DISCUSSION PAPER SERIES

DP19479

# LARGE LANGUAGE MODELS IN ECONOMICS

Elliott Ash, Stephen Hansen and Yabra Muvdi

ORGANIZATIONAL ECONOMICS AND POLITICAL ECONOMY

*ISSN 0265-8003*

# LARGE LANGUAGE MODELS IN ECONOMICS

*Elliott Ash, Stephen Hansen and Yabra Muvdi*

Discussion Paper DP19479
Published 13 September 2024
Submitted 09 September 2024

Centre for Economic Policy Research
33 Great Sutton Street, London EC1V 0DX, UK
Tel: +44 (0)20 7183 8801
www.cepr.org

# LARGE LANGUAGE MODELS IN ECONOMICS


## Abstract

This chapter explores the transformative impact of large language models (LLMs) on text analysis in economics. We trace the evolution from traditional methods like bag-of-words to advanced models such as BERT and GPT, highlighting how these models address limitations in understanding context and allowing higher-order reasoning. Although LLMs are complex, costly, and lacking in transparency, they are powerful tools for research, such as measuring sentiment or predicting metadata associated with documents.



JEL Classification: C18, C45, C55

Keywords: Large Language Models, Transformer models, Text as data, Unstructured Data



Elliott Ash - ashe@ethz.ch
*ETH Zurich and CEPR*

Stephen Hansen - stephen.hansen@ucl.ac.uk
*University College London and CEPR*

Yabra Muvdi - yabra.muvdi@gess.ethz.ch
*ETH Zurich*



Acknowledgements
E.A. and S.H. gratefully and respectively acknowledge financial support from ERC Starting Grant 101042554 and ERC Consolidator Grant 864863.

# Large Language Models in Economics*

Elliott Ash
ETH Zurich

Stephen Hansen
UCL, IFS, and CEPR

Yabra Muvdi
ETH Zurich

September 6, 2024

**Abstract**

This chapter explores the transformative impact of large language models (LLMs) on text analysis in economics. We trace the evolution from traditional methods like bag-of-words to advanced models such as BERT and GPT, highlighting how these models address limitations in understanding context and allowing higher-order reasoning. Although LLMs are complex, costly, and lacking in transparency, they are powerful tools for research, such as measuring sentiment or predicting metadata associated with documents.

**JEL Codes**: C18, C45, C55

**Keywords**: Large Language Models, Transformers, Text-as-Data, Unstructured Data

*E.A. and S.H. gratefully and respectively acknowledge financial support from ERC Starting Grant 101042554 and ERC Consolidator Grant 864863.

# 1 Introduction

The increasing availability of textual data and advancements in natural language processing (NLP) have transformed the field of economics, enabling researchers to extract valuable insights from vast amounts of unstructured data (Gentzkow et al. 2019a, Ash and Hansen 2023). Text-as-data analysis, which began with basic models like bag-of-words, has evolved significantly with the development of large language models (LLMs). These models, such as BERT and GPT, have opened up new possibilities for understanding and analyzing text, particularly by capturing context and nuances in language that earlier methods struggled to address.

Historically, the bag-of-words model and its extensions, like $n$-grams, dominated the text analysis landscape in economics. While these models have been effective in many applications, they are limited in what information they can represent in text given they rely on counting words and phrases rather than understanding longer-run connections within documents. The advent of LLMs, particularly those based on transformer architectures, has overcome many of these limitations by employing attention operations that allow models to weigh the importance of different words in a context-dependent manner (Vaswani et al. 2017). This innovation has significantly improved the ability of models to interpret and generate text, making LLMs a valuable tool for economic research.

In recent years, the application of LLMs in economics and related fields has expanded rapidly. Researchers have begun to fine-tune these models on domain-specific datasets, creating specialized versions like FinBERT (Yang et al. 2020) for financial text and ClimateBERT (Webersinke et al. 2021) for climate-related documents. These adaptations have demonstrated the potential of LLMs to enhance performance in specific fields, providing more accurate and context-aware analyses that are tailored to the unique needs of different areas. In particular, these fine-tuned models can predict metadata variables associated with documents, such as the salaries or remote-work options in job posts (Bana 2022, Hansen et al. 2023).

Most recently, the implementation of tools to help LLMs follow instructions and otherwise respond to human feedback has led to the emergence of a generation of powerful AI assistants such as ChatGPT. These models can perform many tasks previously thought to require human-level intelligence (Korinek 2023). The AI assistants that can interact with users in a meaningful and effective manner, offering new opportunities for applying NLP in economic policy analysis, market research, and beyond.

Despite the potential of LLMs, their adoption in economics is not without challenges. The complexity and scale of these models, coupled with the significant resources required to train them, have raised concerns about accessibility and transparency. Furthermore, the reliance on vast corpora of text, often sourced from the internet, introduces risks re-

lated to bias and the representativeness of the data. These issues underscore the need for ongoing research and collaboration between economists, computer scientists, and policymakers to ensure that LLMs are developed and deployed in ways that are both effective and ethically sound.

This chapter explores the evolution of text analysis in economics, from the early days of bag-of-words models to the current state-of-the-art LLMs. It examines the technical foundations of these models, their applications in economics, and the challenges associated with their use. By providing a comprehensive overview of LLMs in the context of economics, this chapter aims to inform future research and encourage the thoughtful application of these powerful tools in the field.

