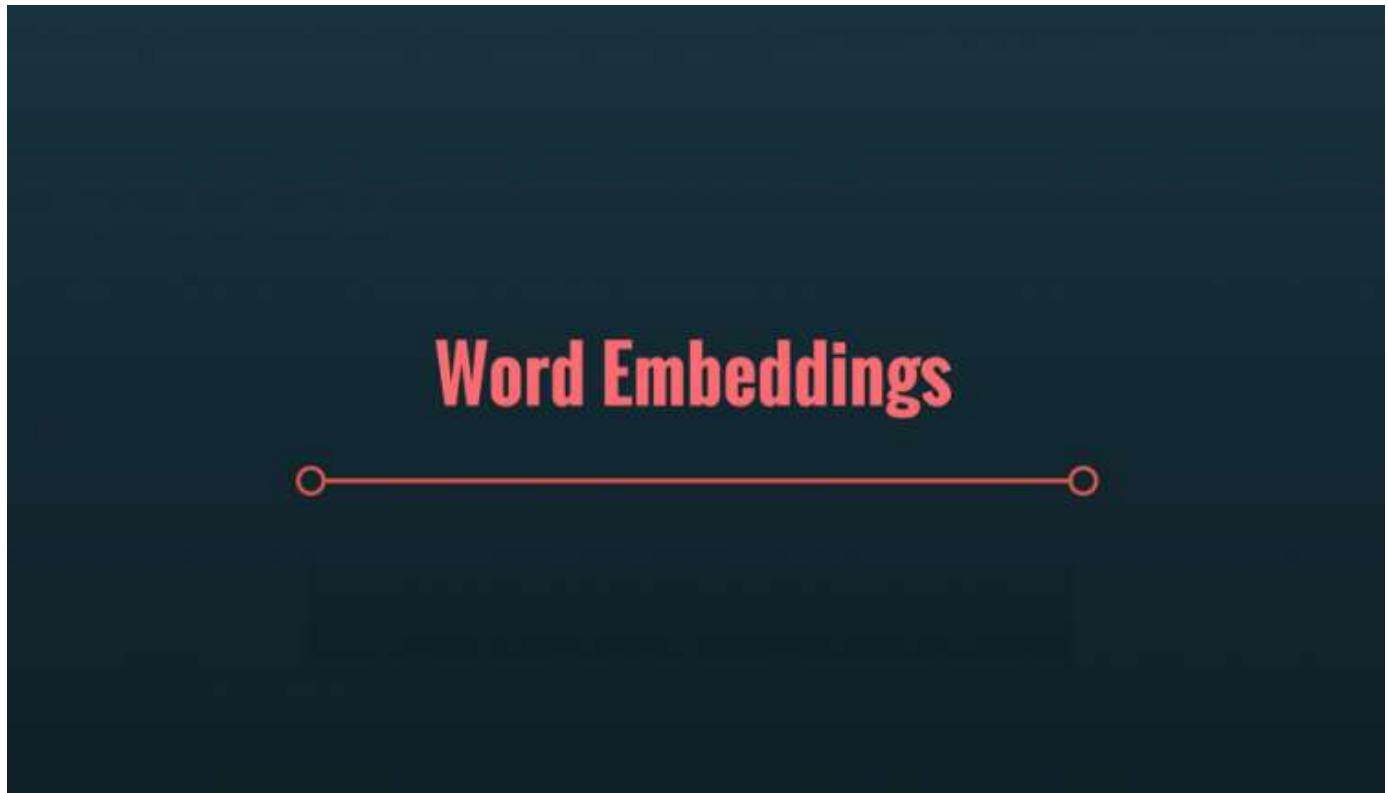


# مرجع یادگیری عمیق ایران

خانه < یادگیری عمیق <



یادگیری عمیق

## Word Embedding چیست؟

آخرین بروزرسانی مه 18, 2019

در مه 2019

...

توسط سید حسین حسن پور ...



بسم الله الرحمن الرحيم

در پست قبلی ما معرفی خیلی کوتاهی از word embedding یا همان "تعییه سازی کلمات" و روش‌های تولید آن داشتیم. در این نوشتار سعی میکنیم یکبار دیگر به این موضوع پرداخته و با زبان ساده‌تری با مفاهیم مطرح شده در قبل مثل تعییه کلمه، شیوه کار word2vec و...بهتر آشنا شویم.

سرفصل‌ها در این نوشتار بصورت زیر است:

- تعریف
- Word embedding چیست؟
- بردار کلمات یا اصطلاحاً Word Vector چیست؟
- یک مثال ابتدایی از Word embedding - معرفی One-hot encoding
- پیاده سازی یک طرح word embedding پیچیده
- خصائص word embedding
- خصائص و ویژگی‌های زبان شناختی word embedding (استدلال با بردار کلمات)

## • فراگیری بردار کلمات

- مدل Continuous Bag of Words
- مدل Continuous Skip-gram
- بهینه سازی Softmax
- نمونه برداری منفی

- الگوریتم های رایج Word2vec,GloVe,Fasttext word embedding مثل
- کتابخانه های مطرح جهت استفاده از word embedding در پایتون
- کاربردهای word embedding

## تعريف

### + نکته

Word embedding ها بردار های عددی هستند که نمایانگر کلمات یک لغت نامه اند و کاربردهای گسترده ای هم در حوزه پردازش زبان طبیعی دارند (البته محدود به این حوزه نیستند!). اگر تا بحال از word embedding در کارهای مرتبط با دسته بندی متن یا احساس و یا سایر وظایف مرتبط با پردازش زبان طبیعی، استفاده نکرده باشید، به احتمال بسیار زیاد با بهره گیری از آن خواهید توانست بطور قابل توجهی دقت مدل خود را افزایش دهید. Word embedding بshima اجزاء میدهد تا بطور غیرصریح اطلاعاتی را از دنیای بیرونی به مدل های زبانی خود اضافه کنید.

از سال ۱۹۹۰ ، مدلهای مبتنی بر فضای بردار در زمینه word embedding مورد استفاده محققان بوده اند. در طول این زمان، مدلهای بسیاری برای تقریب بازنمایی پیوسته کلمات(Continuous representations of words) توسعه داده شده اند که از Latent Dirichlet Allocation(LDA) و Latent Semantic Analysis(LSA) تاریخچه مفصلی در این باره آمده که مطالعه آن خالی از لطف نیست. بنجیو و همکارانش در سال ۲۰۰۳ عبارت word embedding را ابداع کردند. اولین کسانی سودمندی word embedding های از پیش آموزش دیده را نشان دادند Collobert و Weston در سال ۲۰۰۸ بودند. مقاله معروف آنها "A unified architecture for natural language processing" نه تنها word embedding را بعنوان یکی از ابزارهای سودمند برای وظایف پایین دستی حوزه پردازش زبان طبیعی، تثییت کرد، بلکه معماری شبکه عصبی ای معرفی کرد که اساس بسیاری از روش‌های امروزی است. اما همه گیری تدریجی word embedding را میتوان به کار میکلوف و همکارانش در سال ۲۰۱۳ که Word2vec را ارائه کردند منتسب کرد. در سال ۲۰۱۴، پینینتگون و همکرانش GloVe را که مجموعه ای قدرتمند از Word embedding های از پیش آماده را ارائه کردند و با این کار نشان دادند Word embedding وارد استفاده روزمره شده است.(منبع)

## بطور ساده word vectors و word embeddings یعنی چه؟

مفهوم اصلی word embedding این است که تمامی لغات استفاده شده در یک زبان را میتوان توسط مجموعه ای از اعداد اعشاری (در قالب یک بردار) بیان کرد. Word embedding ها بردارهای  $n$ -بعدی ای هستند که تلاش میکنند معنای لغات و محتوای آنها را با مقادیر عددی خود ثبت و ضبط کنند. هر مجموعه ای از اعداد یک "بردار کلمه" معتبر بحساب می آید که الزاماً برای ما سودمند نیست، آن مجموعه ای از بردار کلمات برای کاربردهای مورد نظر ما سودمندند که معنای کلمات، ارتباط بین آنها و محتوای کلمات مختلف را همانطور که بصورت طبیعی [توسط ما] مورد استفاده قرار گرفته اند، بدست آورده باشند.

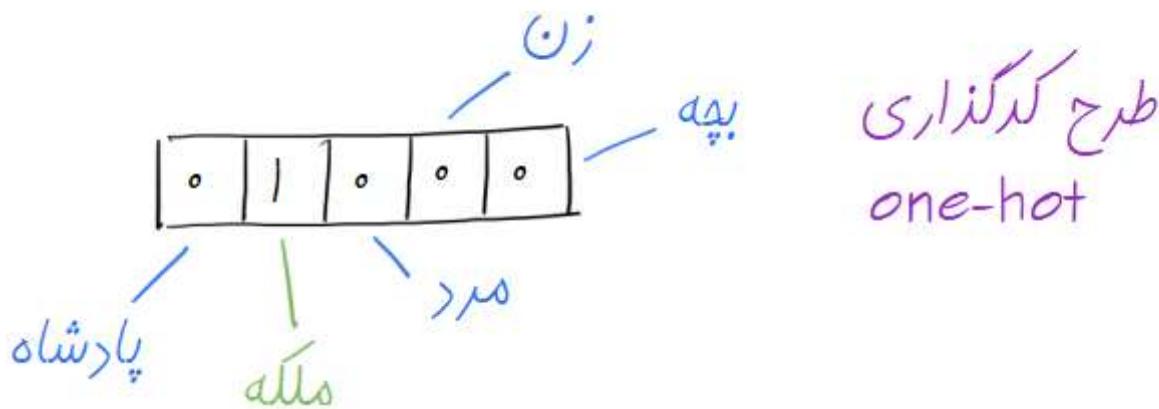
پس word embedding های سودمند چند مشخصه کلیدی دارند :

- هر کلمه دارای یک word embedding (یا بردار) یکتاست که تنها شامل لیستی ساده از اعداد به ازای هر کلمه است.

- word embedding ها چند بعدی اند و عموماً یک مدل خوب، embedding هایی با طولی بین ۵۰۰ الی ۵۰۰۰ بعد دارد.
- به ازای هر کلمه، embedding معنای آن کلمه را بدست می‌آورد.
- کلمات مشابه در نهایت به مقادیر embedding مشابه میرسند.

