Comparative Study of Transformer-Based Embeddings for Topic Coherence

Shengtao (Alex) Ding Tarun Rapaka Jason Yang

Research Mentor: Dr. Willy Rodriguez

Fall-Term PRIMES Conference

October 18, 2025

Outline

Background

- **Natural Language Processing (NLP)** — A field of artificial intelligence that lets machines interpret human text.
- **Topic Modeling** — grouping related content within a corpus to make large collections navigable.
- **Motivation** — new Large Language Models (like the one behind ChatGPT) have an even better performance for extracting topics.

Research question:
Can *smaller* models give topics as clear as *big* ones?

Word Cloud

Bigger words appear more often in the text corpus.
Here: *Lord of the Rings* movie dialogues (Kaggle dataset).

Counts are computed after basic cleaning (lowercasing, removing stop words).

Example of Preprocessing

First, we apply **Preprocessing**, which cleans the text for use, helping to reduce noise and keep models robust.

Example sentence:

`"The quick brown fox jumps over the dog!"`

1. **Tokenize** — split into words

   `[The, quick, brown, fox, jumps, over, the, dog]`

2. **Normalize** — lowercase, remove punctuation

   `[the, quick, brown, fox, jumps, over, the, dog]`

3. **Remove stop words** — uninformative words (e.g., “the”, “over”) that add little meaning

   `[quick, brown, fox, jumps, dog]`

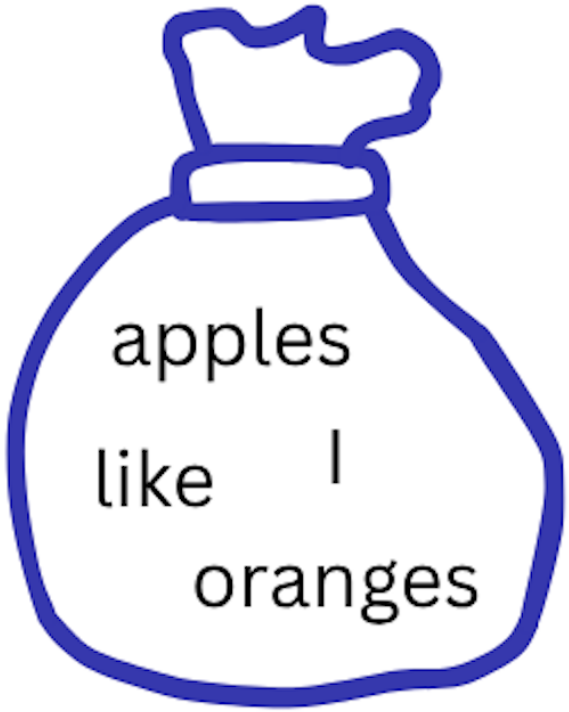
Next, we can move on to **vectorization.**

Bag of Words (BoW)

- Build a vocabulary of all unique words in the dataset
- Count the number of times each word appears in each document
- Represent each document as a vector of word counts

Example:
Sentence 1: *"I like apples"*
Sentence 2: *"I like oranges"*

	I	like	apples	oranges
S_1	1	1	1	0
S_2	1	1	0	1



"Bag" of words — orderless but countable.

TF–IDF

What it does: weights words by importance in a document and rarity in the corpus.

Advantage: highlights distinctive terms; down-weights very common words.

$$\boxed{\mathrm{tfidf}(w, d) = \mathrm{tf}(w, d) \cdot \log\left(\frac{N}{\mathrm{df}(w)} + 1\right)}$$

Variable glossary:

w	word/term
d	a single document
N	total number of documents in corpus
$\mathrm{tf}(w, d)$	frequency (raw or normalized) of w in d
$\mathrm{df}(w)$	number of documents containing w

TF–IDF: Worked Example

Mini-corpus ($N = 3$):

1. d_1: `large language models`
2. d_2: `language models are powerful`
3. d_3: `models revolutionize nlp`

For the term "language" in d_2:

$$\text{tf}(\text{language}, d_2) = \frac{1}{4}, \text{df}(\text{language}) = 2, \text{idf} = \log\left(\frac{3}{2} + 1\right) \approx 0.916$$

$$\boxed{\text{tfidf} = 0.25 \times 0.916 \approx 0.229}$$

Partial TF–IDF matrix:

	language	models	revolutionize
d_1	0.305	0.231	0
d_2	0.229	0.173	0
d_3	0	0.231	0.462

Values rounded; TF normalized by document length.