# 2 Text Analysis Before LLMs

Before describing the structure of LLMs (large language models) and their uses in economics, we provide some brief background on text-as-data analysis in economics prior to the development of LLMs. Interested readers can find more extensive details in Ash and Hansen (2023).

## 2.1 Basic notation

Algorithmic text analysis starts with a machine-readable collection of $D$ documents. In traditional text-as-data analysis, these documents are drawn from the domain of interest to a researcher. For example, a study of political speech would draw on a set of documents of political speeches. By contrast, large language models are often fit on generic text corpora which may or may not align with the interests of a researcher. In any case, the documents contain the knowledge that any model can encode. Likewise, any biases present in the document will be propagated into models fit on them.

Before one applies any algorithm, raw document text must be converted into sequences of linguistic features. We denote the content of document $d$ as an ordered list $\mathbf{w}_d = (w_{d,1}, \ldots, w_{d,t}, \ldots, w_{d,N_d})$. Informally, one can view the list as comprising the sequence of words in a document. More formally, each individual element is called a *token* in the NLP literature (in the remainder of the text we will use 'word' and 'token' interchangeably). Until recently, the standard *preprocessing* approach in economics to generate $\mathbf{w}_d$ has been to represent documents as lists of words, typically reduced to some root form, and to remove both rare and common words.[1] On the other hand, the preprocessing pipeline for LLMs is designed to neither add nor remove information—that is, to split plain texts

[1] Many sources discuss this pipeline. See Manning et al. (2008) for a textbook treatment and Gentzkow et al. (2019a) for a discussion in the context of the economics literature. Particular details vary across applications and models.

into tokens without changing the text. The idea is to preserve as much meaning from word order as possible in the original text.[2]

Let $V$ be the number of unique token features in a corpus. The set of all unique features is called the *vocabulary*. The vocabulary is encoded by assigning to each unique token an integer value in $1:V$.[3] Individual documents are then represented as sequences of integers based on this mapping.

> **Example**. Consider the two-document corpus 'today is hot' and 'today is cold'. The pre-processed forms of the documents are ['today', 'is', 'hot'] and ['today', 'is', 'cold'], respectively. $V = 4$ and we can define the index mapping ('today': 1, 'is': 2, 'hot': 3, 'cold': 4). The encoded documents are $\mathbf{w}_1 = (1, 2, 3)$ and $\mathbf{w}_2 = (1, 2, 4)$.

## 2.2 Bag-of-words model

Prior to the advent of LLMs, the *bag-of-words* model dominated text analysis. In this model, word order is ignored and each document is represented as a histogram count over vocabulary terms. More formally, the corpus is represented by the $D \times V$ *document-term matrix* $X$ where $x_{d,v}$ is the count of word $v$ in document $d$. $X$ is the input data into many well-known algorithms.

One important advantage of the bag-of-words model is simplicity and transparency. But it has several specific problems, which can be illustrated by sentence pairs like the following:

1. **Synonomy**: two words having the same meaning.

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

   These sentences share the same meaning but have different bag-of-words representations, due to the synonyms 'weak' and 'tepid'.

2. **Polysemy**: one word having multiple meanings.

   *economic statistics lie about current well-being.*
   *the happy cats lie on the bed.*

   The word 'lie' is assigned the same bag-of-words vocabulary index in both of these sentences, yet the contextual meaning of the term is totally different.

[2]To reduce the number of characters, capitalization is represented by a special prefix token before a lowercase letter. Words are broken into separate pieces (e.g. 'walking' becomes 'walk' and 'ing'), to help LLMs learn more meaningful word representations, especially for rare and long words. This is called byte-pair encoding (e.g. Goldberg 2017).

[3]The choice of which terms are assigned to which indices is arbitrary, but it is often convenient to sort by corpus frequency.

3. **Word Order**.

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

   These sentences have the same bag-of-words representation but have distinct meanings due to the different ordering of 'weak' and 'strong'.

## 2.3 Addressing limitations

Some well-known strategies already exist for overcoming the problems with the basic bag-of-words model. The synonymy problem has a familiar analogue in factor models: a high-dimensional set of variables (the individual term counts) are highly correlated across a set of observations (the individual documents). The *topic modeling* literature in NLP applies the ideas of factor modeling to the document-term matrix to learn synonyms and allow them to load on shared factors. Latent semantic analysis (LSA, Deerwester et al. 1990) uses principal components analysis; non-negative matrix factorization (NMF, Ding et al. 2006) decomposes $X$ into two non-negative matrices; and latent Dirichlet allocation (LDA, Blei et al. 2003) extends NMF to a Bayesian setting. All of these approaches reduce the dimensionality of the document-term matrix by representing the vocabulary with a factor structure in which individual factors (or *topics* in the NLP jargon) group together words that frequently co-occur. In this way, the models learn that different words can represent the same underlying concept. Such models have been applied in several instances in the economics literature.[4]

While topic models can reveal that multiple words have similar meanings, they still operate on the document-term matrix and so cannot resolve polysemy and word order since these problems depend fundamentally on context. One idea to address both issues is to extend the set of tabulated features to include not just individual words but also word phrases. The $n$-gram model tabulates the counts of all length-$n$ phrases across documents. Even adding $n = 2$ or $n = 3$ can make a big difference. In the examples above for polysemy, we obtain the disambiguating phrases 'statistics lie' and 'cats lie'. For word order, correspondingly, we obtain 'growth is weak', 'trends are strong', 'growth is strong', and 'trends are weak'. In Gentzkow et al. (2019b), using bigrams and trigrams is pivotal to identifying polarized partisan language in U.S. Congress.

