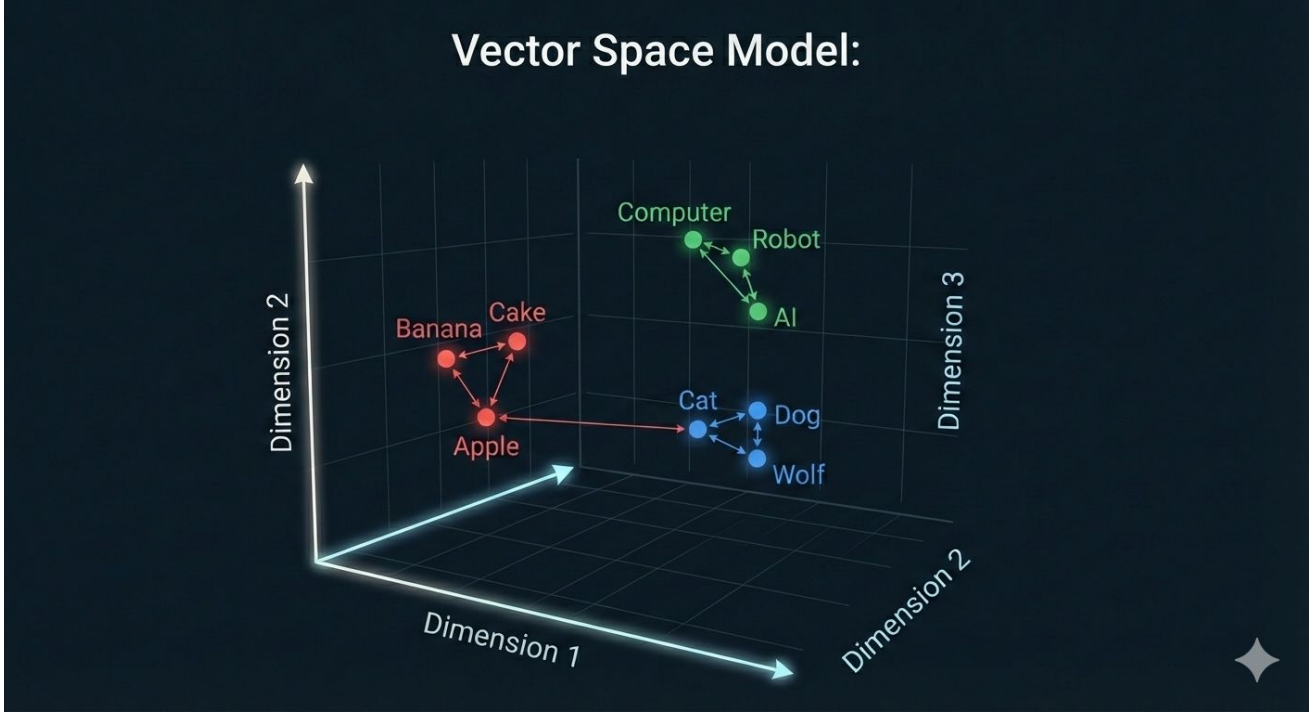


Natural Language Understanding

CSCI 4907/6515

Lecture 5: February 10, 2026

Aya Zirikly

Quiz 2

02/24/2026

Week 3, 4, and 5

- Please know your materials from week 1 & 2

Summary

Sparse vectors

Definition: A vector where **most** of the values are **0**.

Structure: The length of the vector is equal to the size of the entire Vocabulary (e.g., 50,000).

Examples:

- **One-Hot Encoding:** `[0, 0, 1, 0, 0, ...]` (Only one "1" for the specific word).
- **TF-IDF:** `[0, 0.05, 0, 0.92, 0, ...]` (Only non-zero for words present in the document).

The Logic: "Localist" representation. Each dimension represents exactly **one** specific word definition. Dimension 452 is *always* "dog".

- **Pros:** Highly interpretable. If index 5 has a high value, you know exactly which word caused it.
- **Cons:**
  - **Wasteful:** 99% of memory is used to store zeros.
  - **Brittle:** "Car" and "Automobile" are completely different dimensions (orthogonal). The model doesn't know they are related.

Dense vectors

Definition: A vector where **every** value is a real number (float), and almost none are 0.
Structure: The length is fixed and small (e.g., d **= 300**), regardless of vocabulary size.
Examples:

- **Word2Vec / GloVe:** `[0.2, -0.4, 0.8, -0.1, ...]`
- **BERT Embeddings:** `[-0.1, 0.9, 0.3, ...]`

The Logic: "Distributed" representation. The meaning of "dog" is smeared across all 300 dimensions. No single number means "dog"; the *pattern* of numbers means "dog".
Pros:

- **Semantic:** Captures relationships. "Car" and "Auto" will have very similar vector patterns.
- **Efficient:** Compresses massive vocabularies into small memory spaces.

Cons:

- **Black Box:** You cannot look at Dimension 4 and say "Ah, this number represents 'fuzziness'."

Feature	Sparse (Count-Based)	Dense (Prediction-Based)
Dimensionality	High (**~100,000+**)	Low (**~100 - 1,000**)
Values	Mostly **Zeros**	Mostly **Non-Zeros**
Generalization	Poor (Synonyms fail)	Excellent (Synonyms match)
Creation	Counting Statistics	Neural Network Training

Static embeddings

The Limitation of Word2Vec

- **Problem:** Word2Vec assigns **one** fixed vector to every word in the dictionary.

Static embeddings

The Limitation of Word2Vec

- **Problem:** Word2Vec assigns **one** fixed vector to every word in the dictionary.
- **The "Bank" Issue:**
  - Sentence A: "I deposited money at the **bank**."
  - Sentence B: "I sat on the river **bank**."

