



بِهِ نَامِ خَدَا

بخش نهم

گروه هوش مصنوعی، دانشکده مهندسی
کامپیوتر

نبدیل متن به بردار

Text Vectorization

حمیدرضا برادران کاشانی



- ❖ هدف درس:
- ❖ چگونه می توان از الگوریتم های یادگیری ماشین برای پردازش زبان طبیعی استفاده کرد؟
- ❖ یادگیری ماشین بر اساس عملیات ریاضی بر روی اعداد عمل می کند.
- ❖ زبان طبیعی از متن ساخته شده است که توسط کاراکترها و کلمات توصیف می شود (نه اعداد) !
- ❖ کلمات یک سری موجودیت های طبقه بندی شده (categorical) یا گستته (discrete) هستند.

- ❖ سوال اساسی؟
- ❖ چگونه می توان زبان طبیعی (متن) را به اعداد (بردارهای عددی) تبدیل کرد تا یادگیری ماشین قابل استفاده برای آن باشد؟



- ❖ در پاسخ به مساله "تبدیل متن به بردارها" موارد زیر را بررسی می کنیم:
 - ❖ پیش پردازش متن (توکن سازی، ایست واژه ها، ریشه یابی و لم سازی)
 - ❖ برخی روش های اولیه تبدیل متن به بردار
 - بردارهای دودویی
 - بردار فراوانی عبارت (TF)
 - بردار فراوانی عبارت-معکوس فراوای سند (TF-IDF)



مژور برخی مفاهیم

❖ جملات (Sentences)

- ❖ ما با بیان جملات معمولاً با یکدیگر ارتباط برقرار می کنیم (چه در گفتار و چه در نوشتار).
- ❖ جمله دنباله‌ای از کلمات (words) است.
- ❖ انتهای جملات عموماً با علائم نشان گذاری (punctuation) مثل نقطه یا علامت سوال یا تعجب و ... مشخص می شود.
- ❖ در برخی زبان‌ها مثل انگلیسی حرف ابتدای جمله بصورت حرف بزرگ (capitalized) مشخص می شود.

مرور برخی مفاهیم



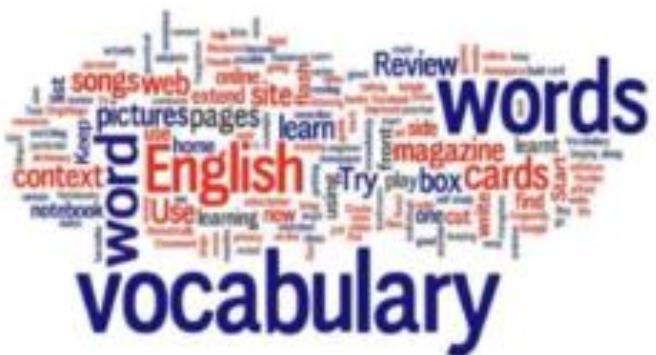
- ❖ توکن (Token)
 - ❖ در حالت کلی می توان جمله را دنباله ای از توکن ها در نظر گرفت.
 - ❖ عبارت توکن یک عبارت **کلی** تر است که می تواند به هر کدام از موارد زیر اشاره کند:
 - ❖ کلمات
 - ❖ علائم نشان گذاری
 - ❖ واحدهای زیرکلمه ای (sub-word units)
 - ❖ در ادامه منظور ما از توکن، همان کلمه است.

مرور برخی مفاهیم



❖ واژگان (Vocabulary)

- ❖ مجموعه تمام کلمات یک زبان مثلا زبان فارسی
- ❖ سایز واژگان؟ ۱۰۰۰ کلمه یا ۱۰۰.۰۰۰ یا ۱۰۰.۰۰۰.۰۰۰ یا ۱ میلیون؟؟
- ❖ می توانیم در یک کاربرد خاص، خودمان انتخاب کنیم که اندازه واژگان ما چقدر باشد.
- ❖ البته در زمانی که از یک مدل از قبل آموخته واژگان مشخص استفاده می کنیم، دیگر ما انتخابی نداریم!



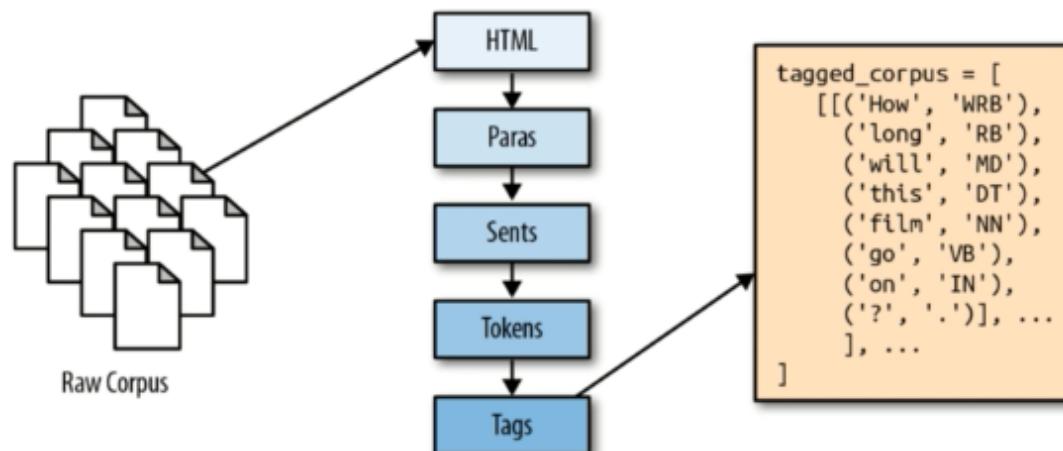
Hamidreza Baradaran Kashani

مرور برخی مفاهیم



❖ پیکره (Corpus)

- ❖ در زبانشناسی پیکره مجموعه‌ای بزرگ از متن‌های ساختار یافته است.
- ❖ معمولاً برای افزایش کارآیی پیکره‌ها از برچسب‌زنی (tagging) بر روی متون استفاده می‌شود.
- ❖ پس در اینجا منظور از پیکره همان مجموعه داده یادگیری ماشین (ML dataset) است که برای آموزش (learn) مدل استفاده می‌کنیم.



Hamidreza Baradaran Kashani



مروز برخی مفاهیم

ادامه پیکره (Corpus) -

❖ پیکره ها می توانند لزوماً یک زبان نداشته باشند، مثلاً پیکره های دو زبانه یا چند زبانه داریم که برای کاربردهای ترجمه ماشینی استفاده می شوند.



- ❖ برخی پیکره های مشهور برای پردازش زبان طبیعی:
- ❖ وردنت / شکسپیر / ویکی پدیا / رویترز و ...

NLTK هم دارای پیکره های فراوانی در آدرس زیر است:

http://www.nltk.org/nltk_data/

Hamidreza Baradaran Kashani



NLTK Corpora

NLTK has built-in support for dozens of corpora and trained models, as listed below. To use these within NLTK we recommend that you use the NLTK corpus downloader, >>> `nltk.download()`. Please consult the README file included with each corpus for further information.

1. *perluniprops: Index of Unicode Version 7.0.0 character properties in Perl* [[download](#) | [source](#)]
id: perluniprops; size: 100266; author: ; copyright: ; license: ;
2. *The monolingual word aligner (Sultan et al. 2015) subset of the Paraphrase Database.* [[download](#) | [source](#)]
id: mwa_ppdb; size: 1594711; author: ; copyright: ; license: Creative Commons Attribution 3.0 Unported (CC-BY);
3. *Punkt Tokenizer Models* [[download](#) | [source](#)]
id: punkt; size: 13905355; author: Jan Strunk; copyright: ; license: ;
4. *RSLP Stemmer (Removedor de Sufixos da Lingua Portuguesa)* [[download](#) | [source](#)]
id: rslp; size: 3805; author: Viviane Moreira Orengo (vmorengo@inf.ufrgs.br) and Christian Huyck; copyright: ; license: ;
5. *Porter Stemmer Test Files* [[download](#) | [source](#)]
id: porter_test; size: 200510; author: ; copyright: ; license: ;
6. *Snowball Data* [[download](#) | [source](#)]
id: snowball_data; size: 6785405; author: ; copyright: ; license: ;
7. *ACE Named Entity Chunker (Maximum entropy)* [[download](#) | [source](#)]
id: maxent_ne_chunker; size: 13404747; author: ; copyright: ; license: ;
8. *Moses Sample Models* [[download](#) | [source](#)]
id: moses_sample; size: 10961490; author: ; copyright: ; license: ;
9. *BLLIP Parser: WSJ Model* [[download](#) | [source](#)]
id: bllip_wsj_no_aux; size: 24516205; author: ; copyright: ; license: ;



مرور برخی مفاهیم

N-gram ❖

به بیان ساده n-gram اشاره به آیتم متوالی دارد (مثلا کلمه، زیرکلمه، کاراکتر) ❖
معمولًا کلمات مورد نظر است.

تک کلمه: ❖

دو کلمه: ❖

سه کلمه ای: ... و trigram ❖

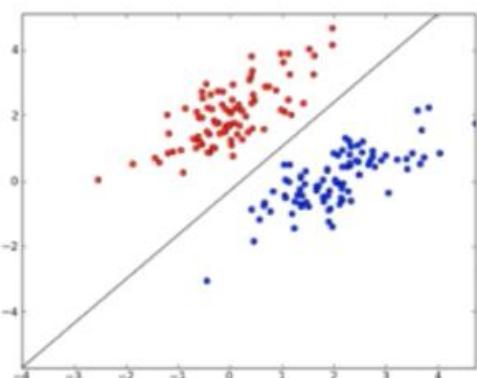
معروفترین استفاده از n-gram در مدلسازی زبانی مبتنی بر مدل‌های مارکوف است.

Text	N-gram
Data	1-gram
Great information	2-gram
I am fine	3-gram
Nice to meet you	4-gram

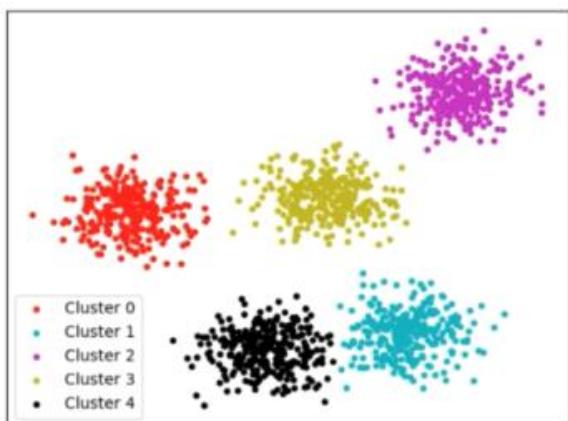
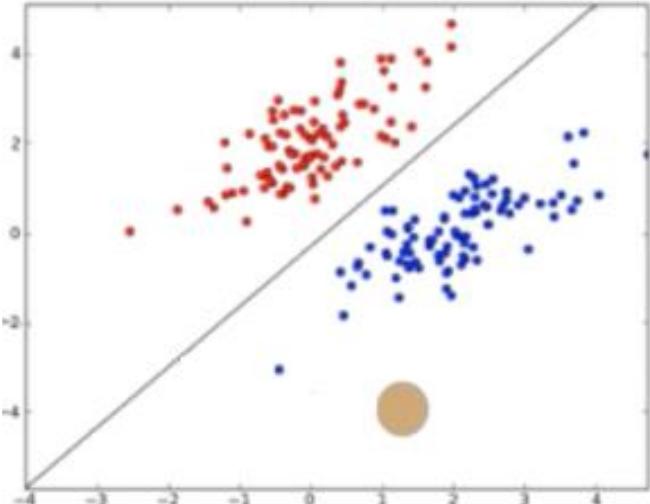
چرا بردارها مفیدند؟



- ❖ بردارها در نمایش عددی متون به ما کمک می کنند.
- ❖ الگوریتم های یادگیری ماشین با نمونه ها (samples) کار می کنند و نمونه ها نیز با بردارهای عددی توصیف می شوند.
- ❖ به عنوان مثال در مساله آشکارسازی ایمیل اسپم می توان تمام های ایمیل را به بردارهای عددی تبدیل کرد مانند شکل زیر:
- ❖ در اینجا مثلا هر ایمیل (نمونه آموزشی) توسط یک بردار دو بعدی مشخص شده است بطوریکه ایمیل های حاوی اسپم بصورت ابری از نقاط آبی رنگ و ایمیل های غیراسپم ابری از نقاط قرمز رنگ مشخص شده است.
- ❖ حال می توان با یک روش یادگیری ماشین این دو کلاس را براحتی از یکدیگر جدا کرد (خط مشخص شده بین دو کلاس)



چرا بردارها مفیدند؟



- ❖ حال فرض کنید یک ایمیل جدید آمده است.
- ❖ کافی است با همان روش قبل این ایمیل جدید را نیز به یک بردار دو بعدی تبدیل کنیم (دایره زرد رنگ)
- ❖ از آنجایی که بردار عددی متناظر با این ایمیل در پایین خط جداساز و ناحیه مربوط به ایمیل های اسپم افتاده است این یک ایمیل اسپم است.
- ❖ مثال دیگر: سازماندهی انبوهی از اسناد متنی
- ❖ خوشه های مختلف می توانند بیانگر اسناد مربوط به موضوعات مشابه باشند.
- ❖ از دیدگاه یادگیری ماشین به این بردارهای عددی، **بردارهای ویژگی** می گوییم.

Hamidreza Baradaran Kashani



تبدیل متن به بردارهای عددی

❖ دو گروه روش برای تبدیل متن به بردارهای عددی داریم:

❖ گروه اول: روش های آماری یا مبتنی بر فراوانی کلمات (Frequency- or Statistical-based)

❖ گروه دوم: روش های مبتنی بر پیشگویی کلمات (Prediction-based)



تبديل متن به بردارهای عددی

- ❖ گروه اول: روش‌های آماری یا مبتنی بر فراوانی کلمات (Frequency- or Statistical-based)
 - (OHE) one-hot encoding
 - روش کیسه لغات (BoW)
 - روش کیسه چند کلمه‌ای (Bag of n-grams)
 - روش فراوانی عبارت-معکوس فراوانی سند (TF-IDF)

- ❖ تمامی روش‌های این گروه بر اساس یک فرآیند دو مرحله‌ای عمل می‌کنند:
 - ❖ مرحله اول: تشکیل یک واژگان (Vocabulary) شامل V کلمه منحصر‌فرد استخراج شده از پیکره یادگیری
 - ❖ مرحله دوم: تولید بردارهای عددی متناظر با هر کلمه یا کل جمله (سند) بر اساس حضور کلمات در جملات



روش One-hot encoding

- در این روش، ابتدا به هر کلمه یک شناسه منحصر بفرد (ID) که مقدار صحیحی بین ۱ تا ۷ است داده می شود.
- سپس هر کلمه با یک بردار باینری ۷ بعدی مشخص می شود بطوریکه:
 - مولفه متناظر با شناسه کلمه ۱ است.
 - سایر مولفه ها ۰ هستند.

Document
<i>the cat sat</i>
<i>the cat sat in the hat</i>
<i>the cat with the hat</i>

مثال: پیکره ای شامل ۳ سند بصورت $\{the, cat, sat, in, hat, with\}$



روش One-hot encoding

- ❖ در این روش، ابتدا به هر کلمه یک شناسه منحصر بفرد (ID) که مقدار صحیحی بین ۱ تا ۷ است داده می شود.
- ❖ سپس هر کلمه با یک بردار باینری ۷ بعدی مشخص می شود بطوریکه:
 - ❖ مولفه متناظر با شناسه کلمه ۱ است.
 - ❖ سایر مولفه ها ۰ هستند.



❖ مثال: فرض کنید پیکره ای داریم مربوط به نظرات در ارتباط با یک فیلم سینمایی

- Review 1: *This movie is very scary and long*
- Review 2: *This movie is not scary and is slow*
- Review 3: *This movie is spooky and good*

Hamidreza Baradaran Kashani



روش One-hot encoding

❖ ابتدا بایستی کلمات منحصر بفرد را از پیکره استخراج کنیم:

- Review 1: *This movie is very scary and long*
- Review 2: *This movie is not scary and is slow*
- Review 3: *This movie is spooky and good*



‘1. This’, ‘2. movie’, ‘3. is’, ‘4. very’, ‘5. scary’, ‘6. and’, ‘7. long’, ‘8. not’, ‘9. slow’, ‘10. spooky’, ‘11. good’

❖ تعداد کل کلمات 11 است پس داریم: $V=11$

❖ فرض شود برای کلمه اول یعنی “this” شناسه 1، برای کلمه دوم یعنی “movie” شناسه 2 و در نهایت برای کلمه “good” شناسه 11 را در نظر می گیریم.



