Large Language Models and Machine Learning for Unstructured Data

Lecture 1:
Large Language Models

Stephen Hansen
University College London

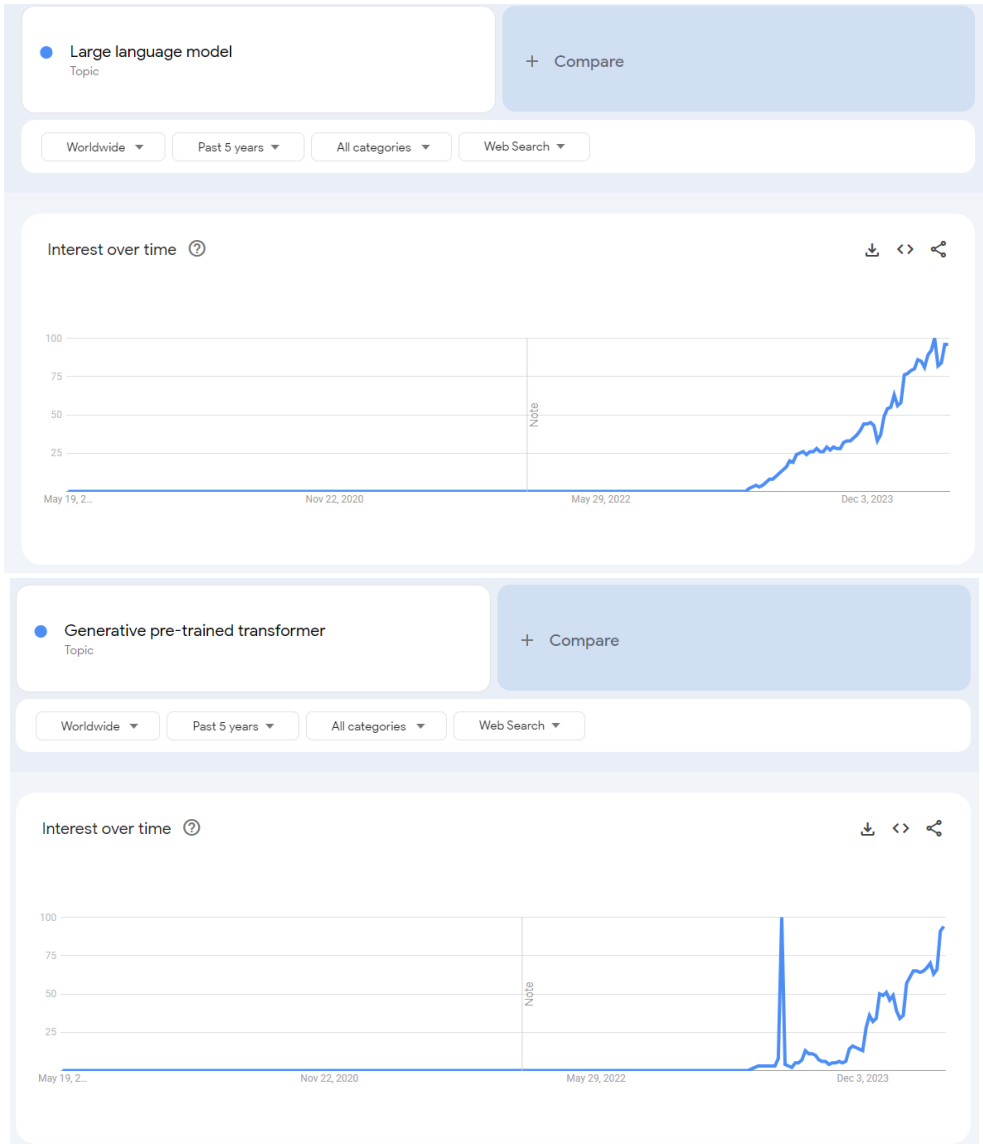
Reading

Background material for the lecture:

1. Ash and Hansen, *Text Algorithms in Economics*, `https://bityl.co/QQRC`
2. Jurafsky and Martin, *Speech and Language Processing*, Ch. 6 `https://bityl.co/QQRS`
3. Jurafsky and Martin, *Speech and Language Processing*, Ch. 10 `https://bityl.co/QQRX`

All lecture materials and example notebooks available from `https://github.com/unstructured-data/IESE-FRA-seminar-24`.

Surge in Interest in AI for Natural Language



What is New Here?

Text data has a long(ish) history in accounting, economics, and finance.

Primary motivation is to measure important phenomena difficult to capture with traditional data.

Well-known examples include competition, policy uncertainty, sentiment, polarization, etc.

Automated methods have largely counted words or their co-occurrence.

Main limitation is limited understanding of context: "economic growth is weak but long-term productivity trends are strong" vs. "economic growth is strong but long-term productivity trends are weak."

Deep Learning

The key breakthrough in modeling context came with deep learning models with Transformer architecture [Vaswani et al., 2017].

Contrast in approach from previous approaches to language:

- Expert systems
- Raw feature counts
- Dimensionality reduction of feature counts

Compared to previous models, LLMs (i) are easier to use; (ii) better incorporate contextual data; (iii) produce more human-like output; (iv) have an increasingly impenetrable internal structure.

Outline

1. Background
2. Word2vec
3. Attention and Word Prediction
4. Applications

Background

Notation

The corpus is composed of D documents indexed by d.

After pre-processing, each document is a finite, length-N_d list of terms $\mathbf{w}_d = (w_{d,1}, \ldots, w_{d,N_d})$ with generic element $w_{d,n}$.

Let $\mathbf{w} = (\mathbf{w}_1, \ldots, \mathbf{w}_D)$ be a list of all terms in the corpus, and let $N \equiv \sum_d N_d$ be the total number of terms in the corpus.

Suppose there are V **unique** terms in $\mathbf{w}$, where $1 \leq V \leq N$, each indexed by v.

We can then map each term in the corpus into this index, so that $w_{d,n} \in \{1, \ldots, V\}$.

Example

Consider three documents:

1. 'stephen is nice'
2. 'john is also nice'
3. 'george is mean'

We can consider the set of unique terms as $\{\text{stephen}, \text{is}, \text{nice}, \text{john}, \text{also}, \text{george}, \text{mean}\}$ so that $V = 7$.