  The vector for **bank** is the exact same in both sentences. ***It has to be an "average" of all meanings***, which makes it fuzzy.

Fuzzy?

Every word gets exactly **one** vector in the dictionary.

- The training data pulls the vector toward "Fruit" (near *banana, pear*).
- The training data *also* pulls the vector toward "Tech" (near *Microsoft, Google*).

Fuzzy?

Every word gets exactly **one** vector in the dictionary.

- The training data pulls the vector toward "Fruit" (near *banana, pear*).
- The training data *also* pulls the vector toward "Tech" (near *Microsoft, Google*).

The final vector gets stuck in the **middle**

- It ends up in a weird location that is neither "Fruit" nor "Tech."
- To a computer, "Apple" looks like a "fruit-tech-hybrid." It is mathematically "fuzzy" because it tries to be two things at once.

If you search for "Apple pie recipes," a static model might retrieve "MacBook repairs" because the vector for "Apple" is contaminated by its tech meaning.

Fuzzy?

Every word gets exactly **one** vector in the dictionary.

- The training data pulls the vector toward "Fruit" (near *banana, pear*).
- The training data *also* pulls the vector toward "Tech" (near *Microsoft, Google*).

The final vecto

- It ends up
- To a com
  tries to be two things at once.

BERT Fix: BERT creates a *brand new* vector for "apple" every time it sees it, based on the sentence. It effectively splits the word into **apple_fruit** and **apple_tech**.

If you search for "Apple pie recipes," a static model might retrieve "MacBook repairs" because the vector for "Apple" is contaminated by its tech meaning.

Contextualized embeddings

Dynamic Vectors Instead of a static lookup table, we use a model that generates vectors **on the fly**.

- **Input:** The entire sentence.
- **Output:** A unique vector for every word, tailored to that specific sentence.

The "Function" View

- **Old Way (Static):**
  - `Vector = Lookup("Bank")` → Returns `[0.2, 0.9, ...]`
- **New Way (Contextual):**
  - `Vector = Model("I sat by the river", focus_word="bank")` → Returns `[0.8, 0.1, ...]` (Nature)
  - `Vector = Model("I deposited cash", focus_word="bank")` → Returns `[0.1, 0.9, ...]` (Finance)

We want the vector for "Bank" to *move* in vector space depending on its neighbors

ELMo (Embeddings from Language Models)

The First Breakthrough (2018) ELMo was the first major model to solve polysemy using **Deep LSTMs** (Recurrent Neural Networks/Long-Short Term Memory Networks).

ELMo reads the sentence twice simultaneously:

ELMo takes the vector from the Forward LSTM and concatenates it with the Backward LSTM

ELMo (Embeddings from Language Models)

The First Breakthrough (2018) ELMo was the first major model to solve polysemy using **Deep LSTMs** (Recurrent Neural Networks/Long-Short Term Memory Networks).

ELMo reads the sentence twice simultaneously:

- **Pass 1 (Forward):** Reads left-to-right. Predicts the next word.
  - *Context:* "The quick brown..." → predicts "fox".
- **Pass 2 (Backward):** Reads right-to-left. Predicts the previous word.
  - *Context:* "...jumps over the dog" → predicts "fox".
- The vector for "fox" contains information from the start of the sentence AND the end of the sentence.

BERT (Bidirectional Encoder Representations)

The Limitation of ELMo ELMo is "Shallow Bidirectional." It effectively pastes two independent views together (Left view + Right view). It doesn't truly let the left side "see" the right side *during* the process.

BERT "True" Bidirectionality

BERT (Bidirectional Encoder Representations)

BERT uses a mechanism called **Self-Attention** (part of the Transformer architecture).

The Mechanism: It doesn't read left-to-right. It reads the **entire sentence at once**.
Every word looks at every other word simultaneously to determine its own meaning.

The "Masked" Training

For training: BERT uses a "Fill in the Blank" game

- **Input:** "The [MASK] brown fox jumps."
- **Goal:** Predict "quick".
- To guess "quick," the model MUST use context from *both* "The" (left) and "brown fox" (right) at the same time.

Computationally heavy, large memory footprint

Why BERT is computationally expensive

The parameter explosion

Word2Vec (Simple): A lookup table. If you have 10,000 words and 300 dimensions, that's roughly **3 million parameters**.
BERT Base (Complex): deep neural network with 12 layers of "Encoder Blocks."

- **Total Size: ~110 Million parameters**.
- **BERT Large: ~340 Million parameters**.

Impact: You need significantly more RAM just to *load* the model → run >>

The "Quadratic" Attention Problem $O(N^2)$

To understand a sentence, BERT's self-attention mechanism makes *every* word look at *every other* word.

- If a sentence has **10 words**, it calculates 10 x 10 = 100 interactions.
- If a sentence has **20 words**, it calculates 20 x 20 = 400 interactions.
- If a sentence has **512 words** (BERT's limit), it calculates **~262,000** interactions per layer!

Result: Doubling the sentence length **quadruples** the work.

Deep vs. Shallow

Word2Vec: A "Shallow" network (1 hidden layer). The signal passes through one matrix multiplication. It's instant.

BERT: A "Deep" network (12 or 24 layers). The signal must pass through 12 separate stages of processing.

- **Word2Vec:** Looking up a word in a dictionary (Instant).
- **BERT:** Reading a 12-page essay about the word to understand its nuance (Slow).

Now that the vectors are dense, they are generally uninterpretable and difficult to examine directly

Evaluating Vector Space Models

Evaluating Similarity

- Extrinsic (task-based, end-to-end) evaluation:
  - Question answering
  - Information retrieval
  - Sentiment analysis / Text categorization

- Intrinsic Evaluation
  - Test the *quality of the vectors in isolation*. Do they capture the "correct" meaning according to humans?