روش One-hot encoding

❖ حال با توجه به واژگان زیر:

'1. This', '2. movie', '3. is', '4. very', '5. scary', '6. and', '7. long', '8. not', '9. slow', '10. spooky', '11. good'

❖ بازنمایی OHE متناظر با جمله سوم "This movie is spooky and good" یعنی بصورت زیر است:

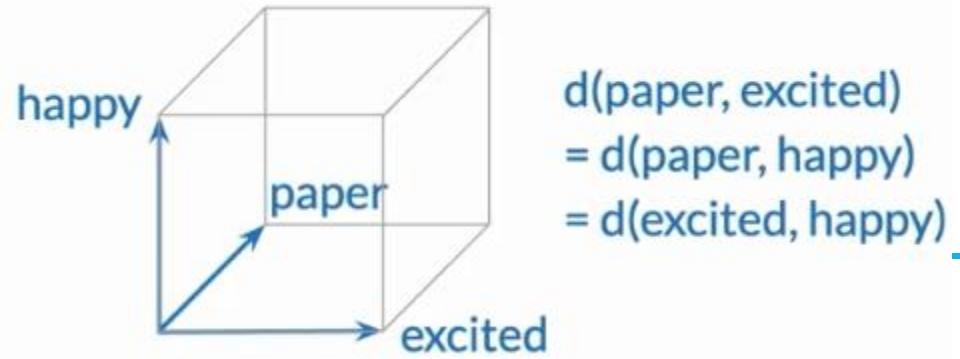
this:	[1 0 0 0 0 0 0 0 0 0 0]
movie:	[0 1 0 0 0 0 0 0 0 0 0]
is:	[0 0 1 0 0 0 0 0 0 0 0]
Spooky:	[0 0 0 0 0 0 0 0 1 0]
and:	[0 0 0 0 1 0 0 0 0 0]
good:	[0 0 0 0 0 0 0 0 0 1]



روش One-hot encoding

- ❖ مزیت روش OHE:
 - ❖ درک و پیاده سازی آن بسیار ساده است.
- ❖ معایب روش OHE:
 - ❖ طول بردار مناسب با اندازه واژگان است. لذا برای یک واژگان واقعی با اندازه $1M$ (یک میلیون) کلمه، بردار بازنمایی متناظر با هر کلمه شامل $1M$ مولفه است!
 - ❖ برای جملات با تعداد کلمات مختلف، بازنمایی های متñی با اندازه مختلف ایجاد می شود. مثلا برای Review2 پیکره قبل که ۸ کلمه دارد بازنمایی بصورت یک ماتریس با ابعاد 8×11 است، در حالیکه برای Review3 پیکره قبل با ۶ کلمه بازنمایی بصورت یک ماتریس با ابعاد 6×11 است. در حالیکه مثلا برای یک مساله طبقه بندی متن باقیستی تمامی متون به یک بردار با طول یکسان تبدیل شوند.

روش One-hot encoding



معایب روش OHE

❖ بردار تمامی کلمات در فضای تعبیه کلمات (word embedding space) بر هم دیگر متعامد است (ضرب درونی هر دو بردار صفر است). بنابراین عملاً معنای کلمات در نظر گرفته نمی شود. مثلاً در مثال قبلی بردارهای متناظر با کلمات "scary" به معنای ترسناک و "spooky" به معنای شبج وار بایستی به یکدیگر نزدیکتر باشند تا بردار متناظر با "scary" و بردار متناظر با "good" در حالیکه این گونه نیست.

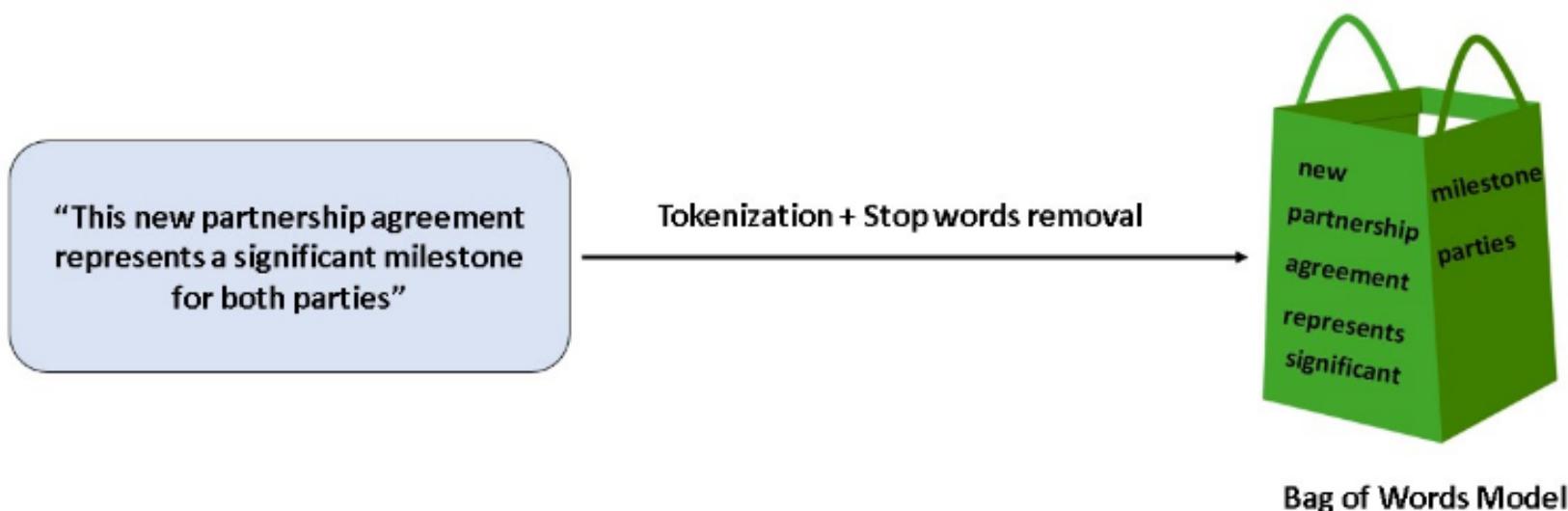
❖ به تمام کلمات پیکره اهمیت یکسانی داده می شود. مثلاً با توجه به مثال قبل، برای یک کاربرد آنالیز احساسات اهمیت کلمه Movie که در تمام نظرات استفاده شده باید کمتر باشد تا کلماتی مثل good, slow, ... و spooky که عملاً قطبیت نظر و کلاس نظر را مشخص می کنند.

❖ حل نکردن مساله کلمات خارج از واژگان (Out-of-Vocabulary) یعنی کلماتی که خارج از پیکره یادگیری هستند و لذا در مجموعه واژگان ظاهر نمی شوند.



روش کیسه لغات

- ❖ مبنای این روش:
- ❖ بازنمایی متن بر اساس کیسه‌ای از لغات موجود در متن
- ❖ عدم توجه به ترتیب کلمات و گرامر



Hamidreza Baradaran Kashani

روش کیسه لغات



❖ ۲ روشن استخراج بازنمایی متن مبتنی بر BoW

❖ **رویکرد اول: استخراج بردار ویژگی باینری (شامل مقادیر صفر یا یک)** برای هر سند (یا جمله) بر اساس حضور یا عدم حضور کلمات در آن سند (یا جمله)

- اگر کلمه در متن رخ داده باشد در مولفه متناظر با آن کلمه مقدار ۱ و در غیر اینصورت مقدار صفر قرار داده می شود.

❖ **رویکرد دوم: استخراج بردار ویژگی (شامل مقادیر صحیح غیرمنفی)** برای هر سند (یا جمله) بر اساس میزان فراوانی رخداد کلمات در آن سند (یا جمله)

- اگر کلمه در متن رخ داده باشد در مولفه متناظر با آن کلمه تعداد رخداد آن کلمه در نظر گرفته می شود.

❖ در هر دو رویکرد

❖ اندازه بردار ویژگی استخراجی برای هر سند (یا جمله) V است.

❖ V اندازه واژگان یا همان تعداد کلمات منحصر بفرد پیکره یادگیری است.

روش کیسه لغات



مثال: پیکره ای شامل نظرات مربوط به غذای پاستا ایتالیایی

- Review 1 : This pasta is very tasty and affordable.
- Review 2: This pasta is not tasty and is affordable.
- Review 3 : This pasta is delicious and cheap.
- Review 4: Pasta is tasty and pasta tastes good.

ساخت واژگان



'1. This', '2. pasta', '3. is', '4. very', '5. tasty', '6. and', '7. affordable', '8. not',
'9. delicious', '10. cheap', '11. tastes', '12. good'

استخراج بردار ویژگی
کیسه لغات دودویی



استخراج بردار ویژگی کیسه
لغات مبتنی بر فراوانی

Review 4: [0 **1** 1 0 1 1 0 0 0 1 1]

Review 4: [0 **2** 1 0 1 1 0 0 0 1 1]

Hamidreza Baradaran Kashani

روش کیسه لغات



❖ مزیت روش کیسه لغات

❖ درک و پیاده سازی بسیار ساده.

❖ ارائه یک بردار با طول ثابت برای هر سند یا جمله (برخلاف روش OHE)

❖ معایب روش کیسه لغات:

❖ طول بردار برابر با اندازه واژگان

○ برای یک واژگان واقعی با اندازه 1M (یک میلیون) کلمه، بردار بازنمایی متناظر با هر کلمه شامل 1M مولفه است!

❖ عدم توجه به ترتیب کلمات

○ بازنمایی یکسان برای جملاتی با کلمات یکسان اما با معنای متفاوت

○ بازنمایی متفاوت برای جملاتی با کلمات متفاوت اما معنای مشابه

❖ حل نکردن مساله OOV

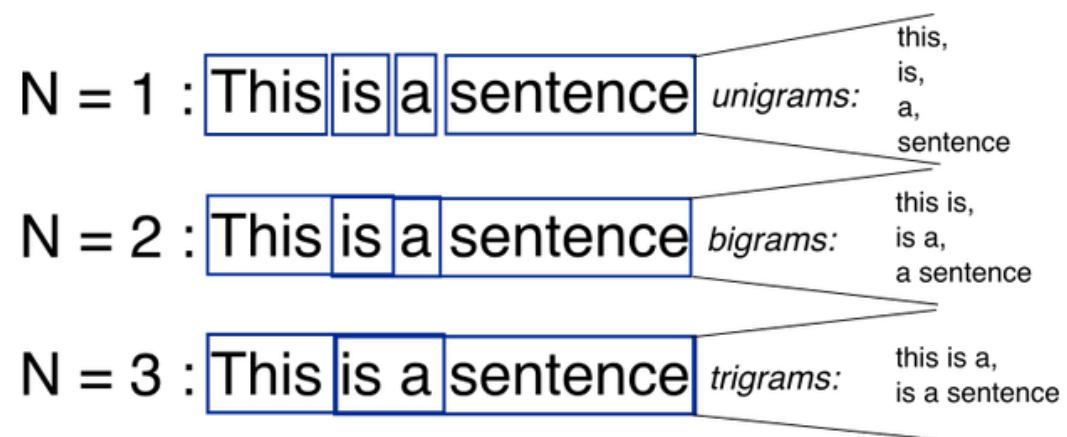
Hamidreza Baradaran Kashani

Jack is clever than Sarah
Sarah is clever than Jack



روش کیسه چند کلمه ای (BoN)

- ❖ روش های قبلی کلمات را بصورت واحدهای جداگانه در نظر می گیرند. در واقع ترتیب کلمات (word order) یا زمینه متن (context) را در نظر نمی گیرند.
- ❖ یک روش برای در نظر گرفتن ترتیب کلمات یا زمینه متن (تا حدی)، روش کیسه چند کلمه ای یا Bag-of-n-grams است. روش BoW همان BoN grams است وقتی $N=1$.
- ❖ مثلاً می توان دو کلمه ای های متوالی (trigram)، سه کلمه ای های متوالی (bigram) و ... را در نظر گرفت.





روش کیسه چند کلمه‌ای (BoN)

مثال

ساخت واژگان مبتنی بر بایگرم

D1: the dog sat
D2: the cat sat in the hat
D3 : the cat with the hat



1. 'the dog' 2. dog sat' 3. 'the cat'
4. 'cat sat' 5. 'sat in' 6. 'in the'
7. 'the hat' 8. 'cat with' 9. 'with the'

استخراج بردار ویژگی مبتنی بر
 $N=2$ با BoN

The cat and dog sat in the hat



[0 1 1 0 1 1 1 0 0]

تمرین: برای همین جمله بردار ویژگی مبتنی بر BoN با $N=3$ را بدست آورید.

Hamidreza Baradaran Kashani

روش کیسه لغات



- ❖ مزیت روش BoN
 - ❖ در نظر گرفتن اطلاعات مربوط به ترتیب کلمات و زمینه متن (تا حدی)
 - ❖ ارائه یک بردار با طول ثابت برای هر سند یا جمله
 - ❖ نزدیک تر بودن بردارهای ویژگی برای اسنادی با n-gram های متفاوت

معایب روش BoN

- ❖ افزایش احتمالی بعد بردارهای ویژگی با افزایش N در n-gram و در نتیجه افزایش پیچیدگی محاسباتی
- ❖ حل نکردن مساله OOV

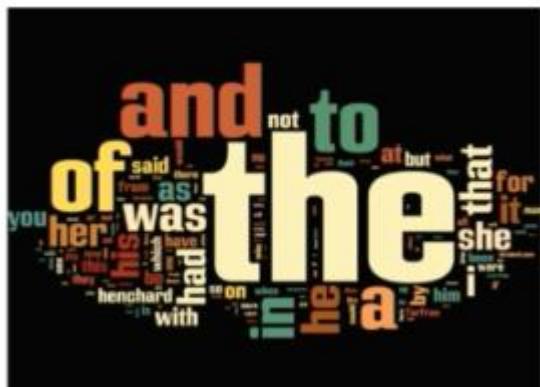


روش فراوانی کلمه-معکوس فراوانی سند (TF-IDF)

- ❖ روش TD-IDF یک روش رایج در بازیابی اسناد و دسته بندی متون است.
- ❖ چرا به TF-IDF نیاز داریم؟
- ❖ این روش بردارسازی متن چگونه کار می کند؟
- ❖ وجه تمايز آن با روش های دیگر چیست؟



روش فراوانی کلمه-معکوس فراوانی سند (TF-IDF)



❖ یادآوری از کلمات توقف (Stop Words)

- ❖ کلمات توقف کلماتی بودند که در اکثر متون و اسناد رخ می دادند.
- ❖ ما می خواهیم این کلمات را در اکثر موارد و کاربردها نادیده بگیریم.
- ❖ مثلا در یک موتور جستجو، یک مساله دسته بندی ایمیل های اسپم از غیر آن و یا یک مساله آنالیز احساسات
- ❖ چرا که احتمالا هر سندی شامل این کلمات است.



روش فراوانی کلمه-معکوس فراوانی سند (TF-IDF)

❖ یادآوری از کلمات توقف (Stop Words)

- ❖ در روش بردارسازی متن مبتنی بر فراوانی کلمات مثل BoW، از آنجایی که کلمات توقف فراوانی بالایی دارند، در بردار ویژگی سند برای مولفه متناظر با این کلمات مقدار بزرگی لحاظ می شود و این مقادیر بزرگ در محاسبه شباهت یا فاصله اسناد از یکدیگر تاثیر گذاری بیشتری دارند.
- ❖ کلمات توقف لیست مشخصی ندارند و وابسته به کاربرد خاصی هستند.
- ❖ مثلا در یک سامانه توصیه گر محصولات کلی و مختلف ممکن است کلمه فیلم، کتاب، موبایل و ... از اهمیت بالایی برخوردار باشند.
- ❖ اما اگر بطور خاص یک سامانه توصیه گر فیلم داریم که در همه نظرات کلمه "فیلم" وجود دارد آنگاه عملا این کلمه به عنوان یک Stop word باید در نظر گرفته شود.



روش فراوانی کلمه-معکوس فراوانی سند (TF-IDF)

❖ سوال؟

- ❖ آیا ما باید برای هر کاربرد لیست این کلمات توقف را بصورت دستی مشخص کنیم؟
- ❖ آیا می توانیم بصورت خودکار تعیین کنیم که کدام کلمات مهم بوده و کدام ها را باید نادیده بگیریم.

❖ ایده اصلی در TF-IDF

- ❖ می خواهیم کلماتی که در اسناد زیادی به کار می روند را نادیده بگیریم.
- ❖ این کلمات تمایزی بین اسناد مختلف ایجاد نمی کنند.
- ❖ می خواهیم اثر این کلمات در بردار ویژگی کم رنگ شود.



