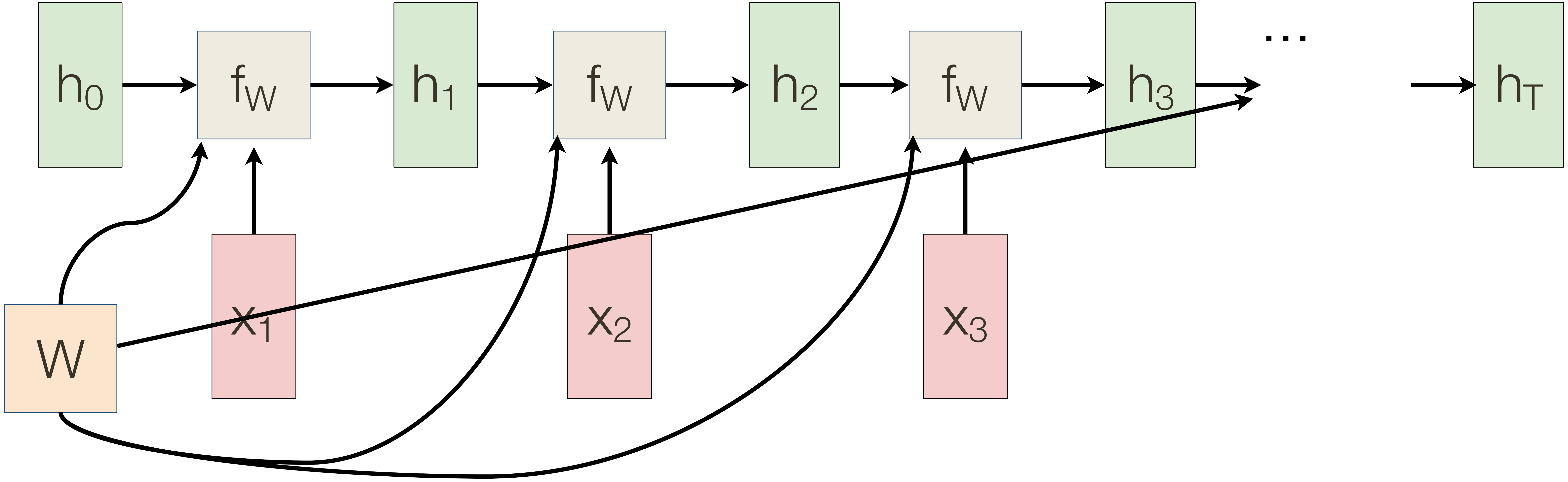
* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

