Word Embeddings

Slido: https://app.sli.do/event/1Bb81igx8eeAatCaEvtUTG

Yu Meng
University of Virginia
yumeng5@virginia.edu

Sept 15, 2025

Overview of Course Contents

- Week 1: Logistics & Overview
- Week 2: N-gram Language Models
- Week 3: Word Senses, Semantics & Classic Word Representations
- Week 4: Word Embeddings
- Week 5: Sequence Modeling & Recurrent Neural Networks (RNNs)
- Week 6: Language Modeling with Transformers
- Week 9: Large Language Models (LLMs) & In-context Learning
- Week 10: Knowledge in LLMs and Retrieval-Augmented Generation (RAG)
- Week 11: LLM Alignment
- Week 12: Reinforcement Learning for LLM Post-Training
- Week 13: LLM Agents + Course Summary
- Week 15 (after Thanksgiving): Project Presentations

(Recap) Why Care About Word Semantics?

- Understanding word meanings helps us build better language models!
- Recall the example from N-gram lectures:

$$\text{[BOS] The cat is on the mat [EOS]}$$
$$\text{[BOS] I have a cat and a mat [EOS]}$$
$$\text{[BOS] I like the cat [EOS]}$$

$$p(\text{“cat”} \mid \text{“the”}) = \frac{2}{3}, \quad p(\text{“mat”} \mid \text{“the”}) = \frac{1}{3},$$

- Sparsity: many valid bigram counts are zero – count-based measures do not account for word semantics!
- If we know “cat” is semantically similar to “dog”, then $p(\text{“dog”} \mid \text{“the”}) \approx p(\text{“cat”} \mid \text{“the”})$

(Recap) What Types of Word Semantics Exist in NLP?

- **Synonyms**: words with similar meanings
  - "happy" & "joyful"
- **Antonyms**: words with opposite meanings
  - "hot" & "cold"
- **Hyponyms** & **hypernyms**: one word is a more specific instance of another
  - "rose" is a hyponym of "flower"
  - "flower" is a hypernym of "rose"
- **Polysemy**: A single word having multiple related meanings
  - "mouse" can mean small rodents or the device that controls a cursor
- The study of these aspects of word meanings is called **lexical semantics** in linguistics

(Recap) Polysemy & Senses

- **Polysemy**: a single word has multiple related meanings
  - "**Light**": "This bag is **light**" / "Turn on the **light**" / "She made a **light** comment"
- **Sense**: a particular meaning or interpretation of a word in a given context
- Word relations (e.g., synonyms, antonyms, hypernyms/hyponyms) are defined between word senses!
- **Word sense disambiguation (WSD):** determine which sense of a word is being used in a specific context
  - She went to the **bank** to deposit money
  - She lives by the river **bank**
- WSD can be challenging especially when the context is short/insufficient
  - Is the query "mouse info" looking for a pet or a tool?

(Recap) Word Similarity

- Most words may not have many perfect synonyms, but usually have lots of similar words
  - “cat” is not a synonym of “dog”, but they are similar in meaning

vanish	disappear	9.8
belief	impression	5.95
muscle	bone	3.65
modest	flexible	0.98
hole	agreement	0.3

Word similarity (on a scale from 0 to 10) manually annotated by humans

- We’ll introduce word embeddings to automatically learn word similarity next week!

Figure source: https://web.stanford.edu/~jurafsky/slp3/5.pdf

(Recap) Word Relatedness & Semantic Field

- **Word relatedness**: the meaning of words can be related in ways other than similarity
  - Functional relationship: “doctor” and “hospital” – doctors work in hospitals
  - Thematic relationship: “bread” and “butter” – often used together in the context of food
  - Conceptual relationship: “teacher” and “chalkboard” – both part of the educational context
- **Semantic field**: a set of words which cover a particular semantic domain and bear structured relations with each other
  - Semantic field of “houses”: door, roof, kitchen, family, bed…
  - Semantic field of “restaurants”: waiter, menu, plate, food, chef…
  - Semantic field of “hospitals”: surgeon, nurse, anesthetic, scalpel…

(Recap) Connotation

- Subjective/cultural/emotional associations that words carry beyond their literal meanings
  - Youthful (positive) vs. childish (negative)
  - Confident (positive) vs. arrogant (negative)
  - Economical (positive) vs. cheap (negative)
- Connotation can be described via three dimensions:
  - Valence: the pleasantness of the stimulus
  - Arousal: the intensity of emotion provoked by the stimulus
  - Dominance: the degree of control exerted by the stimulus

(Recap) Connotation

- Valence: the pleasantness of the stimulus
  - High: “happy” / “satisfied”; low: “unhappy” / “annoyed”
- Arousal: the intensity of emotion provoked by the stimulus
  - High: “excited”; low: “calm”
- Dominance: the degree of control exerted by the stimulus
  - High: “controlling”; low: “influenced”

	Valence	Arousal	Dominance
courageous	8.05	5.5	7.38
music	7.67	5.57	6.5
heartbreak	2.45	5.65	3.58
cub	6.71	3.95	4.24

Earliest work on representing words with multi-dimensional vectors!

Figure source: https://web.stanford.edu/~jurafsky/slp3/5.pdf

(Recap) WordNet

- Word semantics is complex (multiple senses, various relations)!
- How did people represent word senses and relations in early NLP developments?
- WordNet: A manually curated large lexical database
- Three separate databases: one each for nouns, verbs and adjectives/adverbs
- Each database contains a set of lemmas, each one annotated with a set of senses
- Synset (synonym set): The set of near-synonyms for a sense
- Word relations (hypernym, hyponym, antonym) defined between synsets

WordNet: https://wordnet.princeton.edu/

(Recap) WordNet Relations

Relation	Also Called	Definition	Example
Hypernym	Superordinate	From concepts to superordinates	*breakfast*^{1} $\rightarrow$ *meal*^{1}
Hyponym	Subordinate	From concepts to subtypes	*meal*^{1} $\rightarrow$ *lunch*^{1}
Instance Hypernym	Instance	From instances to their concepts	*Austen*^{1} $\rightarrow$ *author*^{1}
Instance Hyponym	Has-Instance	From concepts to their instances	*composer*^{1} $\rightarrow$ *Bach*^{1}
Part Meronym	Has-Part	From wholes to parts	*table*^{2} $\rightarrow$ *leg*^{3}
Part Holonym	Part-Of	From parts to wholes	*course*^{7} $\rightarrow$ *meal*^{1}
Antonym		Semantic opposition between lemmas	*leader*^{1} $\Longleftrightarrow$ *follower*^{1}
Derivation		Lemmas w/same morphological root	*destruction*^{1} $\Longleftrightarrow$ *destroy*^{1}

Noun relations

Relation	Definition	Example
Hypernym	From events to superordinate events	*fly*^{9} $\rightarrow$ *travel*^{5}
Troponym	From events to subordinate event	*walk*^{1} $\rightarrow$ *stroll*^{1}
Entails	From verbs (events) to the verbs (events) they entail	*snore*^{1} $\rightarrow$ *sleep*^{1}
Antonym	Semantic opposition between lemmas	*increase*^{1} $\Longleftrightarrow$ *decrease*^{1}