Construct the following index:

stephen	is	nice	john	also	george	mean
1	2	3	4	5	6	7

We then have $\mathbf{w}_1 = (1, 2, 3)$; $\mathbf{w}_2 = (4, 2, 5, 3)$; $\mathbf{w}_3 = (6, 2, 7)$.

Which Corpus?

Much of traditional text-as-data analysis fits models on corpora drawn from domain of interest.

Large language models were first fit on generic corpora like Common Crawl, Wikipedia, or Google Books.

More recent iterations expand the training data (but details becoming more obscure).

Important to realize that the training data contains the knowledge that a model can encode.

Any biases in the training data can also be inherited by the model.

How to Preprocess?

Preprocessing in traditional text-as-data analysis follows a standard sequence of steps:

1. Break strings into individual tokens.
2. Remove common and rare words.
3. Fold into common linguistic roots (lower case, stem/lemmatize)

Detail outside scope of course but see `https://bityl.co/QIKa` for code demonstration.

How to Preprocess?

Preprocessing in traditional text-as-data analysis follows a standard sequence of steps:

1. Break strings into individual tokens.
2. Remove common and rare words.
3. Fold into common linguistic roots (lower case, stem/lemmatize)

Detail outside scope of course but see `https://bityl.co/QIKa` for code demonstration.

In LLMs there is much more limited pre-preprocessing in order to preserve meaning of language; see notebook for more details.

Document-Term Matrix

Let $x_{d,v}$ be the count of term v in document d.

A popular quantitative representation of text is the *document-term matrix* $\mathbf{X}$, which collects the counts $x_{d,v}$ into a $D \times V$ matrix.

In the previous example, we have

$$\mathbf{X} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

Text analysis that relies on the document-term matrix is called the bag-of-words approach.

Documents as Vectors

We can view the documents that make up the rows of $\mathbf{X}$ as vectors.

Let each vocabulary term v have its own vector $\mathbf{e}_v \in \mathbb{R}^V$ where

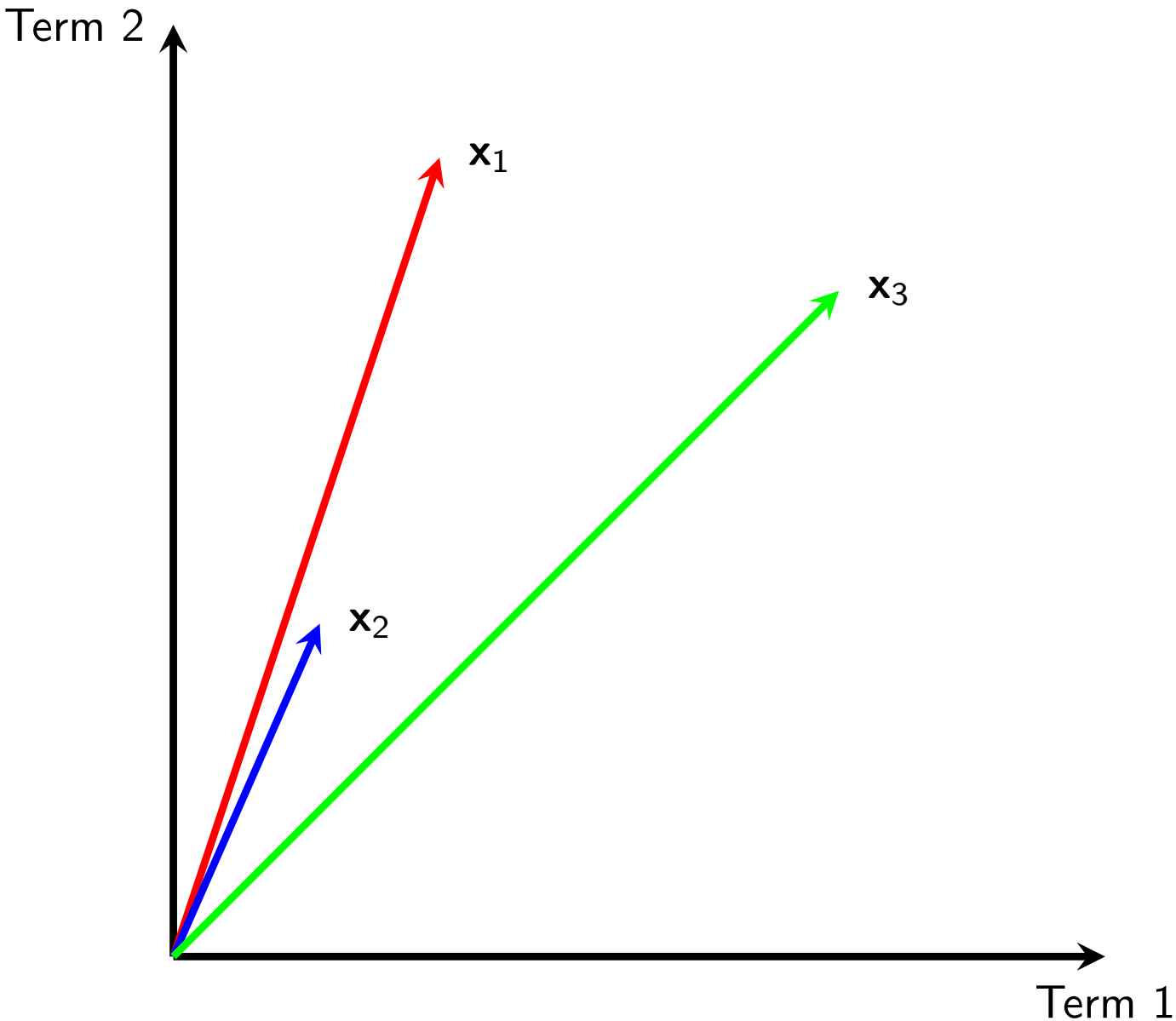
$$e_{v,v'} = \begin{cases} 1 & \text{if } v = v' \\ 0 & \text{otherwise} \end{cases}$$

Note that each term's vector is orthogonal to every other term's vector.