1. Synonym selection

- enormously
  - a. appropriately
  - b. uniquely
  - c. tremendously
  - d. decidedly
- provisions
  - a. stipulations
  - b. interrelations
  - c. jurisdictions
  - d. interpretations
- haphazardly
  - a. dangerously
  - b. densely
  - c. randomly
  - d. linearly
- prominent
  - a. battered
  - b. ancient
  - c. mysterious
  - d. conspicuous
- zenith
  - a. completion
  - b. pinnacle
  - c. outset
  - d. decline
- flawed
  - a. tiny
  - b. imperfect
  - c. lustrous
  - d. crude

TOEFL dataset
(test of English as a foreign language)

1. Synonym selection

- enormously
  - a. appropriately
  - b. uniquely
  - c. tremendously
  - d. decidedly

TOEFL dataset
(test of English as a foreign language)

1. Synonym selection

- enormously
  - a. appropriately
  - b. uniquely
  - c. tremendously
  - d. decidedly

vector(*enormously)*

TOEFL dataset
(test of English as a foreign language)

1. Synonym selection

- enormously
  - a. appropriately
  - b. uniquely
  - c. tremendously
  - d. decidedly

vector(*appropriately)*

vector(*enormously)*

TOEFL dataset
(test of English as a foreign language)

1. Synonym selection

- enormously
  - a. appropriately
  - b. uniquely
  - c. tremendously
  - d. decidedly

vector(*enormously)*

vector(*appropriately)*

vector(*uniquely)*

TOEFL dataset
(test of English as a foreign language)

1. Synonym selection

- enormously
  - a. appropriately
  - b. uniquely
  - c. tremendously
  - d. decidedly

vector(*enormously)*

vector(*appropriately)*

vector(*uniquely)*

vector(*tremendously)*

TOEFL dataset
(test of English as a foreign language)

1. Synonym selection

- enormously
  - a. appropriately
  - b. uniquely
  - c. tremendously
  - d. decidedly

vector(*enormously)*

vector(*appropriately)*

vector(*uniquely)*

vector(*tremendously)*

vector(*decidedly)*

TOEFL dataset
(test of English as a foreign language)

1. Synonym selection

- enormously
  - a. appropriately
  - b. uniquely
  - c. tremendously
  - d. decidedly

vector(*enormously)*

vector(*appropriately)*

vector(*uniquely)*

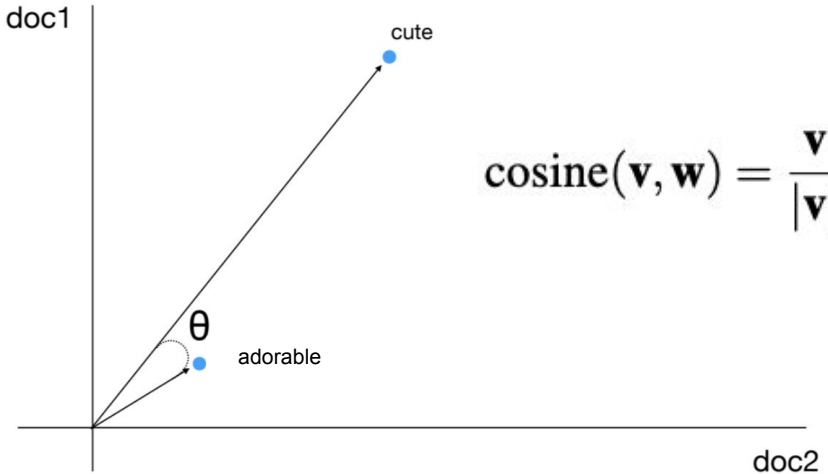
vector(*tremendously)*

vector(*decidedly)*

Select word with highest cosine similarity

TOEFL dataset
(test of English as a foreign language)

Cosine Similarity



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

Cosine similarity

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

	pie	data	computer
cherry	442	8	2
digital	5	1683	1670
information	5	3982	3325

$$\cos(\text{cherry},\text{information}) = \frac{442*5+8*3982+2*3325}{\sqrt{442^2+8^2+2^2}\sqrt{5^2+3982^2+3325^2}} = .018$$

$$\cos(\text{digital},\text{information}) = \frac{5*5+1683*3982+1670*3325}{\sqrt{5^2+1683^2+1670^2}\sqrt{5^2+3982^2+3325^2}} = .996$$

1. Synonym selection

- enormously
  - a. appropriately
  - b. uniquely
  - c. tremendously
  - d. decidedly

vector(*enormously)*

vector(*appropriately)*

vector(*uniquely)*

vector(*tremendously)*

vector(*decidedly)*

Select word with highest cosine similarity

TOEFL dataset
(test of English as a foreign language)

1. Synonym selection

- enormously
  - a. appropriately
  - b. uniquely
  - c. tremendously
  - d. decidedly

vector(*enormously)*

vector(*appropriately)*

vector(*uniquely)*

vector(*tremendously)*

vector(*decidedly)*

Select word with highest cosine similarity

TOEFL dataset
(test of English as a foreign language)

2. Similarity correlations

cat vs *kitten*

dandelion vs *seed*

café vs *lizard*

2. Similarity correlations

cat vs *kitten*

dandelion vs *seed*

café vs *lizard*

human similarity ratings

2. Similarity correlations

cat vs *kitten*

dandelion vs *seed*

café vs *lizard*

human similarity ratings

cos (vector(*cat*) , vector(*kitten*) **)**

cos (vector(*dandelion*) , vector(*seed*) **)**

cos (vector(*café*) , vector(*lizard*) **)**

2. Similarity correlations

Wordsim353: 353 noun pairs rated 0-10

cat vs *kitten*

dandelion vs *seed*

café vs *lizard*

human similarity ratings

cos (vector(*cat*) , vector(*kitten*) **)**

cos (vector(*dandelion*) , vector(*seed*) **)**

cos (vector(*café*) , vector(*lizard*) **)**

The WordSim353 Test

Gold Standard Dataset created by humans (Finkelstein et al., 2001).

