EEP 596: LLMs: From Transformers to GPT ∥ Lecture 5̶ 6

Dr. Karthik Mohan

Univ. of Washington, Seattle

January 27, 2025

Deep Learning References

Deep Learning

Great reference for the theory and fundamentals of deep learning: Book by Goodfellow and Bengio et al Bengio et al
Deep Learning History

Embeddings

SBERT and its usefulness SBert Details

Last lecture

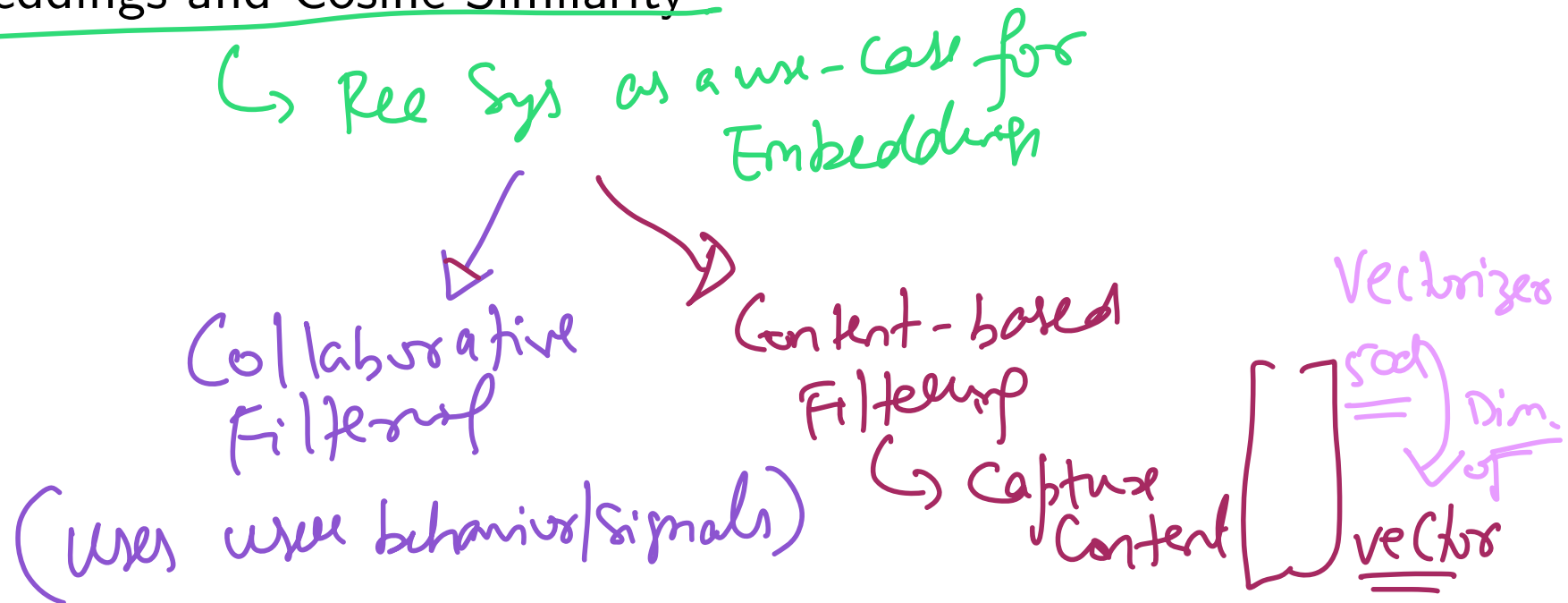- Training Deep Learning Model

Last lecture

- Training Deep Learning Model
- Back-propagation as a way of computing gradients

Last lecture

- Training Deep Learning Model
- Back-propagation as a way of computing gradients
- Hyper-parameter tuning

Last lecture

- Training Deep Learning Model
- Back-propagation as a way of computing gradients
- Hyper-parameter tuning
- Embeddings and Cosine Similarity

↳ Rec Sys as a use-case for Embedding

Collaborative Filtering
(uses user behavior/signals)

Content-based Filtering
↳ Capture Content

Vectorizer

[ ] = Dim. of vector

Objects

Words
Sentences
Movies
Books
  ↳ Chunks
  ↳ Pages

Vectorizer

Glove

Open AI
embedding

Sentence
Transformer

Vector

"eat" → [ ]₅₀₀

"I love chocolate"
→ [ ]₅₀₀

Store in a
vector DB → Retrieval

Semantic
search/
vector
search

Reference:-
Sentence

Candidates:-
Words,
page,chapters,
Categories

Retrieval &
Ranking

Last lecture

- Training Deep Learning Model
- Back-propagation as a way of computing gradients
- Hyper-parameter tuning
- Embeddings and Cosine Similarity
- Movie Recommendations, Cold Start and Content Based Recommendations

Last lecture

- Training Deep Learning Model
- Back-propagation as a way of computing gradients
- Hyper-parameter tuning
- Embeddings and Cosine Similarity
- Movie Recommendations, Cold Start and Content Based Recommendations
- Search demo through web app ✓

$$\text{"chocolate milk"} \quad \frac{[\ ] + [\ ]}{2} = \frac{\text{"milk chocolate"} \ [\ ] + [\ ]}{2}$$

Today's Lecture

- Quick Recap of Embeddings and Cosine Similarity
- Glove Embeddings
- Sentence Embeddings with Glove and Sentence Transformer
- In-Class Coding Exercise (second half)

LLM Market-Disruption

DeepSeek developments shakes up stock prices

Benchmark performance of open-source DeepSeek matching that of top
LLMs on reasoning with less cost, parameters, while also being
open-source.