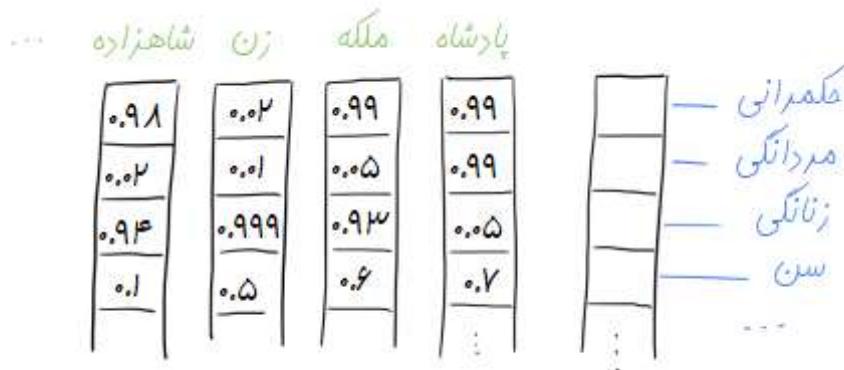
## اما بردار کلمه یعنی چه؟

در یک سطح، بردار کلمه خیلی ساده اشاره به برداری دارد که حاوی یکسری وزن (یا همان مقادیر عددی) است. مثلاً اگر طرح کدگذاری one-hot را در نظر بگیریم، در این طرح ساده، هر المان در بردار به یک کلمه در لغت نامه اختصاص داده می‌شود یعنی هر درایه این بردار معرف یک کلمه از لغت نامه خواهد بود. برای هر کلمه نیز تنها درایه متناظر برابر با ۱ بوده و مابقی درایه‌ها با صفر مقداردهی خواهند شد. مثلاً فرض کنید لغت نامه ما فقط حاوی ۵ کلمه "پادشاه"، "ملکه"، "مرد"، "زن" و "بچه" باشد. اگر از شیوه‌ای که تازه مطرح کردیم استفاده کنیم کلمه "ملکه" بصورت زیر بازنمایی خواهد شد:



با استفاده از این شیوه، ما نمی‌توانیم هیچ قیاس معناداری بین دو بردار داشته باشیم و تنها می‌توانیم مساوی بودن بردارها را بررسی کنیم!

در الگوریتم word2vec (نکته انتهایی را ببینید) اما از بازنمایی توزیع شده برای هر کلمه استفاده می‌شود. بعنوان مثال برداری با چند بعد را در نظر بگیرید. هر کلمه توسط توزیعی از مقادیر عددی (وزنها) بر روی درایه‌های مختلف بردار بازنمایی می‌شود. بنابر این بجای نگاشت یک به یک بین یک درایه در بردار و یک کلمه در لغت نامه، بازنمایی یک کلمه در تمامی درایه‌های یک بردار پخش می‌شود، و هر درایه بردار در مشخص شدن معنای تعداد زیادی از کلمات نقش ایفا می‌کند. برای درک بهتر این قضیه به مثال زیر توجه کنید. در مثال زیر کلمات مختلف توسط مقادیر مختلف هر درایه توصیف شده اند و تمامی درایه‌ها در ثبت و ضبط معنای کلمات مختلف نفش ایفا می‌کنند. (در اینجا ما خصائص ویژه‌ای را به هر درایه اختصاص داده ایم مثلاً درایه اول مشخص کننده "صفت" حکمرانی، درایه دوم بردار مشخص کننده صفت "مردانگی" و الی آخر است. در بردار کلمه پادشاه می‌بینیم که درایه‌های مختلف به ترتیب بر اساس میزان سنتیت با صفت با کلمه مورد بحث، دارای مقادیر (وزن‌های) مختلفی اند. بعنوان مثال صفت مردانگی برای پادشاه بسیار بالا بوده اما صفت زنانگی مقدار پایینی دارد این مساله برای ملکه و زن بر عکس است).



چنین برداری بوسیله روشی انتزاعی، معنای یک کلمه را ارائه میکند. همانطور که در ادامه خواهیم دید، صرفا با استفاده از یک مجموعه متنه بزرگ، میتوان بردار کلماتی را فرا گرفت که قادر به ثبت و ضبط روابط بین کلمات باشند. ما همچنین میتوانیم از این بردارها بعنوان ورودی به یک شبکه عصبی استفاده کنیم.

تمامی این نکات در ادامه با نمونه مثالهایی که جلوتر خواهیم دید روش میشنوند.

## یک مثال ساده از Word Embedding رمزگذاری One-hot

ساده ترین شکل یک طرح word embedding همانطور که تازه دیدیم کد گذاری one-hot است. دیدیم که در طرح کد گذاری one-hot یا کد گذاری of-N-1 فضای embedding دارای ابعاد مشابهی با کلمات موجود در لغت نامه است و هر word متشکل از برداری عموماً ۰ است که تنها بعد(درایه) متناظر با کلمه مورد انتظار مقداری برابر با ۱ دارد.

یک word embedding مبتنی بر روش کد گذاری one-hot برای لغت نامه کوچکی که حاوی ۹ کلمه است در شکل زیر نمایش داده شده است:

۹	۸	۷	۶	۵	۴	۳	۲	۱	
									پیرمرد
									پیرزن
									پسر
									دختر
									شاهزاده
									شاهدخت
									ملکه
									پادشاه
									حکمران

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

چند مشکل در رابطه با این شیوه کدگذاری وجود دارد :

- تعداد ابعاد (در اینجا ستون‌ها) با افزایش تعداد کلمات در لغت نامه بصورت خطی افزایش می‌آید. یعنی برای لغت نامه‌ای با ۵۰ هزار کلمه، هر کلمه بایستی توسط ۴۹,۹۹۹ صفر و تنها یک مقدار ۱ در مکان صحیح نمایش داده شود. به همین دلیل، مصرف حافظه بشدت افزایش پیدا می‌کند.
- ماتریس embedding نهایی خیلی تنک خواهد بود چرا که اکثراً از صفر تشکیل شده است.
- هیچ اشتراک اطلاعاتی بین کلمات و هیچ وجه مشترکی بین کلمات مشابه وجود ندارد. تمامی کلمات در فضای برداری با یک "فاسله یکسان" از یکدیگر قرار دارند (در مثال بالا هر word embedding ما یک بردار  $[1 \times 9]$  است).

## کدگذاری سفارشی!

حالا اگر بخواهیم ابعاد این شیوه کدگذاری را کاهش دهیم یعنی از تعداد عدد کمتری برای نمایش / معرفی هر کلمه استفاده کنیم چه باید کرد؟ یک راه حل این است که این کار را از طریق انتخاب دستی ابعاد لغت نامه انجام دهیم. یعنی چه؟ اجازه دهید این مساله را با یک مثال بیشتر باز کنیم. فرض کنید ما سه بُعد بنام مردانگی، جوانی و حکومت را انتخاب کنیم و مقادیری که این سه بُعد قبول می‌کنند مقادیر اعشاری بین ۰ و ۱ باشد. سعی کنید لیست زیر را با مقادیری که بنظر شما منطقی است پر کنید.

حکومت	جوانی	مردانگی	
◦	◦	◦	پیرمرد
◦	◦	◦	پیرزن
◦	◦	◦	پسر
◦	◦	◦	دختر
◦	◦	◦	شاهزاده
◦	◦	◦	شاهدخت
◦	◦	◦	ملکه
◦	◦	◦	پادشاه
◦	◦	◦	حکمران

ما میتوانیم با انتخاب دستی ابعادی که برای مثال ما منطقی بنظر میرسند یک نگاشت ۳ بعدی بسیار بهینه تر برای لغت نامه خود ایجاد کنیم!

تنها با کمی فکر کردن احتمالا شما هم به چیزی شبیه آنچه در زیر آمده است رسیده اید:

حکومت	جوانی	مردانگی	
۰	۰,۱	۱	پیرمرد
۰	۰,۱	۰	پیرزن
۰	۰,۹۵	۱	پسر
۰	۰,۹۵	۰	دختر
۰	۰,۸۵	۱	شاهزاده
۰	۰,۸۵	۰	شاهدخت
۱	۰,۴۵	۰	ملکه
۱	۰,۳۰	۱	پادشاه
۱	۰,۵	۰,۵	حکمران

ما توانستیم لغت نامه ۹ کلمه ای خود را تنها با بردار کلمات ۳ بعدی بطور نسبتاً بهینه ای ارائه کنیم. در این word embedding جدید، کلمات مشابه embedding embedding ها با بردار های مشابه دارند!

مجموعه word embedding های جدید چند برتری قابل توجه دارند :

۱. این مجموعه embedding ها بهینه تر از embedding هایی است که در ابتدا با کمک روش رمزگذاری one-hot بدست آورده بودیم. در اینجا هر کلمه توسط یک بردار ۳ بعدی ارائه شده است در حالی که روش قبلی هر کلمه توسط یک بردار ۹ بعدی ارائه میشد!

۲. کلمات مشابه دارای بردارهای مشابه هستند بعبارت دیگر ، embedding های تولید شده برای کلمات "دختر" و "شاهدخت" فاصله کمتری نسبت به "دختر" به "شاهزاده" دارد. در اینجا فاصله ای که از آن صحبت میکنیم همان فاصله اقلیدسی (Euclidean distance) است.

۳. ماتریس embedding جدید خیلی متراکم تر از نمونه قبلی است (فضای خالی کمتری دارد)، و ما میتوانستیم بدون آنکه محصور به افزایش ابعاد باشیم لغات بیشتری به لغت نامه اضافه کنیم. بعنوان مثال ما میتوانستیم کلمه "بچه" را با [۰,۵, ۰, ۰] نمایش دهیم! بدون اینکه نیازی به افزایش بُعد جدیدی باشد!

4. روابط بین کلمات ثبت و ضبط شده اند مثلاً حرکت از سمت پادشاه به ملکه، مثل حرکت از طرف پسر به دختر است و میتوان آنرا بصورت  $[1, 0, 0]$  نمایش داد.

## تعمیم به لغت نامه های بزرگتر

گام بعدی گسترش مثال ساده ۹ کلمه ای ما به یک لغت نامه کامل یا حداقل لغت نامه ای که شامل رایج ترین کلمات است میباشد. تشکیل بردارهای  $n$  بعدی که بتوانند معنی را به همان صورتی که در مثال ساده خود دیدیم بدست آورند، و کلمات مشابه دارای embedding های مشابه بوده و روابط بین کلمات حفظ شود کار ساده ای نیست.

انتساب دستی بردارها در عمل غیرممکن است، چرا که مدل های word embedding معمولی صدھا بُعد دارند و برخلاف مثال ساده ای که ما زدیم، در عمل هر بُعد به تنهایی قابل تفسیر نیست. به همین دلیل، الگوریتم های مختلفی توسعه داده شده اند، برخی از آنها قادرند مجموعه های عظیم متنی را دریافت کرده و مدل های بامعنایی تولید کنند. الگوریتم Word2vec شرکت گوگل، الگوریتم GloVe دانشگاه استنفورد و الگوریتم fasttext شرکت فیس بوک از معروف ترین و رایج ترین این الگوریتم ها هستند.

