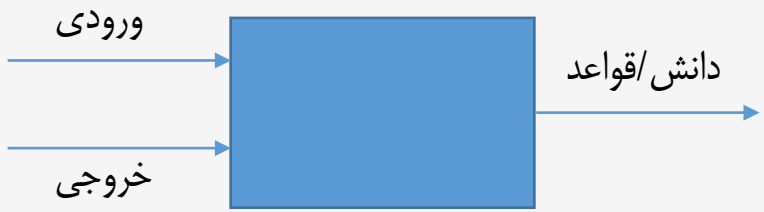
Deep learning in NLP

Part I: W2V, Glove, FastText & ELMO

Zeinab Rahimi, Feb 2022

پردازش مبتنی بر داده در هوش مصنوعی



- یادگیری ماشین
  - توصیف داده های ورودی با ویژگی های قابل فهم برای ماشین
  - الگوریتم یادگیری
- یادگیری بازنمایی (Representation Learning)
  - استخراج خودکار ویژگی ها یا بازنمایی مناسب برای داده ها
- یادگیری عمیق (Deep Learning)
  - استفاده از چندین سطح بازنمایی برای رسیدن به خروجی
- یادگیری عمیق در پردازش زبان طبیعی
  - **اولین گام تبدیل رشته ورودی به بردار**

مثال ساده از فرآیند یادگیری

Data

n	x	y
0	1	0
1	5	16
2	6	20

Model

$$y_n = wx_n + b$$

Cost

$$C(w,b) = \sum_{n\in\{0,1,2\}} (y_n - \hat{y}_n)^2$$

Model Candidate 1

$$y = 1x + 0$$

n	x	ŷ	y	$(y-\hat{y})^2$
0	1	0	1	1
1	5	16	5	121
2	6	20	6	196
			C(1,0)	318

Model Candidate 2

$$y = 2x + 2$$

n	x	ŷ	y	$(y-\hat{y})^2$
0	1	0	4	16
1	5	16	12	16
2	6	20	14	36
			C(2,2)	68

Wang Ling

جاسازی کلمه (Word Embedding)

- نمایش کلمه با یک بردار عددی که نشان دهنده ویژگی های کلمه است
- شیوه ای استاندارد در بازنمایی معنا در پردازش زبان
- قابل استفاده در یافتن شباهت کلمات
  - با معیارهای شباهت برداری مانند شباهت کسینوسی
- مورد استفاده در شبکه های عمیق برای وظایف پردازش زبان

انواع نمایش برداری

- مبتنی بر شمارش تعداد کلمات
  - One-hot
  - Tf-idf
  - ماتریس هم وقوعی
  - ماتریس کلمه–سند
- پیش بینی کننده
  - Word2vec

روش های جاسازی کلمه مبتنی بر شمارش

Document1 = ‘He is a lazy boy. She is also lazy.’
Document2 = ‘Neeraj is a lazy person.’

Dictionary of unique word in the corpus: (also & is: stop words)
[‘He’,’She’,’lazy’,’boy’,’Neeraj’,’person’]

One-Hot:
He:[1,0,0,0,0,0] She:[0,1,0,0,0,0]

Doc-Word matrix

	He	She	lazy	boy	Neeraj	person
D1	1	1	2	1	0	0
D2	0	0	1	0	1	1

TF-IDF

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

Document 1

Term	Count
This	1
is	1
about	2
Messi	4

Document 2

Term	Count
This	1
is	2
about	1
Tf-idf	1

$IDF(This) = \log(2/2) = 0.$

$IDF(Messi) = \log(2/1) = 0.301$

$TF\text{-}IDF(This, Document1) = (1/8) * (0) = 0$

$TF\text{-}IDF(This, Document2) = (1/5) * (0) = 0$

$TF\text{-}IDF(Messi, Document1) = (4/8)*0.301 = 0.15$

Co occurrence matrix

Corpus= Penny wise and pound foolish. A penny saved is a penny earned.

W=2	a	and	earned	foolish	is	penny	pound	saved	wise
a	0	0	0	0	0	2	0	0	0
penny	0	0	1	0	0	0	0	1	1

حرکت به سمت جاسازی های فشرده

- ایرادات
  - بردارهای خیلی بزرگ برای اندازه بزرگ واژگان
  - نیاز به تغییر سایز بردار با ورود کلمه جدید
  - بردارهای تنک
  - هیچ اشتراک اطلاعاتی و وجه مشترکی بین کلمات مشابه وجود ندارد.
- نیاز به
  - بردارهای کوچکتر و متراکم تر
  - مقاوم نسبت به تغییر اندازه واژگان
- راهکار
  - کاهش ابعاد بردار با روشهایی مانند SVD
  - یادگیری مستقیم بردارهای کوچک متراکم

	۱	۲	۳	٤	٥	٦	۷	۸	۹
پیرمرد	۱	۰	۰	۰	۰	۰	۰	۰	۰
پیرزن	۰	۱	۰	۰	۰	۰	۰	۰	۰
پسر	۰	۰	۱	۰	۰	۰	۰	۰	۰
دختر	۰	۰	۰	۱	۰	۰	۰	۰	۰
شاهزاده	۰	۰	۰	۰	۱	۰	۰	۰	۰
شاهدخت	۰	۰	۰	۰	۰	۱	۰	۰	۰
ملکه	۰	۰	۰	۰	۰	۰	۱	۰	۰
پادشاه	۰	۰	۰	۰	۰	۰	۰	۱	۰
حکمران	۰	۰	۰	۰	۰	۰	۰	۰	۱

مثالی از یک طرح تعبیه سازی (embedding) با استفاده از کدگذاری one-hot برای یک لغت نامه ۹ کلمه ای. در این جدول هر سطر نمایانگر یک word embedding است.
همانطور که مشاهده میشود اکثر درایه های این جدول (و به طبع word embedding ها) صفر هستند!

Word2Vec, 2013

Distributed Representations of Words and Phrases and their Compositionality

Tomas Mikolov
Google Inc.
Mountain View
mikolov@google.com

Ilya Sutskever
Google Inc.
Mountain View
ilyasu@google.com

Kai Chen
Google Inc.
Mountain View
kai@google.com

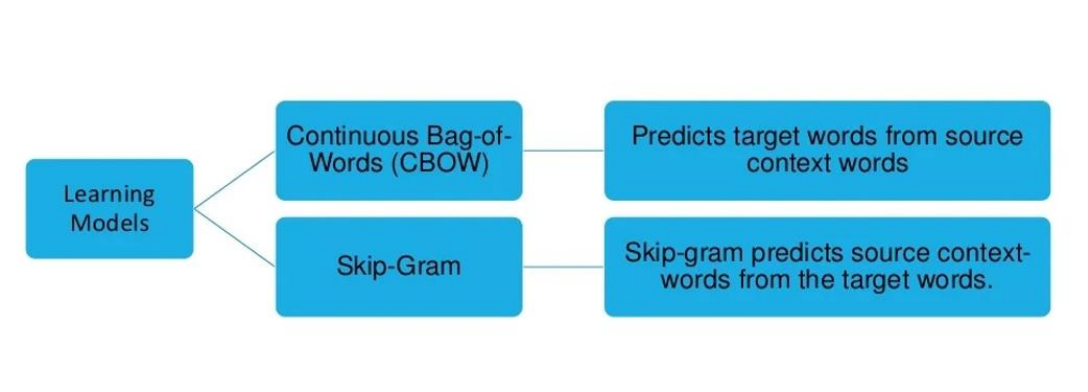
Greg Corrado
Google Inc.
Mountain View
gcorrado@google.com

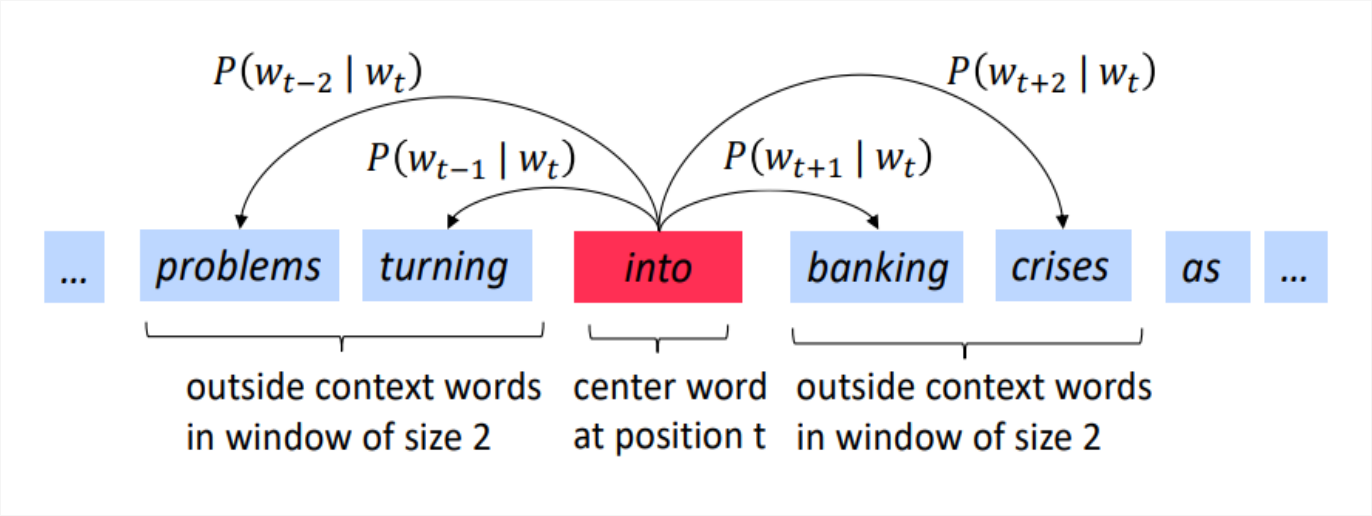
Jeffrey Dean
Google Inc.
Mountain View
jeff@google.com

Word2Vec

- ایده
- یادگیری مستقیم بردارهای کوچک متراکم
- یادگیری بردار کلمه با استفاده از بافتار آن
- مطرح شده در سال ۲۰۱۳ توسط گوگل
- پیاده سازی در قالب دو روش:
  - CBOW
    - حدس کلمه هدف با توجه به بافتار
  - Skip gram
    - حدس بافتار با داشتن کلمه هدف

Model: $P(Context|W_t)$
Loss:$1-P(W_{-t}|W_t)$



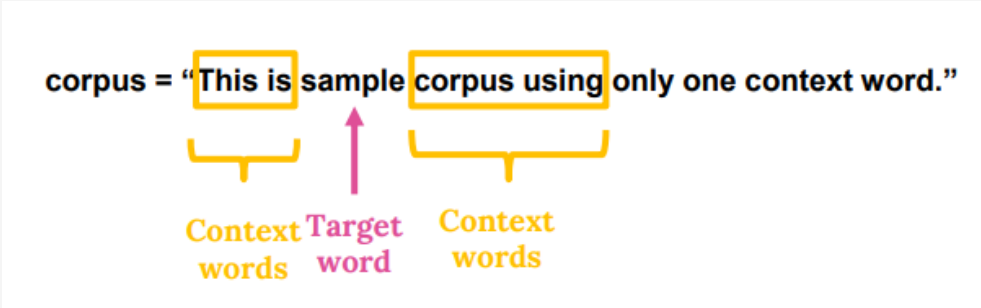


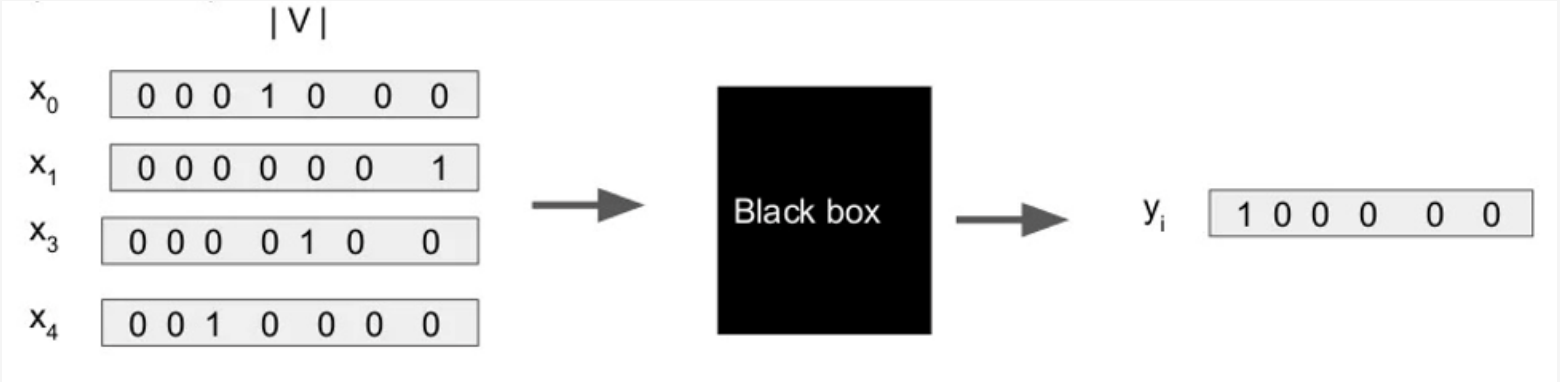
CBOW

Window size=2

$$P(wt \mid wt-1, wt-2, wt+1, wt+2)$$

- حدس کلمه هدف با توجه به بافتار
- هر بار به شبکه کلمات بافتار داده می شود و شبکه باید حدس بزند کلمه هدف چیست.





CBOW

	input				output
1	is	sample			this
2	this	sample	corpus		is
3	this	is	corpus	using	sample
4	is	sample	using	only	corpus
5	sample	corpus	only	one	using
6	corpus	using	one	context	only
7	using	only	context	word	one
8	only	one	word		context
9	one	context			word

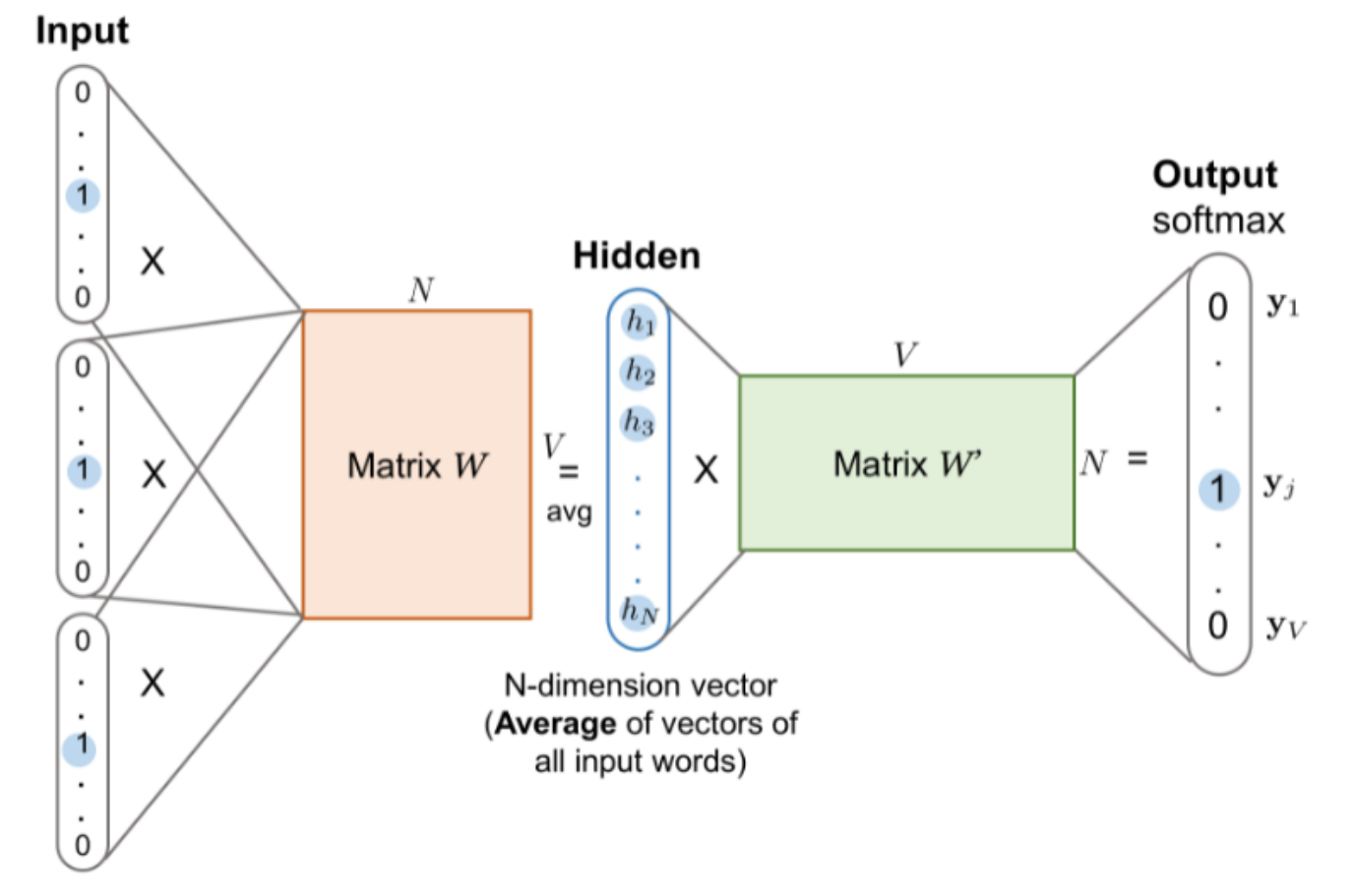
vocabulary	this	is	sample	corpus	using	only	one	context	word
this	1	0	0	0	0	0	0	0	0
is	0	1	0	0	0	0	0	0	0
sample	0	0	1	0	0	0	0	0	0
corpus	0	0	0	1	0	0	0	0	0
using	0	0	0	0	1	0	0	0	0
only	0	0	0	0	0	1	0	0	0
one	0	0	0	0	0	0	1	0	0
context	0	0	0	0	0	0	0	1	0
word	0	0	0	0	0	0	0	0	1

