THE UNIVERSITY OF BRITISH COLUMBIA

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

Lecture 11: Word Vector Representations

Logistics

Assignment 3 … was due **last night**

— This is the most difficult assignment in the course

Assignment 4 … will be out **today**

— Do not wait

Assignment 5 … will be delayed to enable project proposals

Logistics

Paper readings coming up

Project **groups** and topics

Invited talks

Fun Example: **Code Deobfuscating** with DOBF

Obfuscated Code

```python
class CLASS_0(nn.Module):

    def __init__(VAR_0, VAR_1, VAR_2, VAR_3):
        super(CLASS_0, VAR_0).__init__()
        VAR_0.VAR_1 = VAR_1
        VAR_0.VAR_2 = VAR_2
        VAR_0.VAR_4 = nn.Linear(VAR_1, (4 * VAR_2), bias=VAR_3)
        VAR_0.VAR_5 = nn.Linear(VAR_2, (4 * VAR_2), bias=VAR_3)
        VAR_0.FUNC_0()

    def FUNC_0(VAR_6):
        VAR_7 = (1.0 / math.sqrt(VAR_6.VAR_8))
        for VAR_9 in VAR_6.VAR_10():
            VAR_9.data.uniform_((- VAR_7), VAR_7)

    def FUNC_1(VAR_11, VAR_12, VAR_13):
        (VAR_14, VAR_15) = VAR_13
        VAR_14 = VAR_14.view(VAR_14.size(1), (- 1))
        VAR_15 = VAR_15.view(VAR_15.size(1), (- 1))
        VAR_12 = VAR_12.view(VAR_12.size(1), (- 1))
        VAR_16 = (VAR_11.VAR_4(VAR_12) + VAR_11.VAR_5(VAR_14))
        VAR_17 = VAR_16[:, :(3 * VAR_11.VAR_8)].sigmoid()
        VAR_18 = VAR_16[:, (3 * VAR_11.VAR_8):].tanh()
        VAR_19 = VAR_17[:, :VAR_11.VAR_8]
        VAR_20 = VAR_17[:, VAR_11.VAR_8:(2 * VAR_11.VAR_8)]
        VAR_21 = VAR_17[:, (- VAR_11.VAR_8):]
        VAR_22 = (th.mul(VAR_15, VAR_20) + th.mul(VAR_19, VAR_18))
        VAR_23 = th.mul(VAR_21, VAR_22.tanh())
        VAR_23 = VAR_23.view(1, VAR_23.size(0), (- 1))
        VAR_22 = VAR_22.view(1, VAR_22.size(0), (- 1))
        return (VAR_23, (VAR_23, VAR_22))
```

Code Deobfuscated using DOBF

```python
class LSTM(nn.Module):

    def __init__(self, input_size, hidden_size, bias):
        super(LSTM, self).__init__()
        self.input_size = input_size
        self.hidden_size = hidden_size
        self.h1 = nn.Linear(input_size, (4 * hidden_size), bias=bias)
        self.h2 = nn.Linear(hidden_size, (4 * hidden_size), bias=bias)
        self.init_weights()

    def init_weights(self):
        stdv = (1.0 / math.sqrt(self.hidden_size))
        for m in self.modules():
            m.data.uniform_((- stdv), stdv)

    def forward(self, x, prev_state):
        (prev_h, prev_c) = prev_state
        prev_h = prev_h.view(prev_h.size(1), (- 1))
        prev_c = prev_c.view(prev_c.size(1), (- 1))
        x = x.view(x.size(1), (- 1))
        h = (self.h1(x) + self.h2(prev_h))
        s = h[:, :(3 * self.hidden_size)].sigmoid()
        c = h[:, (3 * self.hidden_size):].tanh()
        r = s[:, :self.hidden_size]
        g = s[:, self.hidden_size:(2 * self.hidden_size)]
        o = s[:, (- self.hidden_size):]
        c = (th.mul(prev_c, g) + th.mul(r, c))
        h = th.mul(o, c.tanh())
        h = h.view(1, h.size(0), (- 1))
        c = c.view(1, c.size(0), (- 1))
        return (h, (h, c))
```

[Roziere et al., ArXiv, 2021]

Representing a **Word:** One Hot Encoding

Vocabulary

dog

cat

person

holding

tree

computer

using

*slide from V. Ordonex

Representing a **Word:** One Hot Encoding

Vocabulary

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

*slide from V. Ordonex

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

bag-of-words representation

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

dog cat person holding tree computer using

*slide from V. Ordonex

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

bag-of-words representation

person holding dog {3, 4, 1} [1, 0, 1, 1, 0, 0, 0, 0, 0, 0]

dog cat person holding tree computer using

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

*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]

dog cat person holding tree computer using

*slide from V. Ordonex

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

		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

Vocabulary

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

*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

Word Representations

1. **One-hot encodings** — only non-zero at the index of the word

    e.g., [0, 1, 0, 0, 0, …., 0, 0, 0]

    Good: simple

    Bad: not compact, distance between words always same (e.g., synonyms vs. antonyms)

2. **Word feature representations** — manually define “good” features

    e.g., [1, 1, 0, 30, 0, …., 0, 0, 0] -> 300-dimensional irrespective of dictionary

    e.g., word ends on -ing

3. **Learned word representations** — vector should approximate “meaning” of the word

    e.g., [1, 1, 0, 30, 0, …., 0, 0, 0] -> 300-dimensional irrespective of dictionary

    Good: compact, distance between words is semantic

* Adopted from slides by Louis-Philippe Morency

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.

* 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

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

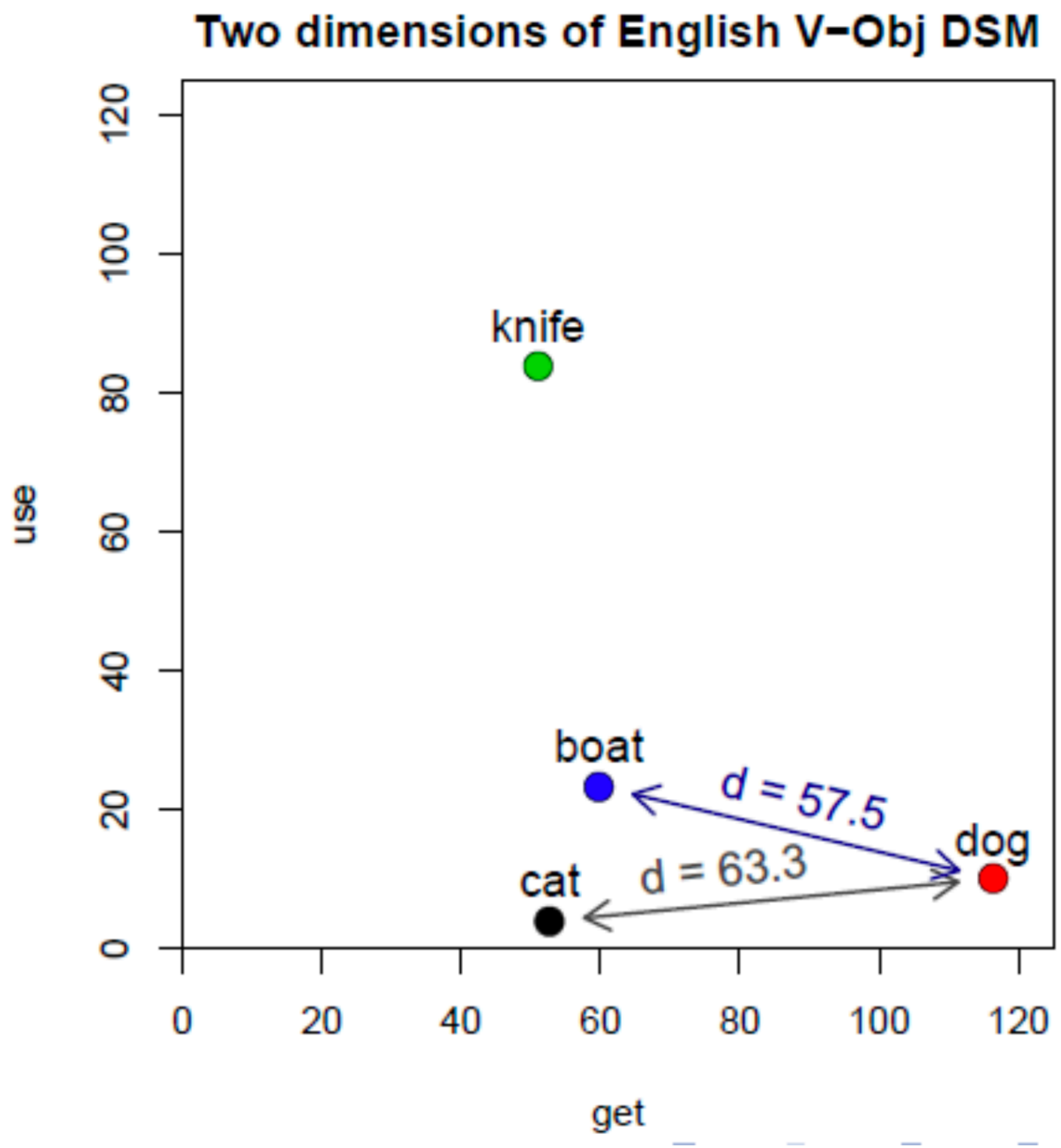
* Slides from Louis-Philippe Morency

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 cat)

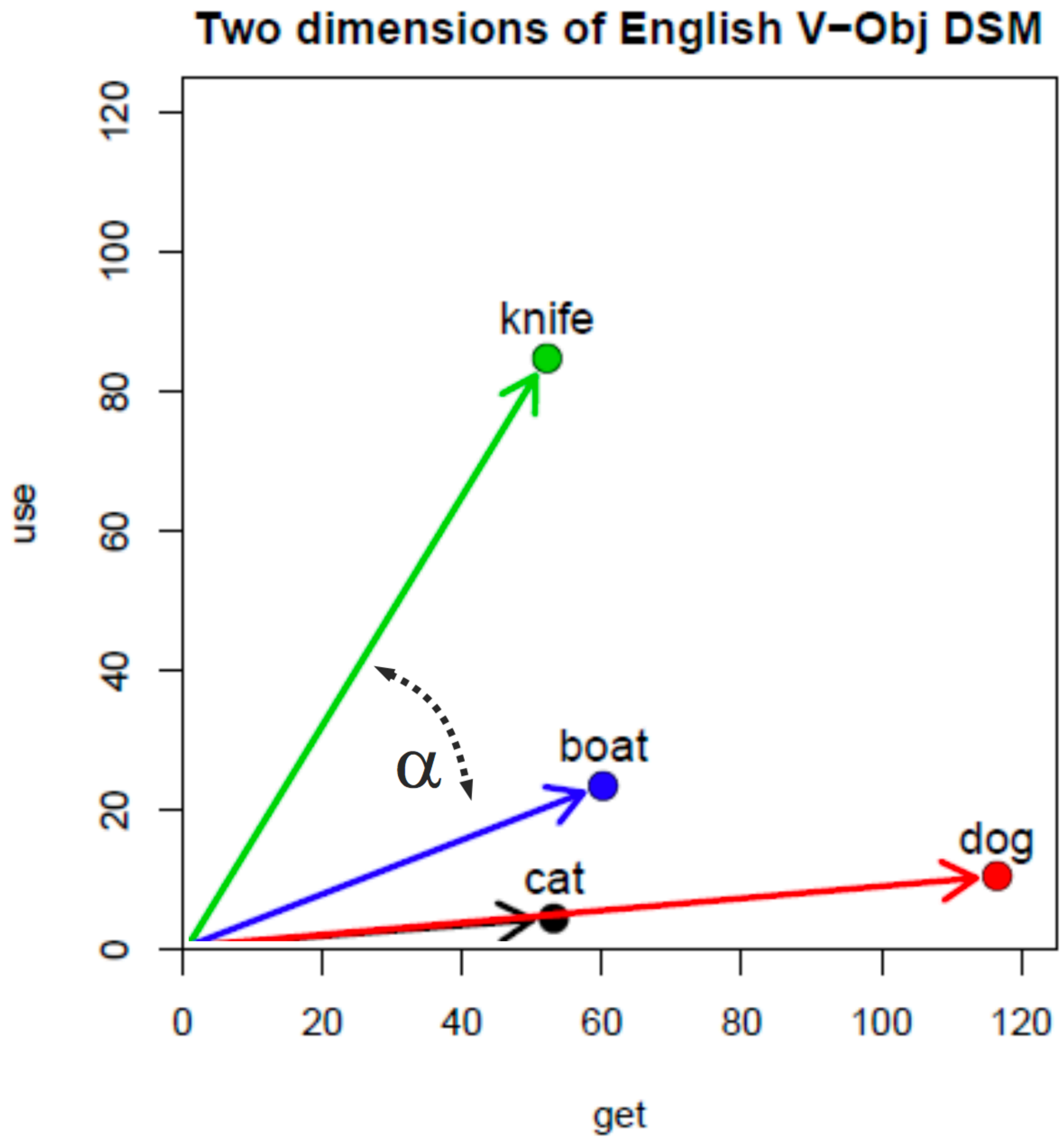


Angle and Similarity

— 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

	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

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

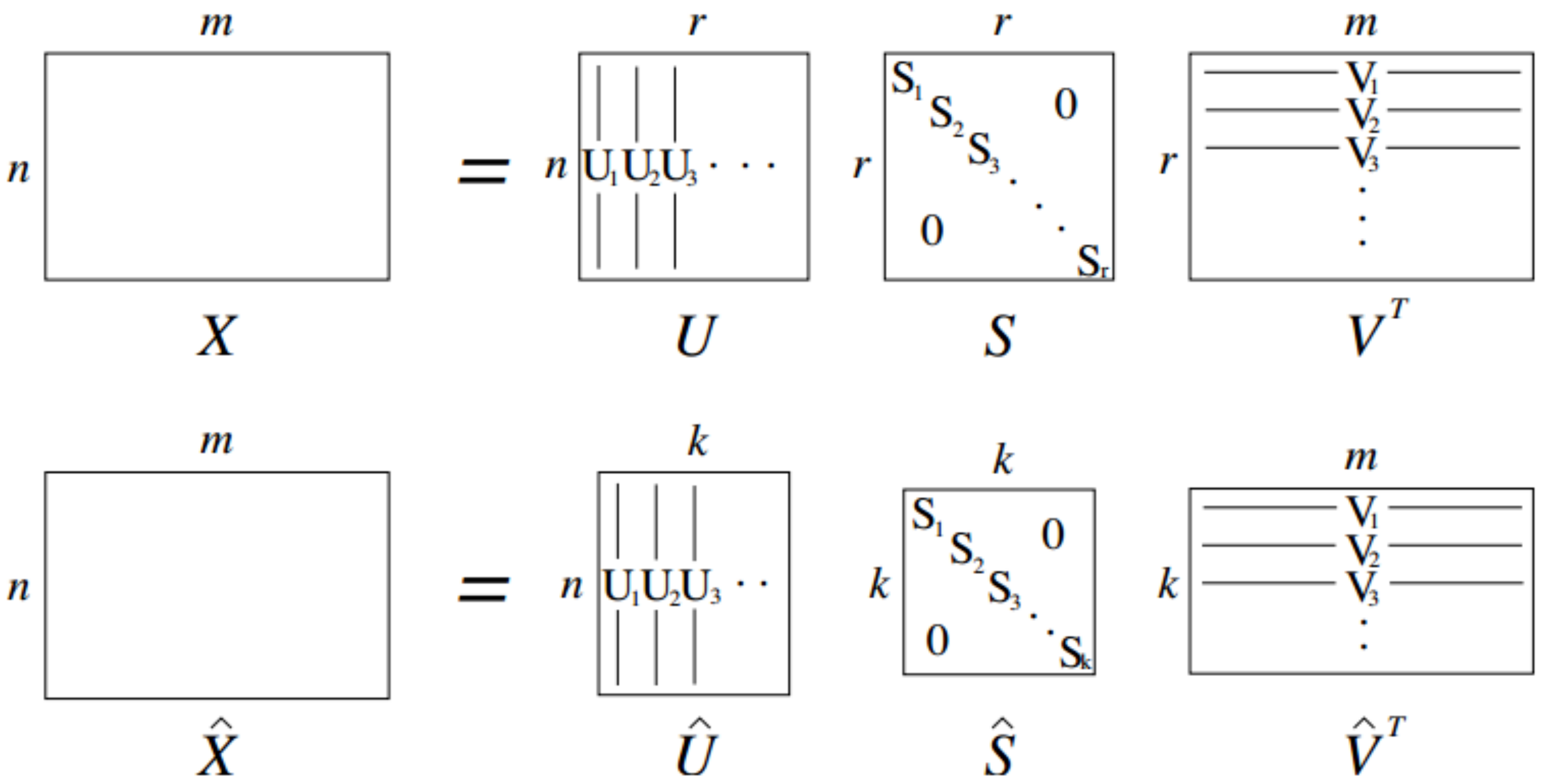
Way too high dimensional!

	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

* Slides from Louis-Philippe Morency

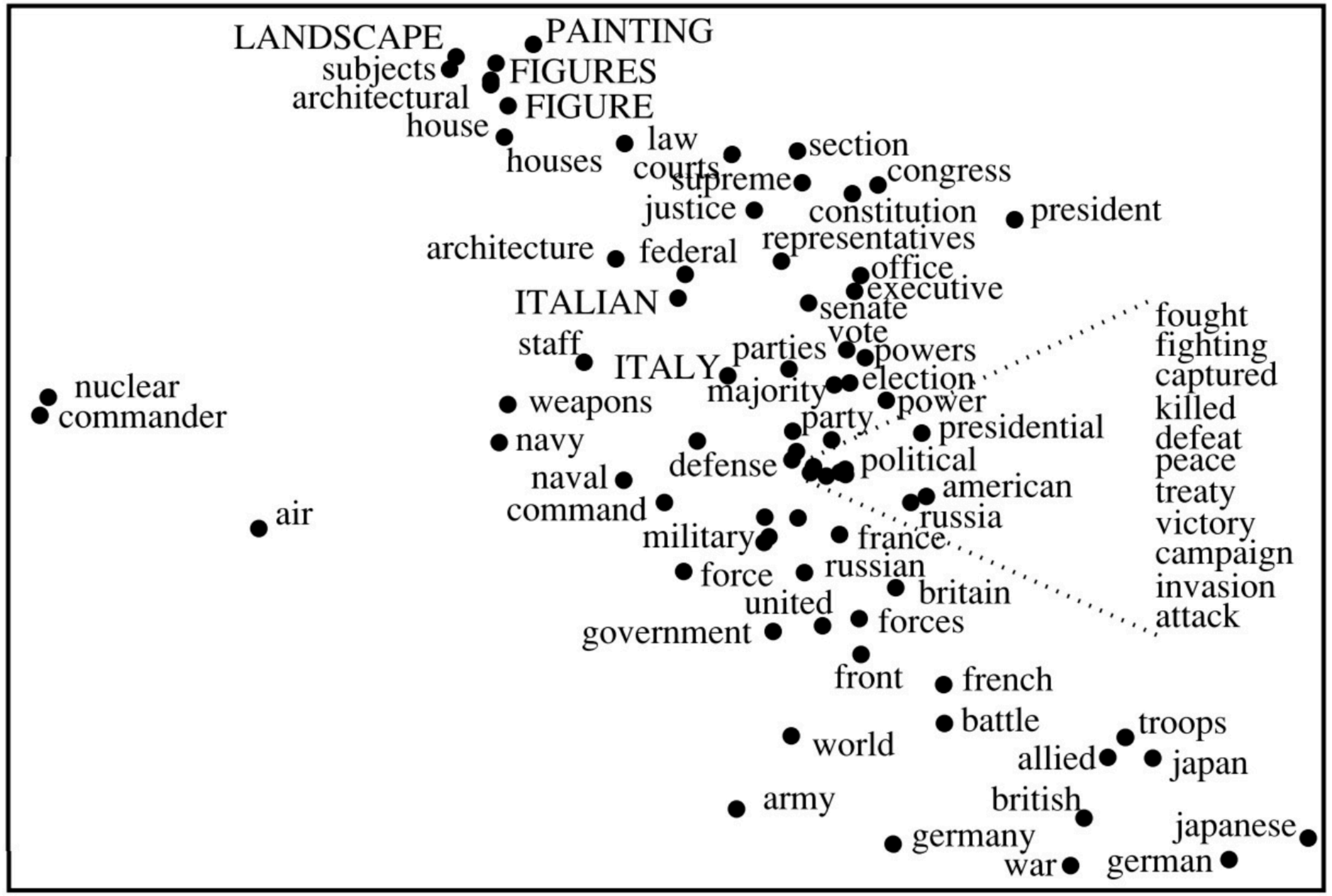
SVD for Dimensionality Reduction



*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

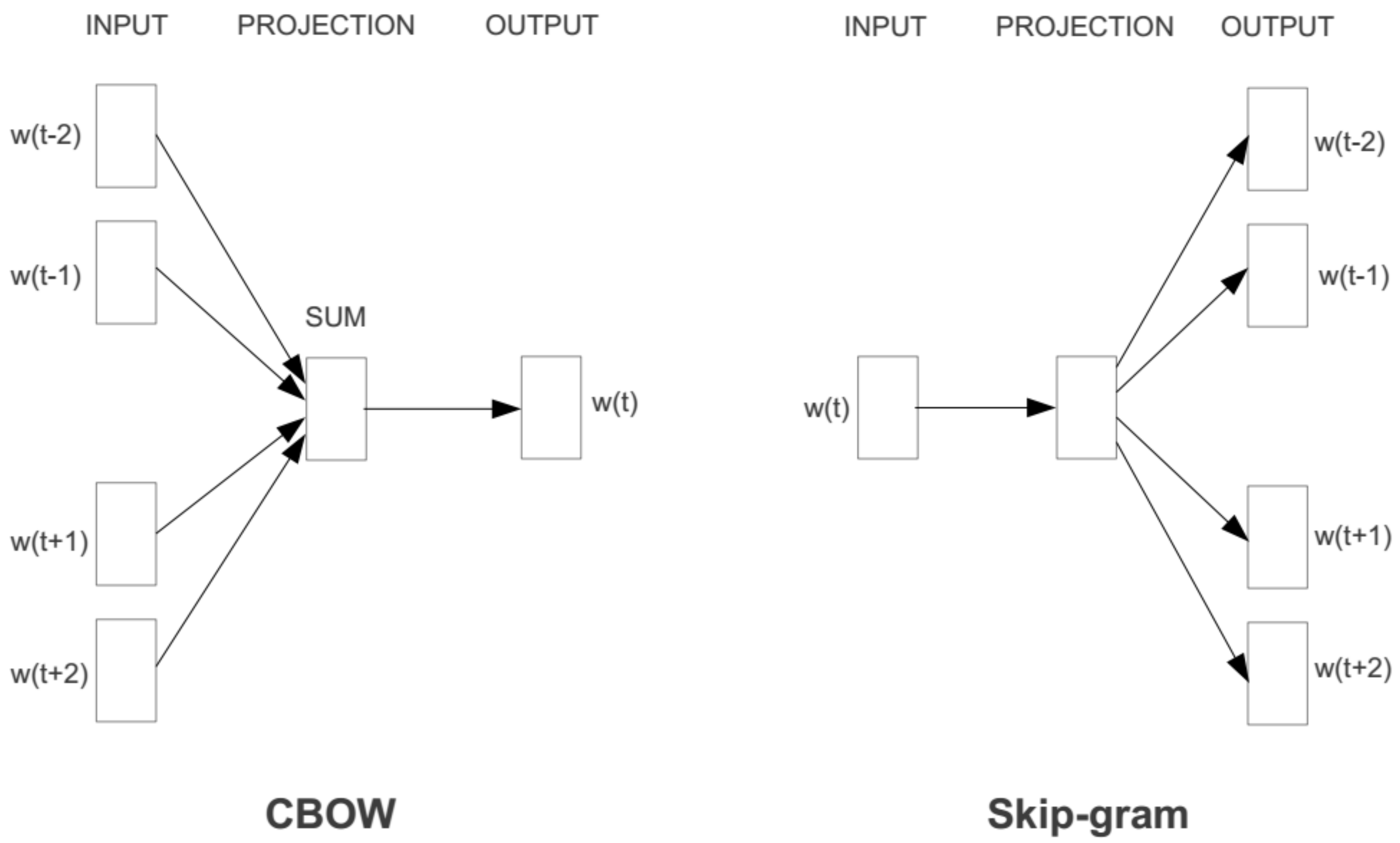
Benefits: Faster and easier to incorporate new document, words, etc.