Market Summary > NVIDIA Corp

118.58 USD

−24.04 (16.86%) ↓ today

Closed: Jan 27, 4:10 PM EST • Disclaimer
After hours 119.69 +1.11 (0.94%)

| 1D | 5D | 1M | 6M | YTD | 1Y | 5Y | Max |

118.23 USD 3:55 PM

Previous close 142.62

Open	124.80	Mkt cap	2.90T	52-wk high	153.13
High	128.40	P/E ratio	46.72	52-wk low	60.70
Low	116.70	Div yield	0.034%		

Handwritten annotations:
4) RL for Train as SFT

Reasoning

Dip:-
1) Low Budget $6MM
2) Hype
3) open source

118

Recap of Cosine Similarity in Embeddings

In-Class Exercise 1 on Cosine Similarity - Work in groups of 3

$$\cos(v_1 v_2) = \frac{v_1 \cdot v_2}{||v_1||_2 \, ||v||_2}$$

Let's reference back to the last lecture. Let's consider three dimensional embeddings for movies. Say we have 3 movies: Avatar, Ironman and Rainman. Given that you like Avatar, which movie would be good to recommend between Ironman and Rainman and why? Use the concept of embeddings and cosine similarity to derive your result.

Let's say Avatar's embedding is $e_1 = [1, 2, 2]$, Ironman's embedding is $e_2 = [3, 7, 8]$ and Rainman's embedding is $e_3 = [1, -2, 6]$.

$$e_1 = \begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix}_{3d} \qquad e_2 = \begin{bmatrix} 3 \\ 7 \\ 8 \end{bmatrix} \quad \|e_2\| =$$

$$\underset{A}{} \qquad \qquad \underset{3d}{T} \qquad \underset{3d}{} \sqrt{9+49+64}$$

$$= \sqrt{122}$$

$$\|e_1\|_2 = \sqrt{1+4+4} \qquad e_3 = \begin{bmatrix} 1 \\ -2 \\ 6 \end{bmatrix} \quad \underset{R}{\|e_3\|_2}$$

$$= \sqrt{17}$$

$$\text{Cosine Sim}(e_1, e_2) = \frac{e_1 \cdot e_2}{\|e_1\|_2 \|e_2\|_2} \qquad = \sqrt{1^2 + (-2)^2 + 6^2}$$

$$= \sqrt{41}$$

$$= \frac{33}{\sqrt{17}\sqrt{122}} \quad \checkmark$$

$$\text{Cosine Sim}(e_1, e_3) = \frac{e_1 \cdot e_3}{(\ )}$$

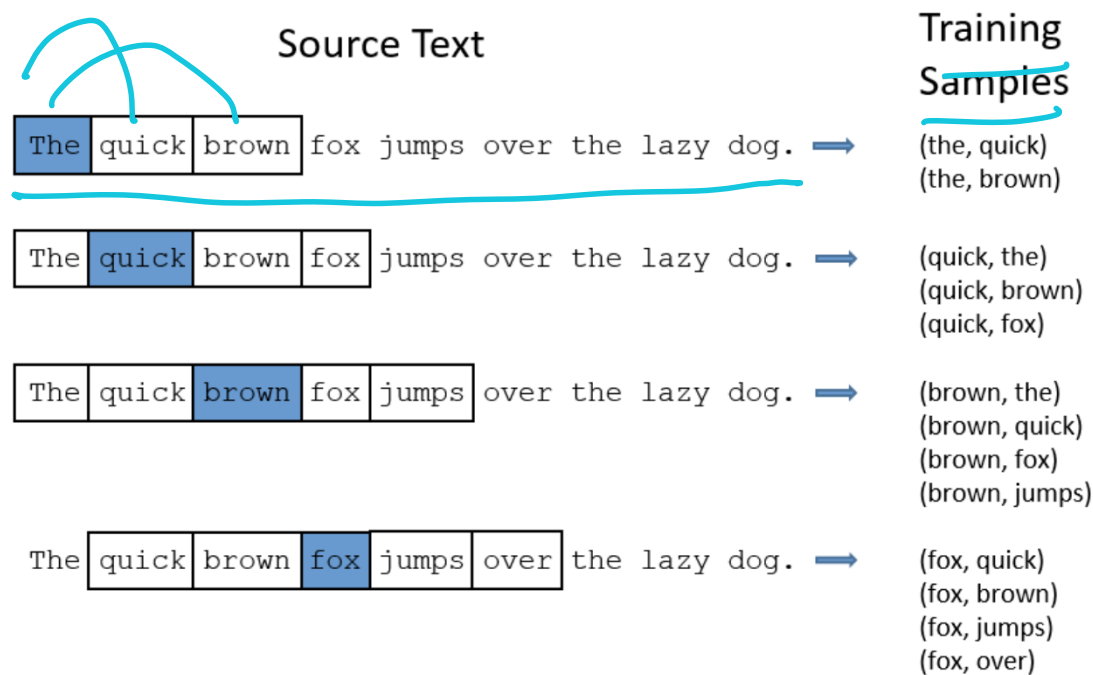$$= \frac{1 \cdot 1 - 2 \cdot 2 + 6 \cdot 2}{(\ )(\ )}$$