- گام ۱:
  - مشخص کردن داده ورودی
- گام ۲:
  - آموزش شبکه

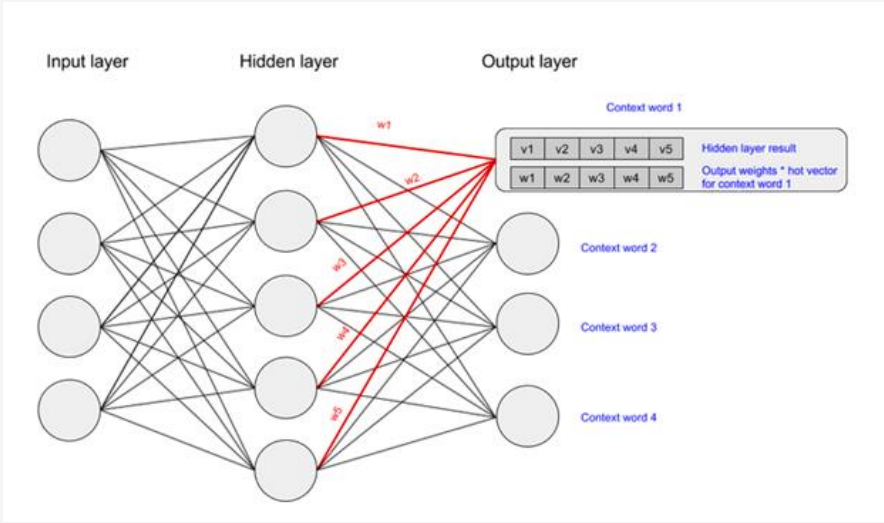
- هر سطر جدول: ورودی و خروجی شبکه در یک بار آموزش
- ورود کلمات به شبکه: بردارهای one-hot

بخشی از تصاویر مثال برداشته شده از Deep NLP workshop (IKT,2021)

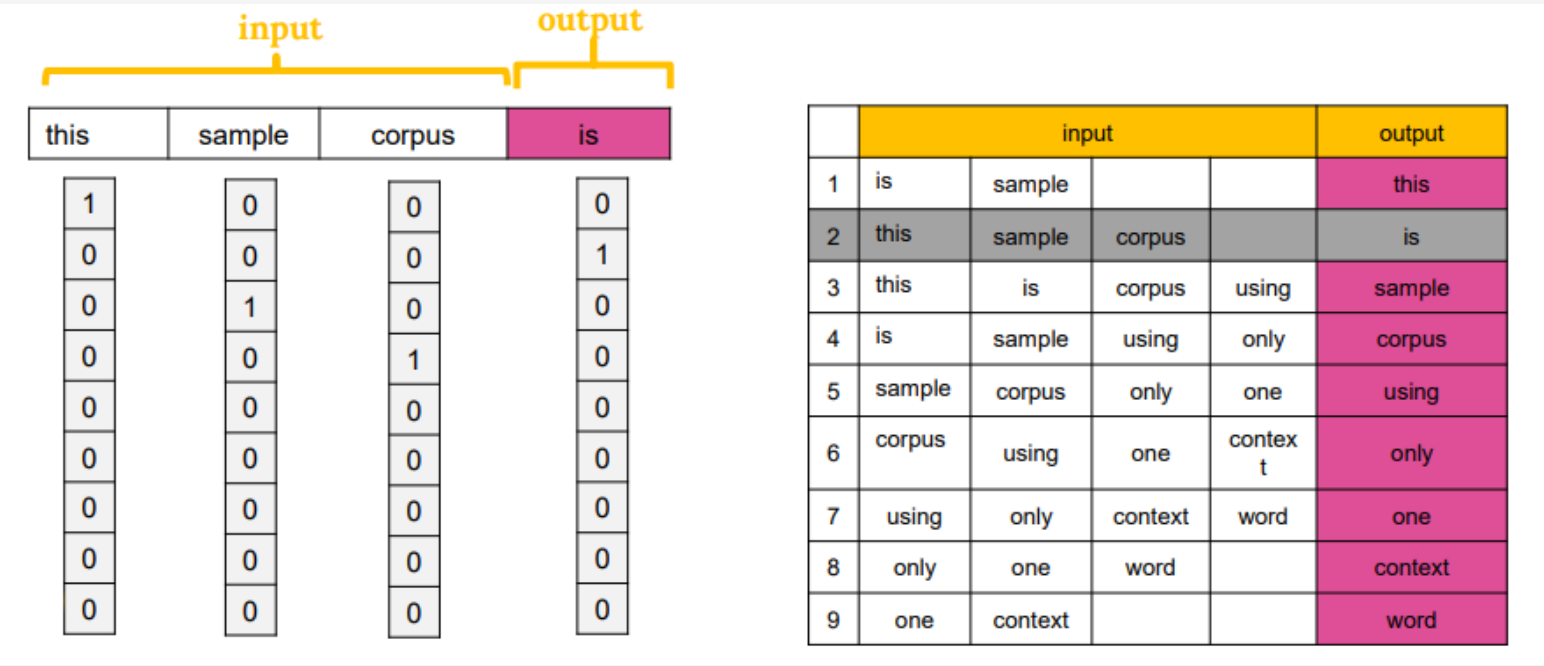
CBOW



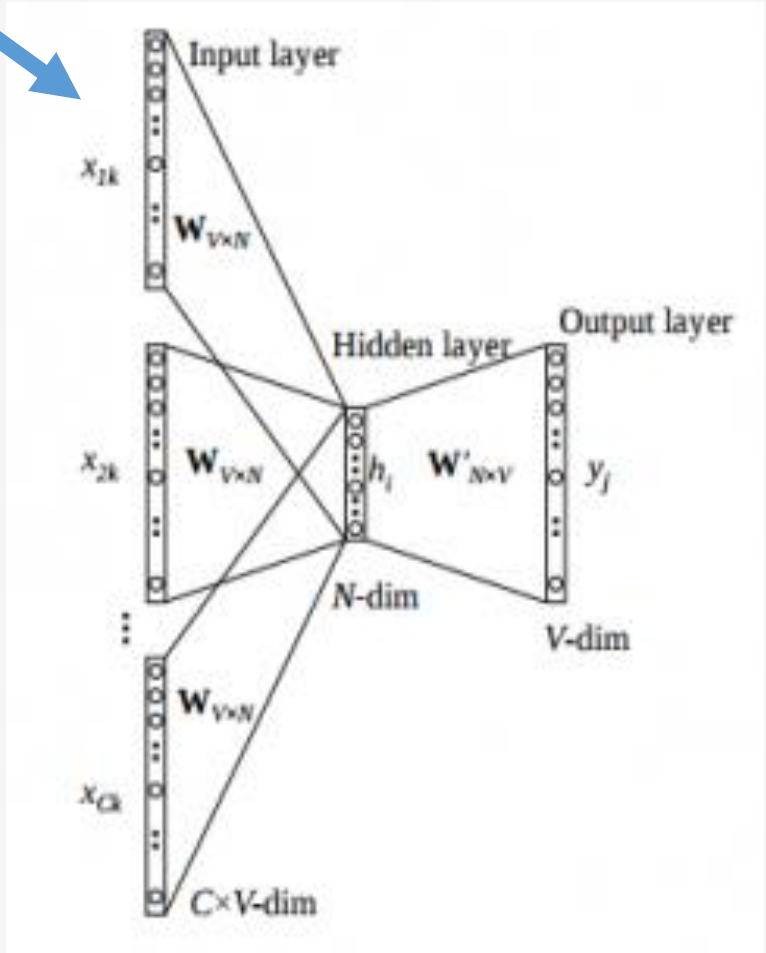
- استفاده از یک شبکه Fully Connected ساده
- ۳ لایه:
  - ورودی، هیدن و خروجی
  - ورودی و خروجی هر دو بردار one-hot
  - W و W′ باید یادگرفته شوند.



CBOW



	input				output
1	is	sample			this
2	this	sample	corpus		is
3	this	is	corpus	using	sample
4	is	sample	using	only	corpus
5	sample	corpus	only	one	using
6	corpus	using	one	context	only
7	using	only	context	word	one
8	only	one	word		context
9	one	context			word



بخشی از تصاویر مثال برداشته شده از Deep NLP workshop (IKT,2021)

CBOW

Input one-hot vectors(1×9)

this	1	0	0	0	0	0	0	0	0
sample	0	0	1	0	0	0	0	0	0
corpus	0	0	0	1	0	0	0	0	0

w(9× 4)

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16
17	18	19	20
21	22	23	24
25	26	27	28
29	30	31	32
33	34	35	36