- 353 pairs of nouns

The Human Task: Real people were asked to rate the similarity of these pairs on a scale from **0 (Totally Unrelated)** to **10 (Synonyms)**.

Examples from the dataset:

- *Tiger - Cat:* **7.35** (High similarity)
- *Tiger - Tiger:* **10.0** (Identical)
- *Book - Paper:* **7.46** (Related)
- *Computer - News:* **4.47** (Somewhat related)
- *Stock - Jaguar:* **0.92** (Unrelated)

The Model's Task

- We take our trained Word2Vec/GloVe model.
- For every pair in the list (e.g., *Tiger - Cat*), we calculate the **Cosine Similarity** between their vectors.
- The model gives us a score between -1 and 1 (e.g., **0.82**).

The WordSim353 Test

Calculating "Correlation" (The Grade)

- We now have two lists of scores:
  - **Human List:** `[7.35, 0.92, ...]`
  - **Model List:** `[0.82, 0.12, ...]`
- We measure the **Spearman Rank Correlation** between these two lists.
- **The Logic:** We don't care if the numbers match exactly. We only care about the **Ranking**.
  - *If humans say "A is more similar than B", the model must also say "A is more similar than B".*

Word Pair	Human Score (0-10)	Model Cosine Score (-1 to 1)	Rank Agreement?
Tiger - Cat	**7.35** (Rank 1)	**0.85** (Rank 1)	✅ Yes
Apple - Sun	**0.50** (Rank 3)	**0.10** (Rank 3)	✅ Yes
Plane - Car	**5.77** (Rank 2)	**0.60** (Rank 2)	✅ Yes

3. Analogies

Relationship between two words is captured as a consistent geometric direction.

A is to B as C is to ?

3. Analogies

big: biggest

small: _____

3. Analogies

big: biggest

small: _____

$$\text{vector}(\textit{biggest}) - \text{vector}(\textit{big}) + \text{vector}(\textit{small})$$

3. Analogies

big: biggest

small: _____

$$\text{vector}(\textit{biggest}) - \text{vector}(\textit{big}) + \text{vector}(\textit{small})$$

$$\mathbf{v}\_d \approx \mathbf{v}\_c - \mathbf{v}\_a + \mathbf{v}\_b$$

3. Analogies

big: biggest

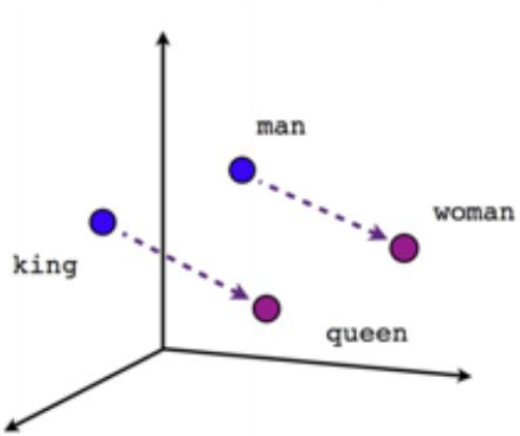
small: _____

$$\mathbf{v}\_d \approx \mathbf{v}\_c - \mathbf{v}\_a + \mathbf{v}\_b$$

vector(*biggest*) – vector(*big*) + vector(*small*)

$$\mathbf{v}\_\text{Queen} \approx \mathbf{v}\_\text{King} - \mathbf{v}\_\text{Man} + \mathbf{v}\_\text{Woman}$$

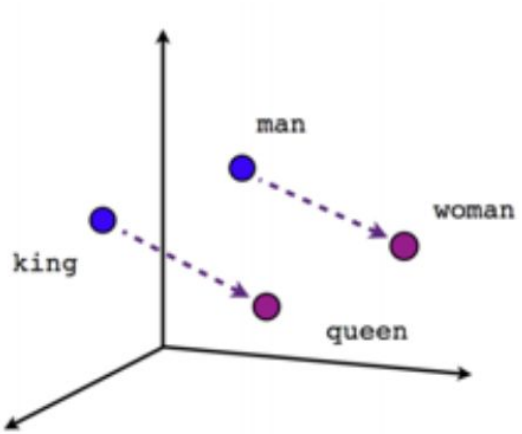
3. Analogies



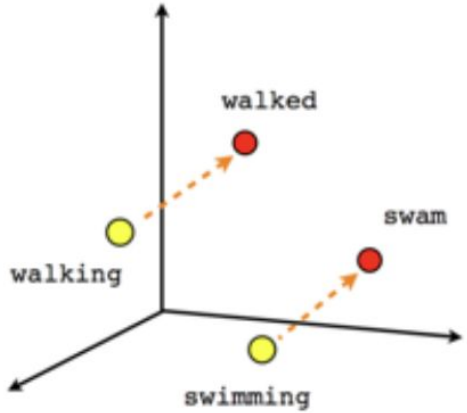
Male-Female

King is to Man as Queen is to...