قبل از آنکه به توضیح این روشها بپردازیم، ابتدا به بیان خصائص embedding هایی که بخوبی آموخته دیده اند میپردازیم و در ادامه هر روش را بصورت خلاصه بیان میکنیم.

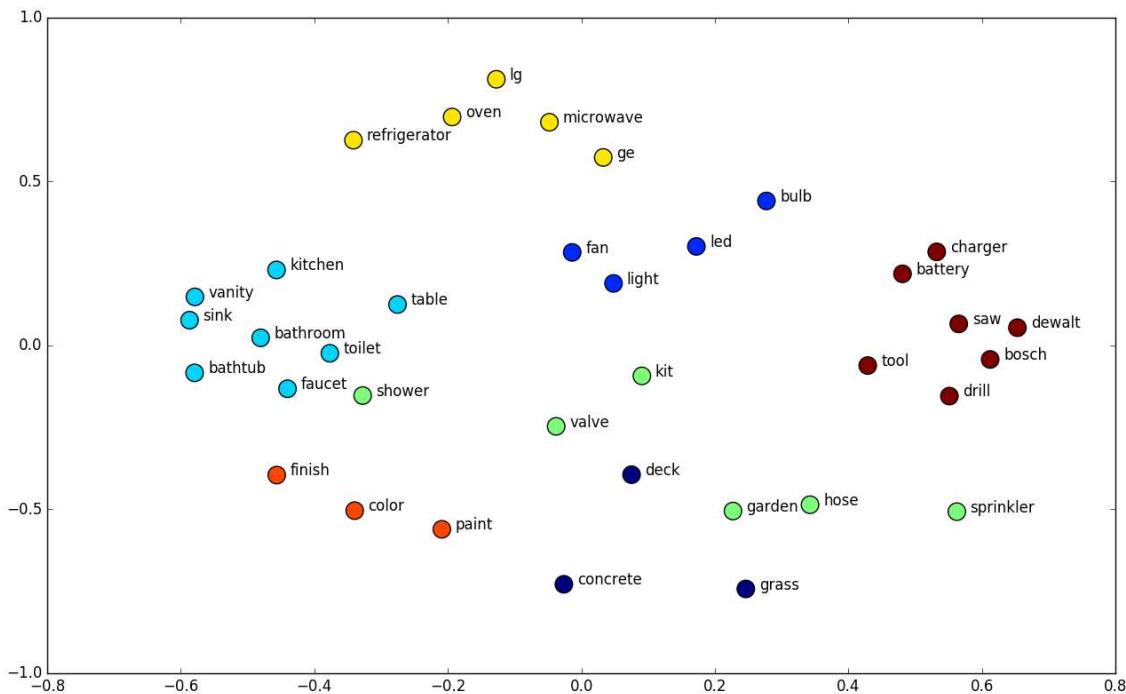
## خصائص Word Embedding

یک مجموعه کامل word embedding خصائص سودمند و فوق العاده ای از خود بروز میدهد، کلمات مشابه را تشخیص داده و بصورت طبیعی روابط بین کلمات را همانگونه که ما آنها را مورد استفاده قرار میدهیم ثبت و ضبط میکند.

### شباهت کلمات / متراffد ها

در فضای word embedding کلمات مشابه به مکان های مشابهی در فضای  $D - N$  بعدی همگرا میشوند. در مثال بالا، کلمه "خودرو"، "وسیله نقلیه" و "وانت" هر سه در مکان مشابهی در فضای embedding قرار میگیرند، بسیار دورتر از مکان کلمات بی ربطی مثل "ماه"، "فضا"، "درخت" و...

شباهتی که در اینجا از آن صحبت میکنیم را میتوان توسط فاصله اقلیدسی (فاصله واقعی بین نقاط در فضای  $D - N$  بعدی) و یا شباهت کسینوسی یا اصطلاحاً Cosine Similarity (زاویه بین دو بردار در فضای برداری) تعریف نمود.



مثالی از یک فضای word embedding 2 بعدی که در آن کلمات مشابه در مکان های مشابهی یافت میشوند. منع

## 1. Nearest neighbors

The Euclidean distance (or cosine similarity) between two word vectors provides an effective method for measuring the linguistic or semantic similarity of the corresponding words. Sometimes, the nearest neighbors according to this metric reveal rare but relevant words that lie outside an average human's vocabulary. For example, here are the closest words to the target word *frog*:

0. frog
1. frogs
2. toad
3. litoria
4. leptodactylidae
5. rana
6. lizard
7. eleutherodactylus



در تحقیقات دانشگاه استنفورد، نزدیک ترین همسایه ها، برای کلمه قورباغه، کلمات آشنا و غیرآشنا بدهت آمده توسط word embedding را نمایش میدهد. منع

این خصیصه را میتوان با استفاده از کتابخانه Gensim در زبان پایتون مشاهده کرد که در آن، نزدیک ترین کلمات به یک کلمه هدف در فضای برداری را میتوان با استفاده مجموعه ای از Word embedding ها براحتی استخراج نمود.

In [38]: `model.most_similar('simple')`

```
Out[38]: [('straightforward', 0.746016800403595),
 ('Simple', 0.7108174562454224),
 ('uncomplicated', 0.6297484636306763),
 ('simplest', 0.6171398162841797),
 ('easy', 0.5990298986434937),
 ('fairly_straightforward', 0.5893306732177734),
 ('deceptively_simple', 0.5743065476417542),
 ('simpler', 0.5537199378013611),
 ('simplistic', 0.551654040813446),
 ('disarmingly_simple', 0.5365327000617981)]
```

In [5]: `model.most_similar('man')`

```
Out[5]: [('woman', 0.7664012312889099),
 ('boy', 0.6824870705604553),
 ('teenager', 0.6586930155754089),
 ('teenage_girl', 0.6147903203964233),
 ('girl', 0.5921714305877686),
 ('suspected_purse_snatcher', 0.571636438369751),
 ('robber', 0.5585119724273682),
 ('Robbery_suspect', 0.5584409832954407),
 ('teen_ager', 0.5549197196960449),
 ('men', 0.5489762425422668)]
```

در مجموعه ای Word embedding های اموزش دیده شده مشاهده میکنیم که کلمات مشابه در یک مکان واحد یا نزدیک به یکدیگر نمایش داده میشوند

در کاربردهای مبتنی بر یادگیری ماشین، خصیصه شباهت Word embedding Word embedding را قادر میسازد تا با کلماتی که تا حال در فاز آموزش با آنها مواجه نشده اند بخوبی کار کند.

بهترین کاربردهای Word embedding از کلمات به مدلسازی پرداخته شود، مدل‌های مبتنی بر یادگیری ماشین از بردار کلمات برای اهداف پیش‌گویانه استفاده میکنند. اگر کلماتی که تا حال در زمان آموزش مشاهده نشده اند، اما در فضای Word embedding شباهت شده هستند، به مدل عرضه شوند، بردار کلمات بخوبی با این مدل به فعالیت خود ادامه خواهد داد. بعبارت دیگر اگر مدلی آموزش دیده تا بردارهای "گربه"، "وانت"، "جیپ" و "خودرو" را تشخیص دهد این مدل در مواجه با بردار "کامیون" بواسطه وجود شباهت در بین بردارها، همچنان بخوبی بکار خود ادامه خواهد داد.

به این شکل، استفاده از Word embedding بصورت ضمنی، اطلاعات اضافی را از مجموعه آموزشی Word embedding به برنامه کاربردی مورد نظر تزریق میکند. توانایی مدیریت عبارات دیده نشده (شامل املاهای غلط) برتری بسیار بزرگی برای روش‌های مبتنی بر Word embedding نسبت به روش‌های رایج قدیمی مثل TF-IDF، Bag-of-words است.

```
custom_word_vectors.most_similar('thx')
```

```
[('thanks', 0.4921847879886627),
 ('thankyou', 0.4289834201335907),
 ('thansk', 0.3909286856651306),
 ('tks', 0.3625342845916748),
 ('thanx', 0.36105877161026),
 ('thnaks', 0.3544262647628784),
 ('plz', 0.3251364529132843),
 ('thnx', 0.31662681698799133),
 ('cheers', 0.31641414761543274),
 ('thnks', 0.3139786422252655)]
```

از آنجایی که embedding ها اغلب بر روی متون واقعی آموزش می‌بینند، املاهای نادرست و عبارات عامیانه هم ثبت و ضبط می‌شوند و اغلب بردارهای بامعنایی به آنها انتساب داده می‌شود. توانایی فهمیدن املای نادرست بصورت صحیح مزیتی برای مدل‌های یادگیری ماشین است که ممکن است داده متنی غیرساخت یافته‌ای را از سمت کاربر دریافت کنند!

## روابط زبان شناختی

یکی از خصوصیت‌های جذاب word embedding های آموزش دیده این است که روابط بین کلمات در شیوه سخن گفتن معمولی نیز از طریق روابط خطی بین بردارها ثبت و ضبط می‌شود!