Tradeoffs of Bag of Words and TF-IDF

- **Pros:**
  - Simple to understand and implement.
  - Fast computation.
- **Cons:**
  - Cannot capture context.
  - Ignores word order:
    - *Sentence 1:* "I like NLP but I don't like AI"
    - *Sentence 2:* "I like AI but I don't like NLP"

	I	like	NLP	AI	don't
S1	2	2	1	1	1
S2	2	2	1	1	1

    Same word frequencies $\Rightarrow$ same vector representation
  - Vectors can be large and sparse with big vocabularies.

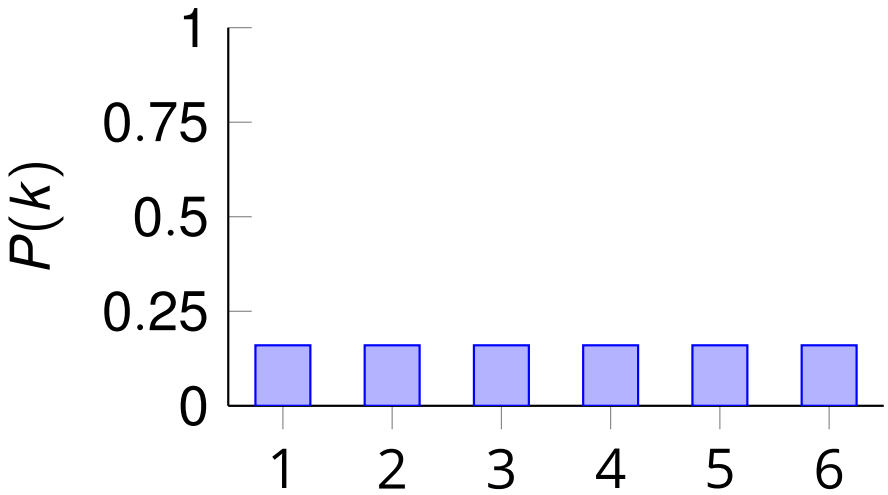
Discrete Probability Distribution: Intuitive example

Imagine you have a **fair six-sided die.** Each side (1 to 6) has the same chance of appearing:

$$P(1) = P(2) = P(3) = P(4) = P(5) = P(6) = \frac{1}{6}.$$

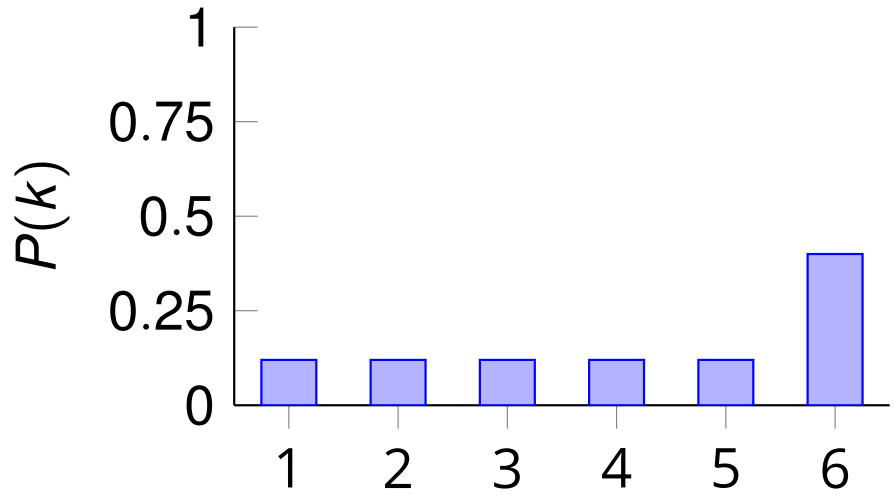
If we draw a bar chart showing those probabilities, all the bars would be equal. A **probability distribution** tells us how likely each possible outcome is.