We can express document d as

$$\mathbf{x}_d = x_{d,1}\mathbf{e}_1 + x_{d,2}\mathbf{e}_2 + \ldots + x_{d,V}\mathbf{e}_V$$

Three Documents



Cosine Similarity

Define the cosine similarity between documents i and j as

$$CS(i,j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\|\mathbf{x}_i\| \, \|\mathbf{x}_j\|}$$

1. Since document vectors have no negative elements $CS(i,j) \in [0,1]$.
2. $\mathbf{x}_i / \|\mathbf{x}_i\|$ is unit-length, correction for different distances.

Limitations of Bag-of-Words + Cosine Similarity

Synonomy

economic growth is weak but long-term productivity trends are strong
economic growth is tepid but long-term productivity trends are strong

Limitations of Bag-of-Words + Cosine Similarity

Synonomy

economic growth is weak but long-term productivity trends are strong
economic growth is tepid but long-term productivity trends are strong

Polysemy

economic statistics lie about current well-being
my cat's favorite activity is to lie on our bed

Limitations of Bag-of-Words + Cosine Similarity

Synonomy

economic growth is weak but long-term productivity trends are strong
economic growth is tepid but long-term productivity trends are strong

Polysemy

economic statistics lie about current well-being
my cat's favorite activity is to lie on our bed

Sequence

economic growth is weak but long-term productivity trends are strong
economic growth is strong but long-term productivity trends are weak

Synonomy and Factor Models

The synonomy problem is similar to the factor model problem.

A large set of correlated features (words) across observations (documents) motivate dimensionality reduction.

Well-known factor models for text:

1. **Latent Semantic Analysis** [Deerwester et al., 1990]

   Application: [Iaria et al., 2018]
2. **Probabilistic LSI / NMF** [Ding et al., 2006]

   Application: [Ke et al., 2021]
3. **Latent Dirichlet Allocation** [Blei et al., 2003]

   Application: [Hansen et al., 2018]

Richer Feature Count Tabulation?

These factor models still operate on the document-term matrix and so are not aware of context.

One possible idea is to tabulate richer features, e.g. sequences of words rather than individual words.

These are called n-gram models ($n = 2$ bigram, $n = 3$ trigram).

But the number of unique length-N sequences composed of V total features is V^N.

We need to work in lower-dimensional space to compare token sequences!

Word2vec

Word Embeddings

A word embedding is a low-dimensional vector representation of a word.

Ideally in this low-dimensional vector space words with similar meanings will lie close together.

The construction of word embeddings via neural networks was a major step that preceded the development of large language models.

Word2vec [Mikolov et al., 2013a, Mikolov et al., 2013b] is a particularly well-known algorithm for the construction of word embeddings.[1]

[1] See also [Bengio et al., 2003].

Distributional Hypothesis

The distributional hypothesis states that words that share similar contexts share similar meanings.

Example from JM:

(6.1) Ongchoi is delicious sauteed with garlic.
(6.2) Ongchoi is superb over rice.
(6.3) ...ongchoi leaves with salty sauces...

And suppose that you had seen many of these context words in other contexts:

(6.4) ...spinach sauteed with garlic over rice...
(6.5) ...chard stems and leaves are delicious...
(6.6) ...collard greens and other salty leafy greens

Formalizing Local Context

Recall that $w_{d,n}$ is the nth word in document d.

The *context* of $w_{d,n}$ is a length-$2L$ window of words around $w_{d,n}$:

$$C(w_{d,n}) = [w_{d,n-L}, w_{d,n-L+1}, \ldots, w_{d,n+L-1}, w_{d,n+L}]$$

Can truncate context appropriately if window stretches past beginning or end of text.

In line with distributional hypothesis, word embedding models seek to generate similar embeddings for words that share similar contexts.

Self-Supervised Learning

The ‘meaning’ of a word is an unobserved and subjective concept.

Difficult to directly formulate an objective function.

Important conceptual idea is to formulate word prediction tasks that are solved using word embeddings.

Although word embeddings are formulated to solve prediction problems, they are nevertheless useful for the primary task of revealing meaning.

The approach of using auxiliary word prediction tasks to build high-quality embeddings is called self-supervised learning.

Prediction Tasks

There are two variants of word2vec which correspond to differing prediction tasks.

Skipgram model

1. Predict presence of each $w_{d,n-l} \in C(w_{d,n})$ given $w_{d,n}$.
2. Predict absence of randomly sampled words from the corpus given $w_{d,n}$.

Continuous Bag-of-Words model

1. Predict presence of $w_{d,n}$ given $C(w_{d,n})$.
2. Predict absence of randomly sampled words from the corpus given $C(w_{d,n})$.

Words and Context in Skipgram Model

"economic growth is weak but long-term productivity trends are strong"

Suppose $L = 2$.

Positive Examples		**Negative Examples**	
Word	Context	Word	Context
economic	growth	economic	down
economic	is	economic	towards
growth	economic	growth	inflation
growth	is	growth	mild
growth	weak	growth	very
is	economic	is	not
is	growth	is	can
is	weak	is	rate
is	but	is	how
.	.	.	.
strong	are	strong	many

The number of negative examples to sample per positive example is a modeling choice.

Parametrization of the Prediction Problems

Endow each word v in the vocabulary with an embedding vector $\boldsymbol{\rho}_v \in \mathbb{R}^K$ and a context vector $\boldsymbol{\alpha}_v \in \mathbb{R}^K$ where $K \ll V$.

The positive examples are modeled as

$$\Pr\left[\, w_{d,n-l} \in \mathrm{C}(w_{d,n}) \mid w_{d,n} \,\right] = \frac{\exp\left(\boldsymbol{\rho}_{w_{d,n}}^T \boldsymbol{\alpha}_{w_{d,n-l}}\right)}{1 + \exp\left(\boldsymbol{\rho}_{w_{d,n}}^T \boldsymbol{\alpha}_{w_{d,n-l}}\right)}$$

and the negative examples are modeled as

$$\Pr\left[\, w_{d,n-l} \notin \mathrm{C}(w_{d,n}) \mid w_{d,n} \,\right] = 1 - \frac{\exp\left(\boldsymbol{\rho}_{w_{d,n}}^T \boldsymbol{\alpha}_{w_{d,n-l}}\right)}{1 + \exp\left(\boldsymbol{\rho}_{w_{d,n}}^T \boldsymbol{\alpha}_{w_{d,n-l}}\right)}$$