روش فراوانی کلمه-معکوس فراوانی سند (TF-IDF)

❖ TF-IDF: Term Frequency – Inverse Document Frequency

$$TF - IDF \approx \frac{\text{Term Frequency}}{\text{Document Frequency}}$$

- رابطه فوق رابطه اصلی TF-IDF نیست تنها برای درک بیشتر به این شکل بیان شده است.
- ایده اصلی آن است که اگر کلمه w دفعات زیادی در جمله $S1$ رخ دهد، اما در بقیه جملات پیکره رخداد زیادی نداشته باشد:
- انگاه کلمه w بایستی برای $S1$ از اهمیت بالا (مقدار بزرگ در بردار ویژگی) و برای بقیه جملات از اهمیت کم (مقدار کوچک در بردار ویژگی) برخوردار باشد.



روش فراوانی کلمه-معکوس فراوانی سند (TF-IDF)

$$tfidf(t, d) = tf(t, d) \times idf(t)$$

- در رابطه اصلی TF-IDF از ضرب دو مقدار TF و IDF استفاده می شود.
- در عبارت TF ما دو آرگومان داریم: کلمه t و سند d
- به عبارتی $TF(t,d)$ یعنی تعداد رخداد کلمه t در سند d
- در عبارت IDF ما یک آرگومان داریم: کلمه t
- در واقع عبارت IDF مربوط به یک سند خاص نیست، بلکه مربوط به یک کلمه خاص است.
- IDF با جمع کردن تعداد اسناد شامل یک کلمه خاص t محاسبه می شود، مثلاً کلمه "جذاب" در ۲۰ سند از ۱۰۰ سند پیکره رخداده است.

فراوانی کلمه TF



$$tfidf(t, d) = \text{tf}(t, d) \times idf(t)$$

- عبارت TF یک ماتریس است.
- فرض شود برای یک پیکره یادگیری V کلمه منحصرفرد و N سند داریم.
- بعد این ماتریس برابر با N^*V یا V^*N است.
- در مازول CountVectorizer در تولباکس Sikit Learn بعد این ماتریس بصورت N^*V نتیجه می شود.



معکوس فراوانی سند IDF

$$tfidf(t, d) = tf(t, d) \times idf(t)$$

$$idf(t) = \log \frac{N}{N(t)}$$

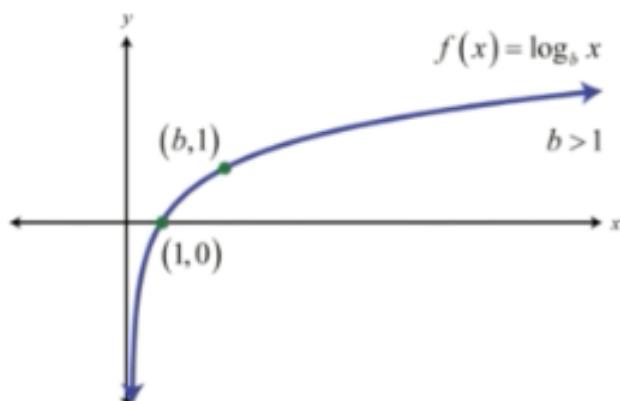
- $N(t)$ تعداد اسناد شامل کلمه t است.
- N تعداد کل اسناد در پیکره یادگیری است.



چرا \log در محاسبه IDF؟

$$idf(t) = \log \frac{N}{N(t)}$$

- یک تابع یکنواخت (monotonic) است، اگر مقدار $N/N(t)$ افزایش یابد، مقدار \log آن نیز افزایش پیدا می کند.
- بنابراین وقتی کلمه در اسناد زیادی استفاده شود مقدار IDF کم می شود و TF-IDF هم کم می شود.



- تابع \log مقدار ورودی اش را به نوعی فشرده می کند.
- فرض شود $N=10^6$ و $N(t)=1$
- یعنی کلمه t تنها در یک سند از یک میلیون سند رخ داده است
- اگر \log نگیریم مقدار TF در 10^6 (یک میلیون) ضرب می شود!!!
- در حالیکه با گرفتن \log مقدار IDF برابر است با 13.8 (مقدار بسیار معقولانه تری در بردارهای ویژگی و یادگیری ماشین)



فرمول های مختلف برای محاسبه TF

باينري

- مقدار 1 اگر کلمه t در سند d رخدید و صفر در غیر اینصورت
- نرمالیزه کردن تعداد (گاهی اوقات به عنوان پیش فرض استفاده می شود)

$$tf(t, d) = \frac{count(t, d)}{\sum_{t' \in terms(d)} count(t', d)}$$

گرفتن log

$$tf(t, d) = \log(1 + count(t, d))$$



فرمول های مختلف برای محاسبه IDF

- Smooth IDF

$$idf(t) = \log \frac{N}{N(t)+1} + 1$$

○ اگر کلمه ای در تمام اسناد رخ دهد ($N(t)=N$), رابطه قبلی مقدار نتیجه می دهد (چون $\log 1=0$)

- IDF Max

$$idf(t) = \log \frac{\max_{t' \in terms(d)} N(t')}{N(t)}$$

○ بجای استفاده از N در صورت کسر، حداکثر تعداد تکرار یک کلمه در همان سند (که tf را محاسبه می کنیم) استفاده می شود. بنابراین آرگومان \log به نسبت سند جاری است نه کل اسناد پیکره

- Probabilistic IDF

$$idf(t) = \log \frac{N - N(t)}{N(t)}$$

Hamidreza Baradaran Kashani

نرمالیزه کردن TF-IDF



- ارتباط نزدیک بین فاصله اقلیدسی با فاصله کسینوسی
- در صورت مرتب کردن (sorting/ranking) مقادیر، نتیجه یکسانی با هر دو فاصله اقلیدسی و کسینوسی حاصل میشود.
- برخلاف **TfidfVectorizer**، در **CountVectorizer** این نوع نرمالسازی وجود دارد.
- کافی است آرگومان **norm** را **l1** یا **l2** بگذاریم (پیش فرض این آرگومان **l2** است)
- **TfidfVectorizer(norm="l2")**

$$\hat{tfidf}(t, d) = \frac{tfidf(t, d)}{\|tfidf(\cdot, d)\|_p} \quad \text{where } p = 1 \text{ or } 2$$



- Sentence A = *The Car is Driven on the Road*
- Sentence B = *The Truck is Driven on the highway*

Word	TF		IDF	TF*IDF	
	A	B		A	B
The	1/7	1/7	$\log(2/2) = 0$	0	0
Car	1/7	0	$\log(2/1) = 0.3$	0.043	0
Truck	0	1/7	$\log(2/1) = 0.3$	0	0.043
Is	1/7	1/7	$\log(2/2) = 0$	0	0
Driven	1/7	1/7	$\log(2/2) = 0$	0	0
On	1/7	1/7	$\log(2/2) = 0$	0	0
The	1/7	1/7	$\log(2/2) = 0$	0	0
Road	1/7	0	$\log(2/1) = 0.3$	0.043	0
Highway	0	1/7	$\log(2/1) = 0.3$	0	0.043

روش TF-IDF



- ❖ مزیت روش
- ❖ در نظر گرفتن اطلاعات مربوط به ترتیب کلمات و زمینه متن (تا حدی)
- ❖ ارائه یک بردار با طول ثابت برای هر سند یا جمله
- ❖ تا حدی می تواند معنای جمله را در نظر بگیرد.

معایب روش:

- ❖ افزایش بعد بردارهای ویژگی با افزایش تعداد کلمات در دیکشنری و در نتیجه افزایش پیچیدگی محاسباتی
- ❖ حل نکردن مساله OOV

روش TF-IDF



- ❖ مزیت روش
- ❖ در نظر گرفتن اطلاعات مربوط به ترتیب کلمات و زمینه متن (تا حدی)
- ❖ ارائه یک بردار با طول ثابت برای هر سند یا جمله
- ❖ تا حدی می تواند معنای جمله را در نظر بگیرد.

معایب روش:

- ❖ افزایش بعد بردارهای ویژگی با افزایش تعداد کلمات در دیکشنری و در نتیجه افزایش پیچیدگی محاسباتی
- ❖ حل نکردن مساله OOV



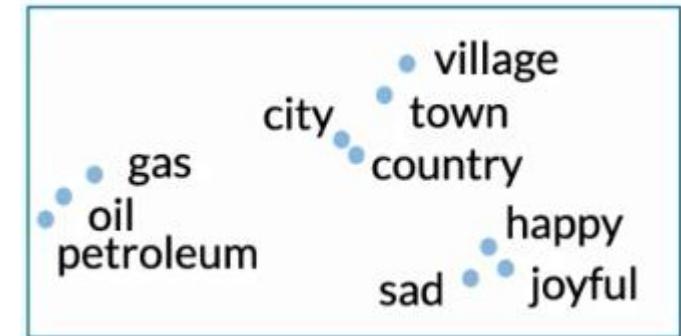
کاربردهای مختلف تعبیه کلمات



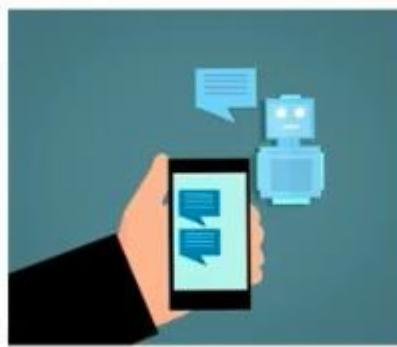
دسته بندی نظرات کاربران



آنالیز احساسات



شباهت معنایی



پاسخگویی سوالات



ترجمه ماشینی

Hamidreza Baradaran Kashani



روش های اولیه تعبیه کلمات

Word	Number
a	1
able	2
about	3
...	...
hand	615
...	...
happy	621
...	...
zebra	1000

"a"	
1	a
0	able
0	about
:	...
0	hand
:	...
0	happy
:	...
0	zebra

"happy"	
0	a
0	able
0	about
:	...
0	hand
:	...
1	happy
:	...
0	zebra

↑ 1000 rows ↓

"zebra"	
0	a
0	able
0	about
:	...
0	hand
:	...
0	happy
:	...
1	zebra

Hamidreza Baradaran Kashani



روش های اولیه تعبیه کلمات

ارتباط دوطرفه بین شناسه هر کلمه با بردار One-Hot متناظر با آن

Word	Number		"happy"	
a	1		1 0	a
able	2		2 0	able
about	3		3 0	about
...
hand	615		615 0	hand
...
happy	621	↔	621 1	happy
...
zebra	1000		1000 0	zebra

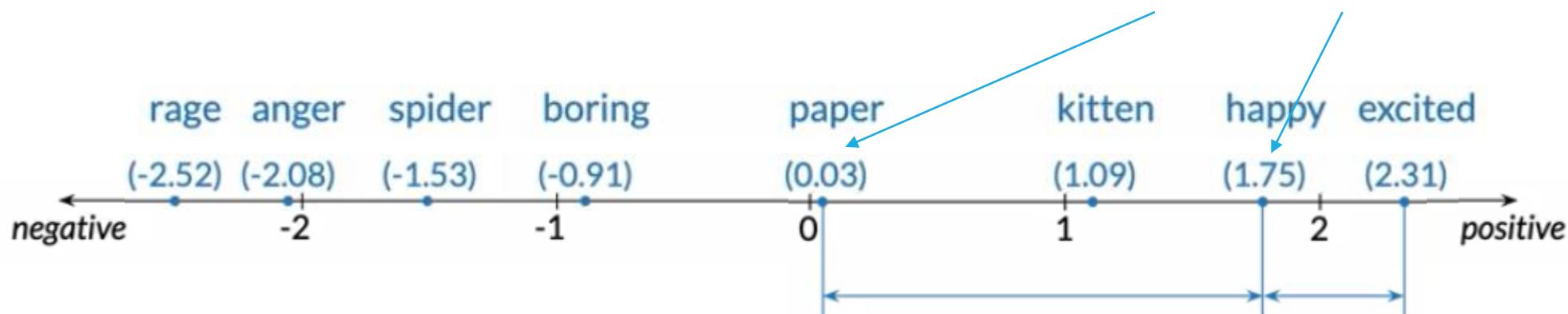
Hamidreza Baradaran Kashani

سوال مهم

چگونه می توان بردارهایی برای کلمات ایجاد کرد که حاوی معنای کلمات باشند؟

Hamidreza Baradaran Kashani

تعبیه کلمات



توصیف هر کلمه با یک مدار اسکالار
(بیانگر موقعیت کلمه بر روی محور)

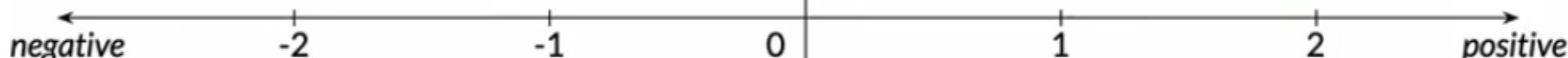
مزیت:
دو کلمه **excited** و **happy** به هم شبیه تر
هستند تا **paper** و **happy**

فرضا هر چه کلمه معنای مثبت تری داشته باشد بر روی این محور مقدار مثبت تر دارد و هر چه کلمه معنای منفی تری داشته باشد بر روی این محور مقدار منفی تر دارد.



توسعه توصیف کلمات در فضایی با بعد بالاتر

دو کلمه یک بردار
دارند؟!

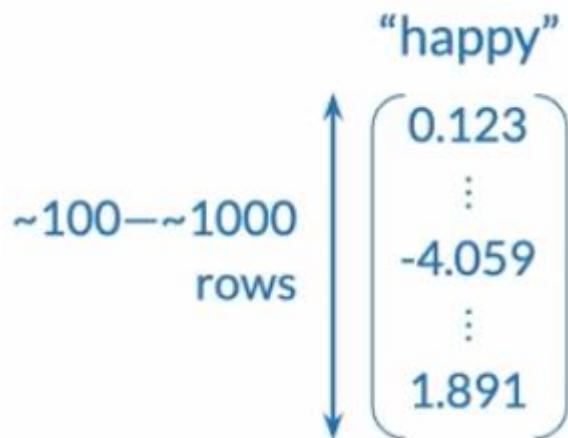


بر روی محور عمودی:
کلمات که مفهوم فیزیکی دارند
مقدار بزرگتر و هرچه مفهوم
انتزاعی تر مقدار کوچک تر

توصیف هر کلمه با یک
بردار ۲ بعدی



سوال مهم



۱) بعد کم (۱۰۰ تا ۱۰۰۰ بعد مثل ۳۰۰)

۲) داشتن معنا در بردار تعییه کلمه

* شبهه (فاسله) معنایی

شبهه بسیار زیاد درخت با جنگل در مقایسه با درخت با کتاب

* قراین یا تناسب بین کلمات

تهران با ایران مثل رم با ایتالیا



فرآیند تولید تعبیه کلمات

-
- ❖ جهت تولید بردارهای تعبیه کلمات به دو مورد نیاز داریم:
 - ❖ پیکره متنی
 - ❖ رویکرد تعبیه کلمات



فرآیند تولید تعبیه کلمات

❖ پیکره متنی

- حتما بایستی شامل متن کامل باشد به عبارتی **کلمات در یک زمینه (context)** مشخص (همراه با **کلمات مجاورشان**) در نظر گرفته شود، نه کلمات کلیدی یا لیست کلمات یک پیکره ...
- زمینه مهم است چرا؟
- معنای یک کلمه بر اساس زمینه ای که کلمه در آن قرار گرفته مشخص می شود.

❖ پیکره متنی

- می تواند با هدف کاربرد عمومی باشد (داده های ویکی پدیا)
- یا کاربردها در حوزه خاص مثل اورزش، سیاست یا ...



فرآیند تولید تعبیه کلمات

❖ رویکرد تعبیه

- هدف: استخراج بردارهای تعبیه کلمات از پیکره متنی
- رویکرد: **یادگیری ماشین خود نظارتی (Self-supervised)**
- ❖ ترکیب رویکردهای بانظارت (Supervised) و بدون نظارت (Unsupervised)
- ❖ شباخت به بدون نظارت؟ داده اصلا برچسب ندارد
- ❖ شباخت به بانظارت؟ بخشی از داده به عنوان برچسب یا ناظر بخش دیگر محسوب می شود
- مثلا کلمات اطراف یک کلمه در متن به عنوان برچسب برای کلمه مرکزی در نظر گرفته می شوند یا بالعکس.