3. Analogies

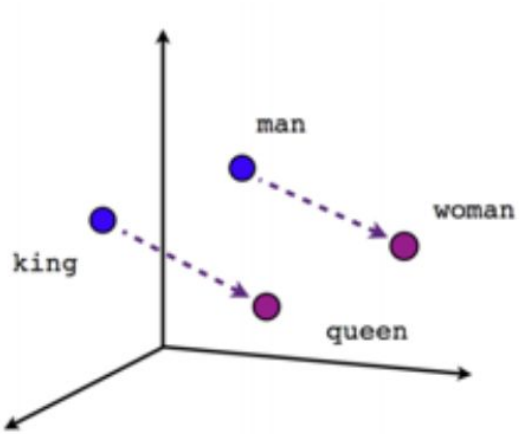


Male-Female

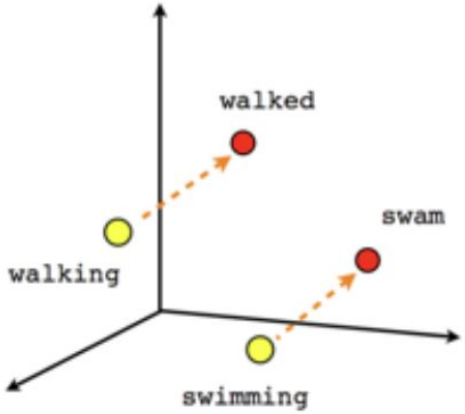


Verb tense

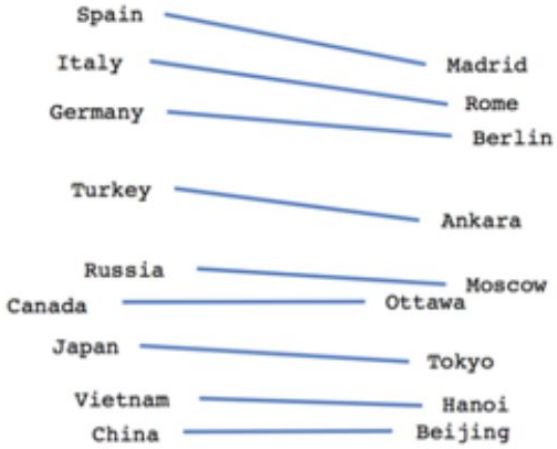
3. Analogies



Male-Female



Verb tense



Country-Capital

Table 1: *Examples of five types of semantic and nine types of syntactic questions in the Semantic-Syntactic Word Relationship test set.*

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

Mikolov et al. (2013)

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

Mikolov et al. (2013)

BUT:

big: biggest

small: _____

- You need to exclude ‘big’, ‘biggest’, ‘small’ as possible answers
- Generally only works for frequent words and not every type of analogy
- You can (relatively) often get the right answer:
  - Just by taking the word that is closest to ‘small’
  - Just by taking the word that is closest to both ‘small’ and ‘biggest’ (ignoring ‘big’)
- You (relatively) often don’t get the right answer:
  - If you flip it: biggest -> big as smallest -> small

Linzen (2016)

Biases

Black is to Criminal as Caucasian is to Police: Detecting and Removing Multiclass Bias in Word Embeddings

Thomas Manzini[†]*, **Yao Chong Lim**[‡]*, **Yulia Tsvetkov**[‡], **Alan W Black**[‡]
Microsoft AI Development Acceleration Program[†], Carnegie Mellon University[‡]
Thomas.Manzini@microsoft.com, {yaochonl,ytsvetko,awb}@cs.cmu.edu

Racial Analogies	
black → homeless	caucasian → servicemen
caucasian → hillbilly	asian → suburban
asian → laborer	black → landowner
Religious Analogies	
jew → greedy	muslim → powerless
christian → familial	muslim → warzone
muslim → uneducated	christian → intellectually

Table 1: Examples of racial and religious biases in analogies generated from word embeddings trained on the Reddit data from users from the USA.

When Analogies Reveal Bias

Researchers (Bolukbasi et al.) ran the analogy test on Google News vectors trained on massive datasets. They asked the model to complete professional analogies:

- *"Man is to Computer Programmer as Woman is to...?"*

Expected: "Software Engineer" or "Developer" (Gender neutral).
Actual Output: "Homemaker".
Other Examples found:

- *Father : Doctor :: Mother :* ***Nurse***
- *Man : Boss :: Woman :* ***Receptionist***

https://arxiv.org/pdf/1607.06520

When Analogies Reveal Bias

Researchers (Bolukbasi et al.) ran the analogy test on Google News vectors trained on massive datasets. They asked the model to complete professional analogies:

- *"Man is to Computer Programmer as Woman is to ...?"*

Expected: "Software Engineer" or "Developer"
Actual Output: "Homemaker".
Other Examples found:

- *Father : Doctor :: Mother :* ***Nurse***
- *Man : Boss :: Woman :* ***Receptionist***

It trained on 100 billion words of human text (news, books, internet) from the past 20-30 years.

Because humans historically wrote more about men being programmers and women being homemakers, the model learned this statistical correlation as a "fact" of language.

https://arxiv.org/pdf/1607.06520

Visualizing the bias -gender axis

We can identify a "Gender Direction" in the vector space by subtracting gendered pairs:

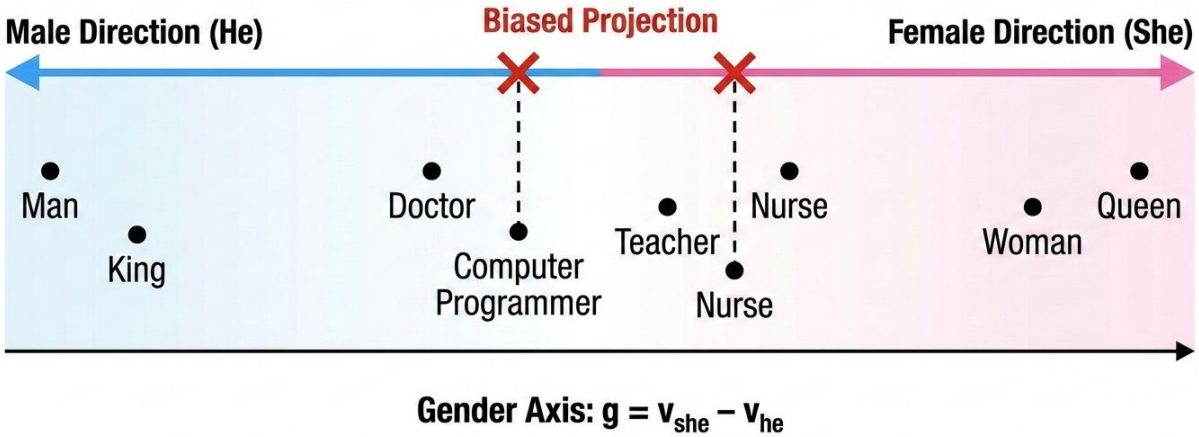
g = she - he

concept of "Gender"