*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

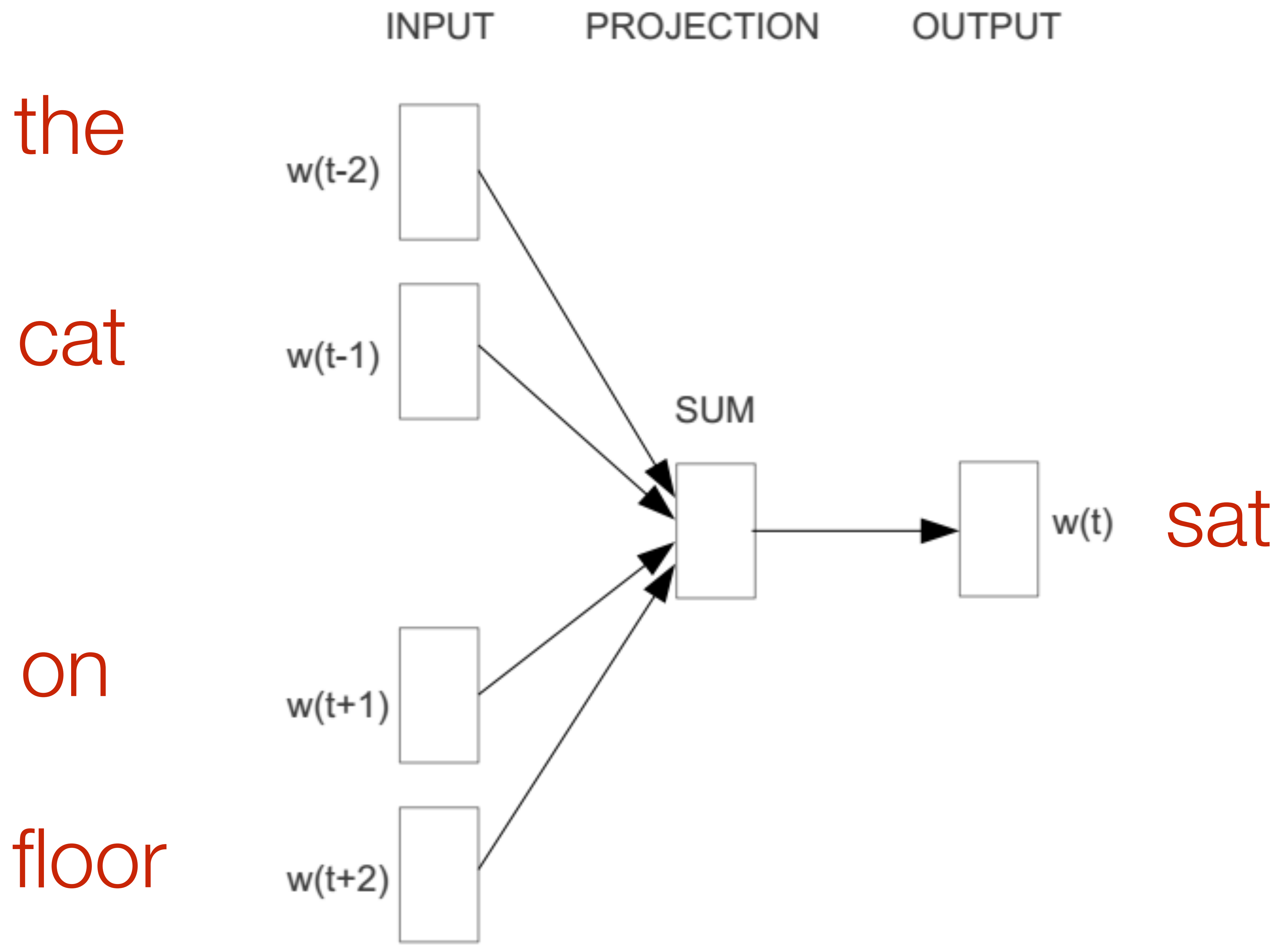
Skip-gram: use the middle word to predict surrounding ones in a window

*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

[Mikolov et al., 2013]

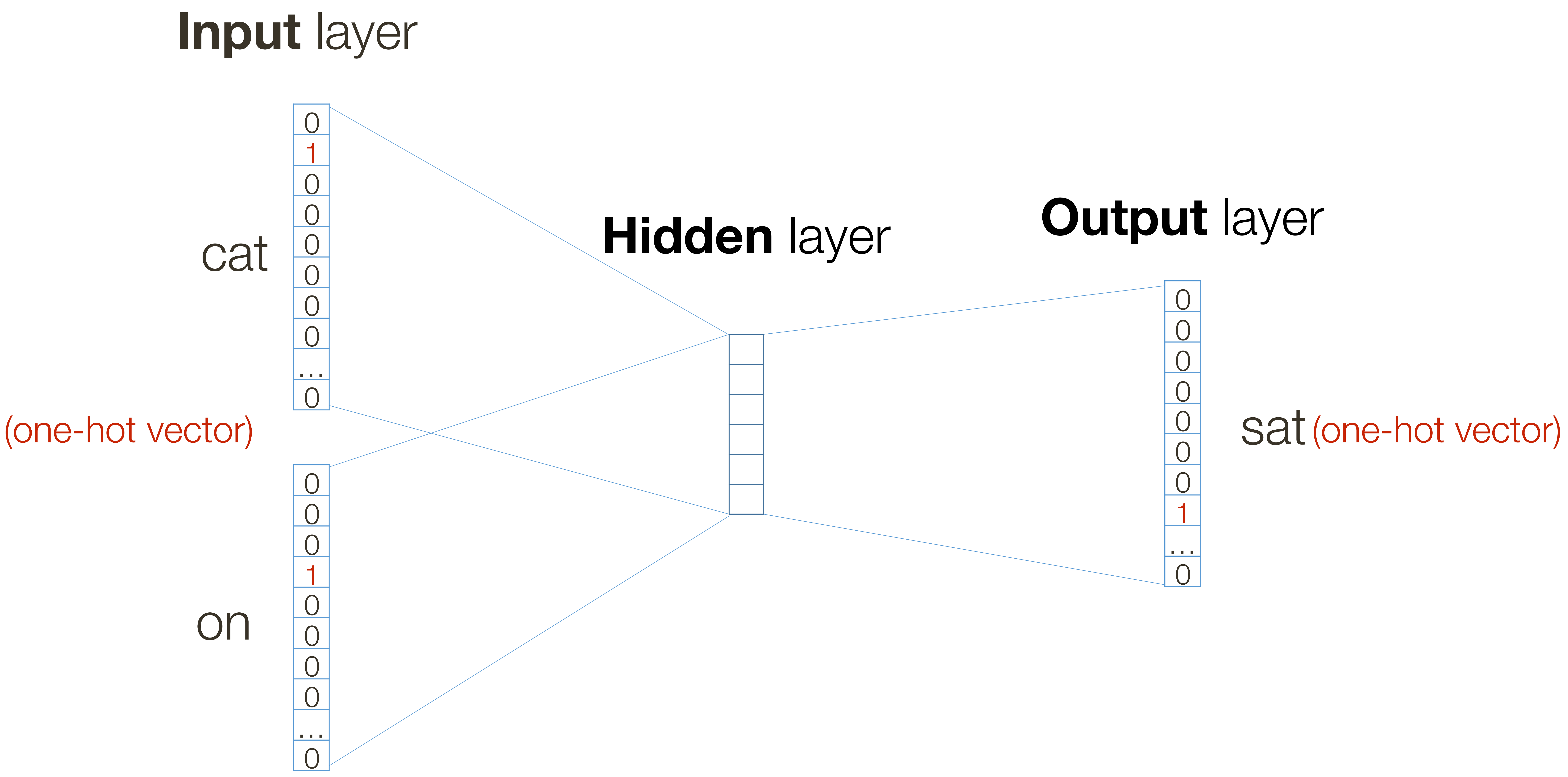
Example: “The cat sat on floor” (window size 2)



*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

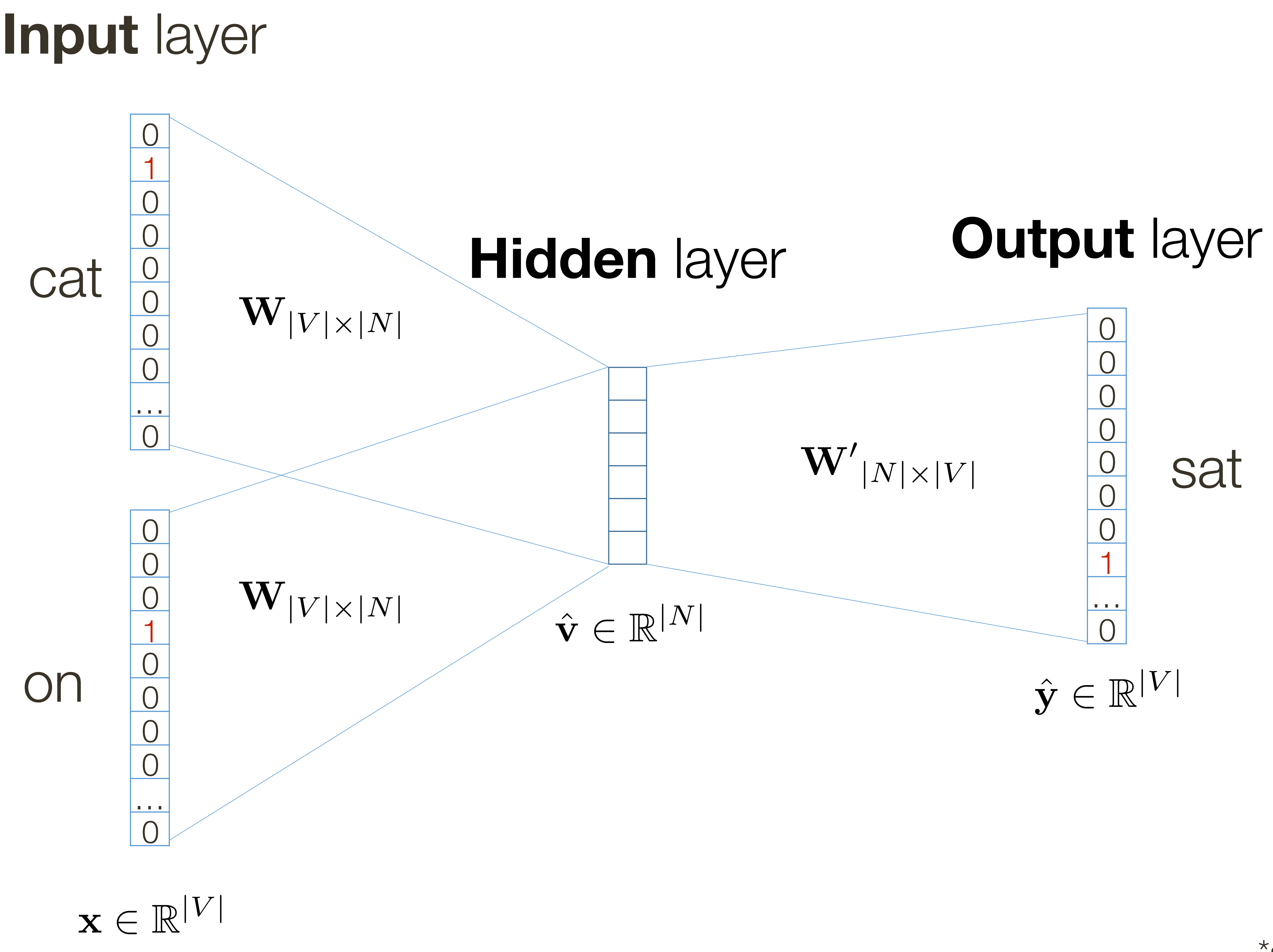
[Mikolov et al., 2013]



*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

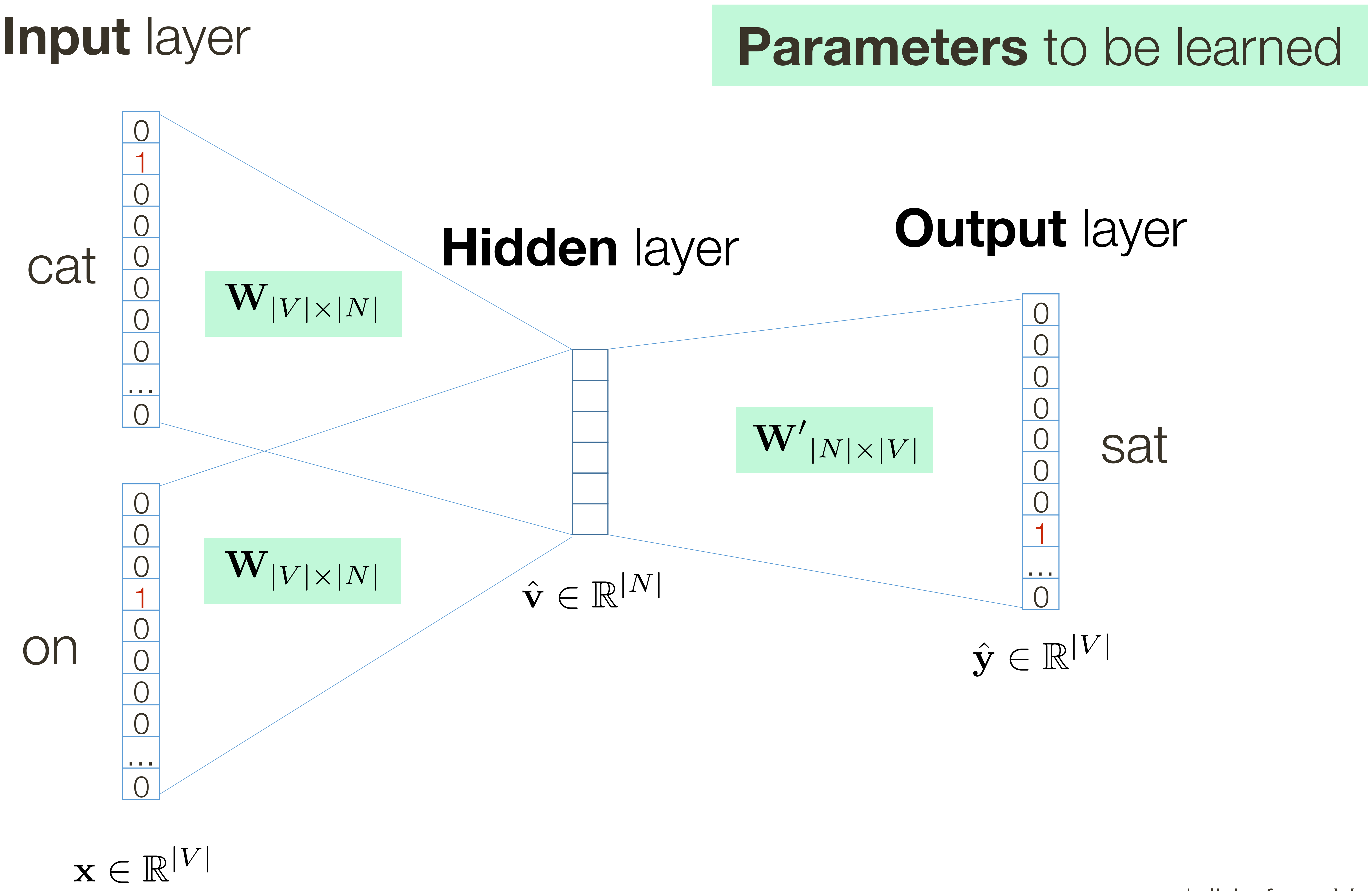
[Mikolov et al., 2013]



*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

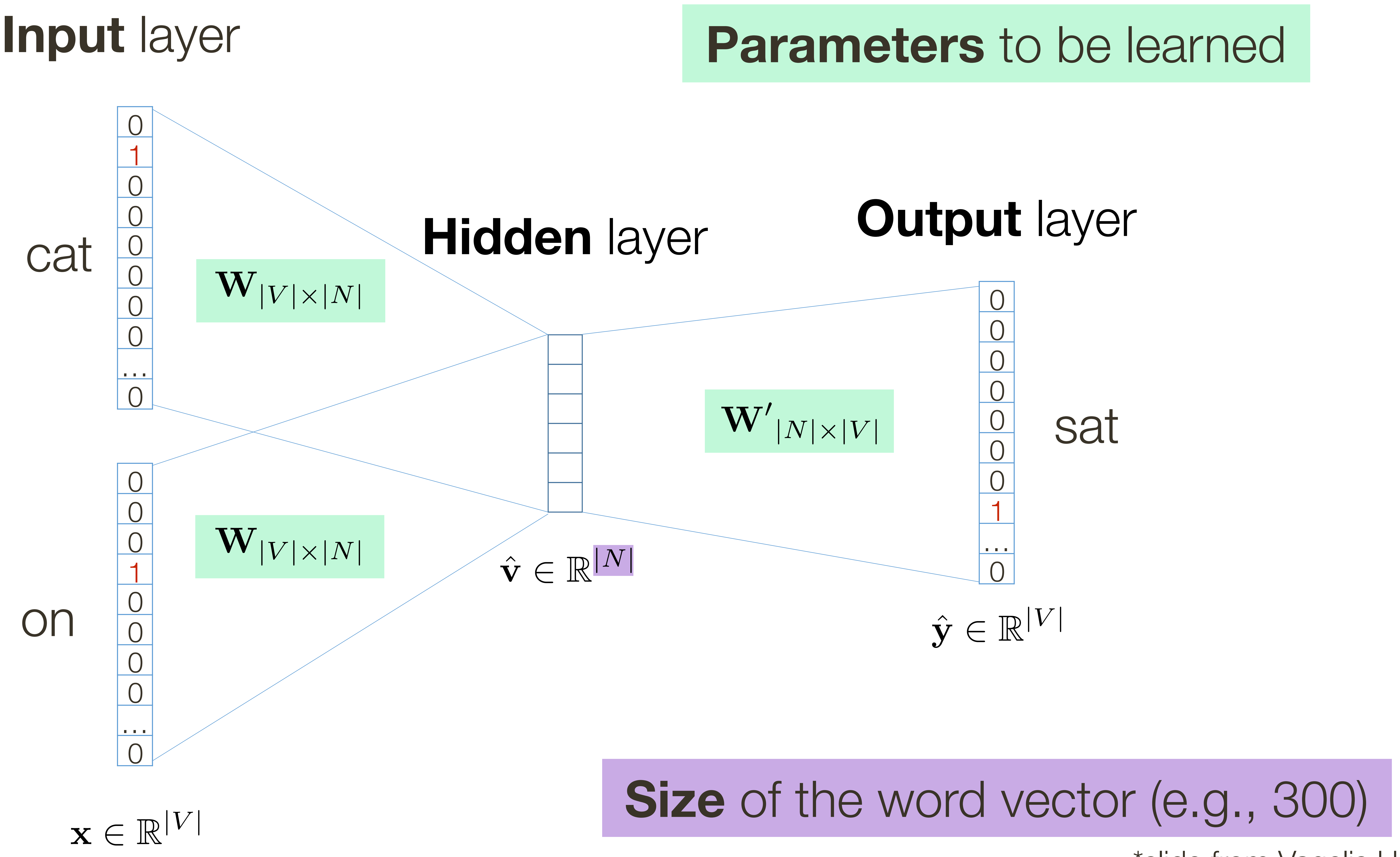
[Mikolov et al., 2013]



*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

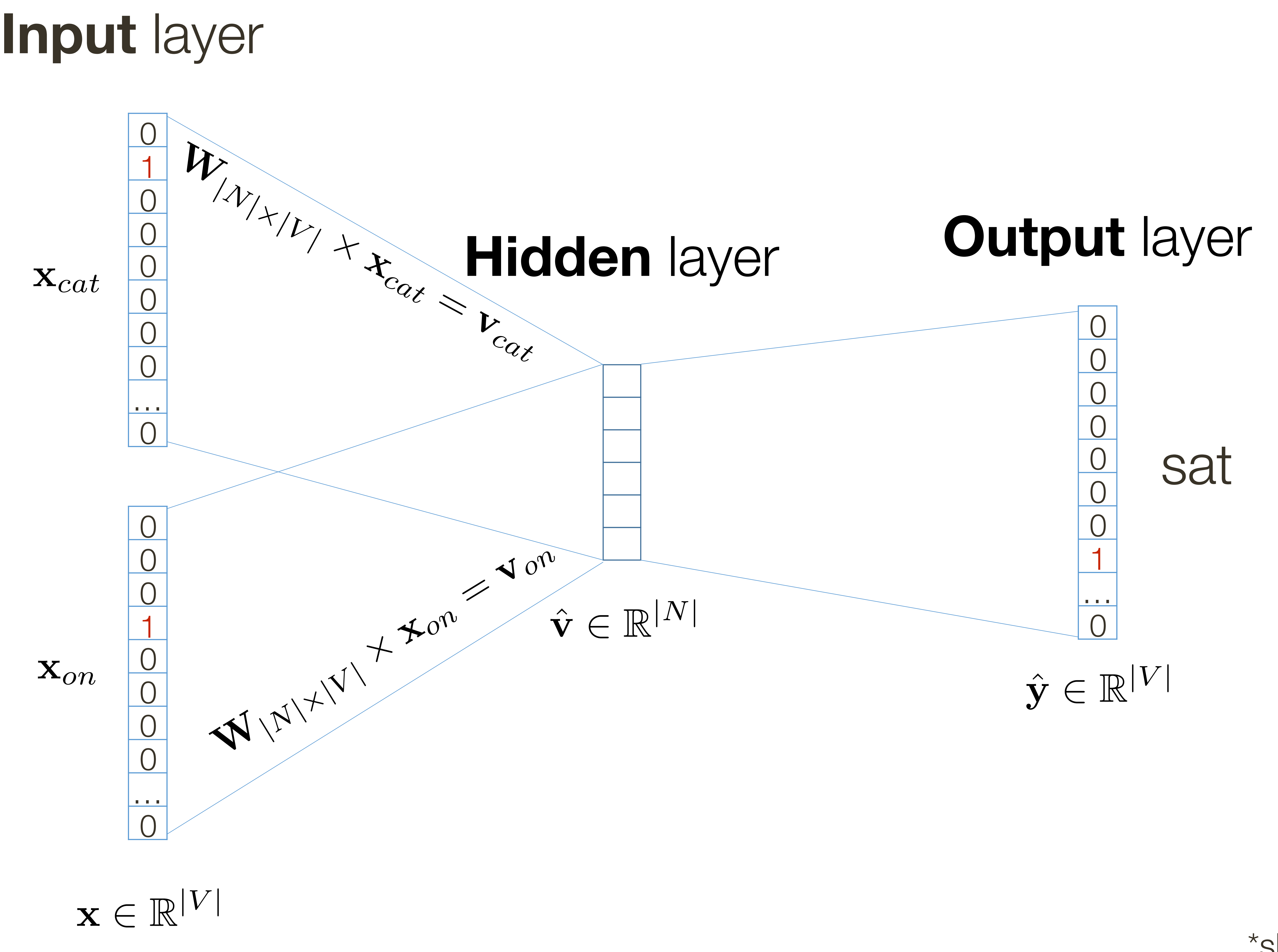
[Mikolov et al., 2013]



*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

[Mikolov et al., 2013]

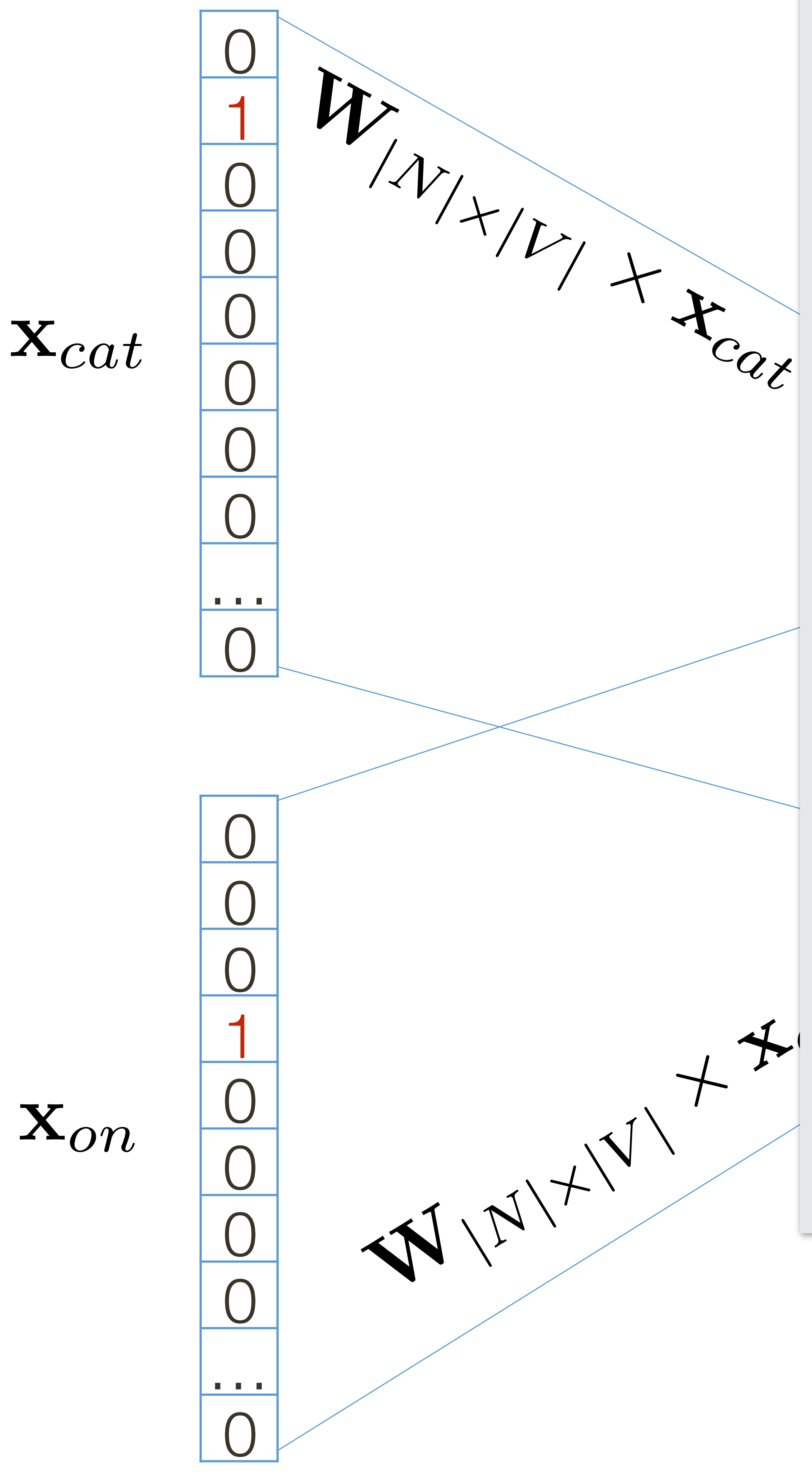


*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

[Mikolov et al., 2013]