Fair Die (Uniform Distribution)



Every side has the same probability.

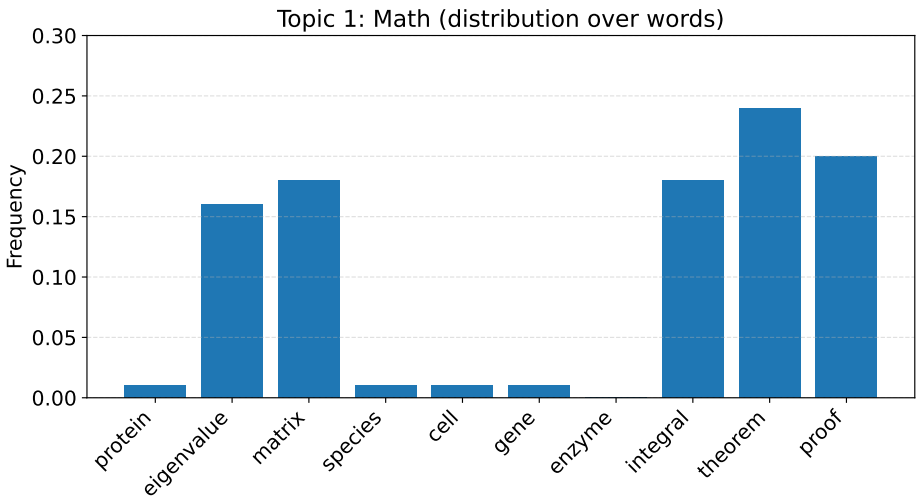
Loaded Die (Biased Distribution)



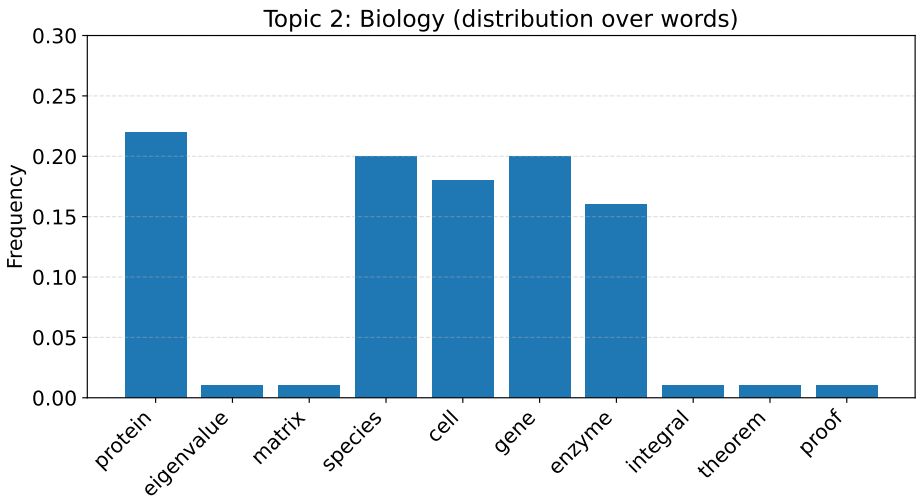
The die is *loaded* — rolling a 6 is more likely.

Example: One Document as a Mixture of Two Topics

- **Mixture for document** d: $\theta_d = (0.20, 0.80)$ over topics Math, Bio.
- **Document word model**: $p(w \mid d) = 0.2\,\phi_{\text{Math},w} + 0.8\,\phi_{\text{Bio},w}$.



