CSCE 689: Special Topics in Trustworthy NLP

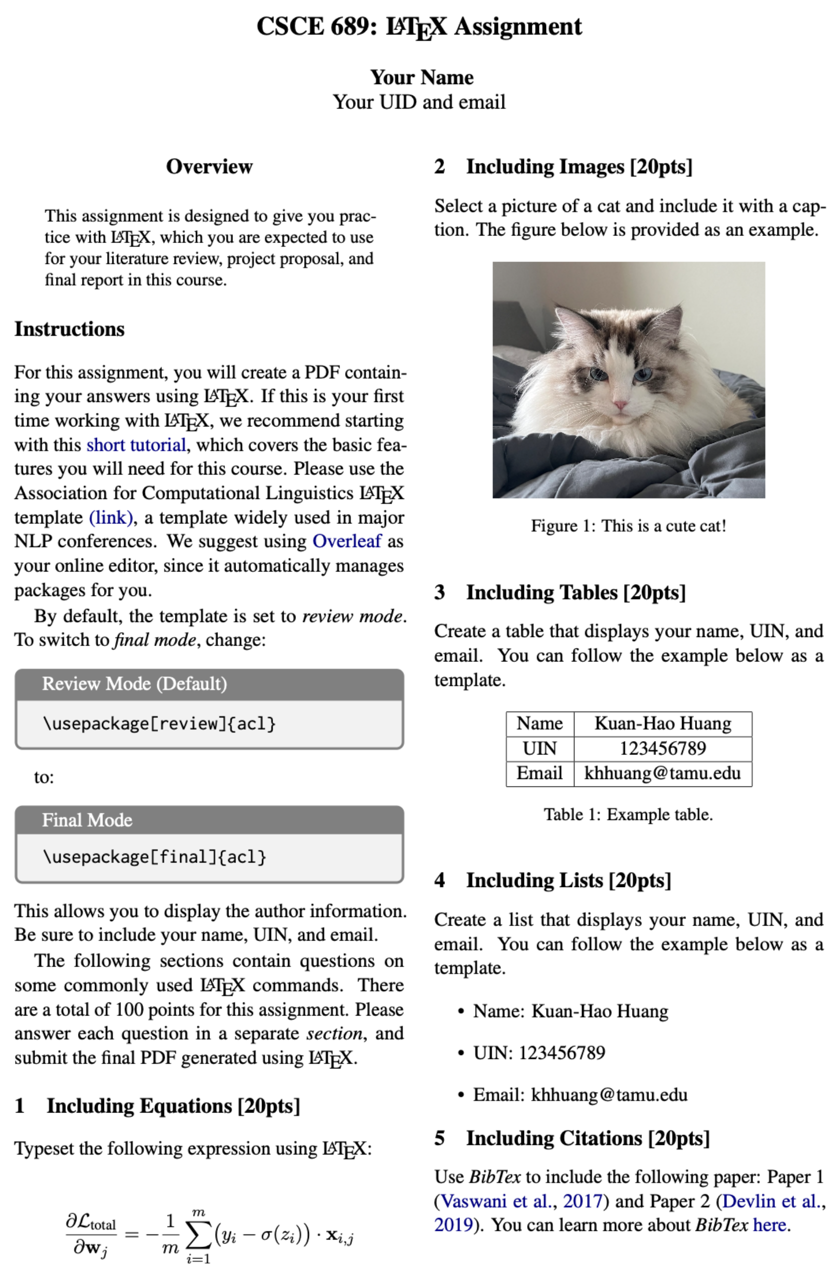
Lecture 3: Word Representations, Language Modeling

Kuan-Hao Huang
khhuang@tamu.edu

(Some slides adapted from Chris Manning, Dan Jurafsky, Danqi Chen, Karthik Narasimhan, Graham Neubig, and Greg Durrett)

LaTeX Assignment

- LaTeX Assignment (1%)
- Due: Sep 11, 11:59pm



CSCE 689: LaTeX Assignment

Your Name
Your UID and email

Overview

This assignment is designed to give you practice with LaTeX, which you are expected to use for your literature review, project proposal, and final report in this course.

Instructions

For this assignment, you will create a PDF containing your answers using LaTeX. If this is your first time working with LaTeX, we recommend starting with this short tutorial, which covers the basic features you will need for this course. Please use the Association for Computational Linguistics LaTeX template (link), a template widely used in major NLP conferences. We suggest using Overleaf as your online editor, since it automatically manages packages for you.

By default, the template is set to *review mode*. To switch to *final mode*, change:

Review Mode (Default)

```
\usepackage[review]{acl}
```

to:

Final Mode

```
\usepackage[final]{acl}
```

This allows you to display the author information. Be sure to include your name, UIN, and email.

The following sections contain questions on some commonly used LaTeX commands. There are a total of 100 points for this assignment. Please answer each question in a separate *section*, and submit the final PDF generated using LaTeX.

1 Including Equations [20pts]

Typeset the following expression using LaTeX:

$$\frac{\partial \mathcal{L}_{\text{total}}}{\partial \mathbf{w}_j} = -\frac{1}{m}\sum_{i=1}^{m}\big(y_i - \sigma(z_i)\big)\cdot \mathbf{x}_{i,j}$$

2 Including Images [20pts]

Select a picture of a cat and include it with a caption. The figure below is provided as an example.

Figure 1: This is a cute cat!

3 Including Tables [20pts]

Create a table that displays your name, UIN, and email. You can follow the example below as a template.

Name	Kuan-Hao Huang
UIN	123456789
Email	khhuang@tamu.edu

Table 1: Example table.

4 Including Lists [20pts]

Create a list that displays your name, UIN, and email. You can follow the example below as a template.

- Name: Kuan-Hao Huang
- UIN: 123456789
- Email: khhuang@tamu.edu

5 Including Citations [20pts]

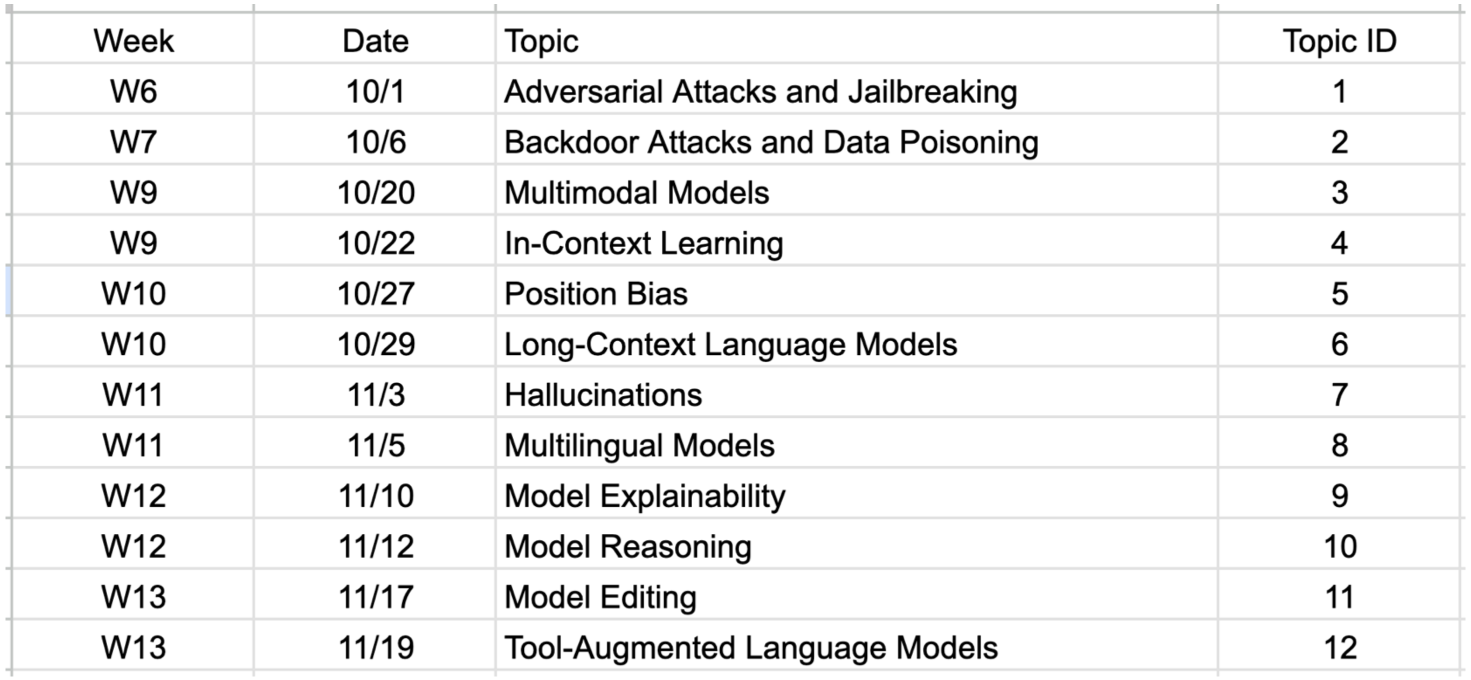
Use *BibTex* to include the following paper: Paper 1 (Vaswani et al., 2017) and Paper 2 (Devlin et al., 2019). You can learn more about *BibTex* here.

Topic Study

- Topic Study (30%)
  - Literature Review (15%) [Due: 10/2]
  - Topic Presentation (15%)
    - Email your slides to the instructor at least 2 days before your presentation

Topic Sign-Up

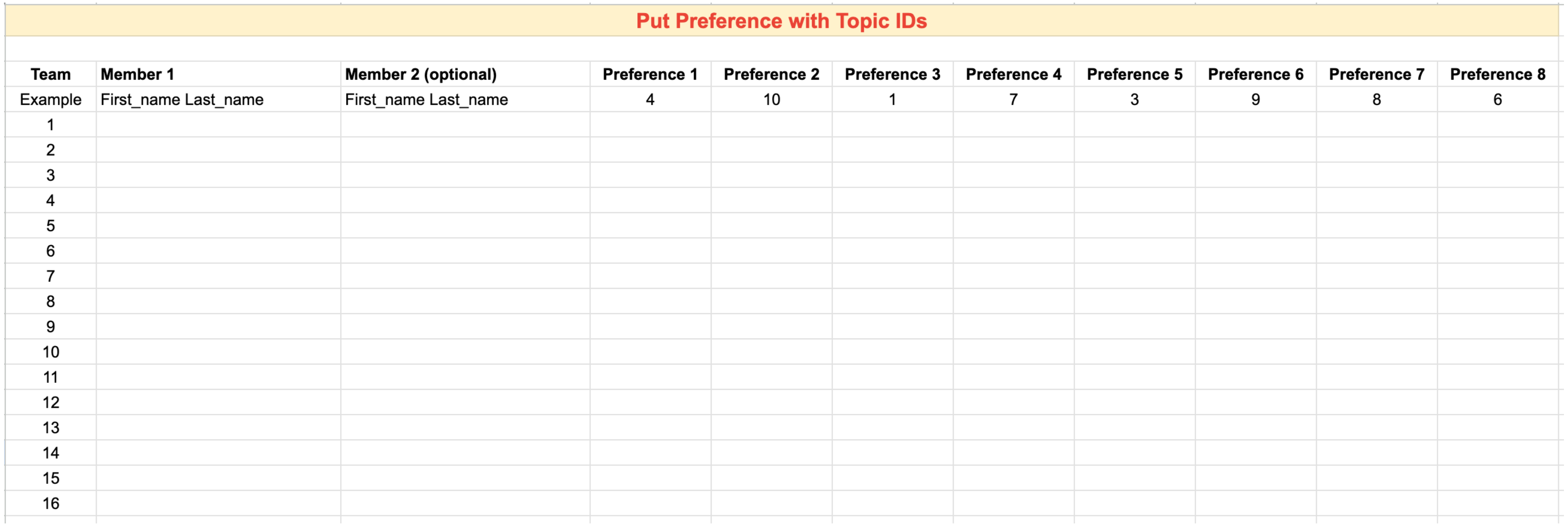
- 16 students
- 12 topics
- Each team can have 1 or 2 people



Week	Date	Topic	Topic ID
W6	10/1	Adversarial Attacks and Jailbreaking	1
W7	10/6	Backdoor Attacks and Data Poisoning	2
W9	10/20	Multimodal Models	3
W9	10/22	In-Context Learning	4
W10	10/27	Position Bias	5
W10	10/29	Long-Context Language Models	6
W11	11/3	Hallucinations	7
W11	11/5	Multilingual Models	8
W12	11/10	Model Explainability	9
W12	11/12	Model Reasoning	10
W13	11/17	Model Editing	11
W13	11/19	Tool-Augmented Language Models	12

Topic Sign-Up

- Sign-up: https://tinyurl.com/2p9mr2wa
  - Log in with TAMU account
  - Due: Sep 10 before lecture



Put Preference with Topic IDs

Team	Member 1	Member 2 (optional)	Preference 1	Preference 2	Preference 3	Preference 4	Preference 5	Preference 6	Preference 7	Preference 8
Example	First_name Last_name	First_name Last_name	4	10	1	7	3	9	8	6
1										
2										
3										
4										
5										
6										
7										
8										
9										
10										
11										
12										
13										
14										
15										
16										

Topic Sign-Up

- We will finalize the topic assignment in class on Sep 10
- Decision Process:
    - Preferences will be handled in order (Preference 1 → Preference 2 → …)
    - If you are the only one choosing a topic, you get it
    - If multiple teams choose the same topic, we will draw a lottery
    - Move on to the next preference order
- So be strategic when choosing your preferences