The Projection Test

- We take "Neutral" words (like *Doctor, Nurse, Smart, Emotional*) and project them onto this Gender Axis.
- **Ideally:** They should be in the middle (Neutral).
- **Reality:**
  - *Doctor, Captain, Architect* → Heavily skewed toward **"He"**.
  - *Nurse, Teacher, Artist* --> Heavily skewed toward **"She"**.

Visualizing the Gender Axis in Word Embeddings (e.g., Word2Vec)



Visualizing the bias -gender axis

If you use these vectors in a Resume Sorting AI:

Visualizing the bias -gender axis

If you use these vectors in a Resume Sorting AI:

- The AI sees the word "Woman" and effectively subtracts points from "Programmer" because the vectors are far apart in the training data.
- This automates and amplifies existing societal discrimination.

Biases

- Embeddings reflect and magnify co-occurrences in the data
- Co-occurrences reflect social structure as reflected in text
- Text != Real world

Summary

- Word vectors/embeddings allow us to represent similarity between words
- Distributional hypothesis: similar words occur in similar contexts (documents, neighboring words)
- Term-term and term-document matrices
- These can contain binary values, counts, or TF-IDF / PPMI values (which help deal with the fact that raw counts can be problematic)

Summary

- Dense vectors are preferable to sparse vectors for reasons related to computational complexity and representational richness / generalizability
- Dense vectors can be created by doing dimensionality reduction on “count” matrices
- Or by adopting a new approach in which we train a classifier to predict whether word / context pairs co-occurred and taking the learned weights as our embeddings
- Static embedding: one vector per word (i.e., financial bank and river bank have the same vector), but stay tuned for contextualized word embeddings

Hard Debiasing Method

To mathematically force the model to be "fair" by removing the gender component from neutral words.

Identify the "Gender Axis"

We calculate the direction $\mathbf{g}$ by averaging the differences of definitional gender pairs:

$$\mathbf{g} \approx \sum (\mathbf{v}\_\text{she} - \mathbf{v}\_\text{he}) / n$$

This vector represents the direction of "gender" in the embedding space.

- For every word that *should* be neutral (e.g., "Doctor", "Programmer"), we remove its projection on the gender axis.
- Calculate how "gendered" the word is (The Bias Component): b = ($\mathbf{v}$_word $\cdot$ $\mathbf{g}$)
- Remove that component: $\mathbf{v}$_debiased = $\mathbf{v}$_word − (b × $\mathbf{g}$)

Hard Debiasing Method

Equalize

For gendered pairs (like *Boy* and *Girl*), we want to make sure they are equidistant to any neutral word (like *Math*).

$$\text{dist}(\mathbf{v}\_\text{math}, \mathbf{v}\_\text{boy}) = \text{dist}(\mathbf{v}\_\text{math}, \mathbf{v}\_\text{girl})$$

(Mathematically, this forces the neutral word to sit exactly on the "middle line" between the two gendered words.)

Soft Debiasing" Method (GN-GloVe)

- Hard debiasing is a post-processing step.
- It is often "superficial." Research showed that even after hard debiasing, "Doctor" and "Nurse" still clustered together in ways that correlated with gender

https://aclanthology.org/N19-1061.pdf

The Alternative: GN-GloVe (Gender-Neutral GloVe)

Training Objective:

- Standard: minimize the standard GloVe error (predict context).
- **PLUS** Minimize the squared distance between gendered pairs in the loss function.
- *Objective:* Ensure that $\mathbf{v}$**_waiter** and $\mathbf{v}$**_waitress** are treated identically when predicting neighbors.

The model learns from the start that gender is "noise" for most tasks, rather than a predictive signal

Can we remove bias

- Even if you make "Doctor" mathematically neutral to "He/She," the model might still associate "Doctor" with "Hospital," and "Hospital" might be biased toward "He."
- Bias is deeply entangled in the geometry; it's not just a single line you can delete.
- Bias is **latent** (hidden) in the relationships between thousands of words.

The Modern Approach (Contextual)

- Newer models like **BERT** and **GPT** are harder to debias because their vectors change dynamically based on the sentence.
- **Current Strategy:** Focus on **Data Curation** (cleaning the training set) rather than math tricks.
- We attempt to feed the model balanced text (e.g., "The doctor, she..." vs "The doctor, he...") so it learns fairness naturally.

Word Meaning

Two core issues from a NLP perspective:

- **Semantic similarity:** given two words, how similar are they in meaning?
- **Word sense disambiguation:** given a word that has more than one meaning, which one is used in a specific context?

“Big rig carrying fruit crashes on 210 Freeway, creates jam”

http://articles.latimes.com/2013/may/20/local/la-me-ln-big-rig-crash-20130520

How do we know that a word (lemma) has distinct senses?

- Linguists often design tests for this purpose
- e.g., **zeugma** combines distinct senses in an uncomfortable way

```
Which flight serves breakfast?

Which flights serve BWI?

*Which flights serve breakfast
and BWI?
```

What is a lemma?