While the $n$-gram model helpfully accounts for local relationships among words, it exponentially expands the feature space $V$. For this reason, the $n$-gram model is typically limited to bigrams ($n = 2$) or trigrams ($n = 3$). These variants do not allow for relationships among words beyond these narrow windows. And they are not robust to

[4]Iaria et al. (2018) applies LSA to scientific articles, while Hansen et al. (2018) and Ke et al. (2021) use LDA and NMF, respectively, to analyze FOMC deliberations. Ash et al. (2024c) apply LDA to analyze diffusion of legislative text across U.S. states.

small differences in word ordering – for example, 'weak growth' gets a different $n$-gram representation from 'growth is weak'. An extension that helps on this margin is to add in grammatical information, using a syntactic dependency parser (Jurafsky and Martin 2024). That allows for subjects, verbs, objects, and adjectives to be tagged and connected up, capturing information on 'who (subject) does what (verb) to whom (object)?' and 'which entity (subject) has which attribute (adjective)'? With this information, relations like ('growth', 'weak') are efficiently encoded. Ash et al. (2024b) show that grammatical connections encode additional information on partisan ideology than simple n-grams. Arold et al. (2024) use grammar to extract worker-rights clauses from collective bargaining agreements and show that these clauses behave like a non-wage amenity.

Even with local word order and grammar, what can be measured with these pre-LLM methods is extremely limited. While grammar adds information on word connections within sentences, it does not look at word connections across sentences. And it cannot look at how sentences are connected, even in a basic way. It does not even begin to approach the conceptual connections within and across documents that we may be interested in. In the rest of the chapter, we provide an overview of how large language models overcome these limitations to incorporate context.

# 3 Language Models and Word Prediction

In this section, we describe the basic building blocks of large language models (LLMs). The common thread is to represent words as vectors formed to successfully complete language prediction tasks.

## 3.1 Word2Vec

A fundamental idea in language modeling is a *word embedding* which is a vector representation of a word. More formally, let $\boldsymbol{\rho}_v \in \mathbb{R}^K$ be a vector representation of the $v$th vocabulary term. Typically, $K \ll V$ so that word embeddings are low dimensional relative to the size of the vocabulary. Loosely speaking, the goal in constructing embeddings is to have the (cosine) similarity between $\boldsymbol{\rho}_v$ and $\boldsymbol{\rho}_{v'}$ grow whenever terms $v$ and $v'$ have more similar meanings, for all pairs $v$ and $v'$.

There are many algorithms for constructing word embeddings, but here we focus on Word2Vec (Mikolov et al. 2013b,a) which is one of the first instances of a scalable, neural-network-based model. The motivating idea behind Word2Vec is the *distributional hypothesis*, a concept from linguistics maintaining that words share similar meaning when they share similar surrounding words. To this end, let the *context* of word $w_{d,t}$ at index

$t$ in document $d$ be a length-$2L$ window of words around $w_{d,n}$:

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

Word2Vec fits word embeddings that solve language prediction tasks relating words to their contexts. In the popular *skip-gram* variant, the prediction task is to guess, given a target word $w_{d,t}$, whether a comparison word $w'$ actually comes from $w$'s context ($w' \in \mathrm{C}(w_{d,t})$), or whether $w'$ is randomly sampled from outside the context.

> **Example**. Consider the sentence 'economic growth is weak but long-term productivity trends are strong'. The prediction problems associated with this document with $L = 2$ is a list of pairs of 'positive examples' and 'negative examples' with the same target word, where the positive example is associated to a true context word appearing within 2 places of the target, and the negative example is associated to a randomly sampled comparison word more than 2 places away.

| **Positive Examples** | | **Negative Examples** | |
|---|---|---|---|
| Target | Comparison | Target | Comparison |
| 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 first prediction problem for positive examples is to predict 'growth' is in the context of 'economic' and so forth. The context words drawn for negative examples are illustrative random draws from a broader corpus. Here there is one negative example per positive example, but the number of negative examples is a modeling choice.

The parametrization of the prediction problem is as follows. Endow each word $v$ in the vocabulary with an embedding vector $\boldsymbol{\rho}_v \in \mathbb{R}^K$ and a context vector $\boldsymbol{\gamma}_v \in \mathbb{R}^K$. The

positive examples are modeled as

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

and the negative examples are modeled as

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

The overall loss function is formed by multiplying together these probabilities for all positive and all negative examples in the corpus.

Solving these word prediction tasks via word embeddings is not an end in itself. Instead, the prediction problems are a means to the end of obtaining word embeddings with the desired property of vector similarity proxying meaning similarity. Suppose two words $v$ and $v'$ always appear in the same context. The Word2Vec loss function will then tend to produce similar vectors. If $\boldsymbol{\rho}_v$ does a good job at predicting words in the context of occurrences of $v$ then $\boldsymbol{\rho}_{v'} \approx \boldsymbol{\rho}_v$ necessarily does a good job at predicting words in the context of occurrences of $v'$.