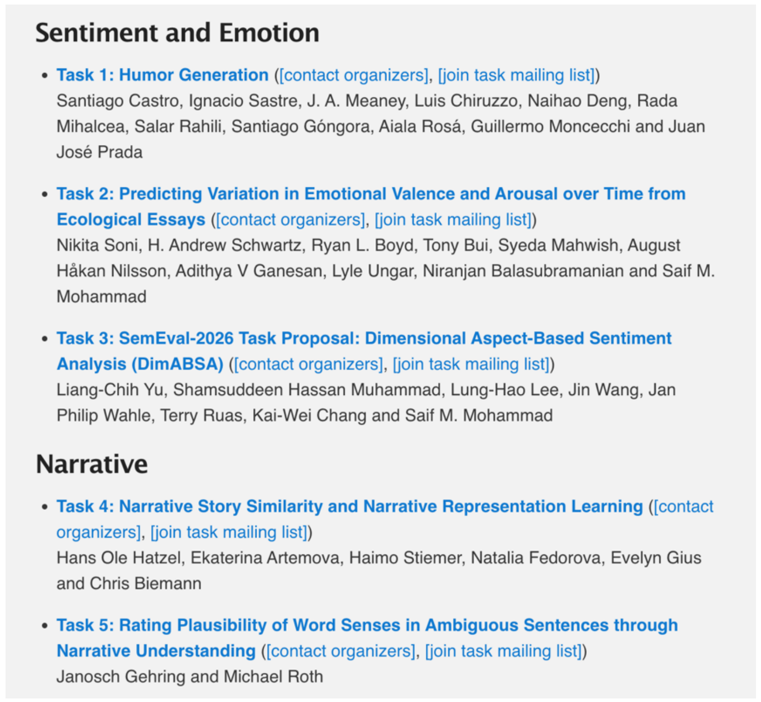
About Teams

- Topic Study (30%) (a team of 1 or 2 people)
  - Literature Review (15%)
  - Topic Presentation (15%)
- Course Project (49%) (a team of 1 or 2 people)
  - Project Proposal (5%)
  - Project Highlight Presentation (5%)
  - Midterm Report (10%)
  - Final Presentation (12%)
  - Final Report (17%)

Not necessary to be the same team

If You Are Looking for Course Project Topics

- The 20th International Workshop on Semantic Evaluation (SemEval-2026)
  - https://semeval.github.io/SemEval2026/tasks

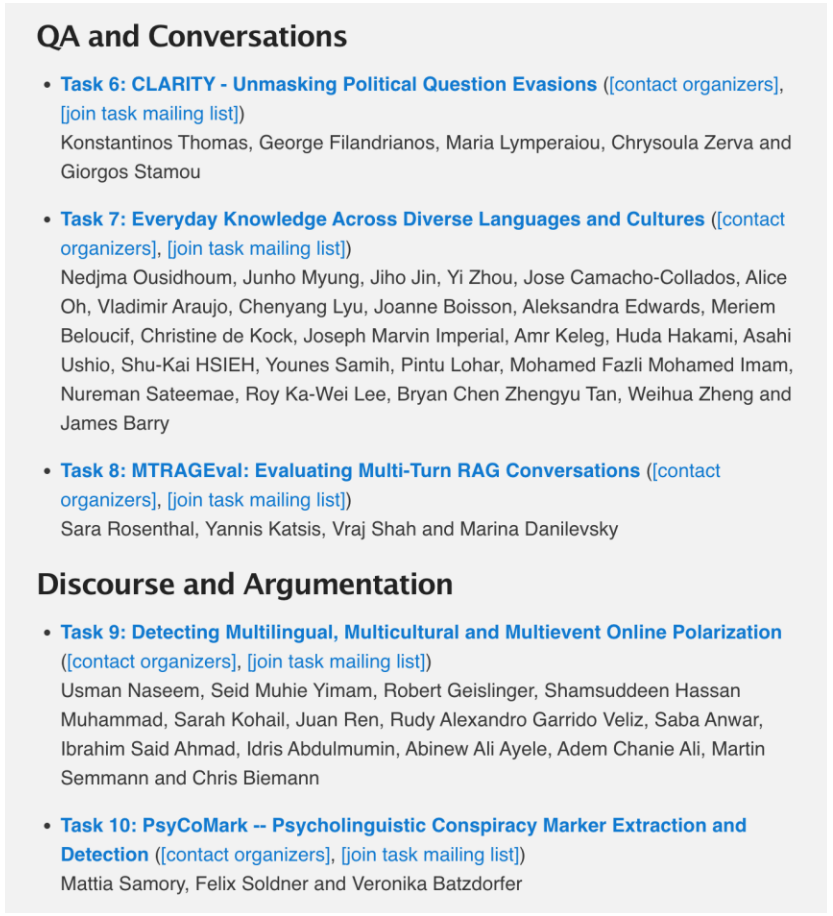


Sentiment and Emotion

- **Task 1: Humor Generation** ([contact organizers], [join task mailing list])
  Santiago Castro, Ignacio Sastre, J. A. Meaney, Luis Chiruzzo, Naihao Deng, Rada Mihalcea, Salar Rahili, Santiago Góngora, Aiala Rosá, Guillermo Moncecchi and Juan José Prada
- **Task 2: Predicting Variation in Emotional Valence and Arousal over Time from Ecological Essays** ([contact organizers], [join task mailing list])
  Nikita Soni, H. Andrew Schwartz, Ryan L. Boyd, Tony Bui, Syeda Mahwish, August Håkan Nilsson, Adithya V Ganesan, Lyle Ungar, Niranjan Balasubramanian and Saif M. Mohammad
- **Task 3: SemEval-2026 Task Proposal: Dimensional Aspect-Based Sentiment Analysis (DimABSA)** ([contact organizers], [join task mailing list])
  Liang-Chih Yu, Shamsuddeen Hassan Muhammad, Lung-Hao Lee, Jin Wang, Jan Philip Wahle, Terry Ruas, Kai-Wei Chang and Saif M. Mohammad

Narrative

- **Task 4: Narrative Story Similarity and Narrative Representation Learning** ([contact organizers], [join task mailing list])
  Hans Ole Hatzel, Ekaterina Artemova, Haimo Stiemer, Natalia Fedorova, Evelyn Gius and Chris Biemann
- **Task 5: Rating Plausibility of Word Senses in Ambiguous Sentences through Narrative Understanding** ([contact organizers], [join task mailing list])
  Janosch Gehring and Michael Roth

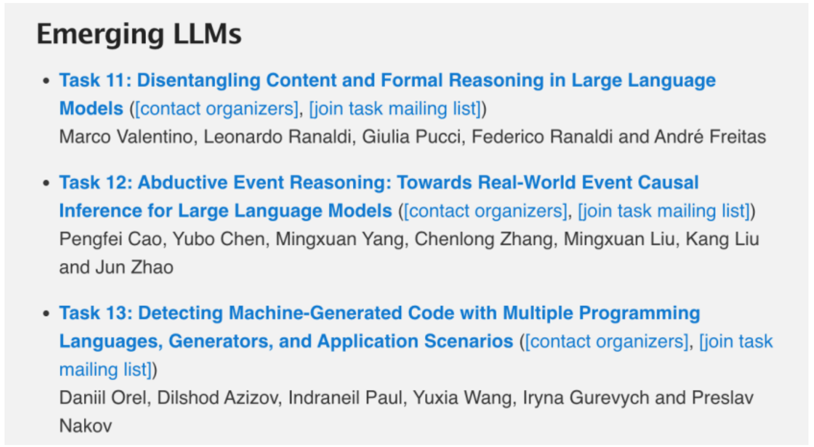


QA and Conversations

- **Task 6: CLARITY - Unmasking Political Question Evasions** ([contact organizers], [join task mailing list])
  Konstantinos Thomas, George Filandrianos, Maria Lymperaiou, Chrysoula Zerva and Giorgos Stamou
- **Task 7: Everyday Knowledge Across Diverse Languages and Cultures** ([contact organizers], [join task mailing list])
  Nedjma Ousidhoum, Junho Myung, Jiho Jin, Yi Zhou, Jose Camacho-Collados, Alice Oh, Vladimir Araujo, Chenyang Lyu, Joanne Boisson, Aleksandra Edwards, Meriem Beloucif, Christine de Kock, Joseph Marvin Imperial, Amr Keleg, Huda Hakami, Asahi Ushio, Shu-Kai HSIEH, Younes Samih, Pintu Lohar, Mohamed Fazli Mohamed Imam, Nureman Sateemae, Roy Ka-Wei Lee, Bryan Chen Zhengyu Tan, Weihua Zheng and James Barry
- **Task 8: MTRAGEval: Evaluating Multi-Turn RAG Conversations** ([contact organizers], [join task mailing list])
  Sara Rosenthal, Yannis Katsis, Vraj Shah and Marina Danilevsky

Discourse and Argumentation

- **Task 9: Detecting Multilingual, Multicultural and Multievent Online Polarization** ([contact organizers], [join task mailing list])
  Usman Naseem, Seid Muhie Yimam, Robert Geislinger, Shamsuddeen Hassan Muhammad, Sarah Kohail, Juan Ren, Rudy Alexandro Garrido Veliz, Saba Anwar, Ibrahim Said Ahmad, Idris Abdulmumin, Abinew Ali Ayele, Adem Chanie Ali, Martin Semmann and Chris Biemann
- **Task 10: PsyCoMark -- Psycholinguistic Conspiracy Marker Extraction and Detection** ([contact organizers], [join task mailing list])
  Mattia Samory, Felix Soldner and Veronika Batzdorfer

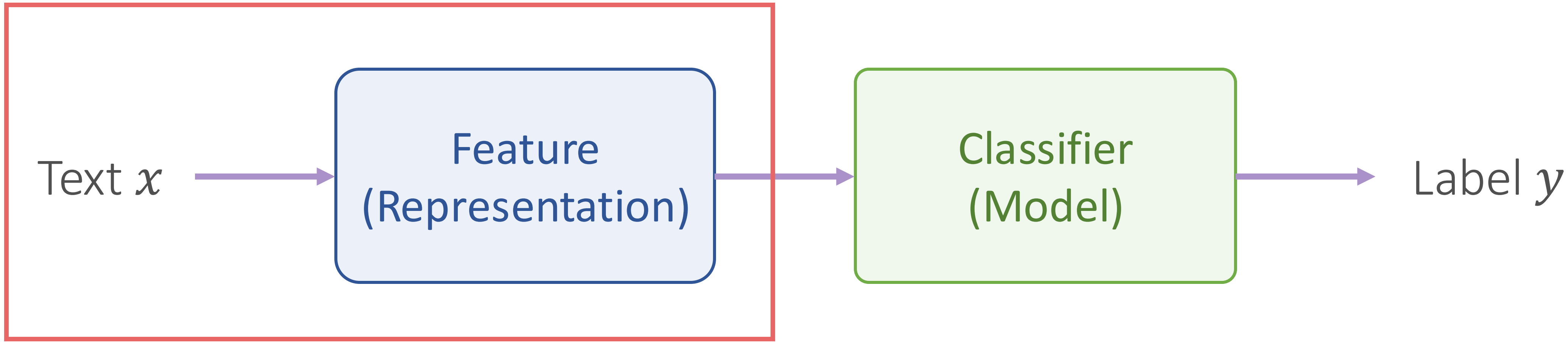


Emerging LLMs

- **Task 11: Disentangling Content and Formal Reasoning in Large Language Models** ([contact organizers], [join task mailing list])
  Marco Valentino, Leonardo Ranaldi, Giulia Pucci, Federico Ranaldi and André Freitas
- **Task 12: Abductive Event Reasoning: Towards Real-World Event Causal Inference for Large Language Models** ([contact organizers], [join task mailing list])
  Pengfei Cao, Yubo Chen, Mingxuan Yang, Chenlong Zhang, Mingxuan Liu, Kang Liu and Jun Zhao
- **Task 13: Detecting Machine-Generated Code with Multiple Programming Languages, Generators, and Application Scenarios** ([contact organizers], [join task mailing list])
  Daniil Orel, Dilshod Azizov, Indraneil Paul, Yuxia Wang, Iryna Gurevych and Preslav Nakov

Question?

Recap: A General Framework for Text Classification

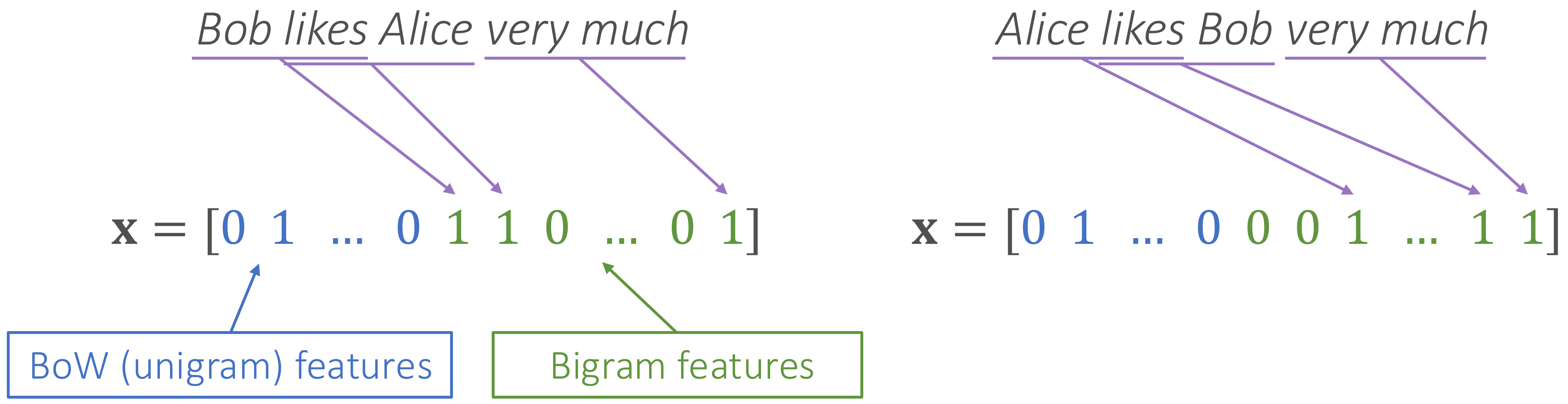


- Teach the model how to understand example x
- Convert the text to a mathematical form
  - The mathematical form captures essential characteristics of the text
- Bag-of-words, n-grams, word embeddings, etc.