Example

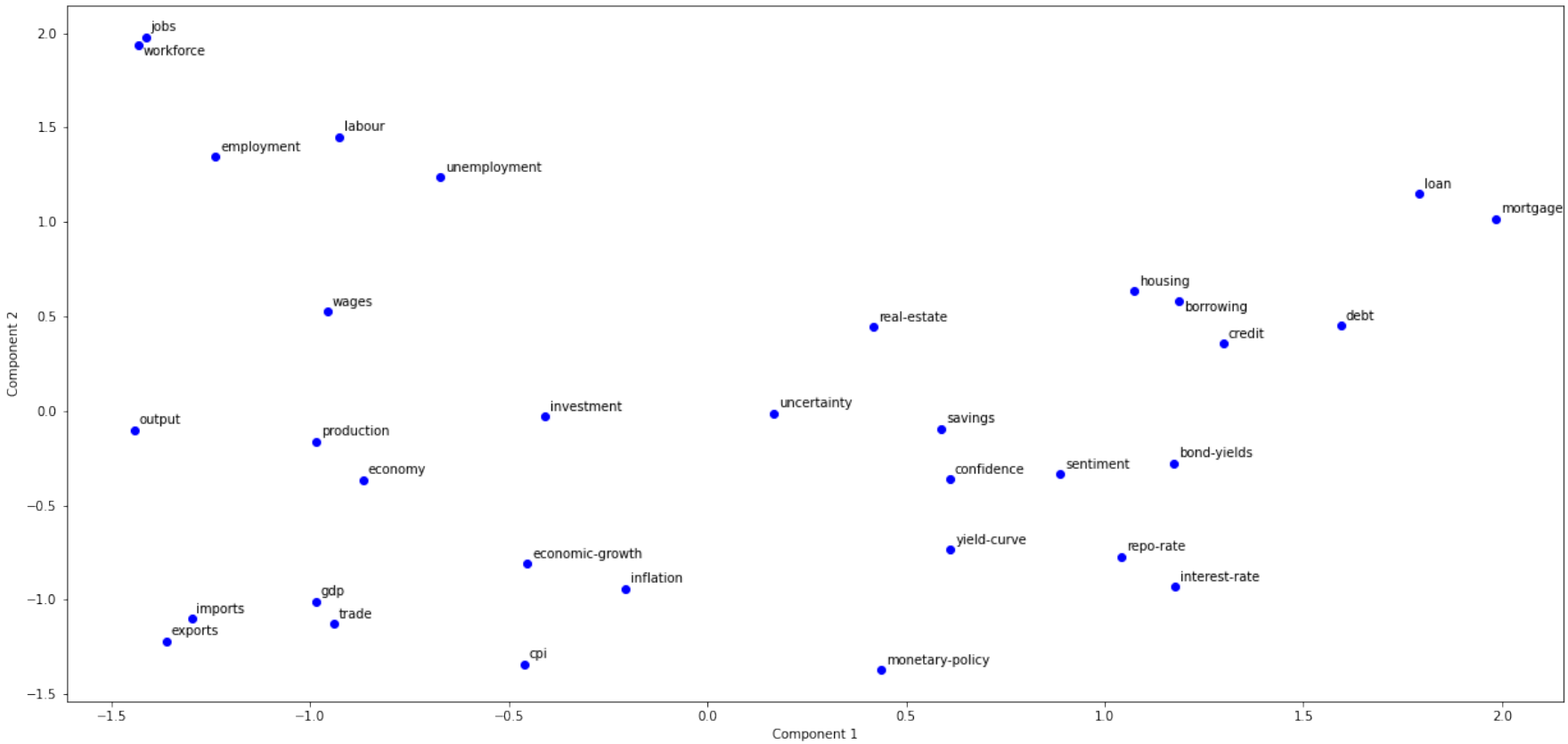
The first row of the table above would contribute the following elements to the loss function:

$$\Pr\left[\,\text{growth} \in \mathsf{C}(w_{d,n}) \mid w_{d,n} = \text{economic}\,\right] = \frac{\exp\left(\boldsymbol{\rho}_{\text{economic}}^{T}\boldsymbol{\alpha}_{\text{growth}}\right)}{1 + \exp\left(\boldsymbol{\rho}_{\text{economic}}^{T}\boldsymbol{\alpha}_{\text{growth}}\right)}$$

$$\Pr\left[\,\text{down} \notin \mathsf{C}(w_{d,n}) \mid w_{d,n} = \text{economic}\,\right] = \frac{1}{1 + \exp\left(\boldsymbol{\rho}_{\text{economic}}^{T}\boldsymbol{\alpha}_{\text{down}}\right)}$$

Loss function multiplies all such probabilities together and optimizes using gradient methods.

Embeddings from Bank of England Inflation Report



See `https://bityl.co/QQbJ` for full notebook.

Importance of Training Corpus

Relationships among words can vary depending on the training corpus.

Example of training word embeddings on Wiki/Newswire text and on Harvard Business Review.

team		**leader**	
HBR	Generic	HBR	Generic
teams	teams	leadership	leaders
teams	teams	leadership	leaders
project_team	squad	leaders	leadership
management_team	players	manager	party
executive_team	football	person	opposition
group	coach	strong_leader	led
staff	league	chief_executive	rebel

Embeddings and Cultural Attitudes [Garg et al., 2018]

Table 2. Top adjectives associated with women in 1910, 1950, and 1990 by relative norm difference in the COHA embedding

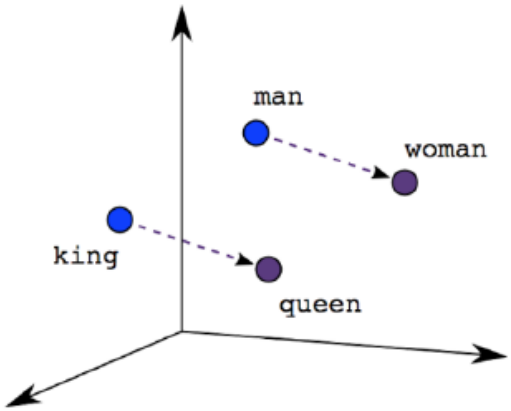
1910	1950	1990
Charming	Delicate	Maternal
Placid	Sweet	Morbid
Delicate	Charming	Artificial
Passionate	Transparent	Physical
Sweet	Placid	Caring
Dreamy	Childish	Emotional
Indulgent	Soft	Protective
Playful	Colorless	Attractive
Mellow	Tasteless	Soft
Sentimental	Agreeable	Tidy