Topic → word: Math (ϕ_{Math})



Topic → word: Bio (ϕ_{Bio})

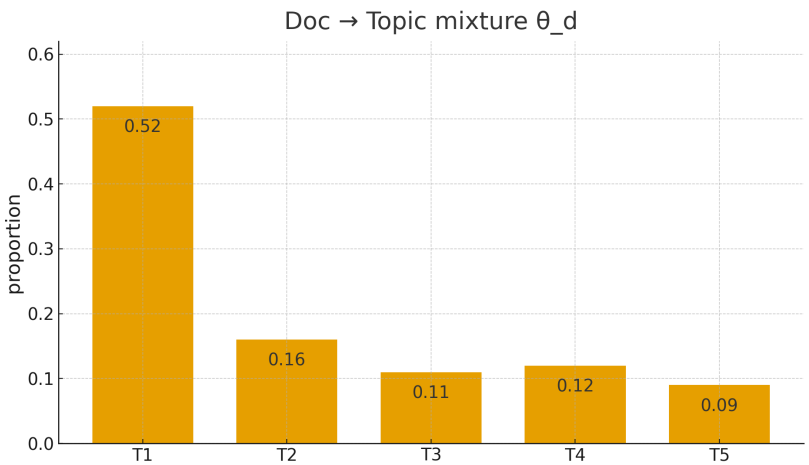
Notation: θ_d = document d's topic mixture; $\phi_{k,w}$ = prob. of word w under topic k; $p(w \mid d)$ mixes them with weights (0.2, 0.8).

How LDA Models a Document

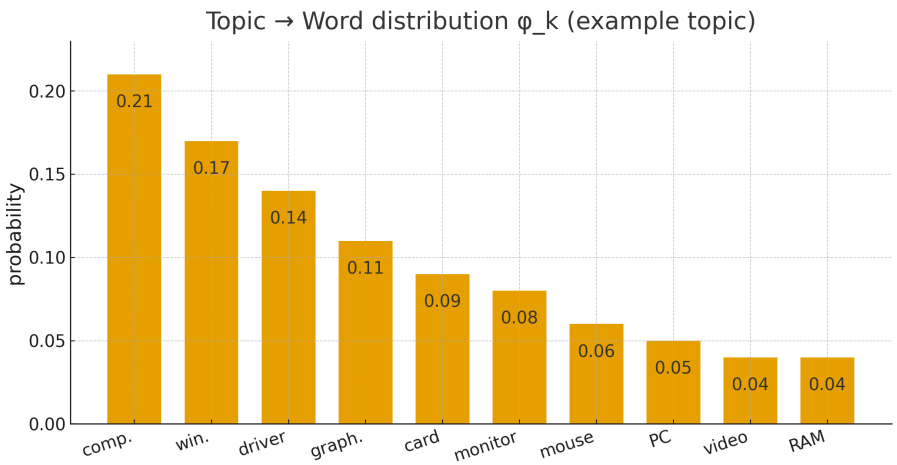
- Token sampling: first choose a topic using θ_d, then choose a word using that topic's ϕ.

Priors: $\theta_d \in \Delta^{K-1}$, $\phi_k \in \Delta^{V-1}$
Per token n: $z_{d,n} \leftarrow \theta_d$, $w_{d,n} \leftarrow \phi_{z_{d,n}}$

Doc → Topic mixture θ_d



Topic → Word distribution ϕ_k



Topic Modeling & Latent Dirichlet Allocation (LDA)

- **LDA**: discovers hidden *topics* in an unlabeled corpus.
- **Topic**: probability distribution over the vocabulary (ϕ_k).
- **Outputs**: document→topic Θ, topic→word Φ.

$$\text{Topic } k: \ \phi_k \in \Delta^{V-1}, \quad \sum_{i=1}^{V} \phi_{k,i} = 1, \ \phi_{k,i} \geq 0$$

$$\text{Doc } d: \ \theta_d \in \Delta^{K-1}, \quad \sum_{k=1}^{K} \theta_{d,k} = 1, \ \theta_{d,k} \geq 0$$

Simplex: $\Delta^{m-1} = \{x \in \mathbb{R}^m_{\geq 0} : \sum_{j=1}^{m} x_j = 1\}$. **Symbols:** D=#docs, V=vocab size, K=#topics, $d \in \{1..D\}$, $k \in \{1..K\}$, $i \in \{1..V\}$, $\Phi \in \mathbb{R}^{K \times V}$ (row k is ϕ_k), $\Theta \in \mathbb{R}^{D \times K}$ (row d is θ_d).