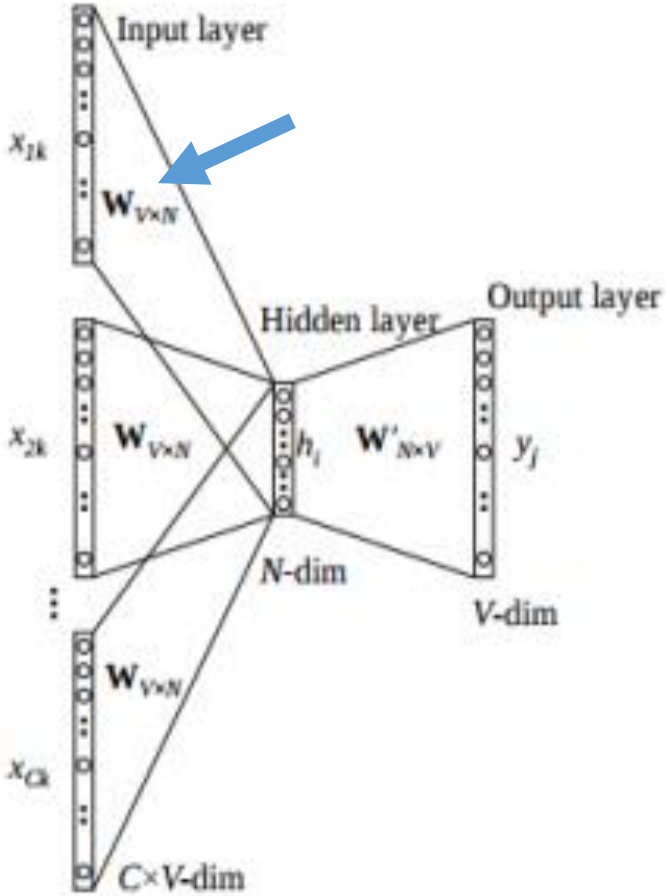
Input embed vectors

1	2	3	4
9	10	11	12
13	14	15	16

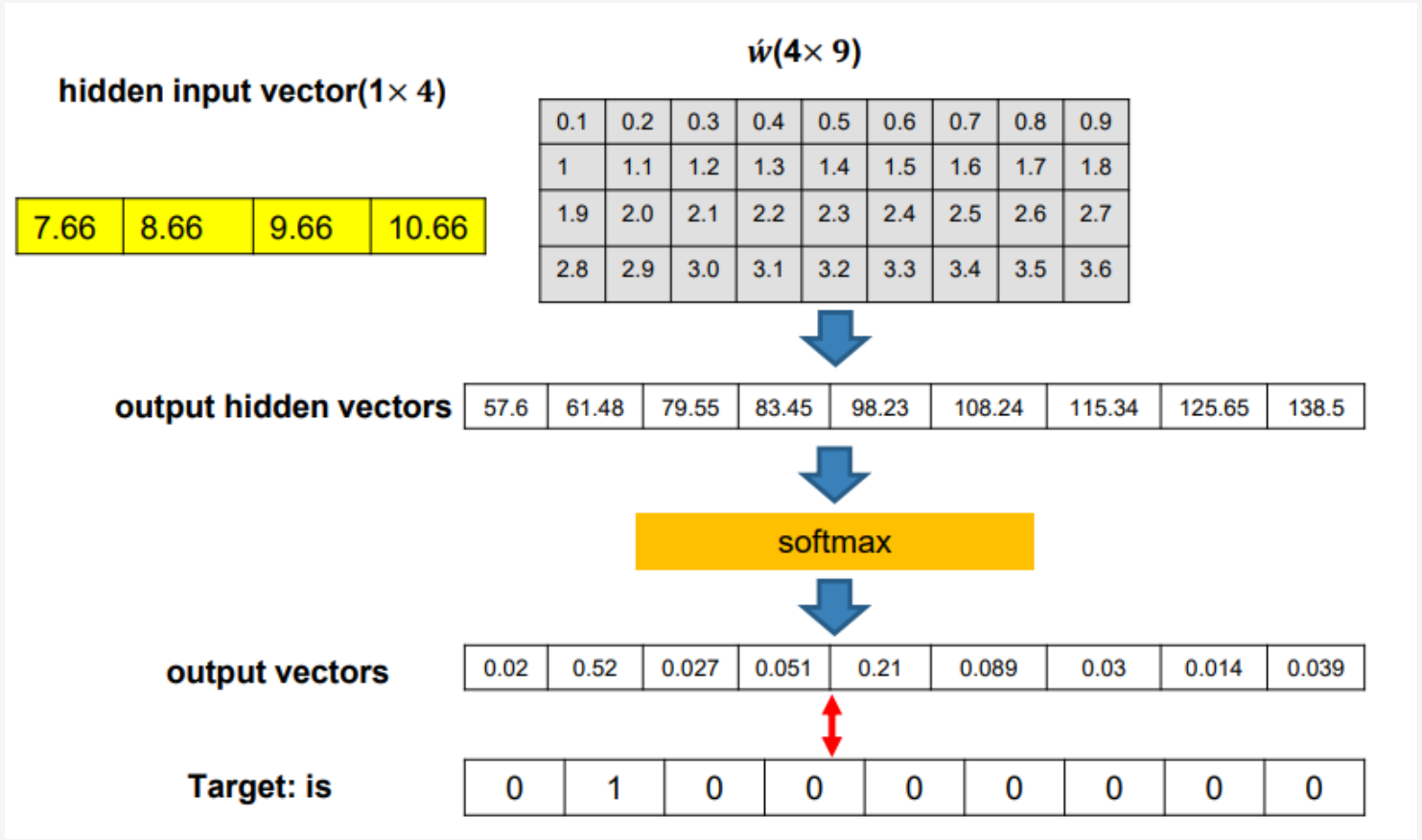
Average hidden Action

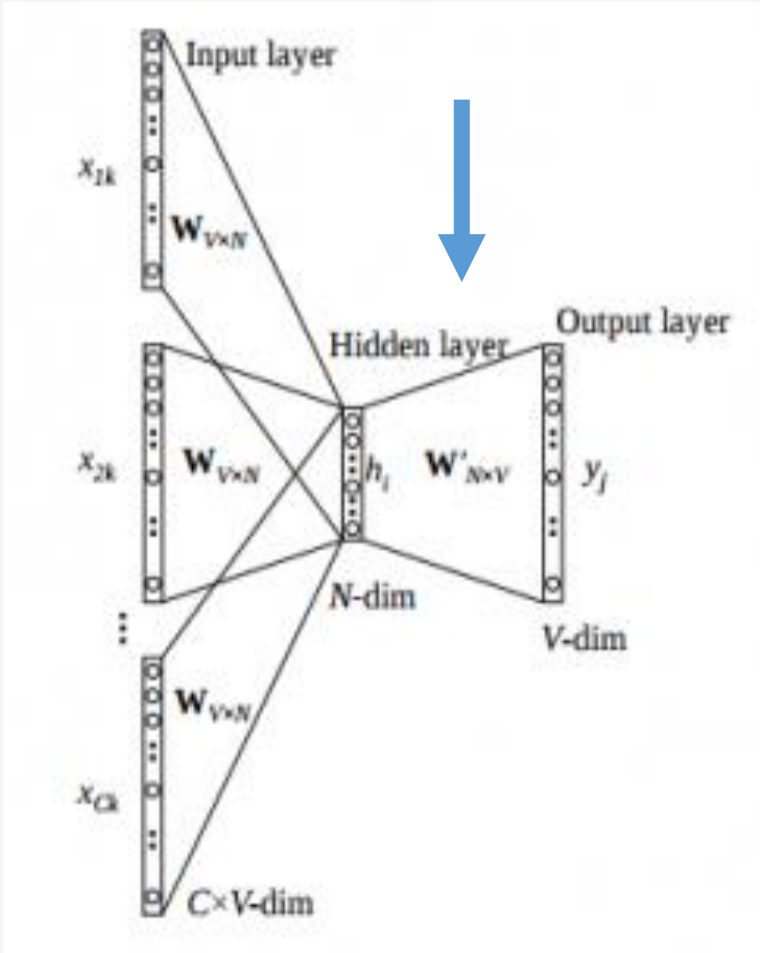
7.66	8.66	9.66	10.66

hidden input vector(1× 4)

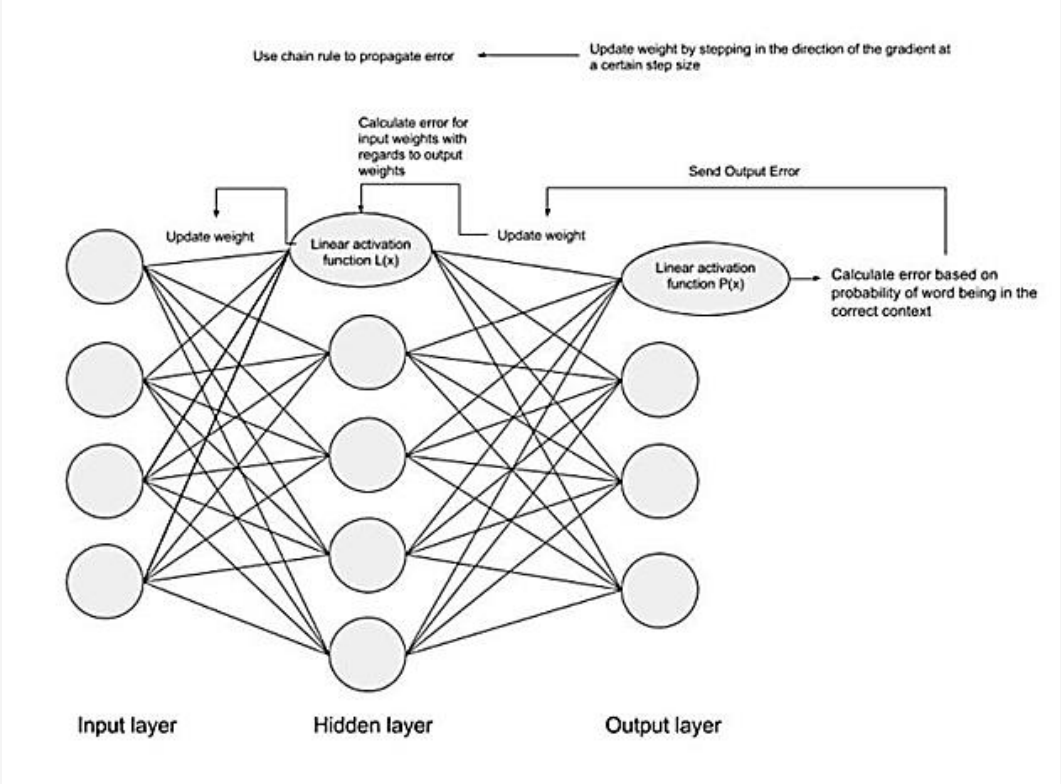


CBOW


hidden input vector(1× 4)
7.66 8.66 9.66 10.66
ẃ(4× 9)
0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9
1 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8
1.9 2.0 2.1 2.2 2.3 2.4 2.5 2.6 2.7
2.8 2.9 3.0 3.1 3.2 3.3 3.4 3.5 3.6
output hidden vectors
57.6 61.48 79.55 83.45 98.23 108.24 115.34 125.65 138.5
softmax
output vectors
0.02 0.52 0.027 0.051 0.21 0.089 0.03 0.014 0.039
Target: is
0 1 0 0 0 0 0 0 0


Input layer
Hidden layer
Output layer
x_{1k}
x_{2k}
x_{Ck}
$\mathbf{W}_{V\times N}$
$\mathbf{W}'_{N\times V}$
h_i
y_j
N-dim
V-dim
C×V-dim

CBOW



- در اینجا تابع خطا Cross entropy خطای بین احتمال پیش بینی و کلمه صحیح را اندازه میگیرد.
- Cross entropy بین دو توزیع p و q:

$$H(p,q) = -\sum_x p(x) \log q(x)$$

- در اینجا همیشه خروجی بردار one-hot است و همه به جز عنصر iام صفر هستند.
- خود عنصر iام مقدار یک دارد.

$$H(\hat{y},y) = -\sum_{j=1}^{|V|} y_j \log(\hat{y}_j)$$

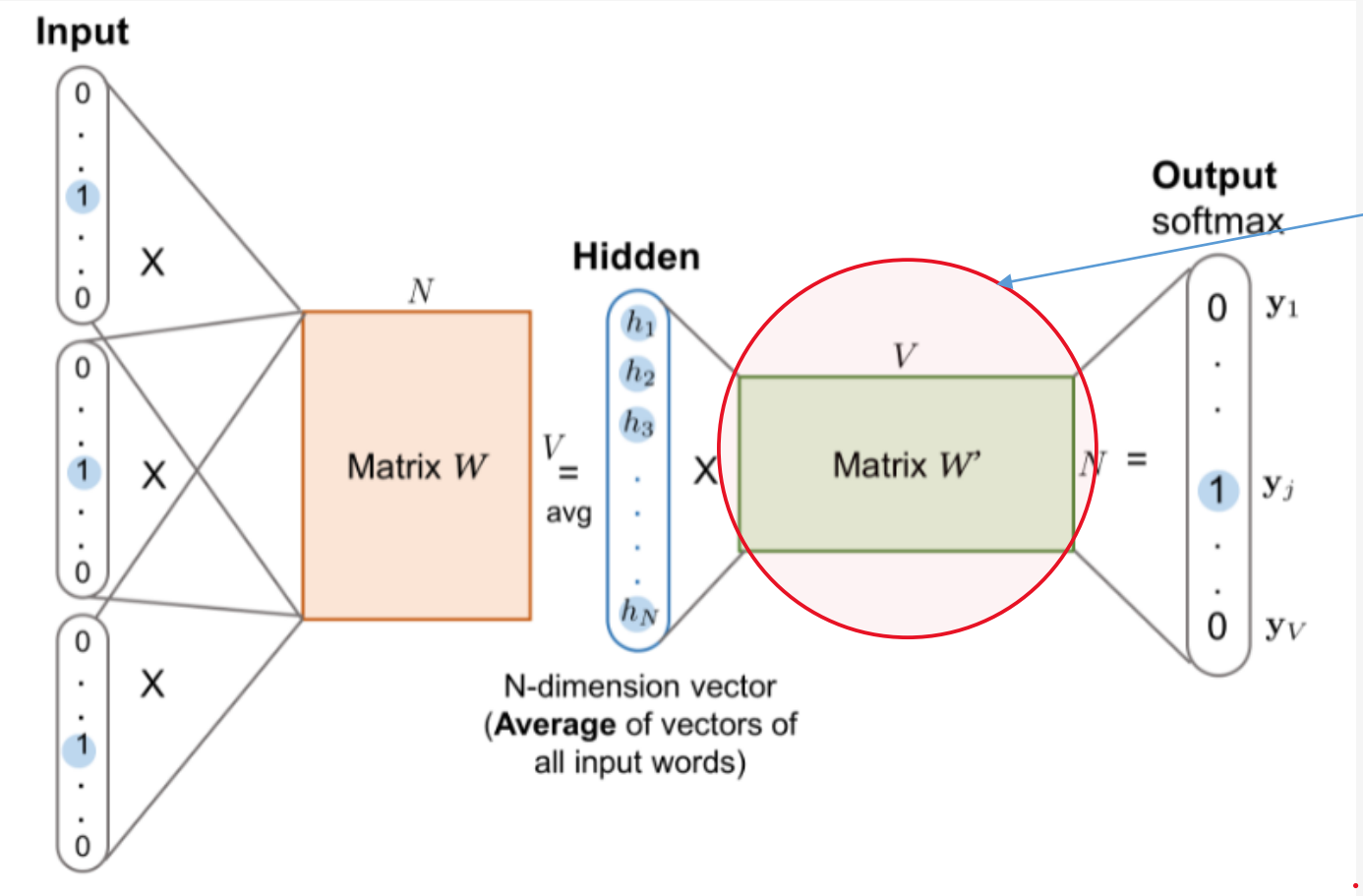
j=i 1
Other 0

$$H(\hat{y},y) = -y_i \log(\hat{y}_i)$$

y_i=1

$$H(\hat{y},y) = -\log(\hat{y}_i)$$

CBOW



بردارهای بازنمایی کلمات (خروجی مورد نظر) همان ماتریس W′ است.

هایپرپارامترهای تاثیرگذار:

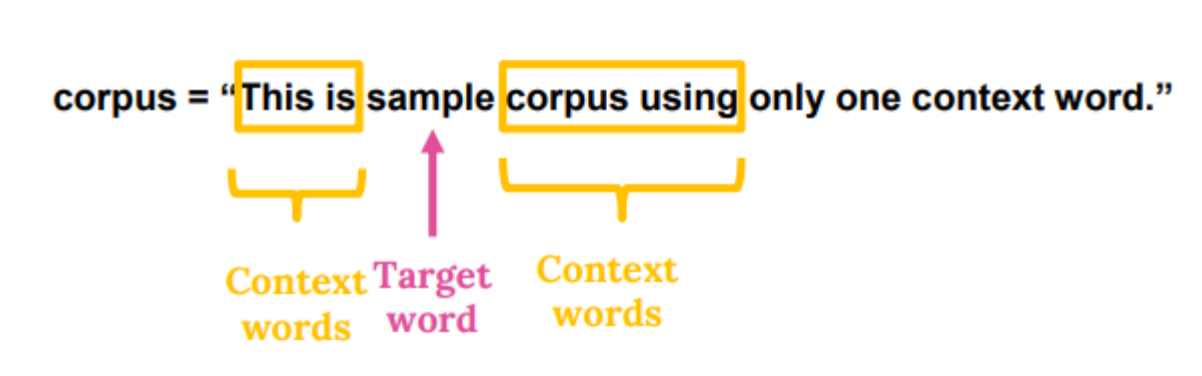
- تعداد بعد مورد نظر برای بازنمایی (۵۰ تا ۵۰۰)
- اندازه پنجره (۱ تا ۱۰)

Skip Gram

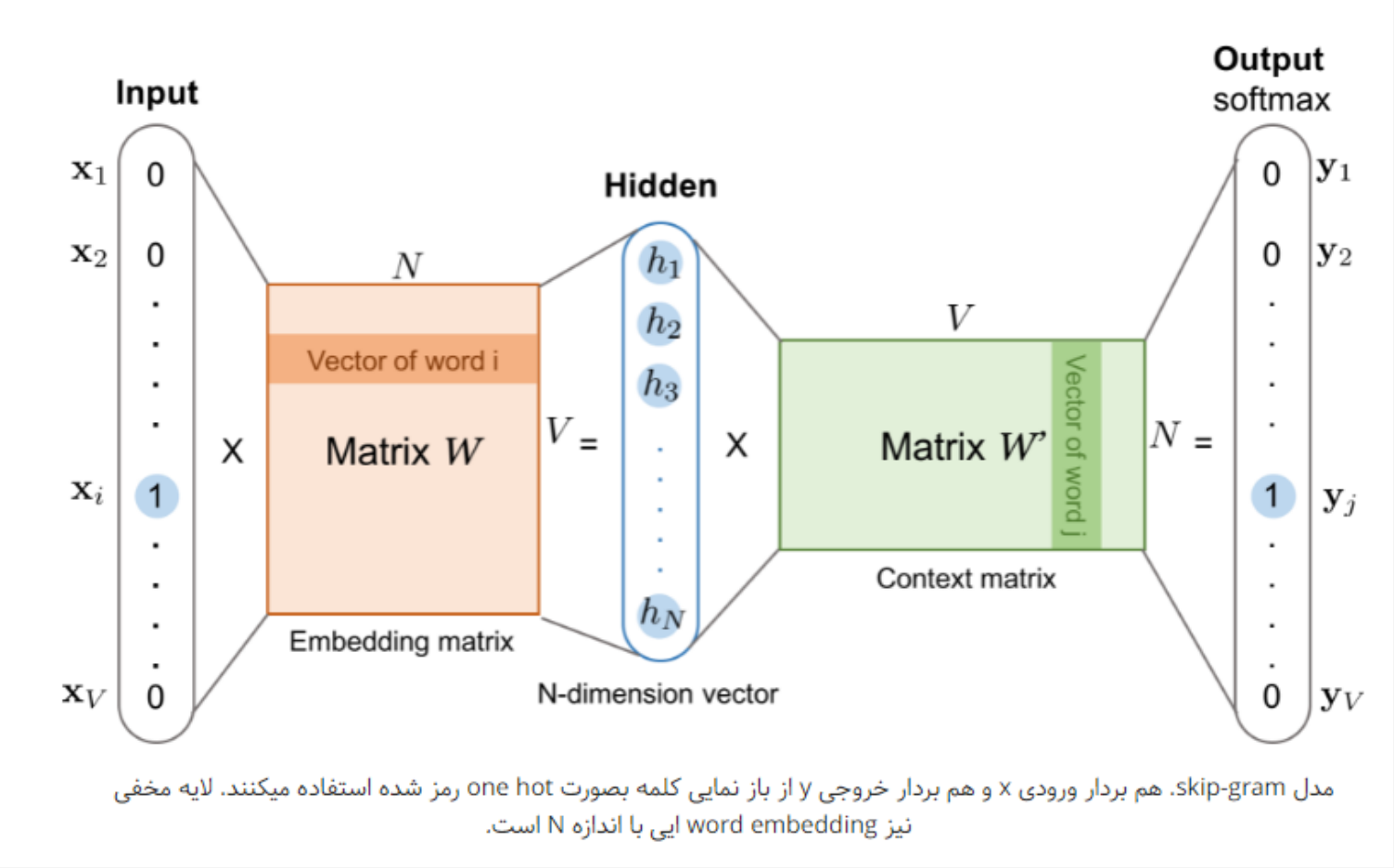
- حدس بافتار با توجه به کلمه هدف
- هر بار یک کلمه هدف به شبکه داده می شود و شبکه باید حدس بزند کلمات بافتار چیست.