Input layer



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

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

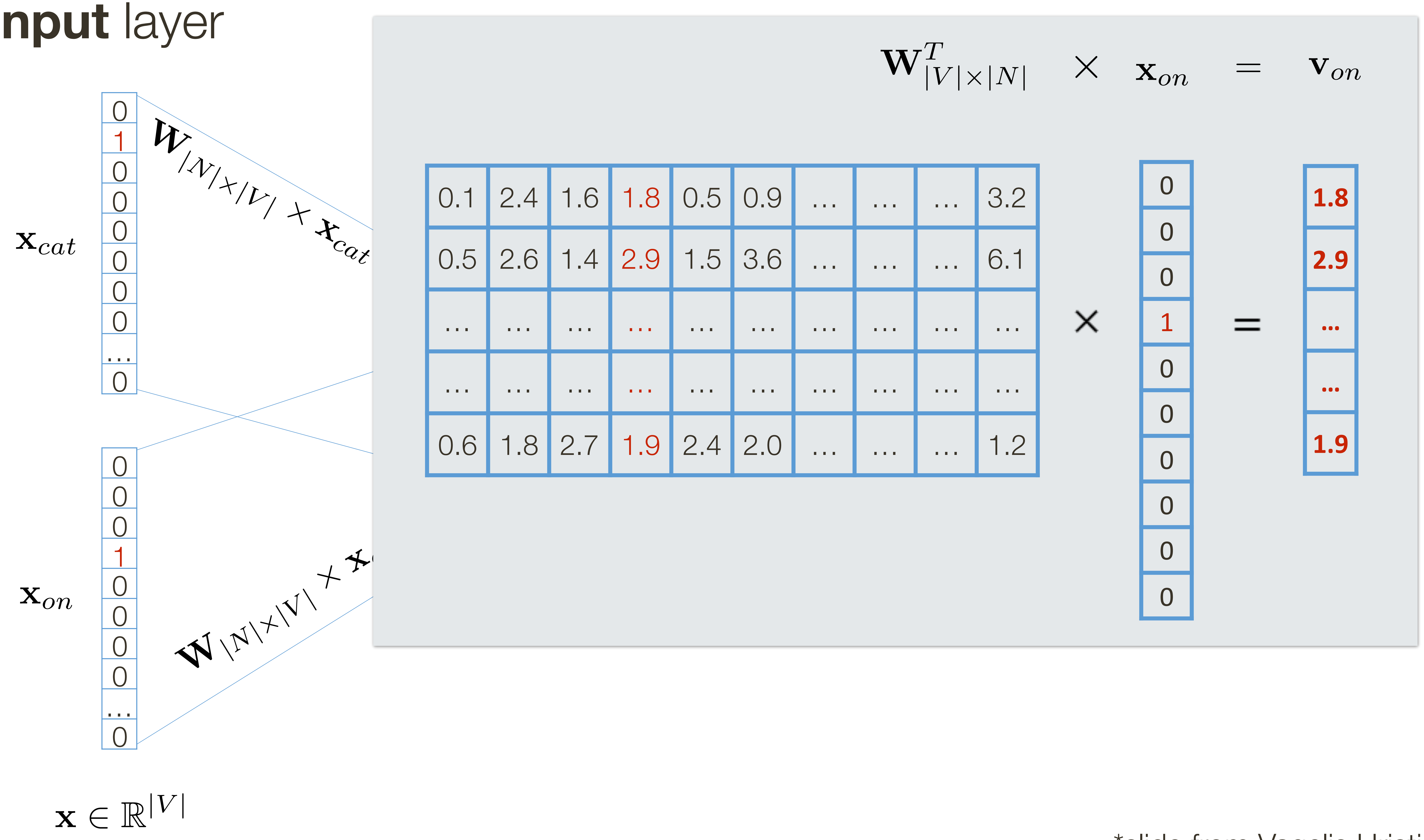
$\mathbf{x} \in \mathbb{R}^{|V|}$

*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

[Mikolov et al., 2013]

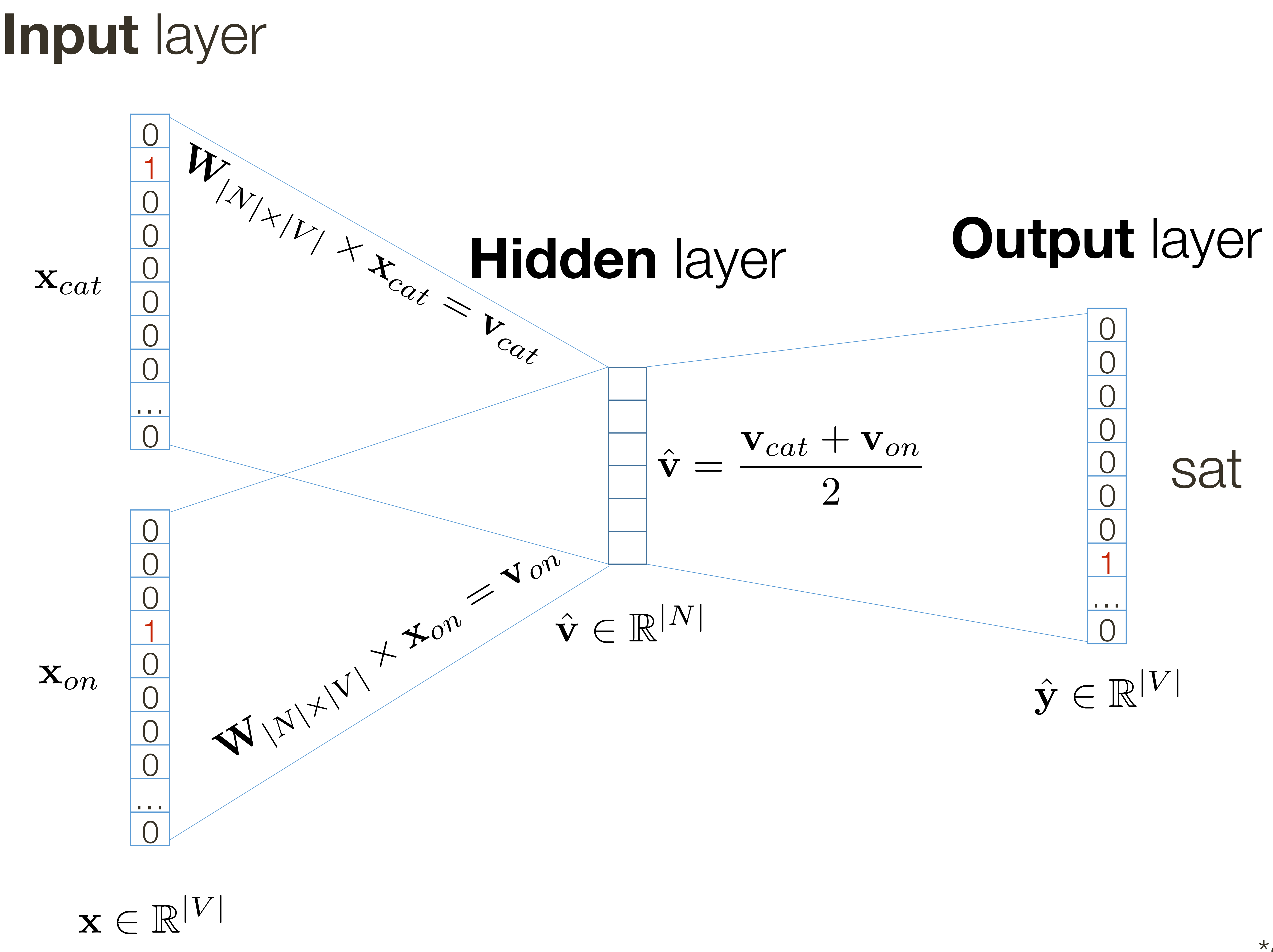
Input layer



*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

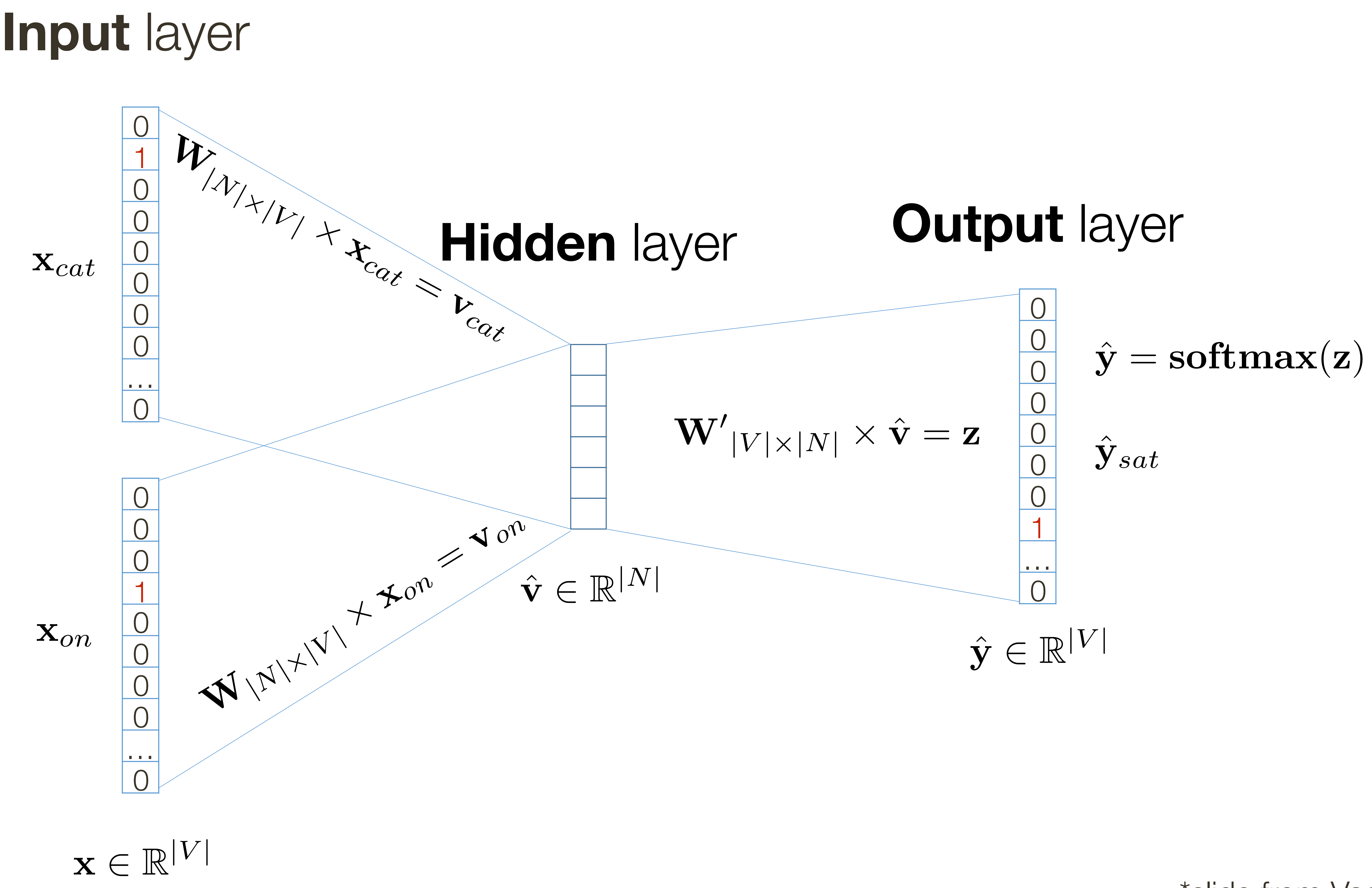
[Mikolov et al., 2013]



*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

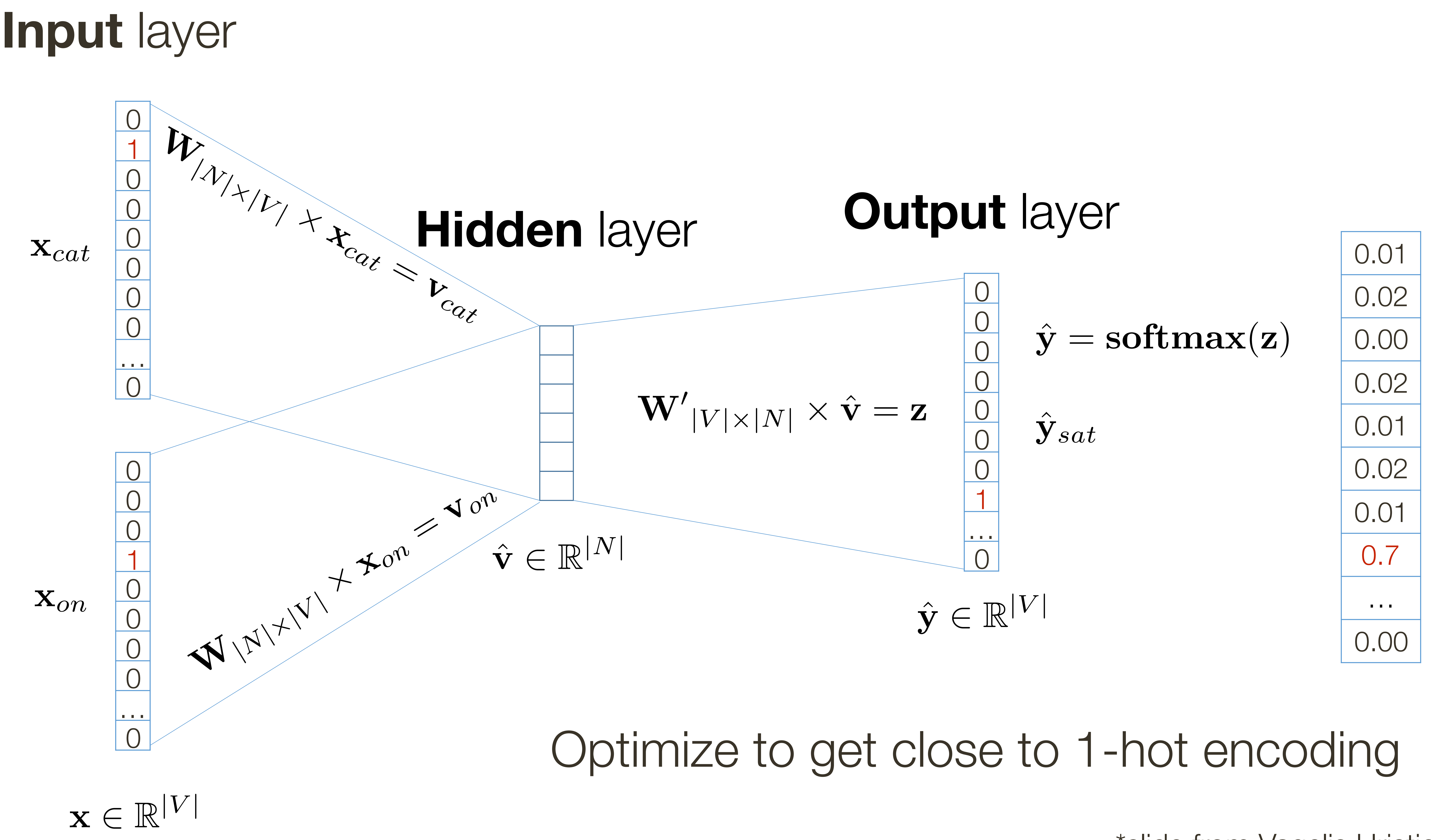
[Mikolov et al., 2013]



*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

[Mikolov et al., 2013]

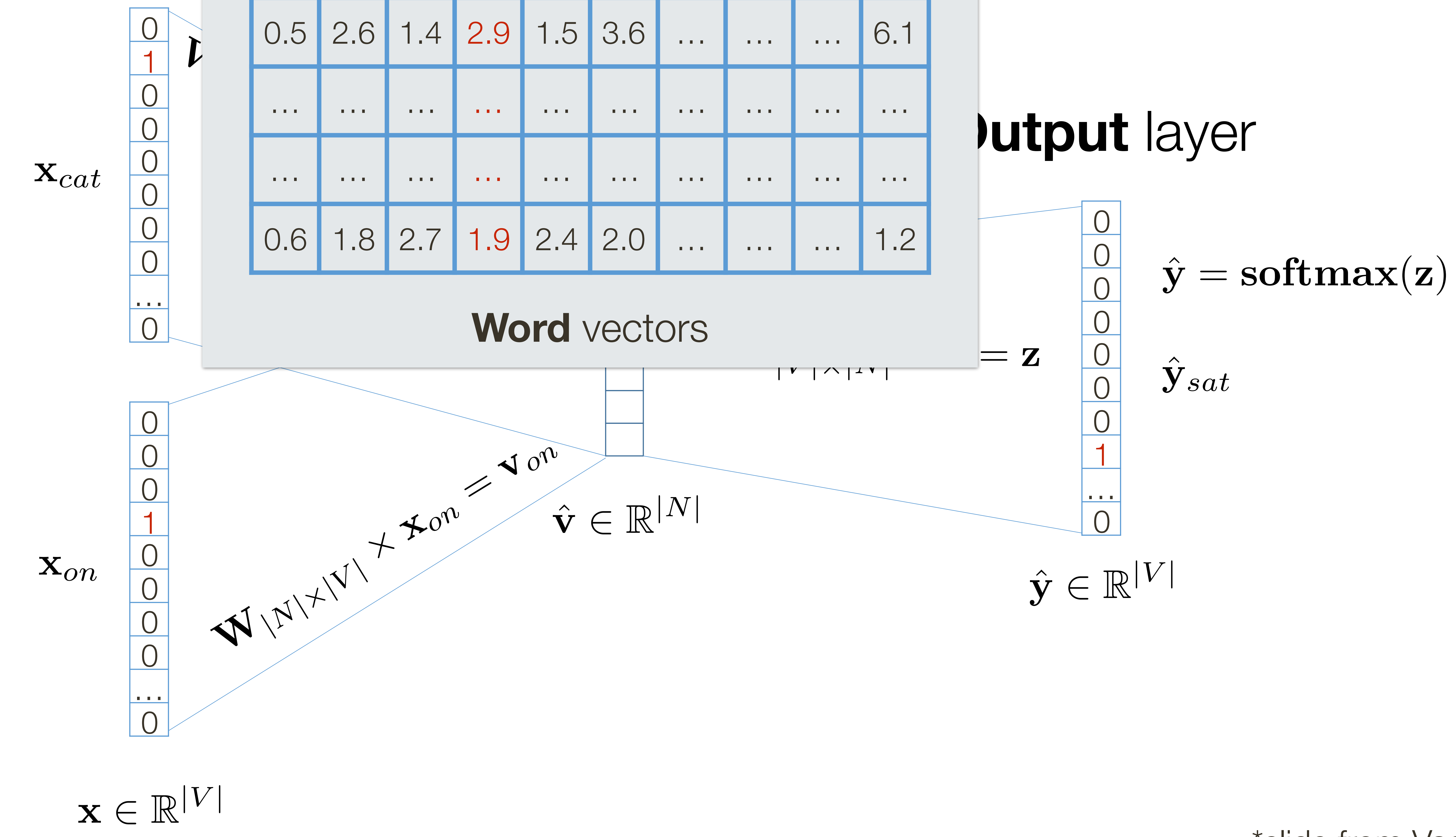


Optimize to get close to 1-hot encoding

*slide from Vagelis Hristidis

CBOW: Continuous Bag of Words

[Mikolov et al., 2013]



$\mathbf{W}^T_{|V|\times|N|}$

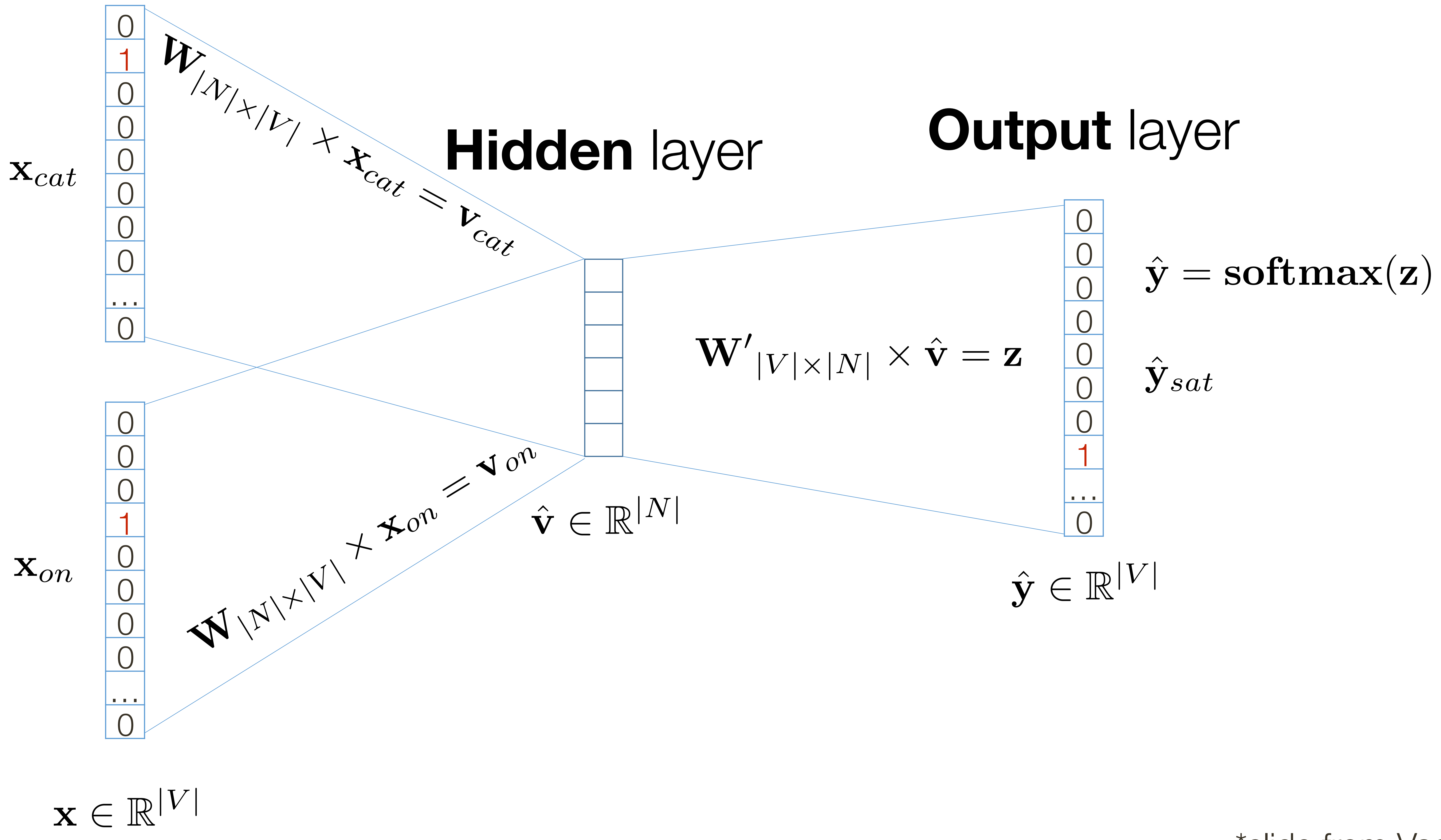
0.1	2.4	1.6	1.8	0.5	0.9	…	…	…	3.2
0.5	2.6	1.4	2.9	1.5	3.6	…	…	…	6.1
…	…	…	…	…	…	…	…	…	…
…	…	…	…	…	…	…	…	…	…
0.6	1.8	2.7	1.9	2.4	2.0	…	…	…	1.2