Quality of a Topic Decomposition

- **Goal**: interpretable topics and non-redundant coverage.
- **Key question**: do a topic's top words co-occur in documents?
- **Also**: avoid overlapping topics; prefer stable results across runs.

$$\text{Quality} \approx \underbrace{\text{Coherence}}_{\text{interpretability}} + \underbrace{\text{Separation}}_{\text{non-redundancy}}$$

Symbols reminder: K = #topics, V = vocab size; $\phi_k \in \Delta^{V-1}$ are topic$\to$word distributions, $\theta_d \in \Delta^{K-1}$ are doc$\to$topic mixtures.

Metrics for Topic Quality

Coherence (interpretability): do top words co-occur above chance?

$$\mathrm{coh}_{\mathrm{NPMI}}(W) = \frac{1}{|W|(|W|-1)} \sum_{\substack{i,j \in W \\ i \neq j}} \frac{\log\left(\dfrac{P(w_i, w_j)}{P(w_i)\,P(w_j)}\right)}{-\log P(w_i, w_j)}$$

Defs: $P(w)$ = marginal probability of word w (fraction of docs or windows containing w); $P(w_i, w_j)$ = co-occurrence probability (same doc or within a window). Bounded in $[-1, 1]$.

$$C_{\mathrm{UMass}}(T) = \frac{2}{M(M-1)} \sum_{j=2}^{M} \sum_{i=1}^{j-1} \log\left(\frac{D(w_j, w_i) + 1}{D(w_i)}\right)$$

Where: $T = (w_1, \ldots, w_M)$ ordered top words; $D(\cdot)$ = doc counts; $+1$ = smoothing; higher (less negative) is better.

Also track: separation (pairwise JS/Hellinger on $\{\phi_k\}$) and stability across seeds.

One-Hot Encoding

- Simplest bag-of-words representation: every token is mapped to a binary vector of length $|V|$.
- Exactly one entry is 1 (the token's index); all others are 0.

Example vocabulary:
$V = \{\texttt{large}, \texttt{language}, \texttt{model}, \texttt{revolutionize}\}$

$$\text{one\_hot}(V) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Neural Networks

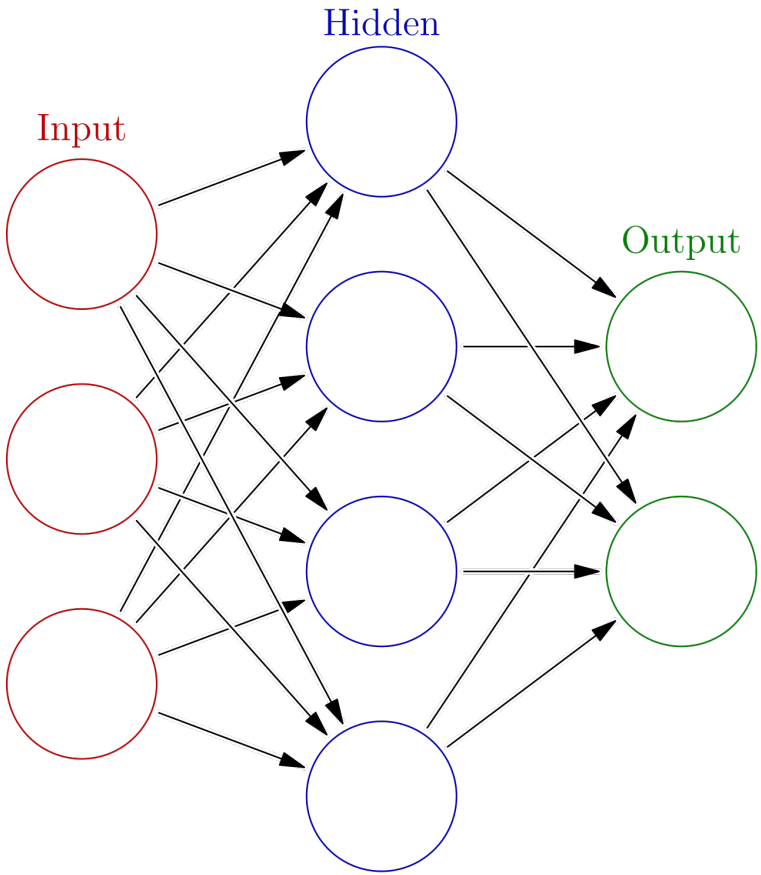
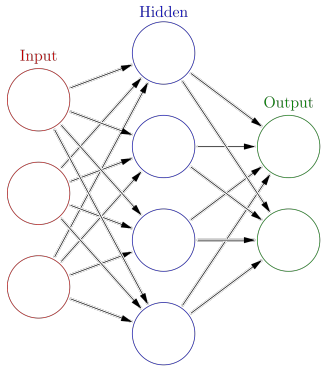