Verb relations

(Recap) WordNet as a Graph

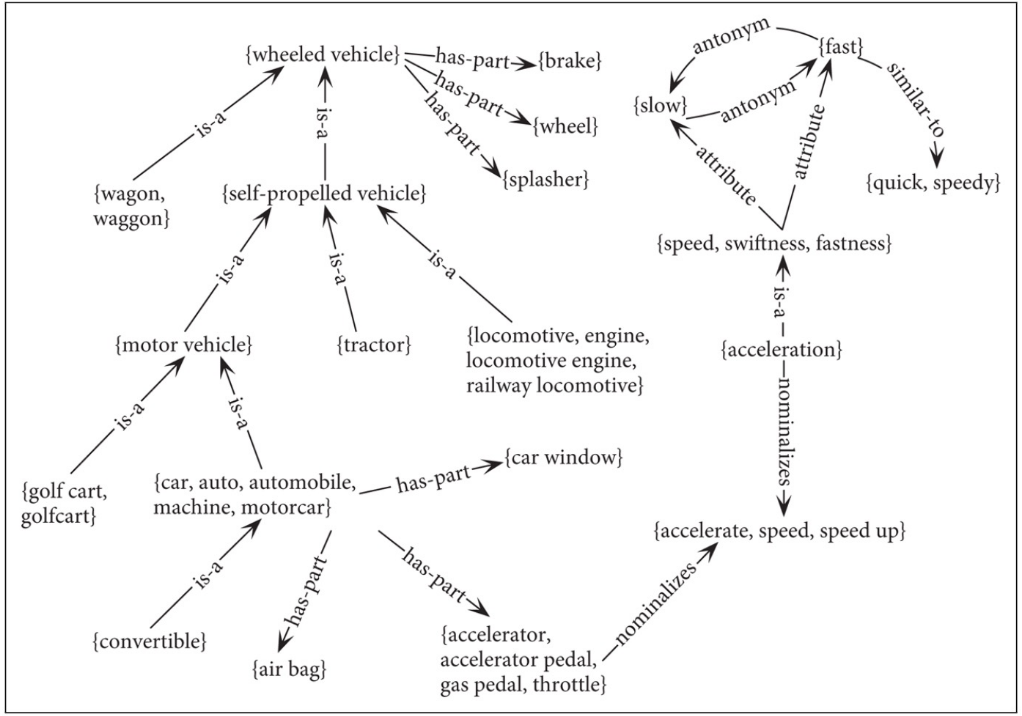


Figure source: https://academic.oup.com/edited-volume/42643/chapter/358151233

(Recap) WordNet Limitations

- Require significant efforts to construct and maintain/update
  - Hard to keep up with rapidly evolving language usage
- Limited coverage of domain-specific terms & low-resource language
  - No coverage of specialized, domain-specific terms (e.g., medical, legal, or technical)
- Only support individual words and their meanings
  - Do not account for idiomatic expressions, phrasal verbs, or collocations

A more automatic, scalable, and contextualized word semantic learning approach is needed!

(Recap) Motivation: Representing Texts with Vectors

- Word similarity computation is important for understanding semantics

Word similarity (on a scale from 0 to 10) manually annotated by humans

vanish	disappear	9.8
belief	impression	5.95
muscle	bone	3.65
modest	flexible	0.98
hole	agreement	0.3

Word semantics can be multi-faceted

	Valence	Arousal	Dominance
courageous	8.05	5.5	7.38
music	7.67	5.57	6.5
heartbreak	2.45	5.65	3.58
cub	6.71	3.95	4.24

- How to represent words numerically? Using multi-dimensional vectors!

Figure source: https://web.stanford.edu/~jurafsky/slp3/5.pdf

(Recap) Vector Semantics

- Represent a word as a point in a multi-dimensional semantic space
- A desirable vector semantic space: words with similar meanings are nearby in space

2D visualization of a desirable high-dimensional vector semantic space

(Recap) Vector Space Basics

- Vector notation: an N-dimensional vector $\boldsymbol{v} = [v_1, v_2, \dots, v_N] \in \mathbb{R}^N$
- Vector dot product/inner product:

$$\text{dot product}(\boldsymbol{v}, \boldsymbol{w}) = \boldsymbol{v} \cdot \boldsymbol{w} = v_1 w_1 + v_2 w_2 + \cdots + v_n w_n = \sum_{i=1}^{N} v_i w_i$$

- Vector length/norm:

$$|\boldsymbol{v}| = \sqrt{\boldsymbol{v} \cdot \boldsymbol{v}} = \sqrt{\sum_{i=1}^{N} v_i^2}$$

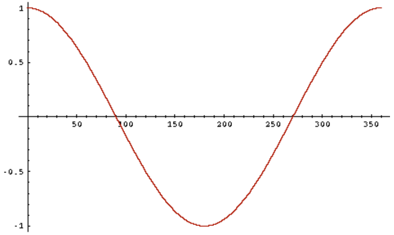
Other (less commonly-used) vector norms: Manhattan norm, *p*-norm, infinity norm…

- Cosine similarity between vectors:

$$\cos(\boldsymbol{v}, \boldsymbol{w}) = \frac{\boldsymbol{v} \cdot \boldsymbol{w}}{|\boldsymbol{v}||\boldsymbol{w}|} = \frac{\sum_{i=1}^{N} v_i w_i}{\sqrt{\sum_{i=1}^{N} v_i^2} \sqrt{\sum_{i=1}^{N} w_i^2}}$$

(Recap) Vector Similarity

- Cosine similarity is the most commonly used metric for similarity measurement
  - Symmetric: $\cos(\boldsymbol{v}, \boldsymbol{w}) = \cos(\boldsymbol{w}, \boldsymbol{v})$
  - Not influenced by vector length
  - Has a normalized range: [-1, 1]
  - Intuitive geometric interpretation



Cosine function values under different angles

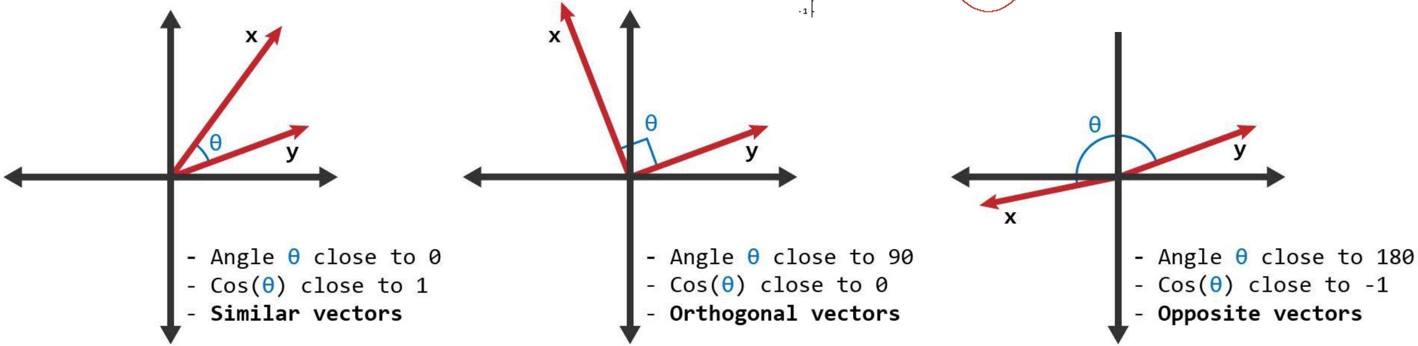


Figure source: https://www.learndatasci.com/glossary/cosine-similarity/

(Recap) How to Represent Words as Vectors?