رویکردهای تعبیه کلمات پایه

(Google, 2013) Word2vec

استفاده از یک شبکه عصبی کم عمق (Shallow network)

۲ تا معماری مدل (شبکه عصبی) رایج:

(CBOW) Continuous bag-of-words

پیش بینی کلمه ماسک شده با توجه به کلمات اطراف آن

(SGNS) Skip-gram with negative sampling یا Continuous Skip-gram

پیش بینی کلمات اطراف یک کلمه با در نظر گفتن خود کلمه مرکزی به عنوان ورودی



رویکردهای تعبیه کلمات پایه

❖ روش (Stanford, 2014) Glove یا Global vectors

❖ تجزیه (factorization) لگاریتم ماتریس کلمه-سندها

❖ روش (Facebook, 2016) FastText

❖ مبتنی بر مدل Skip-gram

❖ در نظر گرفتن ساختار کلمه بصورت یک n-gram از کاراکترها

❖ توانایی استخراج تعبیه کلمات برای کلمات خارج از واژگان (OOV)

- مثلاً امکان استخراج بردار تعبیه مشابه برای کلمه kitty (دیده نشده در مرحله یادگیری) و kitten (دیده شده در مرحله یادگیری) - دنباله کاراکترهای مشابه در دو کلمه امکان متوسط گیری از بردارهای کلمات برای بدست آوردن بردار تعبیه بیانگر جمله یا عبارت



رویکردهای تعبیه کلمات جدیدتر

❖ استفاده از معماری های یادگیری عمیق (Deep learning)

○ بهبود بردارهای تعبیه کلمات معنادارتر با استفاده از زمینه آنها

○ یک کلمه می تواند بردارهای تعبیه متمایزی نسبت به زمینه خود داشته باشد (در نظر گرفتن polysemy)

○ روش (Google, 2018) BERT

○ مبتنی بر بخش رمزگذار (transformer) معماری مبدل (encoder)

○ روش (Allen Institute for AI, 2018) ELMO

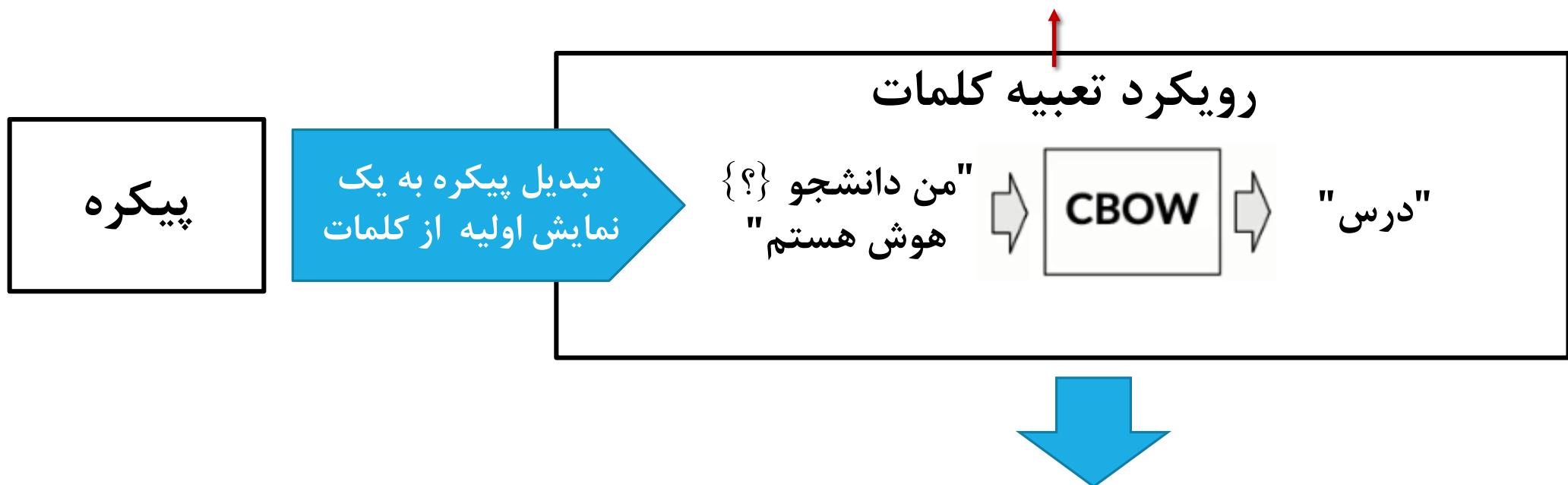
○ روش (OpenAI, 2018) GPT-2

Hamidreza Baradaran Kashani

روش CBOW



یک رویکرد یادگیری
ماشین است

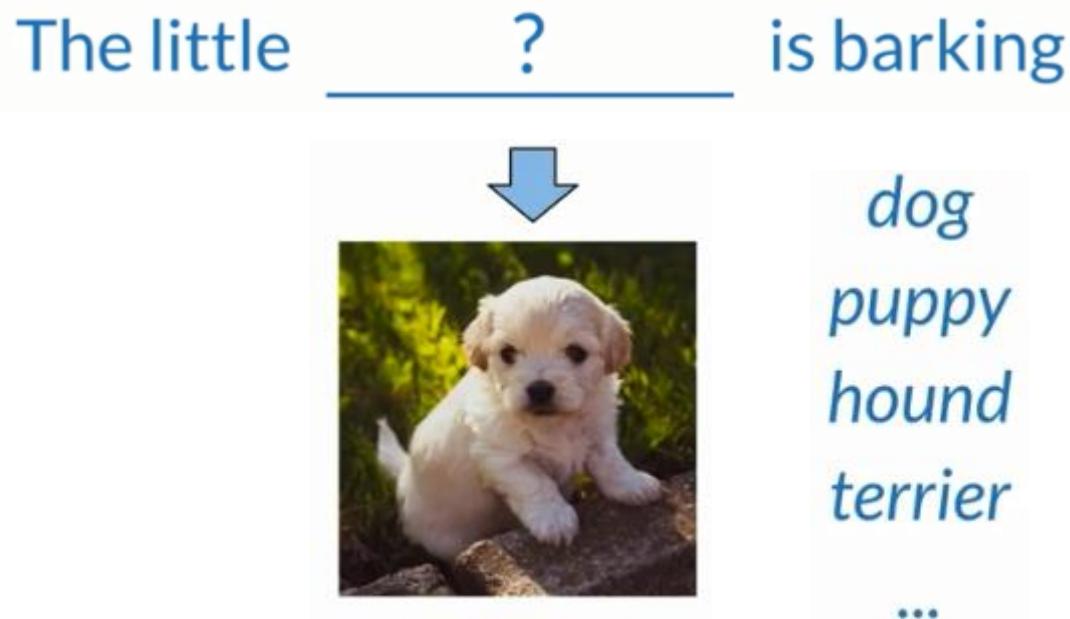


مجموعه ای از بردارهای تعبیه کلمات



ایده اصلی روش CBOW

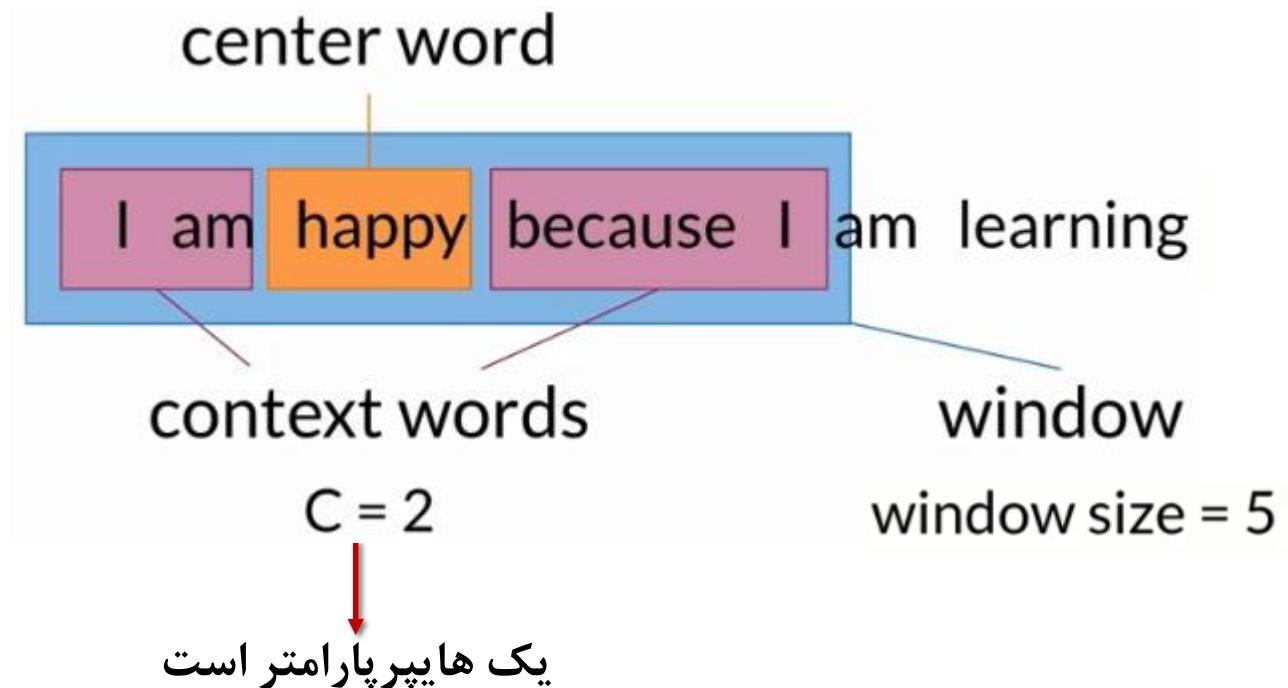
❖ اگر دو کلمه مختلف در اکثر متنون و پیکره ها توسط مجموعه مشابهی از کلمات احاطه شوند، آن دو کلمه احتمالا از لحاظ معنایی به یکدیگر ارتباط نزدیکی دارند.