Figure: A pictorial representation of a neural network. (Source: Wikipedia)

Dense Layer Operations Illustrated in the Graph



Architecture: $\mathbb{R}^3 \xrightarrow{W_1, b_1,\ \mathrm{ReLU}} \mathbb{R}^4 \xrightarrow{W_2, b_2} \mathbb{R}^2$

Matrix forms

$$\underbrace{\mathbf{x}}_{\in \mathbb{R}^3} \xrightarrow{\mathbf{W}_1 \in \mathbb{R}^{4\times 3},\ \mathbf{b}_1 \in \mathbb{R}^4} \mathbf{a} = \mathbf{W}_1\mathbf{x} + \mathbf{b}_1 \quad , \quad \underbrace{\mathbf{h}}_{\in \mathbb{R}^4} = \mathrm{ReLU}(\mathbf{a})$$

$$\underbrace{\mathbf{y}}_{\in \mathbb{R}^2} = \underbrace{\mathbf{W}_2}_{\in \mathbb{R}^{2\times 4}} \mathbf{h} + \underbrace{\mathbf{b}_2}_{\in \mathbb{R}^2}$$

Notes.

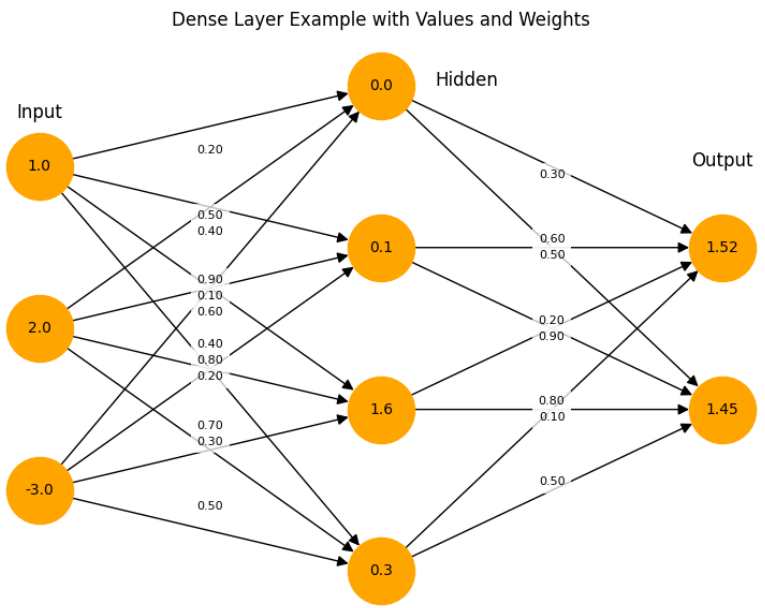
- $\mathrm{ReLU}(t) = \max(0, t)$ applied elementwise. Here, the hidden layer uses ReLU and the output layer is linear (no activation).