Window size=2

$$P(wt-1, wt-2, wt+1, wt+2|wt)$$



Skip Gram



مدل skip-gram. هم بردار ورودی x و هم بردار خروجی y از باز نمایی کلمه بصورت one hot رمز شده استفاده میکنند. لایه مخفی نیز word embedding ایی با اندازه N است.

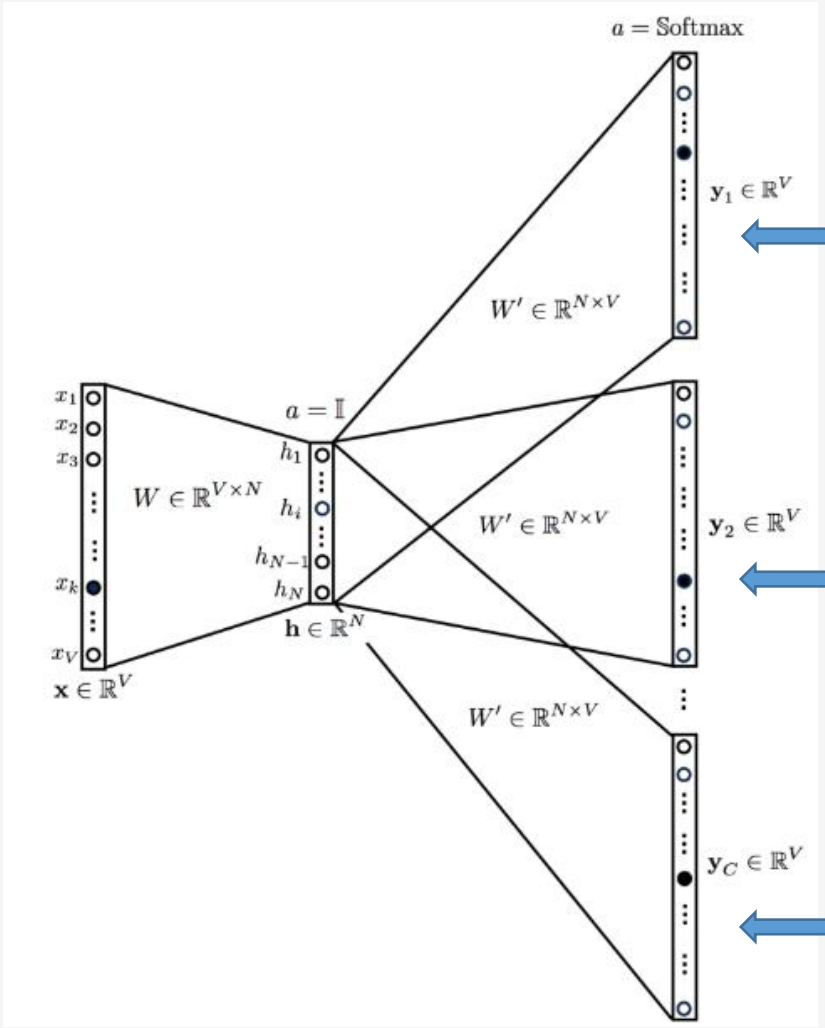
- گام اول:
  - تبدیل پیکره به مجموعه آموزشی
- گام دوم:
  - آموزش شبکه

	input	output
1	this	Is
2	this	sample
3	is	this
4	is	sample
5	is	corpus

	input	output
6	sample	Is
7	sample	using
8	sample	this
9	sample	corpus

بخشی از تصاویر مثال برداشته شده از Deep NLP workshop (IKT,2021)

Skip Gram



This

Sample

Corpus

	input	output
1	this	Is
2	this	sample
3	is	this
4	is	sample
5	is	corpus

Input one-hot vectors(1×9)

is

0	1	0	0	0	0	0	0	0

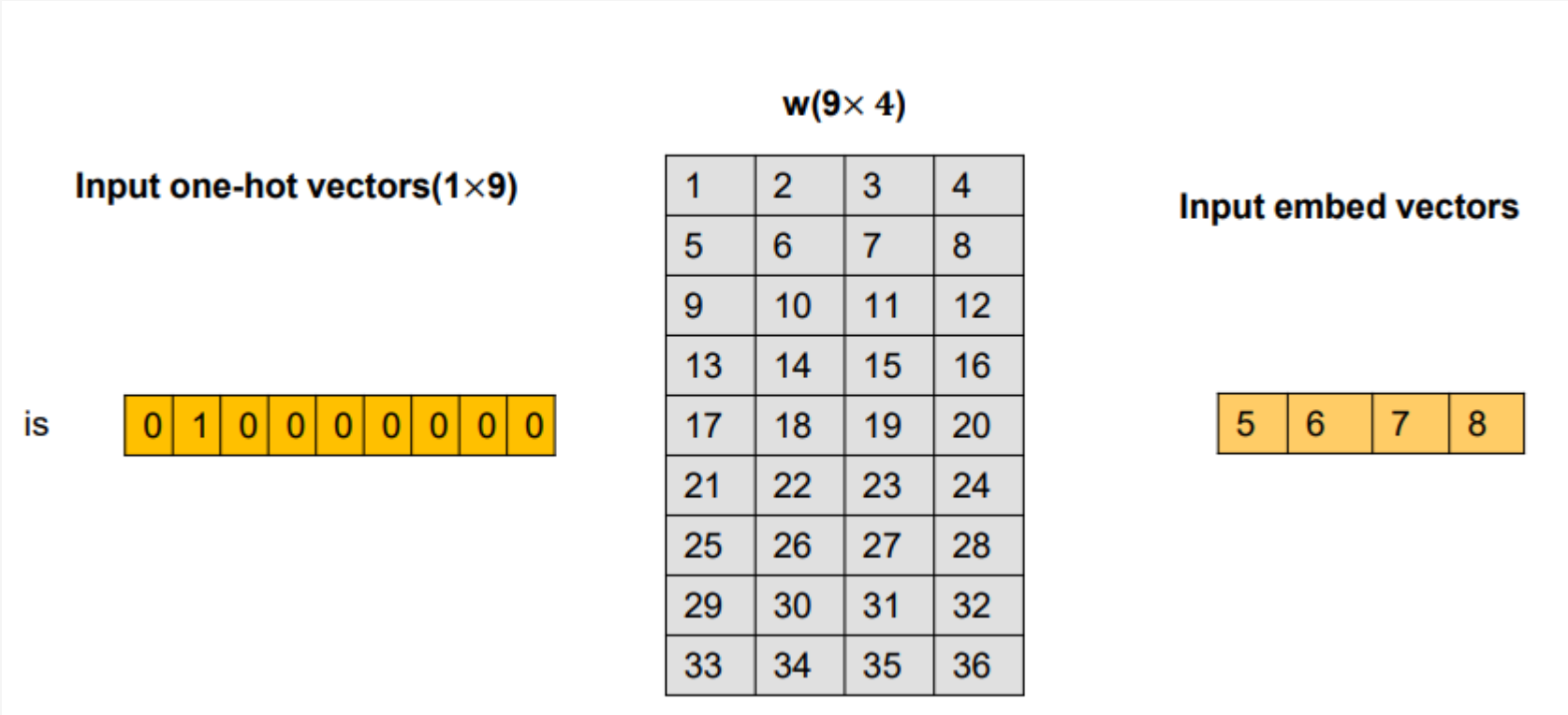
w(9× 4)

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16
17	18	19	20
21	22	23	24
25	26	27	28
29	30	31	32
33	34	35	36

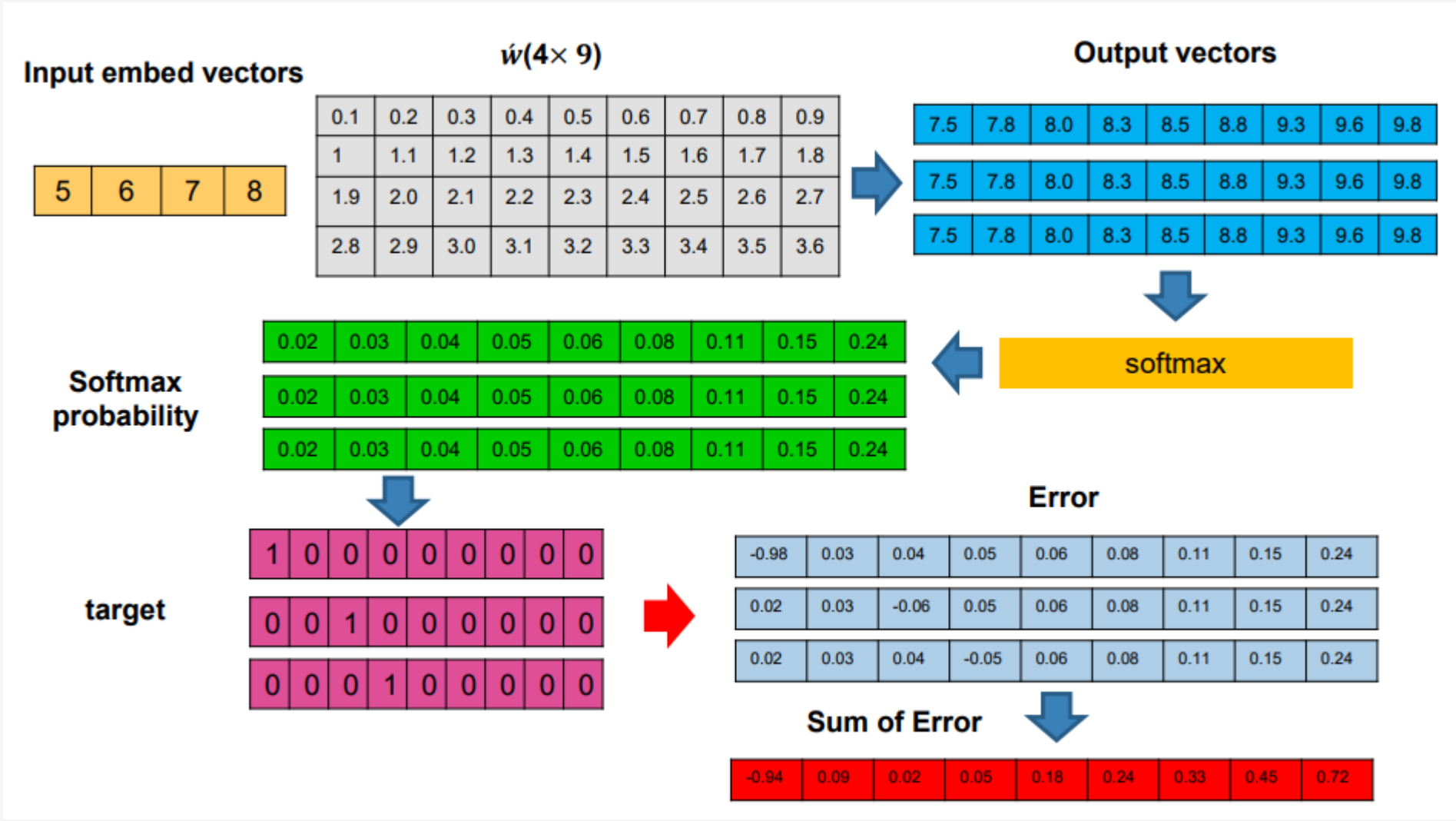
Input embed vectors

5	6	7	8

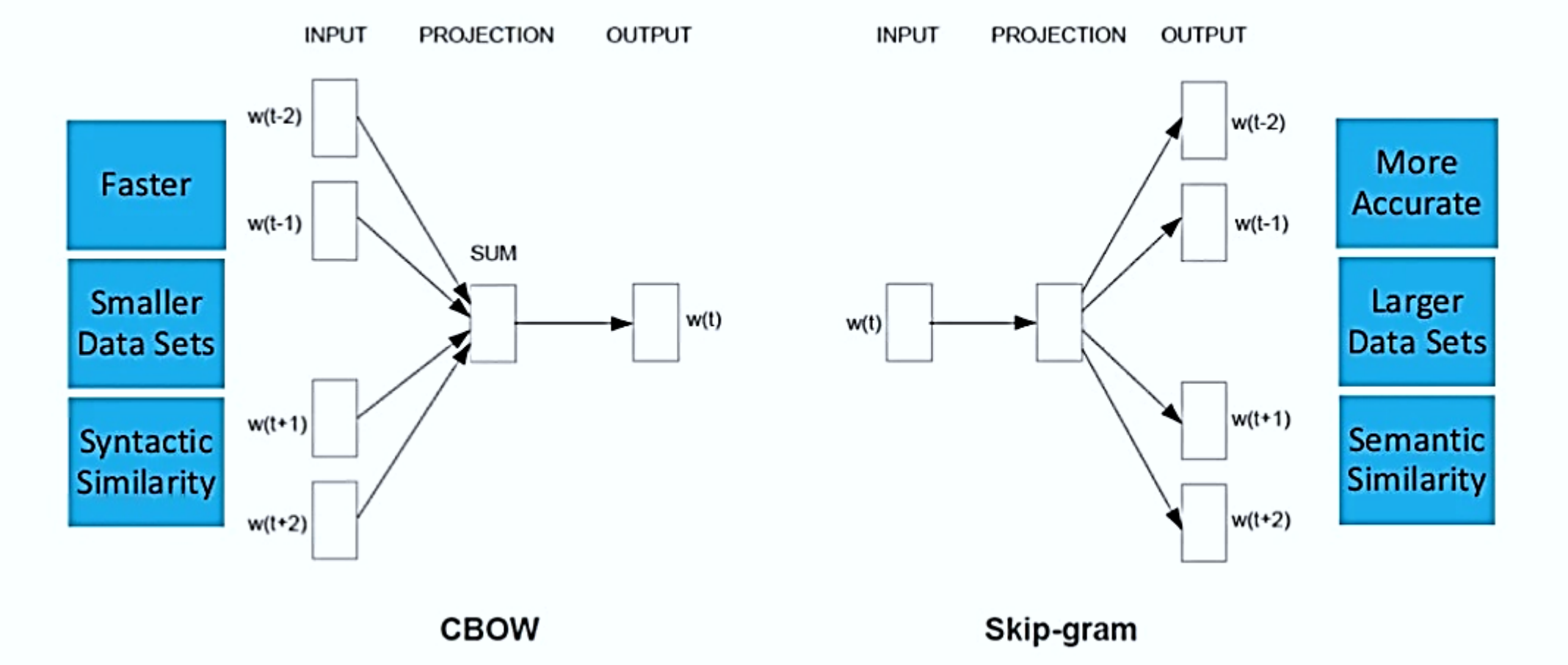
Skip Gram


w(9× 4)
Input one-hot vectors(1×9)
Input embed vectors
is
0 1 0 0 0 0 0 0 0
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
17 18 19 20
21 22 23 24
25 26 27 28
29 30 31 32
33 34 35 36
5 6 7 8