- Given a vocabulary $\mathcal{V} = \{\text{good}, \text{feel}, \text{I}, \text{sad}, \text{cats}, \text{have}\}$
- Most straightforward way to represent words as vectors: use their indices
- One-hot vector: only one high value (1) and the remaining values are low (0)
- Each word is identified by a unique dimension

$$
\begin{aligned}
\boldsymbol{v}_{\text{good}} &= [1, 0, 0, 0, 0, 0] \\
\boldsymbol{v}_{\text{feel}} &= [0, 1, 0, 0, 0, 0] \\
\boldsymbol{v}_{\text{I}} &= [0, 0, 1, 0, 0, 0] \\
\boldsymbol{v}_{\text{sad}} &= [0, 0, 0, 1, 0, 0] \\
\boldsymbol{v}_{\text{cats}} &= [0, 0, 0, 0, 1, 0] \\
\boldsymbol{v}_{\text{have}} &= [0, 0, 0, 0, 0, 1]
\end{aligned}
$$

(Recap) Represent Sequences by Word Occurrences

- Consider the mini-corpus with three documents

$$d_1 = \text{“I feel good”}$$
$$d_2 = \text{“I feel sad”}$$
$$d_3 = \text{“I have cats”}$$

$$\boldsymbol{v}_{\text{good}} = [1, 0, 0, 0, 0, 0]$$
$$\boldsymbol{v}_{\text{feel}} = [0, 1, 0, 0, 0, 0]$$
$$\boldsymbol{v}_{\text{I}} = [0, 0, 1, 0, 0, 0]$$
$$\boldsymbol{v}_{\text{sad}} = [0, 0, 0, 1, 0, 0]$$
$$\boldsymbol{v}_{\text{cats}} = [0, 0, 0, 0, 1, 0]$$
$$\boldsymbol{v}_{\text{have}} = [0, 0, 0, 0, 0, 1]$$

- Straightforward way of representing documents: look at which words are present

$$\boldsymbol{v}_{d_1} = [1, 1, 1, 0, 0, 0]$$
$$\boldsymbol{v}_{d_2} = [0, 1, 1, 1, 0, 0]$$
$$\boldsymbol{v}_{d_3} = [0, 0, 1, 0, 1, 1]$$

Document vector similarity ➡

$$\cos(\boldsymbol{v}_{d_1}, \boldsymbol{v}_{d_2}) = \frac{2}{3}$$
$$\cos(\boldsymbol{v}_{d_1}, \boldsymbol{v}_{d_3}) = \frac{1}{3}$$
$$\cos(\boldsymbol{v}_{d_2}, \boldsymbol{v}_{d_3}) = \frac{1}{3}$$

(Recap) Document Similarity

- Document vector representation with word frequencies:

$$\boldsymbol{v}_{d_1} = [1, 114, \boxed{36, 20}] \quad \boldsymbol{v}_{d_2} = [0, 80, \boxed{58, 15}] \quad \boldsymbol{v}_{d_3} = [7, 62, 1, 2] \quad \boldsymbol{v}_{d_4} = [13, 89, 4, 3]$$

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

- "fool" and "wit" occur much more frequently in d_1 and d_2 than d_3 and d_4
- d_1 and d_2 are comedies $\cos(\boldsymbol{v}_{d_1}, \boldsymbol{v}_{d_2}) = 0.95 \quad \cos(\boldsymbol{v}_{d_2}, \boldsymbol{v}_{d_3}) = 0.81$
- Word frequencies in documents do reflect the semantic similarity between documents!

(Recap) Words Represented with Documents

- "Battle": "the kind of word that occurs in Julius Caesar and Henry V (history plays)"
- "Fool": "the kind of word that occurs in comedies"

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

- Represent words using their co-occurrence counts with documents:

$$\boldsymbol{v}_{\text{battle}} = [1, 0, 7, 13]$$
$$\boldsymbol{v}_{\text{good}} = [114, 80, 62, 89]$$
$$\boldsymbol{v}_{\text{fool}} = [36, 58, 1, 4]$$
$$\boldsymbol{v}_{\text{wit}} = [20, 15, 2, 3]$$

(Recap) Words Represented with Documents

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

$$\boldsymbol{v}_{\text{battle}} = [1, 0, 7, 13]$$
$$\boldsymbol{v}_{\text{good}} = [114, 80, 62, 89]$$
$$\boldsymbol{v}_{\text{fool}} = [36, 58, 1, 4]$$
$$\boldsymbol{v}_{\text{wit}} = [20, 15, 2, 3]$$

$$\cos(\boldsymbol{v}_{\text{fool}}, \boldsymbol{v}_{\text{wit}}) = 0.93$$
$$\cos(\boldsymbol{v}_{\text{fool}}, \boldsymbol{v}_{\text{battle}}) = 0.09$$

Previously:

$$\boldsymbol{v}_{\text{battle}} = [1, 0, 0, 0]$$
$$\boldsymbol{v}_{\text{good}} = [0, 1, 0, 0]$$
$$\boldsymbol{v}_{\text{fool}} = [0, 0, 1, 0]$$
$$\boldsymbol{v}_{\text{wit}} = [0, 0, 0, 1]$$

$$\cos(\boldsymbol{v}_{\text{fool}}, \boldsymbol{v}_{\text{wit}}) = 0$$
$$\cos(\boldsymbol{v}_{\text{fool}}, \boldsymbol{v}_{\text{battle}}) = 0$$

Document co-occurrence statistics provide coarse-grained contexts

(Recap) Fine-Grained Contexts: Word-Word Matrix

Instead of using documents as contexts for words, we can also use words as contexts

4 words to the left	center word	4 words to the right
is traditionally followed by	**cherry**	pie, a traditional dessert
often mixed, such as	**strawberry**	rhubarb pie. Apple pie
computer peripherals and personal	**digital**	assistants. These devices usually
a computer. This includes	**information**	available on the internet

(Recap) Fine-Grained Contexts: Word-Word Matrix

Count how many times words occur in a ±4 word window around the center word

context word

center word

	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

Counts derived from the Wikipedia corpus

(Recap) Word Similarity Based on Word Co-occurrence

- Word-word matrix with ±4 word window

	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

- “digital” and “information” both co-occur with “computer” and “data” frequently
- “cherry” and “strawberry” both co-occur with “pie” and “sugar” frequently
- Word co-occurrence statistics reflect word semantic similarity!
- Issues? Sparsity!

(Recap) Is Raw Frequency A Good Representation?

- On the one hand, high frequency can imply semantic similarity
- On the other hand, there are words with universally high frequencies

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

- Can we reweight the raw frequencies so that distinctively high frequency terms are highlighted?