RNN **Computational Graph**

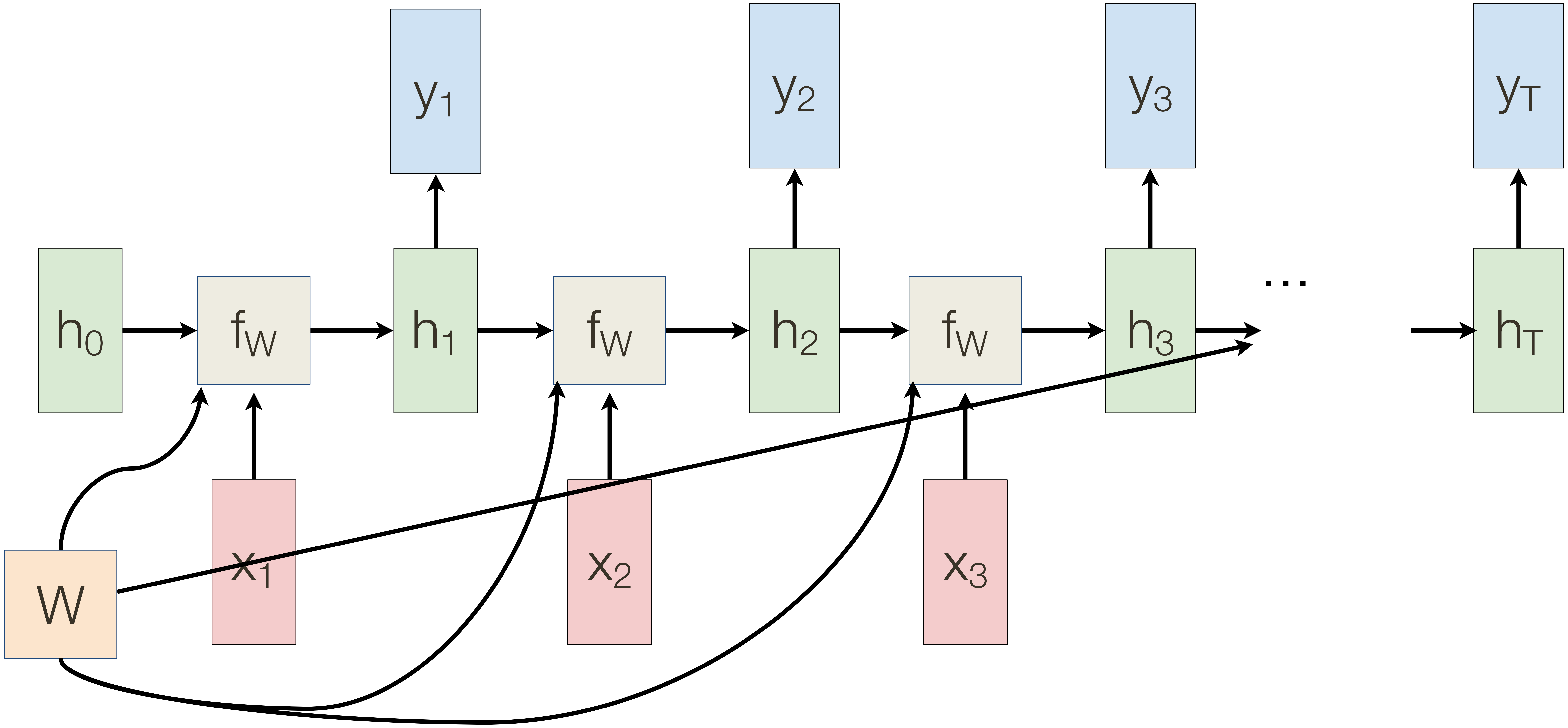


RNN **Computational Graph**

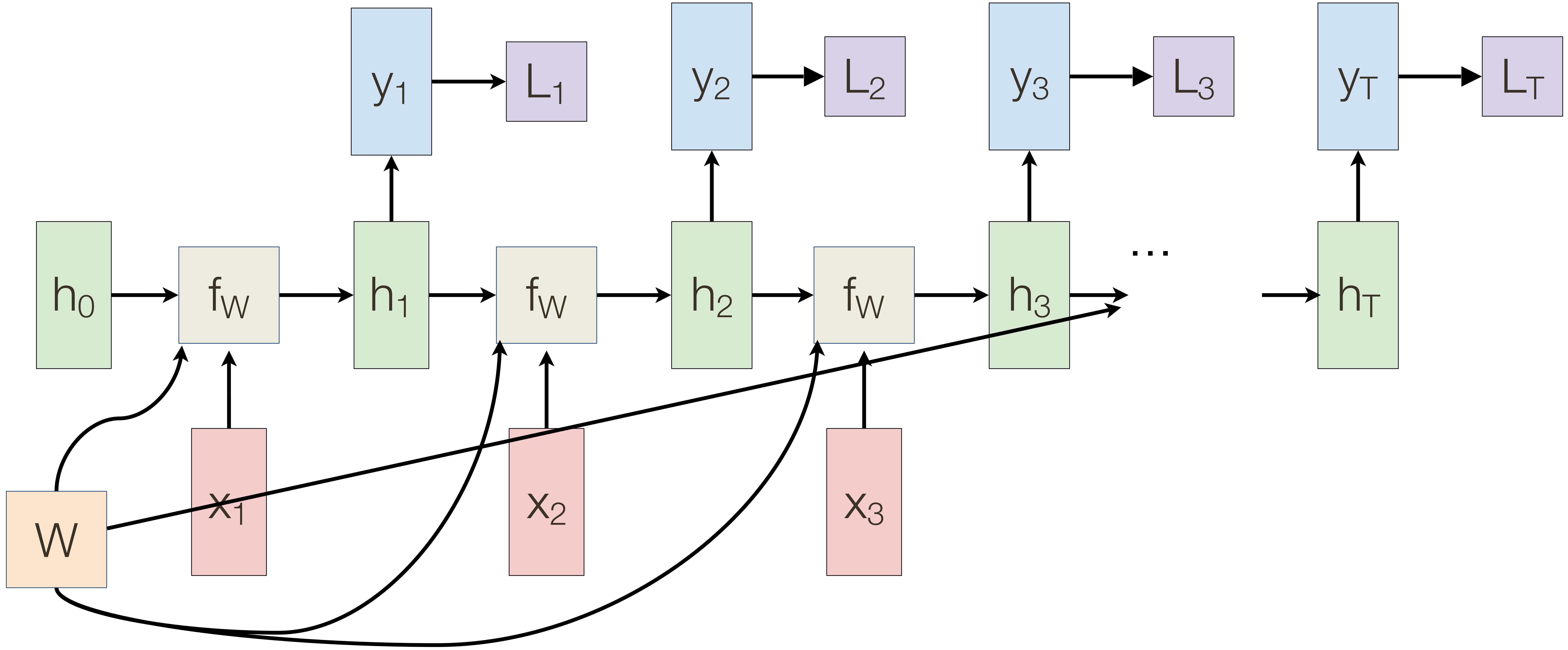
Re-use the same weight matrix at every time-step