## استدلال با استفاده از بردار کلمات

”

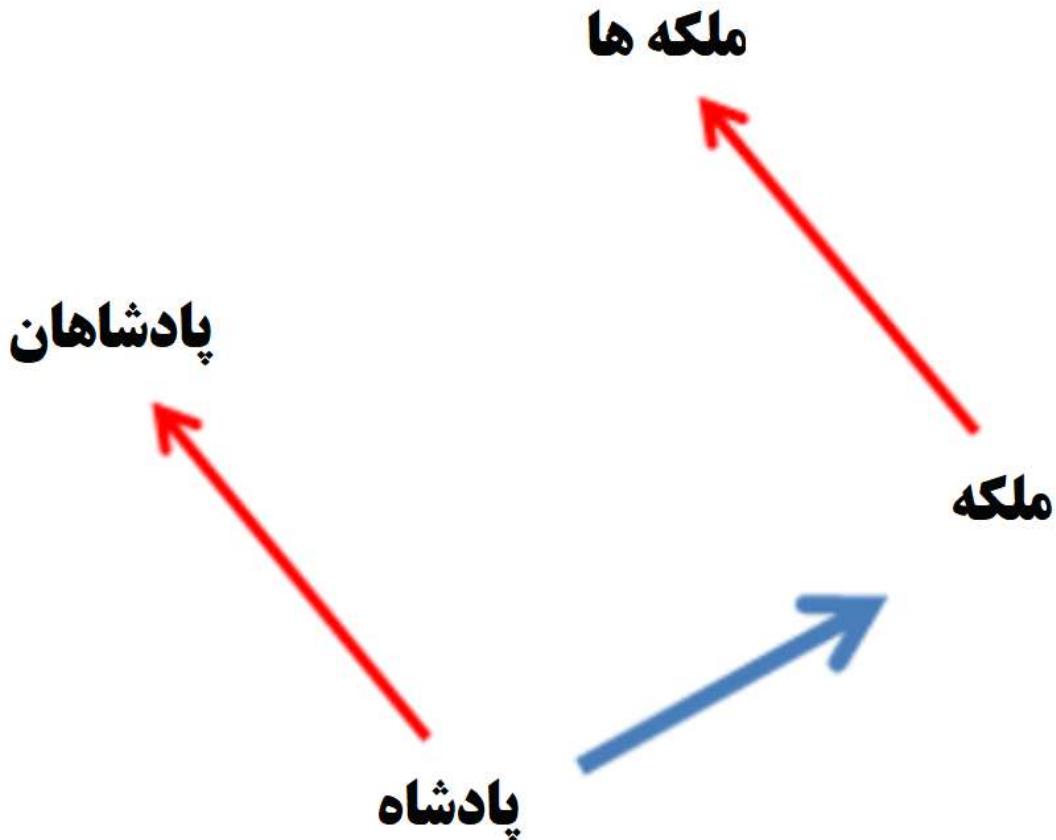
”...ما دریافتیم که بازنمایی کلمات فرآگرفته شده در حقیقت قواعد نحوی و معنایی را بشیوه ساده‌ای ثبت و ضبط میکنند. بطور ویژه، قواعد مورد بحث در قالب آفست های برداری ثابت (vector offsets) بین جفت کلماتی که رابطه خاصی با یکدیگر دارند مشاهده شده است. بعنوان مثال، اگر ما بردار مربوط به کلمه  $a$  را بصورت  $x_i$  در نظر بگیریم، و تمرکزمان را معطوف به رابطه بین صیغه مفرد/جمع کلمات مشاهده نماییم، آنوقت میکنیم که شاید شگفت‌انگیز‌تر این باشد که ما دریافتیم که این حالت برای روابط معنایی متنوعی نیز صادق است! ...“

این بردارها در پاسخ دادن به سوالات قیاسی به فرم ”الف نسبت به ب همانند ج است نسبت به ...“ بسیار خوب عمل میکنند. بعنوان مثال، سوال ”نسبت مرد به زن همانند نسبت عمو است به ....“ در اینجا توسط یک روش مبتنی بر آفست بردار ساده براحتی با استفاده از محاسبه فاصله کسینوسی بسادگی کلمه ”زن عمو“، بدست می‌آید.

بعنوان مثالی دیگر، در اینجا آفست های برداری مربوط به سه جفت کلمه که نمایانگر رابطه ”جنسیت“ هستند را مشاهده میکنیم:



در اینجا هم رابطه مفرد/جمع را مشاهده میکنیم:

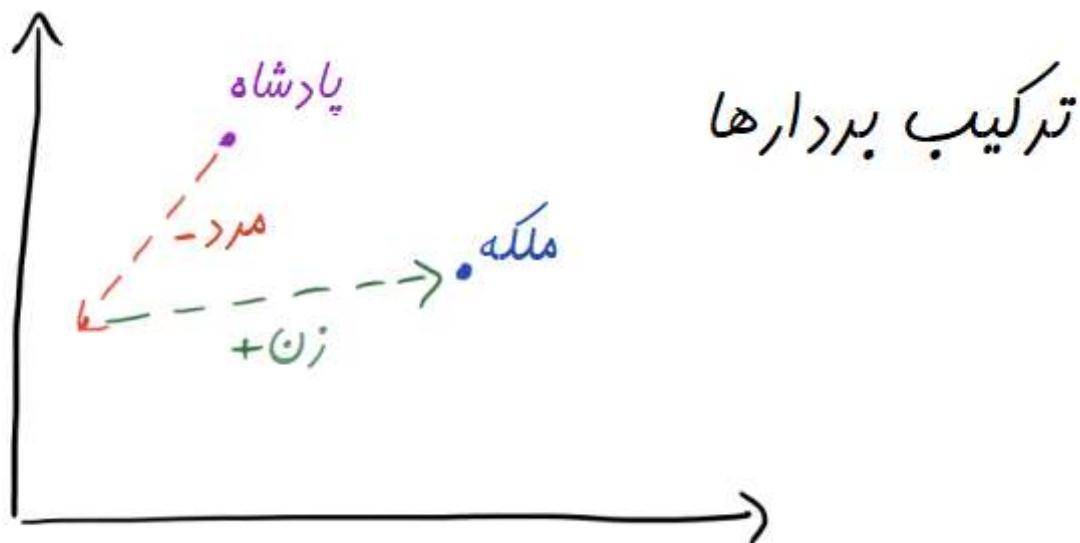
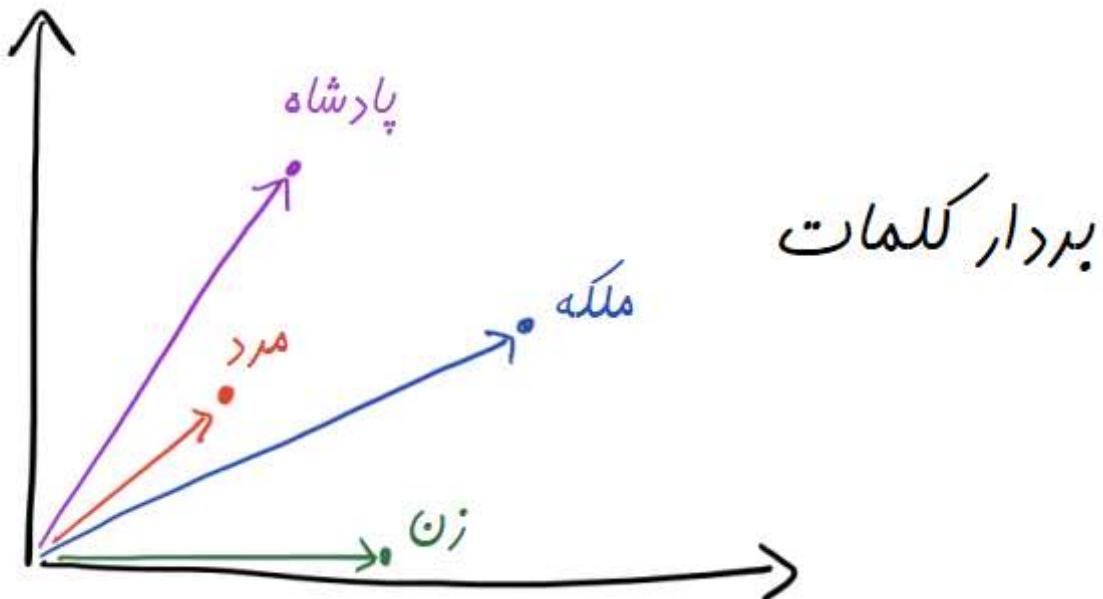


این نوع از ترکیب بردارها با یکدیگر، به ما اجازه میدهد تا بتوانیم به سوالاتی مثل "پادشاه - مرد + زن = ؟" پاسخ داده و بعنوان مثال به جواب "ملکه" برای این سوال برسیم. چنین نتایجی واقعاً شگفت انگیزند خصوصاً زمانی که بیاد می‌آوریم تمام دانشی که در اینجا با آن مواجه هستیم صرفاً با نگاه به تعداد زیادی از کلمات در زمینه محتوایی خاص خود بدست آمده است بدون آنکه هیچ اطلاعات دیگری مبنی بر معنای این کلمات به مدل ارائه شود!

”

مساله‌ای که تا حدودی شگفت‌آور است این است که ما دریافتیم شباهت بین بازنمایی‌های کلمات فراتر از قواعد نحوی است. با استفاده از یک تکنیک آفست کلمه، که در آن عملیات جبری ساده‌ای بر روی بردار کلمات اعمال گردید، مشاهده گردید که بعنوان مثال، بردار(پادشاه) - بردار("مرد") + بردار("زن") برداری را نتیجه میدهد که بسیار به بازنمایی بردار کلمه "ملکه" نزدیک است!

بردارهای مربوط به پادشاه، مرد، ملکه و زن :



در اینجا نتایج بیشتری که توسط همین روش بدست آمده است را مشاهده میکنید:

Table 8: Examples of the word pair relationships, using the best word vectors from Table 4 (Skip-gram model trained on 783M words with 300 dimensionality).

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

همچنین ما میتوانیم از عملیات جمع درایه به درایه بر روی بردارها استفاده کنیم و اینگونه به سوالاتی نظیر چک + واحد پولی و... تنها از طریق مشاهده نزدیک ترین بردار به بردار نتیجه شده خود پاسخ دهیم!:

Czech + currency	Vietnam + capital	German + airlines	Russian + river	French + actress
koruna	Hanoi	airline Lufthansa	Moscow	Juliette Binoche
Czech crown	Ho Chi Minh City	carrier Lufthansa	Volga River	Vanessa Paradis
Polish zolty	Viet Nam	flag carrier Lufthansa	upriver	Charlotte Gainsbourg
CTK	Vietnamese	Lufthansa	Russia	Cecile De

Table 5: Vector compositionality using element-wise addition. Four closest tokens to the sum of two vectors are shown, using the best Skip-gram model.