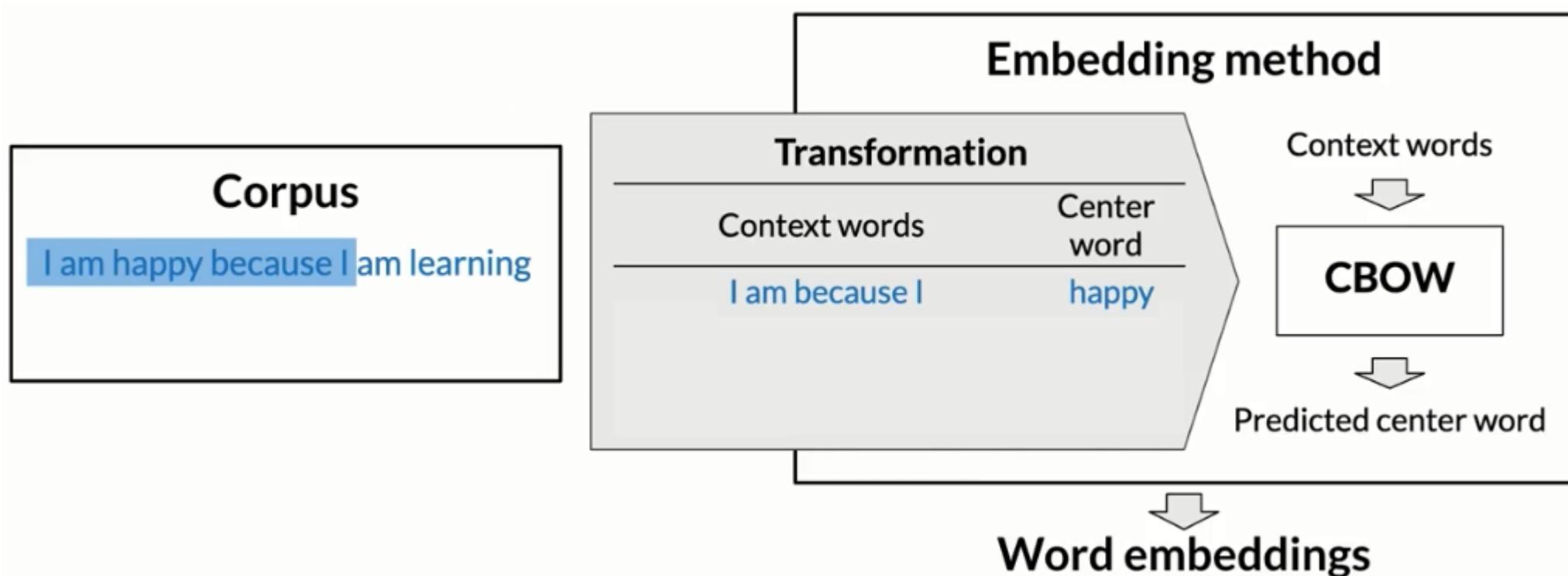


ایجاد یک نمونه یادگیری

چگونه از پیکره موجود، داده یادگیری برای مدل تعبیه کلمات را آماده کنیم؟ ♦



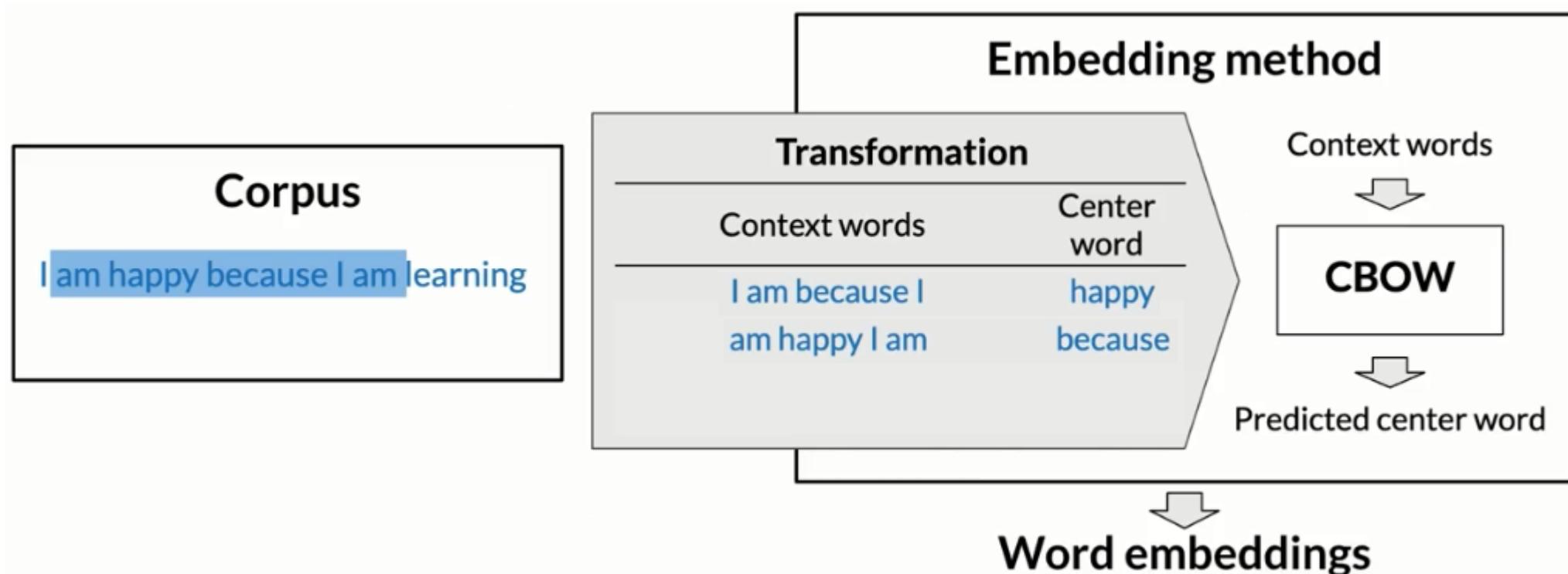
ایجاد نمونه های یادگیری



Hamidreza Baradaran Kashani

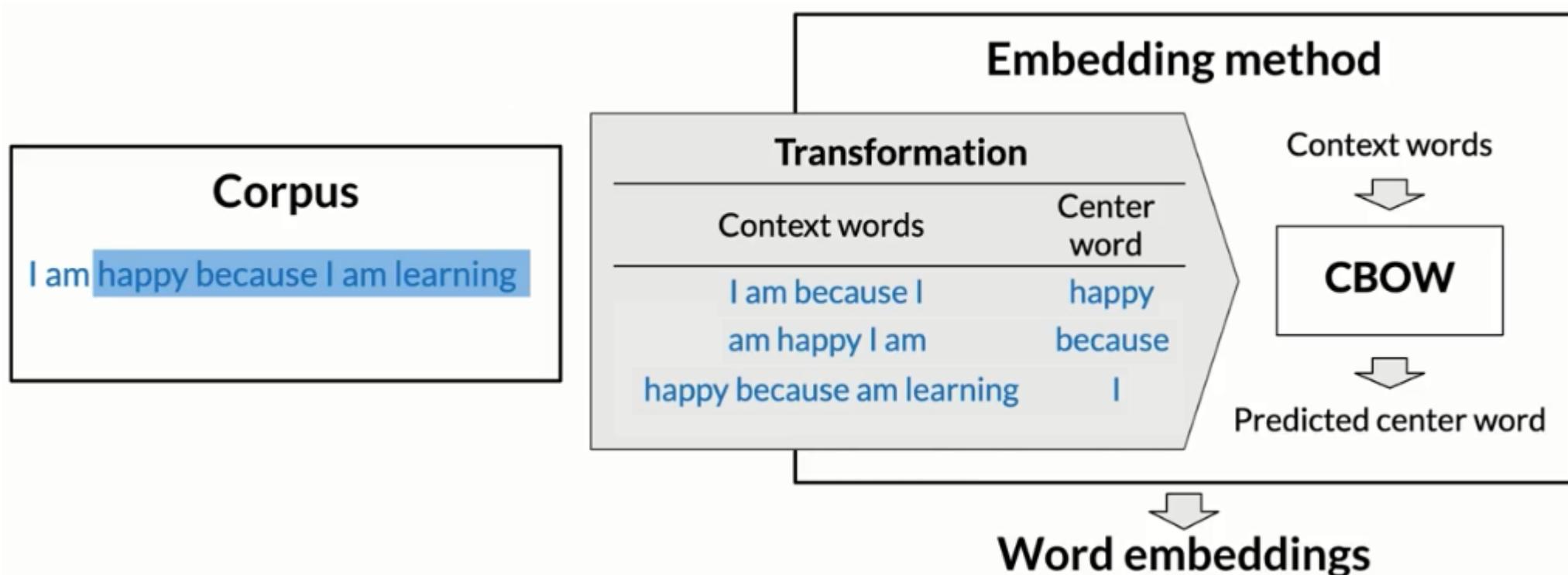


ایجاد نمونه های یادگیری



Hamidreza Baradaran Kashani

ایجاد نمونه های یادگیری

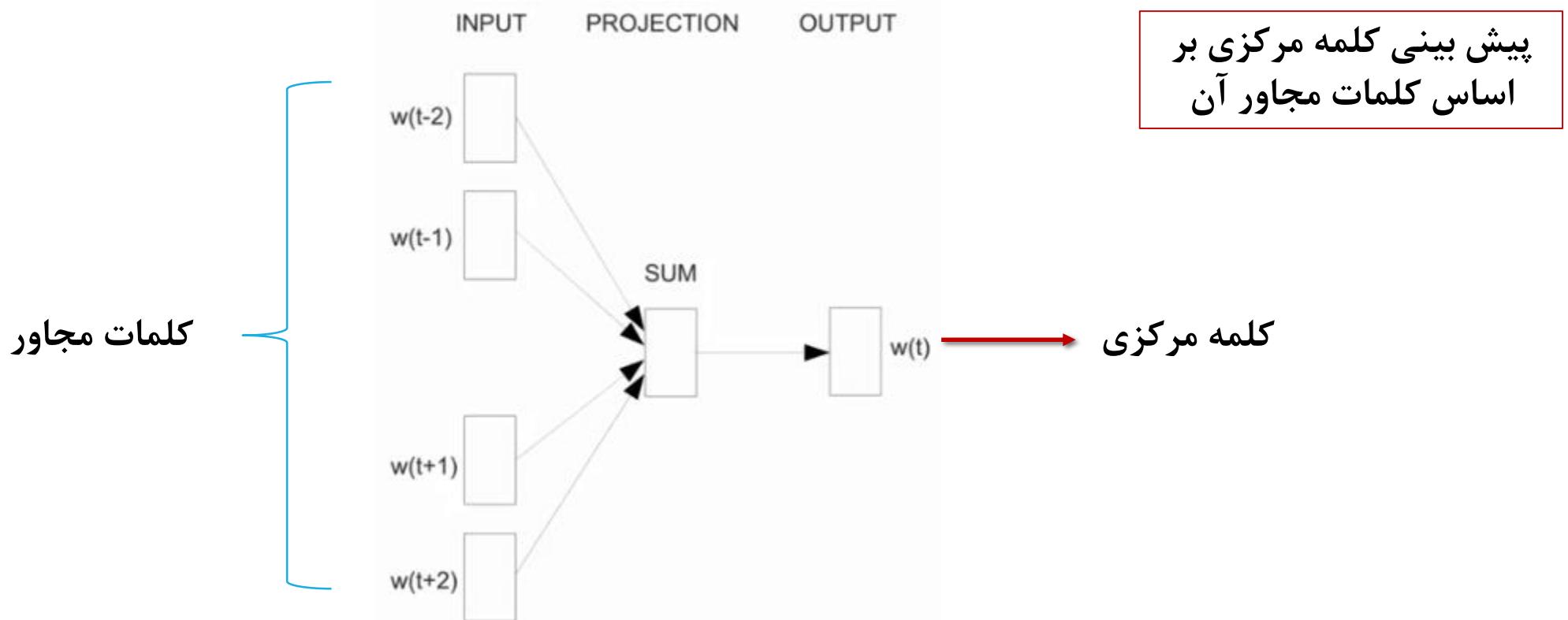


Hamidreza Baradaran Kashani



رویکرد CBOW بطور خیلی خلاصه!

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.



Hamidreza Baradaran Kashani



تبدیل کلمات به بردارها

با هدف یادگیری مدل تعبیه کلمات CBOW، چگونه کلمات مجاور و کلمه مرکزی به بردار تبدیل شوند؟



تبدیل کلمات مرکزی به بردارها

I am happy because I am learning

❖ فرضاً پیکره مقابله داریم:

❖ گام ۱) استخراج واژگان (Vocabulary) شامل کلمات منحصر بفرد در پیکره

am, because, happy, I, learning

❖ گام ۲) تبدیل هر کلمه به یک بردار one-hot

	am	because	happy	I	learning
am	1	0	0	0	0
because	0	1	0	0	0
happy	0	0	1	0	0
I	0	0	0	1	0
learning	0	0	0	0	1

Hamidreza Baradaran Kashani



تبدیل کلمات مجاور به بردارها

❖ تولید یک بردار برای کل کلمات مجاور با متوسط گیری از بردارهای one-hot هر کلمه

$$\left[\begin{array}{c} I \\ am \\ because \\ happy \\ I \\ learning \end{array} \right] = \frac{\left(\begin{array}{c} I \\ am \\ because \\ happy \\ I \\ learning \end{array} \right) + \left(\begin{array}{c} I \\ am \\ because \\ happy \\ I \\ learning \end{array} \right) + \left(\begin{array}{c} I \\ am \\ because \\ happy \\ I \\ learning \end{array} \right) + \left(\begin{array}{c} I \\ am \\ because \\ happy \\ I \\ learning \end{array} \right)}{4}$$



تبدیل کلمات به بردارها

Context words	Context words vector	Center word	Center word vector
<i>I am because I</i>	[0.25; 0.25; 0; 0.5; 0]	<i>happy</i>	[0; 0; 1; 0; 0]
<i>am happy I am</i>	[0.5; 0; 0.25; 0.25; 0]	<i>because</i>	[0; 1; 0; 0; 0]
<i>happy because am learning</i>	[0.25; 0.25; 0.25; 0; 0.25]	<i>I</i>	[0; 0; 0; 1; 0]



تمامی کلمات مجاور و مرکزی (درون پیکره یادگیری) تبدیل به بردارهای عددی اولیه شده اند و اکنون با این بردارها می توان یادگیری مدل تعبیه کلمات CBOW را آغاز کرد.

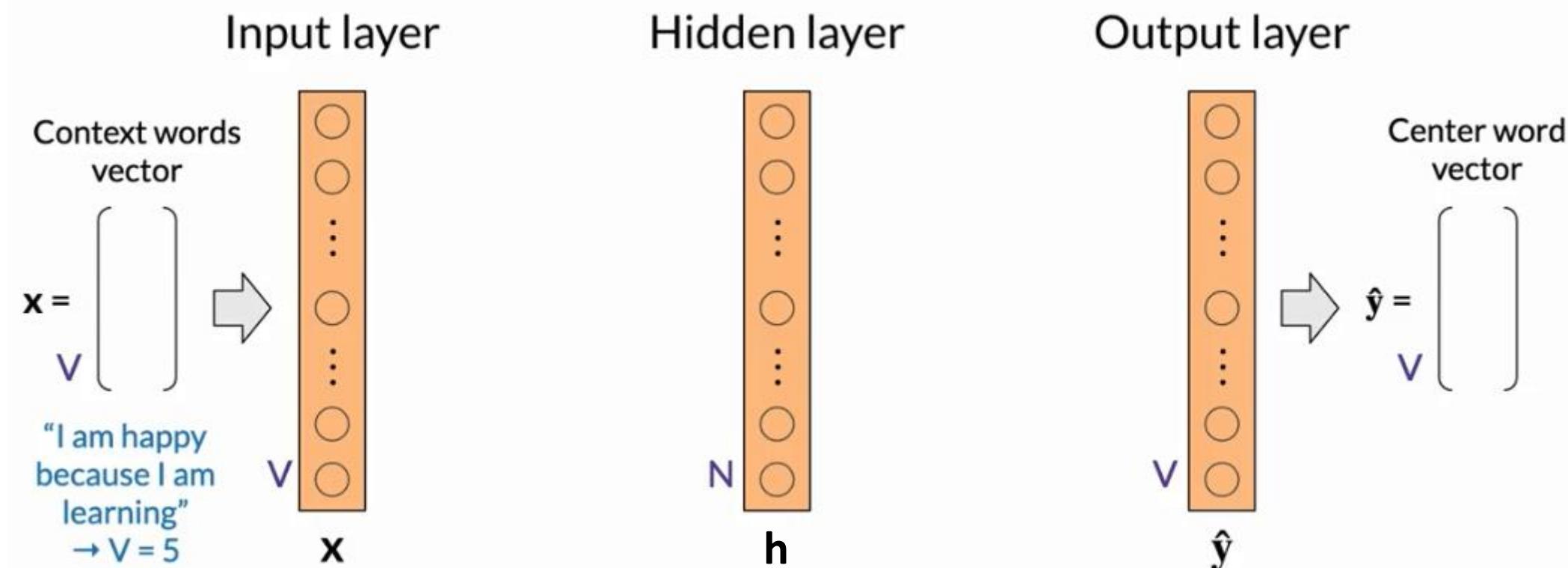
Hamidreza Baradaran Kashani

معماری مدل CBOW



- ❖ مدل CBOW بر اساس یک شبکه عصبی کم عمق است (Shallow NN).
- ❖ مدل شامل ۳ لایه ورودی، مخفی و خروجی است.
- ❖ تعداد نرون های لایه ورودی و خروجی ۷ برابر با تعداد کلمات داخل دیکشنری است.
- ❖ تعداد نرون های لایه مخفی (N) یک هایپرپارامتر است که بعد فضای تعبیه کلمات را بیان می کند.
- ❖ نرون های تمام لایه ها بطور کامل به یکدیگر متصل هستند بنابراین شبکه عصبی مذکور یک شبکه **fully-connected** نیز نامیده می شود.