Topic Study

- Topic Study (30%)
  - Literature Review (15%) [Due: 10/2]
  - Topic Presentation (15%)
    - Email your slides to the instructor at least 2 days before your presentation

Recap: Bag-of-N-Grams



$$\mathbf{x} = [0 \;\; 1 \;\; \ldots \;\; 0 \;\; 1 \;\; 1 \;\; 0 \;\; \ldots \;\; 0 \;\; 1] \qquad \mathbf{x} = [0 \;\; 1 \;\; \ldots \;\; 0 \;\; 0 \;\; 0 \;\; 1 \;\; \ldots \;\; 1 \;\; 1]$$

We can consider trigrams, 4-grams, …

N-gram features capture more sentential structure

Recap: Words as Vectors

$$\begin{array}{cc} & \begin{array}{ccccc} \textit{Bob} & \textit{likes} & \textit{Alice} & \textit{very} & \textit{much} \end{array} \\ W = & \begin{bmatrix} | & | & | & | & | \\ w_{bob} & w_{likes} & w_{Alice} & w_{very} & w_{much} \\ | & | & | & | & | \end{bmatrix} \end{array}$$

Use ***one vector*** to represent ***each word***

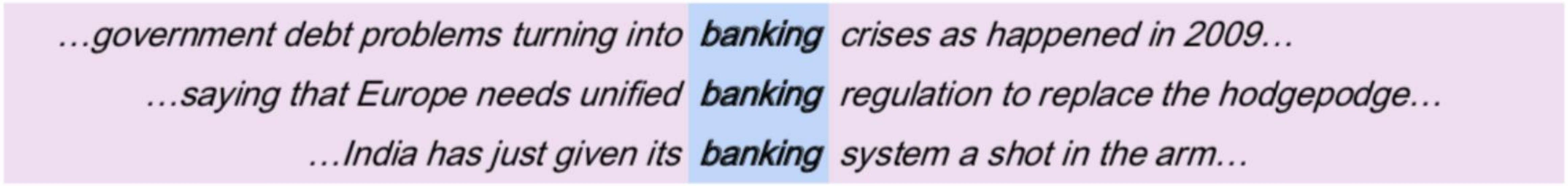
Text = A list of vectors

Recap: Representing Words by Their Contexts

Distributional hypothesis: words that occur in similar contexts tend to have similar meanings

J.R.Firth 1957

- "You shall know a word by the company it keeps"
- One of the most successful ideas of modern statistical NLP!



...government debt problems turning into	***banking***	*crises as happened in 2009...*
...saying that Europe needs unified	***banking***	*regulation to replace the hodgepodge...*
...India has just given its	***banking***	*system a shot in the arm...*

These context words will represent banking

Recap: Word Vectors from Co-Occurrence Matrix

- Main idea: Similar contexts → Similar word co-occurrence
- Collect a bunch of texts and compute co-occurrence matrix
- Words can be represented by row vectors

$$\cos(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\|\|\mathbf{v}\|}$$

Word Vector

High cosine similarity!

	shark	computer	data	eat	result	sugar
apple	0	0	0	8	0	2
bread	0	0	0	9	0	1
digital	0	6	5	0	2	0
information	0	4	10	0	2	0

Low cosine similarity!

Recap: Co-Occurrence Matrix with Positive PMI

Positive Pointwise Mutual Information (PPMI)

$$\text{PPMI}(x, y) = \max\left(\log_2 \frac{P(x,y)}{P(x)P(y)}, 0\right)$$

	shark	computer	data	eat	result	sugar	the	it
apple	0	0	0	1.80	0	0.35	0.08	0
bread	0	0	0	1.54	0	0.29	0	0.14
digital	0	1.47	1.22	0	0.61	0	0.10	0.06

Similarity(apple, bread) ≈ 0.995069

Similarity(apple, digital) ≈ 0.010795

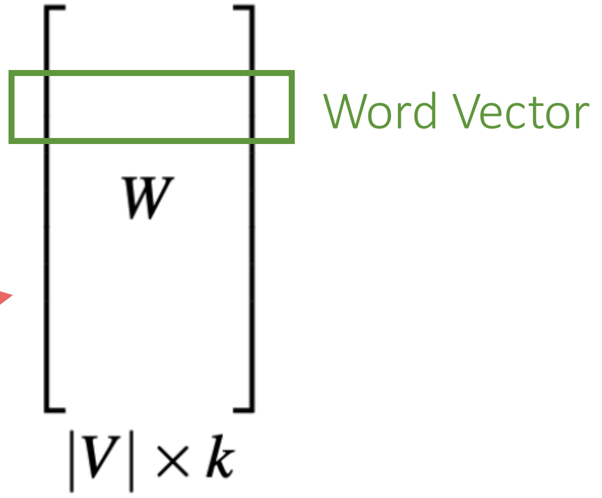
Recap: From Sparse Vectors to Dense Vectors

- Singular value decomposition (SVD) of PPMI weighted co-occurrence matrix

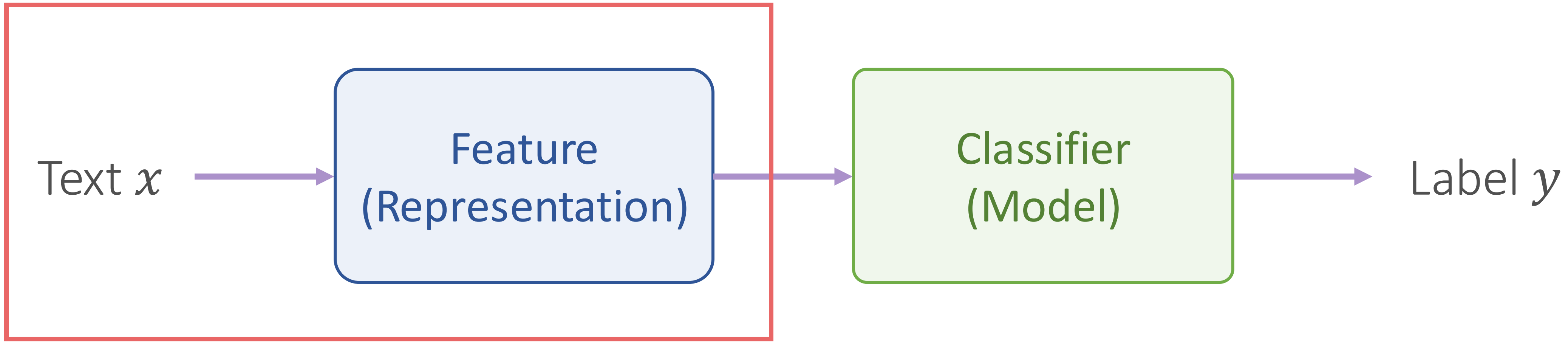
$$\underset{|V|\times|V|}{\begin{bmatrix} & & \\ & X & \\ & & \end{bmatrix}} = \underset{|V|\times|V|}{\begin{bmatrix} & & \\ & W & \\ & & \end{bmatrix}} \underset{|V|\times|V|}{\begin{bmatrix} \sigma_1 & 0 & 0 & \dots & 0 \\ 0 & \sigma_2 & 0 & \dots & 0 \\ 0 & 0 & \sigma_3 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \sigma_V \end{bmatrix}} \underset{|V|\times|V|}{\begin{bmatrix} & & \\ & C & \\ & & \end{bmatrix}}$$

Only keep the top k singular values

$$\underset{|V|\times|V|}{\begin{bmatrix} & & \\ & X & \\ & & \end{bmatrix}} = \underset{|V|\times k}{\begin{bmatrix} \\ W \\ \\ \end{bmatrix}} \underset{k\times k}{\begin{bmatrix} \sigma_1 & 0 & 0 & \dots & 0 \\ 0 & \sigma_2 & 0 & \dots & 0 \\ 0 & 0 & \sigma_3 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \sigma_k \end{bmatrix}} \underset{k\times|V|}{\begin{bmatrix} & C & \end{bmatrix}}$$

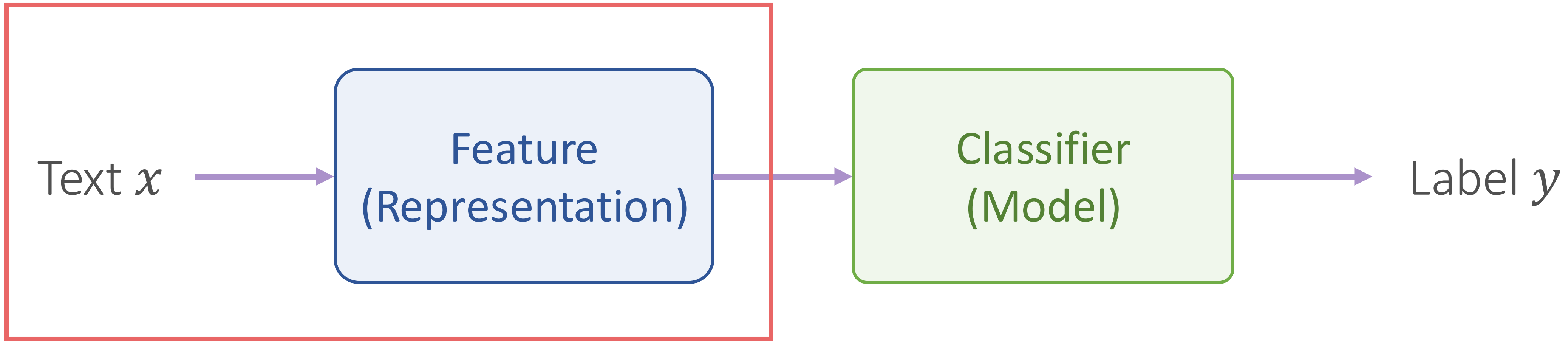


Count-Based Word Vectors



- Use one vector to represent each word
- Get word vectors by singular value decomposition (SVD) of PPMI weighted co-occurrence matrix

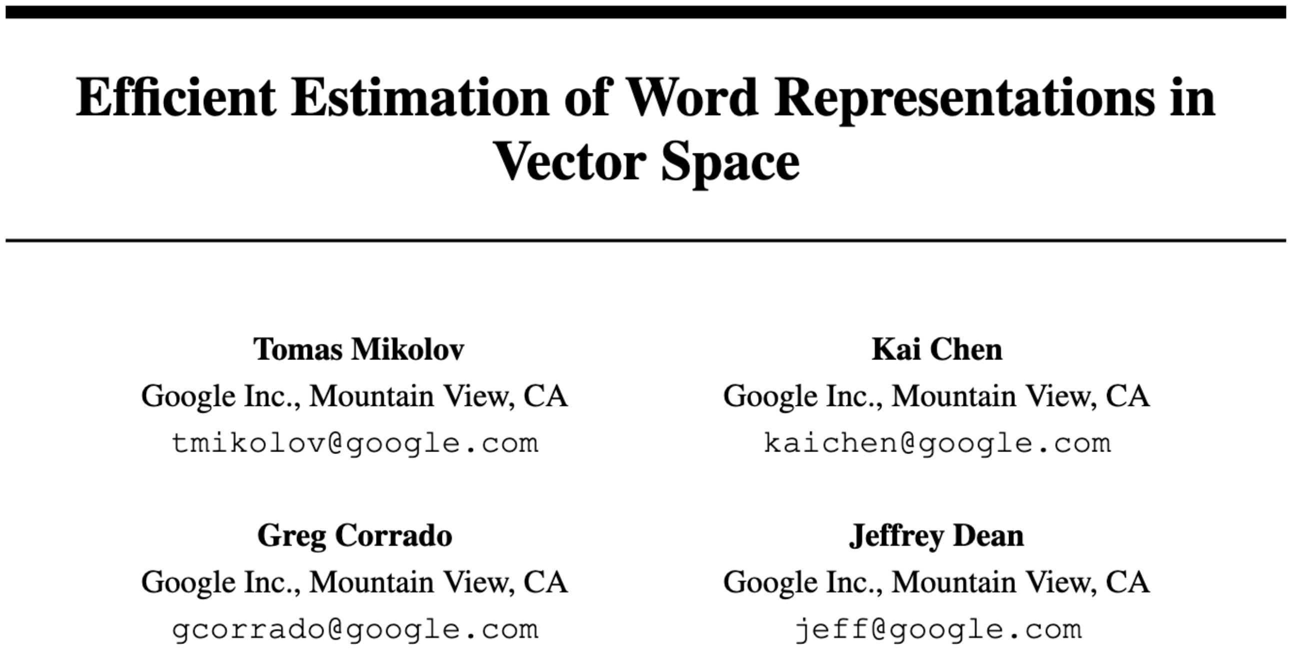
Prediction-Based Word Vectors



- Can we learn word vectors directly from text?

Word2Vec

- Efficient Estimation of Word Representations in Vector Space, 2013
  - 40000+ citations



Efficient Estimation of Word Representations in Vector Space

Tomas Mikolov
Google Inc., Mountain View, CA
tmikolov@google.com

Kai Chen
Google Inc., Mountain View, CA
kaichen@google.com

Greg Corrado
Google Inc., Mountain View, CA
gcorrado@google.com

Jeffrey Dean
Google Inc., Mountain View, CA
jeff@google.com

Word Embeddings as Learning Problem

- Learning vectors (also called embeddings) from text for representing words
- Input:
  - A large text corpus
    - Wikipedia + Gigaword 5: 6B tokens
    - Twitter: 27B tokens
    - Common Crawl: 840B tokens
  - Vocabulary $\mathcal{V}$
  - Vector dimension d (e.g., 300)
- Output:
  - Mapping function $f: \mathcal{V} \rightarrow \mathbb{R}^d$

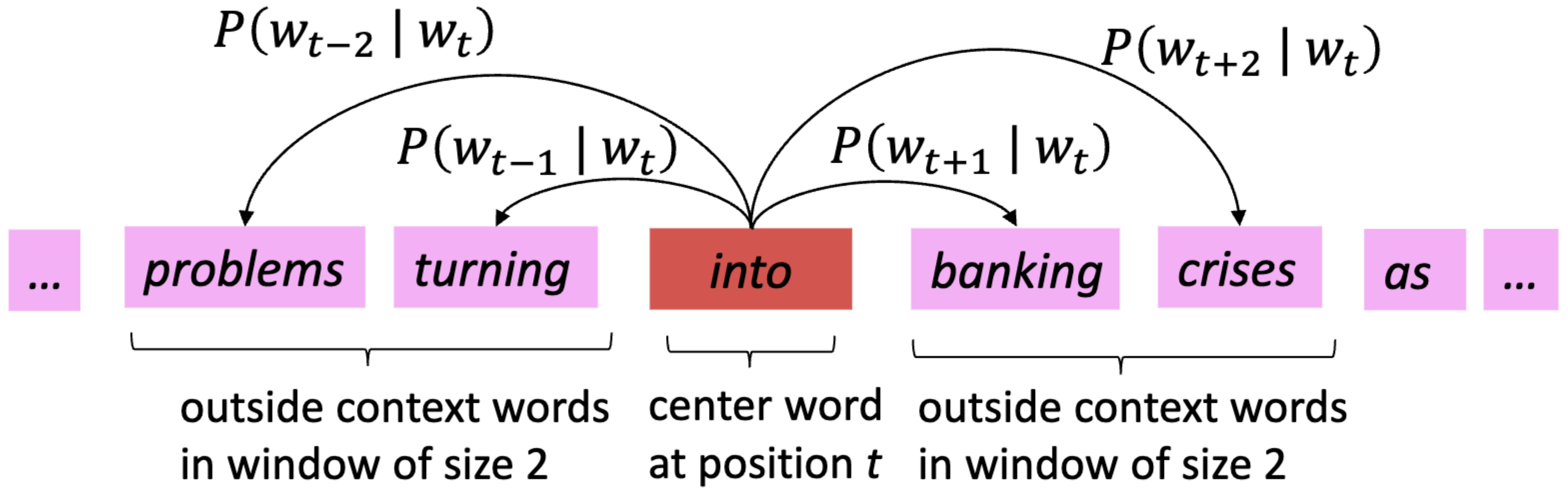
$$v_{apple} = \begin{pmatrix} -0.224 \\ 0.479 \\ 0.871 \\ -0.231 \\ 0.101 \end{pmatrix}$$

$$v_{digital} = \begin{pmatrix} 0.257 \\ 0.587 \\ -0.972 \\ -0.456 \\ -0.002 \end{pmatrix}$$