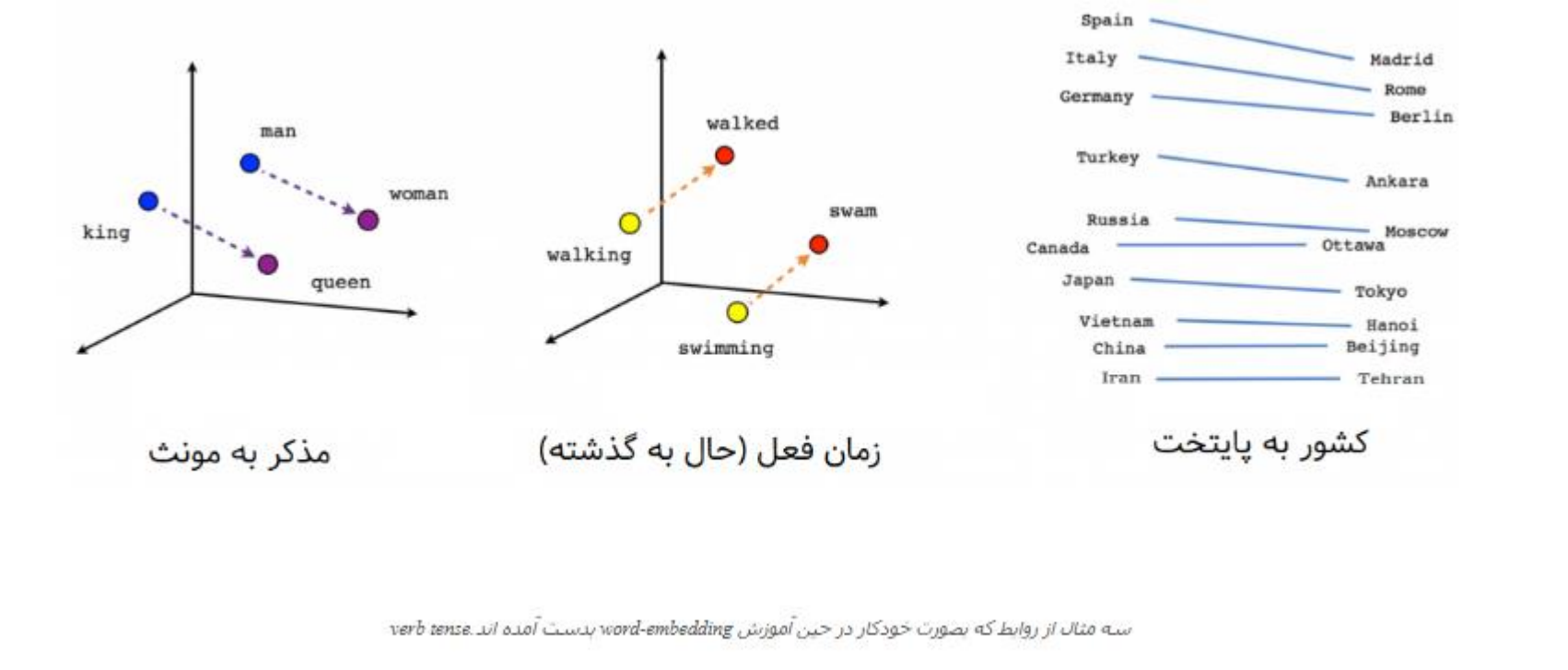
Skip Gram


Input embed vectors
5 6 7 8
$\acute{w}(4\times 9)$
0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9
1 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8
1.9 2.0 2.1 2.2 2.3 2.4 2.5 2.6 2.7
2.8 2.9 3.0 3.1 3.2 3.3 3.4 3.5 3.6
Output vectors
7.5 7.8 8.0 8.3 8.5 8.8 9.3 9.6 9.8
7.5 7.8 8.0 8.3 8.5 8.8 9.3 9.6 9.8
7.5 7.8 8.0 8.3 8.5 8.8 9.3 9.6 9.8
softmax
Softmax probability
0.02 0.03 0.04 0.05 0.06 0.08 0.11 0.15 0.24
0.02 0.03 0.04 0.05 0.06 0.08 0.11 0.15 0.24
0.02 0.03 0.04 0.05 0.06 0.08 0.11 0.15 0.24
target
1 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0
Error
-0.98 0.03 0.04 0.05 0.06 0.08 0.11 0.15 0.24
0.02 0.03 -0.06 0.05 0.06 0.08 0.11 0.15 0.24
0.02 0.03 0.04 -0.05 0.06 0.08 0.11 0.15 0.24
Sum of Error
-0.94 0.09 0.02 0.05 0.18 0.24 0.33 0.45 0.72

CBOW در برابر Skip gram



مثال شهودی



سه مثال از روابط که بصورت خودکار در حین آموزش word-embedding بدست آمده اند. verb tense

vector[Queen] ≈ vector[King] - vector[Man] + vector[Woman]

vector[Paris] ≈ vector[France] - vector[Italy] + vector[Rome]

This can be interpreted as "France is to Paris as Italy is to Rome".

نمونه خروجی Word2vev

wiki.fa.vec

```
420084 300
</s> -0.058774 -0.23785 -0.15044 -0.0076118 0.17768 -0.076418 0.16277
-0.18082 0.023249 -0.24385 -0.55391 -0.31434 0.26337 0.078366 0.20673
-0.037647 -0.1916 0.22767 0.12238 -0.2218 0.086858 -0.056307 0.1331 0
0.1509 -0.089207 -0.30398 -0.19524 0.16396 -0.11648 -0.034968 0.24025
-0.090756 0.082555 0.049204 0.05404 0.37177 -0.14526 0.12496 -0.01455
-0.064753 -0.12338 0.21999 0.081076 0.19399 -0.071562 0.079888 -0.396
0.28789 -0.24307 0.29 0.057158 -0.10354 -0.063251 0.10178 -0.25087 0.
0.090055 0.0080295 0.032952 -0.1596 0.083778 -0.30118 0.21524 -0.3094
-0.15204 0.27346 0.26063 -0.069035 0.005754 -0.24983 0.1011 0.081995
-0.15271 -0.029234 -0.20472 -0.016967 -0.12034 0.21056 -0.022778 -0.0
0.17977 0.26276 0.085994 0.0098577 -0.13044 -0.07621 -0.035098 0.1092
-0.27135 -0.1704 -0.030577 -0.0085062 0.13006 0.10278 0.32299 -0.2984
0.031971 0.089362 -0.23218 0.27914 0.087383 -0.11554 0.22623 -0.22949
-0.25859 0.047336 0.25121 -0.1688 0.15191 -0.14148 -0.26169 0.079751
```

Glove, 2014

GloVe: Global Vectors for Word Representation

Jeffrey Pennington, Richard Socher, Christopher D. Manning
Computer Science Department, Stanford University, Stanford, CA 94305
jpennin@stanford.edu, richard@socher.org, manning@stanford.edu

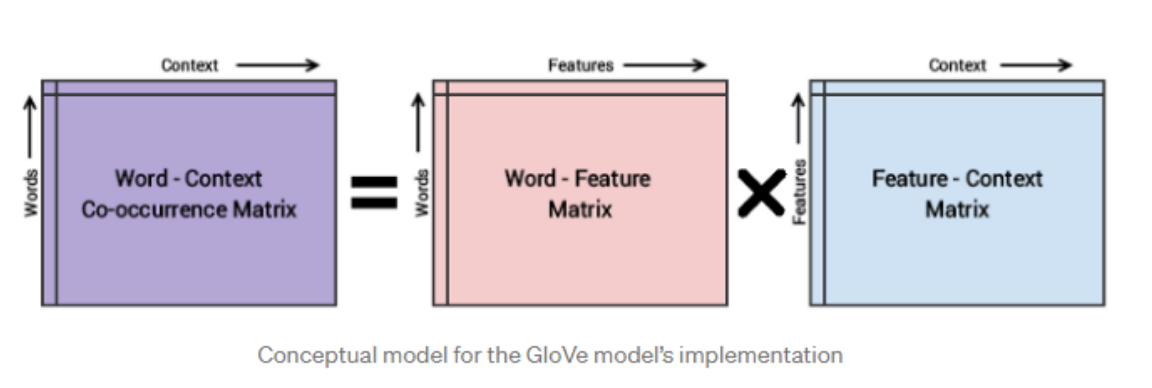
Abstract

Recent methods for learning vector space representations of words have succeeded in capturing fine-grained semantic and syntactic regularities using vector arithmetic, but the origin of these regularities

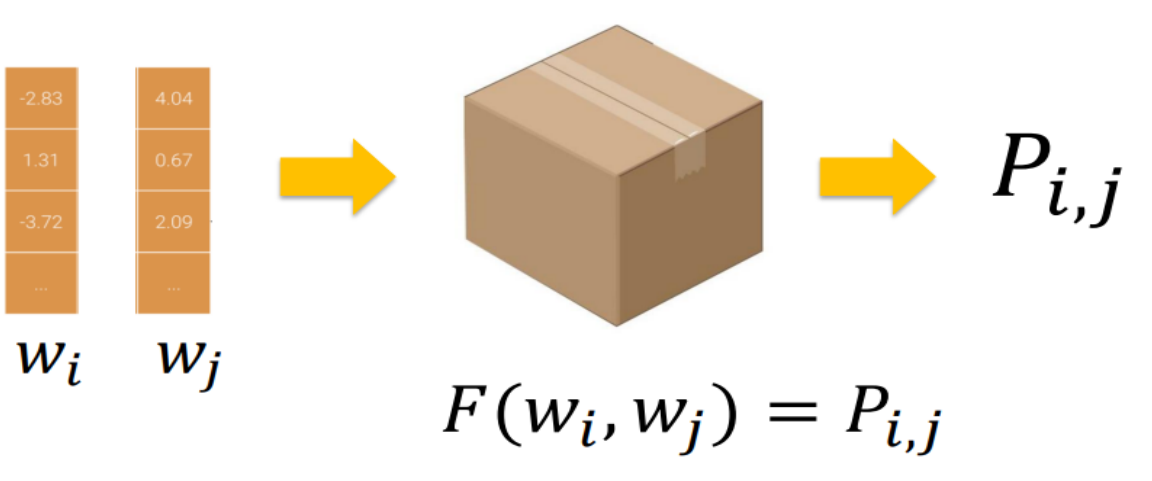
the finer structure of the word vector space by examining not the scalar distance between word vectors, but rather their various dimensions of difference. For example, the analogy "king is to queen as man is to woman" should be encoded in the vector space by the vector equation *king* –

Glove

- ایده
  - مطرح شده در سال ۲۰۱۴ توسط استنفورد
  - رفع ایرادات وارده به Word2Vec
    - پیاده سازی شده به عنوان یک تسک طبقه بندی و در نظر گرفتن بازنمایی به عنوان هدف ثانویه
    - عدم استفاده از اطلاعات مفید کلی (ماتریس هم وقوعی)
  - روش مبتنی بر شمارش
  - سریع و گسترش پذیر
  - پیاده سازی با روش آماری مبتنتی بر رگرسیون
  - روال:
    - با داشتن احتمال P_{ij} تابع نگاشتی پیدا شود که دو بردار i و j را به این احتمال نگاشت کند.



Conceptual model for the GloVe model's implementation



Glove

- Window length 1
- Example corpus:
  - I like deep learning
  - I like NLP
  - I enjoy flying

x_{ij}

counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

CO-OCCURRENCE PROBABILITY

$$P_{ij} = P(j|i) = \frac{X_{ij}}{X_i} = \frac{X_{ij}}{\sum_k X_{ik}},$$

X_{ij} = number of times word j occurs in the context of word i.

Glove

Table 1: Co-occurrence probabilities for target words *ice* and *steam* with selected context words from a 6 billion token corpus. Only in the ratio does noise from non-discriminative words like *water* and *fashion* cancel out, so that large values (much greater than 1) correlate well with properties specific to ice, and small values (much less than 1) correlate well with properties specific of steam.

Probability and Ratio	k = *solid*	k = *gas*	k = *water*	k = *fashion*
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

Crucial insight: Ratios of co-occurrence probabilities can encode meaning components

	x = solid	x = gas	x = water	x = random
$P(x\|\text{ice})$	large	small	large	small
$P(x\|\text{steam})$	small	large	large	small
$\frac{P(x\|\text{ice})}{P(x\|\text{steam})}$	large	small	~1	~1

- مفهوم نسبت هم وقوعی کلمات
  - بیان بهتر ارتباط کلمات
  - دارای اطلاعات بیشتر
- استفاده از کلمه کمکی
  - نیاز به تابع جدید برای ۳ بردار

$$F(w_i, w_j, \tilde{w}_k) = \frac{P_{ik}}{P_{jk}}$$

Glove

$$F(w_i, w_j, \tilde{w}_k) = \frac{P_{ik}}{P_{jk}}$$

Q: How can we capture ratios of co-occurrence probabilities as linear meaning components in a word vector space?

A: Log-bilinear model: $w_i \cdot w_j = \log P(i|j)$

with vector differences $w_x \cdot (w_a - w_b) = \log \frac{P(x|a)}{P(x|b)}$

Glove: روابط

چون اختلاف دو احتمال اسکالر هست، برابر با ضرب داخلی

$$F(w_i - w_j, \tilde{w}_k) = \frac{P_{ik}}{P_{jk}}$$

مدل کردن نسبت احتمالات در فضای برداری

$$F(w_i, w_j, \tilde{w}_k) = \frac{P_{ik}}{P_{jk}}$$

باز کردن ضرب ها و برابر قرار دادن صورت‌ها و مخرج‌ها

$$F\left((w_i - w_j)^T \tilde{w}_k\right) = \frac{P_{ik}}{P_{jk}}$$

$$F\left((w_i - w_j)^T \tilde{w}_k\right) = F\left(w_i^T \tilde{w}_k - w_j^T \tilde{w}_k\right) = \frac{F(w_i^T \tilde{w}_k)}{F(w_j^T \tilde{w}_k)}$$

$$F\left((w_i - w_j)^T \tilde{w}_k\right) = \frac{P_{ik}}{P_{jk}}$$

$$\frac{F(w_i^T \tilde{w}_k)}{F(w_j^T \tilde{w}_k)} = \frac{P_{ik}}{P_{jk}}$$

قانون هوموورفیسم در بردارها

$$F(w_i^T \tilde{w}_k) = P_{ik} = \frac{X_{ik}}{X_i}$$

توابعی هستند برای رابطه فوق، اینجا e^x انتخاب شده

$$F(x) = e^x$$

$$F(w_i^T \tilde{w}_k) = e^{w_i^T \tilde{w}_k} = P_{ik} = \frac{X_{ik}}{X_i}$$

Then taking the natural logarithm of both sides in the above equation

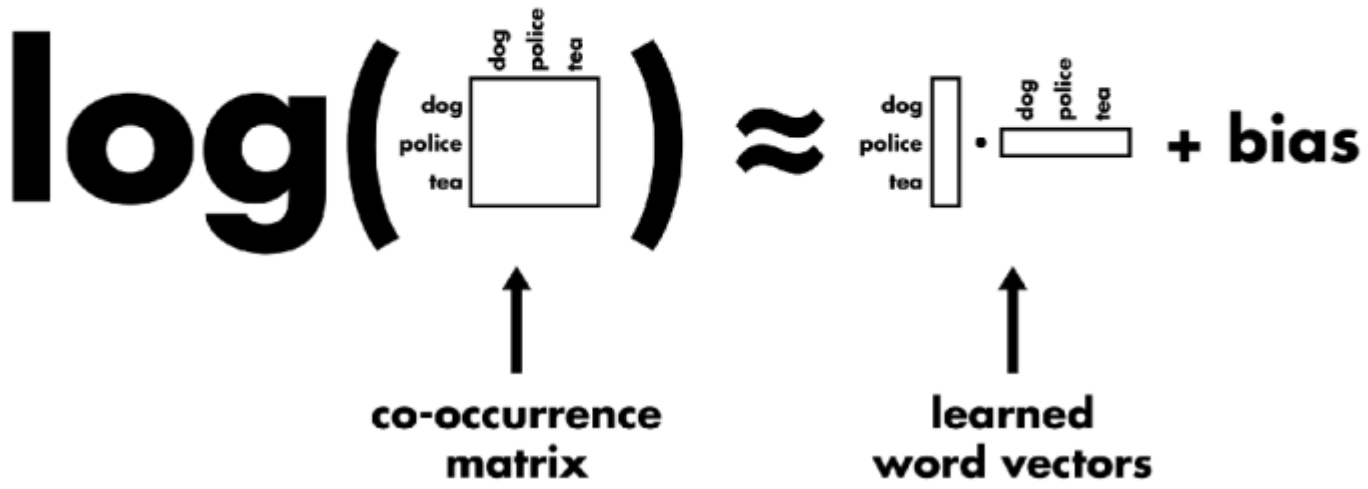
$$w_i^T \tilde{w}_k = \log P_{ik} = \log\left(\frac{X_{ik}}{X_i}\right) = \log X_{ik} - \log X_i$$