In keeping with the idea that the prediction problems are not of innate interest, the quality of word embeddings is judged by their performance in downstream language tasks. Word2Vec became popular after its publication because it excelled at these. For example, the nearest neighbors of particular words in the vector space overlap substantially with synonyms. Word2Vec embeddings also display a surprising ability to resolve analogies. Ash et al. (2024a) leverage this quality of word embeddings to analyze gender attitudes among U.S. appellate judges – that is, how they 'analogize' the male-female language dimension to the career-family language dimension. Judges who use more traditional gender stereotypes tend to vote against women's rights and to reverse the decisions of their female colleagues.

In spite of this success, Word2Vec still has some of the limitations of the bag-of-words model discussed above. Polysemy is still a problem, in the sense that every instance of a word in a corpus has the same embedded representation. Word order is addressed in a limited way during the construction of Word2Vec embeddings, as it has information on whether words tend to share the same neighboring words. But that does not help much for downstream tasks. For example, a common way to represent a sequence of words using an embedding model is to average the embeddings of all the words in the sequence (Arora et al. 2017, Gennaro and Ash 2022). By this logic, the two sentences in the 'Word Order' example in Section 2 would have the same average embeddings even though they

have distinct meanings. The key recent breakthrough in language models is to allow word embedding vectors to vary dynamically with the surrounding context.

## 3.2 Attention

The *attention* operation (Vaswani et al. 2017) lies at the heart of modern large language models. For simplicity, consider a single sequence of words $\mathbf{w} = (w_1, \ldots, w_N)$.[5] The word prediction problem of interest is now $\Pr[\, w_t \mid \mathbf{w}_{-t} \,]$, where $\mathbf{w}_{-t}$ denotes the sequence of words formed by removing the $t$th word. A special case of particular interest is $t = N$ where the problem becomes to predict the next element in a sequence without any information on what comes after it. This is the task of an *autoregressive* language model. A *bidirectional* model instead solves the problem when $1 < t < N$, making use of information both before and after the prediction target.

> **Example**. Consider the sentence 'After a season of positive corporate earnings announcements driven by AI adoption, NVIDIA's share price hit an all-time [MASK].' where [MASK] stands in for the prediction target. Most economists would predict a high probability of the word 'high' underlying [MASK]. The structure of English and the context is highly informative about the next word in the sequence.
>
> Now consider the following two modifications, each of which removes six distinct words from the sequence:
>
> 1. After ~~a season of~~ positive corporate earnings ~~announcements~~ driven by AI ~~adoption~~, NVIDIA's ~~share~~ price hit an all-time [MASK].
> 2. After a season of ~~positive~~ corporate earnings announcements driven by ~~AI~~ adoption, ~~NVIDIA's share price~~ hit an ~~all-time~~ [MASK].
>
> Most economists would continue to place high probability on 'high' with the more limited information in the first case, but in the second case the prediction is much more ambiguous. This illustrates how words in the context have quite different relevance for the prediction problem.

Building on this example, one can heuristically lay out the structure of the attention operation. Each token in the sentence is given an embedding vector, including the final token '[MASK]'. The [MASK] embedding is the input into a multinomial classifier—usually a feed-forward neural network—that gives a probability distribution over all tokens in the

[5] Here we drop the document index for notational simplicity. In practice, the input to a large language model will be a whole set of documents, not just a single sequence.

vocabulary. The key question is: how is the [MASK] embedding formed and how does it draw on the relevant context?

The basic idea of attention is to define a weight $\alpha_{w_t,w_u}$ for each *attended* word $w_t$ at index $t$ and *context* word $w_u$ at index $u$. The weights are normalized so that $\sum_u \alpha_{t,u} = 1$. Attention weights work to 'highlight' the relevant parts of the context surrounding each token. They are estimated during neural network training to maximize performance in prediction tasks, just like Word2Vec. In the case of the example above, when [MASK] is the attended token, one would expect the attention weights for 'AI' and 'NVIDIA' to be higher than for 'a' and 'of', since these named-entity words are more informative than stopwords for the prediction problem of forecasting the underlying masked word. Formally speaking, the embedding vector for '[MASK]' is updated by linearly combining all embeddings in the sequence using the attention weights as the weights. Thus, unlike in Word2Vec, individual word embeddings depend on surrounding tokens and are sensitive to context.

## 3.3 Transformer-based LLMs

The recent generation of popular LLMs, like BERT, GPT, Gemini, Claude, and LlaMa, are transformer-based large language models. As described above, the prediction problem is to learn $\Pr[\, w_t \mid \mathbf{w}_{-t}\,]$. The innovation is the transformer, a neural-net architecture that efficiently assembles attention operators to process massive corpora to build powerful language prediction systems. Particular models differ in their details, but the key building blocks are as follows.

The starting point is a massive training corpus. In the BERT (Bidirectional Encoder Representations from Transformers) model (Devlin et al. 2019), one of the first to demonstrate the power of the Transformer architecture, the main training corpus is the English-language Wikipedia (about 2.5 billion words) and 11,000 unpublished books (about 800 million words). While a 3.3-billion-word corpus was rightly considered massive in 2018, it is tiny compared to the corpora used in more recent LLMs, trained on tens of trillions of words.

As mentioned above, the corpus is then tokenized into word pieces using a model-specific tokenizer. Following this, the corpus is segmented into sequences up to a maximum token length $N$, given by the length of the context window for the attention operator. With BERT, for example, the maximum context length is 512 tokens (about 384 words). In the most recent generation of LLMS, a max context window length of 128,000 tokens is standard. Importantly, in these sequences some words are masked and replaced with a [MASK] token. In BERT 15% of tokens are randomly masked while in GPT models masked words only appear at the ends of token sequences. The goal of the model is to

correctly 'fill in' these masked tokens.