Word2Vec: Overview

- **Main idea:** we want to use words to predict their context words
- Context: a fixed window of size m

Use center word w_t to predict context words w_{t-m} to w_{t+m}



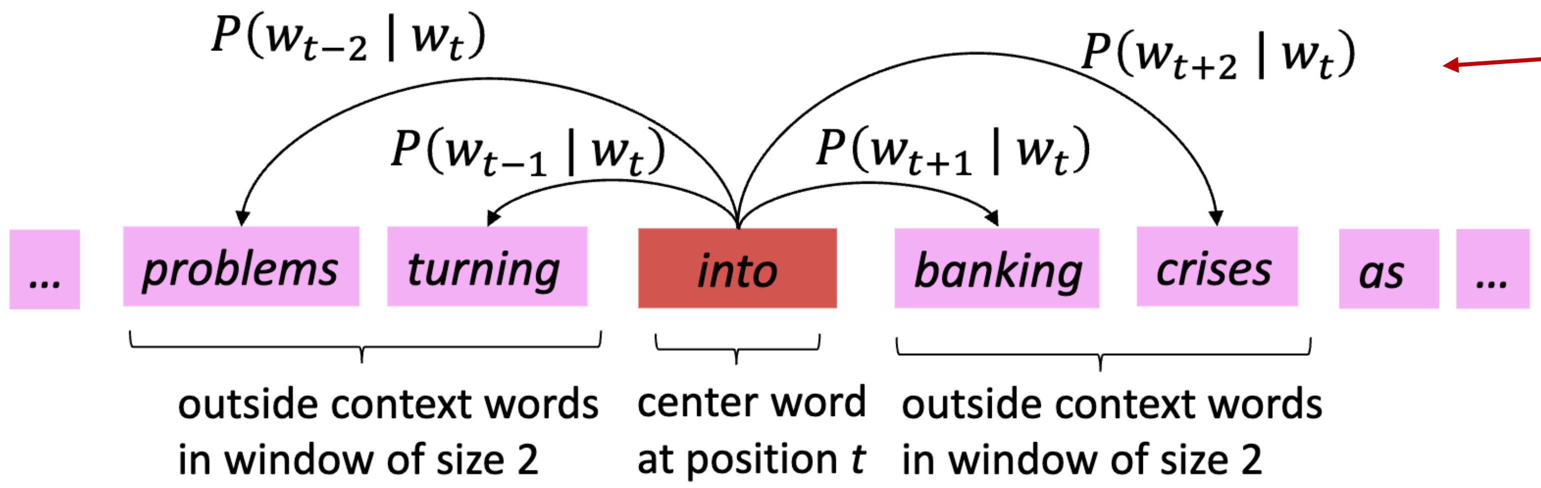
Words that occur in similar contexts tend to have similar meanings

Word2Vec: Overview

- **Main idea:** we want to use words to predict their context words
- Context: a fixed window of size m

Classification Problem

Use center word w_t to predict context words w_{t-m} to w_{t+m}



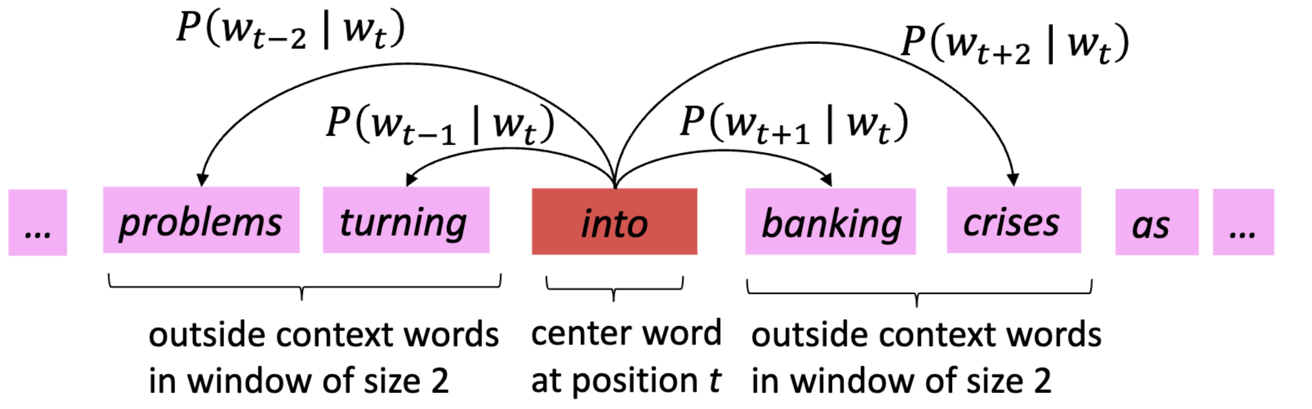
$P(b|a)$ = given the center word is a, what is the probability that b is a context word?

$P(\cdot\,|a)$ is a probability distribution defined over $\mathcal{V}$:

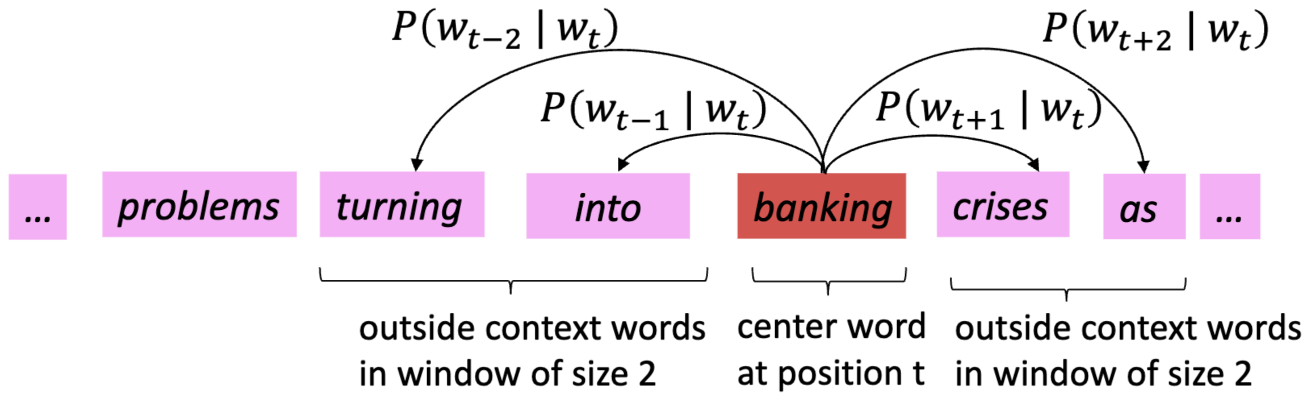
$$\sum_{w \in \mathcal{V}} P(w|a) = 1$$

We will define the distribution soon!

Word2Vec: Overview



Collect into training data
(into, problems)
(into, turning)
(into, banking)
(into, crises)



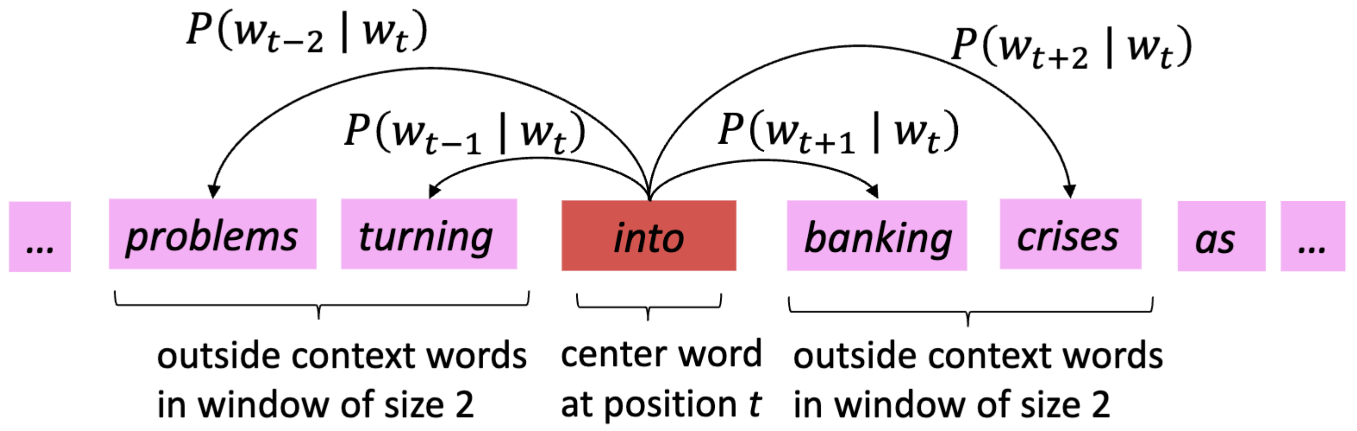
Collect into training data
(banking, turning)
(banking, into)
(banking, crises)
(banking, as)

Maximize the likelihood

$$P(\text{problems} \mid \text{into}) \times P(\text{turning} \mid \text{into}) \times P(\text{banking} \mid \text{into}) \times P(\text{crises} \mid \text{into})$$

$$\times\, P(\text{turning} \mid \text{banking}) \times P(\text{into} \mid \text{banking}) \times P(\text{crises} \mid \text{banking}) \times P(\text{as} \mid \text{banking})$$

Word2Vec: Likelihood



For each position $t = 1, \dots, T$, predict context words within a window of fixed size m, given center word w_t

θ all parameters to be optimized

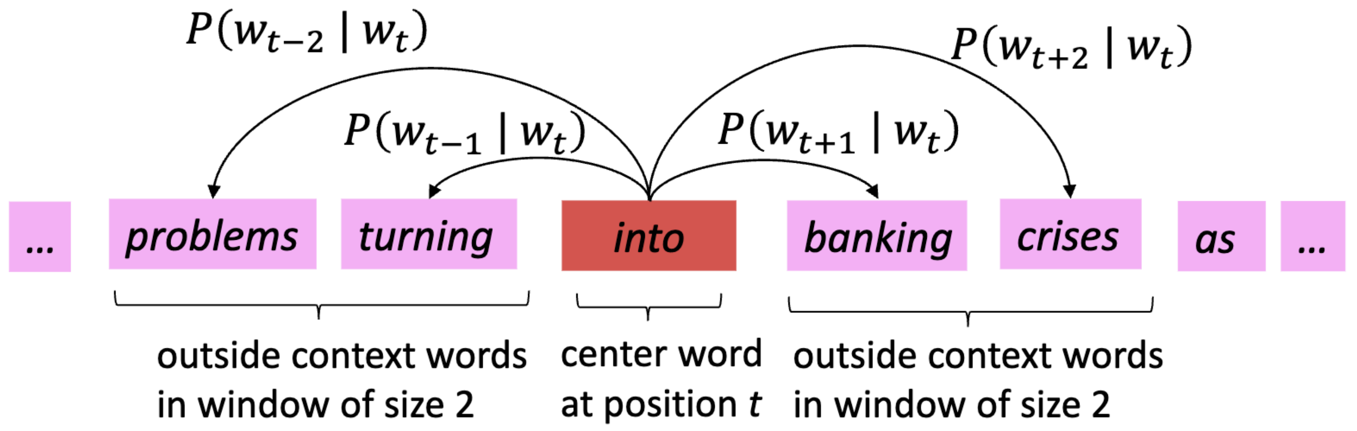
$$\text{Likelihood} = \mathcal{L}(\theta) = \prod_{t=1}^{T} \prod_{-m \le j \le m, j \ne 0} P(w_{t+j} \mid w_t ; \theta)$$

Probability over all vocabulary V

For each position $t = 1, \dots, T$

Likelihood for all context words given center word w_t

Word2Vec: Objective Function



The objective function $J(\theta)$ is the (average) negative log likelihood

$$J(\theta) = -\frac{1}{T}\log \mathcal{L}(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{-m\le j\le m, j\ne 0} \log P\left(w_{t+j} \middle| w_t ; \theta\right)$$

We minimize the objective function (also called cost or loss function)

How to Define Probability?

Question: how to calculate $P\left(w_{t+j} \middle| w_t ; \theta\right)$?

Answer: we have two sets of vectors for each word in the vocabulary

$\mathbf{u}_w \in \mathbb{R}^d$: word vector when w is a center word

$\mathbf{v}_w \in \mathbb{R}^d$: word vector when w is a context word

We consider Inner product $\mathbf{u}_{w_t} \cdot \mathbf{v}_{w_{t+j}}$ as the score to measure how likely the context word w_{t+j} appears with the center word w_t, the larger the more likely!

$$P\left(w_{t+j} \middle| w_t ; \theta\right) = \frac{\exp(\mathbf{u}_{w_t} \cdot \mathbf{v}_{w_{t+j}})}{\sum_{k \in V} \exp(\mathbf{u}_{w_t} \cdot \mathbf{v}_k)} \qquad \theta = \{\{\mathbf{u}_k\}, \{\boldsymbol{v}_k\}\} \text{ all parameters}$$

How to Define Probability?

We have two sets of vectors for each word in the vocabulary

$\mathbf{u}_w \in \mathbb{R}^d$: word vector when w is a center word

$\mathbf{v}_w \in \mathbb{R}^d$: word vector when w is a context word

$$P(w_{t+j} \mid w_t ; \theta) = \frac{\exp(\mathbf{u}_{w_t} \cdot \mathbf{v}_{w_{t+j}})}{\sum_{k \in V} \exp(\mathbf{u}_{w_t} \cdot \mathbf{v}_k)}$$

Normalize over entire vocabulary to give probability distribution

The score to indicate how likely the context word w_{t+j} appears with the center word w_t

Softmax function: mapping arbitrary values to a probability distribution

$$\text{softmax}(t) = \frac{e^t}{\sum_c e^c}$$

Why Two Sets of Vectors?