Word vectors

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]

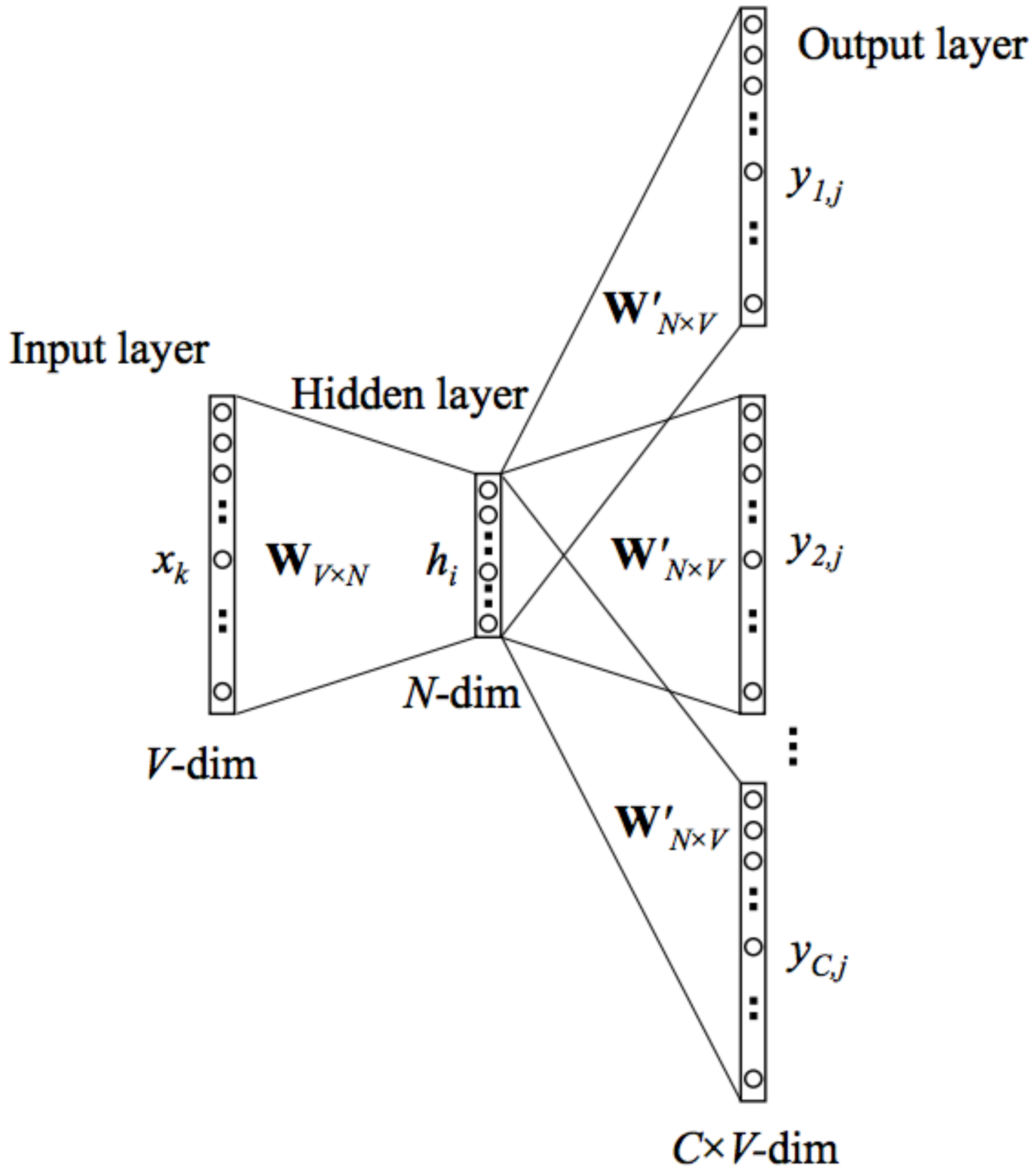
Another way to look at it: Maximize similarity between context word representation and the word representation itself

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

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

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**

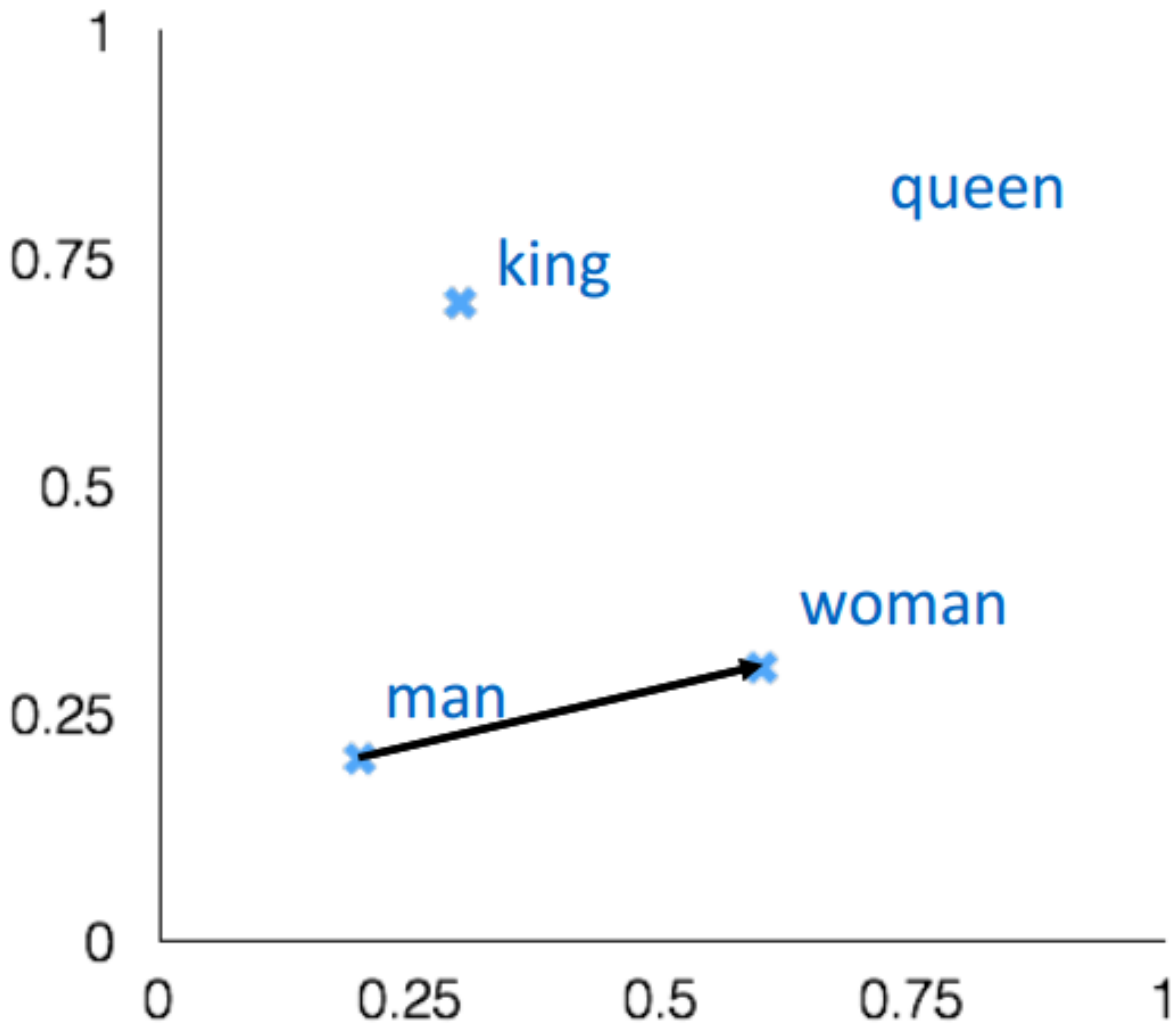
Interesting Results: **Word Analogies**

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

a:b :: c:? → $$d = \arg\max_x \frac{(w_b - w_a + w_c)^T w_x}{||w_b - w_a + w_c||}$$

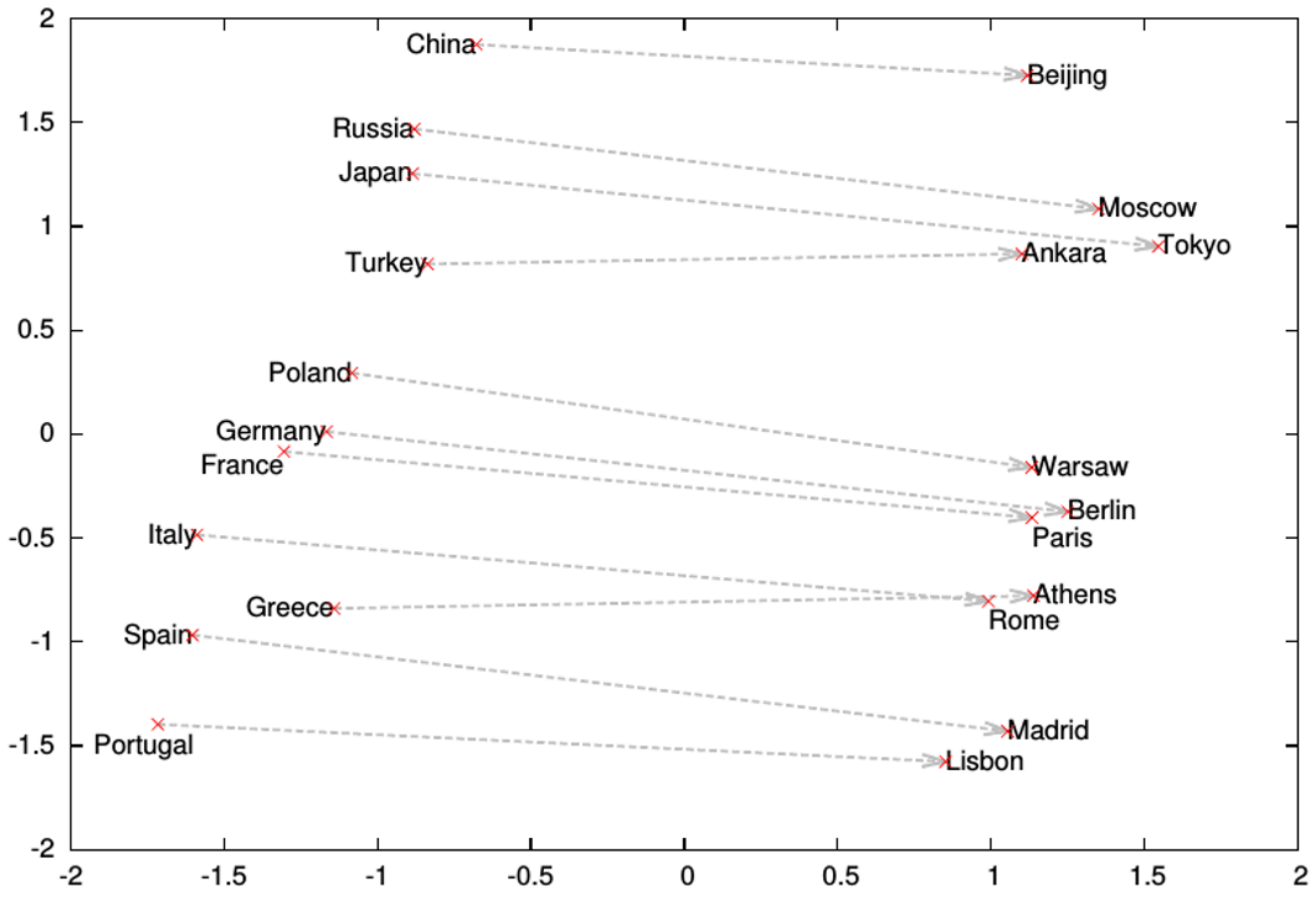
man:woman :: king:?

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

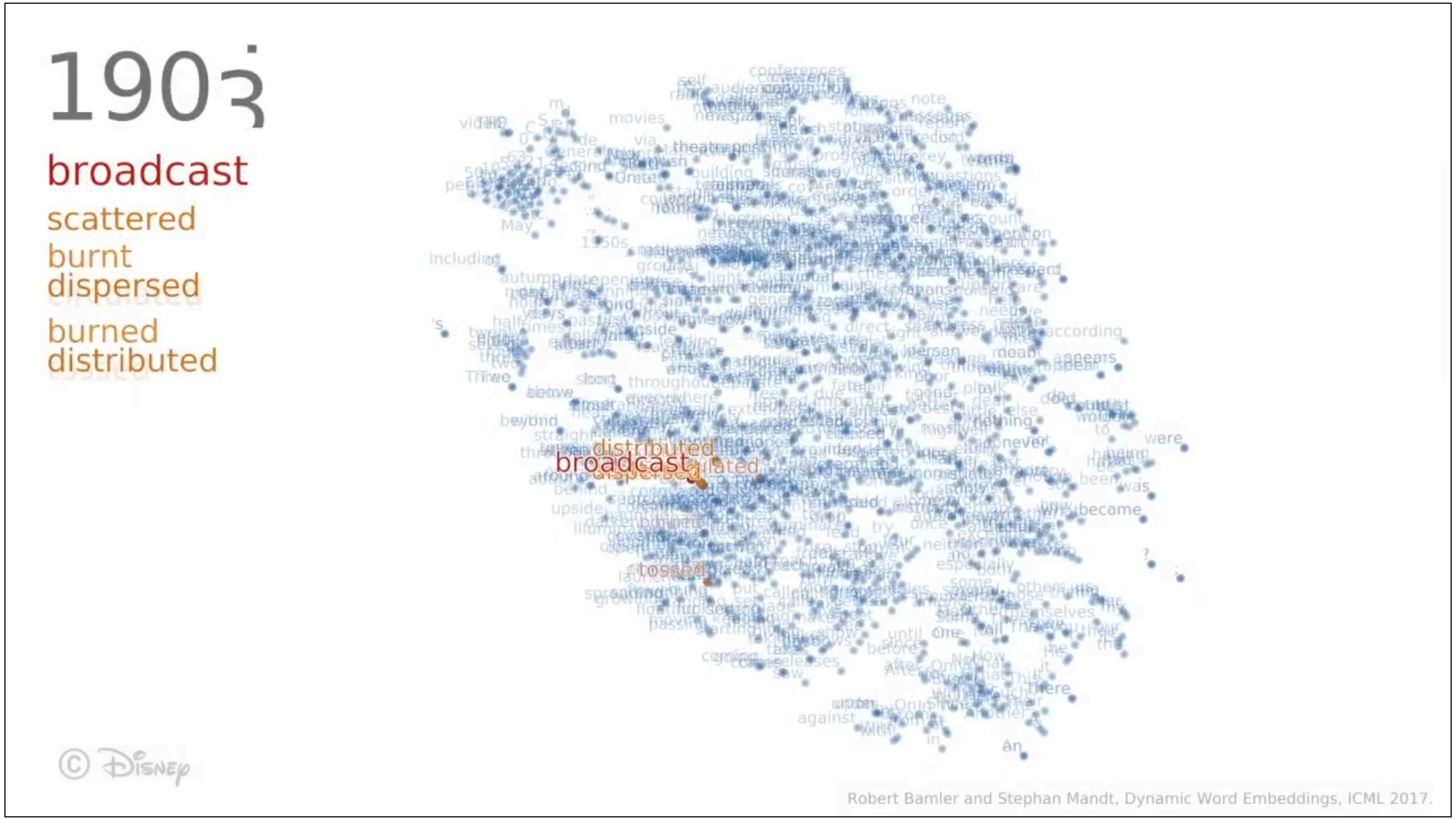


Interesting Results: **Word Analogies**

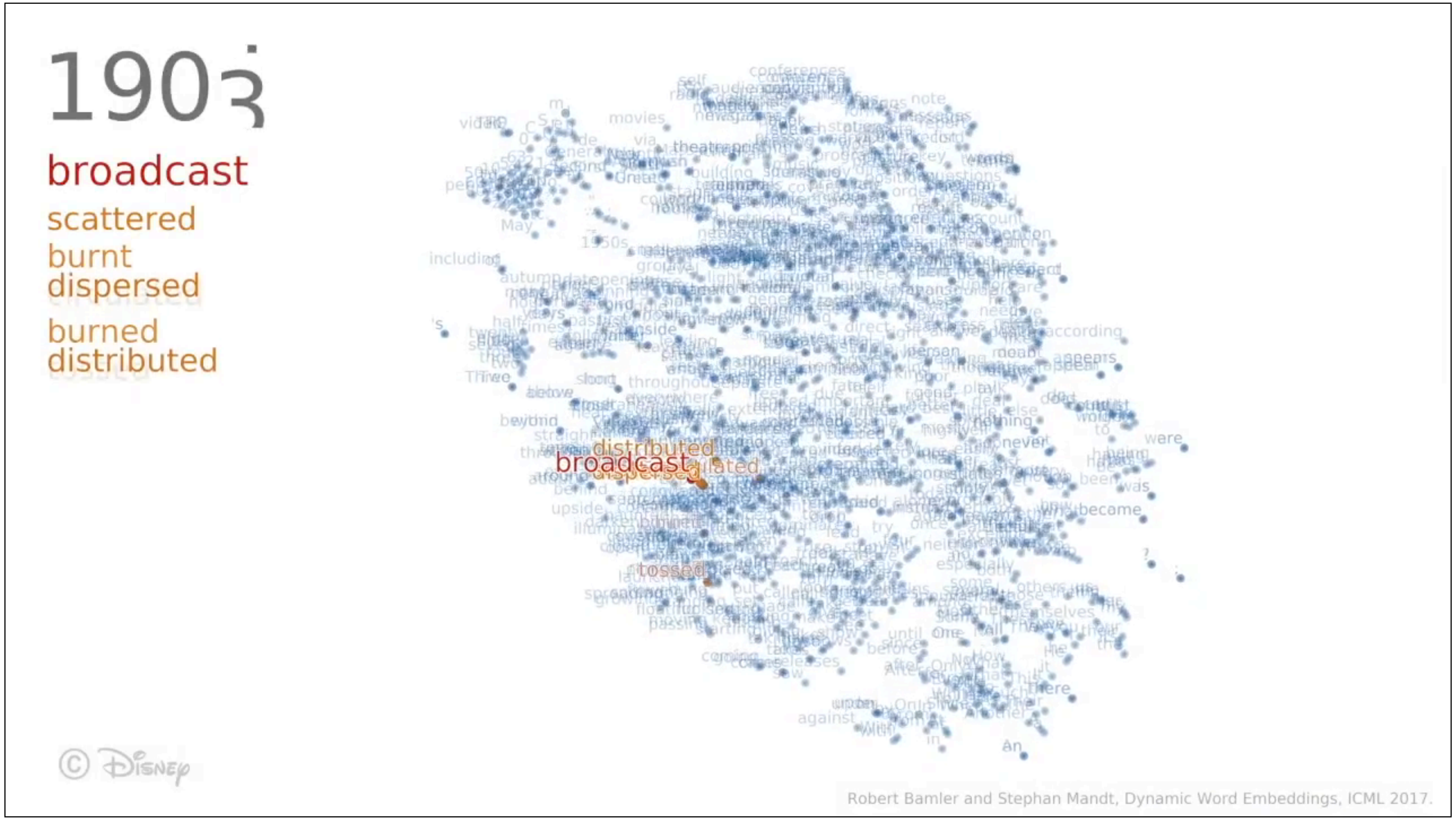
[Mikolov et al., 2013]



Dynamic Word Embeddings



Dynamic Word Embeddings



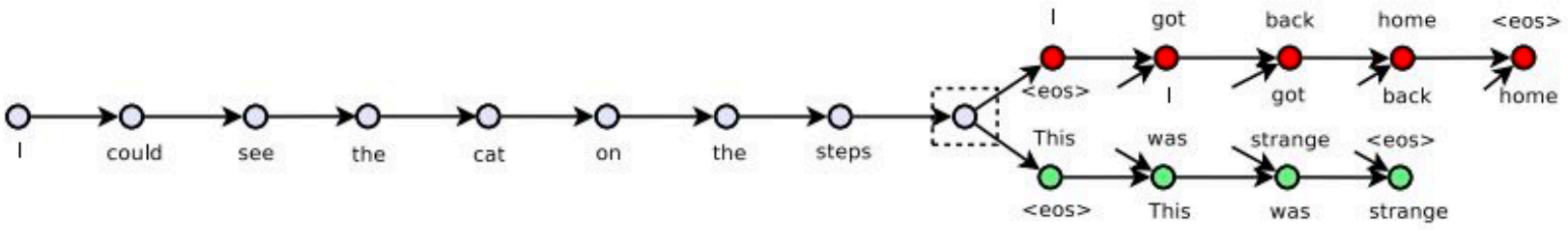
Topics in AI (CPSC 532S):
Multimodal Learning with Vision, Language and Sound

Lecture 11: RNN Applications

Let us look at some actual practical uses of RNNs

Applications: Skip-thought Vectors

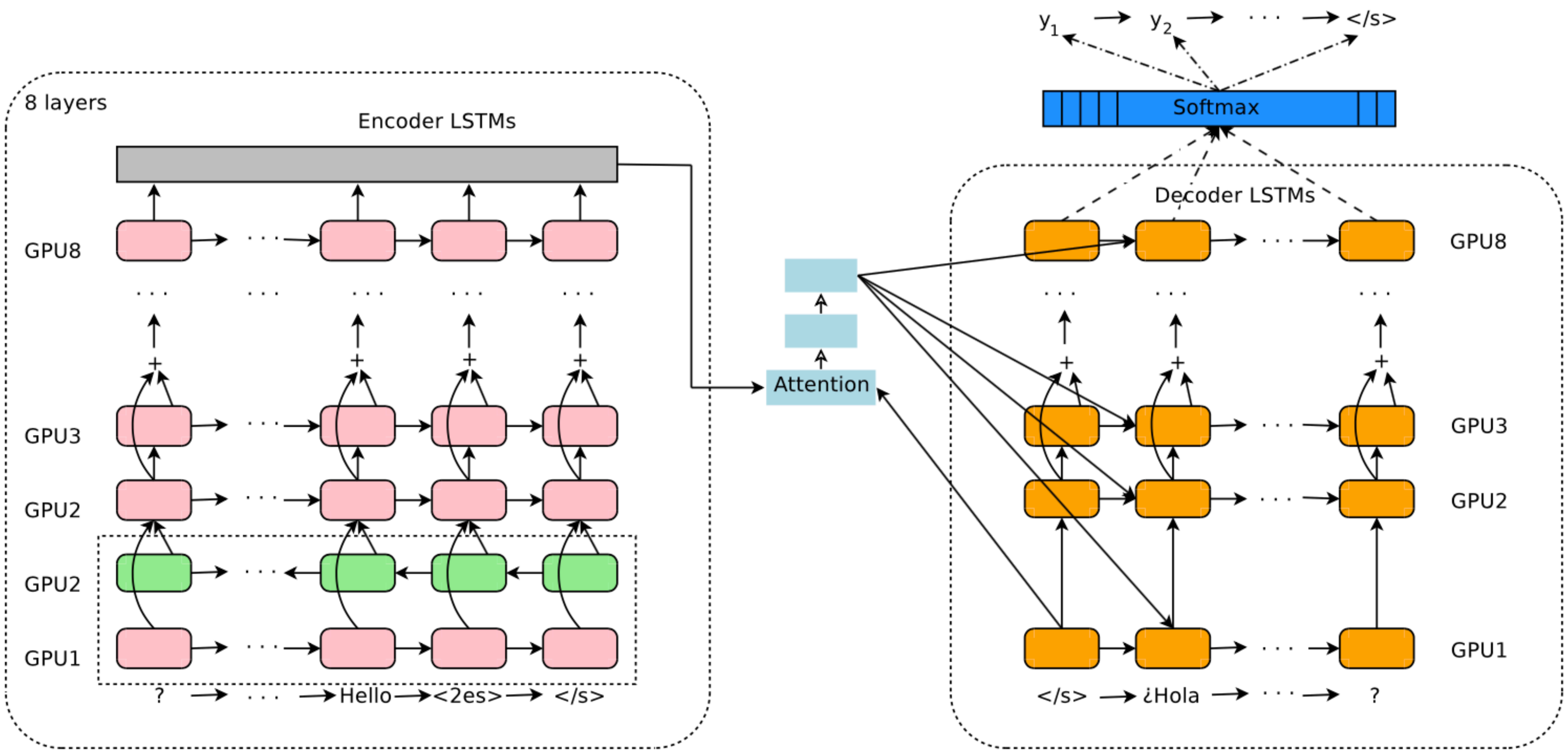
word2vec but for sentences, where each sentence is processed by an LSTM



[Kiros et al., 2015]