RNN **Computational Graph**: Many to Many

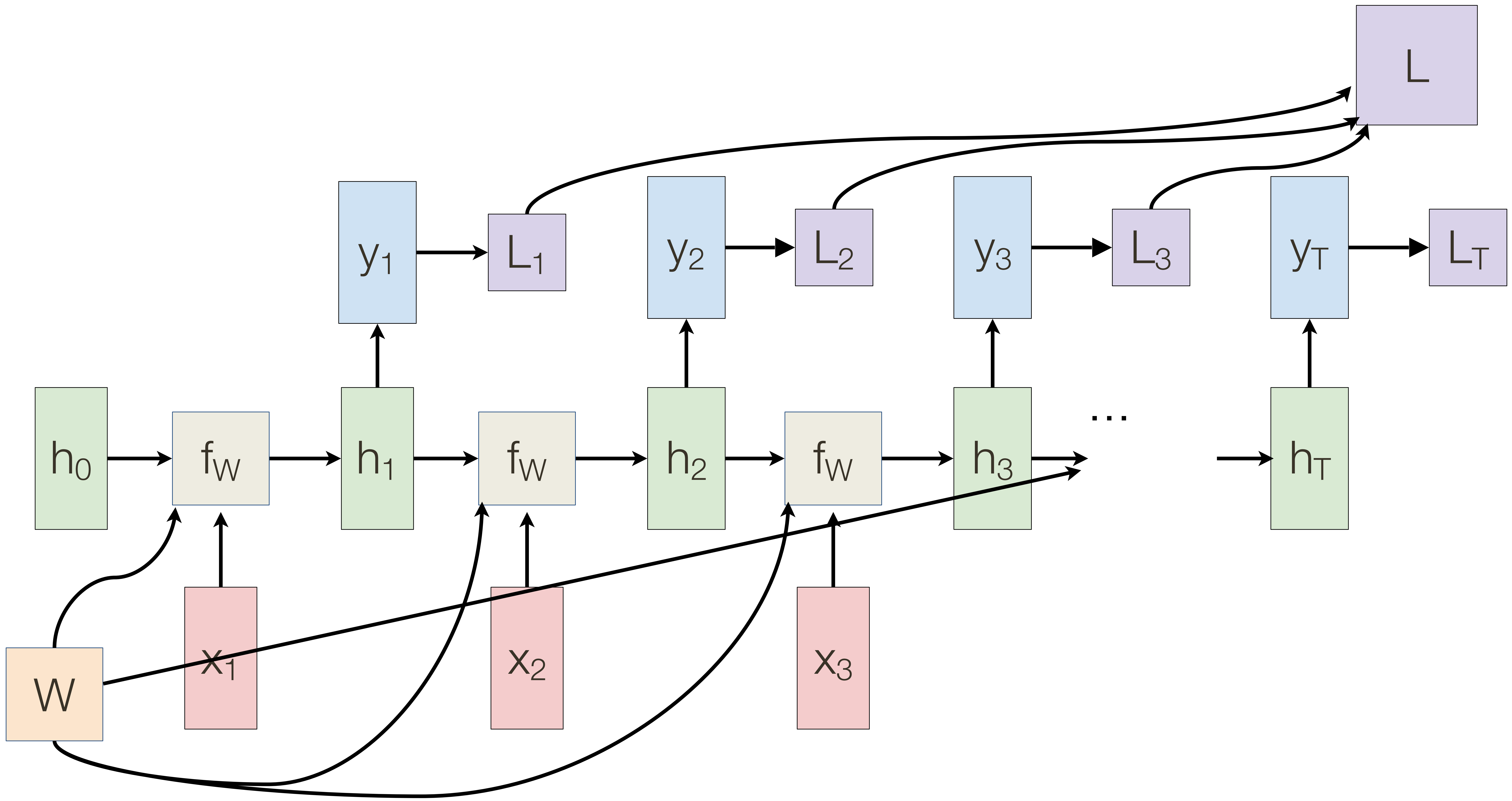


RNN **Computational Graph**: Many to Many



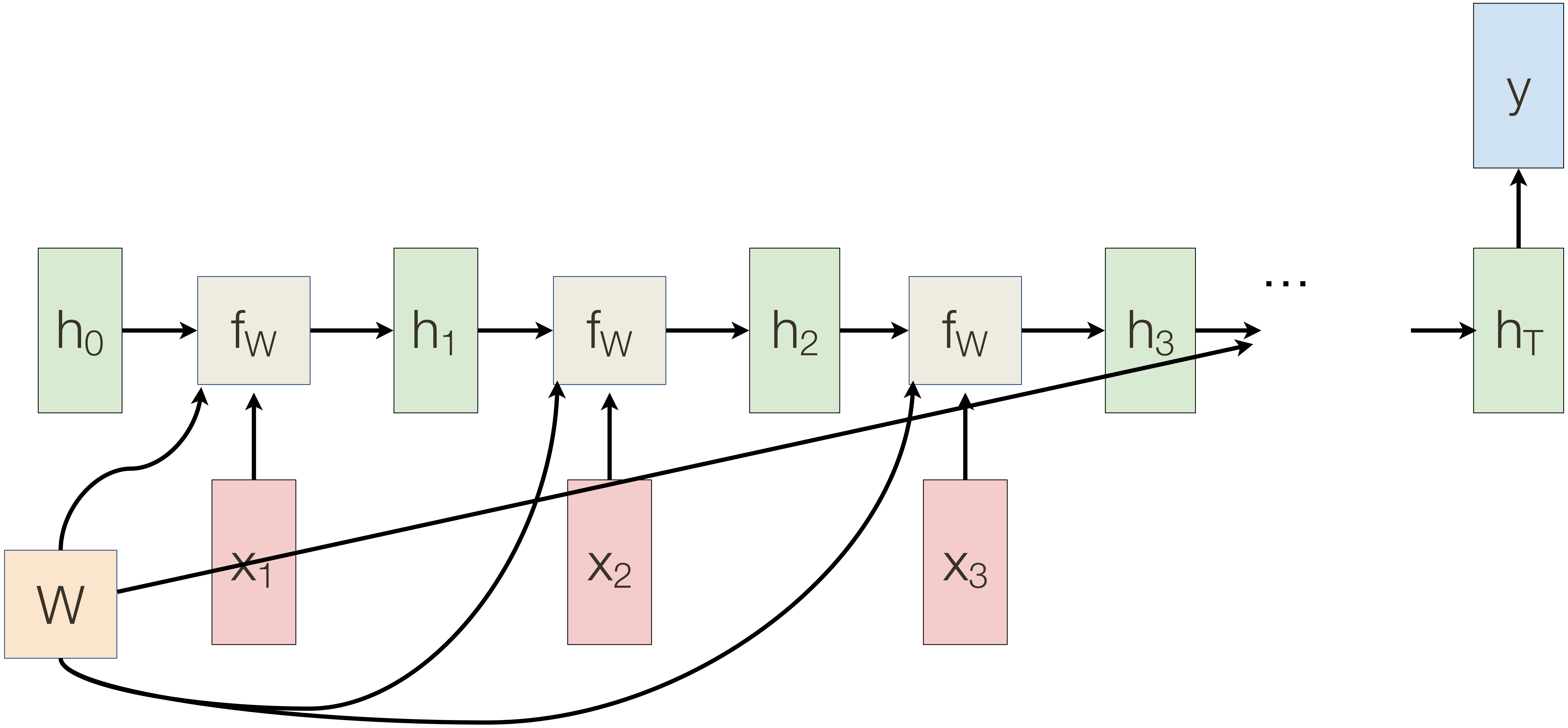
* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

RNN **Computational Graph**: Many to Many



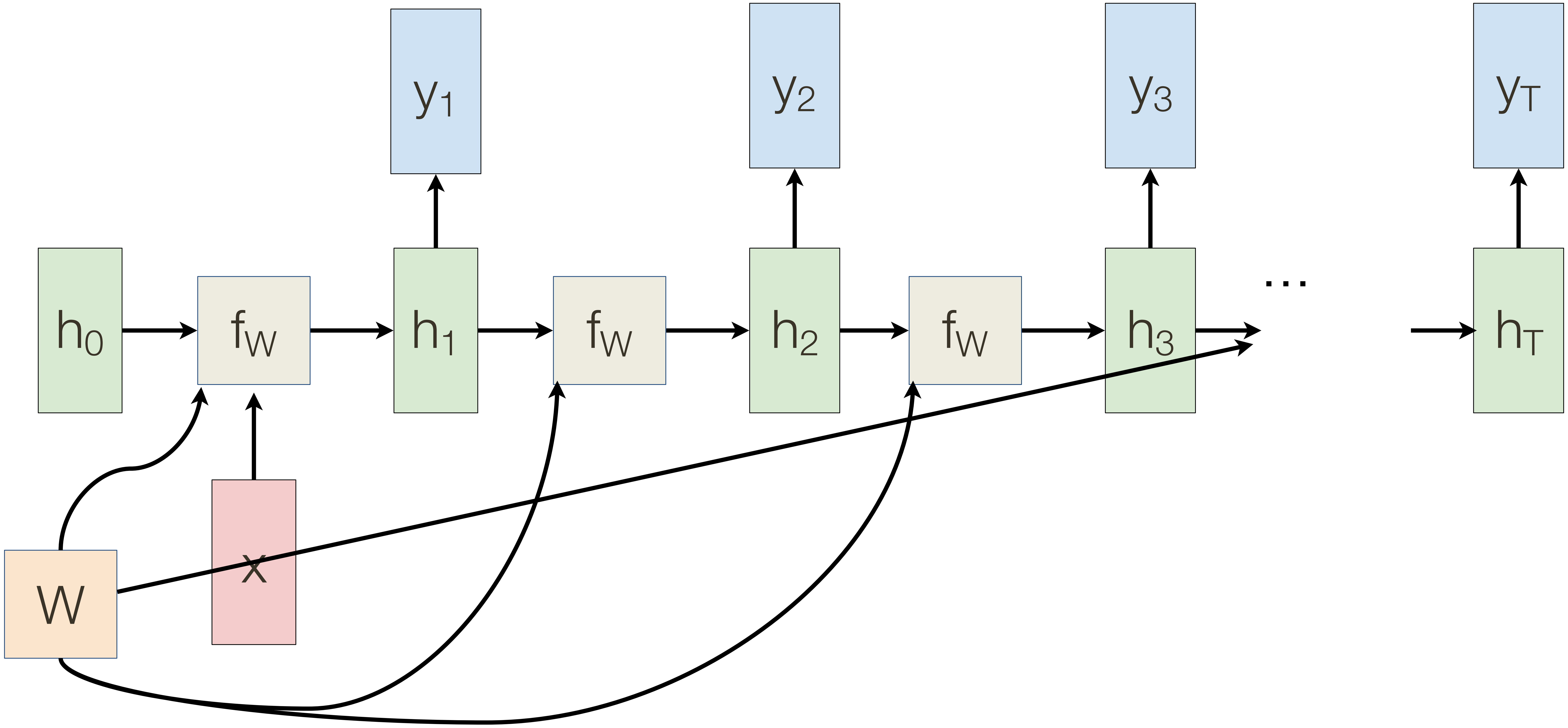
* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