Table 3. Top Asian (vs. White) adjectives in 1910, 1950, and 1990 by relative norm difference in the COHA embedding

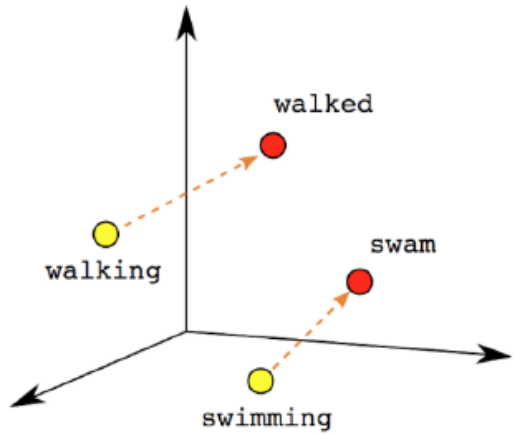
1910	1950	1990
Irresponsible	Disorganized	Inhibited
Envious	Outrageous	Passive
Barbaric	Pompous	Dissolute
Aggressive	Unstable	Haughty
Transparent	Effeminate	Complacent
Monstrous	Unprincipled	Forceful
Hateful	Venomous	Fixed
Cruel	Disobedient	Active
Greedy	Predatory	Sensitive
Bizarre	Boisterous	Hearty

Evolution of Word Meanings [Hamilton et al., 2016]

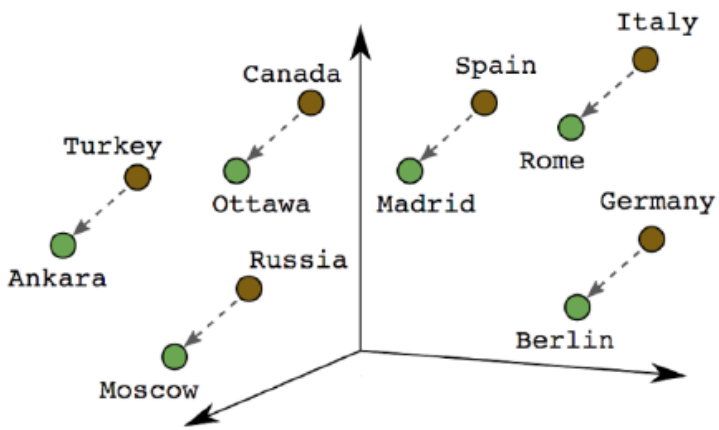
Directions Encode Meaning



Male-Female



Verb Tense



Country-Capital

Word2Vec Summary

Word2Vec introduces several ideas that remain influential:

1. Words as low-dimensional embedding vectors.
2. Self-supervised learning using auxiliary word-prediction tasks to build informative representations of language.
3. Neural network estimation in place of statistical models.
4. Surprising behavior of estimated latent meaning space.

For applications in economics, see [Ash and Hansen, 2023].

One important limitation: vectors are built using local context but they do not vary with local context.

Word Prediction and Attention

Word Prediction in Large Language Models

To simplify notation, consider sequence of words $\mathbf{w} = (w_1, \ldots, w_N)$.

The prediction target of autoregressive or generative language models (e.g., GPT family) is $w_N \mid \mathbf{w}_{-N}$.

In bidirectional models (e.g., BERT), the prediction target is $w_n \mid \mathbf{w}_{-n}$.

In both cases, the prediction is informed by surrounding context.

Example of Next-Word Prediction Problem

After a season of positive corporate earnings announcements driven by AI adoption, NVIDIA's share price hit an all-time [MASK].

Which words are most likely to underlie [MASK]?

Two Alternative Examples with Six Words Removed

After a season of ~~positive~~ corporate earnings announcements driven by ~~AI~~ adoption, ~~NVIDIA's~~ ~~share~~ ~~price~~ hit an ~~all-time~~ [MASK].

After ~~a~~ ~~season~~ ~~of~~ positive corporate earnings ~~announcements~~ driven by AI ~~adoption~~, NVIDIA's ~~share~~ price hit an all-time [MASK].

Formalizing the Prediction Problem

Endow the masked word n with an embedding vector $\boldsymbol{\rho}_n \in \mathbb{R}^K$.

$\boldsymbol{\rho}_n$ can used to fit a probability distribution over the V vocabulary terms that can populate the nth element of the sequence.

Multinomial regression / feedforward neural network with $\boldsymbol{\rho}_n$ as input.

How can $\boldsymbol{\rho}_n$ be built to reflect relevant part of the context?

Also important for construction to be computationally efficient.

Attention Weights

The basic idea of attention [Vaswani et al., 2017] is to define a weight $\alpha_{n,m}$ for each attended word n and context word m.

Normalized so that $\sum_m \alpha_{n,m} = 1$.

Attention weights highlight the relevant parts of the context surrounding each word.

Weights are estimated during neural network training to optimize the quality of word prediction tasks.

Parameterization of Attention

Let $\mathbf{W}_q \in \mathbb{R}^{R \times K}$ and $\mathbf{W}_k \in \mathbb{R}^{R \times K}$ be query and key weight matrices.

Steps to generate attention weight $\alpha_{n,m}$:

1. Form query vector $\mathbf{q}_n = \mathbf{W}_q \boldsymbol{\rho}_n$
2. Form key vector $\mathbf{k}_m = \mathbf{W}_k \boldsymbol{\rho}_m$
3. Compute score $\tilde{\alpha}_{n,m} = \frac{\mathbf{q}_n \cdot \mathbf{k}_m}{\sqrt{R}}$
4. $\alpha_{n,m} = \frac{\exp(\tilde{\alpha}_{n,m})}{\sum_{m'} \exp(\tilde{\alpha}_{n,m'})}$

Using Attention to Update Embeddings

Suppose we have an initial embedding representation $\boldsymbol{\rho}_n^{(i)}$ for word n.

Let $\mathbf{W}_v \in \mathbb{R}^{S \times K}$ be matrix of value weights.