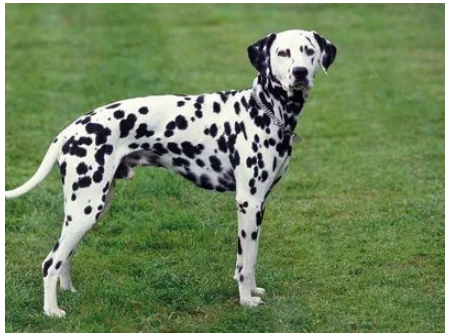
- Canonical form or dictionary form of a set of word forms
- Break -> breaking, breaks, broke, etc.
- Here, “break” is the lemma
- Breaking, breaks, broke are inflected forms
- You can think of it as the form of the word that would appear in the dictionary.

Word Senses

- “Word sense” = distinct meaning of a word/lemma
- Same word, different senses
  - Homonyms (homonymy): unrelated senses; identical orthographic form is coincidental
    - E.g., financial bank vs. river bank
  - Polysemes (polysemy): related, but distinct senses
    - E.g., financial bank vs. blood bank
  - Homographs: distinct senses, same orthographic form, but different pronunciation
    - E.g., bass (fish) vs. bass (instrument)

Relationship Between Senses

- ISA relationships:
  - From specific to general: hypernym
  - From general to specific: hyponym
  - E.g., dog is a hypernym of dalmatian
  - E.g., dalmatian is a hyponym of dog



- Part-whole relationships:
  - Wheel is a **meronym** of car (meronymy)
  - Car is a **holonym** of wheel (holonymy)

Don’t worry, you don’t need to remember all of these terms :)

How many senses of “drive”?

How many senses of “drive”?

1. *"We drive to the university every morning"* (operate or control a vehicle)
2. *"We drive the car to the garage"* (cause someone or something to move by driving)
3. *"He drives me mad"* (force into or from an action or state, either physically or metaphorically)
4. *"She is driven by her passion"* (to compel or force or urge relentlessly or exert coercive pressure on, or motivate strongly)
5. *"Drive a nail into the wall"* (push, propel, or press with force)
6. *" She is driving away at her doctoral thesis"* (strive and make an effort to reach a goal)
7. *"What are you driving at?" (*move into a desired direction of discourse)
8. *"My new truck drives well"* (have certain properties when driven)
9. *"She drives for the taxi company in Newark"* (work as a driver)
10. *"drive the cows into the barn"* (urge forward)
11. *"We drive the turnpike to work"* (proceed along in a vehicle)
12. *"drive a golf ball"* (strike with a driver, as in teeing off)

WordNet: A lexical database for English

https://wordnet.princeton.edu/

- Includes most English nouns, verbs, adjectives, adverbs
- Electronic format makes it amenable to automatic manipulation; used in many NLP applications
- “WordNets” generically refers to similar resources in other languages

Synonymy in WordNet

- WordNet is ordered in terms of “synsets”
  - Unordered set of roughly synonymous “words” (or multi-word phrases)
- Each synset expresses a distinct meaning/concept
- Based on human judgment (no “ground truth”)

WordNet

- https://wordnet.princeton.edu/
- NLTK https://www.nltk.org/howto/wordnet.html
- SpaCy https://spacy.io/

NLTK python example

```
from nltk.corpus import wordnet as wn

# 1. Get Synsets (Groups of synonyms)
synsets = wn.synsets('bank')
print(synsets)
# Output: [Synset('bank.n.01'), Synset('depository_financial_institution.n.01'), ...]

# 2. Get Definition of the first meaning
bank_finance = synsets[1]
print(bank_finance.definition())
# Output: "a financial institution that accepts deposits and channels the money..."

# 3. Get Hypernyms (Parent categories)
print(bank_finance.hypernyms())
# Output: [Synset('financial_institution.n.01')]
```

WordNet 3.0: Size

Part of speech	Word form	Synsets
Noun	117,798	82,115
Verb	11,529	13,767
Adjective	21,479	18,156
Adverb	4,481	3,621
Total	155,287	117,659

WordNet 3.0: Size

Part of speech	Word form	Synsets
Noun	117,798	82,115
Verb	11,529	13,767
		18,156
		3,621
Total	155,287	117,659

We reuse a small set of common verbs ("run", "set", "go", "take") to mean dozens of different things.

Word Sense Disambiguation

- Task: automatically select the correct sense of a word
  - Input: a word in context
  - Output: sense of the word

- Motivated by many applications:
  - Information retrieval
  - Machine translation
  - …

How big is the problem?

- **Most words in English have only one sense**
  - 62% in Longman's Dictionary of Contemporary English
  - 79% in WordNet
- But the others tend to have several senses
  - Average of 3.83 in LDOCE
  - Average of 2.96 in WordNet

- **Ambiguous words are more frequently used**
  - In the British National Corpus, 84% of instances have more than one sense
- **Some senses are more frequent than others**

Baseline Performance

- Baseline: most frequent sense
  - Equivalent to “take first sense” in WordNet
  - Does surprisingly well!

Freq	Synset	Gloss
338	plant[1], works, industrial plant	buildings for carrying on industrial labor
207	plant[2], flora, plant life	a living organism lacking the power of locomotion
2	plant[3]	something planted secretly for discovery by another
0	plant[4]	an actor situated in the audience whose acting is rehearsed but seems spontaneous to the audience

62% accuracy in this case!