Applications: Google Language Translation

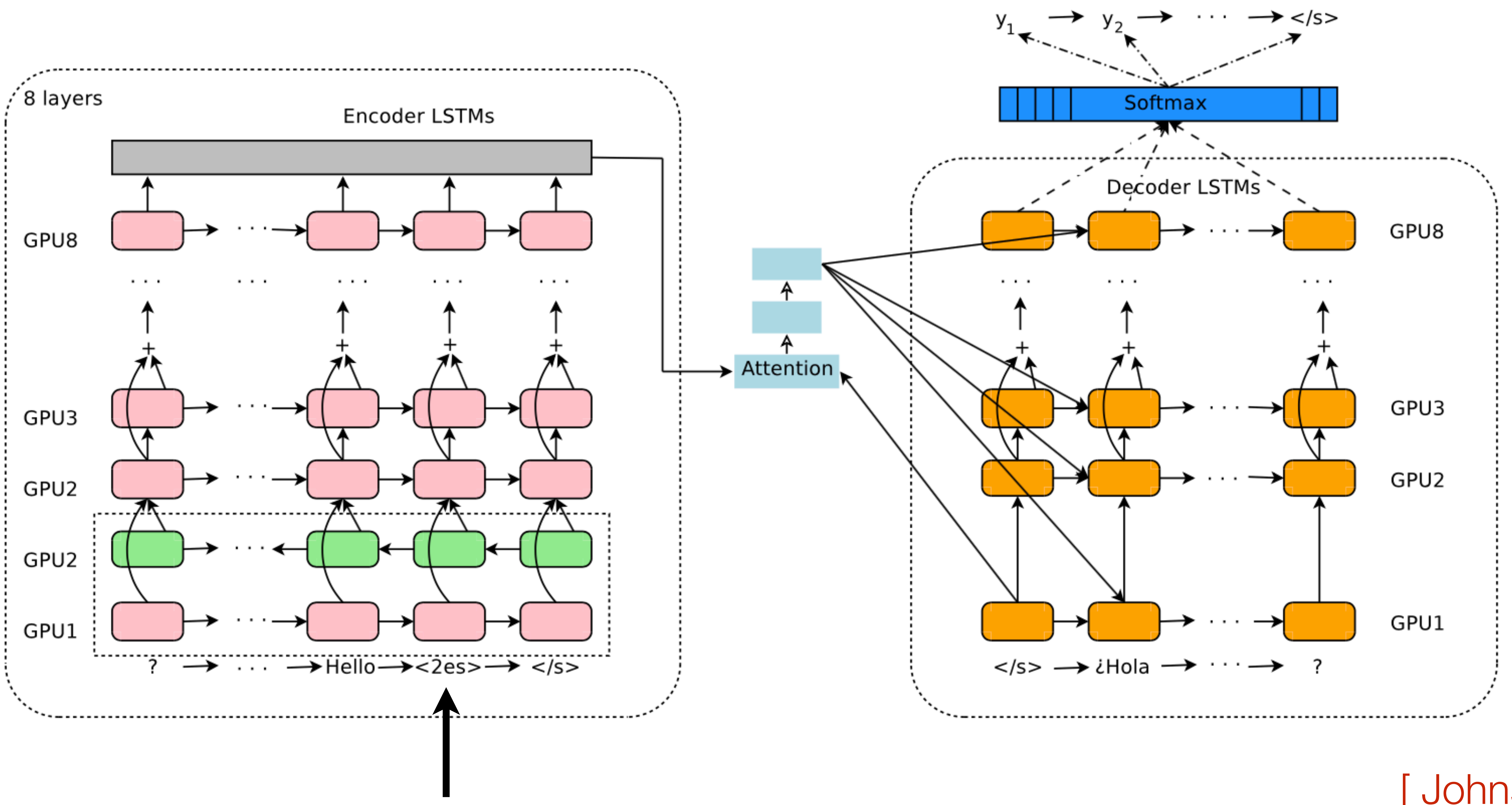
One model to translate from **any language** to any other language



[Johnson et al., 2017]

Applications: Google Language Translation

One model to translate from **any language** to any other language

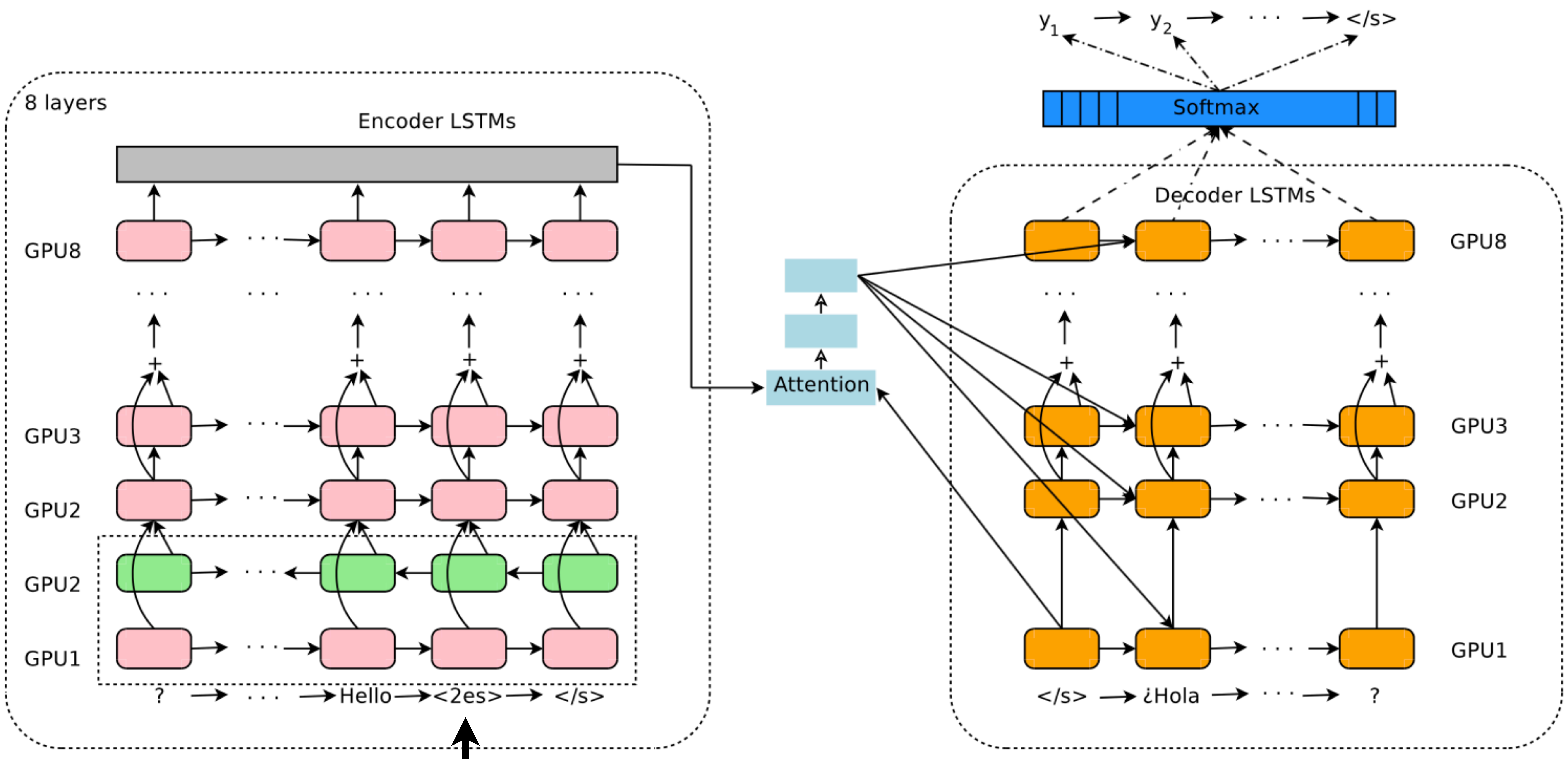


Token designating
target language

[Johnson et al., 2017]

Applications: Google Language Translation

One model to translate from **any language** to any other language



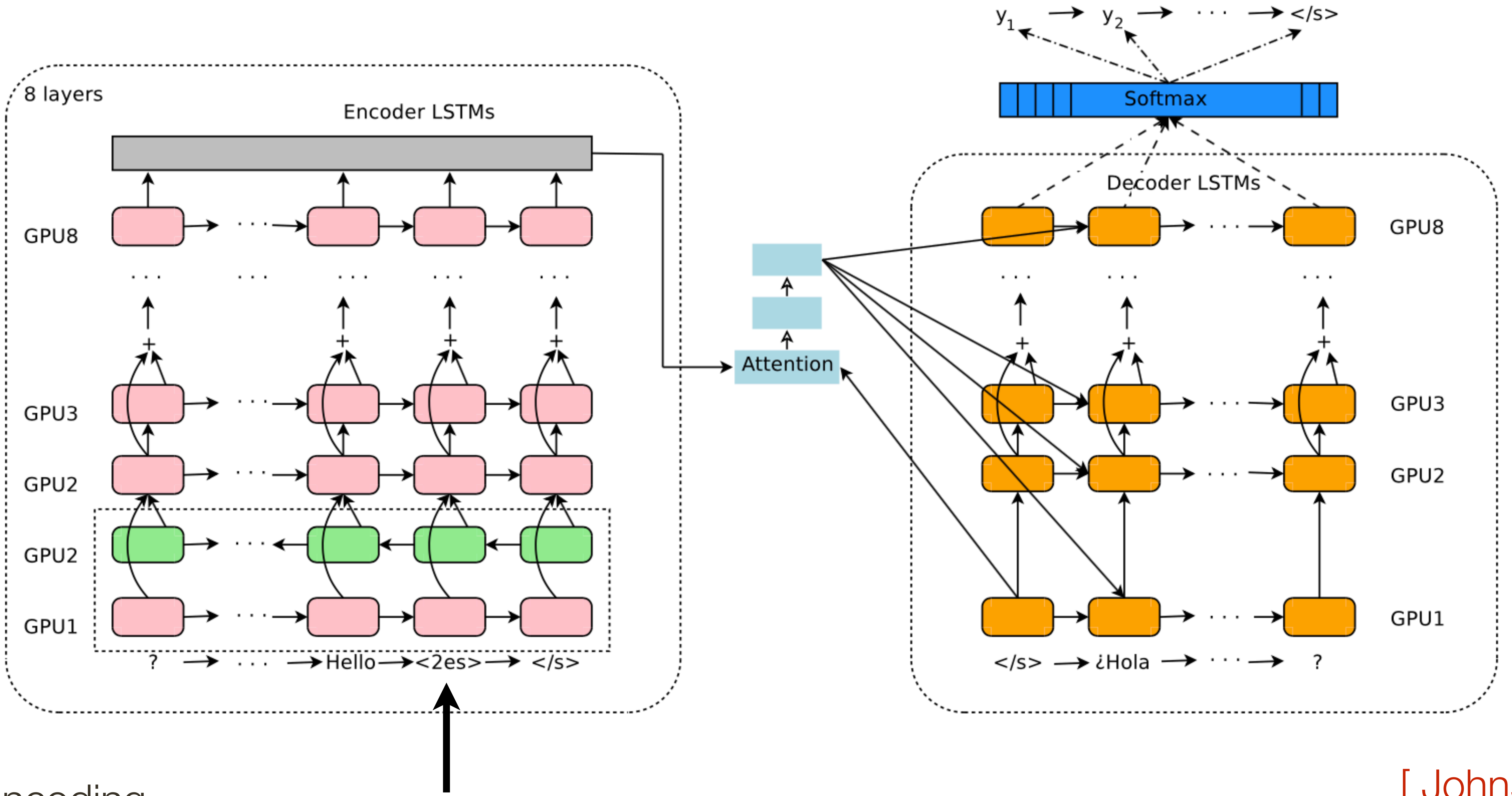
Flipped order encoding

Token designating **target** language

[Johnson et al., 2017]

Applications: Google Language Translation

One model to translate from **any language** to any other language



Flipped order encoding

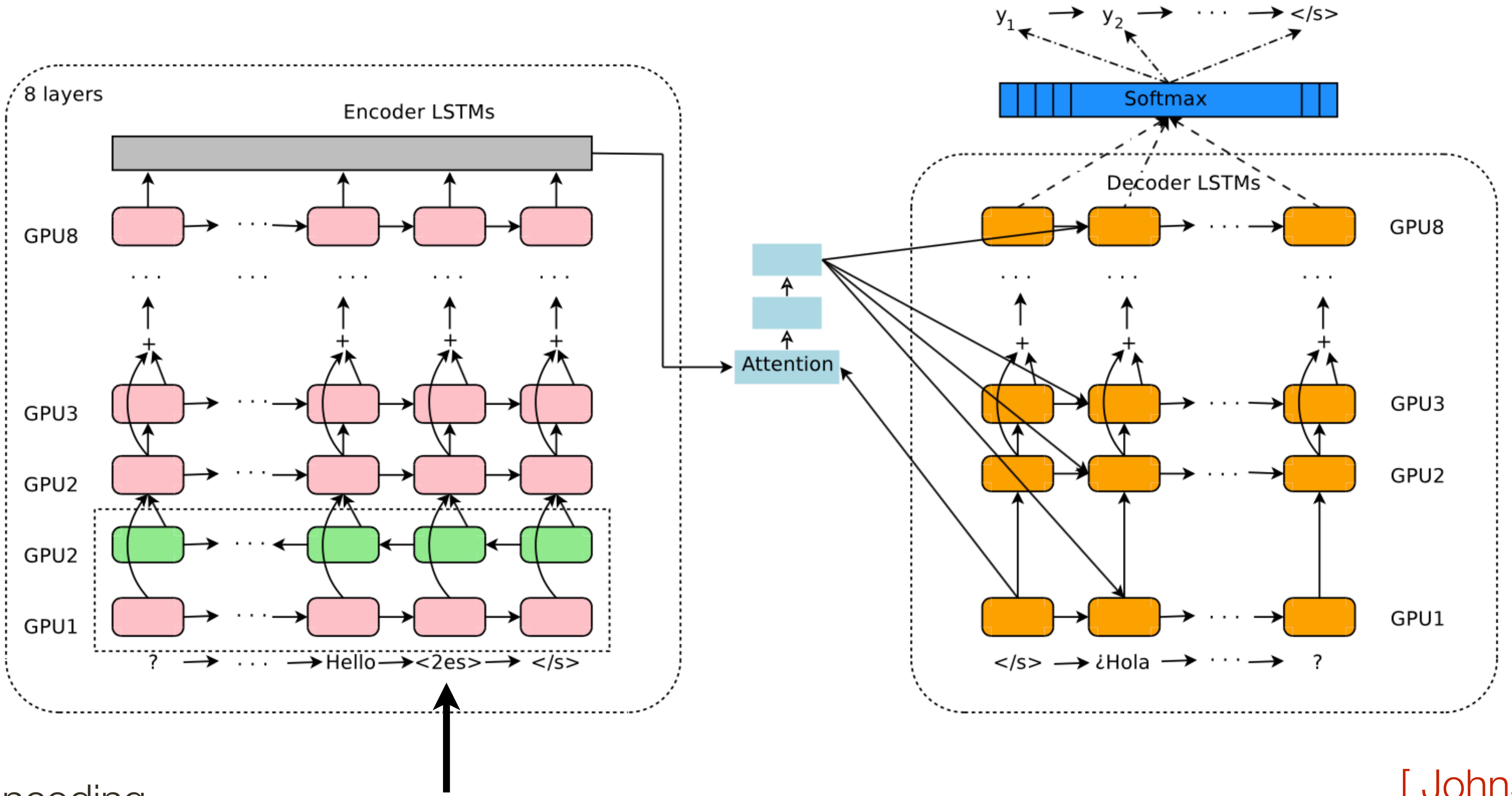
Why?

Token designating **target** language

[Johnson et al., 2017]

Applications: Google Language Translation

One model to translate from **any language** to any other language



Flipped order encoding

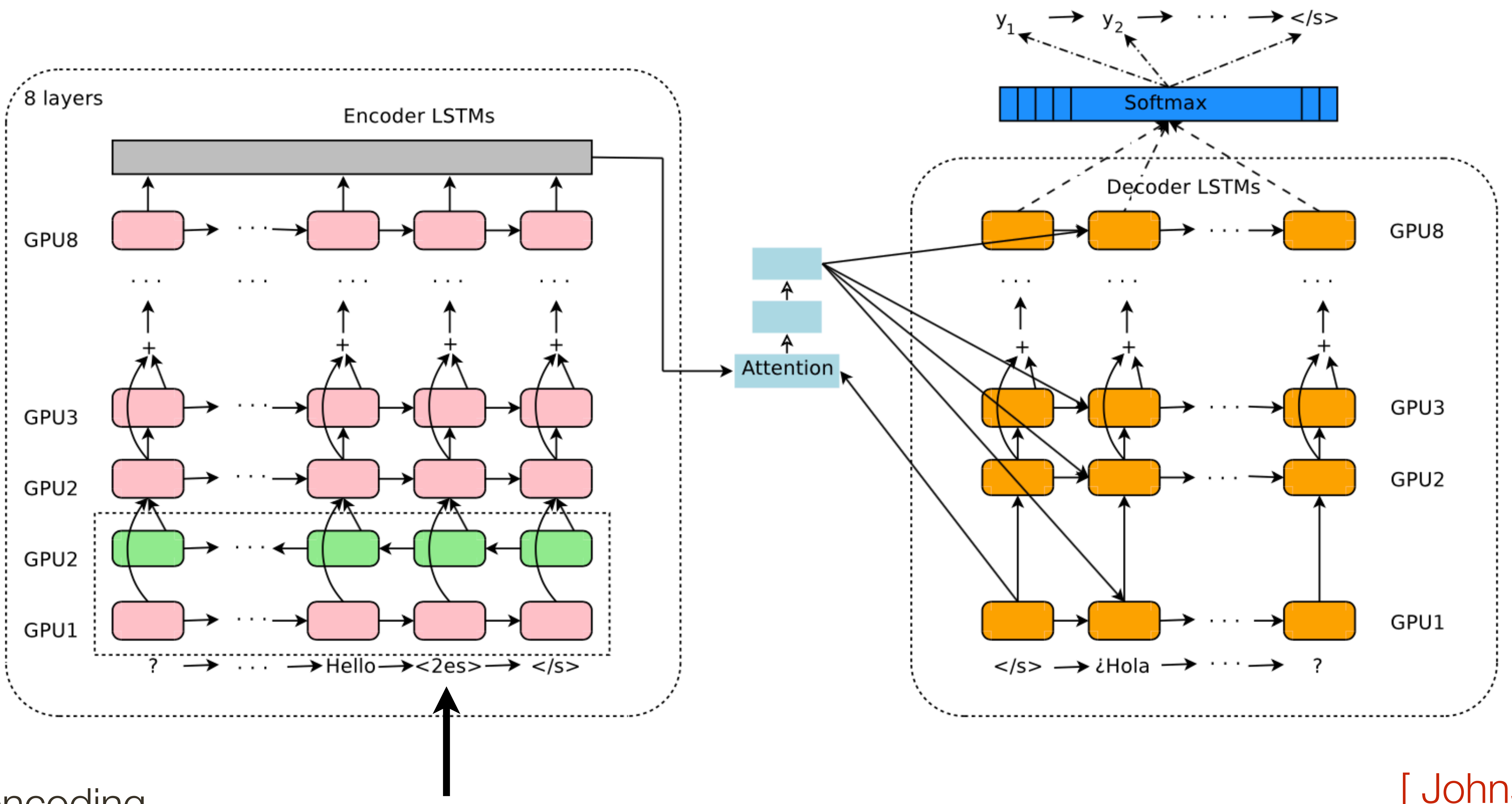
Token designating **target** language

[Johnson et al., 2017]

8! layer LSTM decoder and encoder

Applications: Google Language Translation

One model to translate from **any language** to any other language



Residual at other layers (ResNet style)

Bi-directional at lower layers

Flipped order encoding

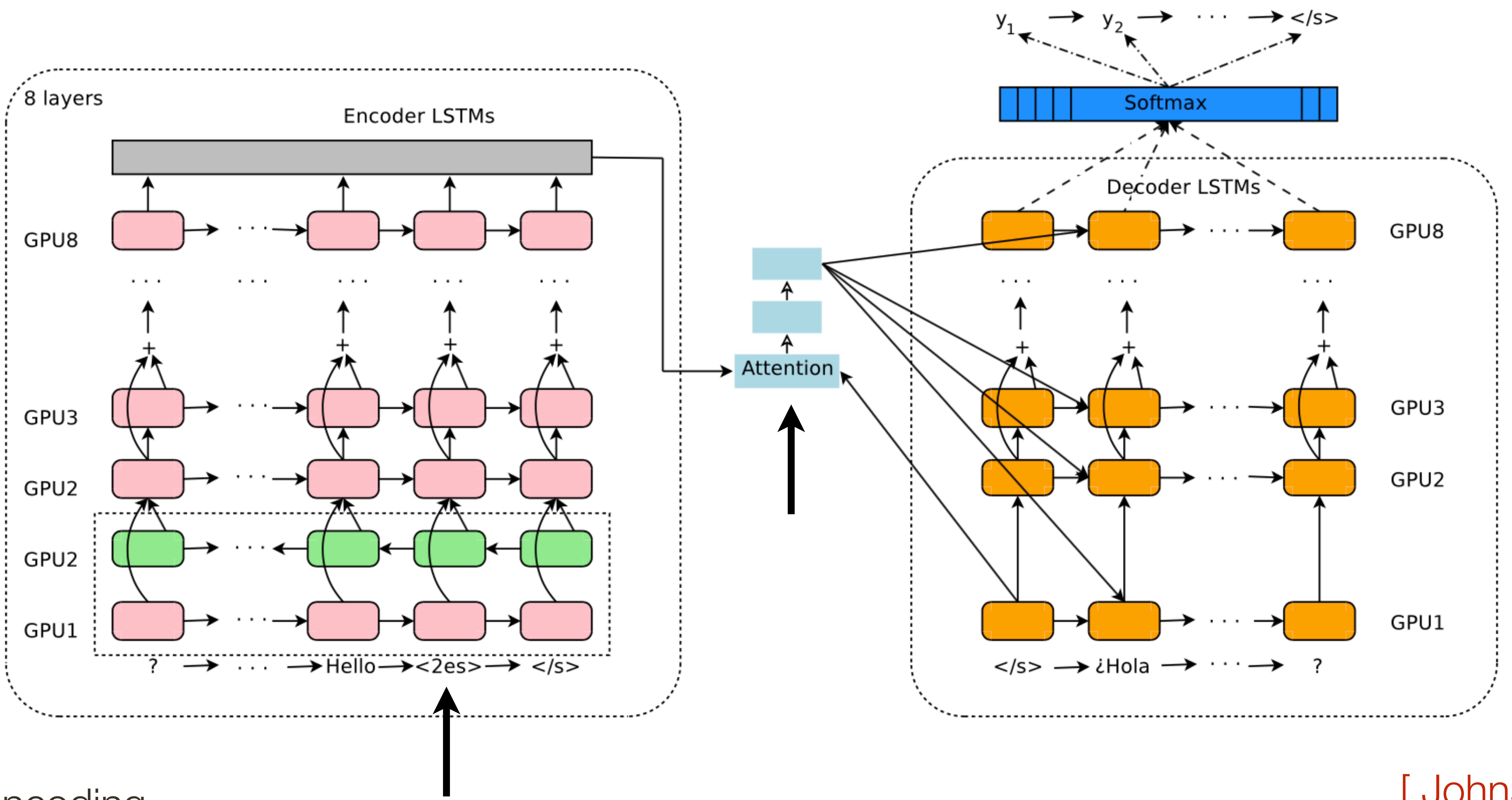
Token designating **target** language

[Johnson et al., 2017]

8! layer LSTM decoder and encoder

Applications: Google Language Translation

One model to translate from **any language** to any other language



Residual at other layers (ResNet style)

Bi-directional at lower layers

Flipped order encoding

Token designating **target** language

[Johnson et al., 2017]