RNN **Computational Graph**: Many to One



* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

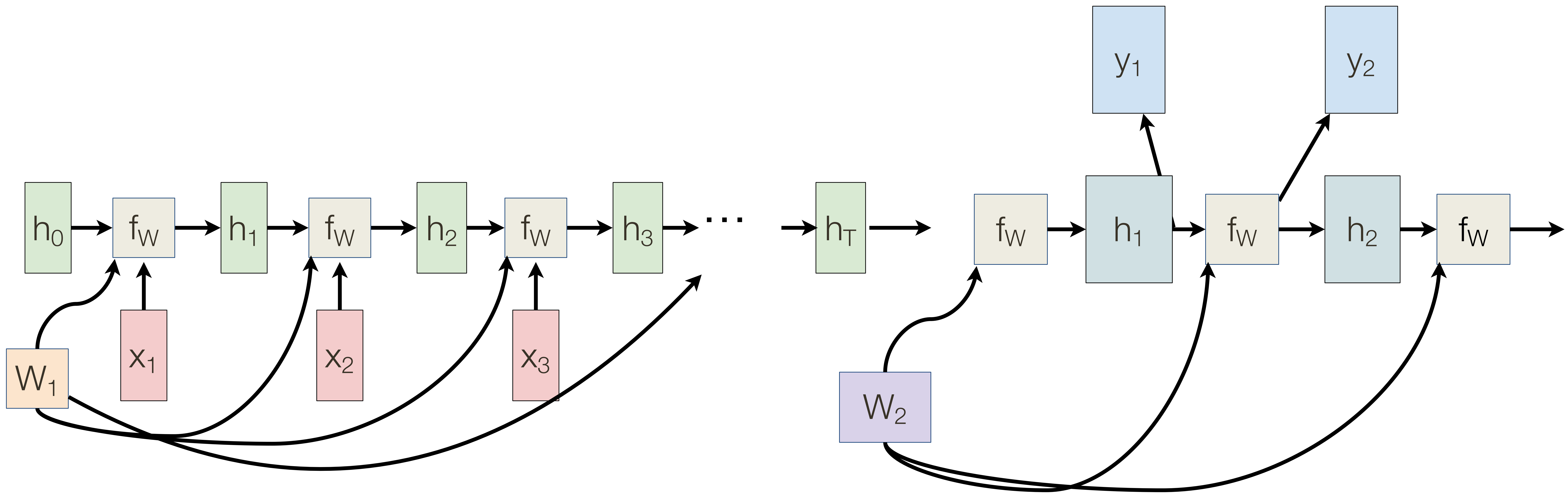
RNN **Computational Graph**: One to Many



Sequence to Sequence: Many to One + One to Many

Many to one: Encode input sequence in a single vector

One to many: Produce output sequence from single input vector

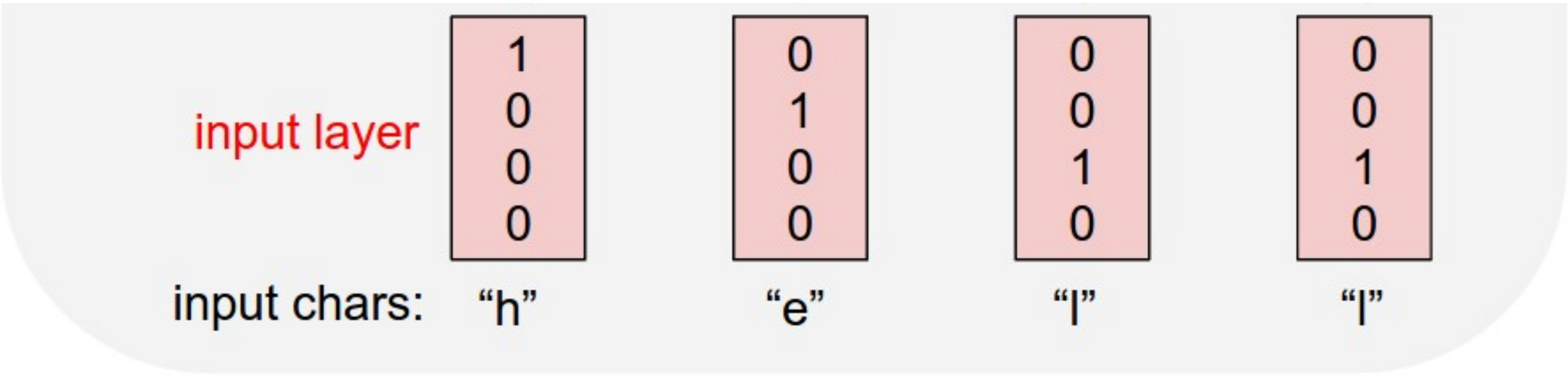


* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Example: Character-level Language Model

Vocabulary:
['h', 'e', 'l', 'o']

Example training sequence:
"hello"