We have two sets of vectors for each word in the vocabulary

$\mathbf{u}_w \in \mathbb{R}^d$: word vector when w is a center word

$\mathbf{v}_w \in \mathbb{R}^d$: word vector when w is a context word

$$P\left(w_{t+j} \middle| \, w_t \, ; \theta\right) = \frac{\exp(\mathbf{u}_{w_t} \cdot \mathbf{v}_{w_{t+j}})}{\sum_{k \in V} \exp(\mathbf{u}_{w_t} \cdot \mathbf{v}_k)}$$

- Scores can be asymmetric
- It is not likely that a word appears in its own context

How to Train Word Vectors?

Parameters: $\theta = \{\{\mathbf{u}_k\}, \{\boldsymbol{v}_k\}\}$

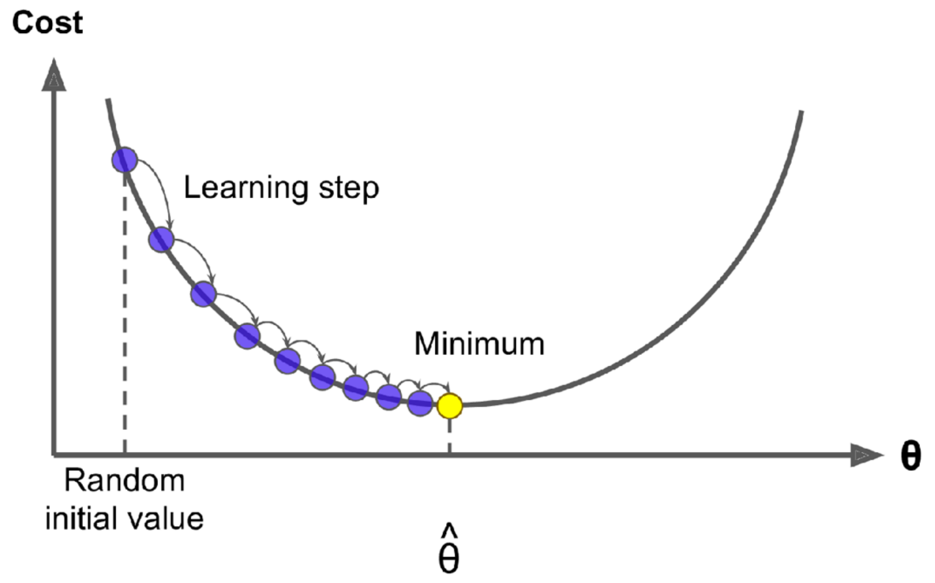
Objective function: $J(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{-m\leq j\leq m, j\neq 0} \log P\left(w_{t+j} \middle| w_t\,; \theta\right)$

Our goal: find parameters θ that minimize the objective function $J(\theta)$

Solution: stochastic gradient descent (SGD)

- Randomly initialize parameters θ
- For each iteration $\theta \longleftarrow \theta - \eta \, \nabla_\theta \, J(\theta)$

Learning step (η) Gradient (∇_θ)



https://insightfultscript.com/collections/programming/machine-learning/sgd/

Computing the Gradients

Objective function

$$J(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{-m\le j\le m, j\ne 0} \log P\left(w_{t+j}\middle|\, w_t\,;\theta\right)$$

$$= \frac{1}{T}\sum_{t=1}^{T}\sum_{-m\le j\le m, j\ne 0} \boxed{-\log P\left(w_{t+j}\middle|\, w_t\,;\theta\right)}$$

The gradients can be calculated separately!

For simplicity, we consider one pair of center/context words (o, c)

$$y = -\log P(c|o\,;\theta) = -\log\left(\frac{\exp(\mathbf{u}_o\cdot\mathbf{v}_c)}{\sum_{k\in V}\exp(\mathbf{u}_o\cdot\mathbf{v}_k)}\right)$$

$$\boxed{\frac{\partial y}{\partial \mathbf{u}_o} \quad \frac{\partial y}{\partial \boldsymbol{v}_c}}$$

We need to compute this!

Computing the Gradients

$$y = -\log P(c|o) = -\log\left(\frac{\exp(\mathbf{u}_o \cdot \mathbf{v}_c)}{\sum_{k\in V}\exp(\mathbf{u}_o \cdot \mathbf{v}_k)}\right) = \boxed{-\log(\exp(\mathbf{u}_o \cdot \mathbf{v}_c))} + \log\left(\sum_{k\in V}\exp(\mathbf{u}_o \cdot \mathbf{v}_k)\right)$$

$$= -\mathbf{u}_o \cdot \mathbf{v}_c$$

$$\frac{\partial \log(x)}{\partial x} = \frac{1}{x} \qquad \frac{\partial \exp(x)}{\partial x} = \exp(x)$$

$$\frac{\partial y}{\partial \mathbf{u}_o} = \frac{\partial(-\mathbf{u}_o \cdot \mathbf{v}_c + \log(\sum_{k\in V}\exp(\mathbf{u}_o \cdot \mathbf{v}_k)))}{\partial \mathbf{u}_o} = -\mathbf{v}_c + \frac{\sum_{k\in V}\frac{\partial \exp(\mathbf{u}_o \cdot \mathbf{v}_k)}{\partial \mathbf{u}_o}}{\sum_{k\in V}\exp(\mathbf{u}_o \cdot \mathbf{v}_k)}$$

$$= -\mathbf{v}_c + \frac{\sum_{k\in V}\exp(\mathbf{u}_o \cdot \mathbf{v}_k)\,\mathbf{v}_k}{\sum_{k\in V}\exp(\mathbf{u}_o \cdot \mathbf{v}_k)} = -\mathbf{v}_c + \sum_{k\in V}\frac{\exp(\mathbf{u}_o \cdot \mathbf{v}_k)\,\mathbf{v}_k}{\sum_{k\in V}\exp(\mathbf{u}_o \cdot \mathbf{v}_k)}$$

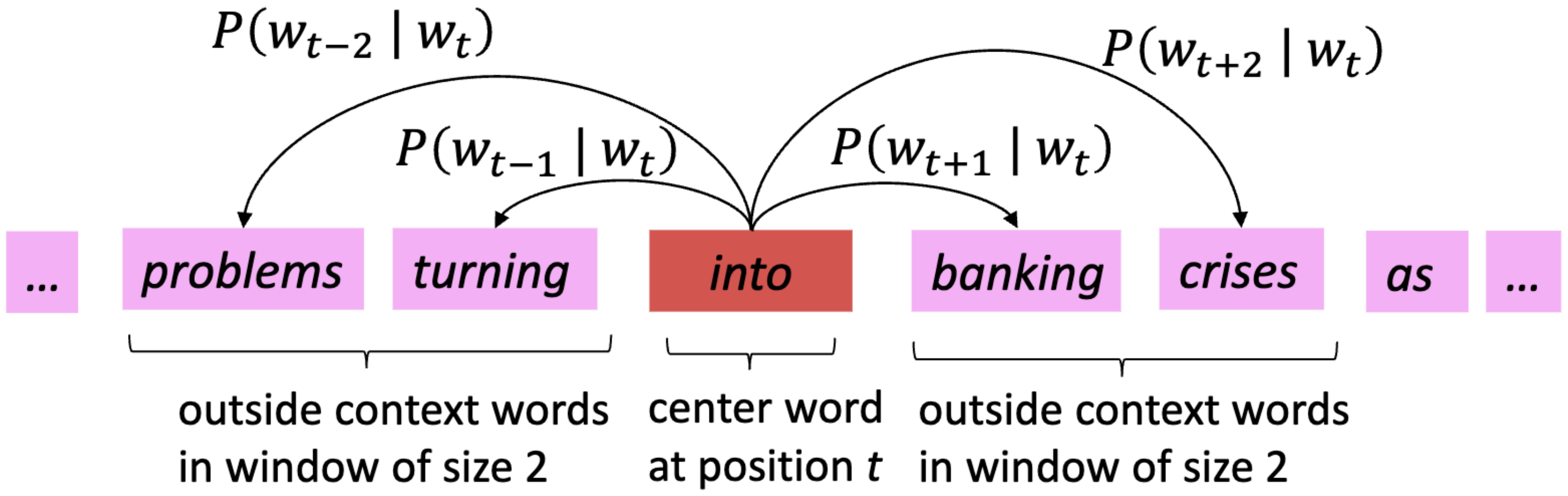
$$= -\mathbf{v}_c + \sum_{k\in V} P(k|o)\,\mathbf{v}_k$$

$$\boxed{\frac{\partial y}{\partial \mathbf{v}_k} = -1(k = c)\mathbf{u}_o + P(k|o)\mathbf{u}_o}$$

Similar calculation step

Training Process

- Randomly initialize parameters $\mathbf{u}_i, \mathbf{v}_i$
- Walk through the training corpus and collect training data (o, c)



$$\mathbf{u}_o \leftarrow \mathbf{u}_o - \eta \frac{\partial y}{\partial \mathbf{u}_o} \qquad \mathbf{v}_k \leftarrow \mathbf{v}_k - \eta \frac{\partial y}{\partial \mathbf{v}_k} \quad \forall k \in V$$

Negative Sampling

Issue: every time we get one pair of (o, c), we have to update $\mathbf{v}_k$ with all the words in the vocabulary.

$$\mathbf{u}_o \leftarrow \mathbf{u}_o - \eta \frac{\partial y}{\partial \mathbf{u}_o} \qquad \mathbf{v}_k \leftarrow \mathbf{v}_k - \eta \frac{\partial y}{\partial \mathbf{v}_k} \quad \forall k \in V$$

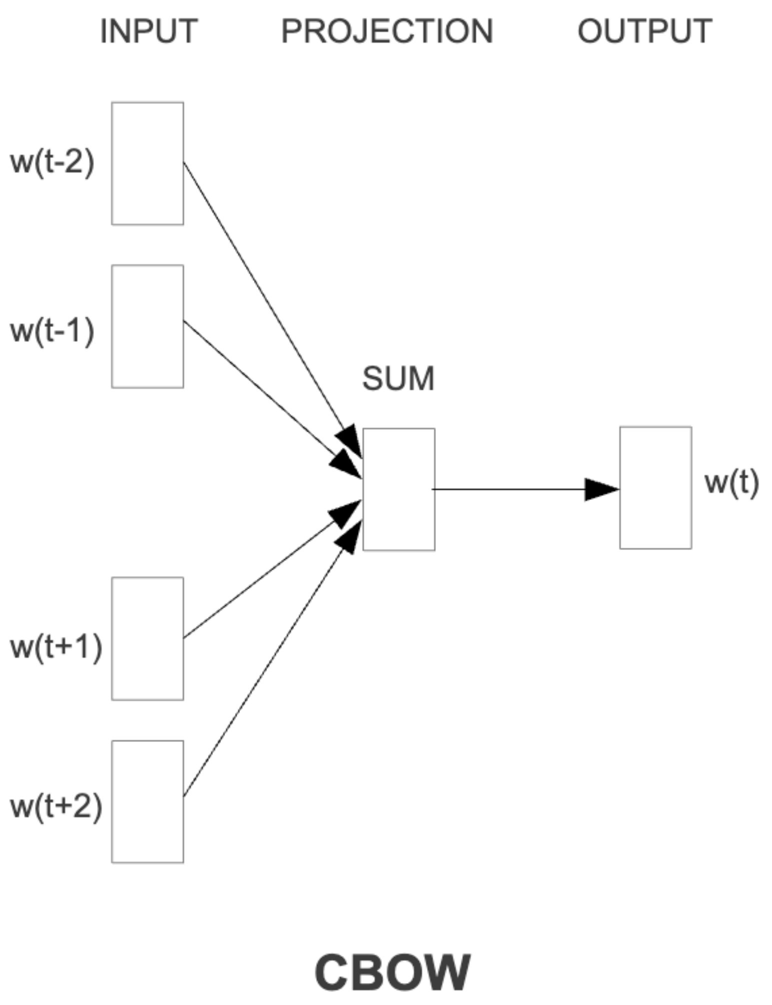
Negative sampling: instead of considering all the words in V, we randomly sample K(5-20) negative examples

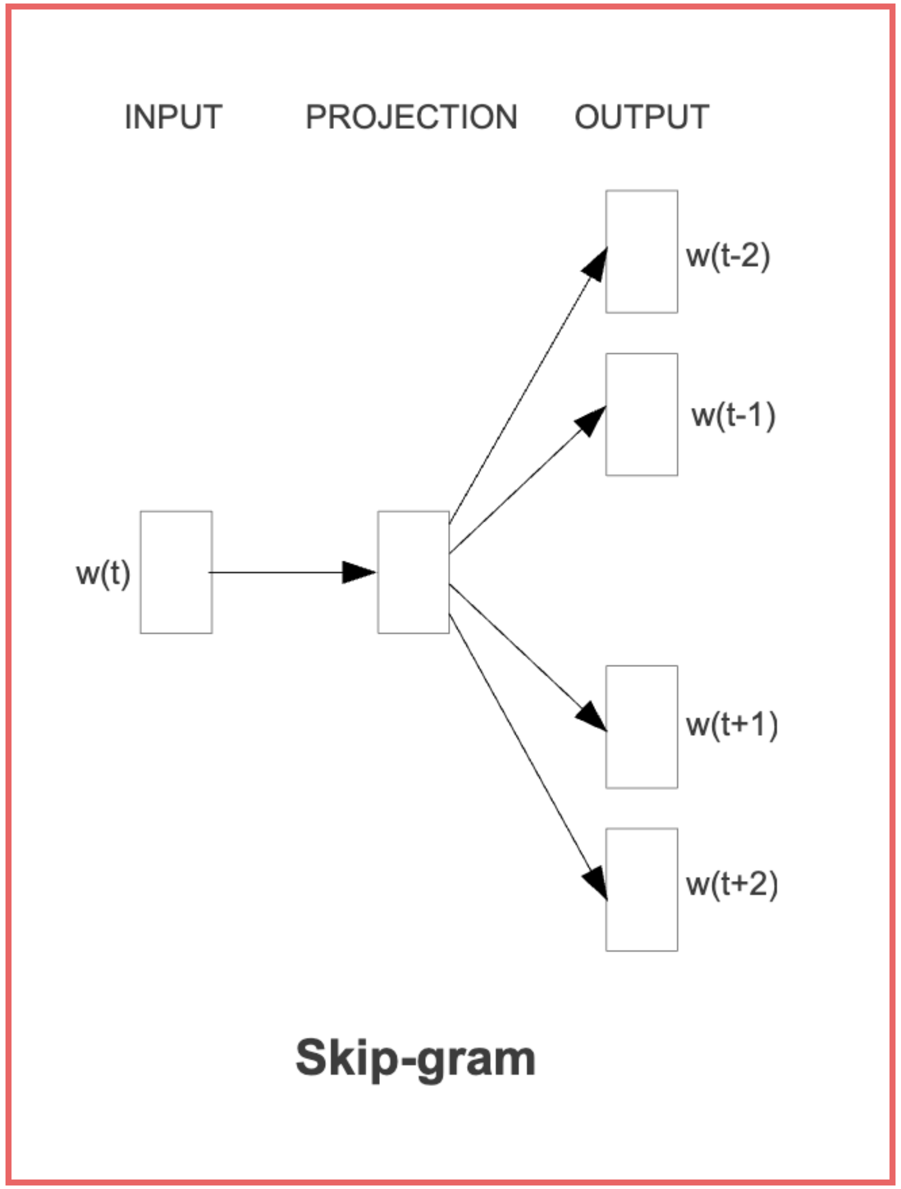
Softmax $$y = -\log\left(\frac{\exp(\mathbf{u}_o \cdot \mathbf{v}_c)}{\sum_{k\in V} \exp(\mathbf{u}_o \cdot \mathbf{v}_k)}\right) = -\log(\exp(\mathbf{u}_o \cdot \mathbf{v}_c)) + \log\left(\sum_{k\in V} \exp(\mathbf{u}_o \cdot \mathbf{v}_k)\right)$$