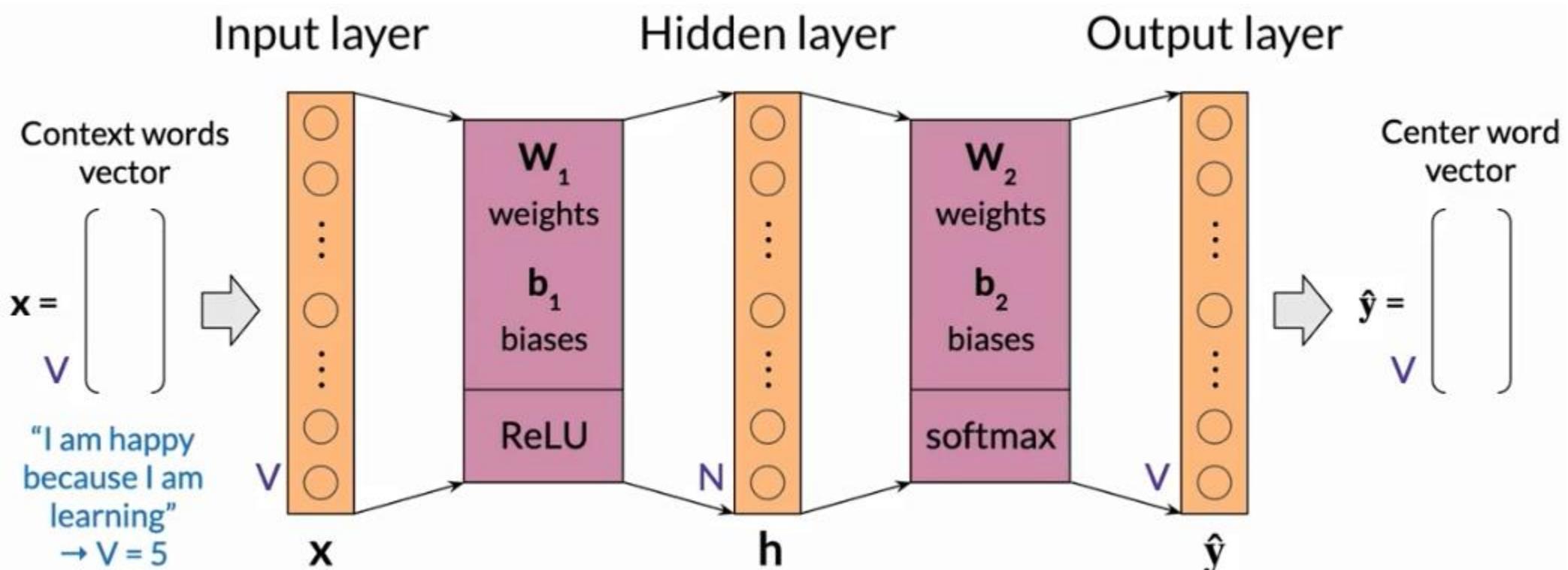
معماری مدل CBOW



Hamidreza Baradaran Kashani



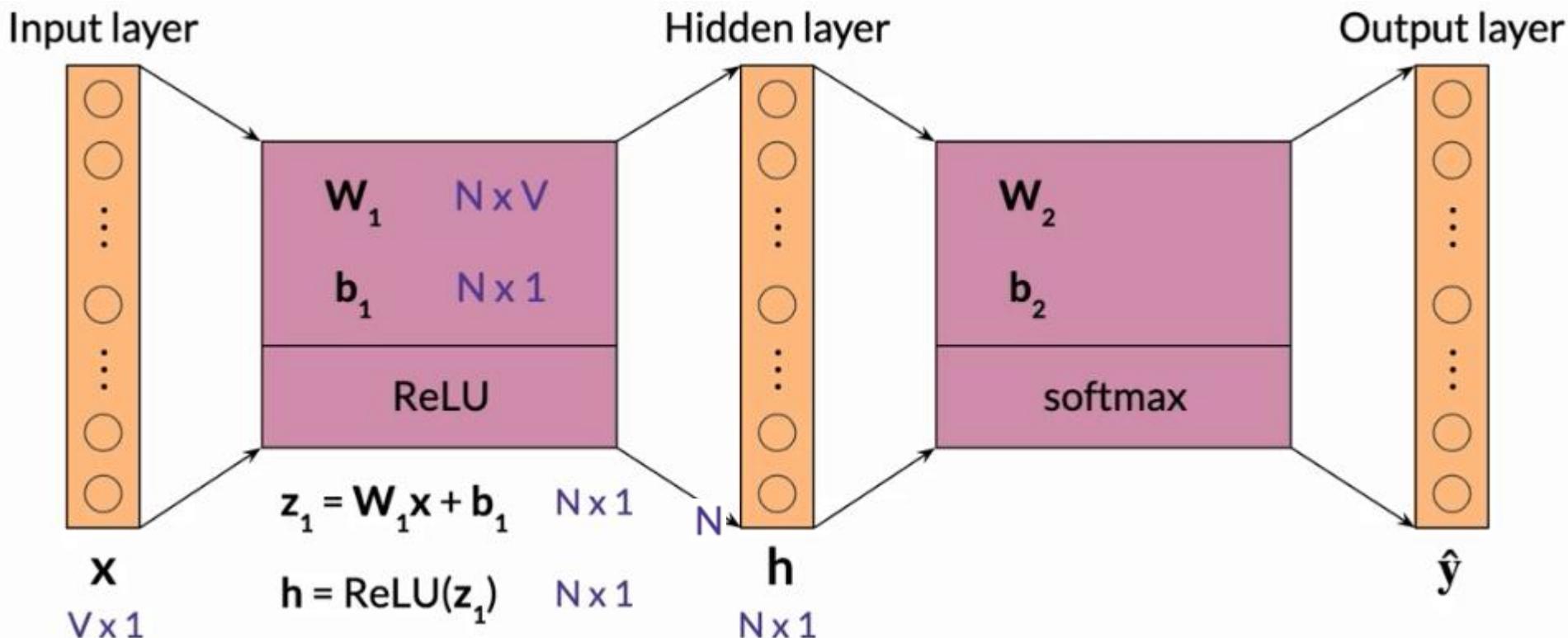
معماری مدل CBOW



Hamidreza Baradaran Kashani

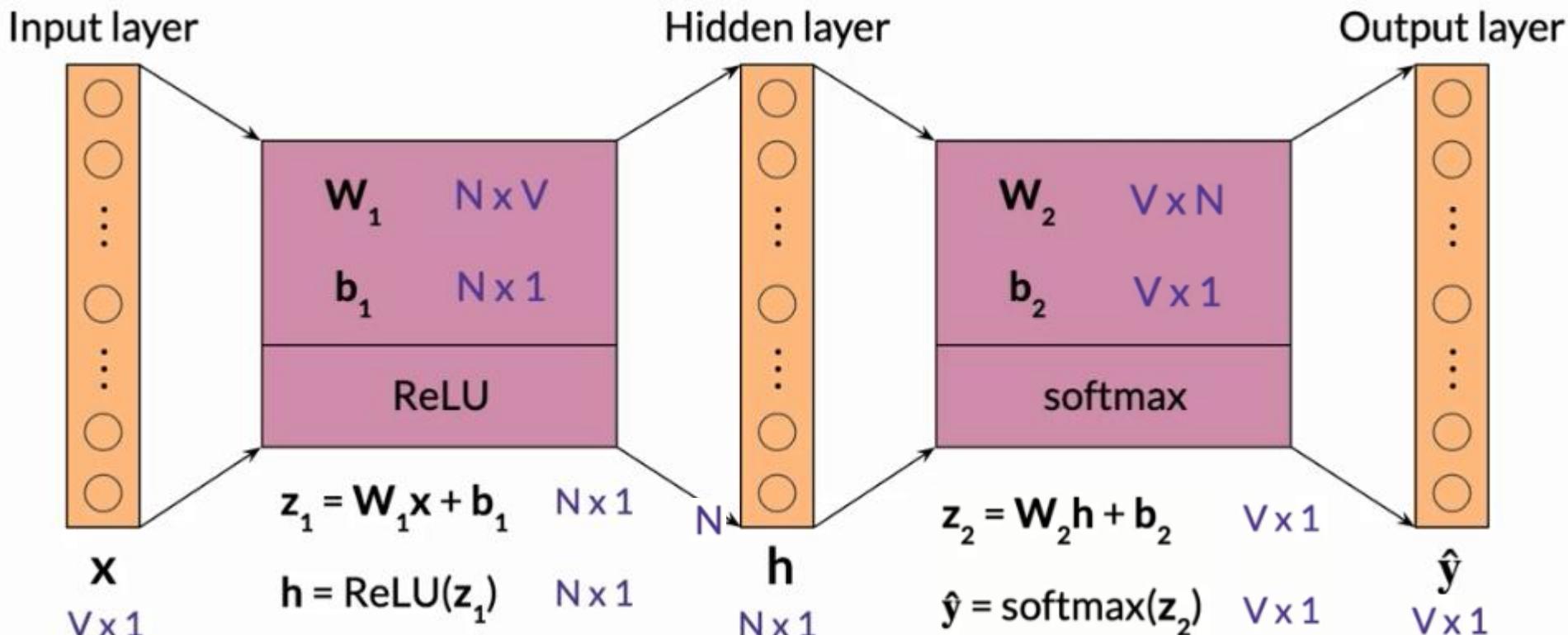


معماری مدل CBOW - ابعاد بردارها و ماتریس ها





معماری مدل CBOW - ابعاد بردارها و ماتریس ها



z_2 is called “**logits**”

Hamidreza Baradaran Kashani



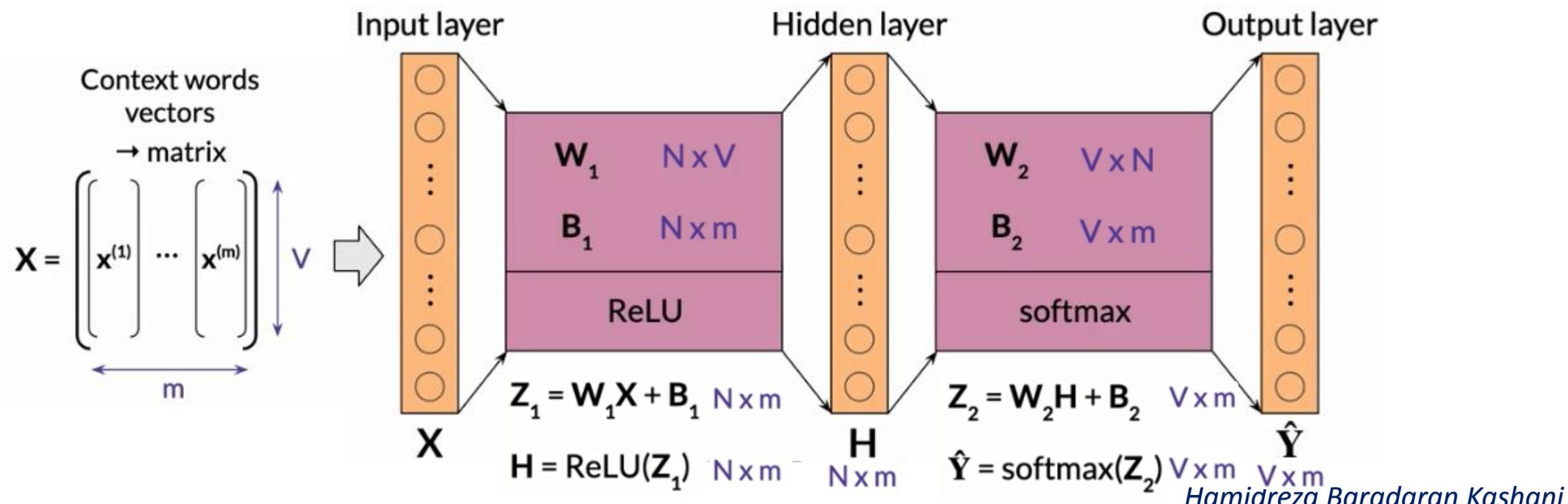
معماری مدل CBOW - ابعاد در پردازش دسته ای

- ❖ تا اینجا فرض شده است که داده ورودی به شبکه در هر زمان یک نمونه است.
- ❖ در شرایط واقعی و عملیاتی **در هر زمان چند نمونه** به عنوان ورودی به شبکه داده می شود که به آن **دسته (batch)** می گویند.
- ❖ پردازش شبکه با یک دسته را هم **پردازش دسته ای (batch processing)** می گویند.
- ❖ در ادامه بررسی می کنیم که ابعاد بردارها و ماتریس ها در مدل CBOW در یک پردازش دسته ای چگونه خواهد بود؟



معماری مدل CBOW - ابعاد در پردازش دسته‌ای

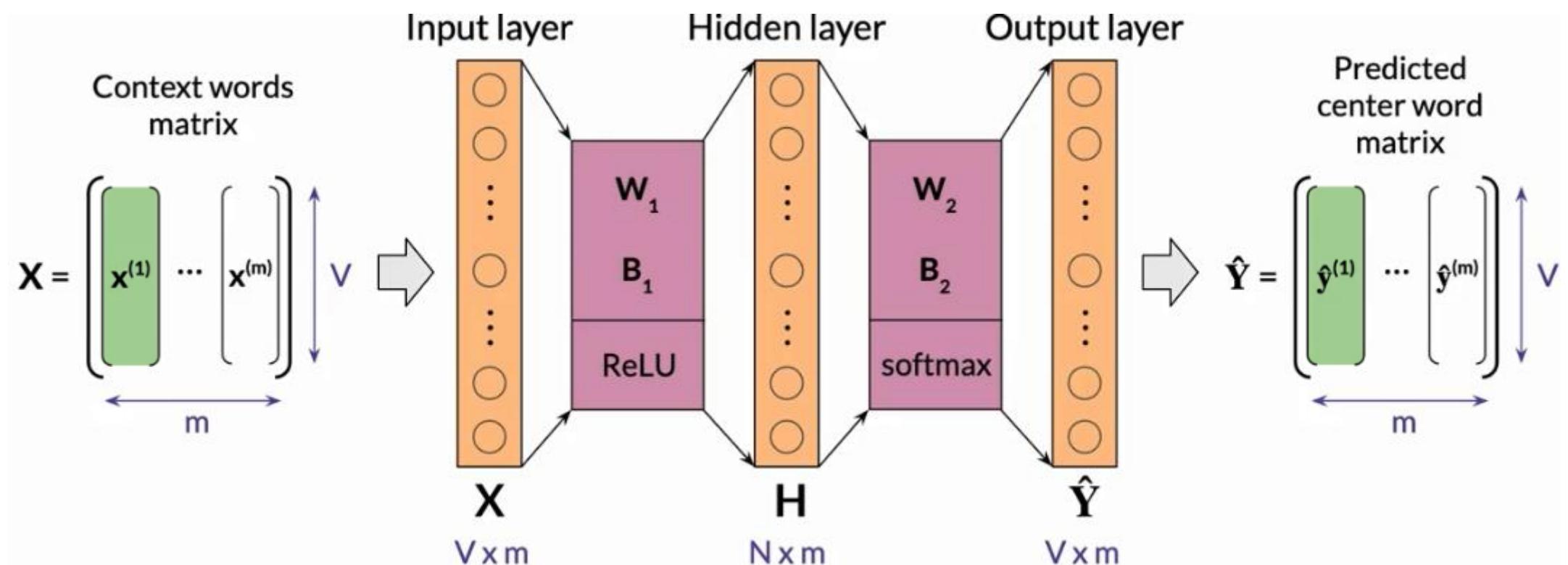
$$\begin{bmatrix} \mathbf{b}_1 \end{bmatrix} \rightarrow \mathbf{B}_1 = \begin{bmatrix} \begin{bmatrix} \mathbf{b}_1 \end{bmatrix} \\ \vdots \\ \begin{bmatrix} \mathbf{b}_1 \end{bmatrix} \end{bmatrix} \xrightarrow[N]{\text{broadcasting}}$$



Hamidreza Baradaran Kashani



معماری مدل CBOW - ابعاد در پردازش دسته‌ای

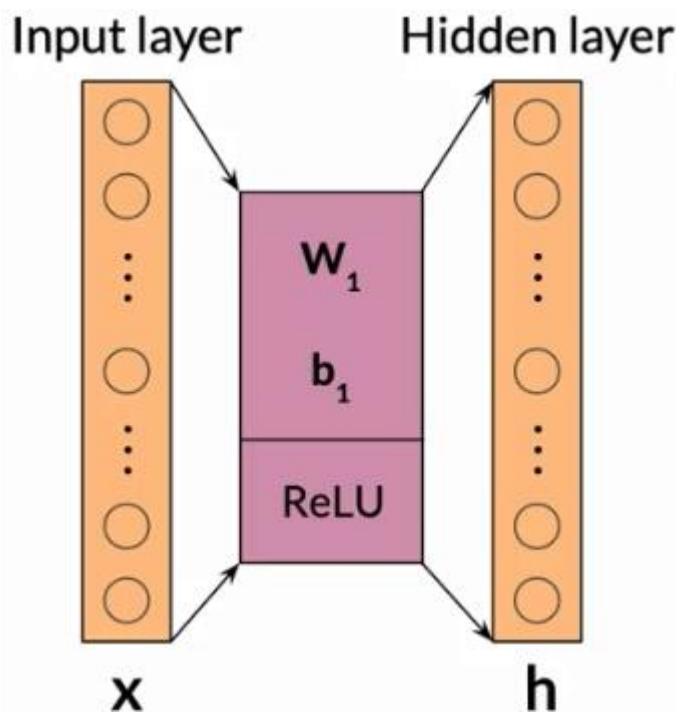


Hamidreza Baradaran Kashani



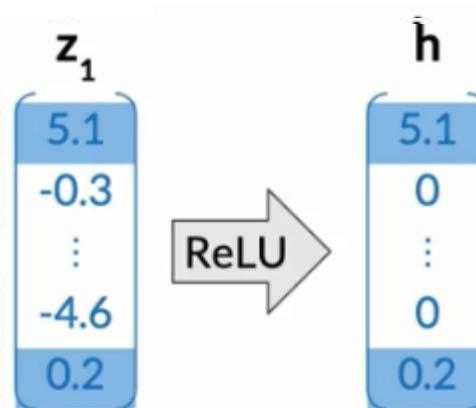
معماری مدل CBOW - توابع فعالسازی

تابع فعالسازی \diamond (Rectified Linear Unit) ReLU

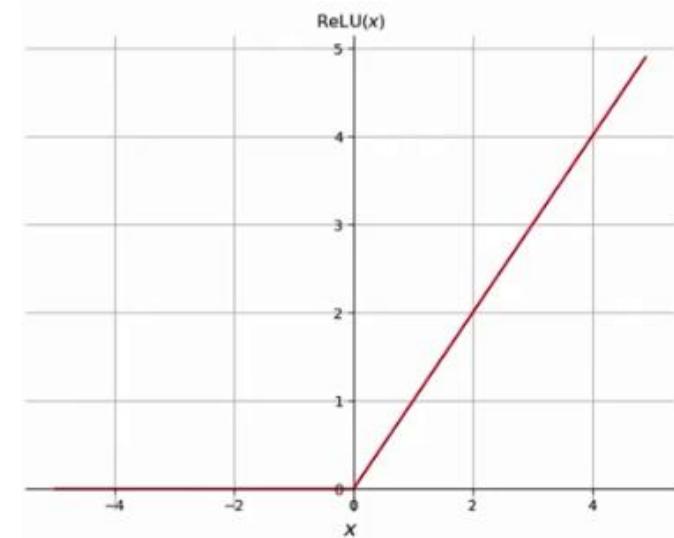


$$z_1 = W_1 x + b_1$$

$$h = \text{ReLU}(z_1)$$



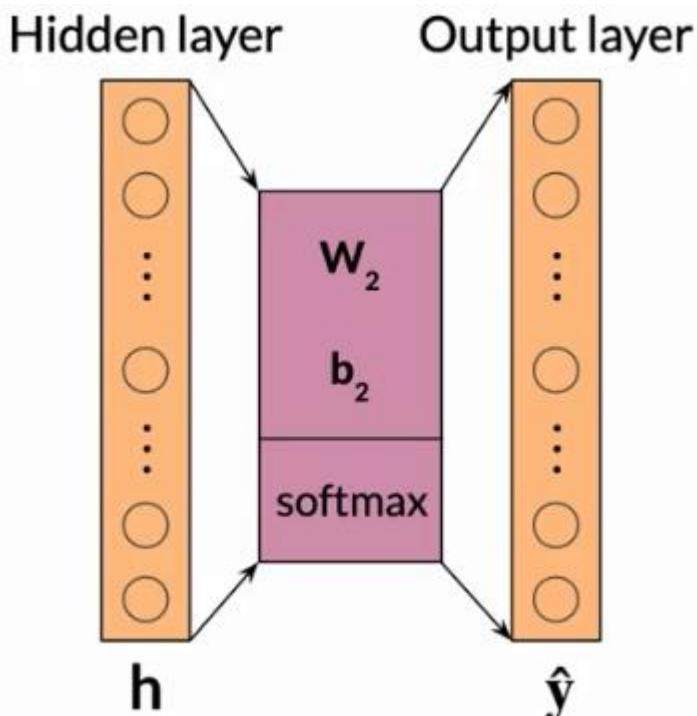
$$\text{ReLU}(x) = \max(0, x)$$



Hamidreza Baradaran Kashani

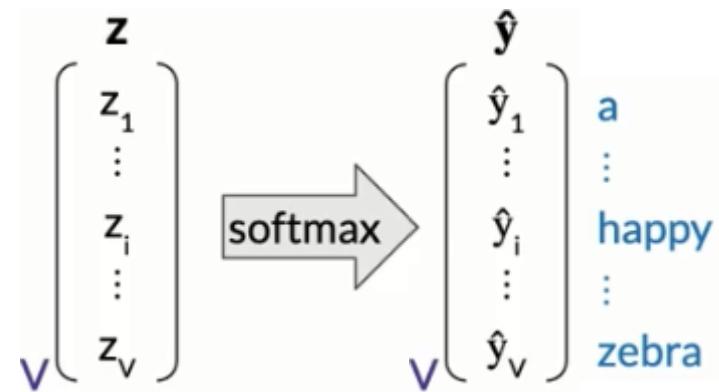
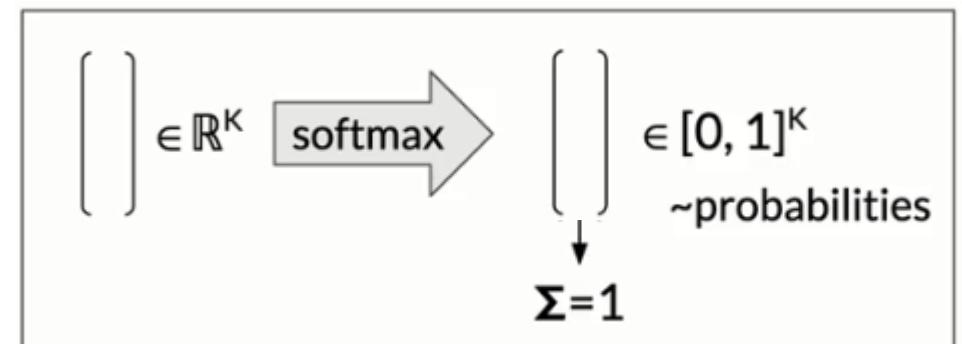