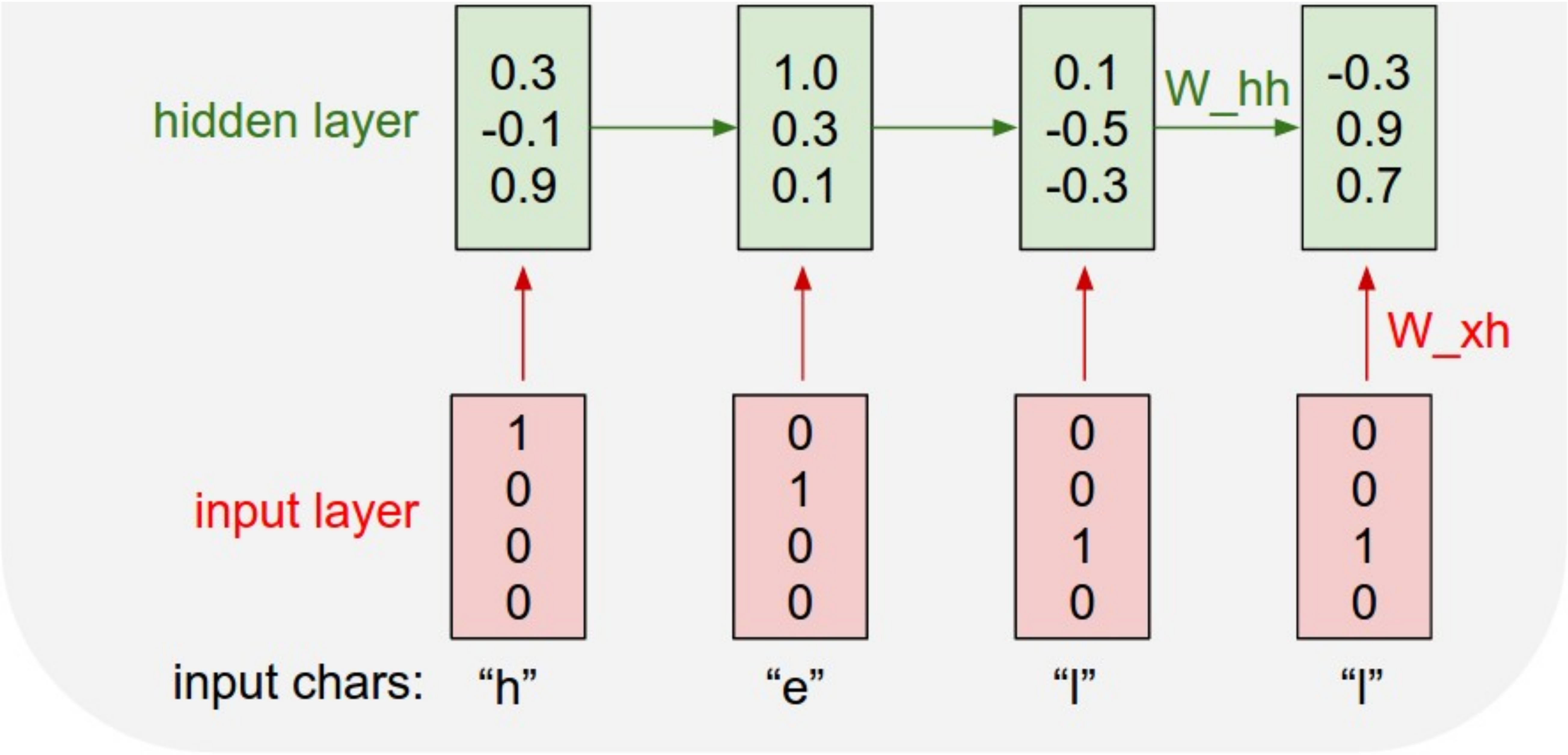
* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Example: Character-level Language Model

Vocabulary:
[‘h’, ‘e’, ‘l’, ‘o’]

$$h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t + b_h)$$

Example training sequence:
“hello”

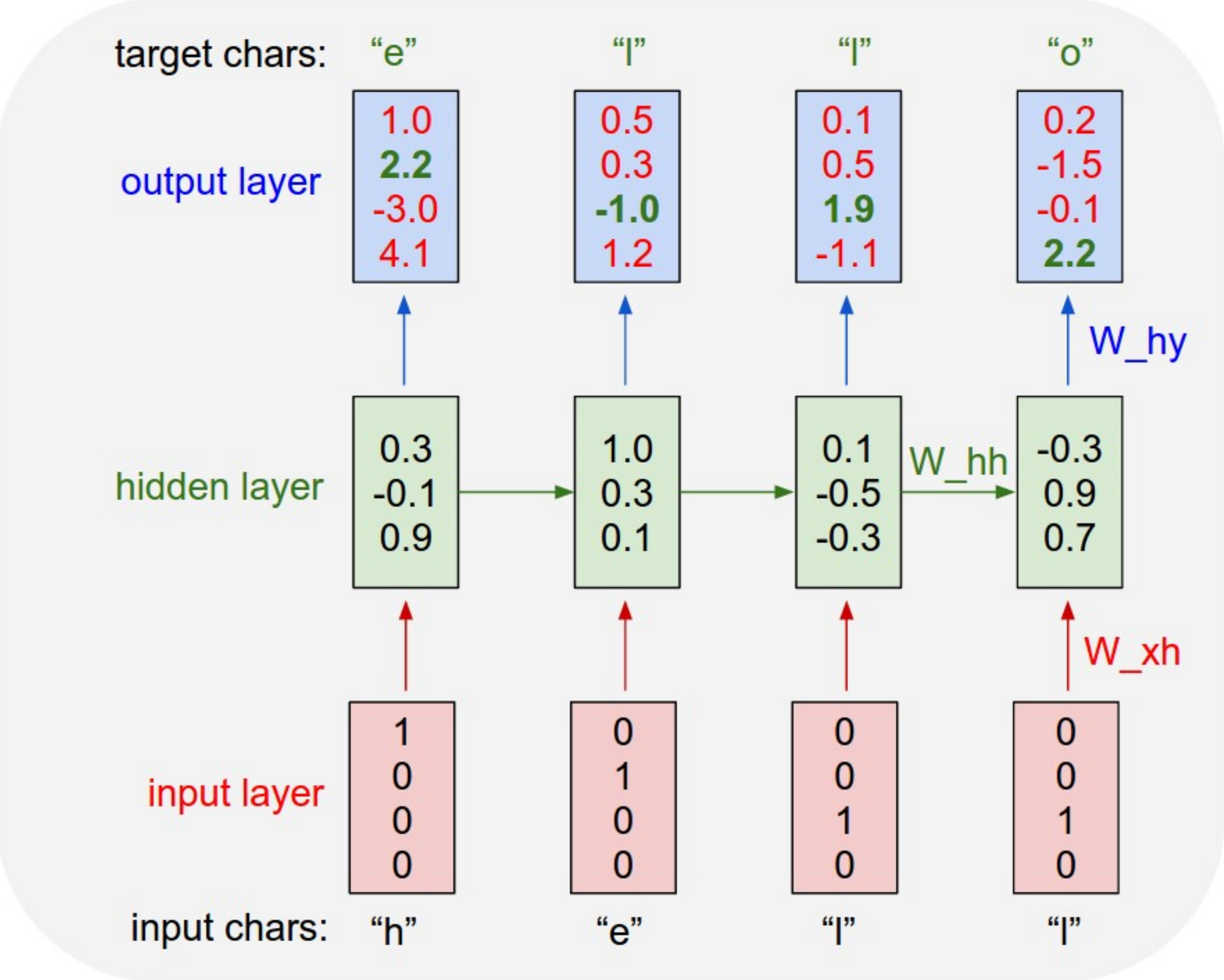


* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Example: Character-level Language Model