Each word (or word piece) in the vocabulary is assigned an initial $K$-dimensional word embedding vector $h_t^0$. This includes the [MASK] tokens, each of which is assigned an embeddings. Initial embeddings depend on (i) the vocabulary term $w_t$ and (ii) the position of the term in the sequence $t$. In this way, not all instances of a term are given the same initial vector representation. Instead, sequence information is retained even in initialization. $K$ varies depending on the model. In BERT, for example, $K = 768$. In the more recent generations of larger LLMs, $K$ can exceed 10,000.

The transformer architecture consists of a stack of $L$ transformer blocks. The central component of the transformer block is the attention operator described above. In the first block, the initial word embeddings are updated using attention weights. More specifically, the embedding at position $t$ becomes

$$\tilde{h}_t^1 = \sum_u \alpha_{t,u} h_t^0 \tag{1}$$

where $\alpha_{t,u}$ is the attention weight when $t$ is the attended word and $u$ is the context word.[6] The attention weights are learnable parameters in the network. Each embedding $\tilde{h}_t^1$ is non-linearly transformed and normalized to obtain $h_t^1$, the output of the first transformer block at position $t$. This process is then repeated $L$ times after which each token has a final embedded representation $h_t^L$. $L$ is again a design choice with BERT, for example, having twelve blocks.

Finally, the embeddings for each [MASK] token are used to predict the underlying word, and the loss function of the network penalizes word predictions that are incorrect. There are an enormous number of parameters in a large language model, but all are adjusted to promote successful word prediction. There are embeddings for the words and the token positions. There are separate attention weights and feed-forward layers in each block. Finally, there are the parameters of the masked word prediction problems at the end. This adds up to about 100 million learnable parameters in the basic BERT model. With more recent LLMS, 10 billion or even 100 billion parameters are the norm.

As with Word2Vec, the completion of the language prediction tasks is a means to an end. The main value of the model is that the final embeddings provide an informative and context-aware representation of language that can be used for downstream tasks. BERT's ability to perform these vastly exceeded previous models' and led directly to the current interest in LLMs more generally.

Transformers work well because they allow the meaning of words to interact with all other words in the context window. The many stacked layers, with non-linearities and

[6] In practice, these operations are performed multiple times in parallel which is called multi-head attention.

interactions, allow for higher-order connections to be learned from the data. It has been documented that the early transformer blocks learn basic language features, such as verb tense and parts of speech, the middle blocks learn syntactic connections between words, and the later blocks begin to learn more conceptual information and semantics (Rogers et al. 2021). While recurrent neural networks can also read through all words in context, they are too slow to train because they read through documents sequentially and cannot be parallelized.

In parallel to BERT, OpenAI developed a sequence of LLMs in the GPT (Generative Pre-trained Transformer) family (Brown et al. 2020). The key difference between BERT and GPT is that GPT models are autoregressive, seeking to predict the next element of a sequence without using information on elements that follow. This training objective makes these models effective in generating new text to conclude a given input sequence; this is the source of the name 'generative model' in the context of LLMs. At a high level, text generation works by sampling one word at a time conditional on the previous words in the sequence. Once a word is sampled, it then becomes the last element of the input sequence for sampling the next word, and so on. Of course, generative models do not produce their output by magic but simply reproduce likely next words in their training data. Any biases in training data will be present in these predictions.

More generally, word prediction tasks fit on generic English corpora like Wikipedia or Common Crawl do not clearly reflect the priorities or needs of the economics profession (or other communities). For this reason, while word prediction tasks are fundamental to the development of LLMS, they are not sufficient in themselves to build a high-value model. For that, the models must be further adapted to particular contexts or user needs. In particular, this earlier generation of language models often fail to follow instructions and don't work well as AI assistants. Taking that last step required alignment with human preferences.

# 4 Adapting and Aligning Large Language Models

While BERT and GPT-3 already transformed work in NLP, it wasn't until ChatGPT that LLMs went mainstream and transformed society. ChatGPT (GPT-3.5, GPT-4, GPT-4o) and other popular AI assistants (e.g. Claude, Gemini) are quite different from and more powerful than the earlier generation. These improvements were gained through fine-tuning LLMs on specialized datasets and then further aligning them with human preference data.

## 4.1 Adapting Language Models

In the previous section, we discussed non-aligned LLMs, also called pre-trained models or foundation models. Those models are trained on the self-supervised word prediction task and can be understood as compressed representations of the training corpus. While they have general knowledge and applicability, these pre-trained models can often be improved for specific tasks or domains by updating its parameters based on a new dataset. This process improves the model's performance on the target task without needing to train a model from scratch, utilizing the general knowledge encoded during pre-training.

The basic approach in LLM adaptation is to take a pre-trained model and do gradient updates on additional data. There are two main kinds of adaptation: self-supervision on additional text and adaptation to predict metadata. If the fine-tuning data is additional text – for example, text from economics articles – the training process is the same as with the base LLM, just using the new specialized data. If the adaptation is for metadata, the approach is to replace the transformer's last softmax layer for word prediction with a new output layer that fits whatever outcome data one is trying to predict. That could be a classifier across categories (e.g. positive or negative sentiment) or a linear regressor with a continuous outcome. We will discuss each of these tasks in more detail below.