معماری مدل CBOW - توابع فعالسازی



$$z = W_2 h + b_2$$
$$\hat{y} = \text{softmax}(z)$$

تابع فعالسازی \diamond



Probabilities of
being center word

$$\hat{y}_i = \frac{e^{z_i}}{\sum_{j=1}^V e^{z_j}}$$

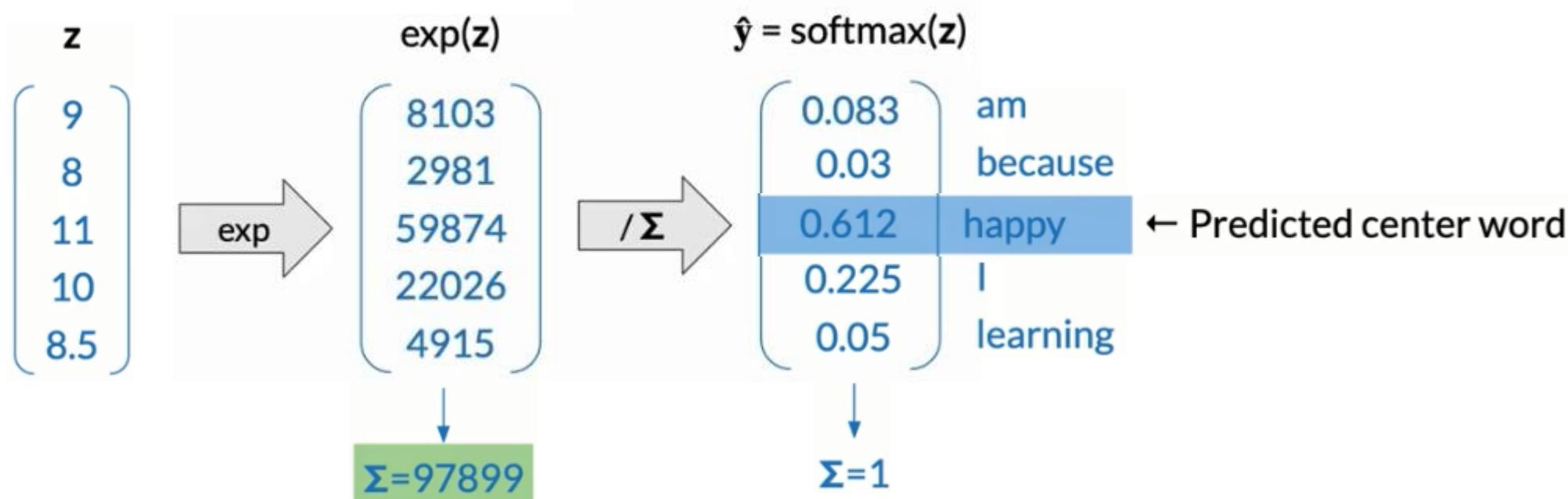
Hamidreza Baradaran Kashani



معماری مدل CBOW - توابع فعالسازی

$$\hat{y}_i = \frac{e^{z_i}}{\sum_{j=1}^V e^{z_j}}$$

تابع فعالسازی Softmax - مثال



Hamidreza Baradaran Kashani

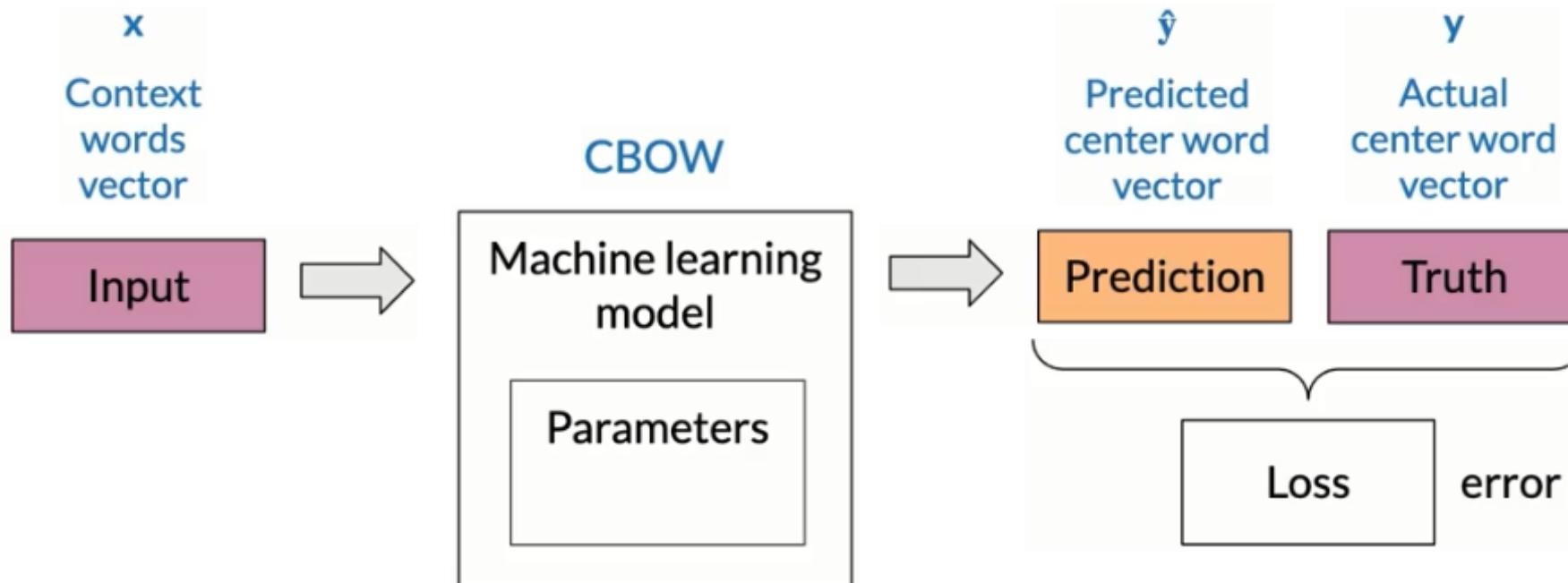


معماری مدل CBOW - تابع هدف

- ❖ در مدل CBOW هدف آن است که برای هر نمونه یادگیری با توجه به بردار زمینه (Context) توسط مدل پیش بینی شود.
- ❖ این فرآیند پیش بینی نیاز به یادگیری مدل دارد.
- ❖ برای یادگیری هر مدلی در حوزه یادگیری ماشین نیاز است که از یک **تابع هدف** (cost function) برای بهینه سازی پارامترهای مدل تا رسیدن به یک عملکرد بهینه از مدل استفاده کرد.
- ❖ در ادامه در ارتباط با تابع هدف در مدل تعییه کلمات CBOW صحبت خواهیم کرد.



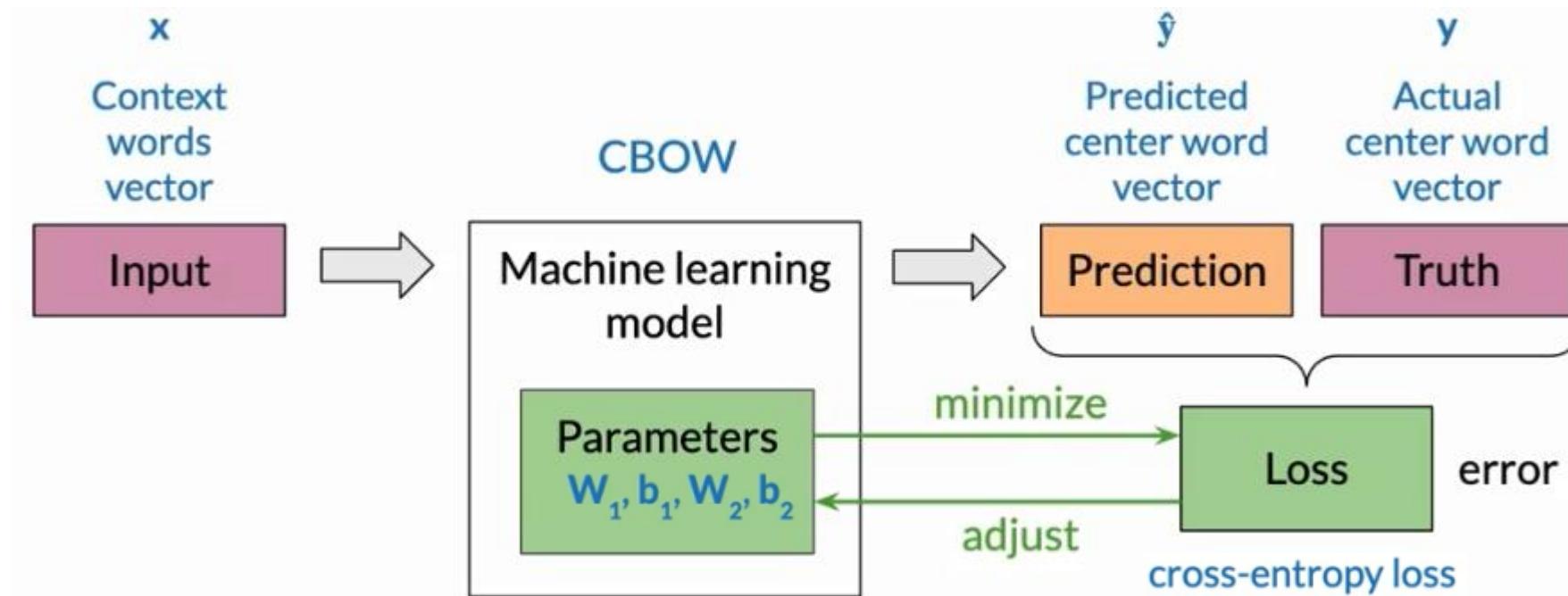
معماری مدل CBOW - تابع هدف



Hamidreza Baradaran Kashani



معماری مدل CBOW - تابع هدف



هدف فرآیند یادگیری مدل، یافتن پارامترهایی است که
تابع هدف را با توجه به مجموعه یادگیری کمینه کند

Hamidreza Baradaran Kashani



معماری مدل CBOW - تابع هدف

Cross-entropy loss

$$J = - \sum_{k=1}^V y_k \log \hat{y}_k$$

: محاسبه تابع آنتروپی متقاطع
برای یک نمونه یادگیری

Actual
 $\mathbf{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_v \end{pmatrix}$

Predicted
 $\hat{\mathbf{y}} = \begin{pmatrix} \hat{y}_1 \\ \vdots \\ \hat{y}_v \end{pmatrix}$

I am happy because I am learning

y		\hat{y}
0	am	0.083
0	because	0.03
1	happy	0.611
0	I	0.225
0	learning	0.05

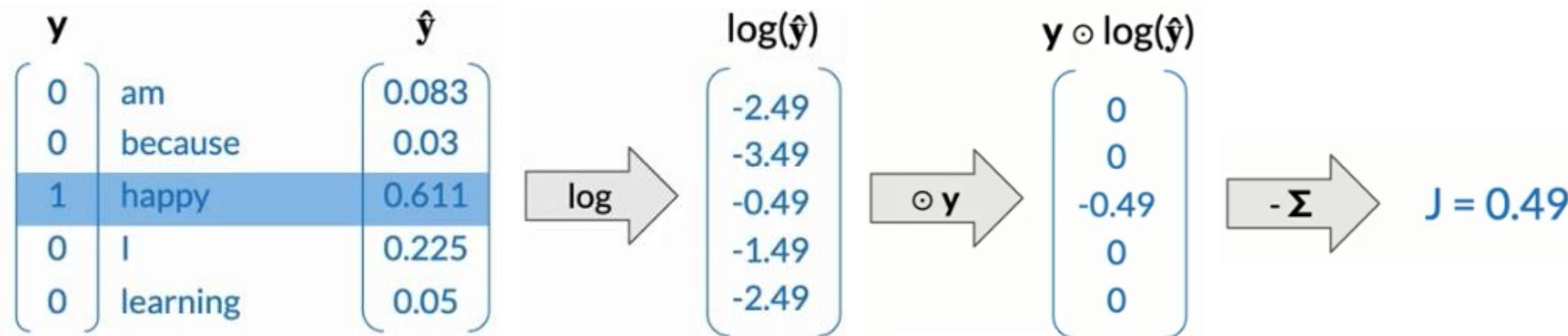
با توجه به آنکه خروجی شبکه متناظر با کلمه happy بیشترین
مقدار را دارد، پس شبکه به درستی کلمه مرکزی را با توجه به
کلمات اطراف پیش بینی کرده است.



معماری مدل CBOW - تابع هدف

Cross-entropy loss

$$J = - \sum_{k=1}^V y_k \log \hat{y}_k$$



Hamidreza Baradaran Kashani



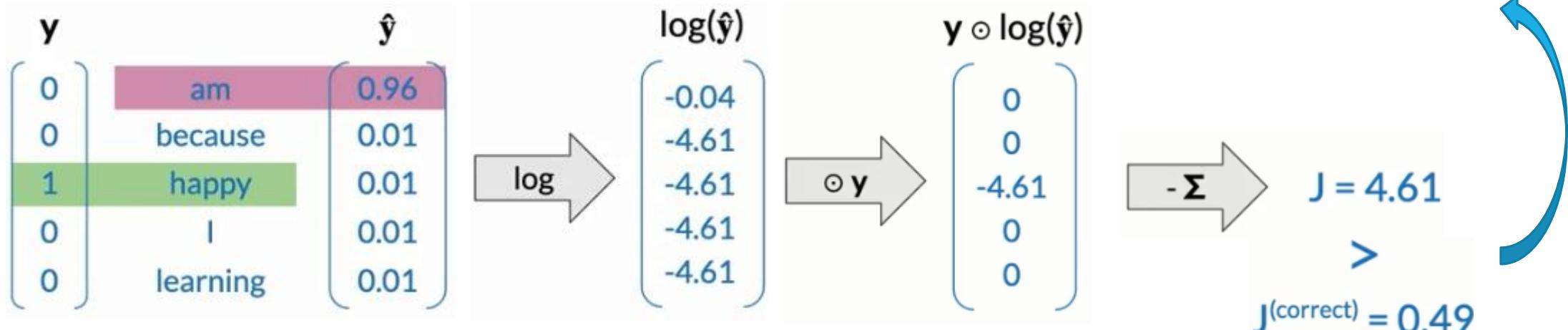
معماری مدل CBOW - تابع هدف

Cross-entropy loss

$$J = - \sum_{k=1}^V y_k \log \hat{y}_k$$

❖ فرض کنیم پیش بینی شبکه اشتباه باشد:

در اینصورت تابع هزینه نتیجه شده مقدار بزرگتری نسبت به حالتی که پیش بینی درست بود دارد.



Hamidreza Baradaran Kashani



معماری مدل CBOW - تابع هدف

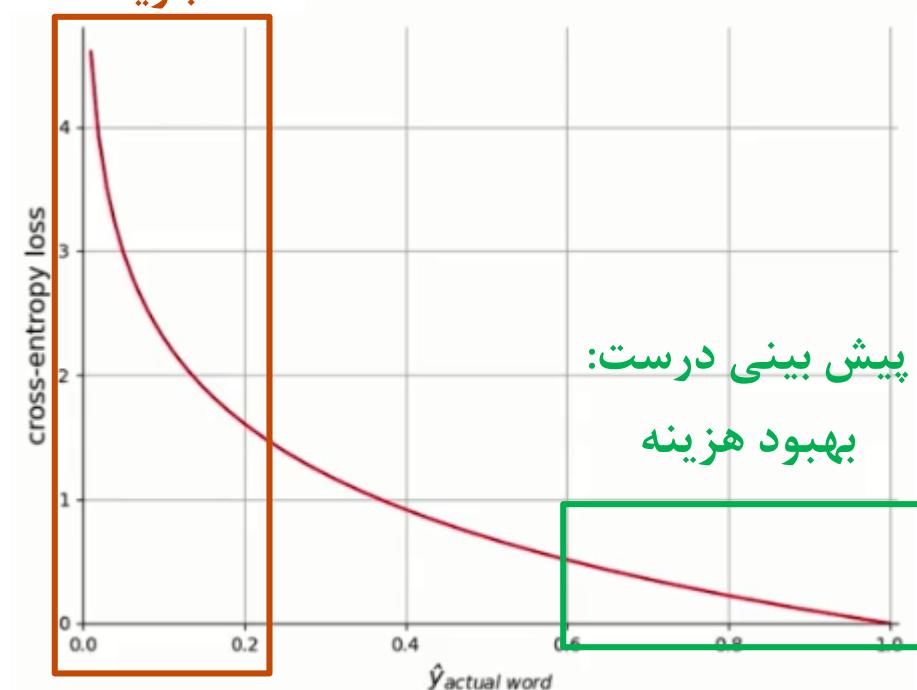
Cross-entropy loss

$$J = -\log \hat{y}_{\text{actual word}}$$

y		\hat{y}	
0	am	0.96	
0	because	0.01	
1	happy	0.01	→ $J = 4.61$
0	I	0.01	
0	learning	0.01	

پیش بینی غلط:

جريمه



پیش بینی درست:

بهبود هزینه

Hamidreza Baradaran Kashani



معماری مدل CBOW - فرآیند یادگیری

❖ در مدل CBOW فرآیند یادگیری شامل سه مرحله اصلی است:

(Forward Propagation) انتشار رو به جلو

- در این مرحله دسته ای از نمونه های یادگیری (با عنوان batch) به شبکه داده شده و تمام محاسبات شبکه تا آخرین لایه انجام می شود.