$$= \frac{9}{\sqrt{17}\sqrt{41}}$$
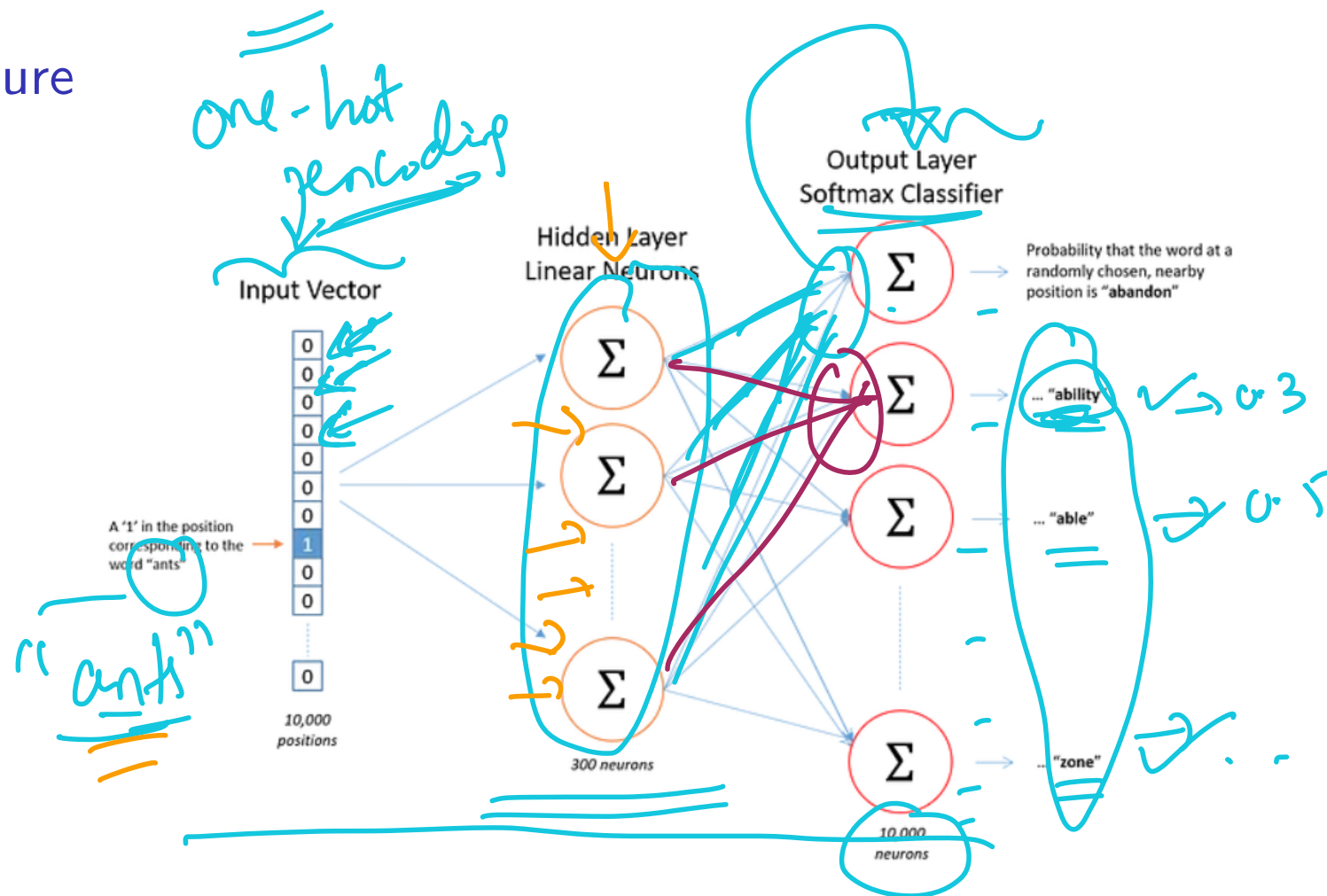
Word2Vec