(Recap) Term Frequency (TF)

- A word appearing 100 times in a document doesn't make it 100 times more likely to be relevant to the meaning of the document
- Instead of using the raw counts, we squash the counts with log scale

$$\mathrm{TF}(w,d) = \begin{cases} 1 + \log_{10} \mathrm{count}(w,d) & \mathrm{count}(w,d) > 0 \\ 0 & \text{otherwise} \end{cases}$$

(Recap) Document Frequency (DF)

- Motivation: Give a higher weight to words that occur only in a few documents
  - Terms that are limited to a few documents are more discriminative
  - Terms that occur frequently across the entire collection aren't as helpful
- Document frequency (DF): count how many documents a word occurs in

$$\mathrm{DF}(w) = \sum_{i=1}^{N} \mathbb{1}(w \in d_i) \longrightarrow$$

Evaluates to 1 if w occurs in d_i
otherwise evaluates to 0

- DF is NOT defined to be the total count of a word across all documents (collection frequency)!

	Collection Frequency	Document Frequency
Romeo	113	1
action	113	31

(Recap) Inverse Document Frequency (IDF)

- We want to emphasize discriminative words (with low DF)
- Inverse document frequency (IDF): total number of documents (N) divided by DF, in log scale

$$\mathrm{IDF}(w) = \log_{10}\left(\frac{N}{\mathrm{DF}(w)}\right)$$

Word	df	idf
Romeo	1	1.57
salad	2	1.27
Falstaff	4	0.967
forest	12	0.489
battle	21	0.246
wit	34	0.037
fool	36	0.012
good	37	0
sweet	37	0

DF & IDF statistics in the Shakespeare corpus

(Recap) TF-IDF Weighting

The TF-IDF weighted value characterizes the "salience" of a term in a document

$$\text{TF-IDF}(w, d) = \text{TF}(w, d) \times \text{IDF}(w)$$

TF-IDF weighted

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	0.246	0	0.454	0.520
good	0	0	0	0
fool	0.030	0.033	0.0012	0.0019
wit	0.085	0.081	0.048	0.054

$$\cos(\boldsymbol{v}_{d_2}, \boldsymbol{v}_{d_3}) = 0.10 \quad \cos(\boldsymbol{v}_{d_3}, \boldsymbol{v}_{d_4}) = 0.99$$

Raw counts

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

$$\cos(\boldsymbol{v}_{d_2}, \boldsymbol{v}_{d_3}) = 0.81 \quad \cos(\boldsymbol{v}_{d_3}, \boldsymbol{v}_{d_4}) = 0.99$$

(Recap) How to Define Documents?

- The concrete definition of documents is usually open to different design choices
  - Wikipedia article/page
  - Shakespeare play
  - Book chapter/section
  - Paragraph/sentence
  - …
- Larger documents provide broader context; smaller ones provide focused insights
- Depends on the analysis need: interested in global trends across documents (e.g., news articles) vs. more local patterns (e.g., specific sections of a legal document)?

(Recap) Probability-Based Weighting

- TF-IDF weighting scheme is based on heuristics
- Can we weigh the raw counts with probabilistic approaches?
- Intuition: the association between two words can be reflected by **how much they co-occur more than by chance**

context word | summed counts

center word | summed counts

	computer	data	result	pie	sugar	count(w)
cherry	2	8	9	442	25	486
strawberry	0	0	1	60	19	80
digital	1670	1683	85	5	4	3447
information	3325	3982	378	5	13	7703
count(context)	4997	5673	473	512	61	11716

(Recap) Word Association Based on Probability

- In probability theory, when two random variables A & B are independent, we have

  Joint probability $p(A,B) = p(A)p(B)$

- When two words co-occur by chance, we expect their probabilities to satisfy the independence assumption: $p(w_1, w_2) = p(w_1)p(w_2)$
- When $p(w_1, w_2) > p(w_1)p(w_2)$, two words co-occur more often than would be expected by chance
- How to develop a probabilistic metric to characterize this association?

(Recap) Pointwise Mutual Information (PMI)

- PMI compares the probability of two words co-occurring with the probabilities of the words occurring independently

$$\text{PMI} = \log_2 \frac{p(w_1, w_2)}{p(w_1)p(w_2)} = \log_2 \frac{\#(w_1, w_2) \cdot N}{\#(w_1)\#(w_2)}$$

N: Total word counts

- PMI = 0: Two words co-occur as expected by chance => no particular association
- PMI > 0: Two words co-occur more often than by chance => the higher the PMI, the stronger the association between the words
- PMI < 0: Two words co-occur less often than expected by chance => negative associations; not much actionable insight
- Positive PMI (PPMI): replaces all negative PMI values with zero

$$\text{PPMI} = \max\left(\log_2 \frac{p(w_1, w_2)}{p(w_1)p(w_2)}, 0\right)$$

(Recap) PPMI Example

Raw counts

	computer	data	result	pie	sugar
cherry	2	8	9	442	25
strawberry	0	0	1	60	19
digital	1670	1683	85	5	4
information	3325	3982	378	5	13

PPMI-weighted matrix

	computer	data	result	pie	sugar
cherry	0	0	0	4.38	3.30
strawberry	0	0	0	4.10	5.51
digital	0.18	0.01	0	0	0
information	0.02	0.09	0.28	0	0

Issue: biased toward infrequent events (rare words tend to have very high PMI values)

PPMI with Power Smoothing

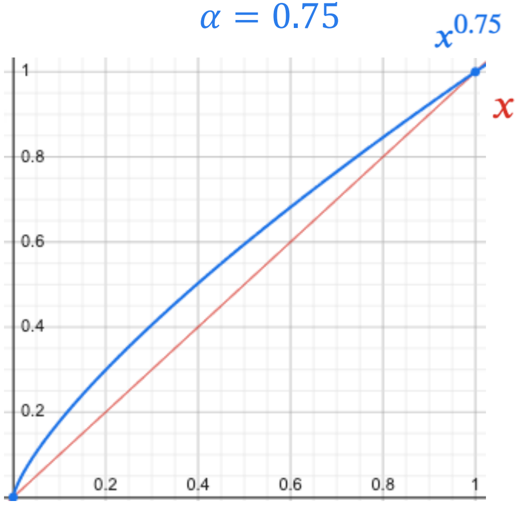
Power smoothing: Manually boost low probabilities by raising to a power α

$$\text{PPMI} = \max\left(\log_2 \frac{p(w_1, w_2)}{p(w_1)p(w_2)}, 0\right)$$

Original: $$p(w) = \frac{\#(w)}{\sum_{w' \in \mathcal{V}} \#(w')}$$

Power smoothed: ($\alpha < 1$) $$p_\alpha(w) = \frac{\#(w)^\alpha}{\sum_{w' \in \mathcal{V}} \#(w')^\alpha}$$

$\alpha = 0.75$



PPMI with Add-*k* Smoothing

- Another way of increasing the counts of rare occurrences is to apply add-*k* smoothing

	computer	data	result	pie	sugar
cherry	2	8	9	442	25
strawberry	0	0	1	60	19
digital	1670	1683	85	5	4
information	3325	3982	378	5	13