Vocabulary:
['h', 'e', 'l', 'o']

Example training sequence:
"hello"



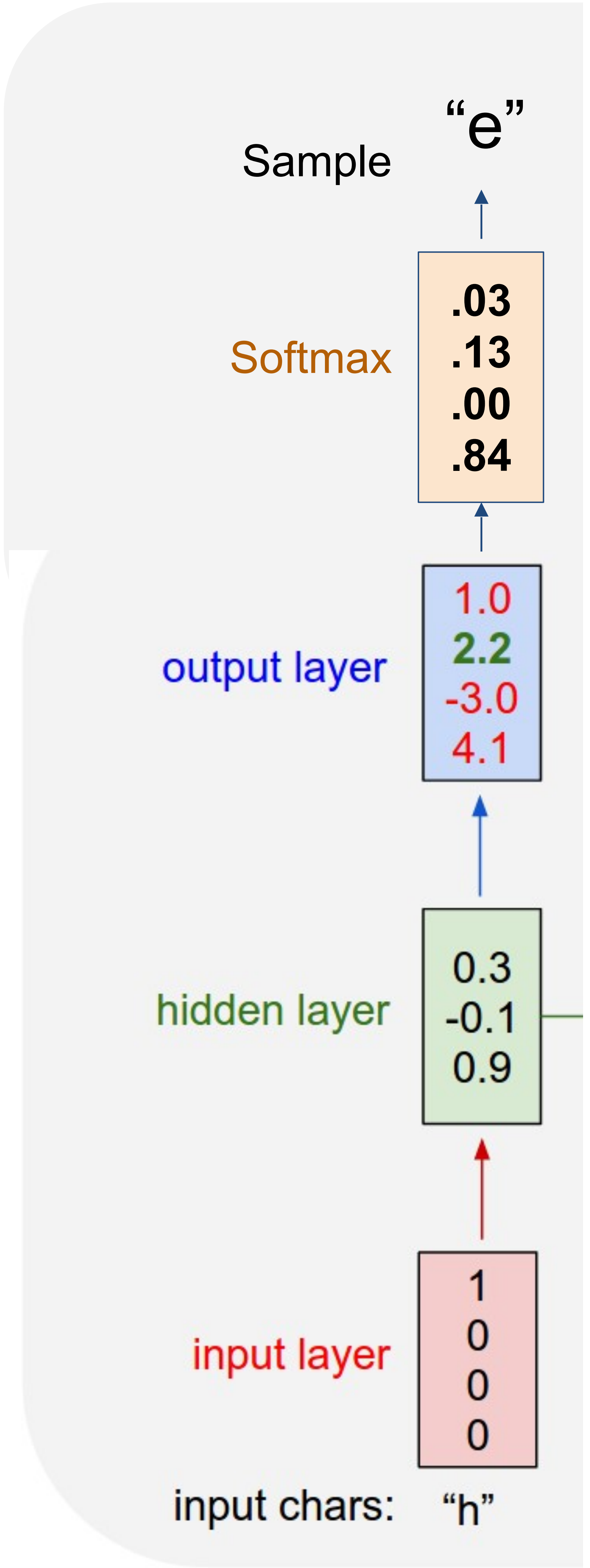
* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Example: Character-level Language Model (**Sampling**)

Vocabulary:

[‘h’, ‘e’, ‘l’, ‘o’]