For both types of fine-tuning, there are some techniques that make it more efficient. The idea is that as the task becomes more specialized, the model does not need as much capacity and can be compressed somewhat. In particular:

- **Freezing layers**. One common technique is to freeze certain layers of the model, meaning their parameters remain unchanged during training (Howard and Ruder 2018). Typically, the lower layers of a language model, which capture more general linguistic features, are frozen. This approach leverages the pre-trained knowledge in these layers and focuses the training on the higher layers, which capture more task-specific features. By freezing lower layers, computational resources are saved, and the risk of overfitting is reduced.

- **Adapters.** Adapters are small neural networks inserted between the layers of a pre-trained model; instead of updating the entire model, only the adapter layers are trained (Houlsby et al. 2019). This method significantly reduces the number of parameters that need to be updated, making the fine-tuning process more efficient. Adapters can be designed to capture specific task-related information, allowing the main model to retain its general language understanding capabilities. The use of adapters is particularly advantageous when dealing with multiple tasks, as different adapters can be swapped in and out for different tasks without retraining the whole model.

- **Low-Rank Adaptation (LoRA).** Low-Rank Adaptation (LoRA) is a technique that modifies the weight matrices of a pre-trained model by decomposing them into lower-rank matrices during fine-tuning. Instead of updating the full-weight matrix, LoRA updates only the low-rank components (Hu et al. 2021). This approach drastically reduces the number of parameters to be fine-tuned, leading to faster training times and reduced memory usage. LoRA maintains the pre-trained model's expressive power while efficiently adapting it to new tasks. The lower-rank matrices capture task-specific variations, enabling effective fine-tuning with minimal computational overhead.

## 4.2 Self-supervision on additional text

Fine-tuning with additional self-supervised text is often used to adapt pre-trained language models to specific tasks or domains. This process involves taking a pre-trained model, such as BERT or GPT, and training it further on additional, domain-specific text data to enhance its performance for particular applications. An example relevant for economists is FinBERT (Finance BERT), which is fine-tuned using finance-related text for finance-related linguistic and prediction tasks (Yang et al. 2020).

Next, a domain-specific dataset is needed, containing text from the target domain. FinBERT is fine-tuned on 4.9 billion tokens from corporate filings, earnings call transcripts, and analyst reports. That corpus should be segmented and tokenized using the same vocabulary and other specifications as the pre-trained model.

Using the processed domain-specific dataset, the pre-trained model is then further trained, a process called fine-tuning. The prediction task is the same as the original model: for BERT, it would be masked token prediction using a bidirectional model; for GPT, it would be autoregressive next-token prediction. During fine-tuning, the model weights are adjusted with gradient updates the same way as during pre-training, potentially with the aforementioned adaptation variants such as frozen layers, adapters, or LoRa.

During and after fine-tuning, the new model is evaluated on a validation set to ensure performance has improved in the target domain. For starters, the specialized model should be able to do token prediction better in the specific domain than the generic pre-trained model. If necessary, further adjustments and fine-tuning iterations are performed.

**Field-specific models.** Specialized models have demonstrated the effectiveness of fine-tuning pre-trained models on domain-specific text. Besides FinBERT, other fine-tuned models include:

- SciBERT (Beltagy et al. 2019): Fine-tuned on a large corpus of scientific literature to enhance performance on scientific text mining tasks.

- ClimateBERT (Webersinke et al. 2021): Adapted to climate-related documents, improving its capability to handle climate-specific terminologies and concepts.
- LegalBERT (Chalkidis et al. 2020): Fine-tuned using legal texts, enabling it to perform better in legal NLP tasks such as legal document classification and contract analysis.

These models show significant improvements in their respective fields by leveraging the contextual understanding gained from the specialized datasets.

**Instruction Fine-Tuning.** A special kind of fine-tuning involves training the model on datasets designed for instruction following. With this kind of model, the fine-tuning corpus consists of lists of questions with answers, instructions with responses, problems with solutions, and coherent conversation threads. These documents should be exemplary of outputs that one would like to see in an AI-powered assistant. For example, Ouyang et al. (2022) fine-tune GPT-3 with about 13,000 instruction-response pairs. This process fine-tunes the model to better understand and follow human instructions, making it more useful for applications that require precise adherence to user prompts. It is a crucial first step for building useful AI assistants in the vein of ChatGPT.

## 4.3 Adaptation to predict metadata

Transformer-based LLMs are not just useful for the task of token prediction. Because they gain deep, contextualized understandings of documents, they can also make broader judgments about those documents. In particular, they can form predictions about metadata associated with those documents. That can include dimensions of the text itself, such as positive or negative tone (Shapiro et al. 2020), or whether a job post is for remote work (Hansen et al. 2023). The metadata could also include features outside the document that a researcher is interested in, such as the party affiliation of a politician speaker (Gentzkow et al. 2019b), or the wages associated with a job post (Bana 2022). As these examples demonstrate, predicting metadata is critical in many natural language processing (NLP) tasks. Fine-tuning pre-trained language models can significantly enhance the LLM's performance on these tasks.

As before, fine-tuning involves adjusting a pre-trained language model on a specific dataset related to the task at hand. With metadata, fine-tuning typically involves adding a classification layer on top of the last hidden layer. For example, in BERT, there is a special token at the beginning of the input, which is often used as the input embedding to the classifier. in GPT, the embedding for the final token can be used. Alternatively, the average of the embeddings for all the tokens in the document can be pooled and input

to the output layer. The model is then trained on labeled data, where the final layer's output corresponds to the metadata category or value.