8! layer LSTM decoder and encoder

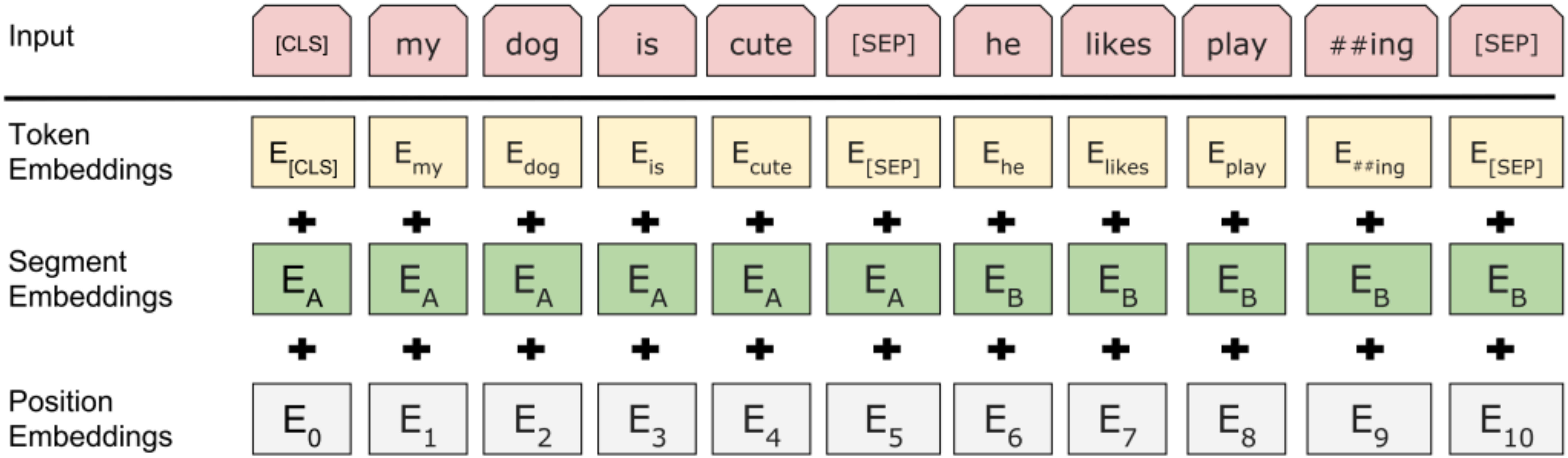
Applications: BERT and SoTA

To learn relationships between sentences, predict whether Sentence B is actual sentence that **proceeds** Sentence A, or a random sentence

Sentence A = The man went to the store.
Sentence B = He bought a gallon of milk.
Label = IsNextSentence

Sentence A = The man went to the store.
Sentence B = Penguins are flightless
Label = NotNextSentence

Applications: BERT and SoTA



Use 30,000 WordPiece **vocabulary**

Each token is a **sum of three** embeddings

Applications: BERT and SoTA

Multi-headed self **attention**

— Models context

Feed-forward layers

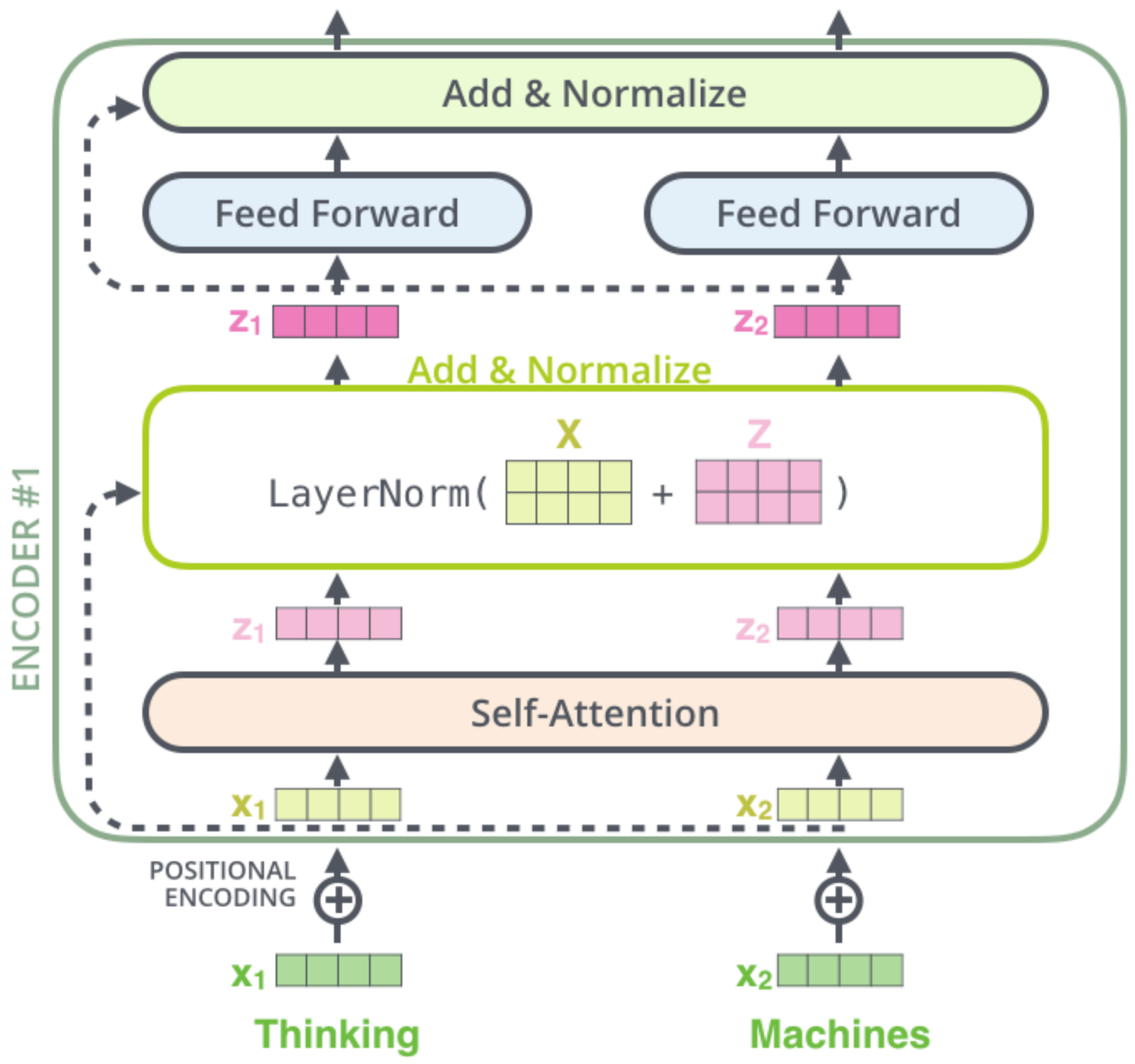
— Computes non-linear hierarchical features

Layer norm and **residuals**

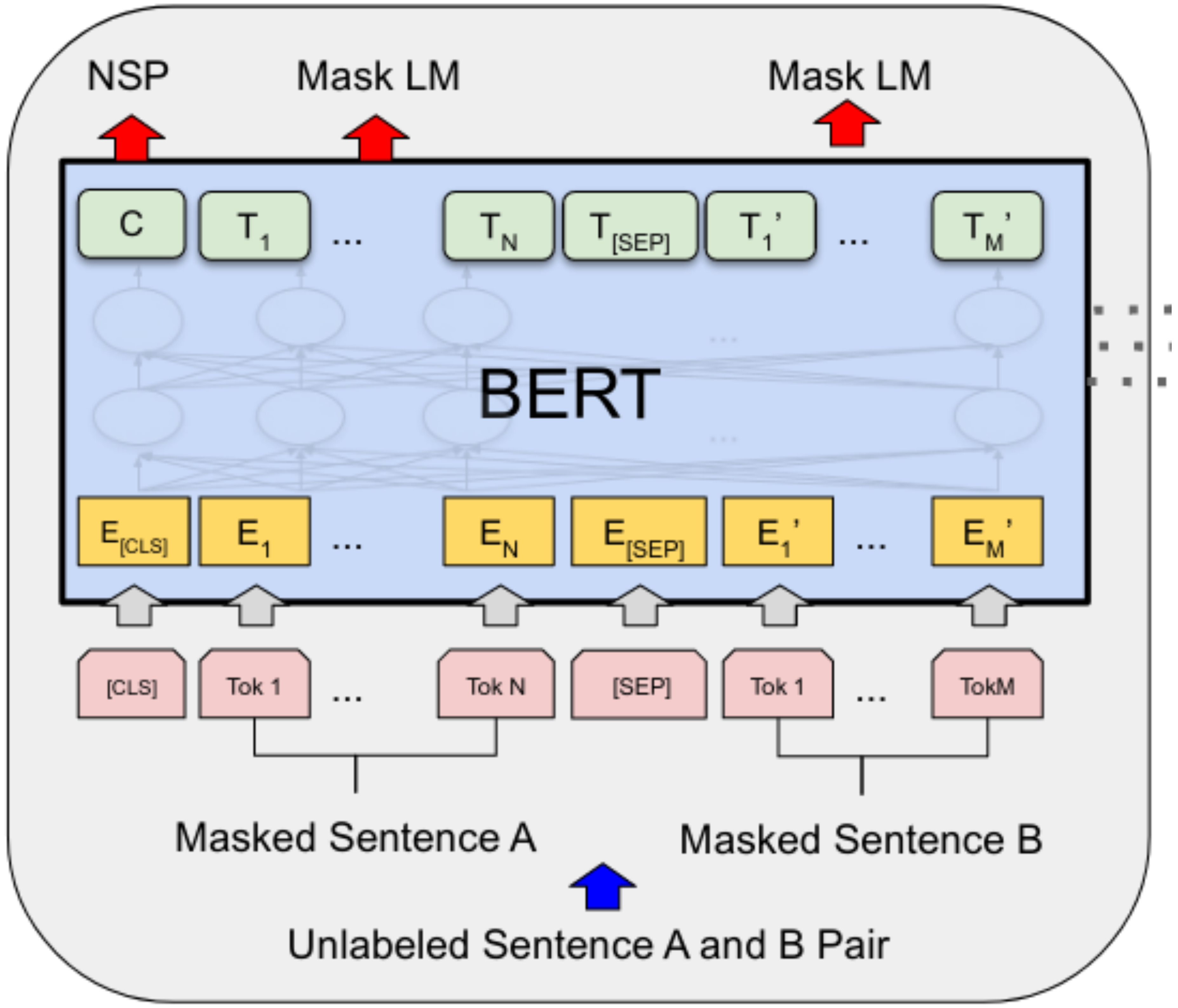
— Makes training deep neural network (e.g., 12 layers possible)

Positional Embeddings

— Allows model to learn relative positioning

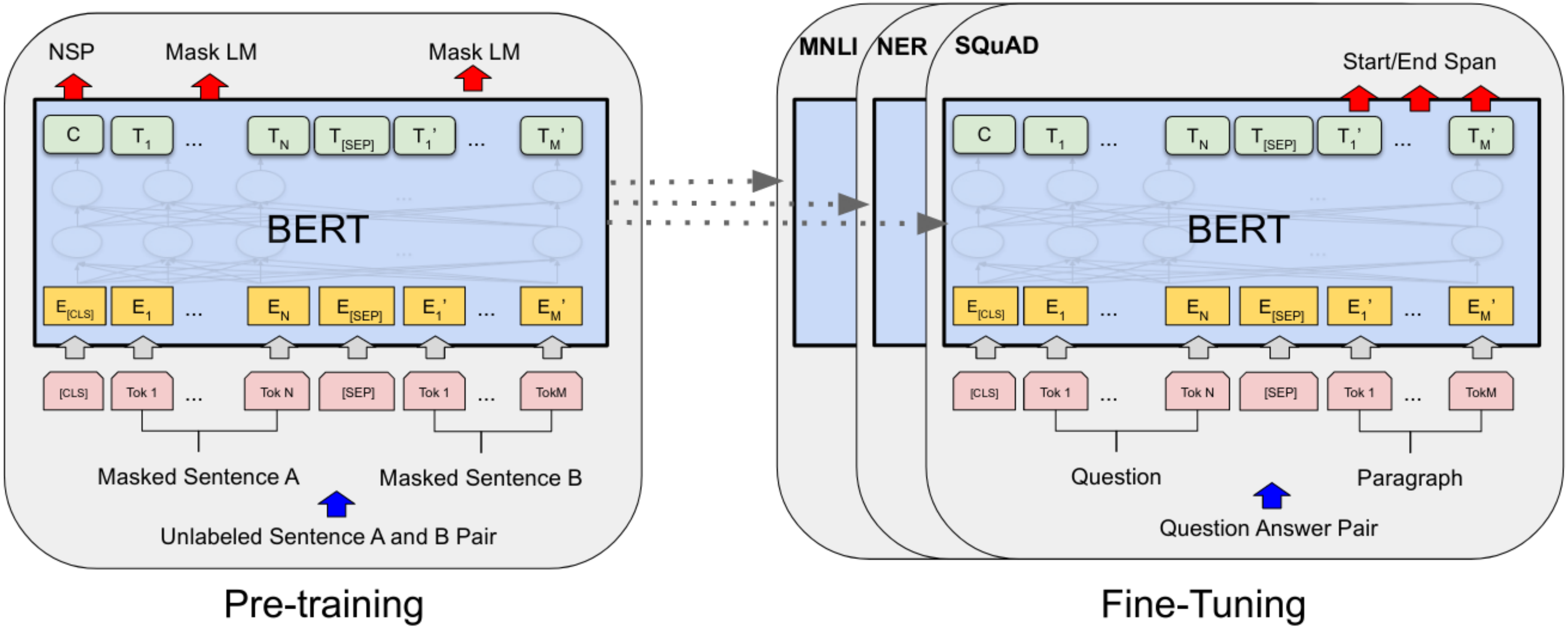


Applications: BERT and SoTA

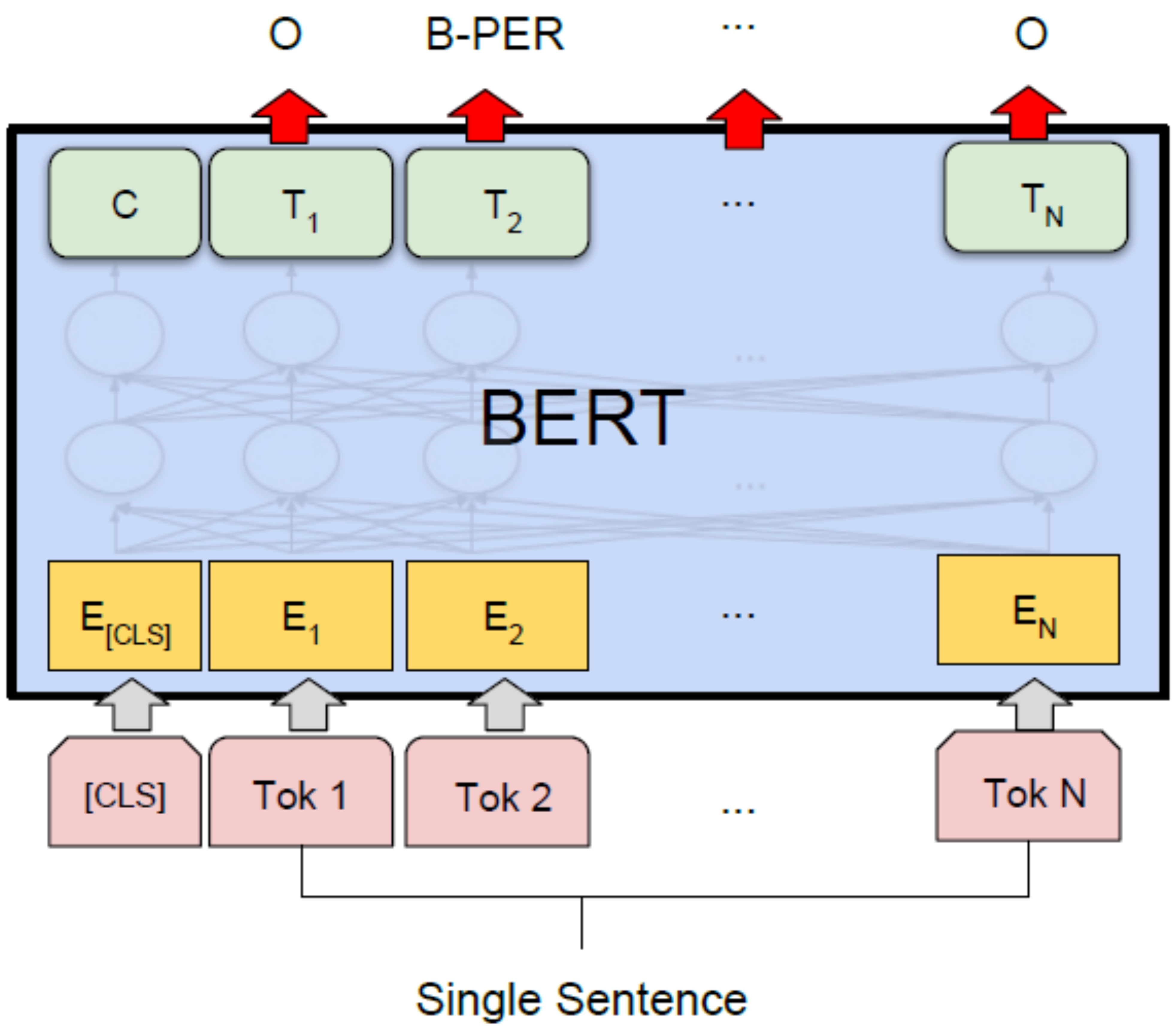


Pre-training

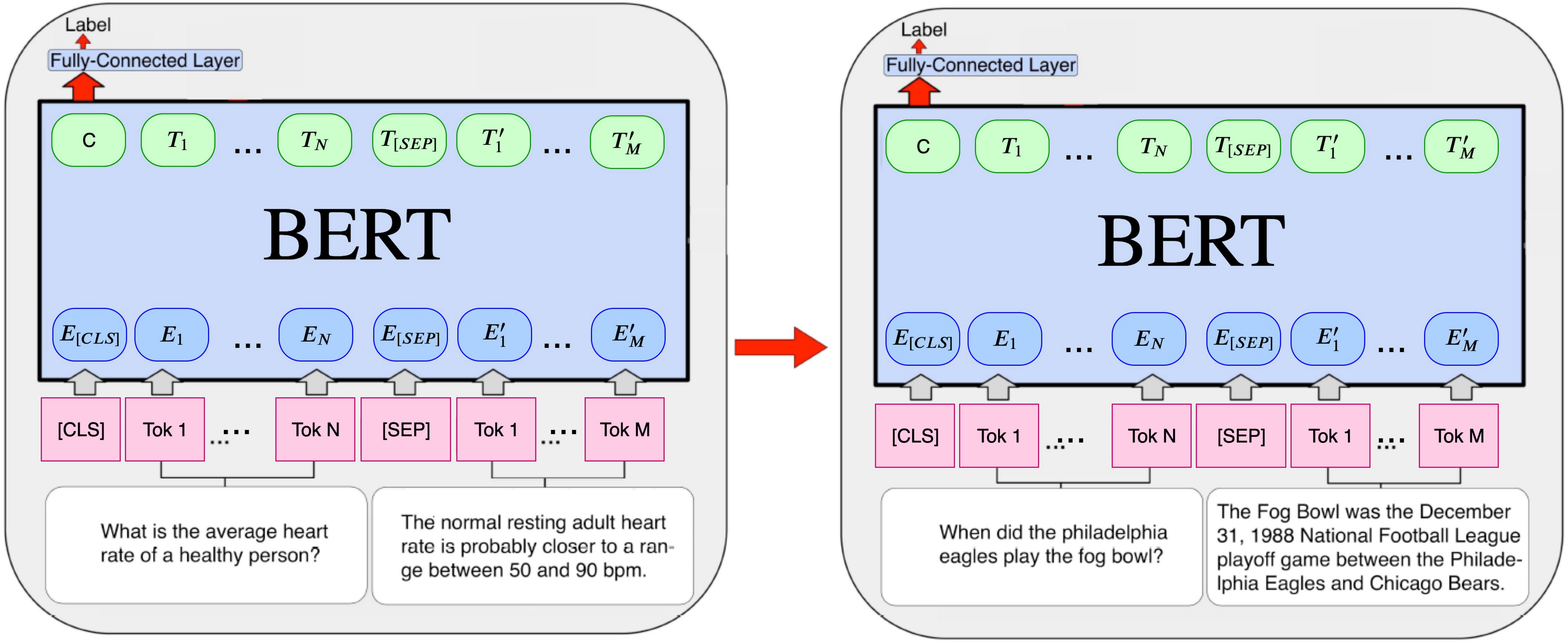
Applications: BERT and SoTA



Applications: BERT and SoTA



Applications: BERT and SoTA



Transfer: ASNQ Dataset

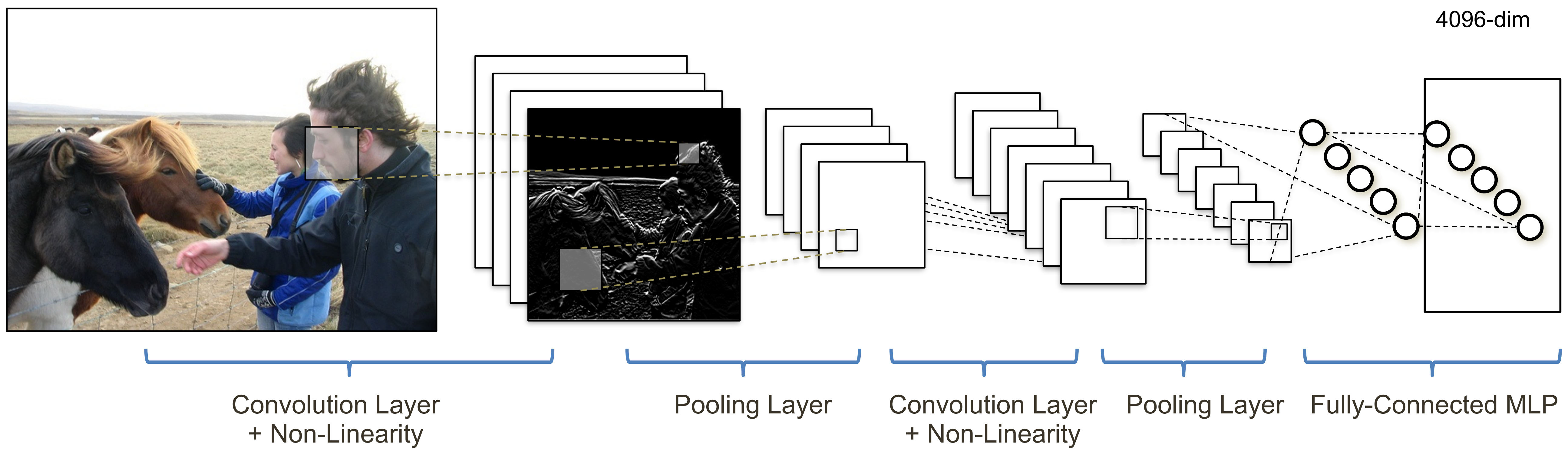
Adapt: Target Dataset

Applications: Neural Image Captioning

* slide from Dhruv Batra

Applications: Neural Image Captioning

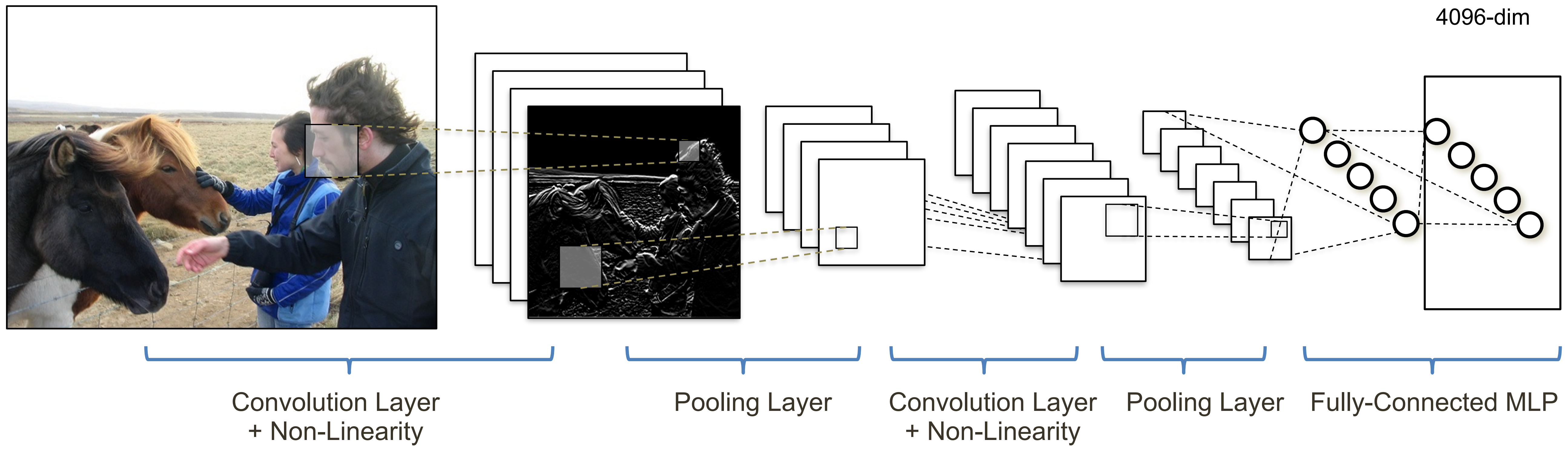
Image Embedding (VGGNet)