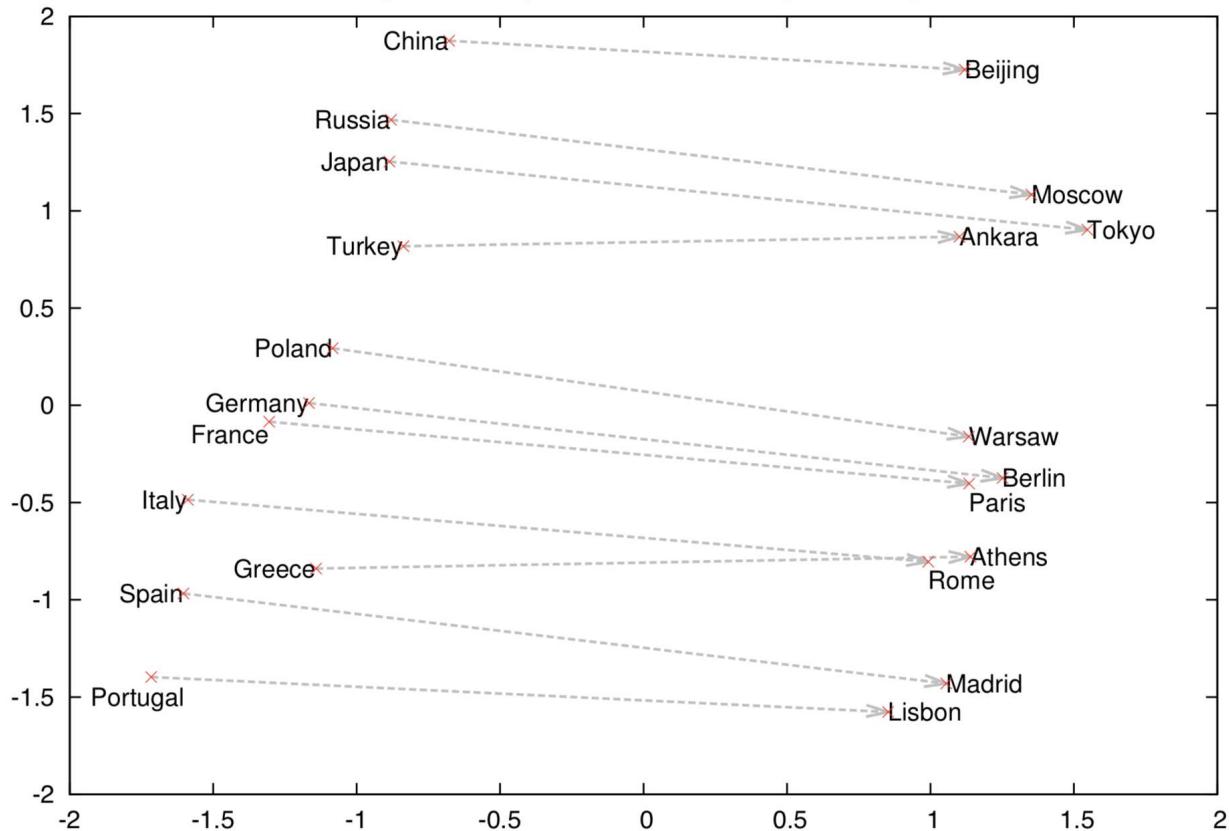
## ۹۹

بردار کلمات با چنین روابط معنایی میتوانند جهت بهبود بسیاری از کاربردهای حوزه پردازش زبان طبیعی، نظیر ترجمه ماشینی، بازیابی اطلاعات و سامانه های پرسش و پاسخ مورد استفاده قرار گیرند و ممکن است کاربردهایی که هنوز اختراع نشده اند را در آینده موجب گردند.

---

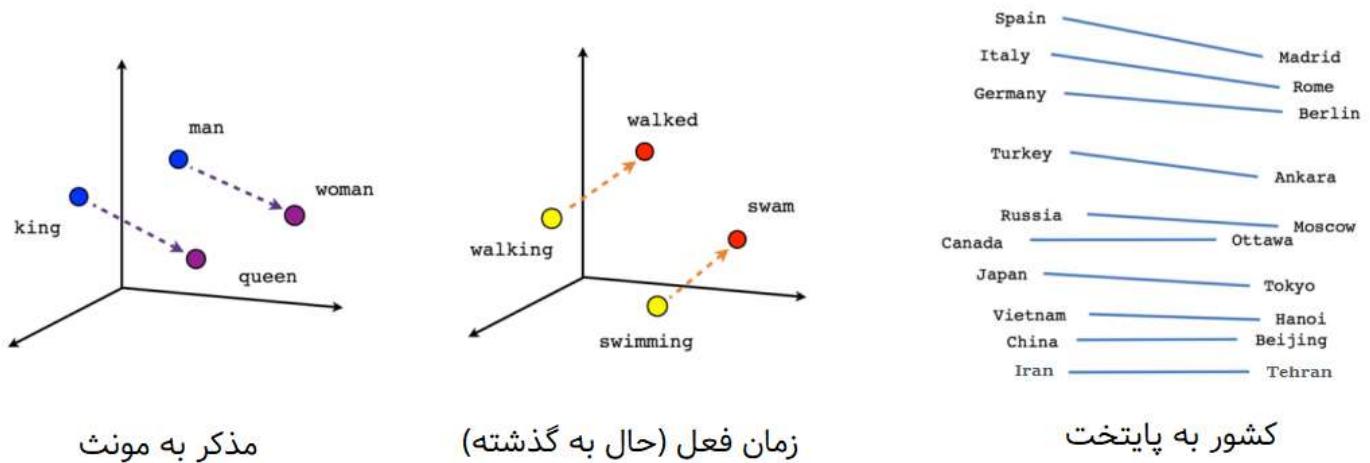
در مقاله اصلی word embedding ، مثالهای جالبی از روابط موجود برای مواردی مثل "پایتخت کشور" ، "رودخانه اصلی در" ، زمان فعل، و... سایر الگوهای جالب آورده شده است. درک و توجه به این موضوع که این روابط بطور صریح در حین فرایند آموزش به مدل ارائه نشده بلکه از طریق استفاده از زبان در دیتاست آموزشی "کشف" شده اند بسیار مهم است

## Country and Capital Vectors Projected by PCA



یک پروژکشن ۲ بعدی PCA از word embedding ها که رابطه خطی پایخت را توسط فرایند آموزش word embedding ثبت کرده است نمایش میدهد. به نقل از نویسندهای مقاله "این شکل توانایی مدل جهت سازماندهی خودکار مفاهیم و فراگرفتن روابط ضمنی بین آنها را نشان میدهد. در حین آموزشی هیچ اطلاعات با نظرارتی در مورد اینکه پایخت به چه معنا میباشد فراهم نگردیده است!"

در زیر مثال دیگری از رابطه میان کلمات را مشاهده میکنید که روابطی مربوط به جنسیت (حرکت از سمت جنسیت نر به ماده) و یا زمان فعل (حال به گذشته) را نشان میدهد:



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

روابط خطی بین کلمات در فضای embedding موجود ظهور جبر کلمه ای غیرمعمولی میشود که به کلمات اجازه جمع و تفریق داده و نتایج منطقی نیز بدنبال دارد! بعنوان مثال، در یک مدل word embedding بخوبی تعریف شده، محاسباتی همانند آنچه در ادامه میبینید واقعا کار میکنند و نتایج صحیح و منطقی ارائه میدهند (در مثال زیر  $[X]$  به معنای بردار کلمه  $X$  است. پس  $[man]$  یعنی بردار embedding مربوط به کلمه  $man$  میباشد)

”

$$\text{king}]] - [[\text{man}]] + [[\text{woman}]] = [[\text{queen}]] [[\text{Paris}]] - [[\text{France}]] + [[\text{Germany}]] = ]]$$

$$[[[[\text{Berlin}]]$$

مثال دوم :

```
In [25]: model.similar_by_vector(model['Obama'] - model['USA'] + model['France'])

Out[25]: [('Sarkozy', 0.704483687877655),
          ('Obama', 0.7015002369880676),
          ('President_Nicolas_Sarkozy', 0.642586350440979),
```

```
In [15]: model.similar_by_vector(model['Dublin'] - model['Ireland'] + model['France'])

Out[15]: [('Paris', 0.740702748298645),
          ('France', 0.6797398924827576),
          ('Issy_les_Moulineaux', 0.6246635317802429),
```

ریاضیات مربوط به بردارها را میتوان بر روی بردار کلمات اعمال کرد و روابط ثبت شده توسط مدل در حین فرایند آموزش، را مشاهده کرد.

## مبانی الگوریتم های آموزش Word embedding محتوای کلمه

در زمان آموزش word embedding برای یک لغت نامه بزرگ، تمرکز بر روی بهینه سازی embedding هاست بگونه ای که معنای اصلی و روابط بین کلمات حفظ شود. این ایده اولین بار توسط جان روپرت فرث که بعنوان زبان شناس بر روی الگوهای زبانی در دهه ۱۹۵۰ کار میکرد ارائه شد.

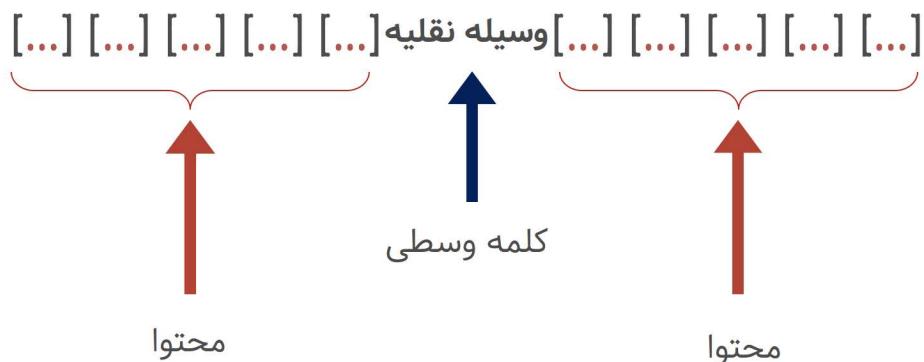
”

شما باید یک کلمه را از اطرافیانش بشناسید!

فرث به اصلی اشاره دارد که در آن معنای یک کلمه آن چیزی است که توسط استفاده آن به همراه سایر کلمات بدست می‌آید  
چرا که کلمات دربرگیرنده، برای هر کلمه‌ای برای دریافت معنای آن کلمه مفیدند!



در الگوریتم‌های *word embedding*، کلمه "وسط" کلمه مورد تمرکز و کلمات "محتوایی" کلماتی‌اند که آنرا در استفاده معمولی در برابر گرفته‌اند. (در برگیرنده)



ایده اصلی آموزش *word embedding* این است که کلمات مشابه عوموماً توسط کلمات "محتوایی" یکسان در کاربردهای معمولی احاطه می‌شوند.

فرض کنید ما کلمه‌ای را بعنوان کلمه "وسطی" انتخاب کنیم سپس به کلمات "معمولی" که ممکن است این کلمه را در یک استفاده زبانی رایج احاطه کنند نگاه کنیم. دیاگرام زیر کلمات محتوایی متحمل برای کلمات "وسطی" مورد نظر را نشان میدهد. در این مثال، کلمات محتوایی برای "وانت" عومولاً "تایر، "جاده، "مسافرت" و... هستند. کلمات محتوایی برای یک کلمه شبیه به "وانت" مثل "خودرو" نیز مشابه هستند! . به همین شکل، کلمات محتوایی برای کلمه نامشابه‌ای مثل "ماه"، نتایج کاملاً متفاوتی خواهد داشت.

دایناسور

ماه

اداره	برج
لندن	لامپ
جاده	رانندگی
موتور	وسیله نقلیه
فرمان گیری	مدل
صندلی	

کلمات محتوایی، "وسیله نقلیه"، انتظار می‌رود که این کلمات مشابه کلمات محتوایی، برای "وانت" و "خودرو" باشند.