→ one of the first "vectorizer" model

Skip Gram Model

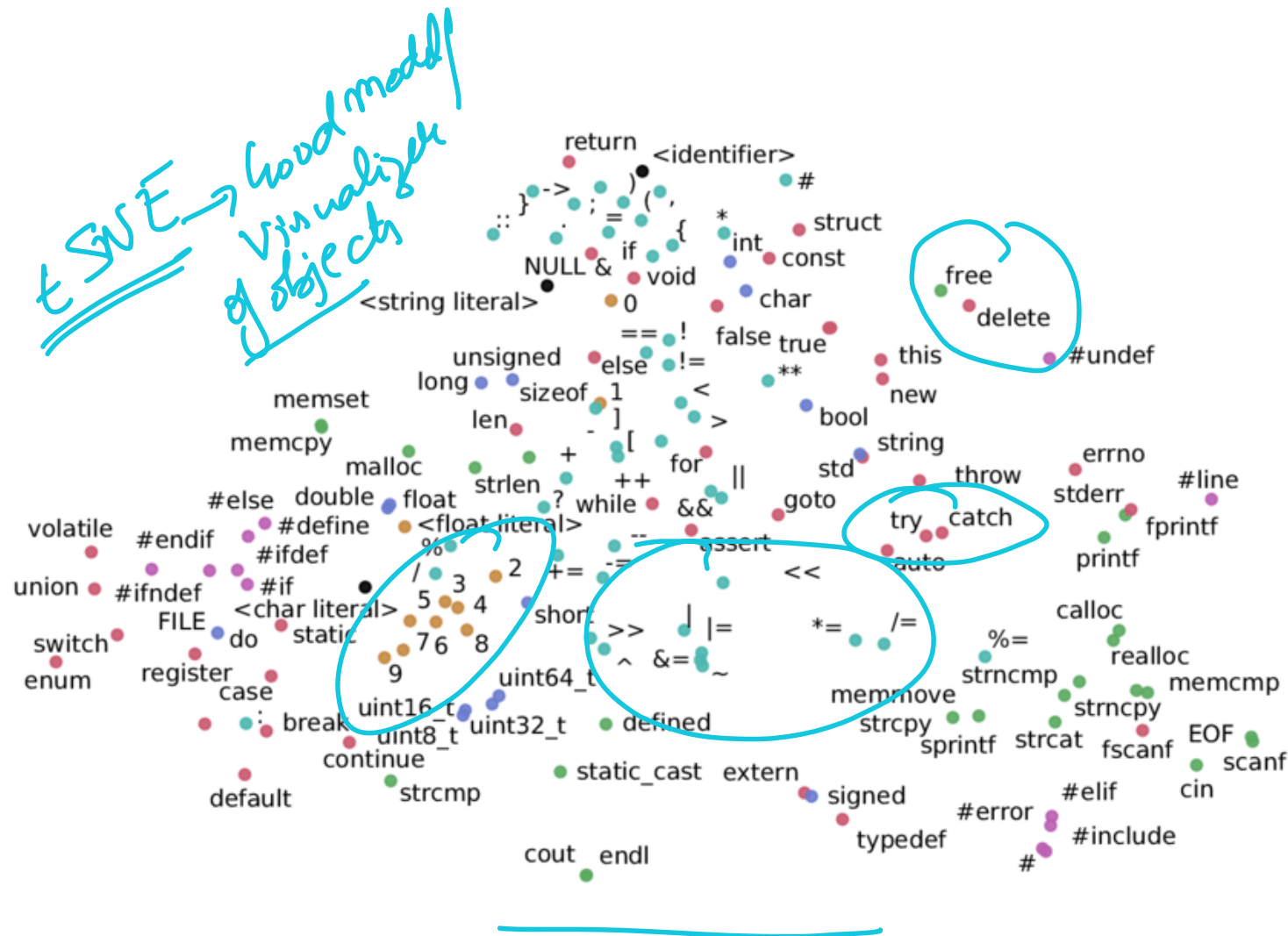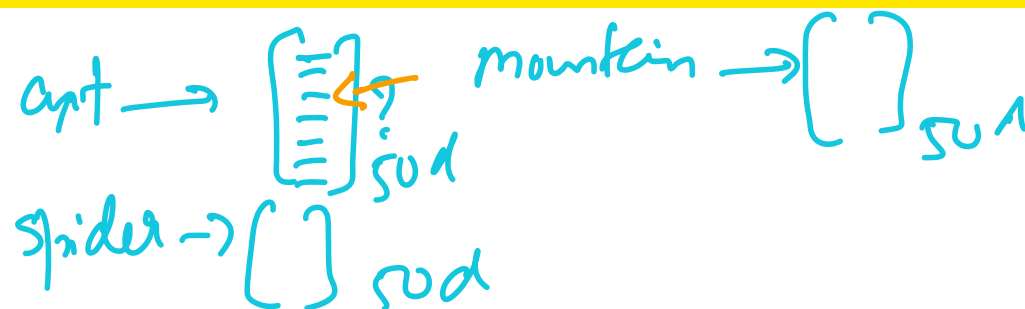Is based on the skip-gram model! How is training done? It's semi-supervised!!

Source Text | Training Samples

The **quick** brown fox jumps over the lazy dog. ⟹ (the, quick) (the, brown)

The **quick** brown fox jumps over the lazy dog. ⟹ (quick, the) (quick, brown) (quick, fox)

The quick **brown** fox jumps over the lazy dog. ⟹ (brown, the) (brown, quick) (brown, fox) (brown, jumps)

The quick brown **fox** jumps over the lazy dog. ⟹ (fox, quick) (fox, brown) (fox, jumps) (fox, over)