Negative sampling $$y = -\log(\sigma(\mathbf{u}_o \cdot \mathbf{v}_c)) - \sum_{i=1}^{K} \mathbb{E}_{j\sim P(w)} \log(\sigma(-\mathbf{u}_o \cdot \mathbf{v}_j))$$

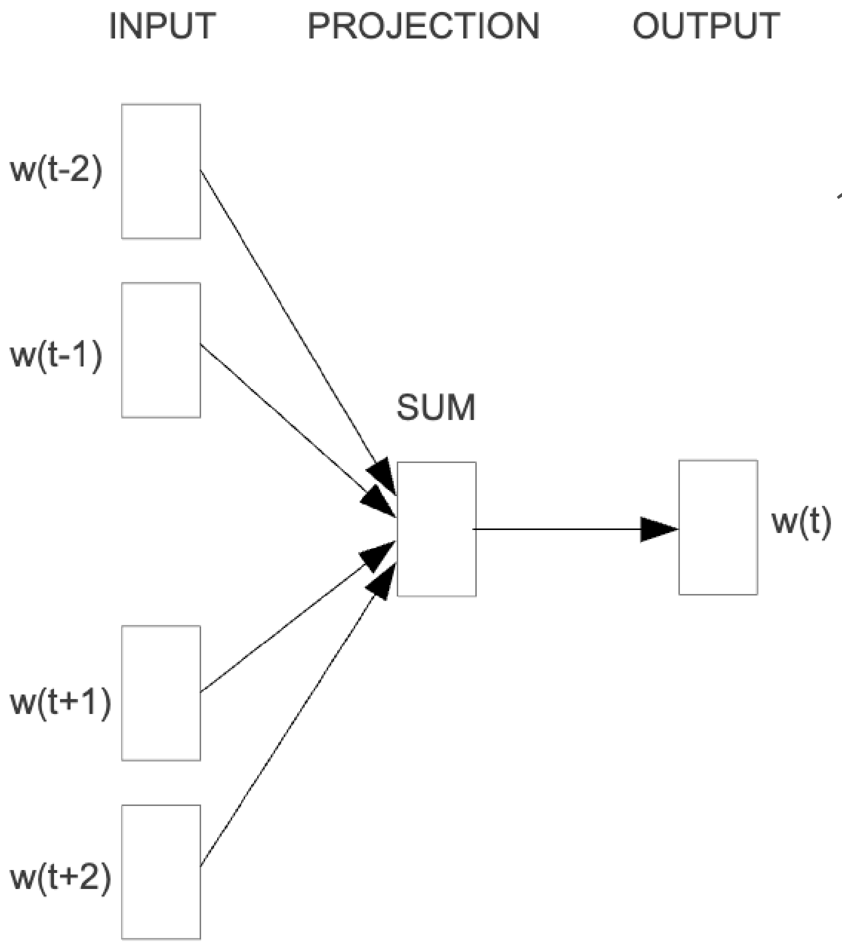
$$\boxed{\sigma(x) = \frac{1}{1+e^{-x}}}$$

Continuous Bag of Words (CBOW) vs Skip-Grams





Continuous Bag of Words (CBOW)



$$\mathcal{L}(\theta) = \prod_{t=1}^{T} P(w_t | \{w_{t+j}\}), -m \le j \le m, j \ne 0$$

$$P(w_t | \{w_{t+j}\}) = \frac{\exp(\mathbf{u}_{w_t} \cdot \bar{\mathbf{v}}_t)}{\sum_{k \in V} \exp(\mathbf{u}_k \cdot \bar{\mathbf{v}}_t)}$$

$$\bar{\mathbf{v}}_t = \frac{1}{2m} \sum_{-m \le j \le m, j \ne 0} \mathbf{v}_{t+j}$$

GloVe: Global Vectors

GloVe: Global Vectors for Word Representation (Pennington et al. 2014)

Idea: capture ratios of co-occurrence probabilities as linear meaning components in a word vector space

Log-bilinear model $w_i \cdot w_j = \log P(i|j)$

Vector difference $w_i \cdot (w_a - w_b) = \dfrac{\log P(x|a)}{\log P(x|b)}$

$$J = \sum_{i,j=1}^{V} f(X_{ij})\left(w_i^{\top}\tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij}\right)^2$$

Global co-occurrence statistics

Training faster and scalable to very large corpora!

FastText: Sub-Word Embeddings

Enriching Word Vectors with Subword Information (Bojanowski et al. 2017)

Similar as Skip-gram, but break words into n-grams with n = 3 to 6

where

3-grams: <wh, whe, her, ere, re>

4-grams: <whe, wher, here, ere>

5-grams: <wher, where, here>

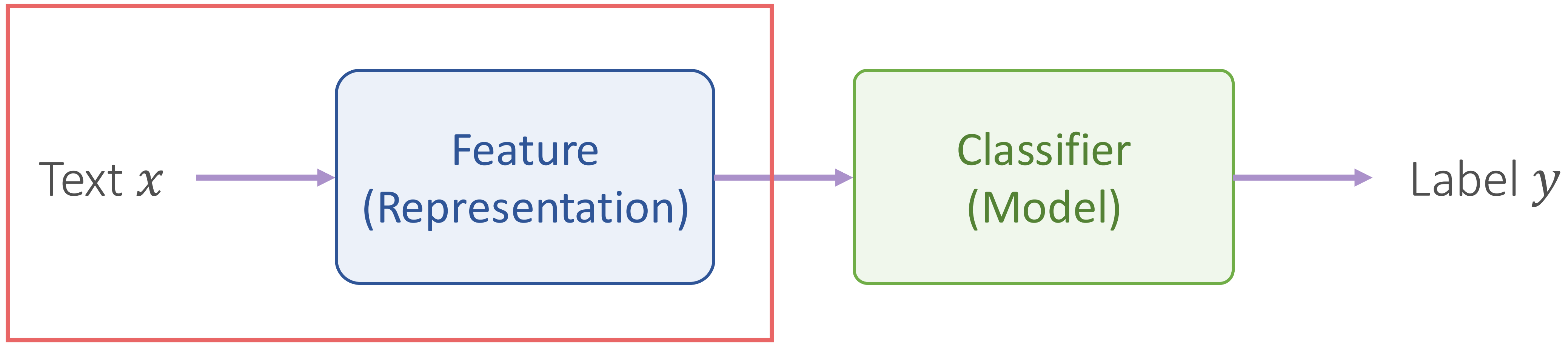
6-grams: <where, where>

Replace $\mathbf{u}_i \cdot \mathbf{v}_j$ with $$\sum_{g \in n-grams(w_i)} \mathbf{u}_g \cdot \mathbf{v}_j$$

Trained Word Vectors Are Available

- Word2Vec: https://code.google.com/archive/p/word2vec/
- GloVe: https://nlp.stanford.edu/projects/glove/
- FastText: https://fasttext.cc/

Prediction-Based Word Vectors



- Learn word vectors directly from text
  - Word2Vec (Skip-Gram and CBOW)
  - GloVe
  - FastText

Intrinsic Evaluation: Word Analogy

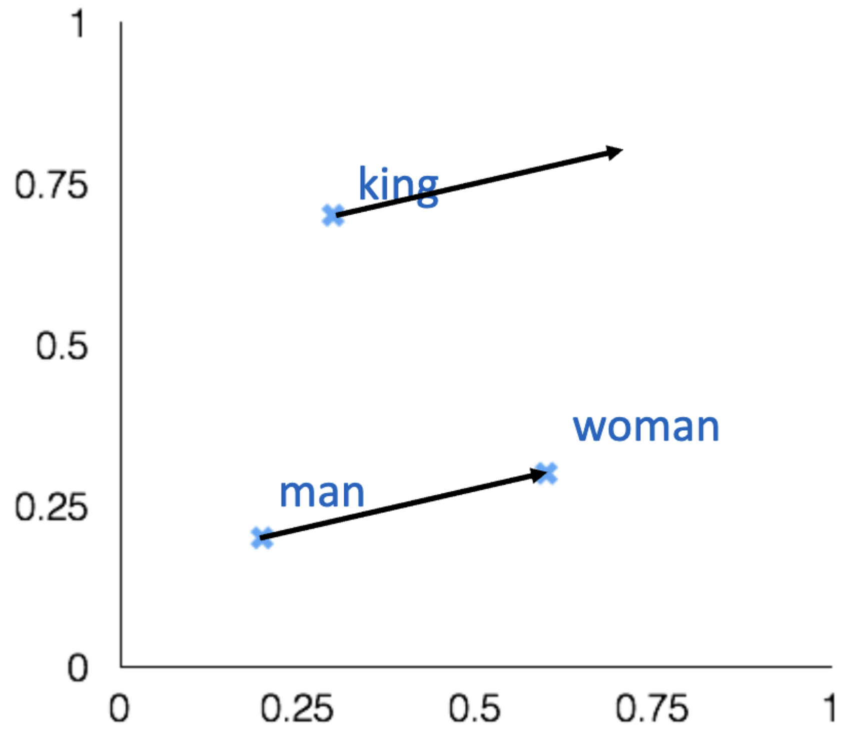
Word analogy

man: woman ≈ king: ?

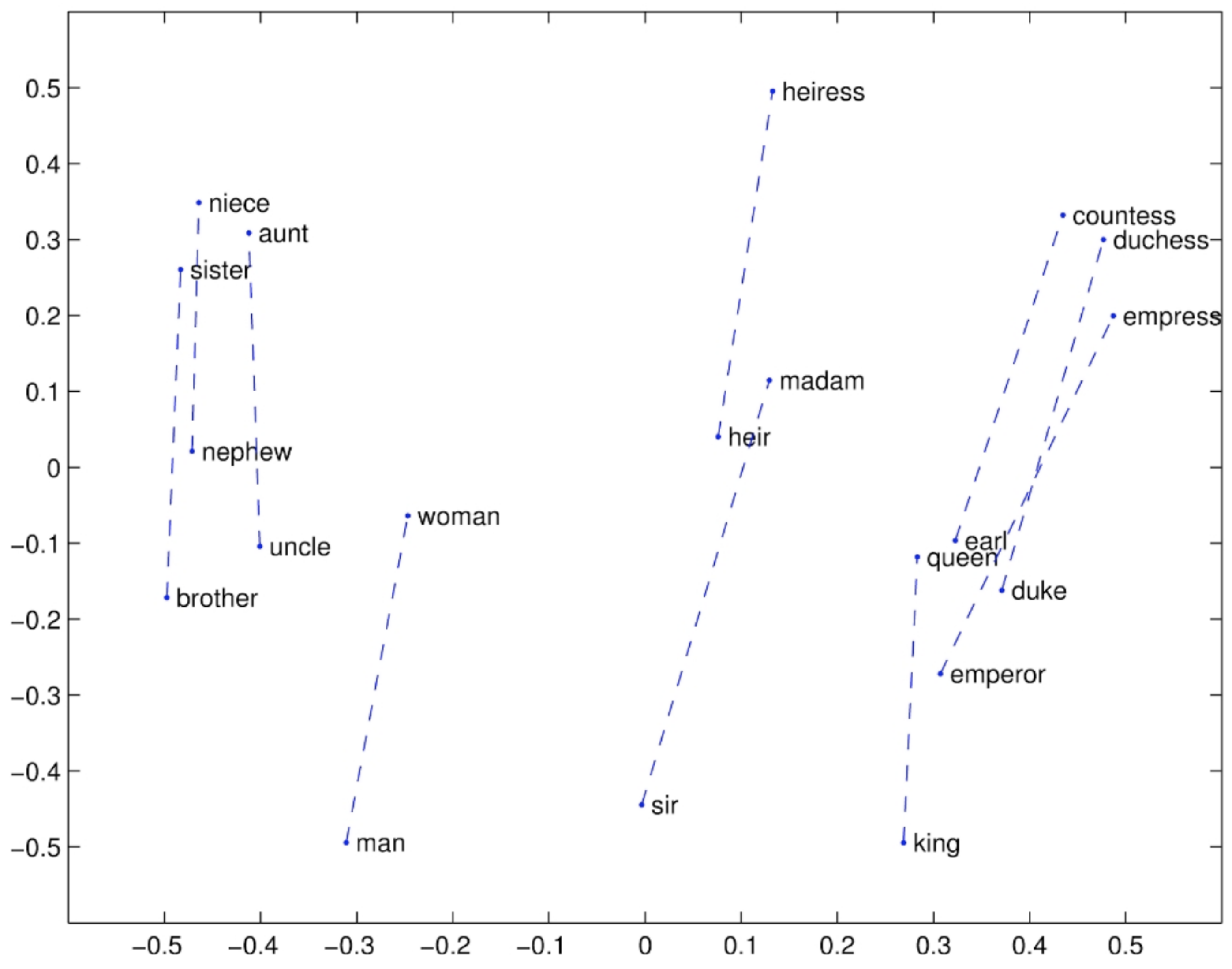
Paris: France ≈ London: ?

bad: worst ≈ cool: ?