Upper Bound Performance (ceiling)

- Upper bound
  - Fine-grained WordNet sense: 75-80% human agreement
  - Coarser-grained inventories: 90% human agreement possible

Simplest WSD algorithm: Lesk's Algorithm

- Intuition: note word overlap between context and dictionary entries
  - **Unsupervised,** but knowledge rich

The bank can guarantee deposits will eventually cover future tuition costs because it invests in adjustable-rate mortgage securities.

bank[1]	Gloss:	a financial institution that accepts deposits and channels the money into lending activities
	Examples:	"he cashed a check at the bank", "that bank holds the mortgage on my home"
bank[2]	Gloss:	sloping land (especially the slope beside a body of water)
	Examples:	"they pulled the canoe up on the bank", "he sat on the bank of the river and watched the currents"

Lesk's Algorithm

- Simplest implementation:
  - Count overlapping content words between glosses and context
- Lots of variants:
  - Include the examples in dictionary definitions
  - Include hypernyms and hyponyms
  - Give more weight to larger overlaps (e.g., bigrams)
  - Give extra weight to infrequent words
  - …

Terminology

- **Unsupervised learning:** Learning from unlabeled data
- **Supervised learning:** Learning from labeled data
- **Self-supervised learning:** Learning from labeled data (that is not generated by annotation; i.e., use the data itself as supervisory signal)

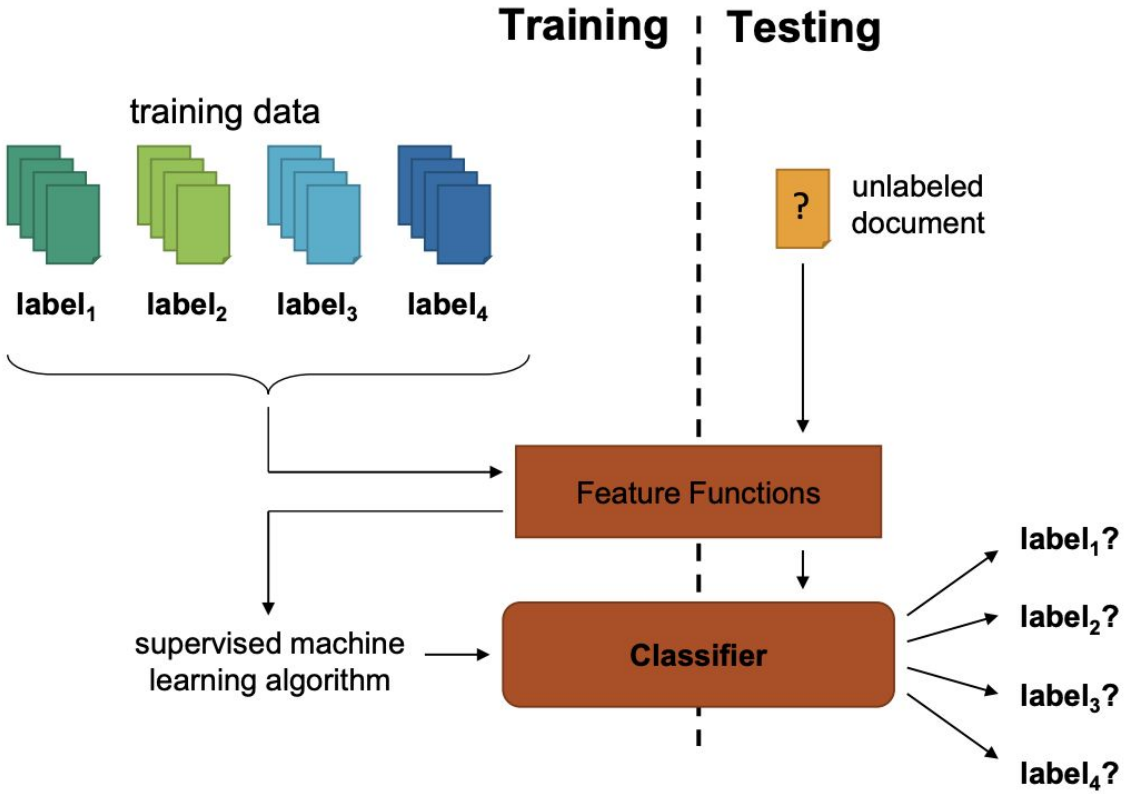
Lesk Algorithm?

Lesk Algorithm

Unsupervised/Knowledge-based

- Requires **Zero** training data
- You can run Lesk on a sentence immediately without ever showing the model a single example of "correct" disambiguation
- **Input:** Sentence + Dictionary
- **Output:** Best Sense

Alternative: WSD as **Supervised** Classification



WSD as supervised classification – Workflow

Input Take the sentence ***"The fisherman approached the bank."***
Feature Extraction: Convert the context into a numerical vector.

- *Is "fish" present?* [1]
- *Is "money" present?* [0]
- *Is previous word "the"?* [1]

Classifier: Feed this vector into an algorithm (Naive Bayes, SVM, or Neural Network).
Output: The classifier predicts the probability for each sense.

WSD as supervised classification – Workflow

The fisherman approached the bank

$P(\text{Sense} \mid \text{Context}) \propto P(\text{Context} \mid \text{Sense}) \cdot P(\text{Sense})$

Context: "fisherman", "approached"

Comparison:

- *Likelihood ("fisherman" | River Bank):* **High**
- *Likelihood ("fisherman" | Financial Bank):* **Low**

Result: Predict **Sense 2 (River)**

Other “tricks”

- One sense per discourse
  - If a polysemous word appears multiple times in a single document, it is extremely likely to have the same meaning every time.
- Word-aligned bilingual corpora

Summary

- Many words (lemmas) have multiple **senses** (meanings)
- The static vector embeddings we've seen so far do not take senses into account.
  - We get only one vector per word
- For many of our applications, we need to know which sense is being referred to (BERT)
  - For instance: how to translate "bank" to another language?
- Word sense disambiguation: given a word in context, output which sense is being used
  - Unsupervised and supervised approaches