Add a constant *k* to all counts

- The larger the *k* (*k* can be larger than 1), the more we boost the probability of rare occurrences

TF-IDF vs. PMI Weighting

- TF-IDF
  - Measures the importance of a word in a document relative to other documents (corpus)
  - Context granularity: document level
  - Based on heuristics
  - High TF-IDF = frequent in a document but infrequent across the corpus
- PMI:
  - Measures the strength of association between two words
  - Context granularity: word pair level (usually based on local context windows)
  - Based on probability assumptions
  - High PMI = words co-occur more often than expected by chance, a strong association

Summary: Word Semantics & Senses

- Understanding word semantics & senses help us build better language models!
- Word semantics is complex
  - Polysemy: a single word having multiple meanings
  - Multi-faceted: word meanings entail various aspects (e.g., valence, arousal, dominance)
- Many types of word relations: synonyms, antonyms, hyponyms & hypernyms…
- Word relations are usually not binarized (e.g., perfect synonyms are rare); word similarity is usually a more flexible measure

Summary: Classic Word Representations

- Large-scale lexical databases (WordNet) were constructed in early NLP developments
- WordNet consists of manually curated synsets linked by relation edges
- WordNet can be used as a database for word sense disambiguation
- WordNet has significant limitations:
  - Require significant efforts to construct and maintain/update
  - Limited coverage of domain-specific terms & low-resource language
  - Only support individual words and their meanings

Summary: Vector Space Models

- Vector semantic space: use vector representations to reflect word semantics
- Cosine similarity is the most-commonly used metric for vector similarity
- Word-document & word-word co-occurrence statistics provide valuable semantic information – count-based vector representations work decently well
- Raw counts are not good representations (e.g., biased to universally frequent terms)
- TF-IDF highlights the important words in a document relative to other documents
- PMI measures the strength of association between two words based on probabilistic (independence) assumptions

Agenda

- Sparse vs. Dense Vectors
- Word Embeddings: Overview
- Word2Vec Training
- Word Embedding Properties & Evaluation

Count-based Vector Limitations

- Count-based vectors are **sparse** (lots of zeros)
  - Zero values in the vectors do not carry any semantics
- Count-based vectors are **long** (many dimensions)
  - Vector dimension = vocabulary size (usually > 10K)
  - “Curse of dimensionality”: metrics (e.g. cosine) become less meaningful in high dimensions

	aardvark	**...**	**computer**	**data**	**result**	**pie**	**sugar**	**...**
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

Many more words!

Dense Vectors

- More efficient & effective vector representations?
- **Dense** vectors!
  - Most/all dimensions in the vectors are non-zero
  - Usually floating-point numbers; each dimension could be either positive or negative
  - Dimension much smaller than sparse vectors (i.e., << 10K)
- Also called “**distributed** representations”
  - The information is **distributed** across multiple units/dimensions
  - Each unit/dimension participates in representing multiple pieces of information
  - Analogous to human brains: the brain stores and processes information in a distributed manner: instead of having a single neuron/region represent a concept, information is represented across a network of neurons

Dense Vector Example

- One dimension might (partly) contribute to distinguishing animals (“cat” “dog”) from vehicles (“car” “truck”)
- One dimension might (partly) capture some aspect of size
- Another might (partly) represent formality or emotional tone
- …
- Each of these dimensions is not exclusively responsible for any single concept, but together, they combine to form a rich and nuanced representation of words!

$$\boldsymbol{v}_{\text{good}} = [-1.34, 2.58, 0.37, 4.32, -3.21, \ldots]$$
$$\boldsymbol{v}_{\text{nice}} = [-0.58, 1.97, 0.20, 3.13, -2.58, \ldots]$$

Only showing two decimal places
(typically they are floating point numbers!)

Dense Vectors Pros & Cons

- **(+) Compactness**: Represent a large number of concepts using fewer resources (richer semantic information per dimension); easier to use as features to neural networks
- **(+) Robustness**: Information is spread across many dimensions => more robust to the randomness/noise in individual units
- **(+) Scalability & Generalization**: Efficiently handle large-scale data and generalize to various applications
- **(-) Lack of Interpretability**: (Unlike sparse vectors) difficult to assign a clear meaning to individual dimensions, making model interpretation challenging

Agenda

- Sparse vs. Dense Vectors
- Word Embeddings: Overview
- Word2Vec Training
- Word Embedding Properties & Evaluation

Distributional Hypothesis

- Words that occur in similar contexts tend to have similar meanings
- A word’s meaning is largely defined by the company it keeps (its context)
- Example: suppose we don’t know the meaning of “Ong choy” but see the following:
  - Ong choy is delicious **sautéed with garlic**
  - Ong choy is superb **over rice**
  - … ong choy **leaves** with **salty** sauces
- And we’ve seen the following contexts:
  - … spinach **sautéed with garlic over rice**
  - … chard stems and **leaves** are **delicious**
  - … collard greens and other **salty** leafy greens
- Ong choy = water spinach!

Example source: https://web.stanford.edu/~jurafsky/slp3/slides/vectorsemantics2024.pdf

Word Embeddings: General Idea

- Learn dense vector representations of words based on distributional hypothesis
- Semantically similar words (based on context similarity) will have similar vector representations
- **Embedding**: a mapping that takes elements from one space and represents them in a different space

$$\boldsymbol{v}_{\text{to}} = [1,0,0,0,0,0,\ldots]$$
$$\boldsymbol{v}_{\text{by}} = [0,1,0,0,0,0,\ldots]$$
$$\boldsymbol{v}_{\text{that}} = [0,0,1,0,0,0,\ldots]$$
$$\boldsymbol{v}_{\text{good}} = [0,0,0,1,0,0,\ldots]$$
$$\boldsymbol{v}_{\text{nice}} = [0,0,0,0,1,0,\ldots]$$
$$\boldsymbol{v}_{\text{bad}} = [0,0,0,0,0,1,\ldots]$$

2D visualization of a word embedding space

Figure source: https://web.stanford.edu/~jurafsky/slp3/6.pdf

Learning Word Embeddings

- Assume a large text collection (e.g., Wikipedia)
- Hope to learn similar word embeddings for words occurring in similar contexts
- Construct a prediction task: use a center word’s embedding to predict its contexts!
- Intuition: If two words have similar embeddings, they will predict similar contexts, thus being semantically similar!