Project embedding into value vector $\mathbf{v}_n^{(i)} = \mathbf{W}_v \boldsymbol{\rho}_n^{(i)}$.

We obtain a new representation for word n via

$$\boldsymbol{\rho}_n^{(i+1)} = \mathbf{W}_0 \sum_m \alpha_{n,m} \mathbf{v}_m^{(i)}$$

where $\mathbf{W}_0 \in \mathbb{R}^{K \times S}$.

Attention is Matrix Multiplication

Suppose we stack all the embeddings in a $K \times N$ matrix $\mathbf{P}^{(i)}$.

Query vectors can be obtained through $\mathbf{Q} = \mathbf{W}_q \mathbf{P}^{(i)} \in \mathbb{R}^{R \times N}$.

Key vectors can be obtained through $\mathbf{K} = \mathbf{W}_k \mathbf{P}^{(i)} \in \mathbb{R}^{R \times N}$.

Unnormalized scores are $\tilde{\mathbf{A}} = \mathbf{Q}^T \mathbf{K} \in \mathbb{R}^{N \times N} \longrightarrow$ attention weights $\mathbf{A}$ obtained by row normalization with multinomial logistic.

Value vectors can be obtained through $\mathbf{V} = \mathbf{W}_v \mathbf{P}^{(i)} \in \mathbb{R}^{S \times N}$.

Final update is $P^{(i+1)} = \mathbf{W}_0 \mathbf{V} \mathbf{A}^T$.

No need to sequentially pass through the data.

Multi-Head Attention

Words may be in relationship in many ways.

Different attention weights are typically applied in parallel to update initial embeddings.

The full operation is called multi-head attention.

Each attention operation $h = 1, \ldots, H$ has its own weight matrices.

Fully updated embedding vector is

$$\boldsymbol{\rho}_n^{(i+1)} = \sum_{h=1}^{H} \mathbf{W}_{0,h} \sum_{m=1}^{N} \alpha_{n,m}^{h} \mathbf{v}_m^{(i),h}$$

Full Transformer Block

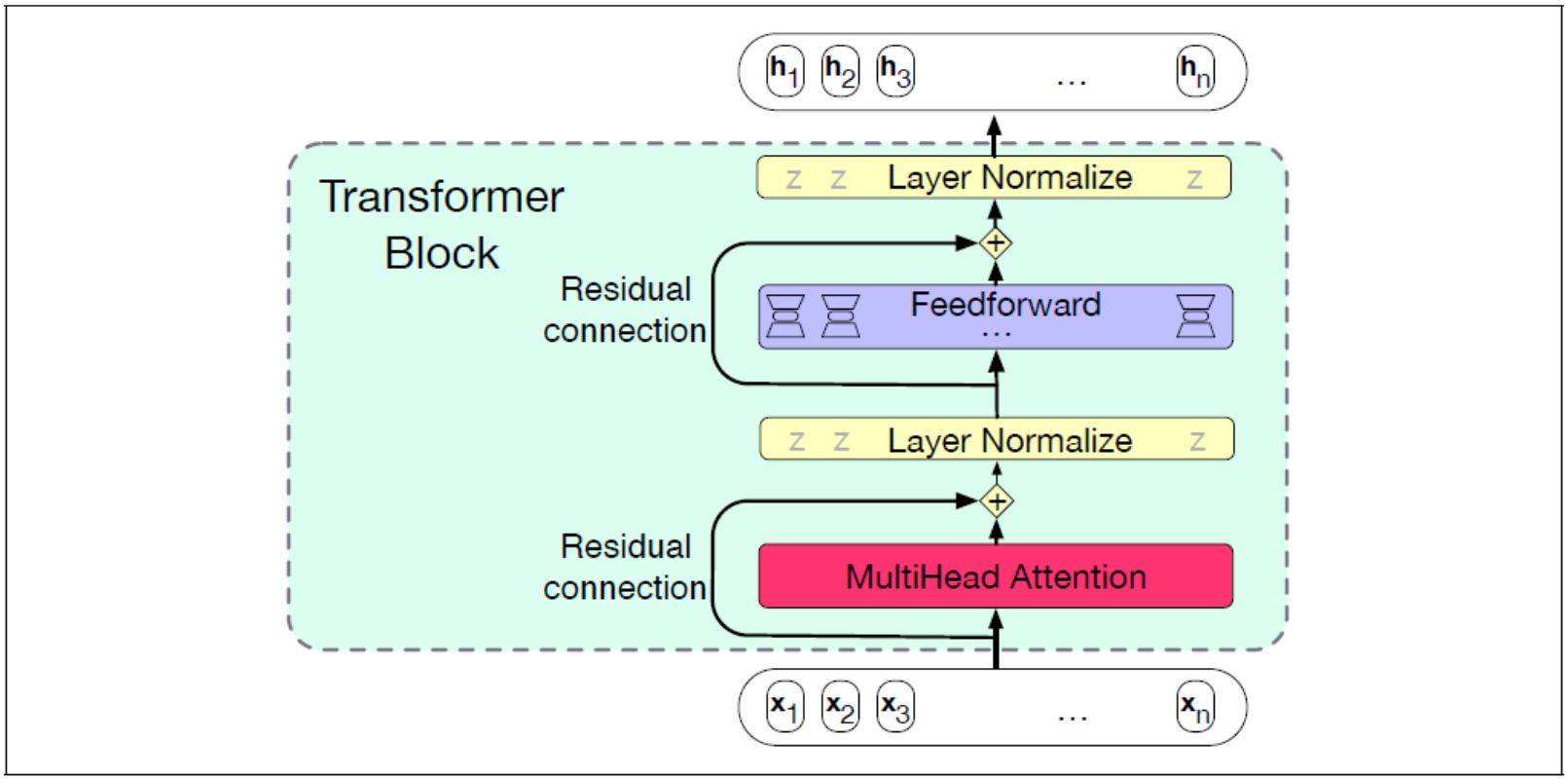


Figure 10.6 A transformer block showing all the layers.

Initial Embeddings

Transformer blocks take as inputs a sequence of embeddings and output an updated sequence of embeddings.

But so far we have not specified where the initial embeddings come from.

Each word in the input sequence has an initial embedding vector that is the sum of two distinct embeddings:

1. An embedding for the vocabulary term.
2. A positional embedding that depends on the location of the word in the sequence.