Several domain-specific adaptations of BERT highlight its utility. ClimateBERT has been fine-tuned to detect climate-related paragraphs and to evaluate their sentiment and specificity (Webersinke et al. 2021). Similarly, LegalBERT has been fine-tuned to classify topics in legal data (Chalkidis et al. 2020). In economics, Bana (2022) fine-tunes a BERT model to predict salaries from the text of job postings and performs counterfactual exercises on salary outcomes by varying the language input. Hansen et al. (2023) compare several supervised learning models for predicting human labels for remote work, and find that BERT-like models achieve outstanding performance.

A noteworthy form of LLM-based metadata prediction involves predicting human annotations about the quality or usefulness of documents. This is particularly useful in the process of developing AI assistants. The quality prediction models can be used to build reward models, which are critical for reinforcement learning applications. Fine-tuning for this purpose involves training the model on a dataset of documents labeled by human annotators, with the goal of predicting these labels accurately (Ouyang et al. 2022).

## 4.4 Aligning Models with Human Preferences

The new generation of powerful AI assistants in the vein of ChatGPT, including Claude, Gemini, LlaMa, and Mistral, draw on all of the ingredients described so far. The starting point is a pre-trained LLM trained on a massive corpus – including basically all of the text ever digitized, in the range of tens of trillions of tokens. For these large models, it turns out that autoregressive models in the style of GPT (rather than bidirectional encoder models in the style of BERT) are much more effective for AI.

The first alignment step is instruction fine-tuning. We fine-tune the LLM using a dataset of instructions and responses, such as questions and answers or directives and reactions. The raw language model is fine-tuned to follow instructions, enhancing its task-specific performance Ouyang et al. (2022). However, instruction fine-tuning has limitations, and it is not enough to produce a powerful AI assistant. The issues include the high cost of collecting task-specific data and the difficulty in representing certain tasks, like story generation, through definitive instructions. Additionally, language models penalize all mistakes equally at the token level, which can be problematic when some errors are more critical than others.

To further align language models with human preferences, a subsequent process of reinforcement learning with both positive and negative feedback is needed. Data on quality dimensions – such as helpfulness and harmlessnes – are collected through large-

scale annotator surveys. For example, in Ouyang et al. (2022), about 50,000 human judgments on response quality were collected. A supervised-learning model, like a fine-tuned BERT or GPT-type model, is trained to predict these dimensions from the text. For example, in Stiennon et al. (2020), a GPT-3 model is fine-tuned to predict summary quality. This process produces a reward function $R(s)$, which assigns a score to a given sequence $s$.

The response labels can be subjective scores of individual outputs. Or they can be pairwise comparisons between two different outputs to the same prompt. For the latter, a matching approach is used to reduce the impact of confounding variables, where two similar answers to the same query are ranked by humans. A Bradley-Terry model is then fitted to predict the probability that a new document would be preferred over a median alternative. From an econometric perspective, the matching approach is preferred because the rating differences are driven just by the responses rather than by the questions.

The reward model is then used to adjust the instruction-fine-tuned model with reinforcement learning (RL), a process known as RLHF (reinforcement learning with human feedback). The intuitive goal of this alignment process is to teach a language model to generate words that increase expected human rewards, rather than just to guess the next token from the training corpus. Policy gradient methods, a class of RL algorithms, adjust model parameters to maximize an expected reward. For instance, given a reward function $R(\hat{s})$, the goal is to adjust the model parameters $\theta$ to maximize:

$$\theta^* = \arg\max_{\theta} \mathbb{E}_{\hat{s}\sim p_\theta(s)} R(\hat{s})$$

Gradient ascent methods are then used to update the model parameters iteratively.

This alignment methodology has resulted in powerful AI assistant systems like ChatGPT. Those models can perform many tasks requiring human-level intelligence, including many tasks that economists often perform in their work (Korinek 2023). For example, Chopra and Haaland (2023) use AI agents to conduct qualitative interviews about financial literacy.

In the realm of text analysis, LLMs have particular usefulness in annotating documents. For example, an AI assistant could be prompted with questions like "Does this sentence have a happy or sad tone?" or "Does this job post mention remote work flexibility?" Performance on questions like these with AI assistants is very good. But there are new issues that arise, because LLMs can be sensitive to prompting. For example, allowing for an "other" category, or asking for an explanation, can change the model's outputs. Researchers should be aware of this and experiment with different prompts. In particular, they should break down complex questions into multiple simpler questions. This is an active area of research.

# 5 Conclusion

The development and application of large language models (LLMs) in economics have brought about significant advancements in the way text data is analyzed and utilized. The progression from basic text-as-data methods, such as the bag-of-words model, to sophisticated LLMs has addressed many of the limitations previously encountered, such as issues with polysemy, synonymy, and word order. By leveraging attention mechanisms and transformer architectures, LLMs like BERT, GPT, and their successors have enabled a more nuanced and context-aware understanding of language.

However, the journey from foundational models to high-value, task-specific tools requires further adaptation and alignment. Fine-tuning these models on domain-specific data, as seen with specialized models like FinBERT and ClimateBERT, has shown the potential for improving performance in specific areas of interest. These models also are proven to help with NLP for economics research, such as detecting remote work (Hansen et al. 2023).