Word Embedding Is Self-Supervised Learning

- **Self-supervised learning**: a model learns to predict parts of its input from other parts of the same input

Input: *Ong choy is superb over rice*

Prediction task: Ong choy

is

superb

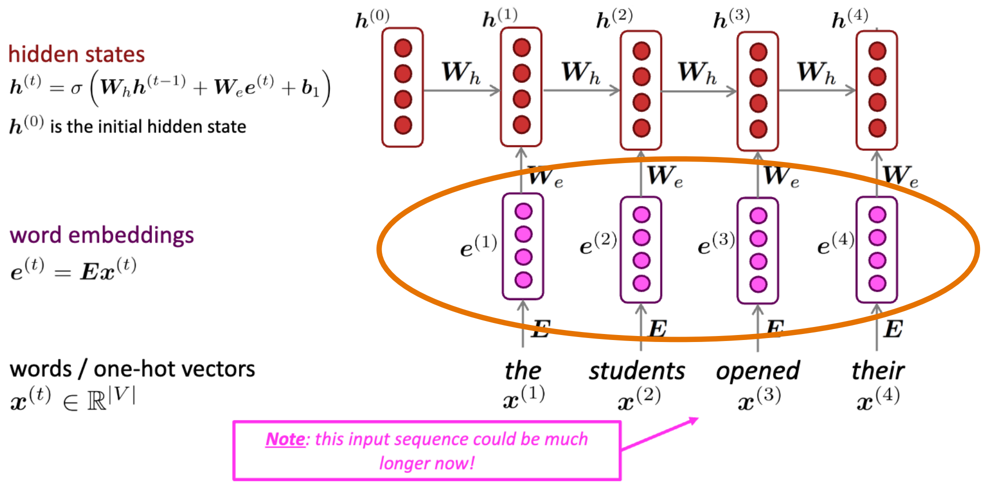
over

rice

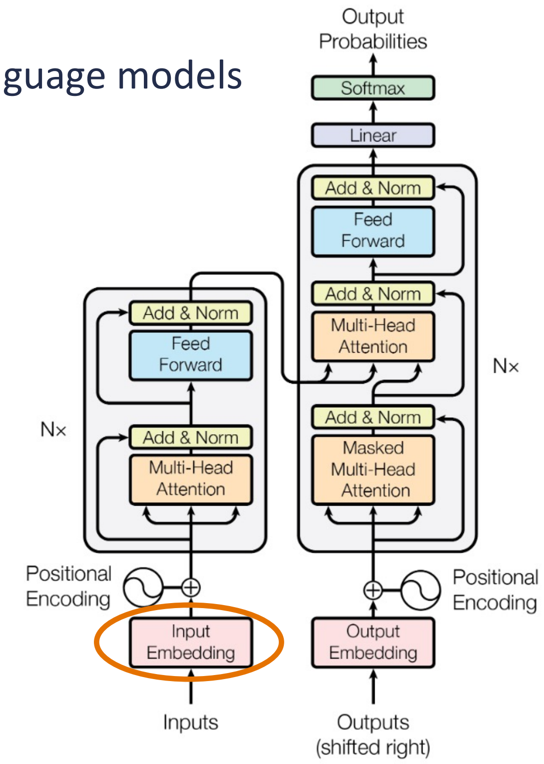
- Self-supervised learning vs. supervised learning:
  - Self-supervised learning: **no human-labeled data** – the model learns from unlabeled data by generating supervision through the structure of the data itself
  - Supervised learning: **use human-labeled data** – the model learns from human annotated input-label pairs

Word Embedding as Input Features

Word embeddings are commonly used as input features to language models



RNN Language Model: https://web.stanford.edu/class/cs224n/slides/cs224n-spr2024-lecture05-rnnlm.pdf



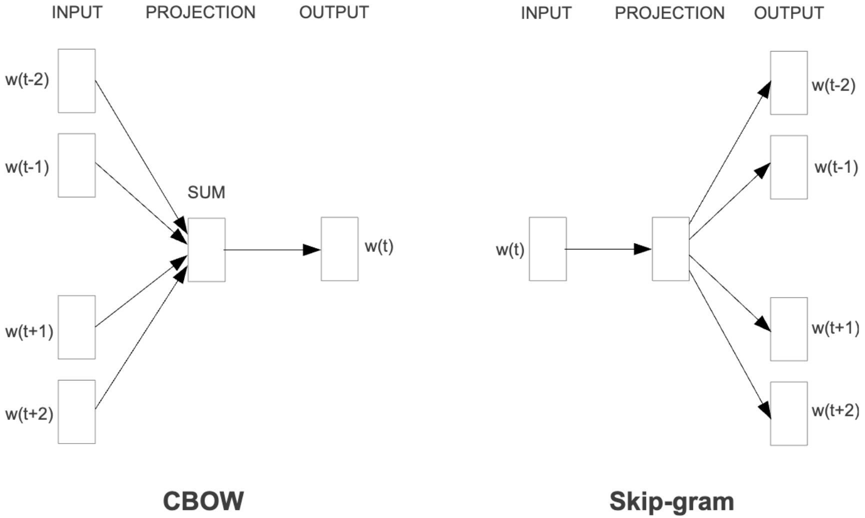
Transformer: https://arxiv.org/pdf/1706.03762

Agenda

- Sparse vs. Dense Vectors
- Word Embeddings: Overview
- Word2Vec Training
- Word Embedding Properties & Evaluation

Word2Vec Overview

- The earliest & most well-known word embedding learning method (published in 2013)
- Two variants: Skip-gram and CBOW (Continuous Bag-of-Words)
- We will mainly cover Skip-gram in this lecture



Word2Vec paper: https://arxiv.org/pdf/1301.3781

Word2Vec Setting

- Input: a corpus D – the larger, the better!
- Training data: word-context pairs (w, c) where w is a center word, and c is a context word
  - Each word in the corpus can act as center word
  - Context words = neighboring words of the center word in a local context window ($\pm l$ words)
- Parameters to learn: $\boldsymbol{\theta} = \{\boldsymbol{v}_{\boldsymbol{w}}, \boldsymbol{v}_{\boldsymbol{c}}\}$ – each word has two vectors (center word representation & context word representation)
- The center word representations $\boldsymbol{v}_{\boldsymbol{w}}$ are usually used as the final word embeddings
- Number of parameters to store: $d \times |V|$
  - d is the embedding dimension; usually 100-300
  - $|V|$ is the vocabulary size; usually > 10K
  - Sparse vector representations will have $|V|^2$ parameters!

Word2Vec Training Data Example

- Input sentence: “there is a cat on the mat”
- Suppose context window size = 2
- Word-context pairs as training data:
  - (there, is), (there, a)
  - (is, there), (is, a), (is, cat)
  - (a, there), (a, is), (a, cat), (a, on)
  - (cat, is), (cat, a), (cat, on), (cat, the)
  - (on, a), (on, cat), (on, the), (on, mat)
  - (the, cat), (the, on), (the, mat)
  - (mat, on), (mat, the)

there is a cat on the mat
there is a cat on the mat
there is a cat on the mat
there is a cat on the mat
there is a cat on the mat
there is a cat on the mat
there is a cat on the mat

- “Skip-gram”: skipping over some context words to predict the others!
- Training data completely derived from the raw corpus (no human labels!)

Word2Vec Objective (Skip-gram)

- Intuition: predict the contexts words using the center word (semantically similar center words will predict similar contexts words)
- Objective: using the parameters $\boldsymbol{\theta} = \{\boldsymbol{v_w}, \boldsymbol{v_c}\}$ to maximize the probability of predicting the context word c using the center word w

$$\max_{\boldsymbol{\theta}} \prod_{(w,c)\in\mathcal{D}} \boxed{p_{\boldsymbol{\theta}}(c|w)}$$

Probability expressed as a function of the model parameters

- How to parametrize the probability?