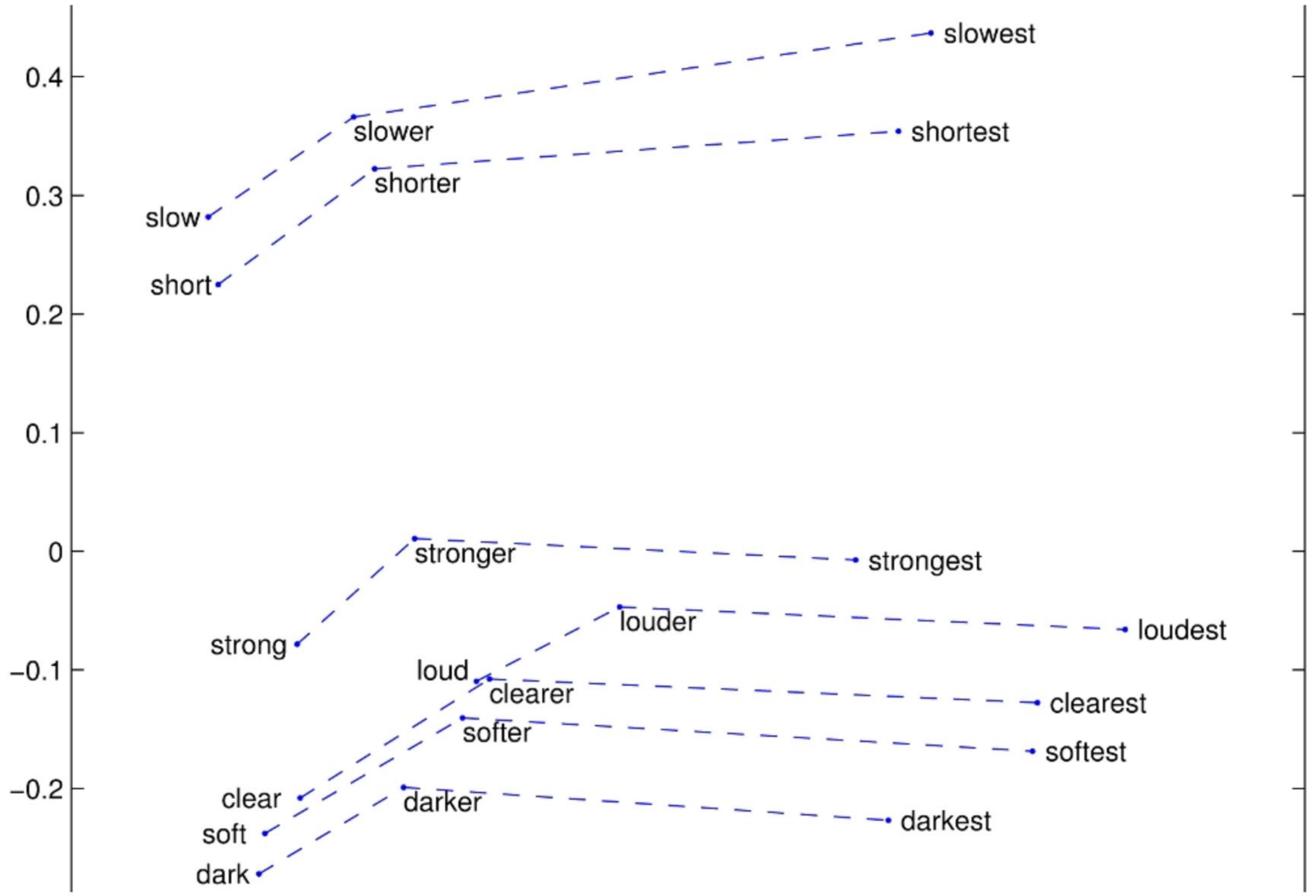
$$\arg\max_{w}\left(\cos(\mathbf{u}_w, \mathbf{u}_{woman} - \mathbf{u}_{man} + \mathbf{u}_{king})\right)$$



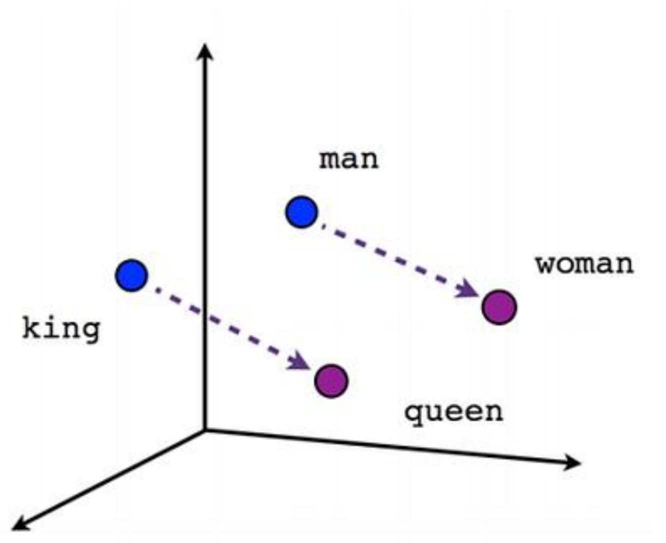
Intrinsic Evaluation: Word Analogy



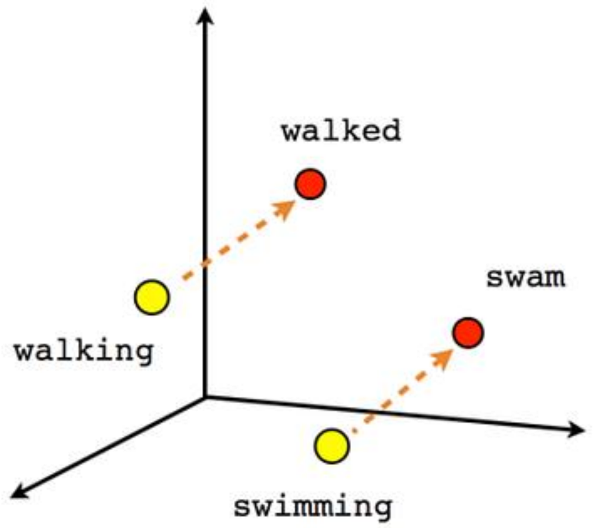
Intrinsic Evaluation: Word Analogy



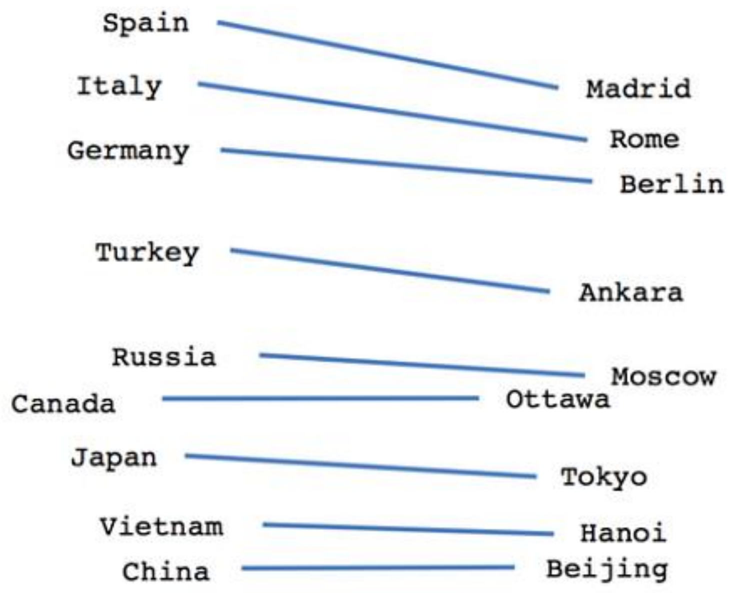
Intrinsic Evaluation: Word Analogy



Male-Female

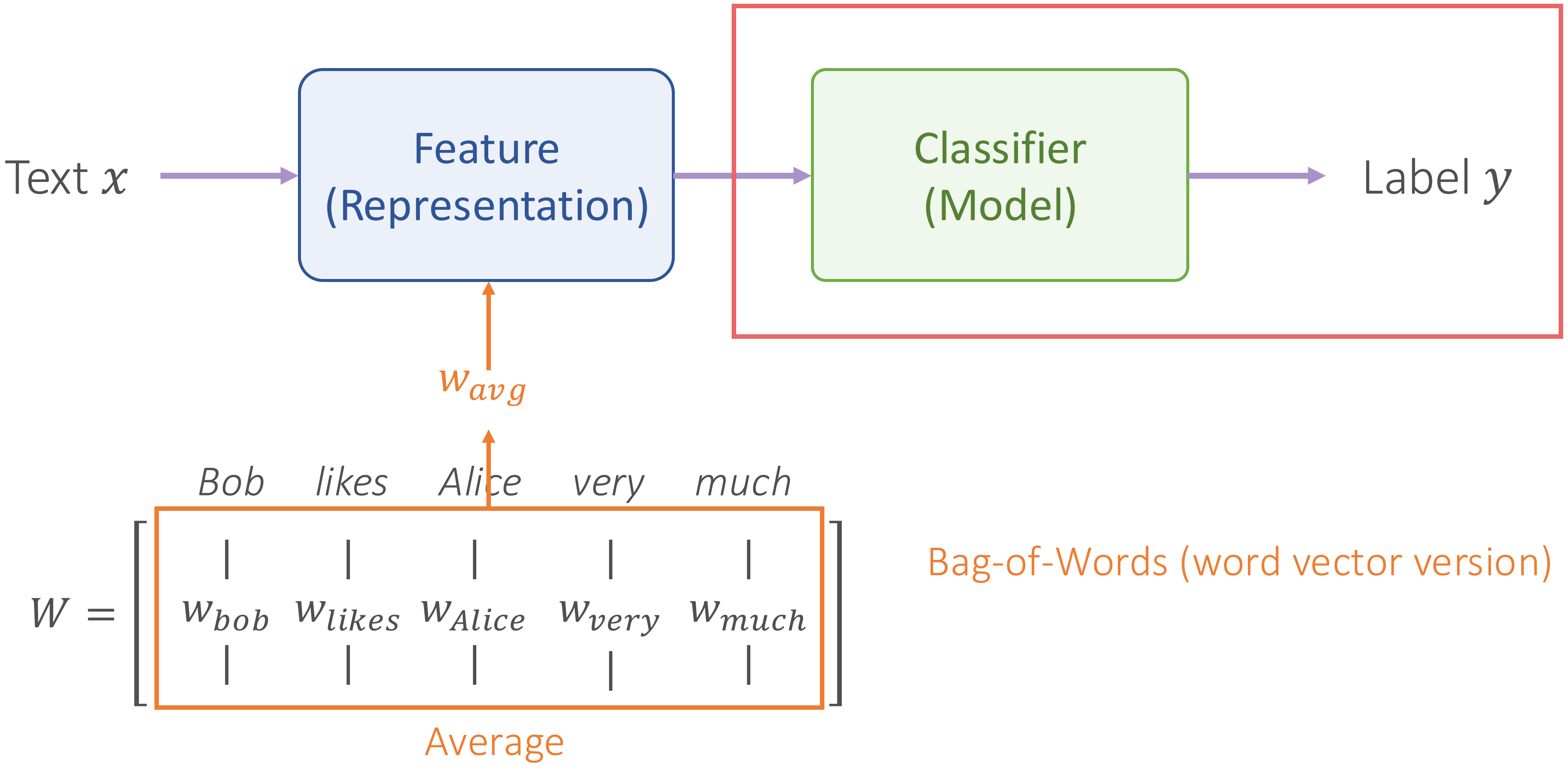


Verb tense



Country-Capital

Extrinsic Evaluation: Downstream Performance

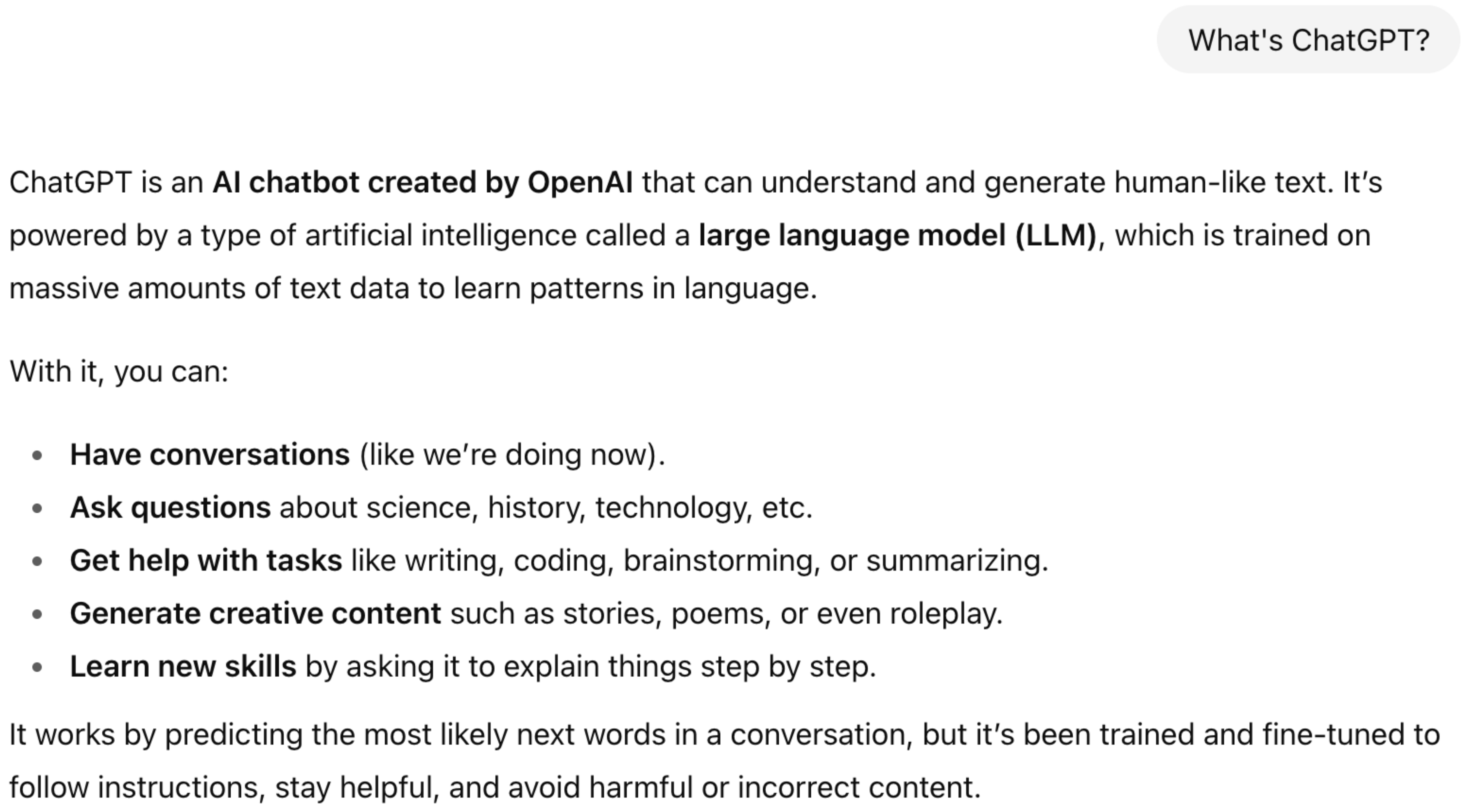


Text x
Feature (Representation)
Classifier (Model)
Label y
w_{avg}
Bob likes Alice very much
$W = \begin{bmatrix} | & | & | & | & | \\ w_{bob} & w_{likes} & w_{Alice} & w_{very} & w_{much} \\ | & | & | & | & | \end{bmatrix}$
Average

Bag-of-Words (word vector version)

Question?