Word2Vec

Architecture

ICE #2

(handwritten annotations)
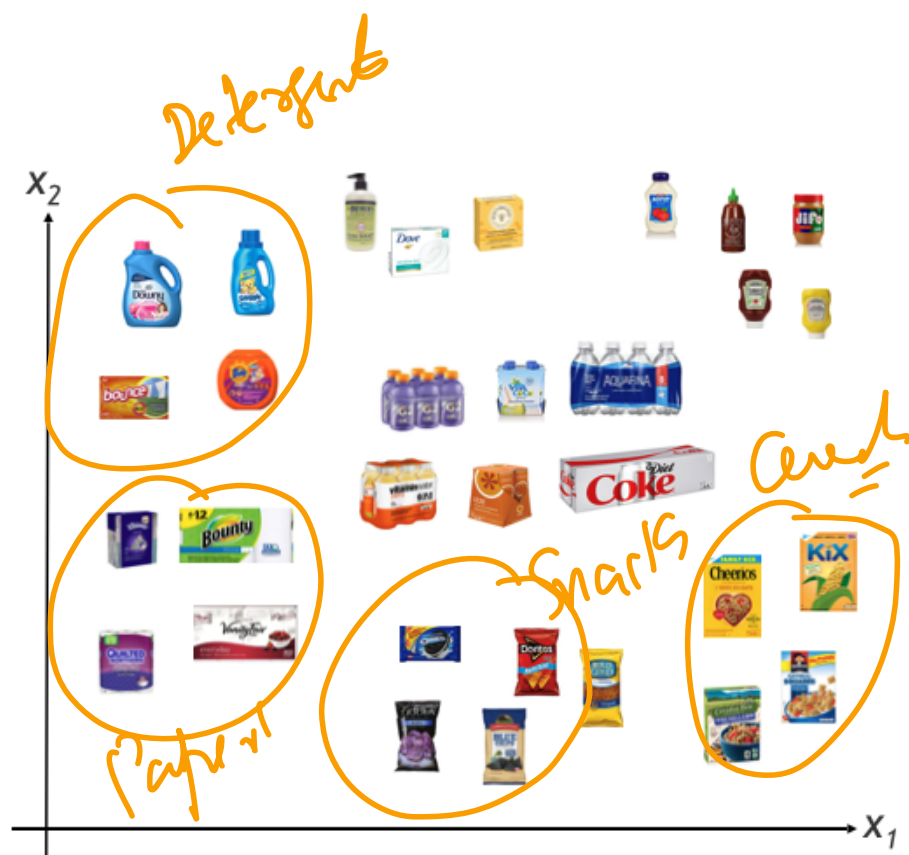
cat → [...] 50 ↑

mountain → [ ] 50 ↑

spider → [ ] 50 d

What do the embedding dimensions of word2vec represent?

1. Fixed words decided by word2vec
2. Topics that are common among the words
3. Parts of speech of the words (nouns, adjectives, etc)
4. Book titles that these words came from
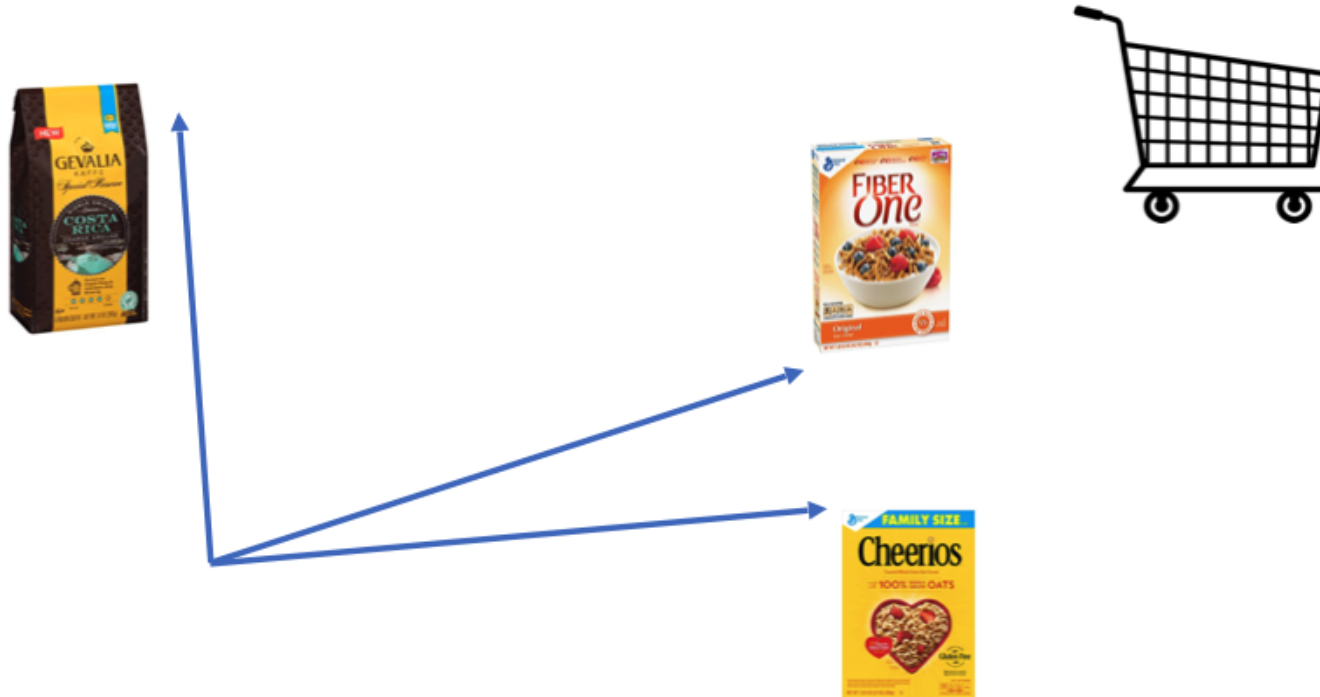
Product2Vec