At test time sample one character at a time and feed back to the model

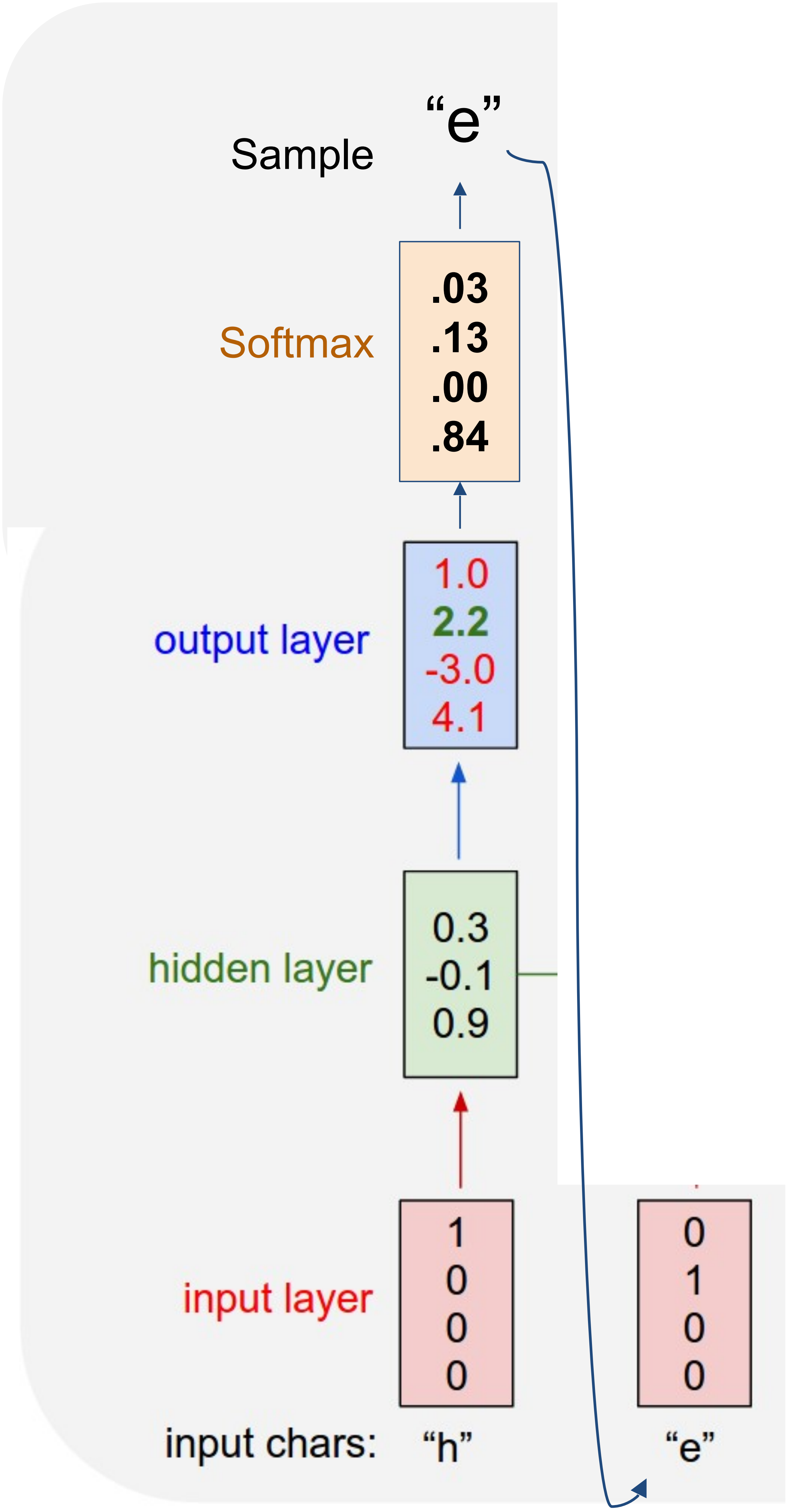


* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Example: Character-level Language Model (**Sampling**)

Vocabulary:
['h', 'e', 'l', 'o']

At test time sample one character at a time and feed back to the model



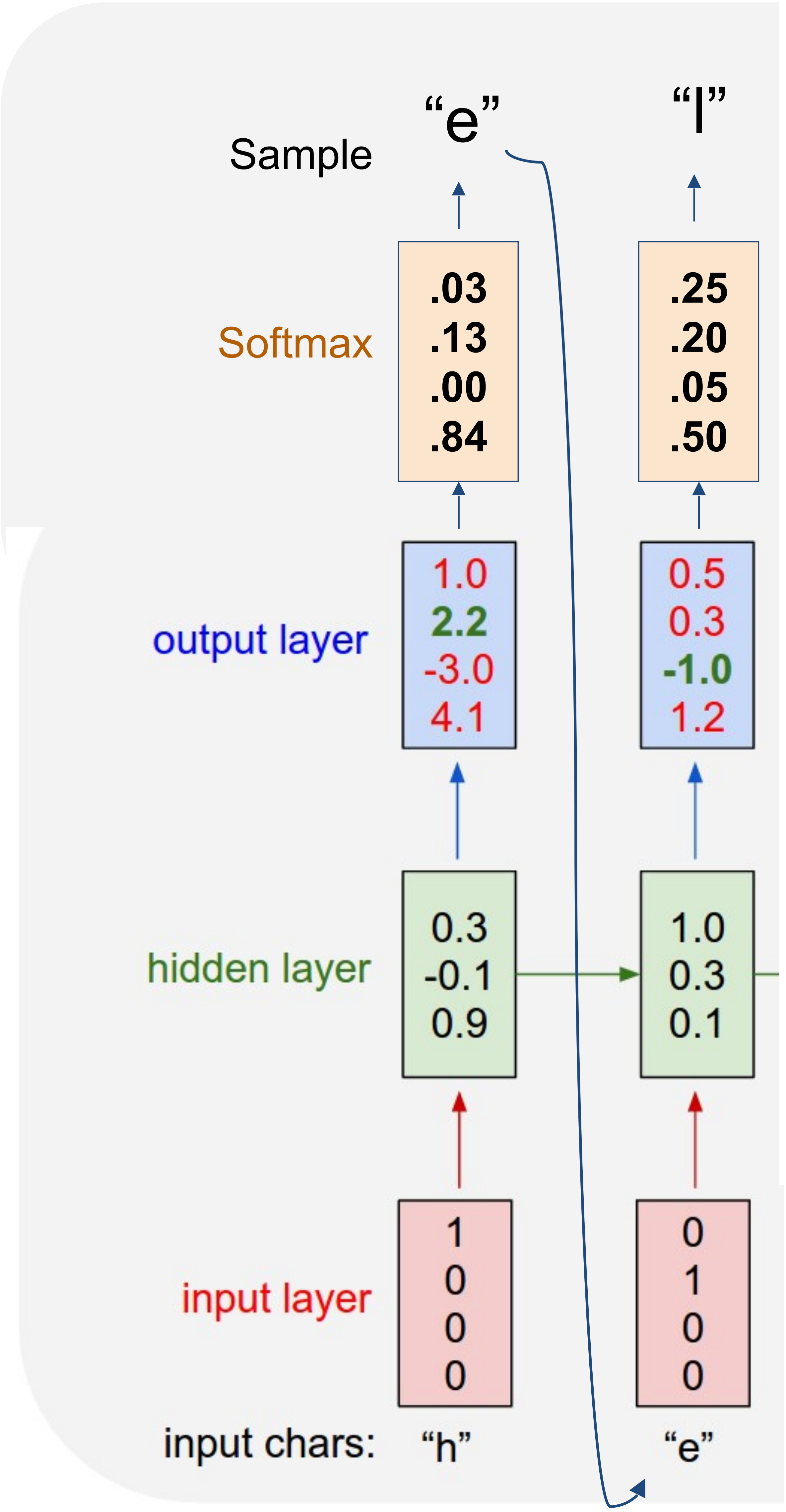
* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Example: Character-level Language Model (**Sampling**)

Vocabulary:

['h', 'e', 'l', 'o']