وابسته به K نیست، عدد ثابت: بایاس b_i

Glove

$$w_i^T \tilde{w}_k + b_i = \log X_{ik}$$

رابطه نگاشت پیدا شد ☺



مقایسه Glove و Word2vec

	Word2vec	Glove
نوع	مبتنی بر پیش بینی	مبتنی بر شمارش
تسک هدف	طبقه بندی	رگرسیون
معماری	شبکه عصبی	ضرب داخلی (آماری)
استفاده از اطلاعات گلوبال	خیر	بله
سرعت	کندتر (به علت استریم خواندن داده)	تندتر
تزریق اطلاعات هم وقوعی به سیستم	تدریجی	یک باره
مقیاس پذیری	متوسط	بالا

FastText, 2016

Enriching Word Vectors with Subword Information

Piotr Bojanowski* and **Edouard Grave*** and **Armand Joulin** and **Tomas Mikolov**
Facebook AI Research
{bojanowski,egrave,ajoulin,tmikolov}@fb.com

Abstract

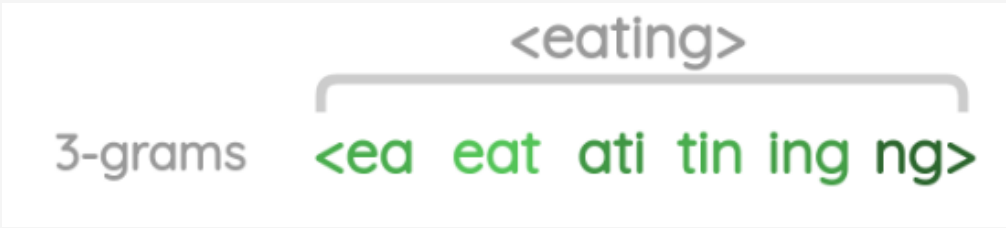
Continuous word representations, trained on large unlabeled corpora are useful for many natural language processing tasks. Popular

et al., 2010; Baroni and Lenci, 2010). In the neural network community, Collobert and Weston (2008) proposed to learn word embeddings using a feed-forward neural network, by predicting a word based

FastText, 2016

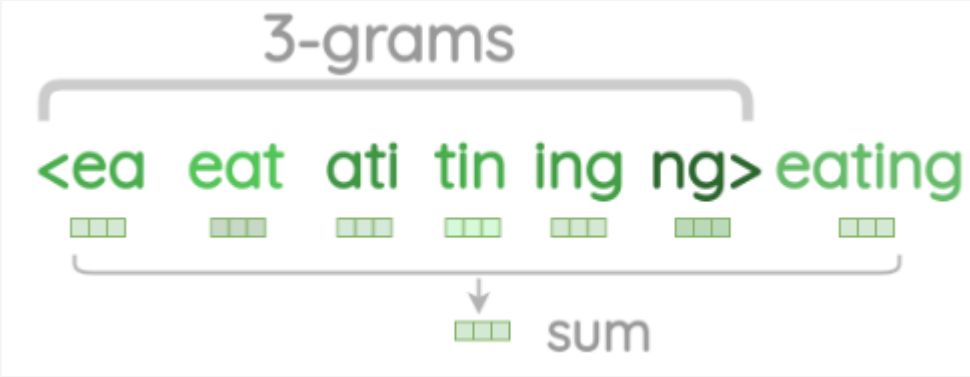
- ایده
- مطرح شده در سال ۲۰۱۶ توسط فیسبوک
- مبتنی بر word2vec
- رفع ایرادات وارده به Word2Vec و Glove
  - در نظر نگرفتن اطلاعات مورفولوژیکی (کلمات هم ریشه یا صرف افعال)
  - ایجاد بازنمایی فقط برای کلمات دیده شده در پیکره
- استفاده از n-gram های کارکترها و خود کلمه
- مناسب استفاده در دادگان با حجم کوچکتر

FastText



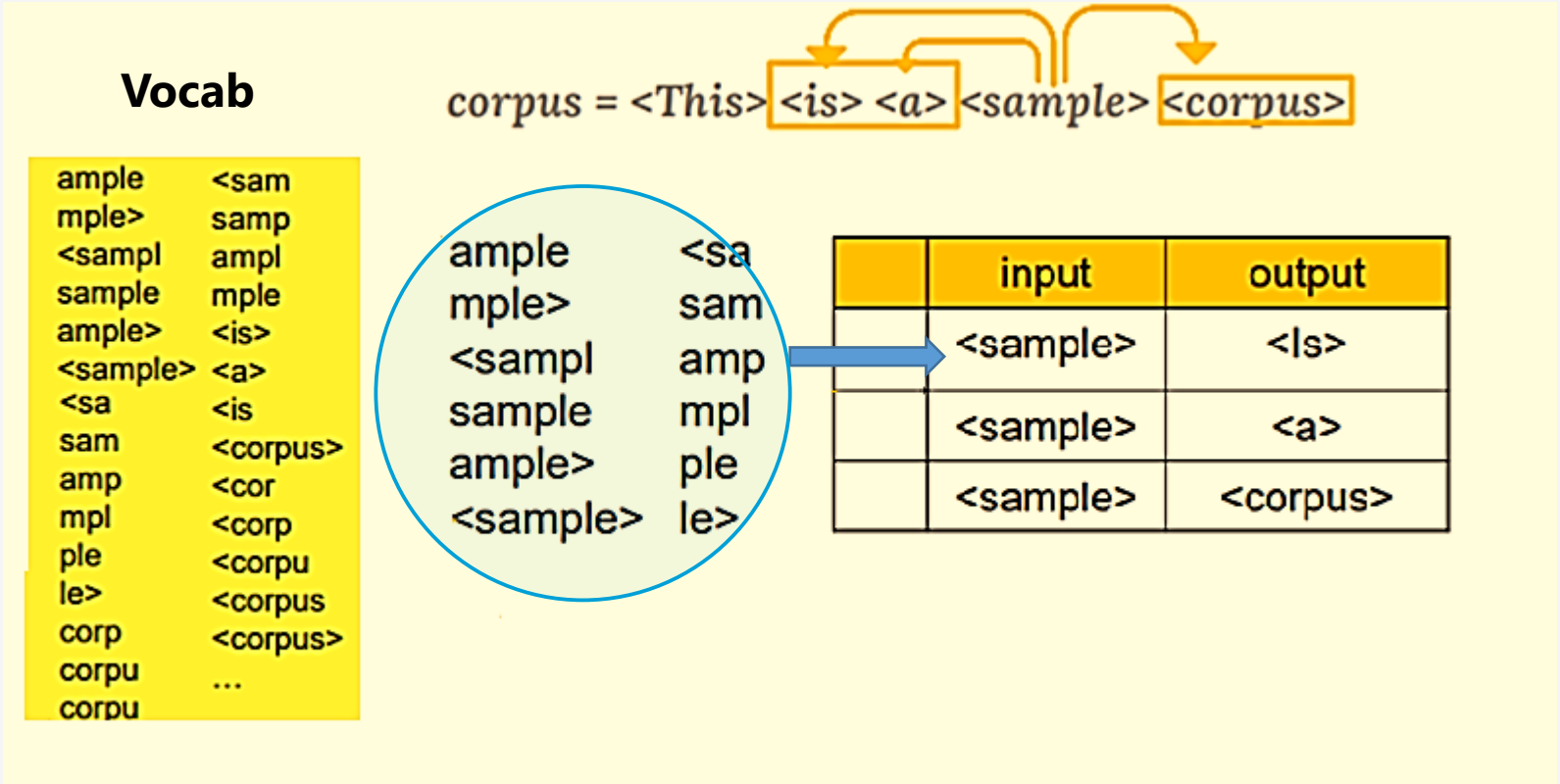
Word	Length(n)	Character n-grams
eating	3	<ea, eat, ati, tin, ing, ng>
eating	4	<eat, eati, atin, ting, ing>
eating	5	<eati, eatin, ating, ting>
eating	6	<eatin, eating, ating>

- مرحله اول: تعیین مرزهای هر کلمه
- استخراج n-gram های کلمه با استفاده از پنجره لغزان n تایی
- استفاده از n-gram های کارکترها و خود کلمه
  - پیش فرض n بین ۳ و ۶
  - خود کلمه
- در نهایت استفاده از مجموع بردارهای n-gram ها برای جاسازی کلمه



FastText

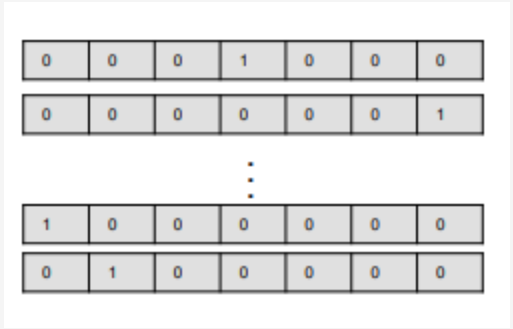
- پیاده سازی Fasttext Skip-gram
- Sample و <Sample> متفاوت هستند.
- n-gram های بین ۳ و ۶ در نظر گرفته می شوند.



بخشی از تصاویر مثال برداشته شده از Deep NLP workshop (IKT,2021)

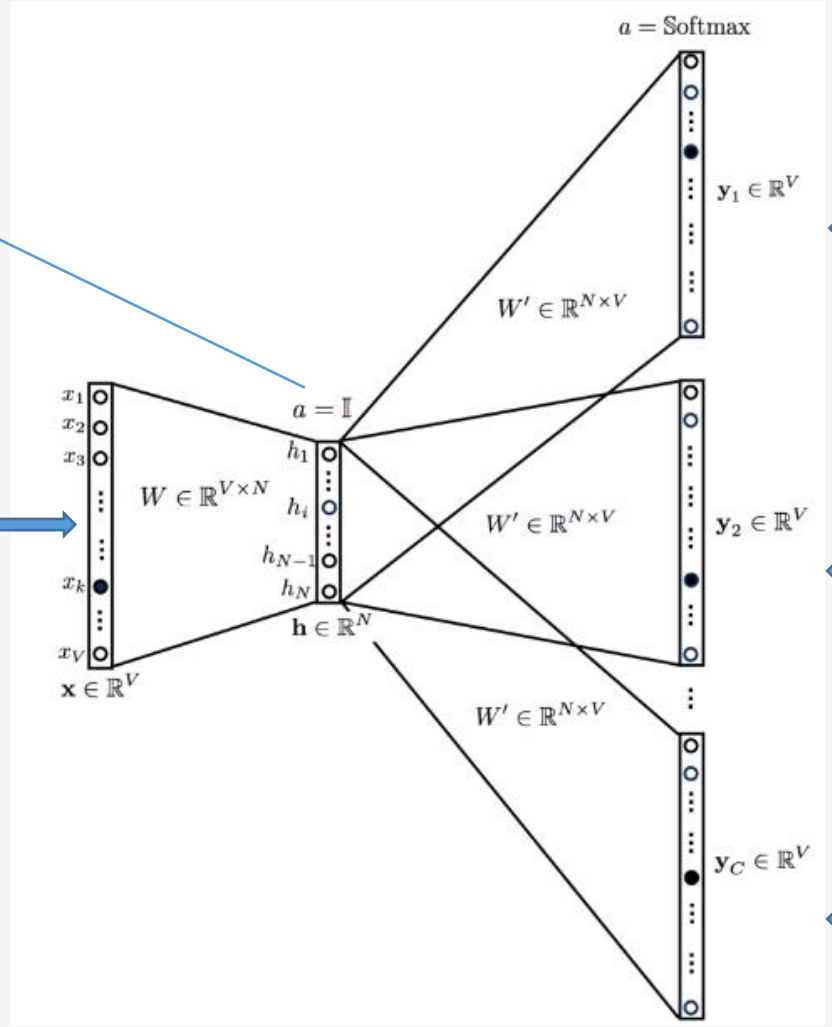
Skip Gram

جمع بردارهای n-gram



<Sample>

ample <sa
mple> sam
<sampl amp
sample mpl
ample> ple
<sample> le>

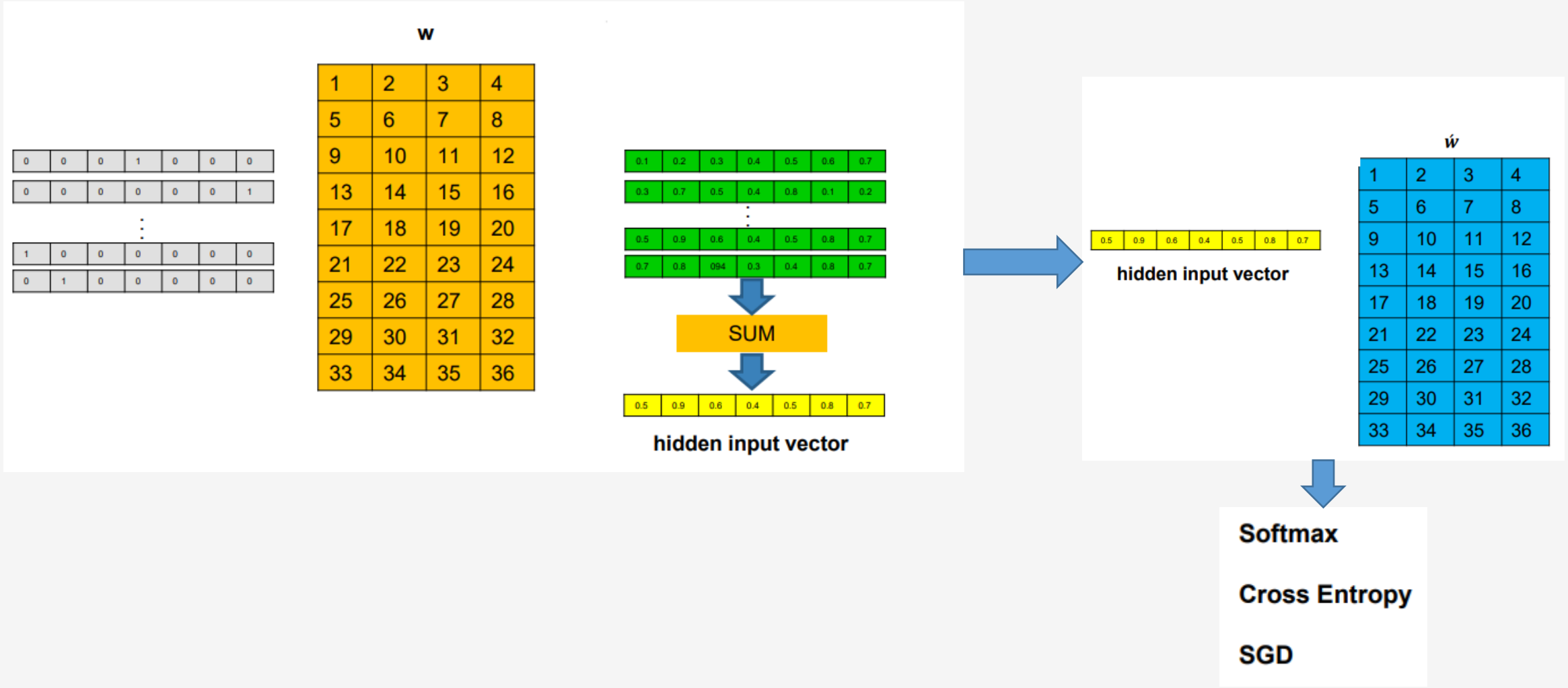


<is>

<a>

<corpus>

FastText


w
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
17 18 19 20
21 22 23 24
25 26 27 28
29 30 31 32
33 34 35 36
SUM
hidden input vector
hidden input vector
ẃ
Softmax
Cross Entropy
SGD

ELMO, 2018

NAACL 2018 best paper

Deep contextualized word representations

Matthew E. Peters[†], **Mark Neumann**[†], **Mohit Iyyer**[†], **Matt Gardner**[†],
{matthewp,markn,mohiti,mattg}@allenai.org

Christopher Clark[*], **Kenton Lee**[*], **Luke Zettlemoyer**[†*]
{csquared,kentonl,lsz}@cs.washington.edu