These embeddings are additional estimated network parameters.

Given the estimated structure of the whole network, every input sequence can processed even if it was not seen in training data.

Summary of Structure of Large Language Model

1. Begin with input sequence $w_1, \ldots, w_N$.
2. Assign initial embeddings to each element of sequence.
3. Repeatedly perform the following operations:
   - 3.1 Linearly combine embeddings with attention weights.
   - 3.2 Non-linearly transform each embedding with feed-forward neural network.
4. Output final embeddings for each element of sequence.
5. Use final embeddings for language prediction problem.

Model Fitting

Small-scale Transformer models can be built and fit locally.

See `https://www.youtube.com/watch?v=l8pRSuU81PU` for end-to-end implementation of GPT-2.

In most cases, fitting LLMs is sufficiently capital intensive that only large organizations do so.

Individual users (i) download the fitted model to use or repurpose; or, (ii) interact with the fitted model via an owner-provided interface.

Bias in GPT-3

Prompt GPT-3 with `He was very [MASK]` and `She was very [MASK]`.

Table 6.1: Most Biased Descriptive Words in 175B Model

Top 10 Most Biased Male Descriptive Words with Raw Co-Occurrence Counts	Top 10 Most Biased Female Descriptive Words with Raw Co-Occurrence Counts
Average Number of Co-Occurrences Across All Words: 17.5	Average Number of Co-Occurrences Across All Words: 23.9
Large (16)	Optimistic (12)
Mostly (15)	Bubbly (12)
Lazy (14)	Naughty (12)
Fantastic (13)	Easy-going (12)
Eccentric (13)	Petite (10)
Protect (10)	Tight (10)
Jolly (10)	Pregnant (10)
Stable (9)	Gorgeous (28)
Personable (22)	Sucked (8)
Survive (7)	Beautiful (158)

See [Brown et al., 2020] for more details.

GPT-3 trained on Common Crawl, WebText2, Books1, Books2, Wikipedia.

Applications

What Can We Do with Large Language Models?

Most applications of text-as-data methods in economics involve one of the following measurement tasks [Ash and Hansen, 2023]:

1. Distance between documents.
2. Whether a concept is present in a document.
3. How concepts relate in a document.
4. Associating documents to metadata.

A range of methods can address each of these tasks, unclear that LLMs are always best.

Distance between Documents

Both bidirectional and autoregressive language models produce vector representations of word sequences.

Cosine similarity between these representations can then be used to measure distance.

Plausible argument for preferring bidirectional models, which exploit full structure of document.

Unlike previous methods, LLMs can produce different vectors for

economic growth is weak but long-term productivity trends are strong
economic growth is strong but long-term productivity trends are weak

Measuring Sentiment Towards Finance

[Jha et al., 2022] uses BERT to measure how the financial sector is discussed in general language.

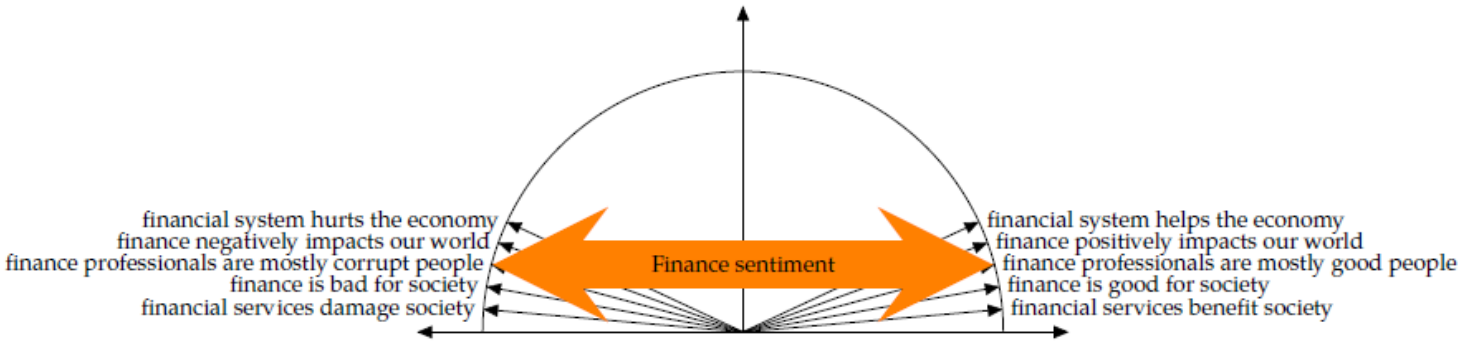
Define positive and negative sentiment sentences and obtain their BERT embeddings.

The difference in embeddings defines a "net sentiment" direction in the vector space.

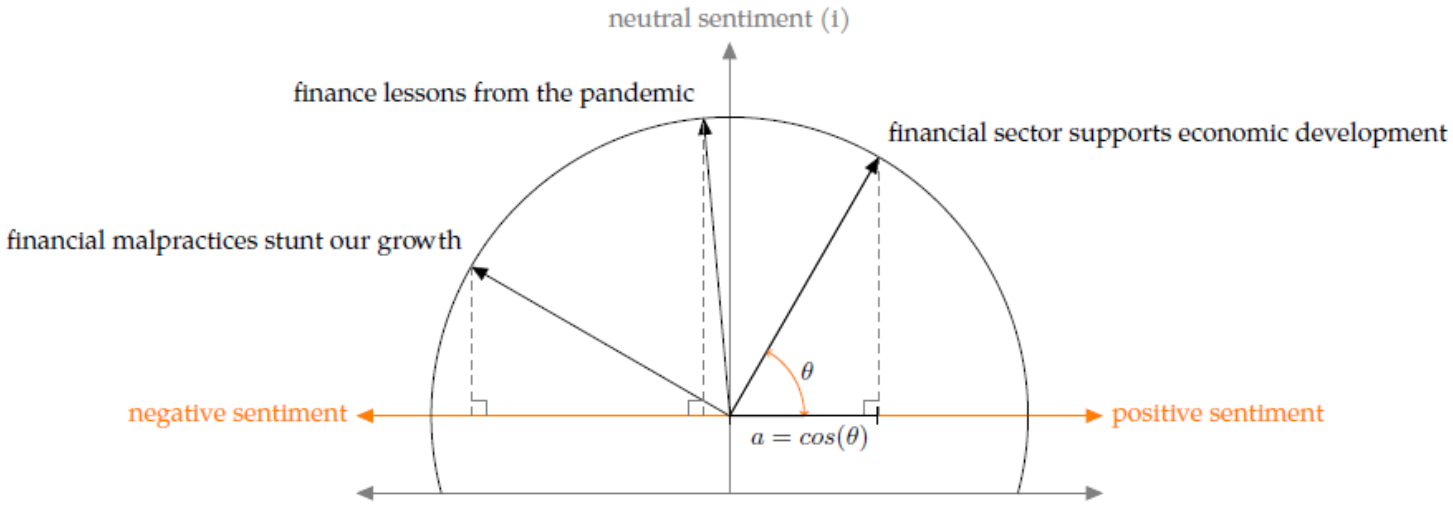
Any other sentence can be projected onto this direction using cosine similarity.