Dense Layer Example with Numerical Values

Given:

$$\mathbf{x} = \begin{bmatrix} 1 \\ 2 \\ -3 \end{bmatrix}, \quad \mathbf{W}_1 = \begin{bmatrix} 0.2 & 0.4 & 0.6 \\ 0.5 & 0.1 & 0.2 \\ 0.9 & 0.8 & 0.3 \\ 0.4 & 0.7 & 0.5 \end{bmatrix}$$

$$\mathbf{W}_2 = \begin{bmatrix} 0.3 & 0.5 & 0.9 & 0.1 \\ 0.6 & 0.2 & 0.8 & 0.5 \end{bmatrix}$$



Hidden layer:

$$\mathbf{h} = \text{ReLU}(\mathbf{W}_1\mathbf{x}) = \begin{bmatrix} 0 \\ 0.1 \\ 1.6 \\ 0.3 \end{bmatrix}$$

Output layer:

$$\mathbf{y} = \mathbf{W}_2\mathbf{h} = \begin{bmatrix} 1.52 \\ 1.45 \end{bmatrix}$$

Word Embeddings

- A **word embedding** is a function $\mathcal{C}$ from the vocabulary V to $\mathbb{R}^d$ for some d.
- Think of it as a way to assign each word in our vocabulary a real-valued vector which accurately captures the meaning of the word.
- We would expect, for instance, that

$$\mathcal{C}(\text{"king"}) - \mathcal{C}(\text{"man"}) \approx \mathcal{C}(\text{"queen"}) - \mathcal{C}(\text{"woman"}).$$

- The most popular embedding algorithm is **word2vec**, which makes use of neural networks and one-hot encoding.
  - One problem with this approach: "The bank vault lies on the bank of the river." In this sentence, "bank" has two meanings but only one embedding.

Transformer Structure and Self Attention

The transformer architecture, specifically the **self-attention mechanism**, solve the issue of dealing with contextual information.

- **Self attention**

$$\boxed{\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) \cdot V}$$

where Q, K, V are matrices and d_K is a real number (usually a dimension of Q or K).

- Letting $x \in \mathbb{R}^n$, the **softmax** function is defined as

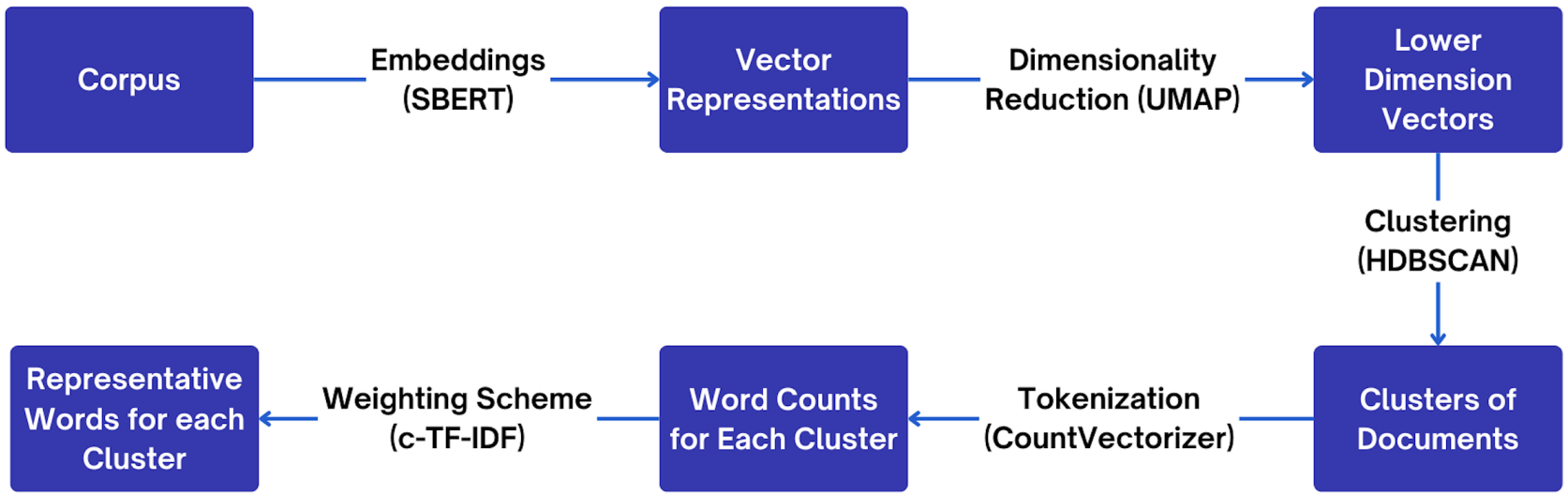
$$\boxed{\text{softmax}(\mathbf{x}) = \left(\frac{e^{x_1}}{e^{x_1} + \cdots + e^{x_n}}, \ldots, \frac{e^{x_n}}{e^{x_1} + \cdots + e^{x_n}}\right).}$$