Word2Vec Probability Parametrization

- Word2Vec objective: $\max_{\boldsymbol{\theta}} \prod_{(w,c)\in\mathcal{D}} p_{\boldsymbol{\theta}}(c|w)$
- Assume the log probability (i.e., logit) is proportional to vector dot product

$$\log p_{\boldsymbol{\theta}}(c|w) \propto \boldsymbol{v}_c \cdot \boldsymbol{v}_w$$

- Rationale: a larger vector dot product *can* indicate a higher vector similarity
- Why not use cosine similarity?
  - Cosine similarity is a non-linear function; more complicated to optimize than dot product
  - With advanced optimization techniques, optimizing cosine similarity is more beneficial (Meng et al.)

Spherical Text Embedding: https://arxiv.org/pdf/1911.01196

Word2Vec Parameterized Objective

- Word2Vec objective: $$\max_{\boldsymbol{\theta}} \prod_{(w,c)\in\mathcal{D}} p_{\boldsymbol{\theta}}(c|w)$$

- Assume the log probability (i.e., logit) is proportional to vector dot product

$$\log p_{\boldsymbol{\theta}}(c|w) \propto \boldsymbol{v}_c \cdot \boldsymbol{v}_w$$

- The final probability distribution is given by the softmax function:

$$p_{\boldsymbol{\theta}}(c|w) = \frac{\exp(\boldsymbol{v}_c \cdot \boldsymbol{v}_w)}{\sum_{c'\in|\mathcal{V}|} \exp(\boldsymbol{v}_{c'} \cdot \boldsymbol{v}_w)} \Rightarrow \sum_{c'\in|\mathcal{V}|} p_{\boldsymbol{\theta}}(c'|w) = 1$$

- Word2Vec objective (log-scale):

$$\max_{\boldsymbol{\theta}} \sum_{(w,c)\in\mathcal{D}} \log p_{\boldsymbol{\theta}}(c|w) = \sum_{(w,c)\in\mathcal{D}} \left(\boldsymbol{v}_c \cdot \boldsymbol{v}_w - \log \sum_{c'\in|\mathcal{V}|} \exp(\boldsymbol{v}_{c'} \cdot \boldsymbol{v}_w) \right)$$

Word2Vec Negative Sampling

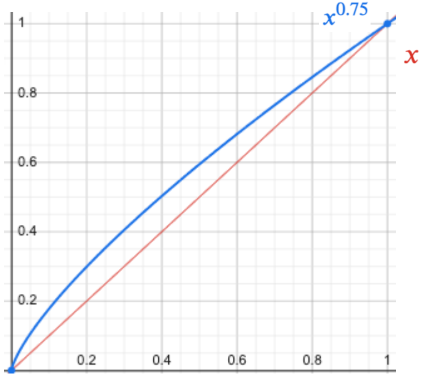
- Challenges with the original objective: Sum over the entire vocabulary – expensive!

$$\max_{\boldsymbol{\theta}} \sum_{(w,c)\in\mathcal{D}} \log p_{\boldsymbol{\theta}}(c|w) = \sum_{(w,c)\in\mathcal{D}} \left(\boldsymbol{v}_c \cdot \boldsymbol{v}_w - \log \boxed{\sum_{c'\in|\mathcal{V}|}} \exp(\boldsymbol{v}_{c'} \cdot \boldsymbol{v}_w) \right)$$

- Randomly sample a few negative terms from the vocabulary to form a negative set N
- How to sample negatives? Based on the (power-smoothed) unigram distribution

$$p_{\text{neg}}(w) \propto \left(\frac{\#(w)}{\sum_{w'\in\mathcal{V}} \#(w')} \right)^{0.75}$$

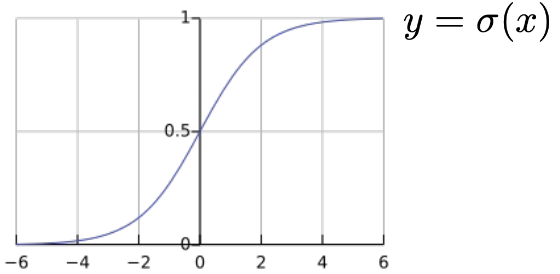
Rare words get a bit boost in sampling probability



Word2Vec Negative Sampling

- Formulate a binary classification task; predict whether (w, c) is a real context pair:

$$p_{\boldsymbol{\theta}}(\text{True}|c, w) = \sigma(\boldsymbol{v}_c \cdot \boldsymbol{v}_w) = \frac{1}{1 + \exp(-\boldsymbol{v}_c \cdot \boldsymbol{v}_w)}$$



- Maximize the binary classification probability for real context pairs, and minimize for negative (random) pairs

$$\max_{\boldsymbol{\theta}} \log \sigma(\boldsymbol{v}_c \cdot \boldsymbol{v}_w) - \sum_{c' \in \mathcal{N}} \log \sigma(\boldsymbol{v}_{c'} \cdot \boldsymbol{v}_w)$$

Real context pair

Negative context pair

Word2Vec Optimization

- How to optimize the following objective?

$$\max_{\boldsymbol{\theta}} \log \sigma(\boldsymbol{v}_c \cdot \boldsymbol{v}_w) - \sum_{c' \in \mathcal{N}} \log \sigma(\boldsymbol{v}_{c'} \cdot \boldsymbol{v}_w)$$

- Stochastic gradient descent (SGD)!
- First, initialize parameters $\boldsymbol{\theta} = \{\boldsymbol{v}_w, \boldsymbol{v}_c\}$ with random d-dimensional vectors
- In each step: update parameters in the direction of the gradient of the objective (weighted by the learning rate)

$$\boldsymbol{\theta}^{(t+1)} \leftarrow \boldsymbol{\theta}^{(t)} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}\big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(t)}}$$

Learning rate Loss function

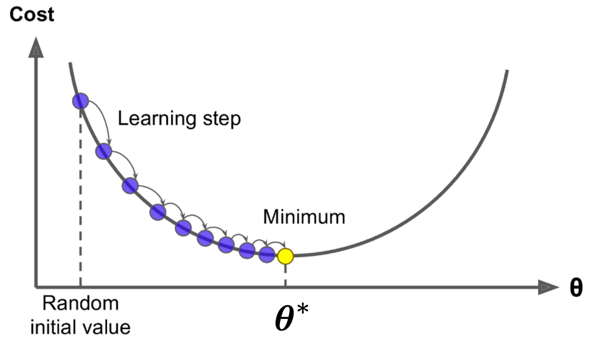


Figure source: https://web.stanford.edu/class/cs224n/slides/cs224n-spr2024-lecture02-wordvecs2.pdf

Word2Vec Hyperparameters

- Word embedding dimension d (usually 100-300)
  - Larger d provides richer vector semantics
  - Extremely large d suffers from inefficiency and curse of dimensionality
- Local context window size l (usually 5-10)
  - Smaller l learns from immediately nearby words – more syntactic information
  - Bigger l learns from longer-ranged contexts – more semantic/topical information
- Number of negative samples k (usually 5-10)
  - Larger k usually makes training more stable but also more costly
- Learning rate η (usually 0.02-0.05)