At test time sample one character at a time and feed back to the model



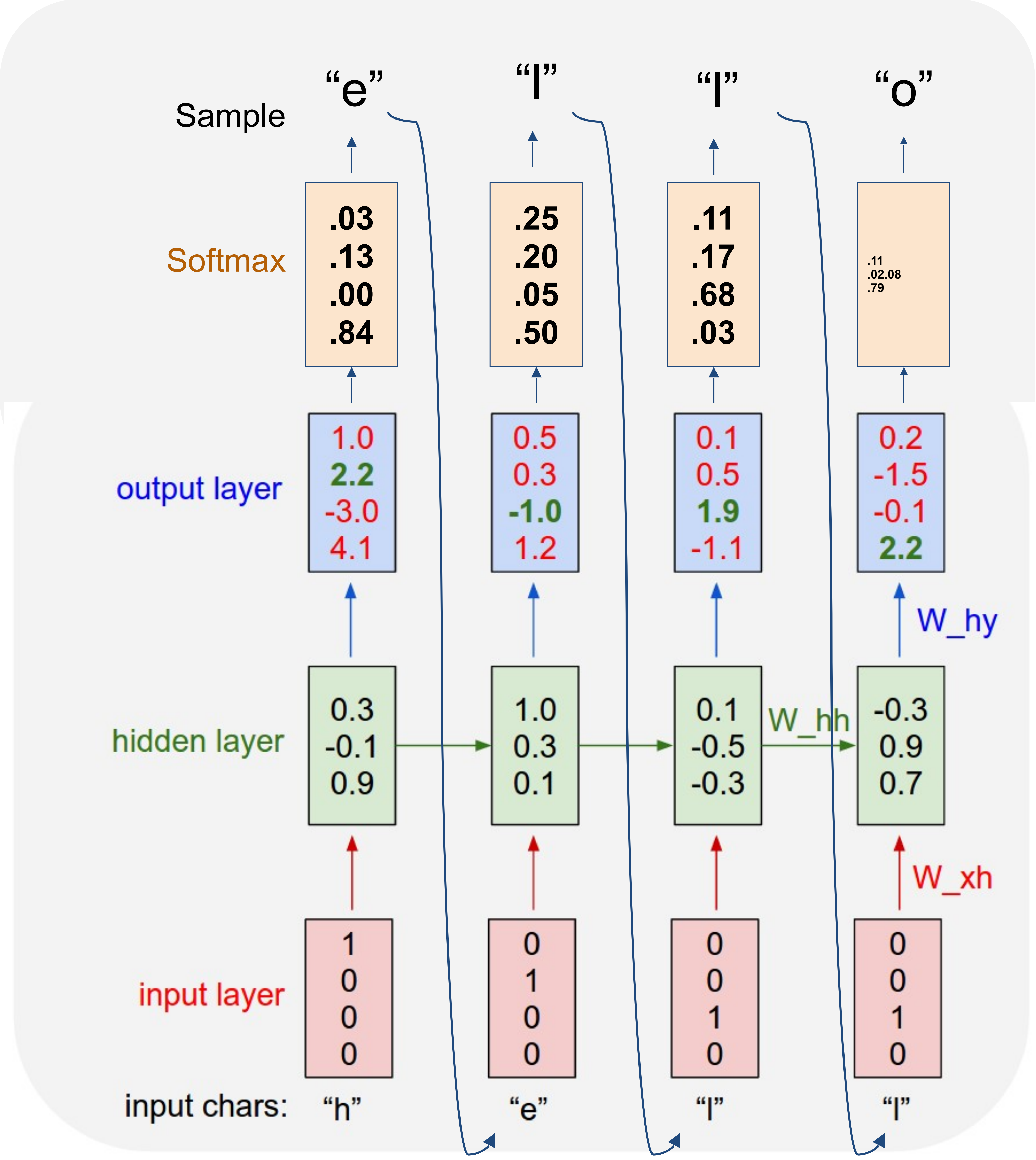
* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Example: Character-level Language Model (**Sampling**)

Vocabulary:

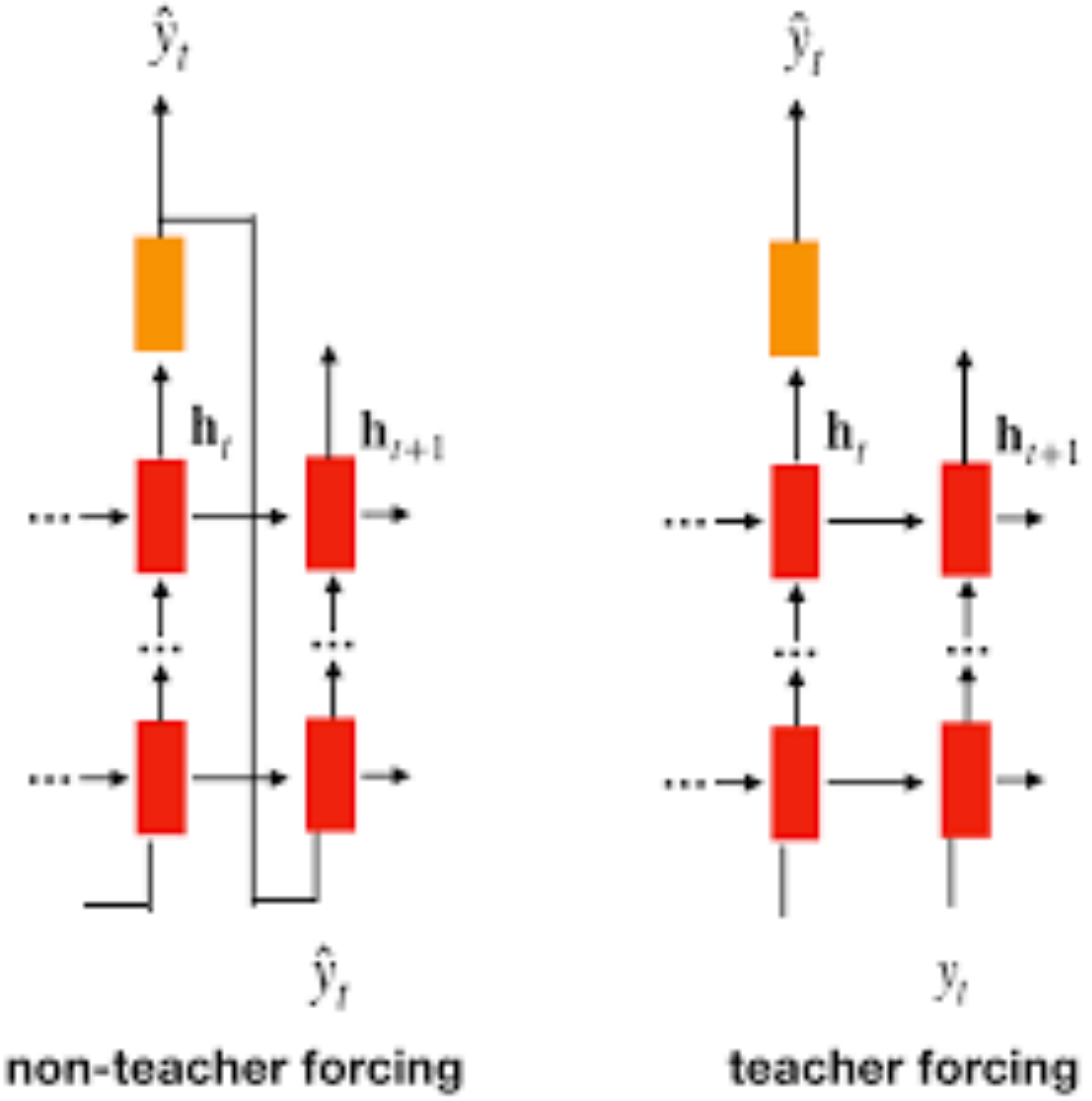
['h', 'e', 'l', 'o']

At test time sample one character at a time and feed back to the model



* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Teacher Forcing



Sampling vs. **ArgMax**

Sampling: allows to generate diverse outputs

ArgMax: could be more stable in practice