Our question

- With these advanced tools (specifically the transformer architecture), it is hence possible to capture the semantic relations inside a text.
- If one can capture these semantic relations, topic decomposition can become more accurate.
- Our question: **How good are small models (in terms of #parameters) with respect to larger ones in regards to topic decomposition?**

BERTopic Pipeline

- **BERTopic** is a topic modeling technique that finds topics or themes across documents in a corpus
- The pipeline follows these steps where each step is independent from the others, meaning any building block could be replaced:



Results & Discussion

- **Bigger $\neq$ always better**: small/medium models gave similar topic coherence/divergence.
- **Experiments applied on two datasets**: the *20 newsgroups* and a subset of articles downloaded from *PubMed*.
- **Reproducible pipeline**: models on Hugging Face · code/data on GitHub.

Encoder	Size	20Newsgroups		PubMed	
		Coherence	Diversity	Coherence	Diversity
all-MiniLM-L6-v2	22M	0.7422	0.9947	0.7004	0.9954
microsoft/MiniLM-L12-H384-uncased	33M	0.7374	0.9947	0.7069	0.9951
distilbert-base-uncased	66M	0.7450	0.9955	0.7137	0.9954
bert-base-uncased	110M	0.7253	0.9946	0.7012	0.9955
roberta-base	125M	0.7420	0.9950	0.7121	0.9949
meta-llama/Llama-2-7b-hf	7B	0.7310	0.9946	0.7047	0.9952
meta-llama/Llama-2-13b-hf	13B	0.7447	0.9948	0.6977	0.9954

Table: Coherence and diversity on 20 Newsgroups and PubMed dataset

Takeaways

- **Vectorization** is the first step toward machine-readable text
- **LDA vs. BERTopic**: LDA excels in interpretability; BERTopic leverages contextual embeddings and scalability
- **Objective metrics** (coherence & divergence) guide model and parameter selection

Acknowledgements

- **Research mentor:** Sincere thanks to Dr. Willy Rodriguez for continuous guidance and support.
- **Scientific contributors:** We thank Dr. Felix Gotti and Dr. Tanya Khovanova for helpful discussions and feedback related to this project.
- **Programs & organizers:** Grateful to the MIT PRIMES program and organizers for providing this research opportunity and community.
- **Parents & friends:** Glad to have y'all for coming or supporting us on our way.
- **Audiences:** Thank you for being here for our talk.

References

D. D. Lee and H. S. Seung.
Learning the parts of objects by non-negative matrix factorization.
Nature, 401(6755):788–791, 1999.

K. Stevens, P. Kegelmeyer, D. Andrzejewski, and D. Buttler.
Exploring topic coherence over many models and many topics.
In *Proceedings of EMNLP-CoNLL*, pages 952–961, 2012.

R. Deveaud, E. SanJuan, and P. Bellot.
Accurate and effective latent concept modeling for ad hoc information retrieval.
Document Numérique, 17(1):61–84, 2014.

P. J. O'Hara and M. W. Davies.
Two-stage topic modelling of scientific publications: A case study.
PLOS ONE, 2021.

A. Vaswani et al.
Attention is All You Need.
Advances in neural information processing systems, 2017.

L. McInnes, J. Healy, and J. Melville.
UMAP: Uniform manifold approximation and projection for dimension reduction.
arXiv preprint arXiv:1802.03426, 2018.

M. Grootendorst.
BERTopic: Neural topic modeling with a class-based TF-IDF procedure.
arXiv preprint arXiv:2203.05794, 2022.

D. M. Blei, A. Y. Ng, and M. I. Jordan.
Latent Dirichlet Allocation.
Journal of Machine Learning Research, 3:993–1022, 2003.

Q&A!