Moreover, the alignment of LLMs with human preferences, particularly through processes like reinforcement learning with human feedback (RLHF), has been crucial in developing AI assistants that can effectively respond to user instructions. This step ensures that models not only predict language accurately but also generate outputs that align with user expectations and needs, making them more reliable and useful in practical applications.

Despite these advancements, several challenges remain. A number of choices must be made when fitting an LLM: the prediction target, the training data, the length of the context window, and the specific architecture of the neural network (e.g. number of attention operations and size of embeddings). While in the early stages of development individual researchers could tweak these to assess model performance, this is difficult for the current generation of models. One reason is that models have become large enough that estimating them from scratch is too capital-intensive for all but the most well-funded tech companies. Beyond this, however, some developers of LLMs also do not publish the sources of their training data nor do they make available the estimated network parameters for download. The tension between tech companies' profit-driven choices researchers' desire for replicability and transparency has yet to be resolved.

Ultimately, while LLMs offer powerful tools for economic analysis, their full potential can only be realized through careful adaptation and alignment to specific contexts and user needs. We have seen in this paper a number of ways to adapt the models to make them more useful. As the field continues to evolve, ongoing collaboration between researchers, developers, and policymakers will be essential to ensure that these models are developed and applied in ways that are both effective and ethically sound.

# References


Arold, B. W., Ash, E., MacLeod, W. B., and Naidu, S. (2024). Do words matter? the value of collective bargaining agreements.

Arora, S., Liang, Y., and Ma, T. (2017). A simple but tough-to-beat baseline for sentence embeddings. In *International conference on learning representations*.

Ash, E., Chen, D. L., and Ornaghi, A. (2024a). Gender attitudes in the judiciary: Evidence from us circuit courts. *American Economic Journal: Applied Economics*, 16(1):314–350.

Ash, E., Gauthier, G., and Widmer, P. (2024b). Relatio: Text semantics capture political and economic narratives. *Political Analysis*, 32(1):115–132.

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

Ash, E., Morelli, M., and Vannoni, M. (2024c). More laws, more growth? evidence from u.s. states. *Journal of Political Economy*.

Bana, S. H. (2022). Work2vec: using language models to understand wage premia. *Unpublished Manuscript*.

Beltagy, I., Lo, K., and Cohan, A. (2019). SciBERT: A Pretrained Language Model for Scientific Text. *arXiv preprint arXiv:1903.10676*.

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.

Chalkidis, I., Fergadiotis, M., Malakasiotis, P., Aletras, N., and Androutsopoulos, I. (2020). LEGAL-BERT: The Muppets straight out of Law School. In *Findings of the Association for Computational Linguistics: EMNLP 2020*.

Chopra, F. and Haaland, I. (2023). Conducting qualitative interviews with ai.

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.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational*

*Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

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.

Gennaro, G. and Ash, E. (2022). Emotion and reason in political language. *The Economic Journal*, 132(643):1037–1059.

Gentzkow, M., Kelly, B., and Taddy, M. (2019a). Text as Data. *Journal of Economic Literature*, 57(3):535–574.

Gentzkow, M., Shapiro, J. M., and Taddy, M. (2019b). Measuring Group Differences in High-Dimensional Choices: Method and Application to Congressional Speech. *Econometrica*, 87(4):1307–1340.

Goldberg, Y. (2017). Introduction. In Goldberg, Y., editor, *Neural Network Methods for Natural Language Processing*, pages 1–9. Springer International Publishing, Cham.

Hansen, S., Lambert, P. J., Bloom, N., Davis, S. J., Sadun, R., and Taska, B. (2023). Remote work across jobs, companies, and space. Technical report, National Bureau of Economic Research.

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.

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., Dehghani, M., Phoo, E., et al. (2019). Parameter-Efficient Transfer Learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, pages 2790–2799.

Howard, J. and Ruder, S. (2018). Universal Language Model Fine-tuning for Text Classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339.

Hu, E. J., Shen, T., Wallis, P., Allen-Zhu, Z., Li, S., Wang, L., and Chen, W. (2021). LoRA: Low-Rank Adaptation of Large Language Models. *arXiv preprint arXiv:2106.09685*.

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.

Jurafsky, D. and Martin, J. H. (2024). Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition.

Ke, S., Olea, J. L. M., and Nesbit, J. (2021). Robust Machine Learning Algorithms for Text Analysis. Unpublished manuscript.

Korinek, A. (2023). Generative ai for economic research: Use cases and implications for economists. *Journal of Economic Literature*, 61(4):1281–1317.

Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press, New York, illustrated edition edition.

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]*.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., et al. (2022). Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*.

Rogers, A., Kovaleva, O., and Rumshisky, A. (2021). A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Shapiro, A. H., Sudhof, M., and Wilson, D. J. (2020). Measuring news sentiment. *Journal of Econometrics*.

Stiennon, N., Ouyang, L., Wu, J., Ziegler, D., Lowe, R., Voss, C., Radford, A., Amodei, D., and Christiano, P. F. (2020). Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021.

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.

Webersinke, N., Kraus, M., Bingler, J. A., and Leippold, M. (2021). Climatebert: A pretrained language model for climate-related text. *arXiv preprint arXiv:2110.12010*.

Yang, Y., UY, M. C. S., and Huang, A. (2020). Finbert: A pretrained language model for financial communications.