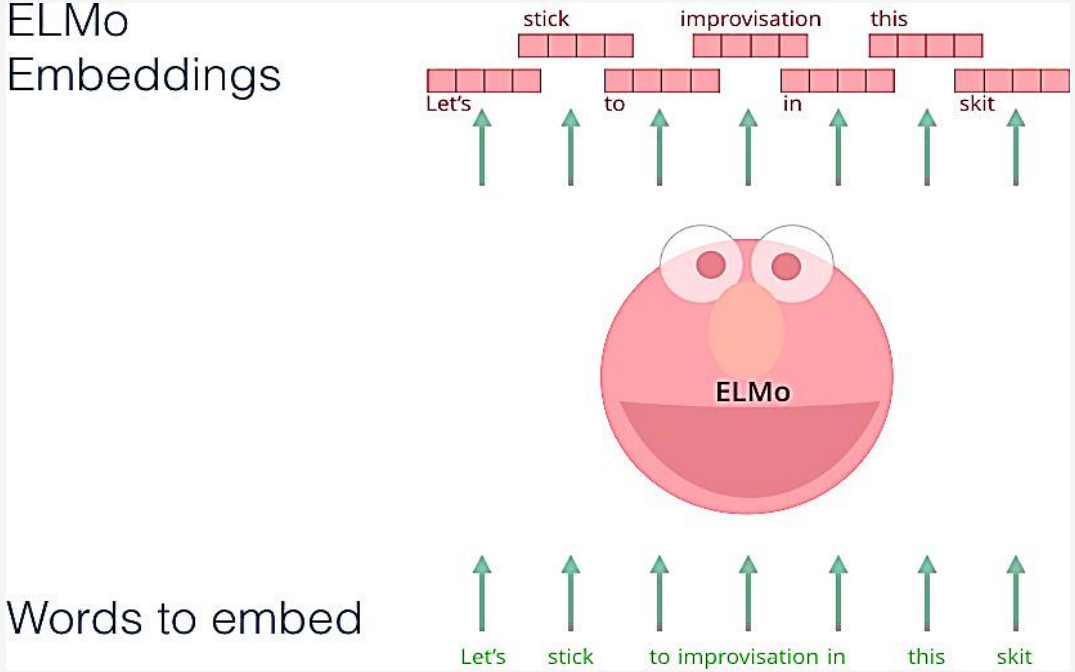
[†]Allen Institute for Artificial Intelligence
[*]Paul G. Allen School of Computer Science & Engineering, University of Washington

Abstract

We introduce a new type of *deep contextualized* word representation that models both (1) complex characteristics of word use (e.g., syntax and semantics), and (2) how these uses vary across linguistic contexts (i.e., to model polysemy). Our word vectors are learned functions of the internal states of a deep bidirectional language model (biLM), which is pre-guage model (LM) objective on a large text corpus. For this reason, we call them ELMo (Embeddings from Language Models) representations. Unlike previous approaches for learning contextualized word vectors (Peters et al., 2017; McCann et al., 2017), ELMo representations are deep, in the sense that they are a function of all of the internal layers of the biLM. More specifically, we learn a linear combination of the vectors stacked

ELMO

Embedding from **L**anguage **Mo**del



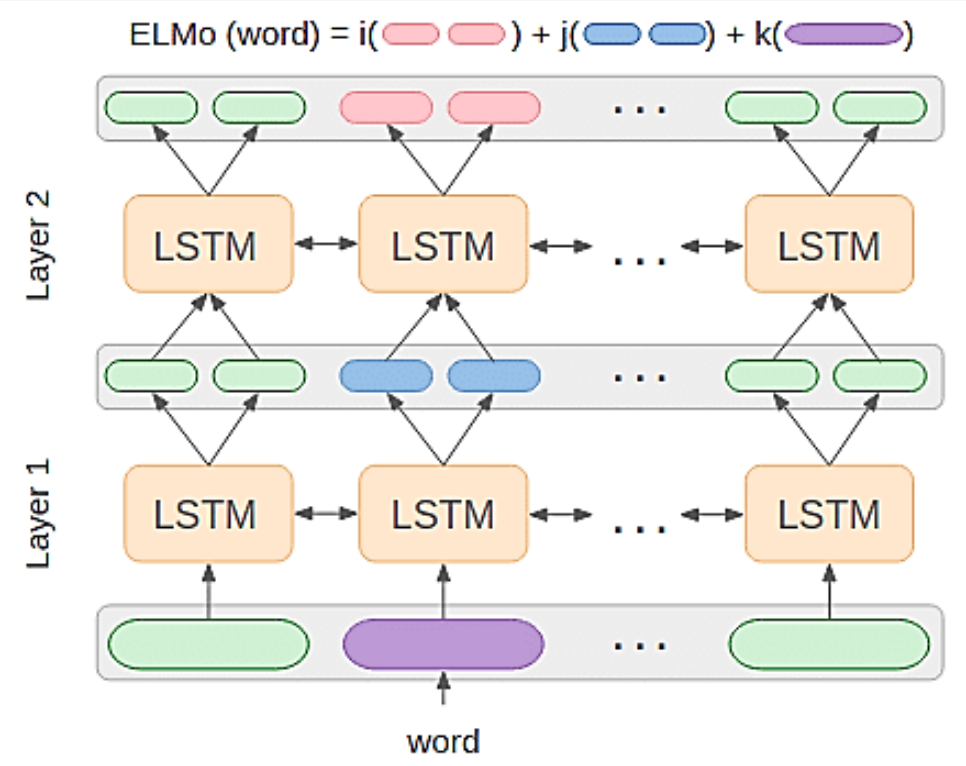
- استفاده از خصوصیات نحوی و معنایی کلمه در تولید بازنمایی
- به جای تولید جاسازی ثابت برای کلمه
  - دیدن کل جمله و استفاده از بافتار طولانی به جای پنجره
  - ایجاد بازنمایی های مختلف برای یک کلمه در بافتارهای مختلف
- استفاده از مدل زبانی دوطرفه
  - آموزش آن با ۳۰ میلیون داده خام (در مقاله اصلی)

1. Jobs was the CEO of **apple**
2. He finally ate the **apple**

Figure from http://jalammar.github.io/illustrated-bert/

ELMO

biLM: Bidirectional Language Models

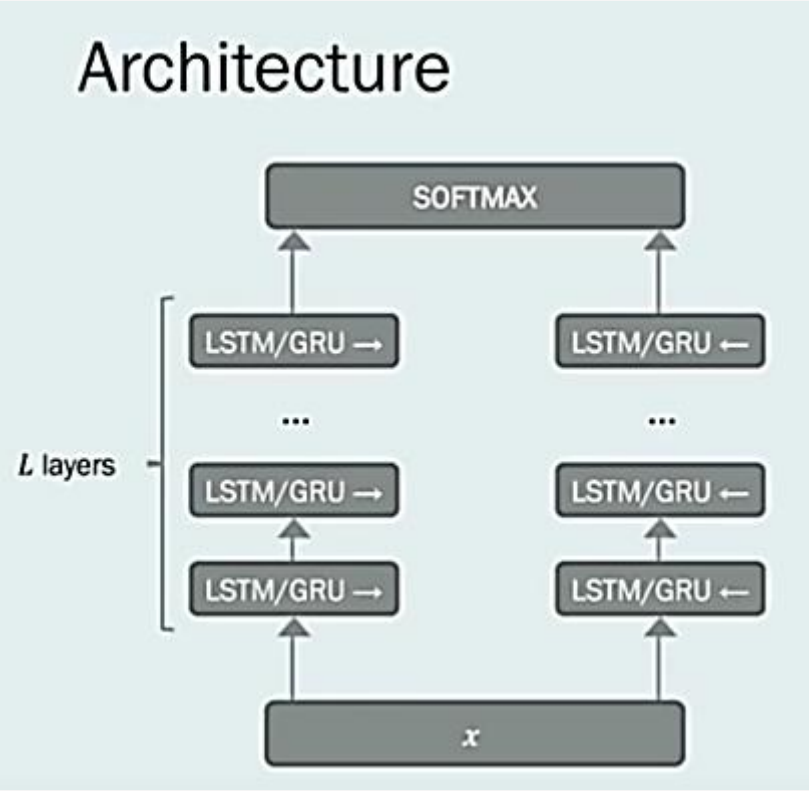


- ۲ گام:
  - استفاده از مدل زبانی دو طرفه
    - دیدن یک SEQ و حدس احتمال کلمه بعدی
    - نگاه کردن از هر دو طرف
    - استفاده از LSTM های پشت سر هم
    - استفاده از بازنمایی همه لایه ها برای هر کلمه
  - آموزش یک ترکیب خطی از بازنمایی ها به یک تسک نهایی
    - ترکیب مبتنی بر تسک بازنمایی لایه های میانی مدل زبانی

biLMs consist of forward and backward LMs

- Forward: $p(t_1, t_2, \ldots, t_N) = \prod_{k=1}^{N} p(t_k \mid t_1, t_2, \ldots, t_{k-1})$
- Backward: $p(t_1, t_2, \ldots, t_N) = \prod_{k=1}^{N} p(t_k \mid t_{k+1}, t_{k+2}, \ldots, t_N)$

ELMO



- استفاده از مدل زبانی با هدف مدل کردن کلمات به صورت وابسته به هم
- استفاده از LSTM های پشت سرهم با هدف اینکه هر لایه یکسری اطلاعات را در نظر می گیرند.
  - اطلاعات نحوی در لایه های پایین
  - اطلاعات معنایی در لایه های بالایی

- مدل زبانی
  - یک بازنمایی مستقل از بافتار برای هر توکن آماده می کند.
    - بر اساس کارکتر (مفید برای کلمات دیده نشده)

- آن را از L لایه LSTM عبور می دهد.
- در هر لایه LSTM یک بازنمایی وابسته به بافتار ارائه می کند. (ویژگی هیدن)
- $H_{k,j}: j=1,..,L$

ELMO

$$\sum_{k=1}^{N} \Big(\log p(t_k \mid t_1, \ldots, t_{k-1}; \Theta_x, \overrightarrow{\Theta}_{LSTM}, \Theta_s) + \log p(t_k \mid t_{k+1}, \ldots, t_N; \Theta_x, \overleftarrow{\Theta}_{LSTM}, \Theta_s) \Big).$$

- آموزش پارامترها با کمینه کردن loglikelihood در مسیر رفت و برگشت
- پارامترهای مشترک برای بازنمایی توکن و لایه softmax
- پارامترهای جداگانه برای LSTM ها در هر مسیر
- برای هر توکن در یک مدل زبانی دوطرفه با L لایه
  - 2L+1 بازنمایی

$$\begin{aligned} R_k &= \{\mathbf{x}_k^{LM}, \overrightarrow{\mathbf{h}}_{k,j}^{LM}, \overleftarrow{\mathbf{h}}_{k,j}^{LM} \mid j = 1, \ldots, L\} \\ &= \{\mathbf{h}_{k,j}^{LM} \mid j = 0, \ldots, L\}, \end{aligned}$$

بازنمایی مستقل از بافتار ورودی برای هر توکن

برای k=0 لایه توکن است و برای kهای بزرگتر بازنمایی به ازای هر لایه LSTM

ELMO

- برای هر تسک ELMO وزن های بازنمایی ها را یاد می گیرد.
- در حالت ساده فقط لایه بالا را انتخاب می کند. $E(R_k) = \mathbf{h}_{k,L}^{LM}$
- با استفاده ازضرایب S میزان توجه به هر سطح (معنایی، نحوی و ..) تعیین می شود.
- ضریب گاما
- مفید بودن کلی ELMO برای هر تسک

$$\mathbf{ELMo}_k^{task} = E(R_k; \Theta^{task}) = \gamma^{task} \sum_{j=0}^{L} s_j^{task} \mathbf{h}_{k,j}^{LM}. \quad (1)$$

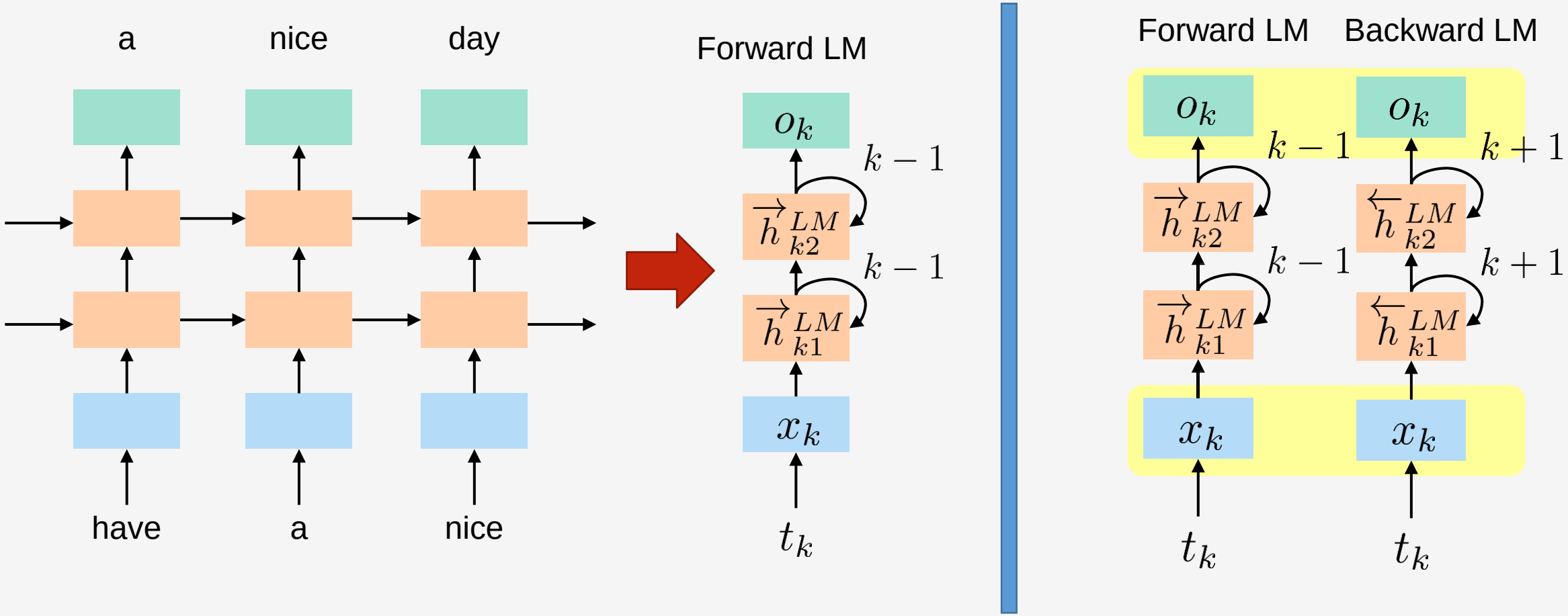
scale parameter

آنچه یاد گرفته می شود:
softmax-normalized weights

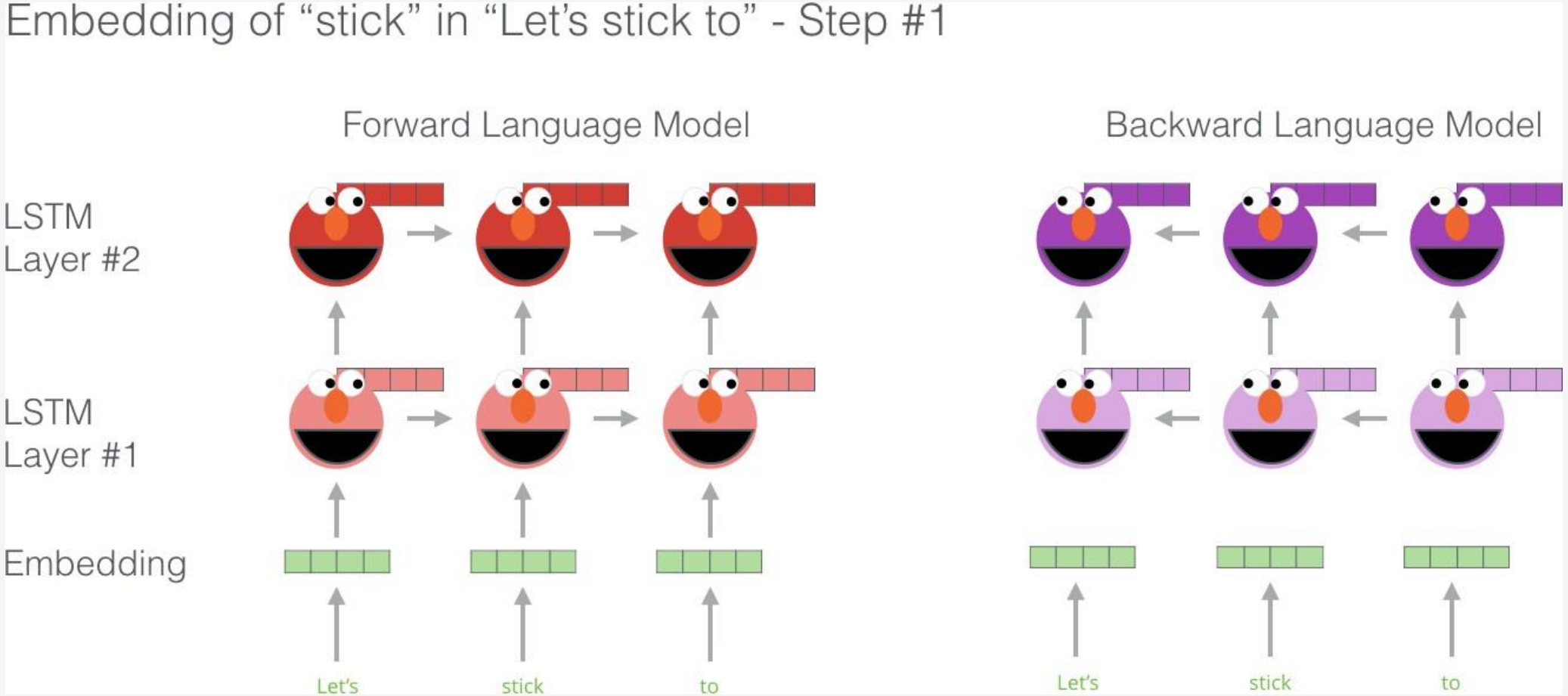
$$\mathrm{ELMo}_k^{\mathrm{task}} = \gamma^{\mathrm{task}} \times \sum \begin{cases} s_2^{\mathrm{task}} \times h_{k2}^{LM} & \overrightarrow{h}_{k2}^{LM} \quad \overleftarrow{h}_{k2}^{LM} \\ s_1^{\mathrm{task}} \times h_{k1}^{LM} & \overrightarrow{h}_{k1}^{LM} \quad \overleftarrow{h}_{k1}^{LM} \\ s_0^{\mathrm{task}} \times h_{k0}^{LM} & x_k \quad x_k \end{cases}$$