Represent products in product space with a large matrix of embedding coordinate vectors "**L**"
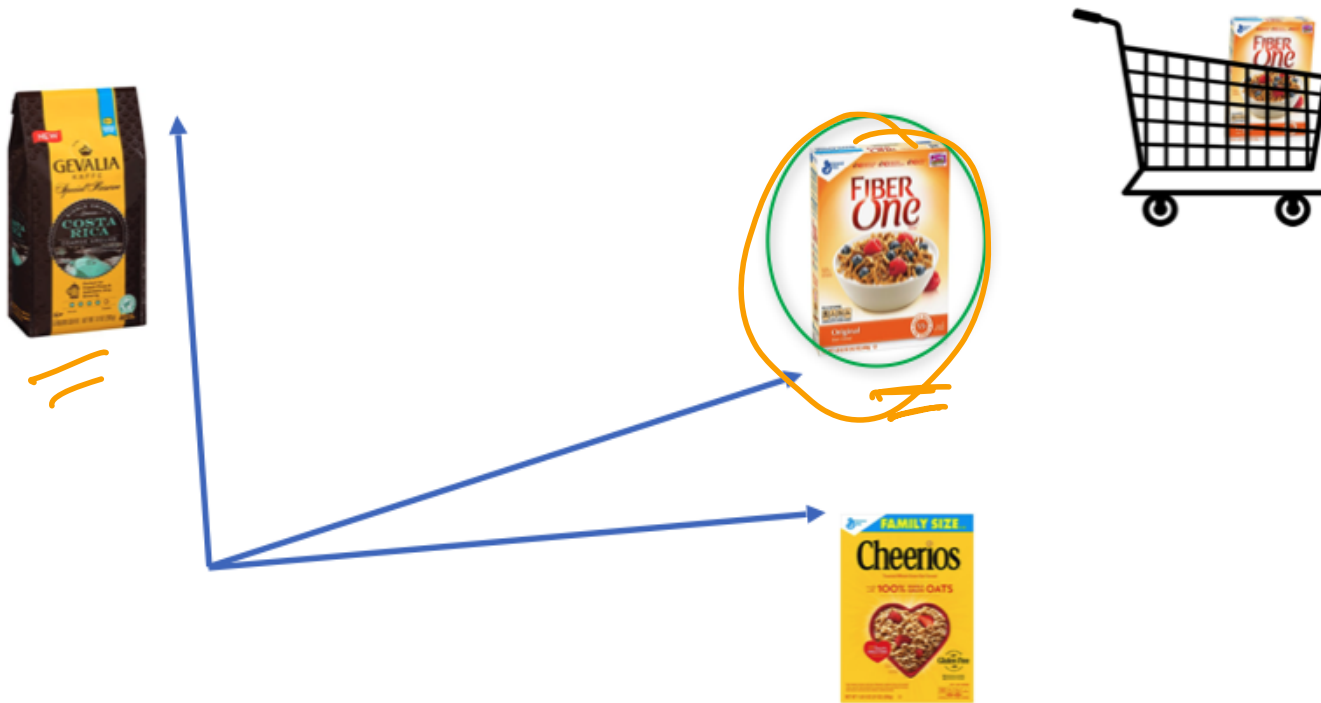
$$L = \begin{pmatrix} 1.5 & 1.9 & 1.8 & 1.4 & \cdots & 0.4 \\ 0.6 & 0.1 & 1.0 & 1.6 & \cdots & 1.9 \\ 0.6 & 1.6 & 1.6 & 1.6 & \cdots & 1.8 \\ 0.6 & 1.0 & 0.1 & 1.6 & \cdots & 0.6 \\ 0.8 & 1.4 & 1.9 & 0.8 & \cdots & 0.7 \end{pmatrix}$$