* slide from Dhruv Batra

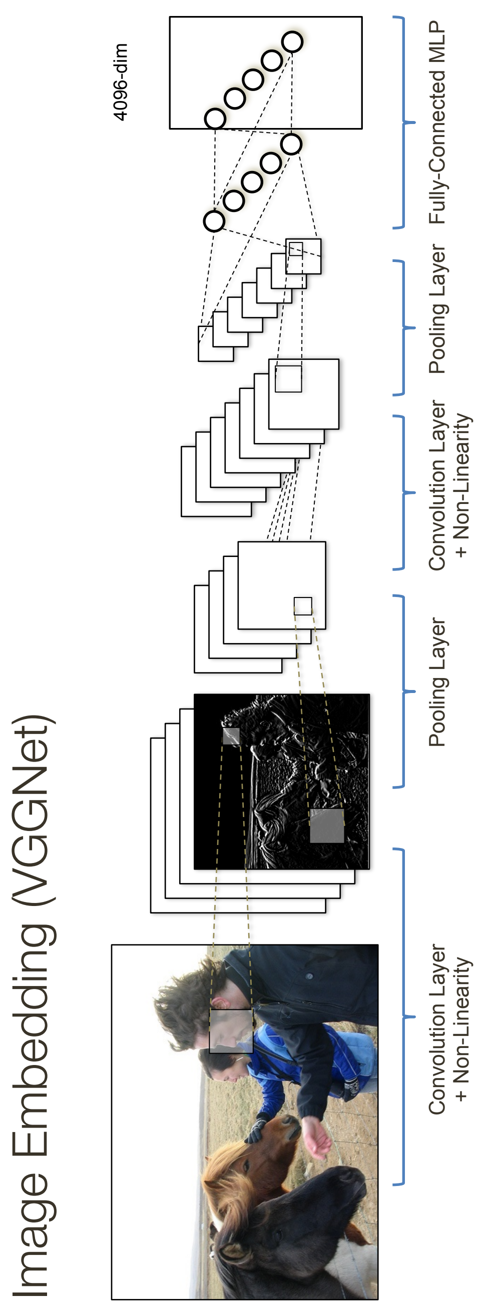
Applications: Neural Image Captioning

Image Embedding (VGGNet)



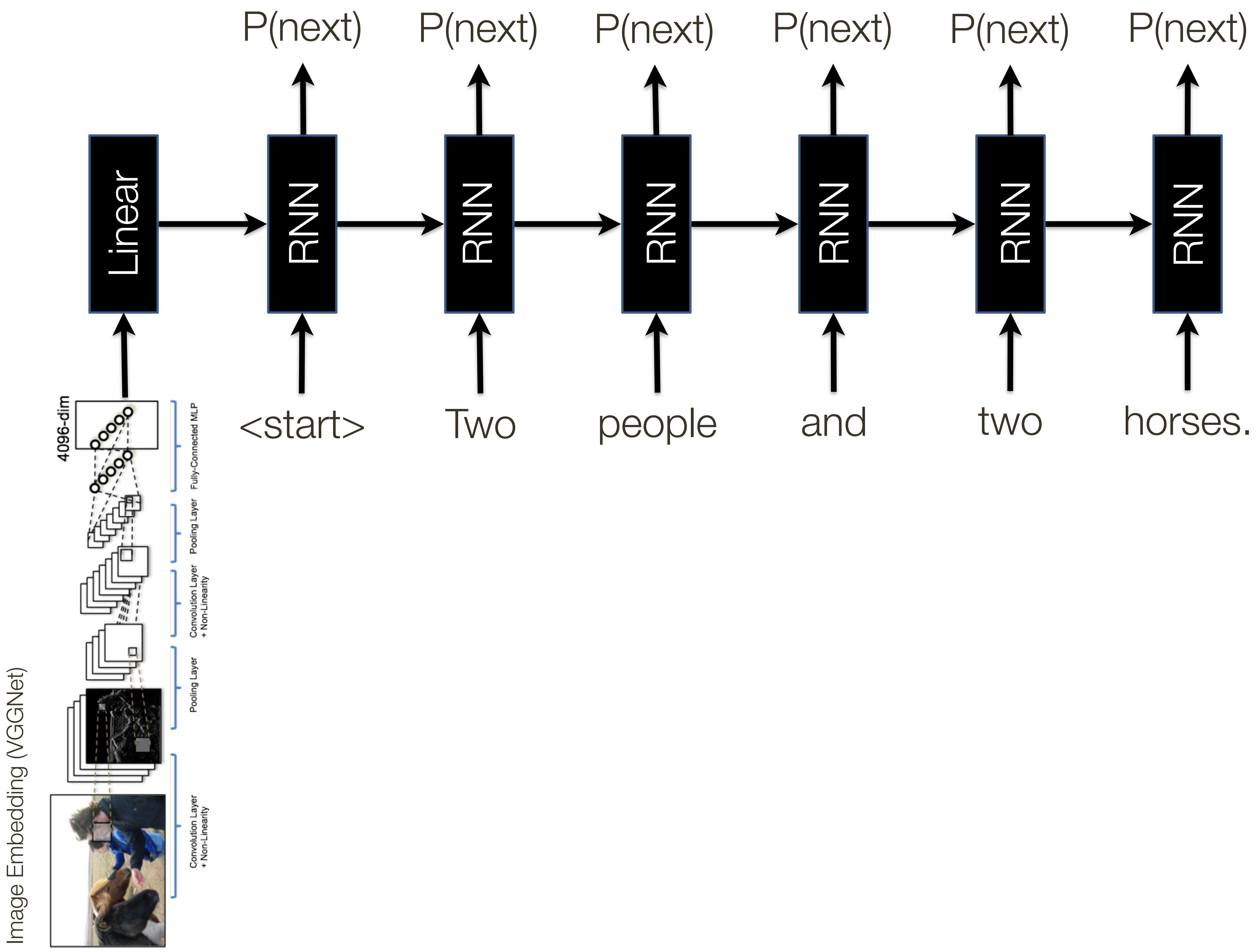
* slide from Dhruv Batra

Applications: Neural Image Captioning



* slide from Dhruv Batra

Applications: Neural Image Captioning



* slide from Dhruv Batra

Applications: Neural Image Captioning

Good results

A cat sitting on a suitcase on the floor

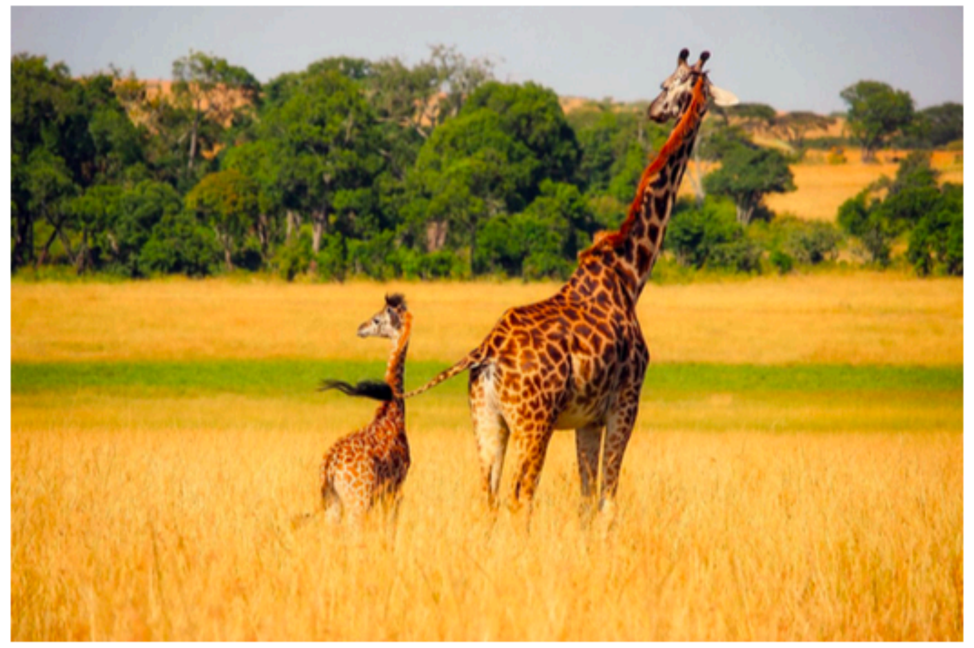
A cat is sitting on a tree branch

A dog is running in the grass with a frisbee

A white teddy bear sitting in the grass

Two people walking on the beach with surfboards

A tennis player in action on the court



Two giraffes standing in a grassy field

A man riding a dirt bike on a dirt track

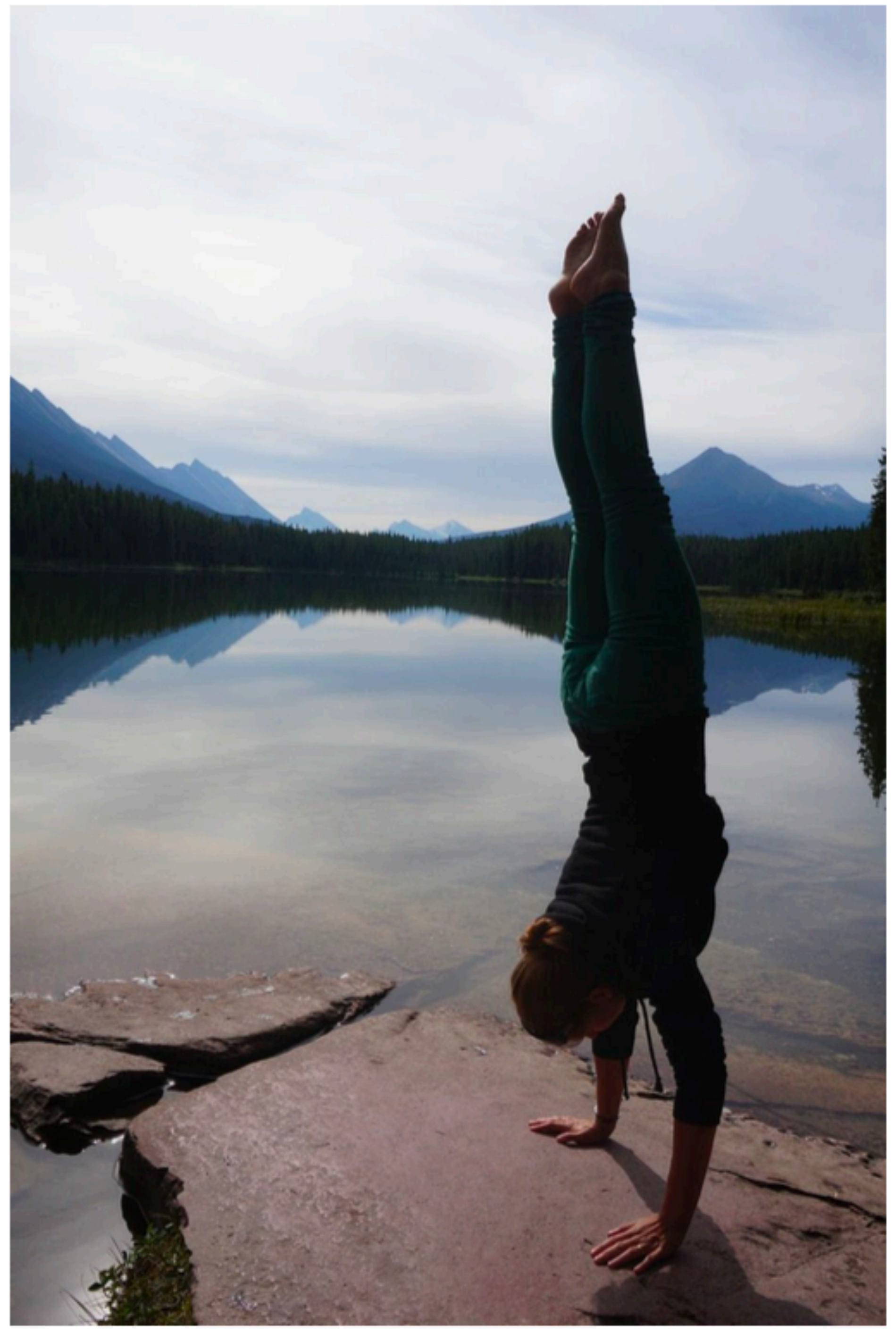
* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Applications: Neural Image Captioning

Failure cases

A woman is holding a cat in her hand

A person holding a computer mouse on a desk



A woman standing on a beach holding a surfboard

A bird is perched on a tree branch

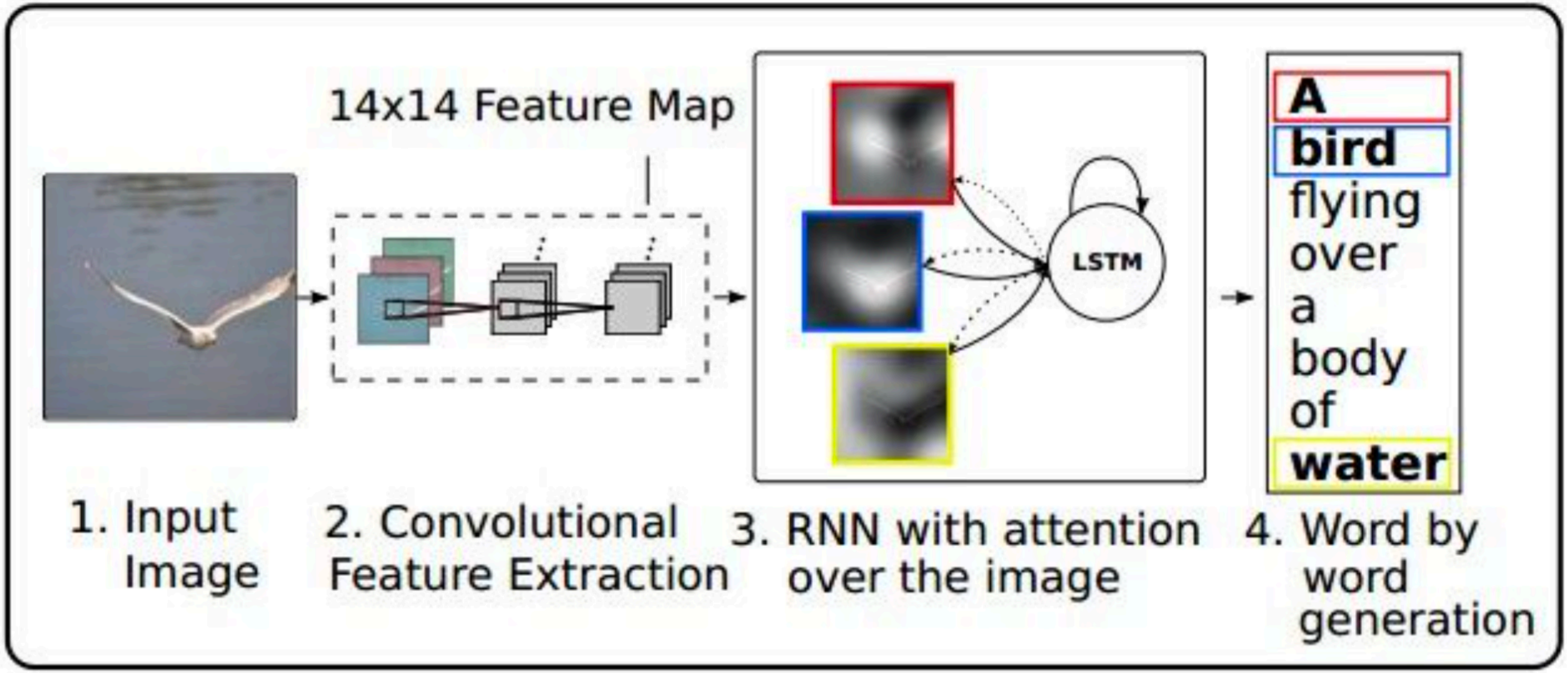
A man in a baseball uniform throwing a ball

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

Applications: Image Captioning with Attention

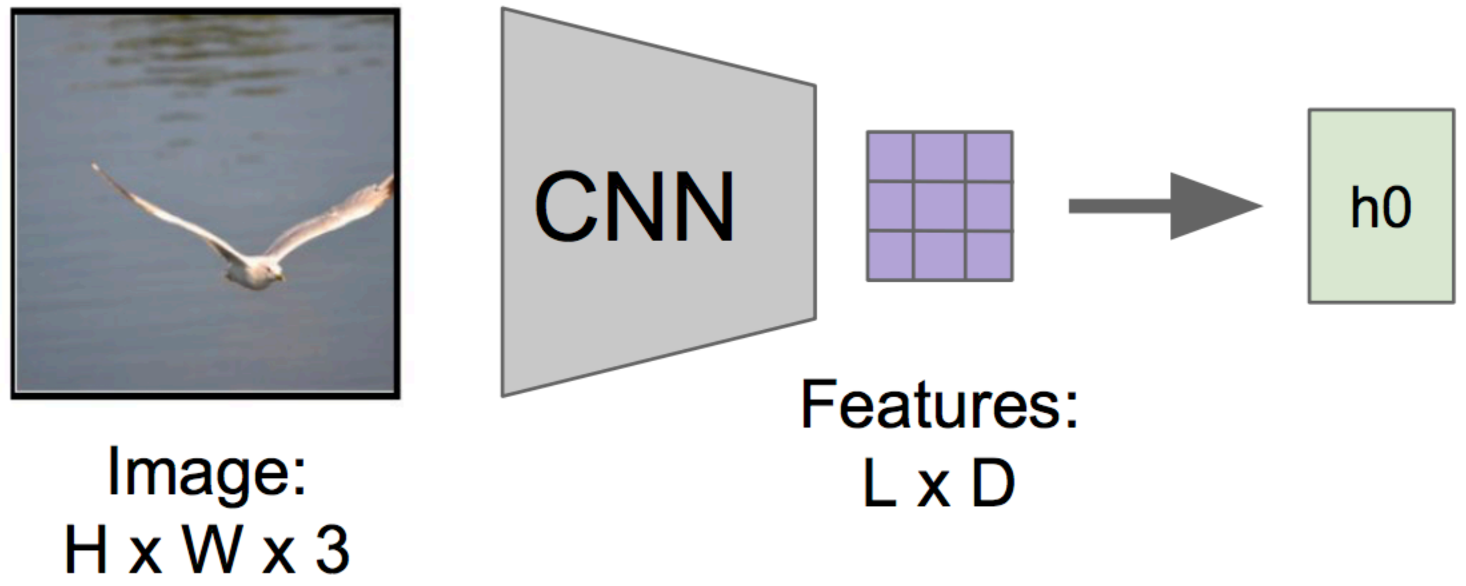
[Xu et al., ICML 2015]

RNN focuses its attention at a different spatial location when generating each word



Applications: Image Captioning with Attention

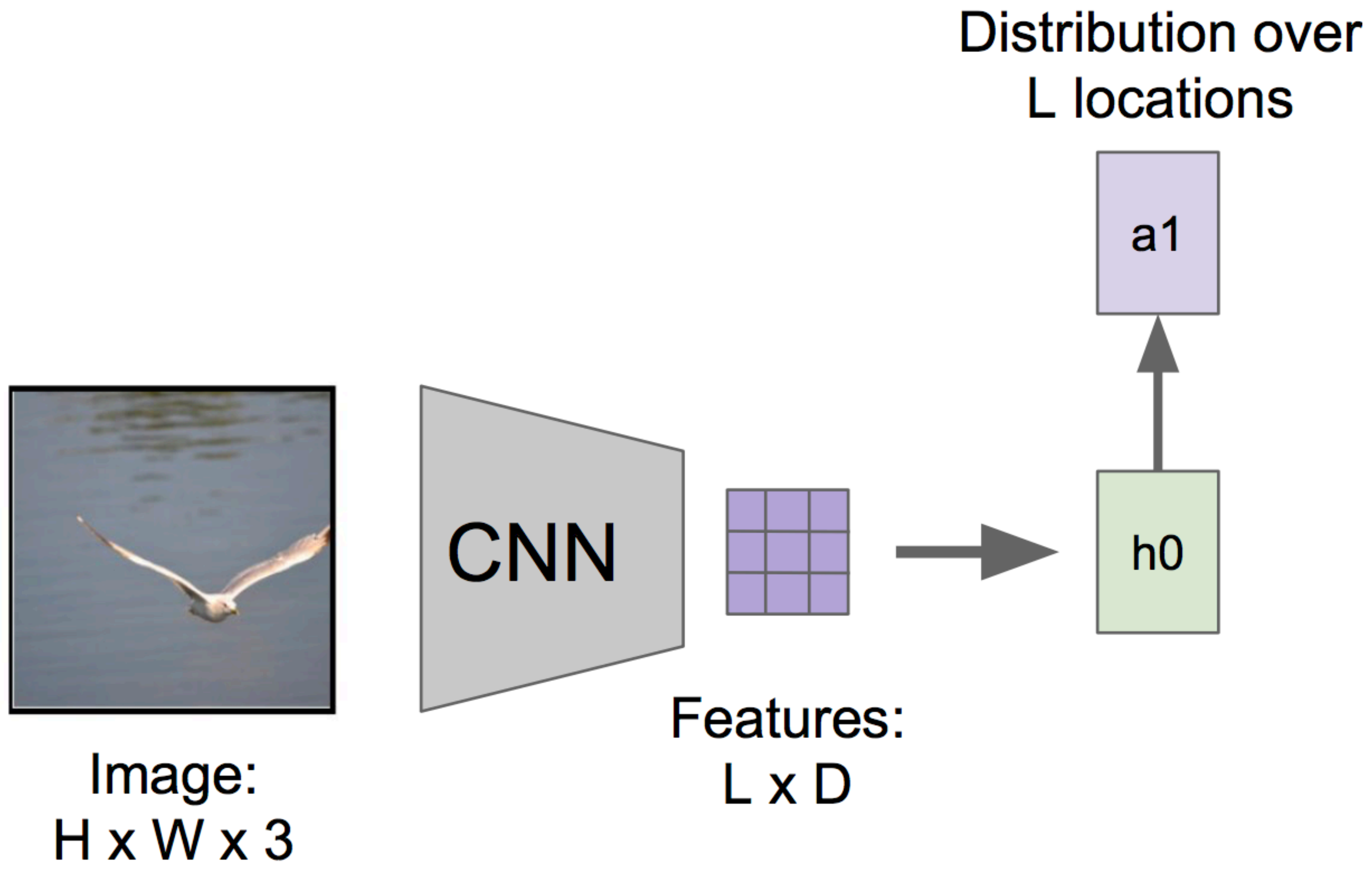
[Xu et al., ICML 2015]



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

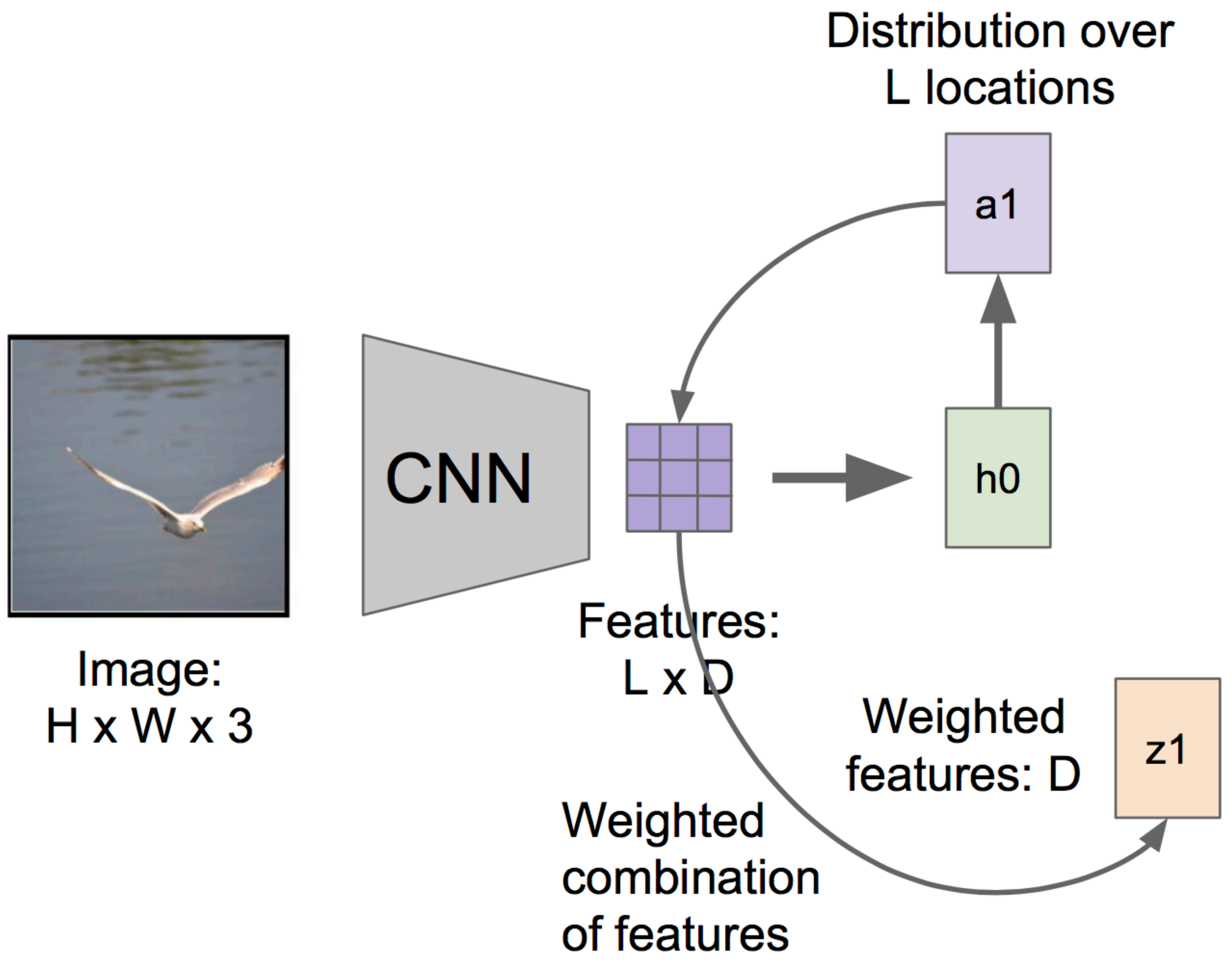
Applications: Image Captioning with Attention