ELMO

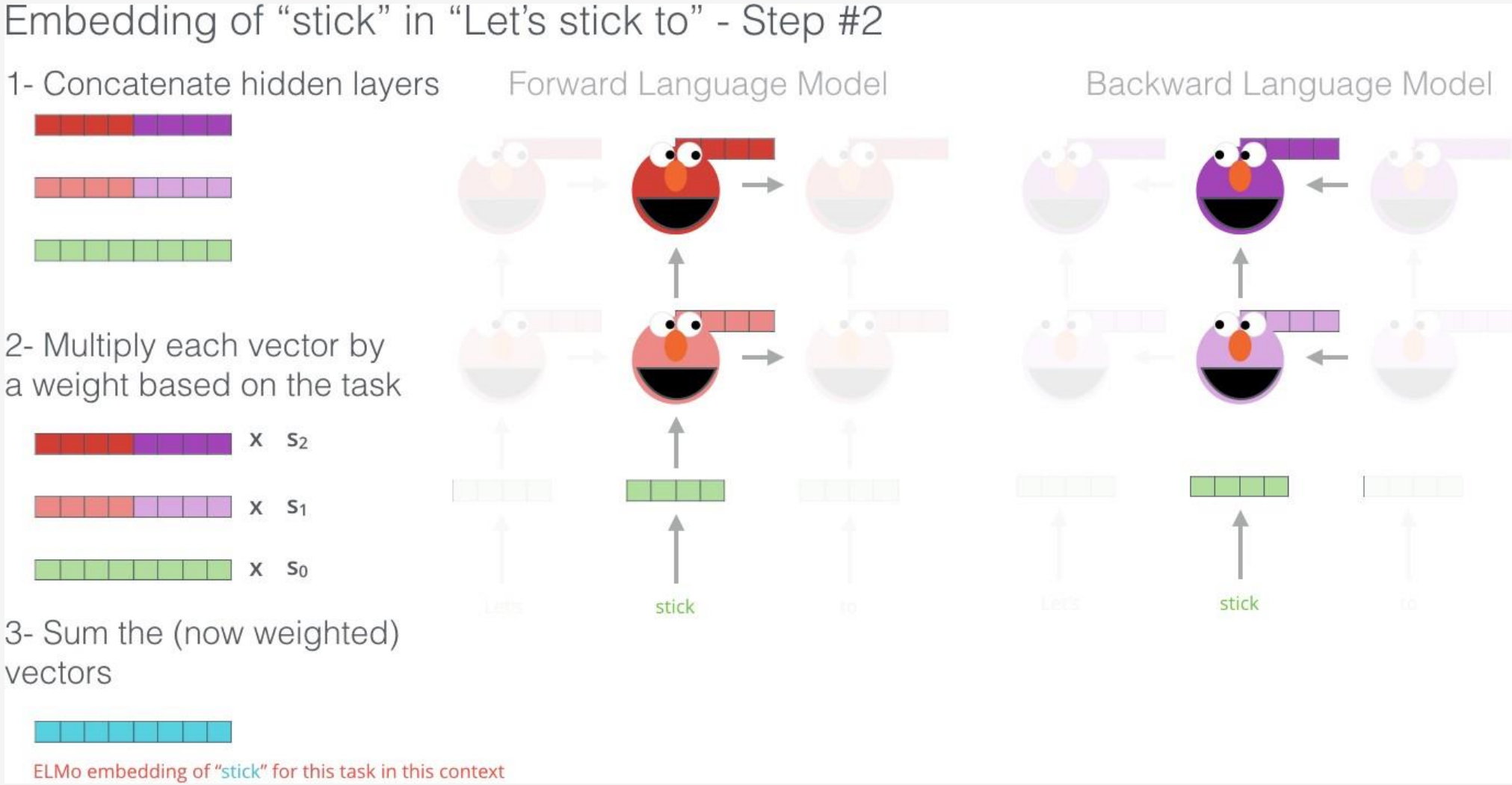
Bidirectional LM $p(t_1, t_2, \cdots, t_N) = \prod_{k=1}^{N} p(t_k \mid t_1, \cdots, t_{k-1})$



ELMO

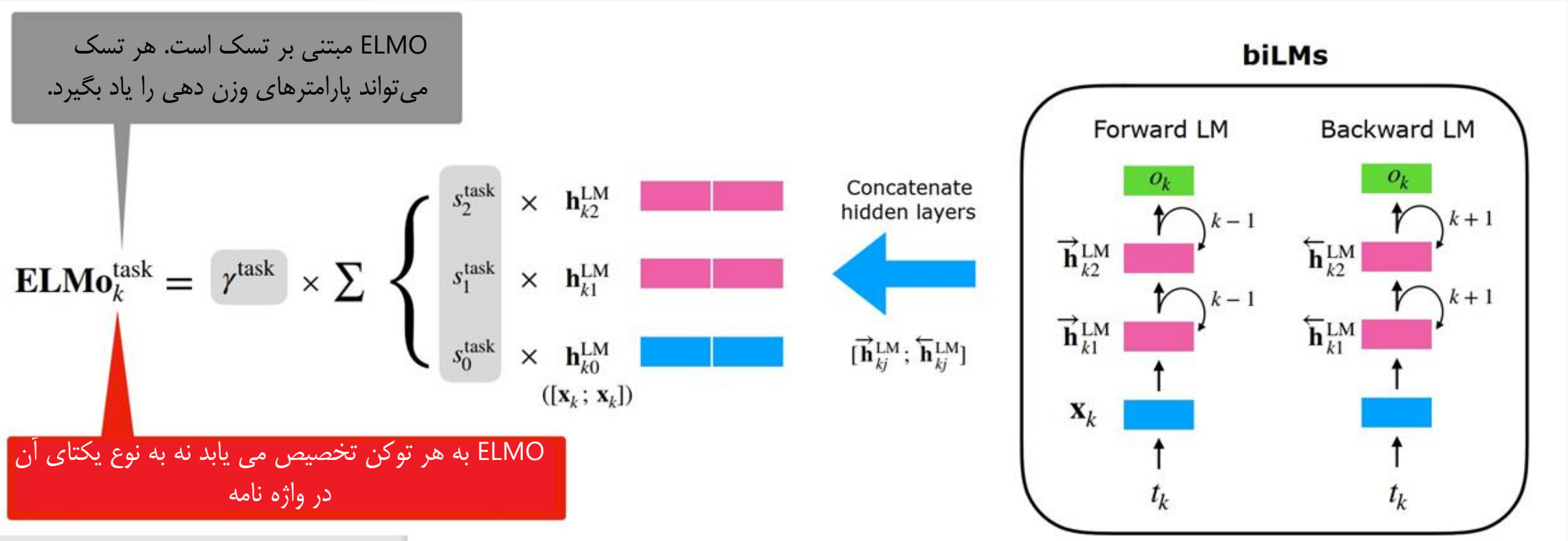

Embedding of “stick” in “Let’s stick to” - Step #1
Forward Language Model
Backward Language Model
LSTM Layer #2
LSTM Layer #1
Embedding
Let’s
stick
to
Let’s
stick
to

ELMO


Embedding of “stick” in “Let’s stick to” - Step #2
1- Concatenate hidden layers
2- Multiply each vector by a weight based on the task
x s_2
x s_1
x s_0
3- Sum the (now weighted) vectors
ELMo embedding of “stick” for this task in this context
Forward Language Model
Backward Language Model
stick
stick

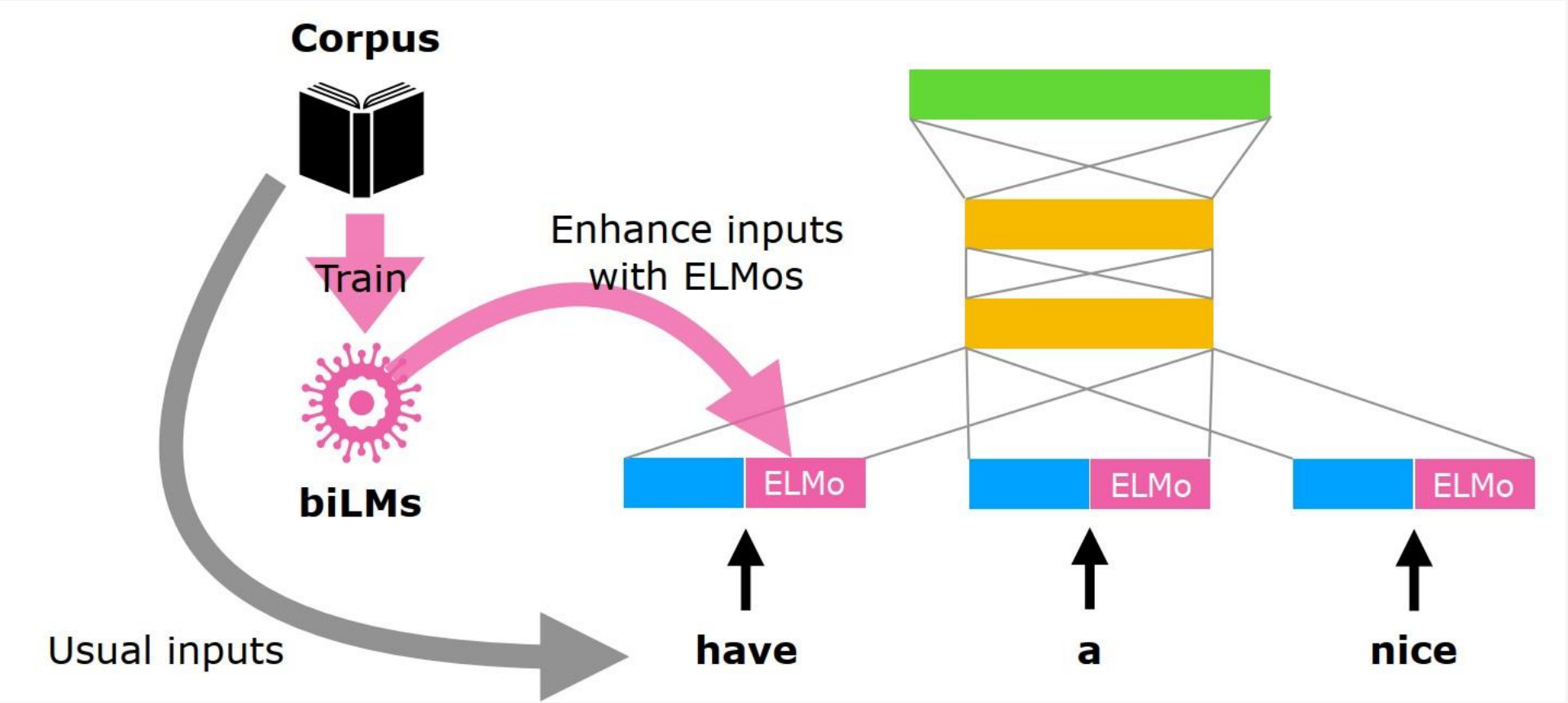
ELMO

بازنمایی هر کلمه t_k به صورت ترکیب خطی لایه های هیدن مربوطه



ELMO

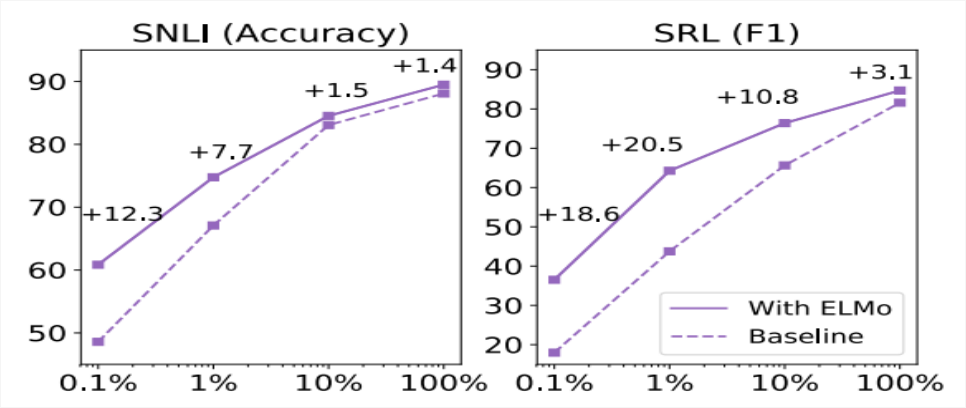
استفاده از ELMO در اکثر تسک های NLP فقط با اتصال ساده به لایه جاسازی



ELMO

	Task	Previous SOTA		Our baseline	ELMo + baseline	Increase (absolute/ relative)
Q&A	SQuAD	Liu et al. (2017)	84.4	81.1	85.8	4.7 / 24.9%
Textual entailment	SNLI	Chen et al. (2017)	88.6	88.0	88.7 ± 0.17	0.7 / 5.8%
Semantic role labelling	SRL	He et al. (2017)	81.7	81.4	84.6	3.2 / 17.2%
Coreference resolution	Coref	Lee et al. (2017)	67.2	67.2	70.4	3.2 / 9.8%
Named entity recognition	NER	Peters et al. (2017)	91.93 ± 0.19	90.15	92.22 ± 0.10	2.06 / 21%
Sentiment analysis	SST-5	McCann et al. (2017)	53.7	51.4	54.7 ± 0.5	3.3 / 6.8%

Table 1: Test set comparison of ELMo enhanced neural models with state-of-the-art single model baselines across six benchmark NLP tasks. The performance metric varies across tasks – accuracy for SNLI and SST-5; F_1 for SQuAD, SRL and NER; average F_1 for Coref. Due to the small test sizes for NER and SST-5, we report the mean and standard deviation across five runs with different random seeds. The "increase" column lists both the absolute and relative improvements over our baseline.



- مقایسه عملکرد تسک ها با و بدون ELMO
- کاهش تعداد داده آموزشی و تعداد epoch ها