انتظار مشاهده این کلمات در محتوای "انت" نمی‌رود

کلمات مورد انتظار در محتوای "وانت"

مشاهده این کلمات در محتوای وانت مورد انتظار است

دایناسور

لندن

اداره

فرمان گیری

صندلی

مدل

رانندگی

جاده

موتور

وانت

رانندگی

مدل

پرچ

صندلی

موتور

وانت

وانت

دایناسور

ماه

[...][...][...][...][...][...]**خودرو**[...][...][...][...][...]

داناسور

81

اداره	برج
لندن	لامپ
فرمان گیری	مدل
موتور	رانندگی
جاده	خودرو
صندوقی	

کلمات محتوایی برای خودرو ممکن است شامل "جاده"، "فرمان گیری" و... باشد و نه ماه و پا پرج!

[...] [...] [...] [...] [...] [...] [...] [...] [...] [...] [...] [...] [...] [...] [...]]



کلمات محتوایی برای "ماه" در یک متن ساده بطور اساسی با کلماتی که برای "خودرو"، وسیله نقلیه و "وانت" ارائه میشوند متفاوت است.

## فراگیری بردار کلمات

میکولوف و همکارانش اولین کسانی نبودند که از بازنمایی های برداری پیوسته کلمات استفاده کردند، او و همکارانش اما اولین کسانی بودند که چگونگی کاهش پیچیدگی محاسبات مربوط به یادگیری چنین بازنمایی ها را نشان دادند. فعالیت ایشان موجب شد تا بتوان بردار کلمات با ابعاد بالایی را با استفاده از حجم عظیمی از داده ها فرا گرفت چیزی که سابقا ممکن نبود. بعنوان مثال، همانطور که در مقاله اشاره شده است:

“

”...ما از مجموعه متن اخبار گوگل برای آموزش بردار کلمات بهره بردیم. این مجموعه متن شامل بیش از ۶ میلیارد توکن است که البته ما اندازه لغت نامه خود را به ۱ میلیون کلمه پرکاربرد محدود نمودیم...”

دو روش اصلی برای آموزش مدل‌های word embedding وجود دارد:

۱. **مدل های معنایی توزیع شده** (Distributed Semantic Models) : این مدلها مبتنی بر هم رخداد/نزدیکی کلمات به یکدیگر در یک مجموعه متنی بزرگ میباشند. یک ماتریس هم رخداد (co-occurrence matrix) برای مجموعه متنی بزرگی تشکیل میشود (یک ماتریس  $N$  با مقادیری که هر کدام نمایانگر احتمال رخداد کلمات در کنار همیگر است)، و این ماتریس تجزیه شده (توسط SVD/PCA و یا روش‌های مشابه) تا یک ماتریس بردار کلمه را تشکیل دهد. تکنیک های مدل‌سازی word embedding به این شیوه به روش‌های مبتنی بر شمارش یا همان Count approaches یا Count based معروف اند.

۲. **مدل‌های شبکه عصبی** (Neural Network Models) : روش‌های مبتنی بر شبکه های عصی عموماً "روش های مبتنی بر پیش بینی" یا بهتر بگوییم "پیش بینی محور" هستند که در آنها مدلها برای پیش بینی کلمات محتوایی با استفاده از کلمات وسطی، یا بر عکس (پیش بینی کلمه وسط با استفاده از مجموعه ای از کلمات محتوایی) ساخته میشوند.

مساله ای که در رابطه با مدل های مبتنی بر شبکه عصبی وجود دارد بحث پیچیدگی زیاد این نوع شبکه هاست. این پیچیدگی در مدل های شبکه عصبی، (چه پیش خور و چه مدل های بازگشتی) از لایه های مخفی غیرخطی نشات میگیرد:

”

...هرچند این همان خصیصه ای است که شبکه های عصبی را بسیار جذاب میکند، اما ما تصمیم گرفتیم مدل های ساده تری که قادر به بازنمایی داده ها با همان دقیقت شبکه های عصبی نیستند، اما در عین حال اجازه آموزش بهینه بر روی انبوهای ازداده ها را فراهم میکنند، مورد بررسی و کنکاش قرار دهیم.

به همین جهت دو معماری جدید ارائه شد: مدل Continuous Skip-gram و مدل Continuous Bag of words

نکته:

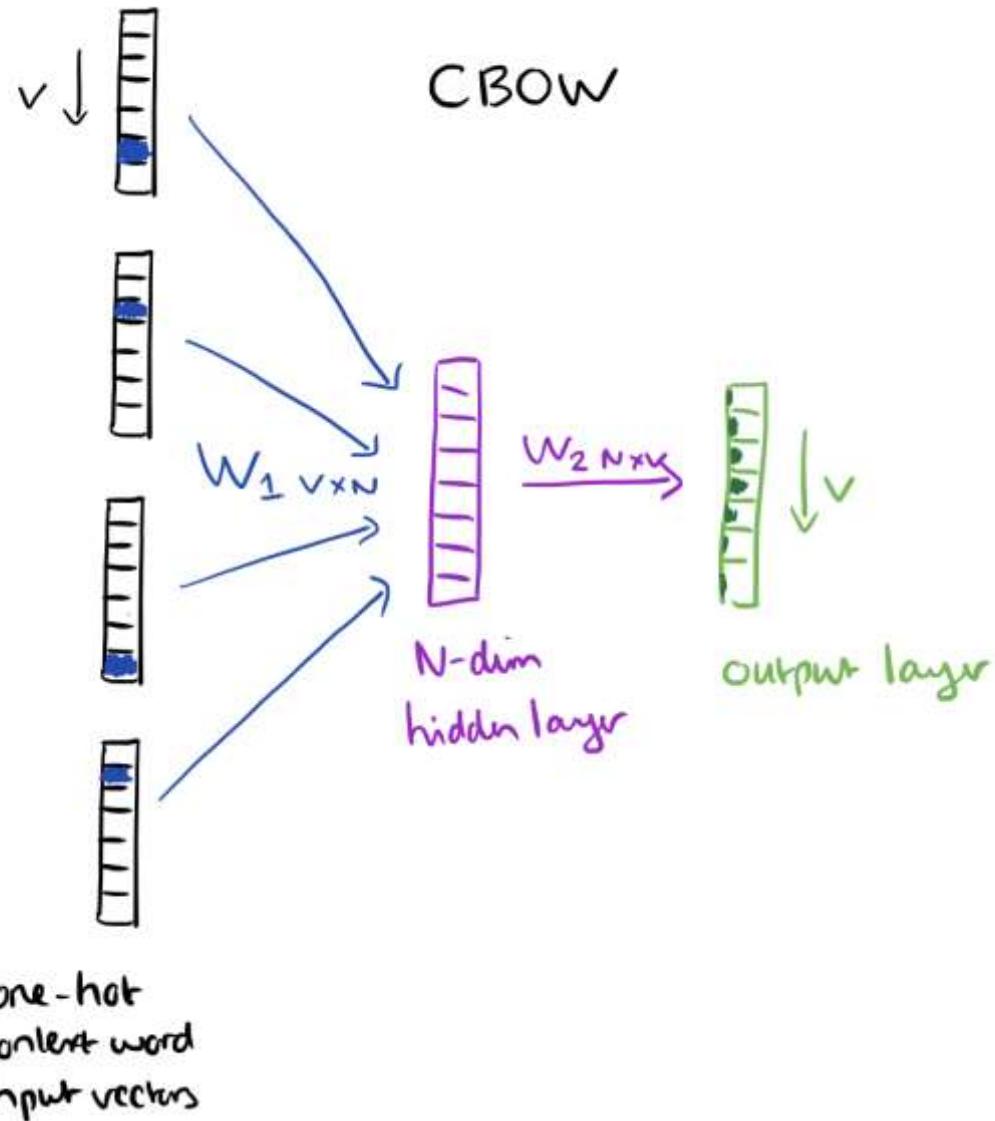
روش های پیش بینی محور، عموما از نقطه نظر کارایی روش های مبتنی بر شمارش را پشت سر میگذارند و بهتر از آنها عمل میکنند. بعضی از رایج ترین الگوریتم های word embedding ، مثل Skip-Gram ، Continuous Bag of Words (CBOW) و Word2Vec همگی روشهای پیش بینی محور هستند.

## مدل Continuous Bag of Words

اجازه دهید با مدل CBOW شروع کنیم. متئی مثل "من هر هفته با وسیله نقلیه عمومی به دانشگاه میروم." را در نظر بگیرید. فرض کنید یک پنجره لغزان بر روی این متن وجود دارد و کلمه ای که روی آن تمرکز شده، همان کلمه وسطی باشد که ۴ کلمه قبل و بعد از آن وجود دارد. :



کلمات محتوایی لایه ورودی را تشکیل میدهند و هر کلمه نیز، در قالب یک بردار one-hot ارائه میشود، اندازه لغت نامه را  $V$  در نظر میگیریم. بنابر این، این بردارها در نتیجه  $V$  بُعدی خواهند بود و تنها یک درایه آنها مقداری برابر با ۱ خواهد داشت و مابقی درایه ها صفر خواهند بود. در اینجا تنها یک لایه مخفی و یک لایه خروجی وجود دارد.



هدف آموزش، بیشینه سازی احتمال شرطی مشاهده کلمه خروجی واقعی (کلمه وسطی) به ازای کلمات محتوایی ورودی، با توجه به وزن انهاست. در مثال ما، به ازای دریافت ورودی ("من", "هر", "با", "هفتہ", "عمومی", "به", "دانشگاه", "میروم") ما میخواهیم احتمال دریافت "وسیله نقلیه" را بعنوان خروجی بیشینه کنیم.

از آنجایی که بردارهای ورودی ما بصورت one-hot هستند، ضرب یک بردار ورودی در ماتریس وزن  $W_1$  به مثابه انتخاب سطری از  $W_1$  خواهد بود:

$$\begin{array}{c}
 \text{input} \quad \quad \quad W_1 \quad \quad \quad \text{hidden} \\
 1 \times v \quad \quad \quad v \times N \quad \quad \quad 1 \times N
 \end{array}$$

$$[0 \ 1 \ 0] \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{bmatrix} = \begin{bmatrix} e & f & g & h \end{bmatrix}$$

$W_1$

به ازای  $C$  بردار کلمه ورودی،تابع فعالسازی لایه مخفی  $h$  خیلی ساده اقدام به جمع "سطر های فعال" در ماتریس وزن  $W_1$  میکند (سطرهایی که در ضرب با ورودی انتخاب شدند) و سپس نتیجه را بر  $C$  تقسیم کرده تا میانگین آنها را بدست آید.