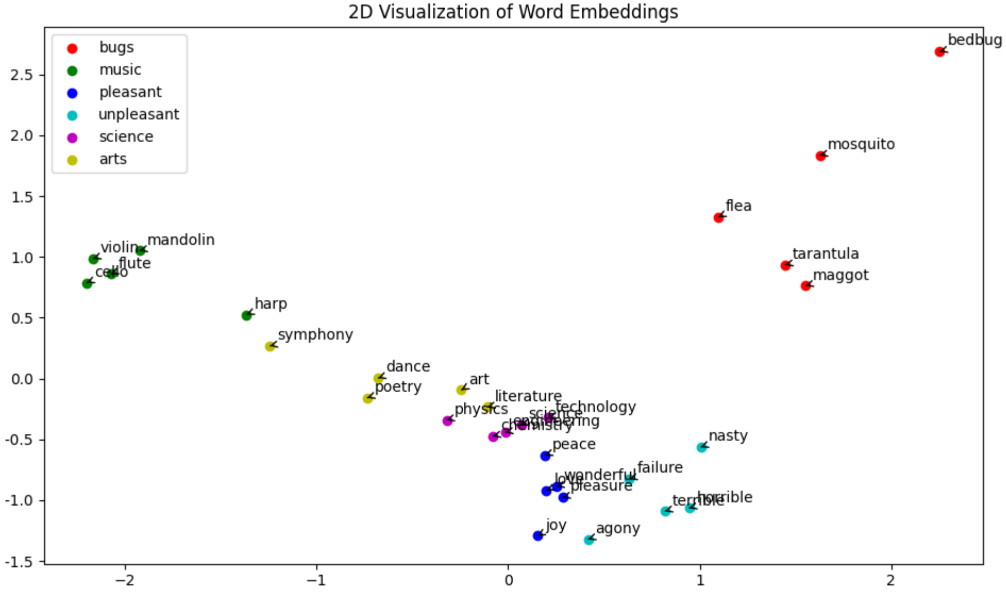
UNIVERSITY of VIRGINIA

Agenda

- Sparse vs. Dense Vectors
- Word Embeddings: Overview
- Word2Vec Training
- Word Embedding Properties & Evaluation

Word Similarity

- Measure word similarity with cosine similarity between embeddings $\cos(\boldsymbol{v}_{w_1}, \boldsymbol{v}_{w_2})$
- Higher cosine similarity = more semantically close



Word Similarity Evaluation

- An **intrinsic** word embedding evaluation
- Measure how well word vector similarity correlates with human judgments
- Example dataset: WordSim353 (353 word pairs with their similarity scores assessed by humans)

Word 1	Word 2	Human (mean)
tiger	cat	7.35
book	paper	7.46
computer	internet	7.58
plane	car	5.77
professor	doctor	6.62
stock	phone	1.62
stock	CD	1.31
stock	jaguar	0.92

Figure source: https://web.stanford.edu/class/cs224n/slides/cs224n-spr2024-lecture02-wordvecs2.pdf

Correlation Metric

Spearman rank correlation: measure the correlation between two rank variables

Word 1	Word 2	Human (mean)
tiger	cat	7.35
book	paper	7.46
computer	internet	7.58
plane	car	5.77
professor	doctor	6.62
stock	phone	1.62
stock	CD	1.31
stock	jaguar	0.92

Rank by human

$$\begin{bmatrix} 6 \\ 7 \\ 8 \\ 4 \\ 5 \\ 3 \\ 2 \\ 1 \end{bmatrix}$$

Covariance

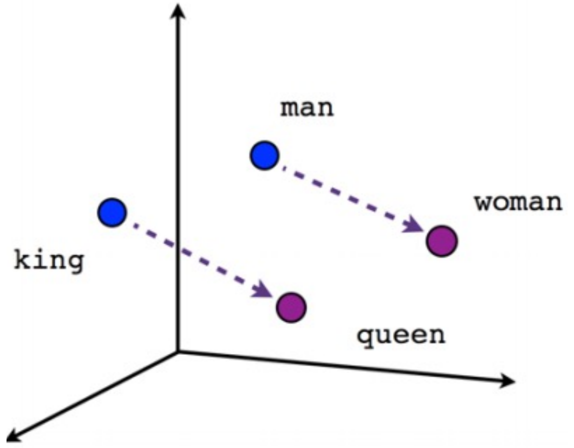
$$r = \frac{\mathrm{Cov}[R[X], R[Y]]}{\sigma_{R[X]}\sigma_{R[Y]}}$$

Standard deviations

Word Analogy

- Word embeddings reflect intuitive semantic and syntactic analogy
- Example: man : woman :: king : ? $\boldsymbol{v}_{\text{queen}} \approx \boldsymbol{v}_{\text{woman}} - \boldsymbol{v}_{\text{man}} + \boldsymbol{v}_{\text{king}}$
- General case: find the word such that a : b :: c : ?
- Find the word that maximizes the cosine similarity

$$
\begin{aligned}
w &= \arg\max_{w' \in \mathcal{V}} \cos(\boldsymbol{v}_b - \boldsymbol{v}_a + \boldsymbol{v}_c, \boldsymbol{v}_{w'}) \\
&= \arg\max_{w' \in \mathcal{V}} \frac{(\boldsymbol{v}_b - \boldsymbol{v}_a + \boldsymbol{v}_c) \cdot \boldsymbol{v}_{w'}}{|\boldsymbol{v}_b - \boldsymbol{v}_a + \boldsymbol{v}_c||\boldsymbol{v}_{w'}|}
\end{aligned}
$$



Word Analogy Evaluation

- Word analogy is another **intrinsic** word embedding evaluation
- Encompass various types of word relationships
- Usually use accuracy as the metric

Type of relationship	Word Pair 1		Word Pair 2	
Common capital city	Athens	Greece	Oslo	Norway
All capital cities	Astana	Kazakhstan	Harare	Zimbabwe
Currency	Angola	kwanza	Iran	rial
City-in-state	Chicago	Illinois	Stockton	California
Man-Woman	brother	sister	grandson	granddaughter
Adjective to adverb	apparent	apparently	rapid	rapidly
Opposite	possibly	impossibly	ethical	unethical
Comparative	great	greater	tough	tougher
Superlative	easy	easiest	lucky	luckiest
Present Participle	think	thinking	read	reading
Nationality adjective	Switzerland	Swiss	Cambodia	Cambodian
Past tense	walking	walked	swimming	swam
Plural nouns	mouse	mice	dollar	dollars
Plural verbs	work	works	speak	speaks

Figure source: https://arxiv.org/pdf/1301.3781

Extrinsic Evaluation of Word Embeddings

- Word embeddings can be used as input features to task-specific NLP models
- Example 1: Text classification (topic/sentiment classification)
  - Sentence/document embeddings are obtained by applying sequence modeling architectures on top of word embeddings
  - Classification accuracy is used as the extrinsic metric
- Example 2: Named entity recognition (NER)
  - Find and classify entity names (e.g., person, organization, location) in text
  - Concatenated word embeddings can be used to represent spans of words (entities)
  - Precision/recall/F1 are used as the extrinsic metrics
- Word embedding demo

Thank You!

Yu Meng
University of Virginia
yumeng5@virginia.edu