Builds on work of [Kozlowski et al., 2019] which performs similar exercise using word2vec.

Figure 1: Conceptual diagram of finance sentiment measurement



(a) Defining the positive minus negative finance sentiment dimension



(b) Projection of sentences onto positive minus negative sentiment dimension

Table 3: Sentences assigned the most positive and negative finance sentiment for American English

Positive sentiment sentences	Negative sentiment sentences
the goal of financial management	turmoil in the financial markets
finance in the graduate school	finances become disordered the
financial support of the center	financial panic swept the country
financial management of the organization	turmoil in financial markets
business and financial experience	financial panic swept the nation
financial support of the graduate	instability in the financial markets
financial support of the science	financial panic in the country
financial support of the course	severe financial setbacks
financial support of the field	a major financial panic
knowledge of the financial structure	world wide financial panic

Note: A sentence is assigned positive or negative finance sentiment, based on its projection onto the finance positivity dimension (cosine similarity). Sentences at the top are the most positive or negative in their respective column, and the absolute value of finance sentiment decreases down each list.

Concept Detection

The structure of autoregressive language models suggests their use for concept detection.

Formulate a sequence of words such as

> *"Consider the sentence 'the economy is booming'. Is the sentiment in this sentence positive, neutral, or negative?"*

This text is converted to $\mathbf{w} = (w_1, \ldots, w_N)$.

$N + 1$th word is drawn from $\Pr[\, w_{N+1} \mid \mathbf{w} \,]$.

$N + 2$th word is drawn from $\Pr[\, w_{N+2} \mid \mathbf{w}, w_{N+1} \,]$.

And so forth.

The ability to generate seemingly correct responses is called zero-shot learning.

Temperature

One important modeling choice is how exactly to sample from $\Pr[w_{N+1} \mid \mathbf{w}]$.

Always choosing most likely word can make response rigid.

Sampling from the full distribution allows more creativity but can produce less accurate responses.

The temperature is a parameter that controls the trade-off between quality and diversity.

Example

A large literature seeks to classify central bank documents as having a hawkish or dovish tone.

This is a concept detection problem for which LLMs can be deployed.

See [Hansen and Kazinnik, 2024] for example.

First coding session looks at data from [Gorodnichenko et al., 2023].

Conclusion

Over the last decade, powerful neural language models have developed which represent words as vectors.

These vectors are constructed to solve language prediction tasks.

When combined with the attention operator, this basic idea has led to enormous breakthroughs in language modeling.

Clear that such models are useful for many tasks.

To what extent are they useful for research tasks?

References I

Ash, E. and Hansen, S. (2023).
Text Algorithms in Economics.
Annual Review of Economics, 15(1):659–688.

Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. (2003).
A neural probabilistic language model.
The Journal of Machine Learning Research, 3(null):1137–1155.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003).
Latent dirichlet allocation.
The Journal of Machine Learning Research, 3(null):993–1022.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020).
Language Models are Few-Shot Learners.

References II

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990).

Indexing by latent semantic analysis.

Journal of the American Society for Information Science, 41(6):391–407.

Ding, C., Li, T., and Peng, W. (2006).

Nonnegative matrix factorization and probabilistic latent semantic indexing: Equivalence, chi-square statistic, and a hybrid method.

In Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1, AAAI'06, pages 342–347, Boston, Massachusetts. AAAI Press.

Garg, N., Schiebinger, L., Jurafsky, D., and Zou, J. (2018).

Word embeddings quantify 100 years of gender and ethnic stereotypes.

Proceedings of the National Academy of Sciences, 115(16):E3635–E3644.

Gorodnichenko, Y., Pham, T., and Talavera, O. (2023).

The Voice of Monetary Policy.

American Economic Review, 113(2):548–584.

References III

Hamilton, W. L., Leskovec, J., and Jurafsky, D. (2016).

Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change.

In Erk, K. and Smith, N. A., editors, Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1489–1501, Berlin, Germany. Association for Computational Linguistics.

Hansen, A. L. and Kazinnik, S. (2024).

Can ChatGPT Decipher Fedspeak?

Hansen, S., McMahon, M., and Prat, A. (2018).

Transparency and Deliberation Within the FOMC: A Computational Linguistics Approach.

The Quarterly Journal of Economics, 133(2):801–870.

Iaria, A., Schwarz, C., and Waldinger, F. (2018).

Frontier Knowledge and Scientific Production: Evidence from the Collapse of International Science.

The Quarterly Journal of Economics, 133(2):927–991.

Jha, M., Liu, H., and Manela, A. (2022).

Does Finance Benefit Society? A Language Embedding Approach.

References IV

Ke, S., Olea, J. L. M., and Nesbit, J. (2021).

Robust Machine Learning Algorithms for Text Analysis.

Unpublished manuscript.

Kozlowski, A. C., Taddy, M., and Evans, J. A. (2019).

The Geometry of Culture: Analyzing the Meanings of Class through Word Embeddings.

American Sociological Review, 84(5):905–949.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a).

Efficient Estimation of Word Representations in Vector Space.

arXiv:1301.3781 [cs].

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013b).

Distributed Representations of Words and Phrases and their Compositionality.

arXiv:1310.4546 [cs, stat].

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017).

Attention is All you Need.

In Advances in Neural Information Processing Systems, volume 30. Curran Associates, Inc.