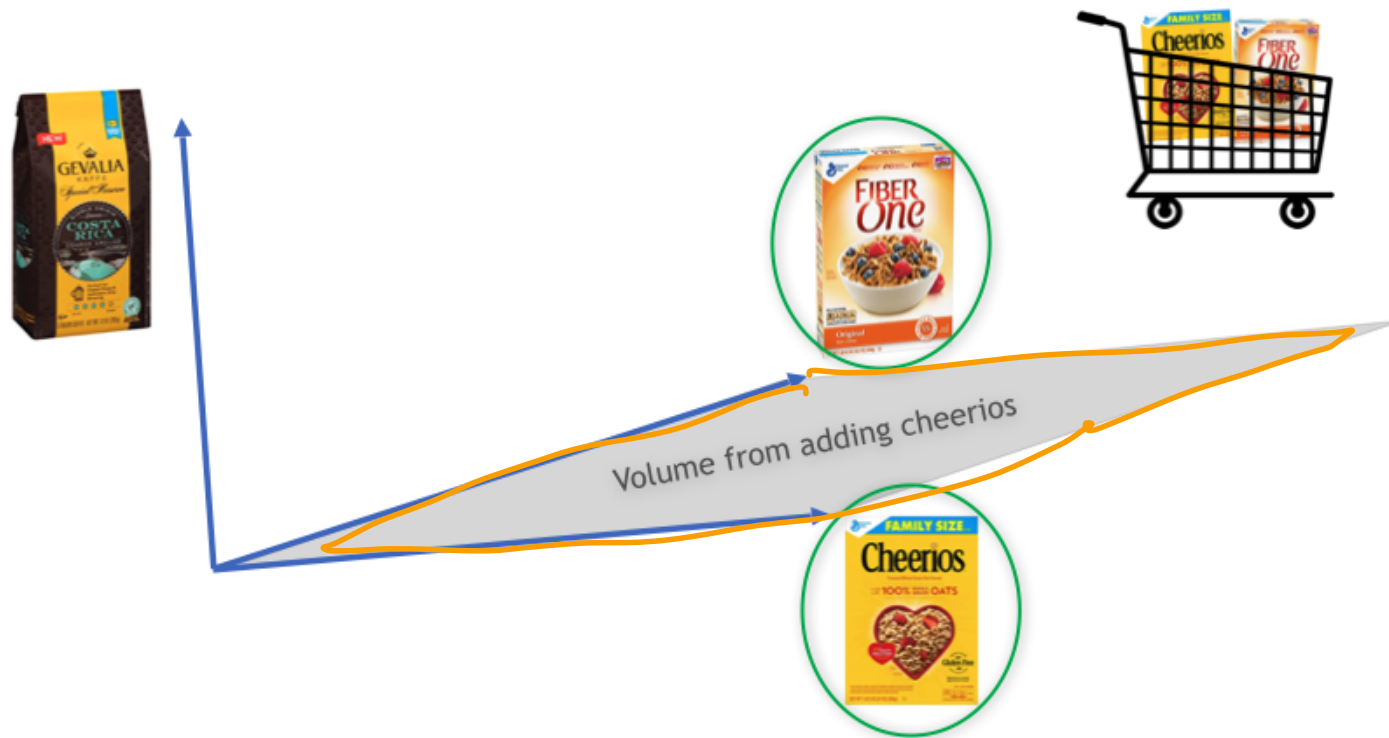We obtain these embedding vectors from the Product2Vec service [London et al, 2017]
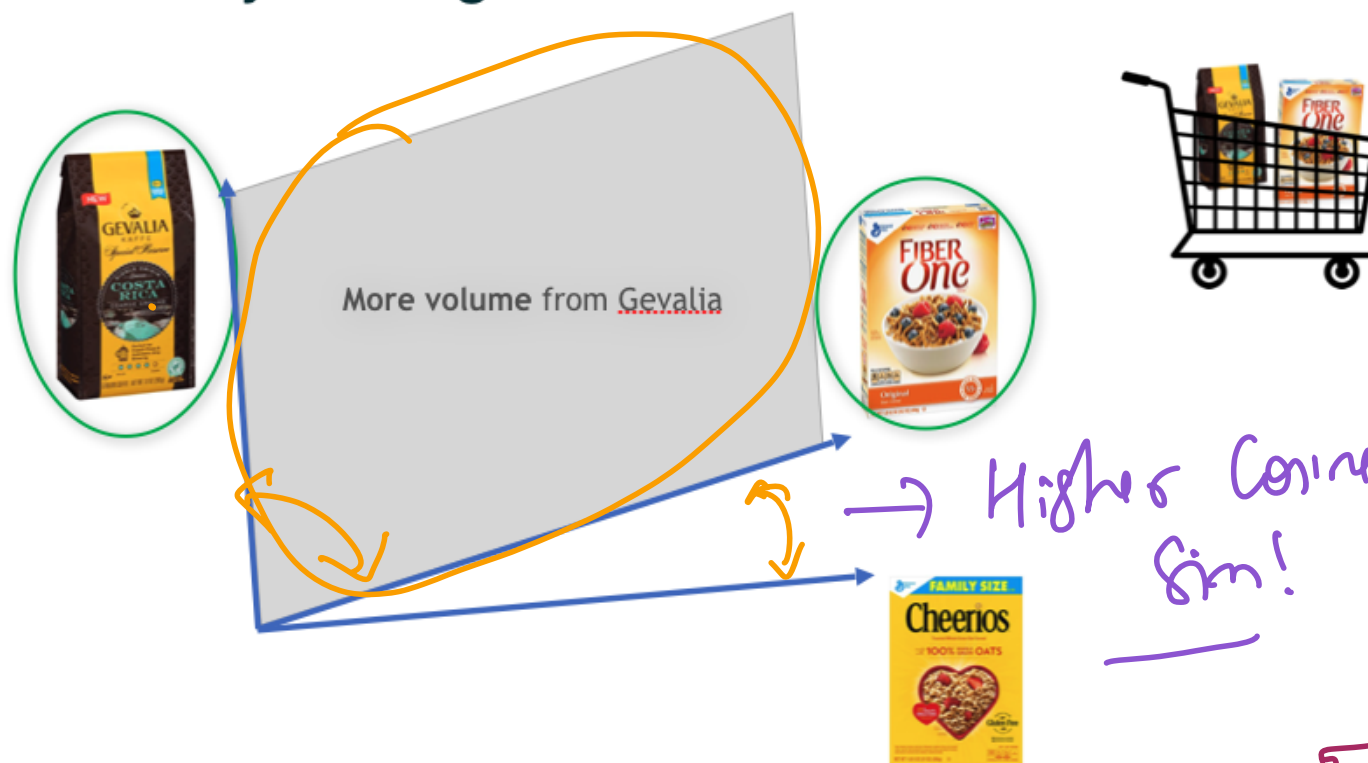
Product2Vec application

Product2Vec application

Product2Vec application

Product2Vec application

More volume from Gevalia

→ Higher Cosine Sim!

Low Diversity!!
Low Volume

Diversity measured thou Volume!!

Breakout 1: Discuss your favorite X2Vec!

X2Vec

In your group - Discuss an application that requires machine learning. Be specific about it - Example, data, features, the type of problem (classificaiton, clustering, etc). Can you see how X2Vec would benefit your application. What would be your X in this case? How would you learn X2vec for your application? And how would you use it?