(Gradient descent) محاسبه تابع هزینه

- در این مرحله تابع هزینه آنتروپی متقاطع برای هر دسته نمونه ها محاسبه می شود.

(Backward Propagation) و نزول گرادیان انتشار رو به عقب

- در این مرحله مشتق تابع هدف نسبت به تمام پارامترهای شبکه محاسبه شده و با استفاده از الگوریتم نزول گرادیان پارامترها به روز می شوند.

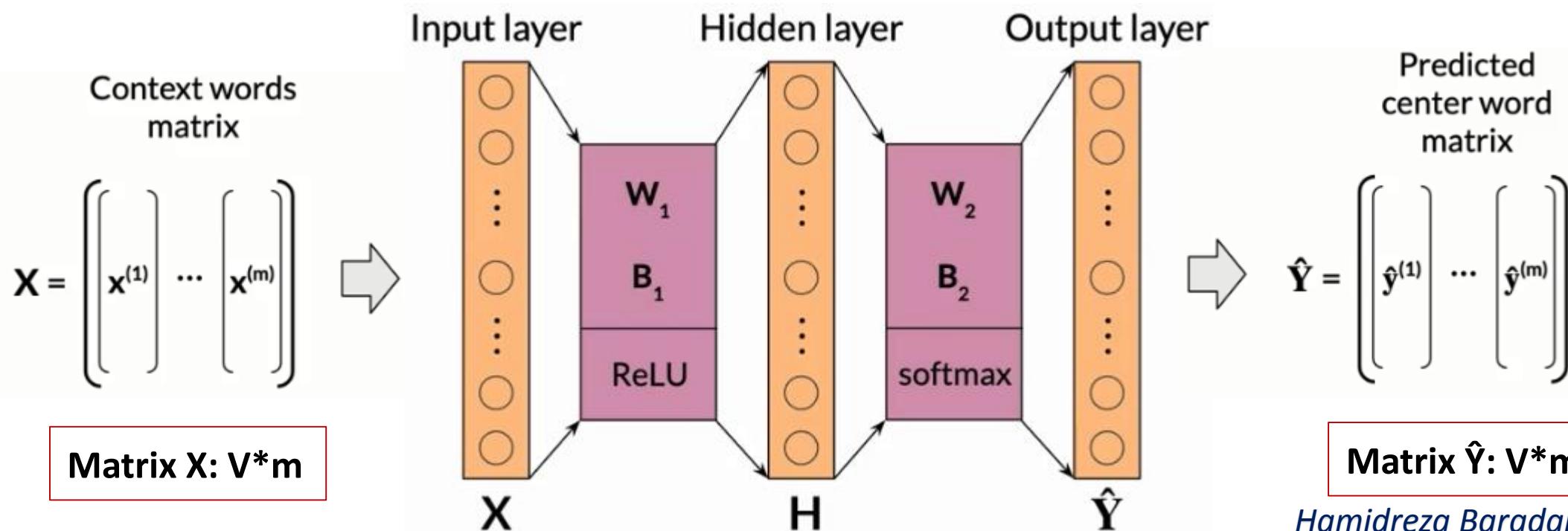


معماری مدل CBOW - انتشار رو به جلو

m: batch size

V: Vocabulary size

$$\begin{aligned} Z_1 &= W_1 X + B_1 & Z_2 &= W_2 H + B_2 \\ H &= \text{ReLU}(Z_1) & \hat{Y} &= \text{softmax}(Z_2) \end{aligned}$$



Hamidreza Baradaran Kashani



معماری مدل CBOW - محاسبه تابع هزینه

Cost: mean of losses

$$J_{batch} = -\frac{1}{m} \sum_{i=1}^m \sum_{j=1}^V y_j^{(i)} \log \hat{y}_j^{(i)}$$

$$J_{batch} = -\frac{1}{m} \sum_{i=1}^m J^{(i)}$$

Predicted
center word
matrix

$$\hat{\mathbf{Y}} = \left[\begin{array}{c} \hat{\mathbf{y}}^{(1)} \\ \vdots \\ \hat{\mathbf{y}}^{(m)} \end{array} \right]$$

Actual center
word matrix

$$\mathbf{Y} = \left[\begin{array}{c} \mathbf{y}^{(1)} \\ \vdots \\ \mathbf{y}^{(m)} \end{array} \right]$$



معماری مدل CBOW - انتشار رو به عقب و نزول گرادیان

❖ انتشار رو به عقب (Backward Propagation)

- این الگوریتم مشتقات جزیی (partial derivatives) یا همان گرادیان تابع هدف نسبت به تمام پارامترهای شبکه (شامل وزن ها و بایاس ها) را محاسبه می کند.

$$\frac{\partial J_{batch}}{\partial \mathbf{W}_1}, \frac{\partial J_{batch}}{\partial \mathbf{W}_2}, \frac{\partial J_{batch}}{\partial \mathbf{b}_1}, \frac{\partial J_{batch}}{\partial \mathbf{b}_2}$$

❖ نزول گرادیان (Gradient descent)

- به روز رسانی وزن ها و بایاس با این الگوریتم انجام می شود

Hamidreza Baradaran Kashani



معماری مدل CBOW-انتشار رو به عقب

$$\frac{\partial J_{batch}}{\partial \mathbf{W}_1} = \frac{1}{m} \text{ReLU} \left(\mathbf{W}_2^\top (\hat{\mathbf{Y}} - \mathbf{Y}) \right) \mathbf{X}^\top$$

$$\frac{\partial J_{batch}}{\partial \mathbf{W}_2} = \frac{1}{m} (\hat{\mathbf{Y}} - \mathbf{Y}) \mathbf{H}^\top$$

$$\frac{\partial J_{batch}}{\partial \mathbf{b}_1} = \frac{1}{m} \text{ReLU} \left(\mathbf{W}_2^\top (\hat{\mathbf{Y}} - \mathbf{Y}) \right) \boxed{\mathbf{1}_m^\top}$$

$$\frac{\partial J_{batch}}{\partial \mathbf{b}_2} = \frac{1}{m} (\hat{\mathbf{Y}} - \mathbf{Y}) \mathbf{1}_m^\top$$

$$\mathbf{1}_m = [1, \dots, 1] \quad \xrightarrow[m]{}$$
$$\mathbf{A} \cdot \mathbf{1}_m^\top = \left[\begin{array}{c|c} \hline \text{---} & \mathbf{1} \\ \hline \end{array} \right] \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} = \boxed{\Sigma}$$

```
import numpy as np  
# code to initialize matrix a omitted  
np.sum(a, axis=1, keepdims=True)
```

در این درس به نحوه محاسبه مشتقات جزئی کار نداریم!

Hamidreza Baradaran Kashani



معماری مدل CBOW-نزول گرادیان

Hyperparameter: learning rate α

$$\mathbf{W}_1 := \mathbf{W}_1 - \alpha \frac{\partial J_{batch}}{\partial \mathbf{W}_1}$$

$$\mathbf{W}_2 := \mathbf{W}_2 - \alpha \frac{\partial J_{batch}}{\partial \mathbf{W}_2}$$

$$\mathbf{b}_1 := \mathbf{b}_1 - \alpha \frac{\partial J_{batch}}{\partial \mathbf{b}_1}$$

$$\mathbf{b}_2 := \mathbf{b}_2 - \alpha \frac{\partial J_{batch}}{\partial \mathbf{b}_2}$$

- پارامتر آلفا مقدار کمتر از ۱ انتخاب می شود.
- این پارامتر بیان می کند که چه مقدار از گرادیان تابع هزینه نسبت به یک پارامتر در بروزرسانی آن نقش داشته باشد.
- مقدار کوچک آلفا بروزرسانی های آرام (یا کم) پارامترها را نتیجه می دهد و مقدار بزرگ آلفا بروزرسانی های سریع (یا زیاد) پارامترها را حاصل می کند.



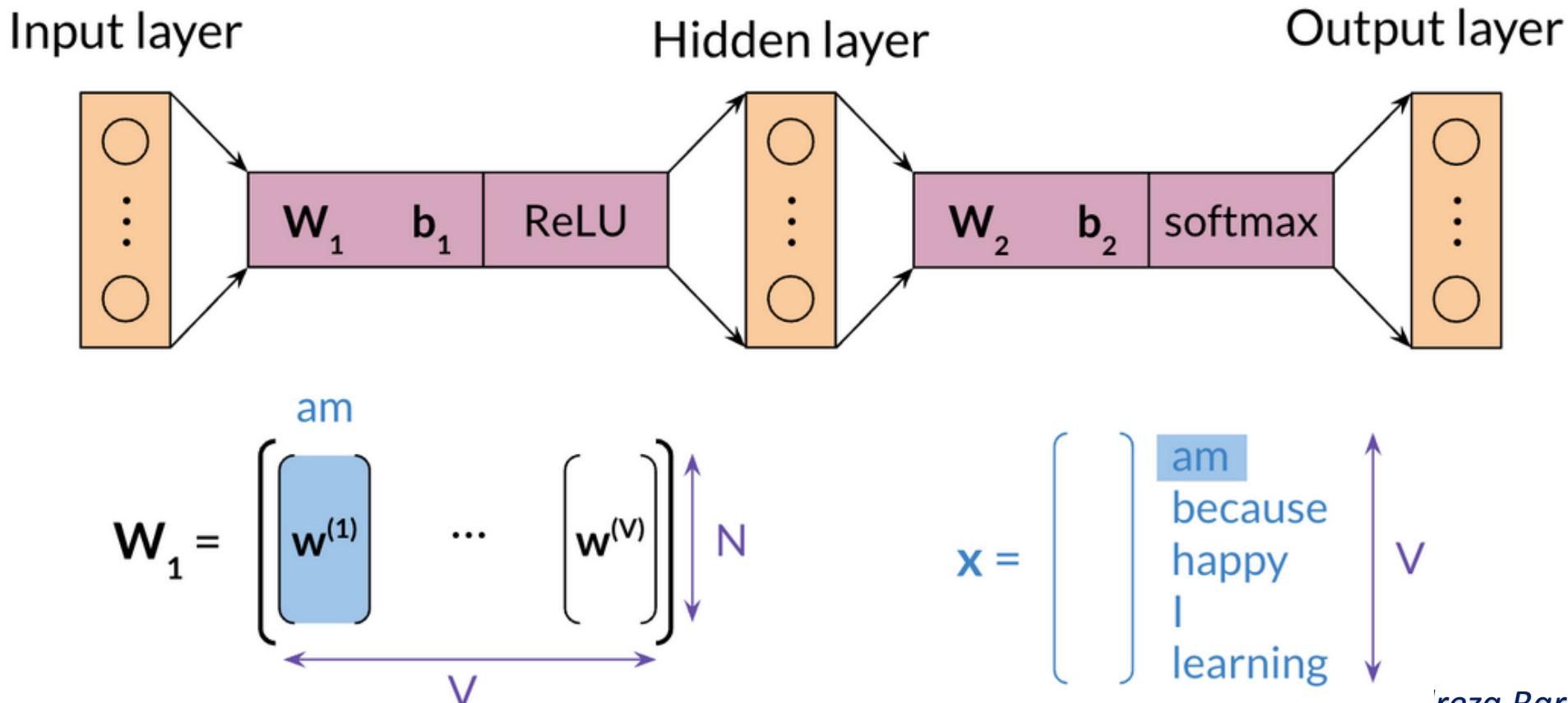
استخراج بردار تعبیه کلمات از مدل CBOW

- ❖ فرض شود که مدل CBOW با نمونه های یادگیری آموزش داده شده و وزن های بهینه حاصل شده اند.
- ❖ حال می خواهیم بردار تعبیه کلمات را برای یک کلمه مشخص استخراج کنیم.
- ❖ سه روش برای استخراج بردارهای تعبیه کلمات وجود دارند.



استخراج بردار تعییه کلمات از مدل CBOW

روش ۱: هر ستون W_1 بردار تعییه متناظر با یک کلمه مشخص است

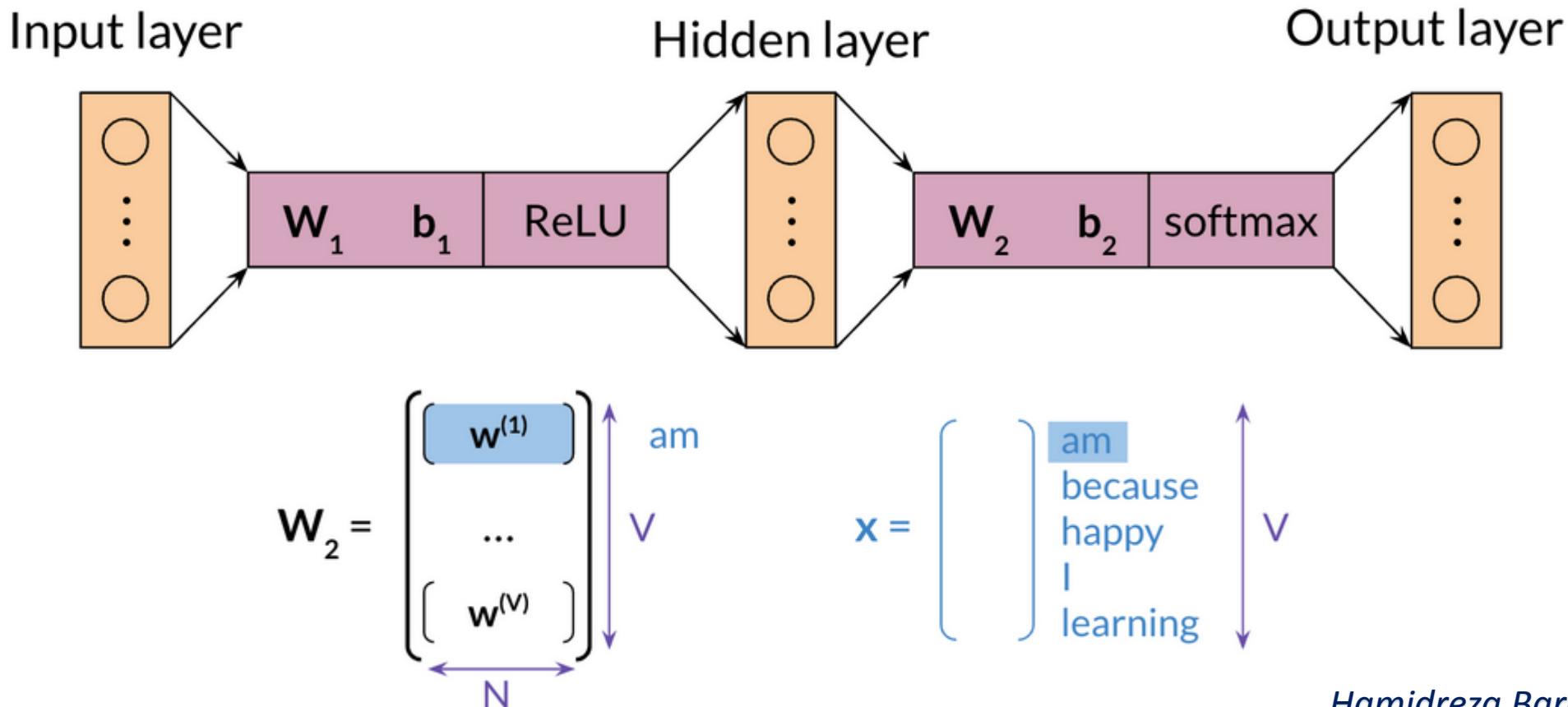


reza Baradaran Kashani



استخراج بردار تعییه کلمات از مدل CBOW

روش ۲: هر ستون W_2 بردار تعییه متناظر با یک کلمه مشخص است





استخراج بردار تعبیه کلمات از مدل CBOW

روش ۳: متوسط بردارهای نتیجه شده از روش ۱ و ۲

$$\mathbf{W}_3 = 0.5 (\mathbf{W}_1 + \mathbf{W}_2^T) = \begin{pmatrix} \mathbf{w}_3^{(1)} & \dots & \mathbf{w}_3^{(V)} \end{pmatrix}$$

V