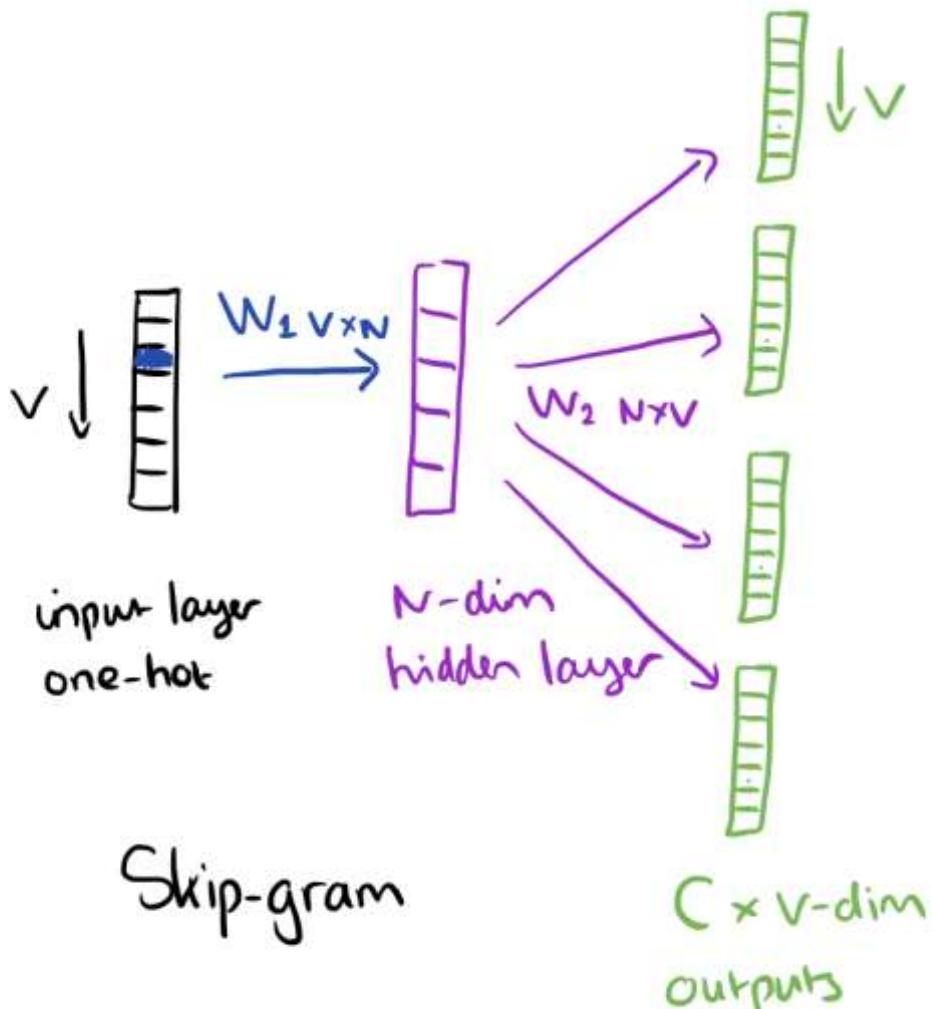
”

این به این معناست که تابع (فعالسازی) پیوند واحدهای لایه مخفی، خطی هستند! (یعنی، بصورت مستقیم جمع وزن دار ورودی را به لایه بعدی ارسال میکنند!)

بین لایه مخفی و لایه خروجی، ماتریس وزن دومی بنام  $W_2$  وجود دارد که از آن برای محاسبه امتیازی برای تمامی کلمات موجود در لغت نامه استفاده میشود و بعد از آن میتوان از سافتمنکس برای کسب توزیع خلفی (posterior distribution) کلمات بهره برد.

## Skip-gram مدل

مدل skip-gram بر عکس مدل CBOW است. این مدل بگونه ای ساخته شده است تا کلمه وسطی را بعنوان یک بردار ورودی دریافت کرده و کلمات محتوایی هدف را در لایه خروجی تولید کند.



تابع فعالسازی لایه مخفی تنها در کپی کردن سطرهای متناظر از ماتریس وزن ۱ (خطی) خلاصه میشود. در لایه خروجی، ما بجای اینکه یک توزیع چند جمله ای داشته باشیم!  $C$  توزیع چندجمله ای خواهیم داشت. هدف آموزش کمینه سازی مجموع خطای پیش بینی همه کلمات محتوایی در لایه خروجی است. در مثال ما ورودی "وسیله نقلیه" بوده و ما امیدواریم ("من", "هر", "هفته", "با", "عمومی", "به", "دانشگاه", "میروم") را در خروجی ببینیم!

## بهینه سازی

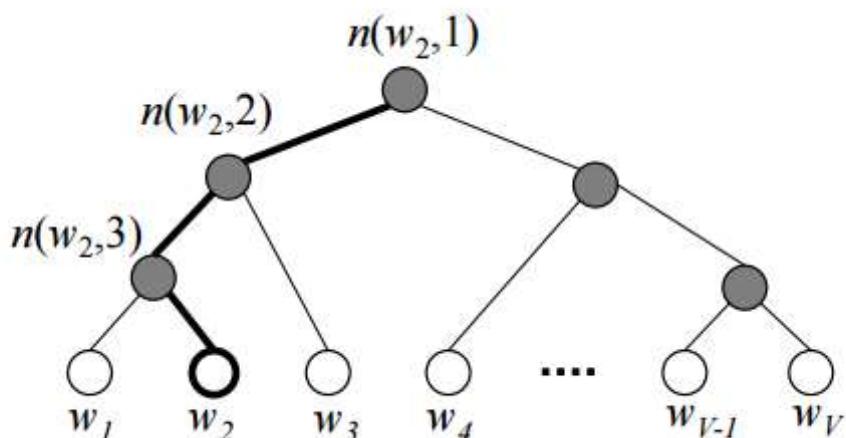
بروز کردن تمامی بردار کلمات خروجی به ازای تمامی کلمات در یک فرایند آموزش عملیاتی بشدت هزینه بر است :

”

به منظور حل این مساله، یک راه حل شهودی محدود سازی تعداد بردارهای خروجی است که ممکن است در هر فرایند آموزش بروزشوند. یک روش زیبا برای دست یابی به این مساله، استفاده از سافتمنکس سلسله مراتبی است. روش دیگر از طریق نمونه برداری محقق میشود.

## Softmax سلسله مراتبی

سافتمنکس سلسله مراتبی، از یک درخت دودویی برای بازنمایی تمامی کلمات یک لغت نامه بهره میبرد. کلمات خود، برگهای درخت هستند و برای هر برگ، مسیری یکتا از ریشه به آن برگ وجود دارد. این مسیر همان چیزی است که برای تقریب احتمال کلمه ای که آن برگ معرف آن است استفاده میشود. ما احتمال را بصورت احتمال پیاده روی تصادفی با شروع از ریشه و رسیدن به برگ مورد نظر تعریف میکنیم.



شمایلی از یک درخت دودویی سافتمنکس سلسله مراتبی. برگهای (گره های) سفید رنگ همان کلمات در لغتنامه اند. گره های داخلی خاکستری حامل اطلاعات در رابطه با احتمالات تا گره های فرزند میباشند. یک مسیر که از ریشه اغاز و به برگ  $w_i$  ختم میگردد.  $n(w_i, j)$  نشانگر  $j$  امین گره در این مسیر است.

”

مزیت اصلی در اینجا، این است که بجای ارزیابی  $V$  گره خروجی در شبکه عصبی به منظور دست یافتن به توزیع احتمالاتی، تنها نیاز به ارزیابی حدودا  $\log_2(V)$  کلمه است... برای کار خود، ما از یک درخت هافمن دودویی، که کدهای کوتاهی به کلمات مکرر منتبه میکند و در نتیجه موجب تسريع فرآیند آموزش میشود، استفاده میکنیم.

## نمونه برداری منفی (NEG)

نمونه برداری منفی به بیان ساده، همان بروز رسانی "نمونه ای" از کلمات خروجی در هر تکرار است. کلمه خروجی هدف در اینجا باید در نمونه برداری مورد نظر، حفظ شده و بروز رسانی گردد. در این نمونه برداری علاوه بر کلمه مورد نظر، چند کلمه (غیر-هدف) نیز بعنوان نمونه های منفی قرار داده میشود و به همین دلیل به این شیوه، روش نمونه برداری منفی گفته میشود. "یک توزیع احتمالاتی برای این فرآیند نمونه برداری مورد نیاز بوده که میتوان بصورت دلخواه آنرا انتخاب کرد... فرد میتواند یک توزیع خوب را بصورت تجربی انتخاب کند."

میکولوف و همکارانش همچنین از یک روش زیرنمونه برداری ساده جهت مقابله با عدم توازن موجود بین کلمات متداول و نادر در مجموعه آموزشی استفاده کردند.(بعنوان مثال در، و، را و... ارزش اطلاعاتی کمتری نسبت به کلمات نادر فراهم میکنند). در طرح پیشنهادی آنها، هر کلمه در مجموعه آموزشی، با توزیع  $P(W_i)$  حذف میشود :

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}}$$

در اینجا  $f(w_i)$  نرخ تکرار کلمه  $w_i$  است.  $t$  نیز در اینجا مقدار آستانه انتخاب شده است که عموماً دارای مقداری برابر با  $10^{-5}$  است.

+ نکته

## الگوریتم های رایج Word embedding

یکی از رایج ترین روش های آموزش Word2vec است که توسط تیمی به سرپرستی میکولوف در گوگل توسعه پیدا کرد. این روش در زمان آموزش در اصل از تکنیک های Skip-gram و CBOW بطور همزمان با هم استفاده میکند. روش های آموزش دیگری نیز وجود دارند که از آنها میتوان به بردارهای سراسری برای بازنمایی کلمات یا همان GloVe و یا الگوریتم fasttext اشاره کرد که الگوریتم اول توسط تیم پردازش زبان طبیعی دانشگاه استنفورد و الگوریتم دوم توسعه فیس بوک توسعه داده شده است.

# Distributed Representations of Words and Phrases and their Compositionality

<b>Tomas Mikolov</b> Google Inc. Mountain View <a href="mailto:mikolov@google.com">mikolov@google.com</a>	<b>Ilya Sutskever</b> Google Inc. Mountain View <a href="mailto:ilyasut@google.com">ilyasut@google.com</a>	<b>Kai Chen</b> Google Inc. Mountain View <a href="mailto:kai@google.com">kai@google.com</a>
<b>Greg Corrado</b> Google Inc. Mountain View <a href="mailto:gcorrado@google.com">gcorrado@google.com</a>	<b>Jeffrey Dean</b> Google Inc. Mountain View <a href="mailto:jeff@google.com">jeff@google.com</a>	

مقاله اصلی (2013) Word2vec که توسط تیم گوگل ارائه شد. Word یک الگوریتم آموزش بهینه برای بدست آوردن word embedding است که باعث پیشرفت اساسی در حوزه پردازش زبان طبیعی شد.