Let's list out some X's in X2Vec!
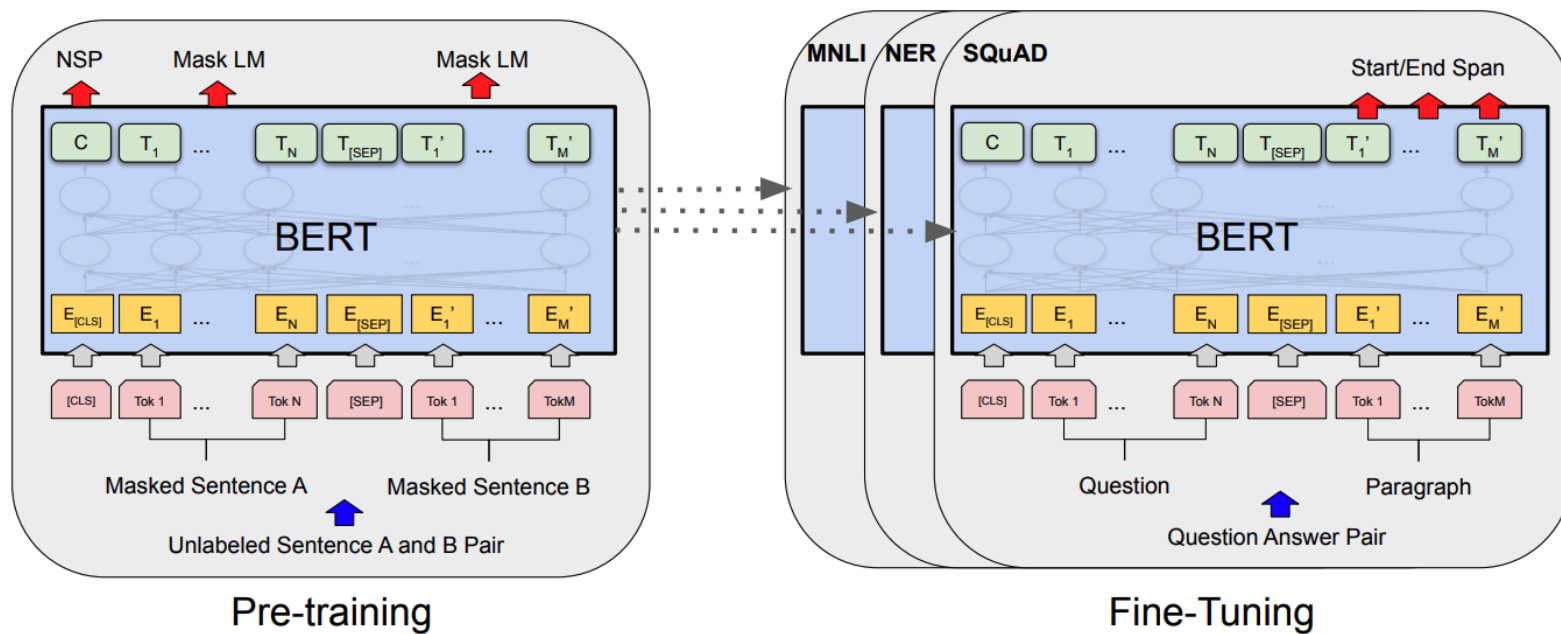
Generating Sentence Embeddings from Glove

Averaging embeddings of words: If we have a word embedding, how do we generate the sentence embedding?
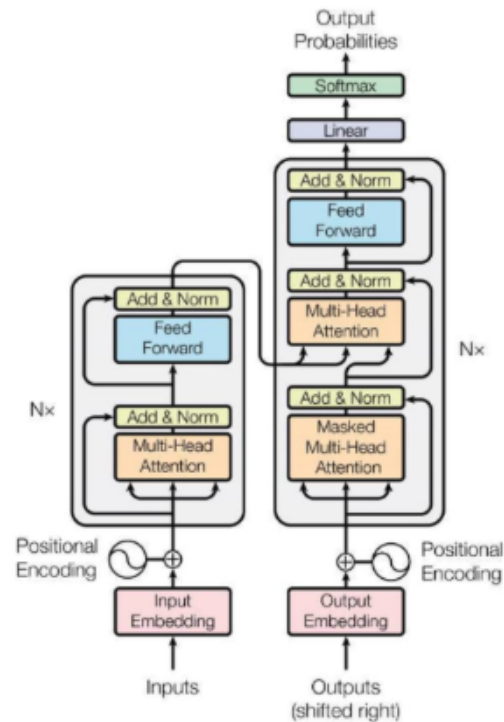
Generating Sentence Embeddings from Glove

Averaging embeddings of words: If we have a word embedding, how do we generate the sentence embedding?

Simple Solution: Just average the word embeddings

BERT - Bi-directional Encoders from Transformers

BERT Embeddings

BERT pre-training

Two Tasks

1. **Masked LM Model:** Mask a word in the middle of a sentence and have BERT predict the masked word

2. **Next-sentence prediction:** Predict the next sentence - Use both positive and negative labels. How are these generated?

BERT pre-training

Two Tasks

1. **Masked LM Model:** Mask a word in the middle of a sentence and have BERT predict the masked word

2. **Next-sentence prediction:** Predict the next sentence - Use both positive and negative labels. How are these generated?

ICE: Supervised or Un-supervised?

1. Are the above two tasks supervised or un-supervised?

BERT pre-training

Two Tasks

1. **Masked LM Model:** Mask a word in the middle of a sentence and have BERT predict the masked word
2. **Next-sentence prediction:** Predict the next sentence - Use both positive and negative labels. How are these generated?

ICE: Supervised or Un-supervised?

1. Are the above two tasks supervised or un-supervised?

Data set!

English Wikipedia and book corpus documents!

BERT - Bi-directional Encoders from Transformers

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**Average** -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
BERT$_{BASE}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
BERT$_{LARGE}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

Let's work on an in-class coding exercise