[Xu et al., ICML 2015]



Applications: Image Captioning with Attention

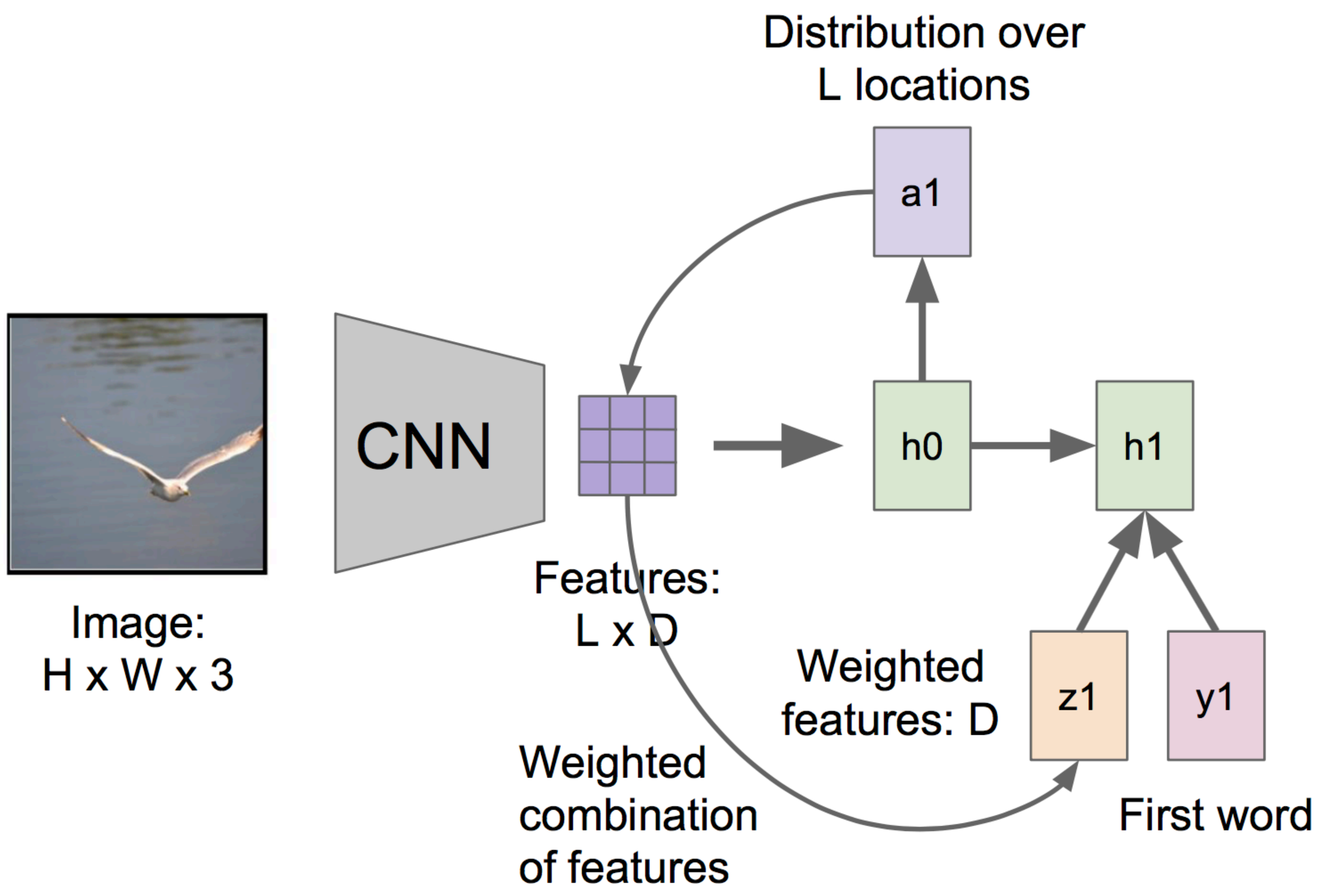
[Xu et al., ICML 2015]



$$z = \sum_{i=1}^{L} p_i v_i$$

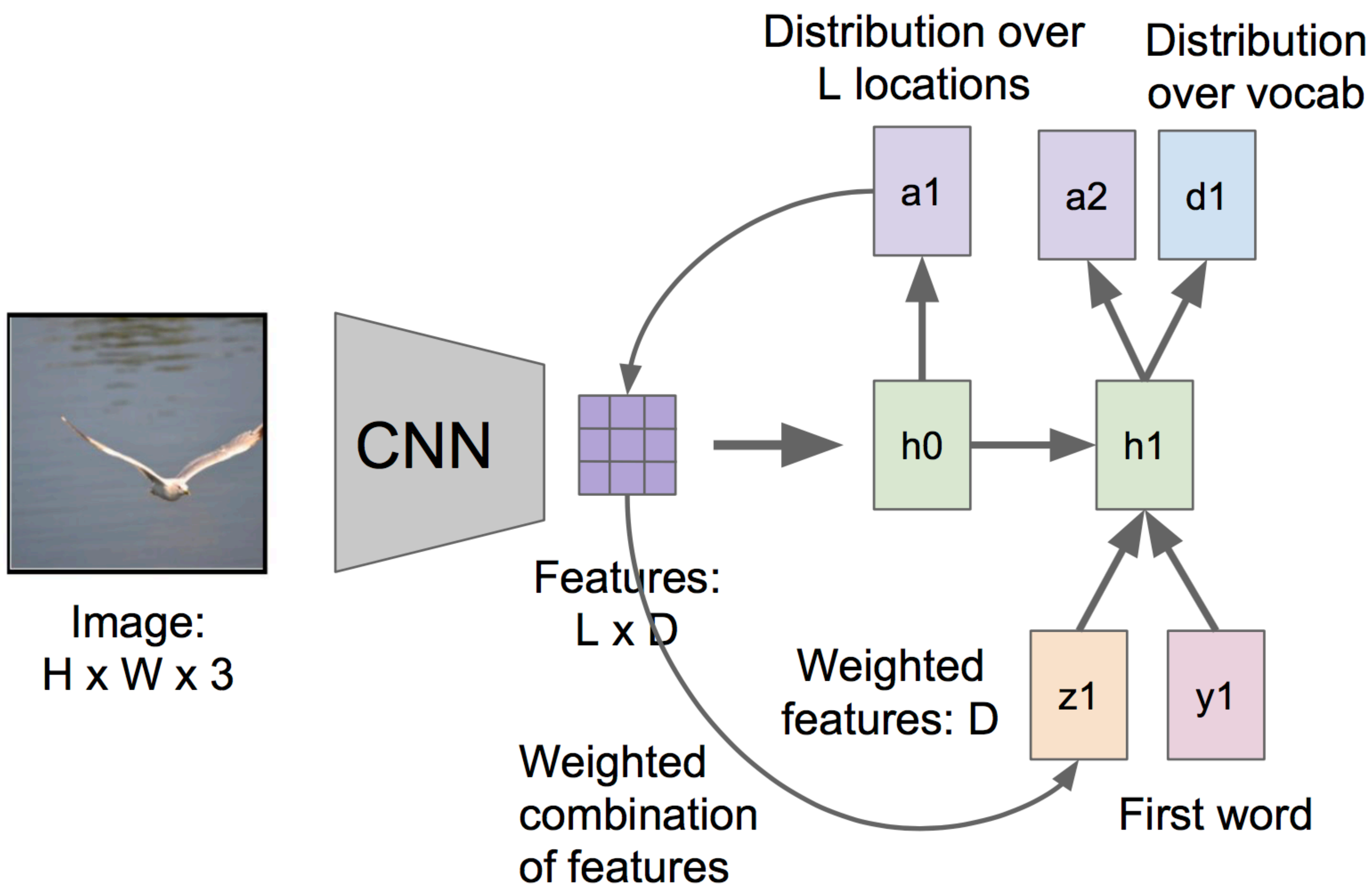
Applications: Image Captioning with Attention

[Xu et al., ICML 2015]



Applications: Image Captioning with Attention

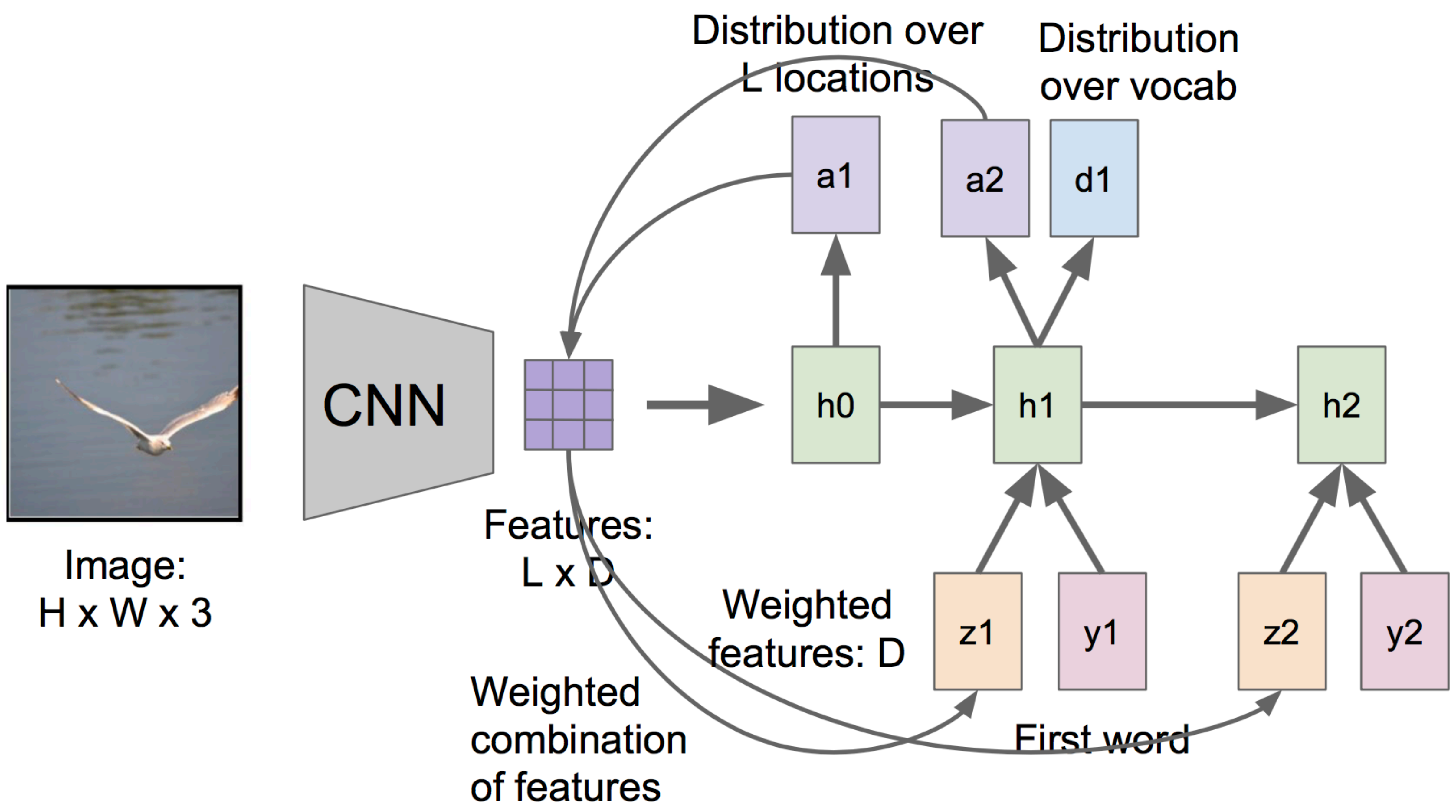
[Xu et al., ICML 2015]



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

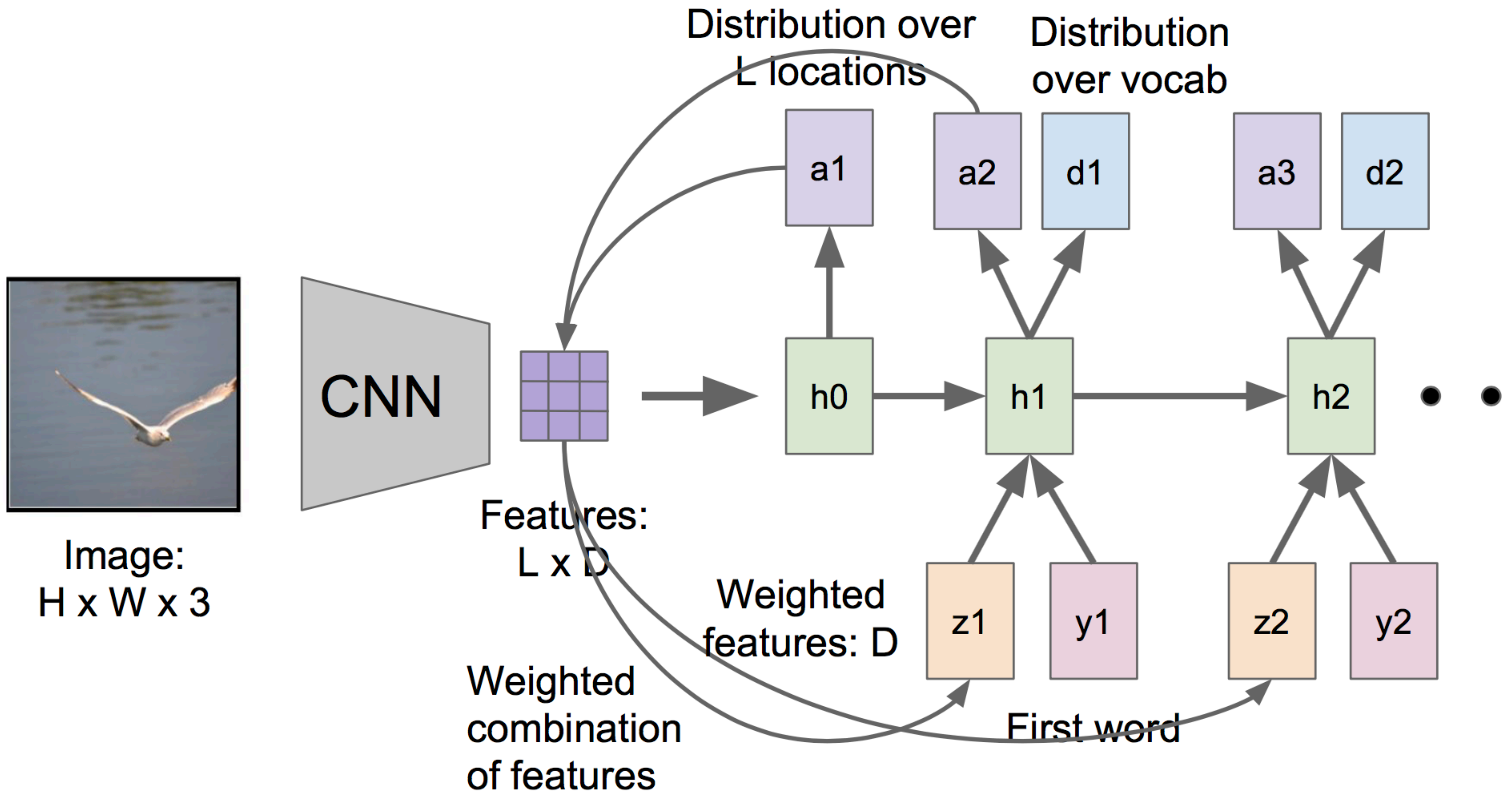
Applications: Image Captioning with Attention

[Xu et al., ICML 2015]



Applications: Image Captioning with Attention

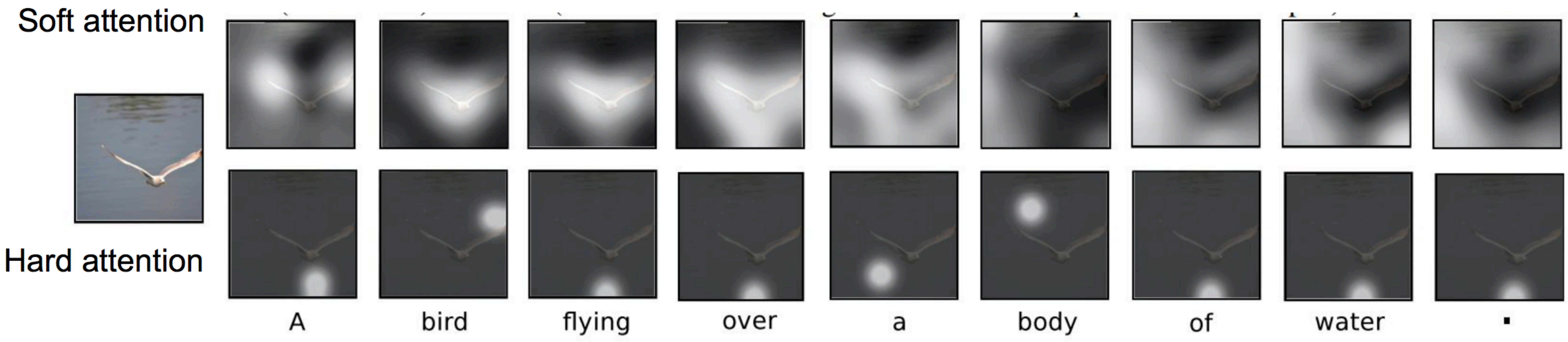
[Xu et al., ICML 2015]



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

Applications: Image Captioning with Attention

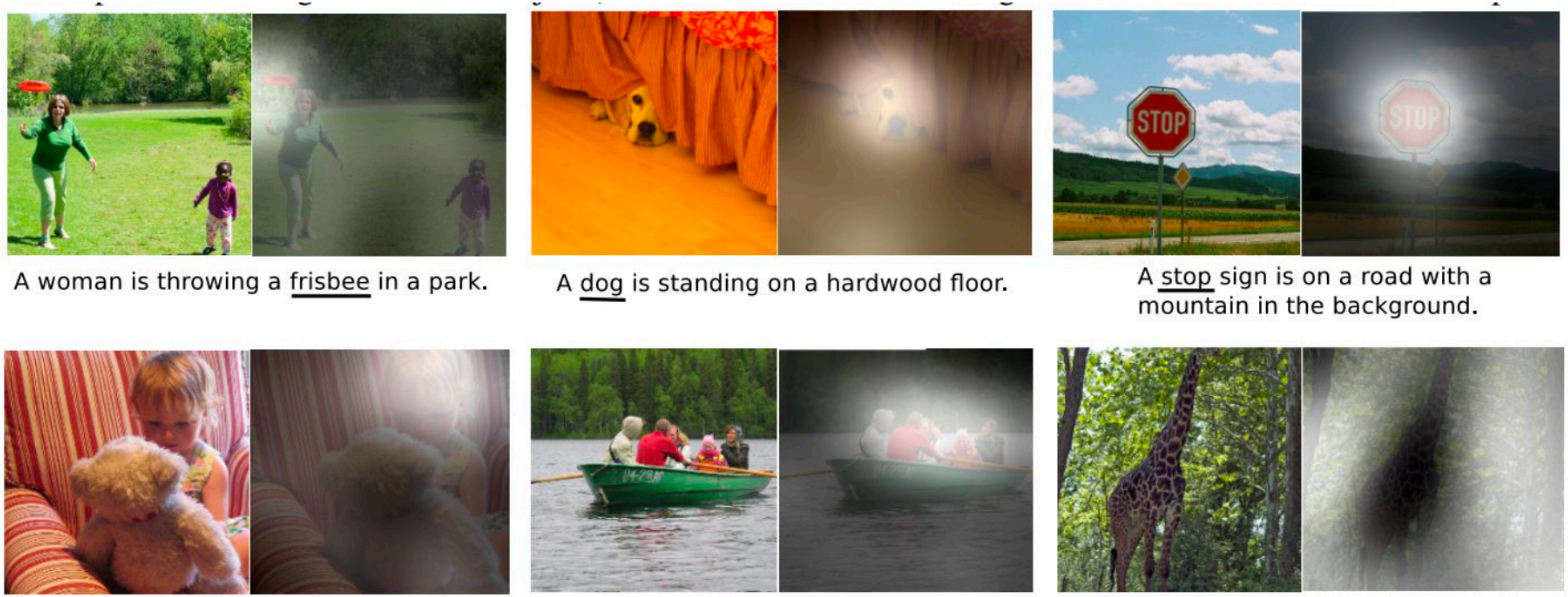
[Xu et al., ICML 2015]



Applications: Image Captioning with Attention

[Xu et al., ICML 2015]

Good results



A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.

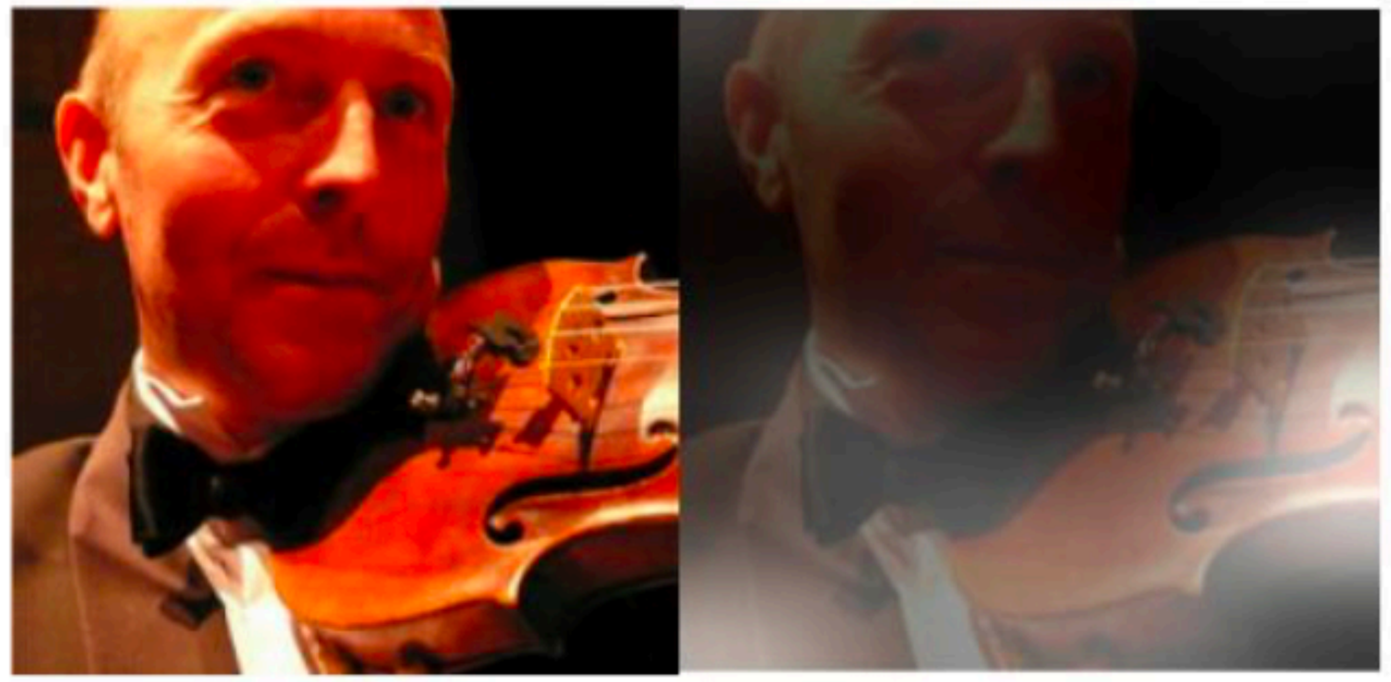
Applications: Image Captioning with Attention

[Xu et al., ICML 2015]

Failure results

A large white bird standing in a forest.

A woman holding a clock in her hand.



A man wearing a hat and a hat on a skateboard.

A person is standing on a beach with a surfboard.

A woman is sitting at a table with a large pizza.

A man is talking on his cell phone while another man watches.

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