کیفیت مدل‌های word embedding مختلف را میتوان از طریق سنجش روابط بین یک مجموعه کلمات شناخته شده مورد ارزیابی قرار داد. میکولوف و همکاران روشی را توسعه دادند که از طریق آن میتوان مجموعه‌ای از بردارها را ارزیابی کرد. دقت مدل و سودمنی وابسته به داده آموزشی مورد استفاده، انتخاب فرآپارامترهای مرتبط با الگوریتم آموزش، الگوریتم مورد استفاده، و ابعاد مدل است.

## استفاده از Word embedding در پایتون

گزینه‌های مختلفی برای استفاده از word embedding در پروژه شما در پایتون وجود دارد. دو کتابخانه اصلی که عموماً از آن استفاده می‌شود Spacy- که یک کتابخانه پردازش زبان طبیعی سرعت بالاست- و Gensim که کتابخانه‌ای مرکز بر روی کاربردهای مدلسازی مبتنی بر موضوع است میباشدند.

مجموعه‌های از پیش آموزش دیده word embedding ، که با استفاده از تمامی محتویات ویکی پیدیا، یا تاریخچه اخبار گوگل ایجاد شده اند را میتوان بصورت مستقیم دانلود کرده و با مدلها و سامانه‌های خود ادغام کنید. علاوه بر این، شما میتوانید مدل‌های خود را با استفاده از زبان پایتون و کتابخانه Genesis آموزش دهید. یک مدل سفارشی میتواند مدل‌های استاندارد را در دامنه‌ها و لغت نامه‌های خاص پشت سر بگذارد.

## کاربردهای word embedding

word embedding ها در گستره زیادی از عملیات‌های مرتبط با پردازش زبان طبیعی مورد استفاده قرار میگیرند

- علاوه بر تکنیک‌های مدلسازی نظری شبکه‌های عصبی مصنوعی، word embeddings بطرز شگفت آوری دقت دسته بندی متن را در بسیاری از دامنه‌های نظری، سرویس مشتری، تشخیص هرزنامه، دسته بندی اسناد و... را بهبود داده است.

- aligning single-language word embedding برای بهبود کیفیت ترجمه زبانهای مختلف از طریق embedding با استفاده از یک ماتریس تبدیل مورد استفاده قرار گرفته است. برای اطلاعات بیشتر میتوانید به اینجا

نگاهی بیندازید.

- بردار کلمات همچنین جهت بهبود دقت در کاربردهای مبتنی بر جستجوی اسناد و بازیابی اطلاعات، که در آن تنظیمات جستجو نیازمند جستجوی کلید واژه‌های دقیق نیست و نسبت به املا نیز حساسیت ندارد استفاده شده‌اند.

لازم به ذکر است کاربردهای مبتنی بر بردار کلمات و بطور خاص Word2vec محدود به تجزیه جملات نبوده و چیزی فراتر از آن است. از آن میتوان بر روی گستره زیادی از داده‌ها همانند، ژن، سورس کد، پسندیدهای شبکه اجتماعی، playlist، ها، گراف‌های شبکه‌های اجتماعی و سری‌های سمبولیک و... که میتوانند حاوی الگوهایی باشد، استفاده کرد.

این داده‌ها همانند متن دارای ماهیت یا عبارت بهتر حالات گستته اند، و از این جهت ما میتوانیم همانند آنچه با متن انجام دادیم با آنها نیز انجام دهیم. بعنوان مثال ما میتوانیم به گذار احتمالات بین دو حالت دقت کنیم و ببینیم احتمال رخ دادن هریک به چه صورت است. بنابر این gene2vec، like2vec، follower2vec، ... مواردی از این دست همه قابل انجام هستند. این دید، بشما کمک خواهد کرد تا چگونگی ایجاد embedding‌های عصبی برای انواع مختلف داده را بخوبی درک کنید.

نکته :

بعضی‌ها قائل به این هستند که Word2Vec بعنوان یک الگوریتم معرفی نشده و بر عکس بعنوان یک نرم افزار به اون نگاه بشه چرا که در درون خودش از دو الگوریتم مختلف (که بحثش رو بالا کردیم) بهره میبره. اگر در متونی با لفظ برنامه word2vec یا نرم افزار word2vec مواجه شدید به این علت هست.

## مطالعه بیشتر(منابع و مأخذ) :

شما میتوانید از پیوند‌های زیر برای تثبیت آنچه در اینجا یادگرفتید استفاده کنید :

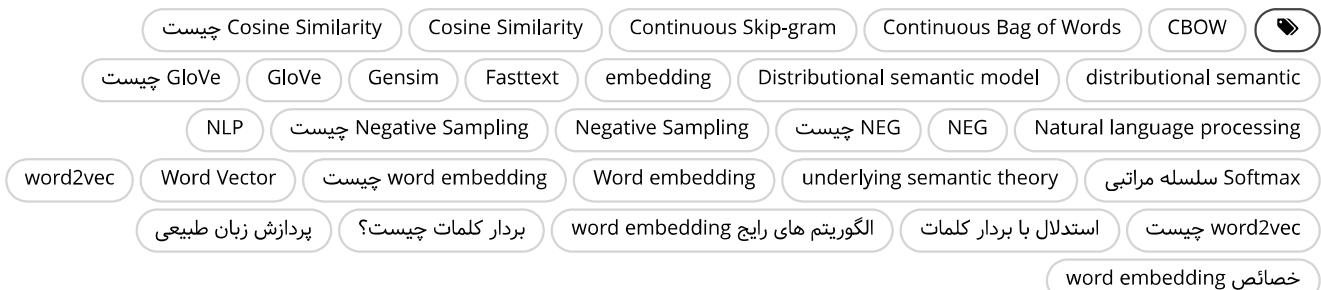
- .Coursera Deep Learning course video on Word Embeddings •
- .Google Tensorflow Tutorial on Word Embeddings •
- .Excellent break down for Skip-Gram algorithm •
- Chris McCormick – The Skip-Gram Model •
- The amazing power of word vectors” – Adrian Colye” •
- Get Busy with Word Embeddings – An Introduction •
- (مجموعه ۵ بخشی) On word embeddings – Part 1 (خیلی خوبه) •

مقالات :

:The papers are

- Efficient Estimation of Word Representations in Vector Space – Mikolov et al. 2013 •
- Distributed Representations of Words and Phrases and their Compositionality – Mikolov et al. 2013 •
- Linguistic Regularities in Continuous Space Word Representations – Mikolov et al. 2013 •
- word2vec Parameter Learning Explained – Rong 2014 •
- word2vec Explained: Deriving Mikolov et al's Negative Sampling Word-Embedding Method – Goldberg and Levy 2014 •

در بخش بعد اگر فرصت شد انشاءالله شیوه کار با word2vec و embedding ها توسط چند کتابخونه معتبر انجام میشه.



## سید حسین حسن پور متی کلابی



موسس و مدیر سایت. اطلاعات در مورد فعالیت های کاری و تحصیلی : linkedIn . برای ارتباط از بخش تماس با ما یا در باره من استفاده کنید.

## 10 نظرات

2 سال پیش

Mohsen می گوید

جالب بود  
مرسى

2 سال پیش

الهام پارسايی مهر می گوید

بسیار عالی و شفاف توضیح دادید. من مقالات انگلیسی زیادی در مورد word embedding خوندم و به اندازه ۷۰ درصد درکی که الان کسب کردم، رسیده بودم. واقعا خیلی شفاف و سلیس به زبان ساده توضیح دادید. بخصوص مثالهایی که بیان کرده بودید.  
واقعا خدا قوت میگم.  
امیدوارم کارهای بیشتری به این گونه از شما ببینیم.

2 سال پیش

شهرام می گوید



فقط ترجمه بود هیچ مفهوم خاصی را در بر نداشت. شما که این سایت را راه اندازی کردید باید جنبه آموزش تفهیمی را دنبال کند نه فقط یک سری مقاله را ترجمه کنید بزارید. به نظر من که هیچ ارزشی نداشت حداقل واسه من.

2 سال پیش

Ali می‌گوید

عالی بود با قدرت ادامه دهید

2 سال پیش

Sara می‌گوید

خیلی عالی بود من هر چی انگلیسی میخوندم نمیفهمیدم دستتون درد نکنه واقعا

2 سال پیش

میترا می‌گوید

آقای شهرام متاسفانه ما ایرانیا که هیچکدام عرضه نداریم یک علم یا شاخه علمی یا حداقل یک صنعت جدید یا ... را درست کنیم، پس مجبوریم هر چی که دیگران میسازن را ترجمه کنیم و یاد بگیریم برای مثال از برق تا کامپیوتر و اتومبیل تا موبایل و همه چیزهای دیگه را غیر ایرانیها اختراع و ابداع کردن اما این دلیل نمیشه که حداقل کار با محصولات اونها را هم با استفاده از مقاله های خودشون یاد نگیریم، پس مطالب آموزشی در هر کجای ایران و توسط هر کسی که نوشته بشه ترجمه + تجربه مترجم است و این یکی از بهترین ترجمه های ایرانیست که من تا حالا دیدم، پس بهتره از همین ترجمه عالی هم متشکر باشیم تا عقب موندگیمون بیشتر از این نشه.

من که خیلی از این سایت استفاده کردم دستتون درد نکنه ، عالیه

1 سال پیش

دانشجوی کتابداری می‌گوید

خیلی خیلی ممنون. منابع فارسی در این زمینه نداریم و این سایت کمک بسیار خوب و بزرگی می‌کند.  
از شما بسیار متشکرم.



1 سال پیش

میترامیرشفیعی می‌گوید

واقعا عالی، خسته نباشد.

8 ماه پیش

فرناز می‌گوید

عالی بود بسیار ممنونم

2 ماه پیش

حمید پارسا می‌گوید

با سلام خدمت مهندس حسن پور  
مطالبی که توضیح دادید بسیار روان و عالی بود و بنده هیچ سایت فارسی رو ندیدم که به این شکل کار کرده باشند.  
ان شاء الله که به کارتون ادامه بدید.

این سایت از اکیسمت برای کاهش هرزنامه استفاده می‌کند. بیاموزید که چگونه اطلاعات دیدگاه های شما پردازش می‌شوند.