Language Models



What's ChatGPT?

ChatGPT is an **AI chatbot created by OpenAI** that can understand and generate human-like text. It's powered by a type of artificial intelligence called a **large language model (LLM)**, which is trained on massive amounts of text data to learn patterns in language.

With it, you can:

- **Have conversations** (like we're doing now).
- **Ask questions** about science, history, technology, etc.
- **Get help with tasks** like writing, coding, brainstorming, or summarizing.
- **Generate creative content** such as stories, poems, or even roleplay.
- **Learn new skills** by asking it to explain things step by step.

It works by predicting the most likely next words in a conversation, but it's been trained and fine-tuned to follow instructions, stay helpful, and avoid harmful or incorrect content.

What are Language Models?

- A probabilistic model of a sequence of words
  - Evaluate the probability of whether a text is acceptable
- How likely are the following sentences?

The dog is barking at the stranger in the yard.

Yesterday, I went to the park and saw a group of children playing soccer.

The sky colorful because painted an artist.

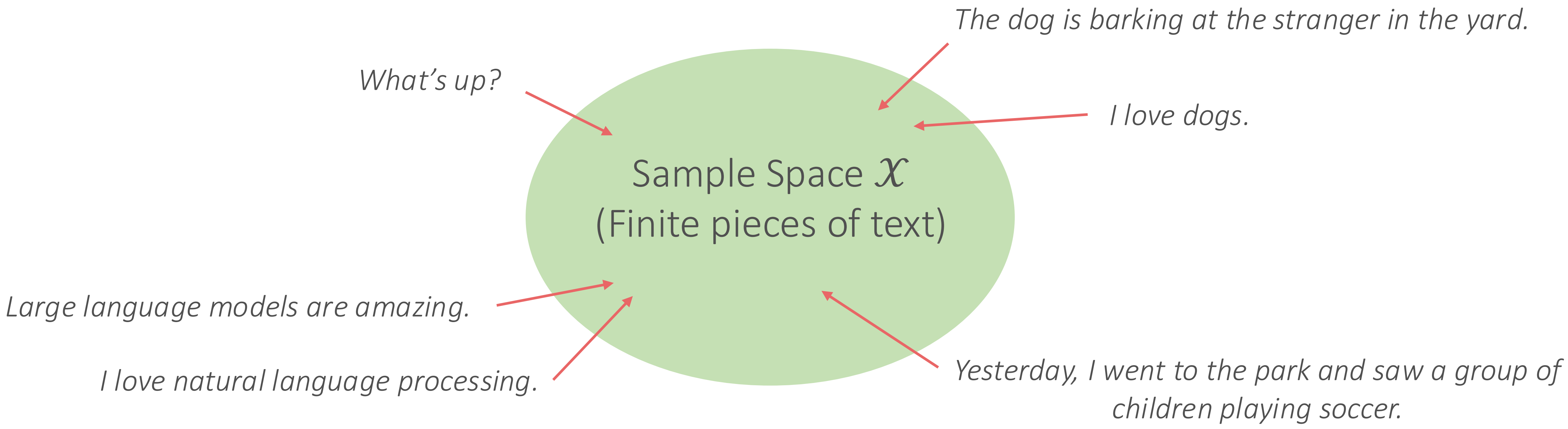
Cats upon they chairs sleeping their dreams fall.

Plorp zix flanned the quibble through treemunk.

Language Models

- Learn the probability distribution over texts $x = [w_1, w_2, \ldots, w_l] \in \mathcal{X}$

$$P(x) = P(w_1, w_2, \ldots, w_l)$$


The dog is barking at the stranger in the yard.
What’s up?
I love dogs.
Sample Space $\mathcal{X}$
(Finite pieces of text)
Large language models are amazing.
I love natural language processing.
Yesterday, I went to the park and saw a group of children playing soccer.

What Can Language Models Do?

- Score texts

$P(\textit{The dog is barking at the stranger in the yard.})$ → High

$P(\textit{Cats upon they chairs sleeping their dreams fall.})$ → Low

- Generate texts

$$\tilde{x} \sim P(\mathcal{X})$$

Auto-Regressive Language Models

$$
\begin{aligned}
P(w_1, w_2, w_3, \ldots, w_l) &= P(w_1)P(w_2, w_3, \ldots, w_l | w_1) \\
&= P(w_1)P(w_2|w_1)(w_3, \ldots, w_l | w_1, w_2) \\
&= P(w_1)P(w_2|w_1)P(w_3|w_1, w_2)(w_4, \ldots, w_l | w_1, w_2, w_3) \\
&= \prod_{i=1}^{l} P(w_i | w_1, w_2, \ldots, w_{i-1})
\end{aligned}
$$

$$
\begin{aligned}
P(\textit{She likes to go hiking}) = P(\textit{She}) \cdot P(\text{likes}|\text{She}) \cdot P(\text{to}|\text{She likes}) \\
\cdot P(\text{go}|\text{She likes to}) \cdot P(\text{hiking}|\text{She likes to go})
\end{aligned}
$$

Auto-Regressive Language Models

$$P(w_1, w_2, w_3, \dots, w_l) = \prod_{i=1}^{l} P(\underbrace{w_i}_{\text{Next Token}} | \underbrace{w_1, w_2, \dots, w_{i-1}}_{\text{Context}})$$

Next token prediction problem based on context

Challenge: How to predict $P(w_i | w_1, w_2, \dots, w_{i-1})$?

Unigram Language Models

Assumption: $P(w_i|w_1, w_2, \ldots, w_{i-1}) \approx P(w_i)$

$$P(w_1, w_2, w_3, \ldots, w_l) \approx P(w_1)P(w_2)P(w_3) \ldots P(w_l)$$

Similar to the concept of bag-of-words!

How to calculate $P(w_i)$?

A count-based solution: Collect training corpus and count

$$P(w_i) = \frac{C_{train}(w_i)}{\sum_t C_{train}(w_t)}$$

Bigram Language Models

Assumption: $P(w_i|w_1, w_2, \dots, w_{i-1}) \approx P(w_i|w_{i-1})$

$$P(w_1, w_2, w_3, \dots, w_l) \approx P(w_1)P(w_2|w_1)P(w_3|w_2)P(w_4|w_3)\dots P(w_l|w_{l-1})$$

The prediction of the next token depends only on the previous token

A count-based solution: Collect training corpus and count

$$P(w_i|w_{i-1}) = \frac{C_{train}(w_{i-1}, w_i)}{C_{train}(w_{i-1})}$$

Trigram Language Models

Assumption: $P(w_i|w_1, w_2, \dots, w_{i-1}) \approx P(w_i|w_{i-2}, w_{i-1})$

$$P(w_1, w_2, w_3, \dots, w_l) \approx P(w_1)P(w_2|w_1)P(w_3|w_1, w_2)P(w_4|w_2, w_3) \dots P(w_l|w_{l-2}, w_{l-1})$$

The prediction of the next token depends on the previous **two** tokens

A count-based solution: Collect training corpus and count

$$P(w_i|w_{i-1}, w_{i-2}) = \frac{C_{train}(w_{i-2}, w_{i-1}, w_i)}{C_{train}(w_{i-2}, w_{i-1})}$$

N-Gram Language Models

Assumption: $P(w_i|w_1, w_2, \dots, w_{i-1}) \approx P(w_i|w_{i-n+1}, \dots, w_{i-1})$

$$P(w_1, w_2, w_3, \dots, w_l) \approx \prod_i P(w_i|w_{i-n+1}, \dots, w_{i-1})$$

The prediction of the next token depends on the previous **n** tokens

A count-based solution: Collect training corpus and count

$$P(w_i|w_{i-n+1}, \dots, w_{i-1}) = \frac{C_{train}(w_{i-n+1}, \dots, w_{i-1}, w_i)}{C_{train}(w_{i-n+1}, \dots, w_{i-1})}$$

How to Evaluate Language Models?

- A good language model should assign higher probability to typical, grammatically correct sentences
- Train a language model on a suitable training corpus
    - Assumption: observed sentences ≈ good sentences
- Test on different, unseen corpus
    - The higher probability that the language model assigns to the test set, the better (why?)
- Evaluation metric: Perplexity

Perplexity (PPL)

For a corpus $\mathcal{X}$ with sentences $\{x_1, x_2, \ldots, x_n\}$

Likelihood $$P(\mathcal{X}) = \prod_{i=1}^{n} P(x_i)$$ The higher, the better

Log-Likelihood $$\log P(\mathcal{X}) = \sum_{i=1}^{n} \log P(x_i)$$ The higher, the better

Per-Word Log-Likelihood $$WLL(\mathcal{X}) = \frac{1}{W} \sum_{i=1}^{n} \log P(x_i)$$ The higher, the better

The number of words in the test corpus

Perplexity (PPL)

For a corpus $\mathcal{X}$ with sentences $\{x_1, x_2, \dots, x_n\}$

Perplexity $$e^{-WLL(\mathcal{X})}$$

The lower, the better

$$WLL(\mathcal{X}) = \frac{1}{W}\sum_{i=1}^{n} \log P(x_i)$$

Computed by language model

Unigram Language Model

$$P\left(w_j \middle| w_1, w_2, \dots, w_{j-1}\right) \approx P(w_j)$$

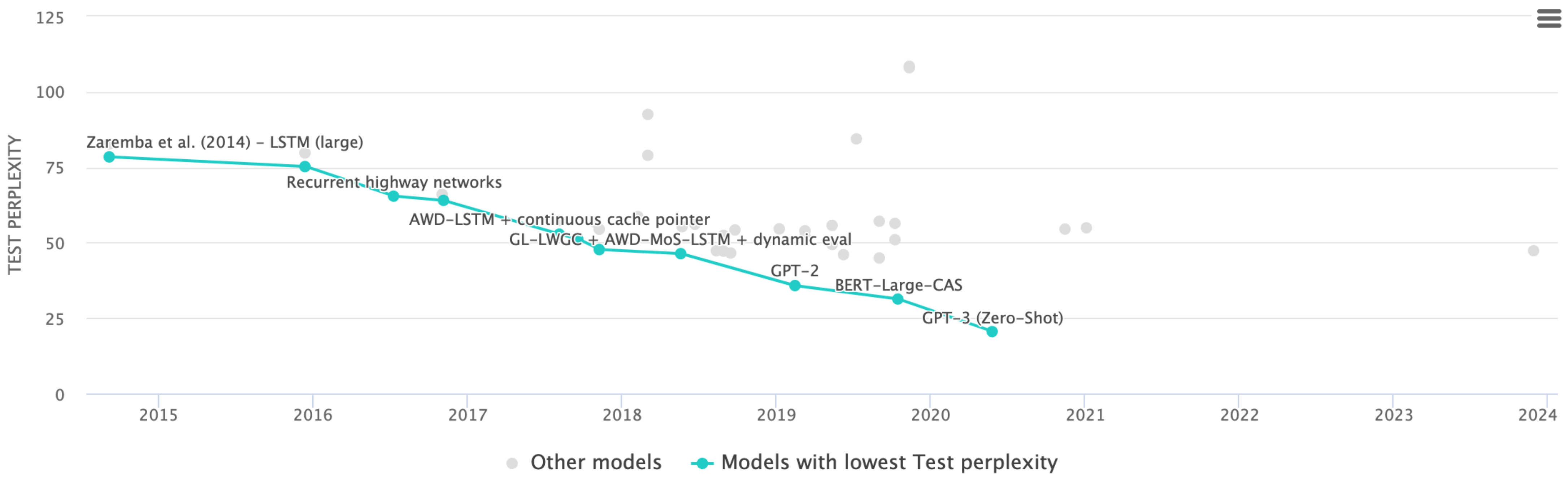
$$P(x) = \prod_j P(w_j)$$

Unigram Language Model

$$WLL(\mathcal{X}) = \frac{1}{W}\sum_i \sum_j \log P(w_{i,j})$$

Minimizing perplexity → maximizing probability of corpus

Perplexity (PPL)



https://paperswithcode.com/sota/language-modelling-on-penn-treebank-word

Text Generation with Language Models

Trigram Language Model

$$P(w_1, w_2, w_3, \dots, w_l) \approx \prod_i P(w_i | w_{i-2}, w_{i-1})$$

- Generate the first word $w_1 \sim P(w)$
- Generate the second word $w_2 \sim P(w|w_1)$
- Generate the third word $w_3 \sim P(w|w_1, w_2)$
- Generate the fourth word $w_4 \sim P(w|w_2, w_3)$
- Generate the fifth word $w_5 \sim P(w|w_3, w_4)$
- …
- Until the end of the sentence <eos>

Generation Examples

Unigram	*release millions See ABC accurate President of Donald Will cheat them a CNN megynkelly experience @ these word out- the*
Bigram	*Thank you believe that @ ABC news, Mississippi tonight and the false editorial I think the great people Bill Clinton . ''*
Trigram	*We are going to MAKE AMERICA GREAT AGAIN! #MakeAmericaGreatAgain https: //t.co/DjkdAzT3WV*

Typical LMs are not sufficient to handle long-range dependencies

"Alice/Bob could not go to work that day because she/he had a doctor's appointment"