

# How ChatGPT and OpenAI Large Language Models Are Developed



Written by Michael Schade

OpenAI's large language models, including the models that power ChatGPT, are developed using three primary sources of information: (1) information that is publicly available on the internet, (2) information that we license from third parties, and (3) information that our users or our human trainers provide. This article provides an overview of the publicly available information we use to help develop our models and how we collect and use that information in compliance with privacy laws. To understand how we collect and use information from users of our services, including how to opt out of having ChatGPT conversations used to help teach our models, please see our [Privacy Policy](#) and [this help center article](#).

## What is ChatGPT, and how does it work?

ChatGPT is an artificial intelligence-based service that you can access via the internet. You can use ChatGPT to organize or summarize text, or to write new text. ChatGPT has been developed in a way that allows it to understand and respond to user questions and instructions. It does this by "reading" a large amount of existing text and learning how words tend to appear in context with other words. It then uses what it has learned to predict the next most likely word that might appear in response to a user request, and each subsequent word after that. This is similar to auto-complete capabilities on search engines, smartphones, and email programs.

As an example, during the model learning process (called "training"), we might have a model try to complete the sentence: "instead of turning left, she turned \_\_\_\_." Before training, the model will respond with random words, but as it reads and learns from many lines of text, it better understands this type of sentence and can predict the next word more accurately. It then repeats this process across a very large number of sentences.

Because there are many possible words that could come next in this sentence (e.g., instead of turning left, she turned "right," "around," or "back"), there is an element of randomness in the way a model can respond, and in many cases our models will answer the same question in different ways.

Machine learning models are made up of large strings of numbers, called "weights" or "parameters," and code that interprets and executes those numbers. Models do not contain or store copies of information that they learn from. Instead, as a model learns, some of the numbers that make up the model change slightly to reflect what it has learned. In the example above, the model read information that helped it improve from predicting random incorrect words to predicting more accurate words, but all that actually happened in the model itself was that the numbers changed slightly. The model did not store or copy the sentences that it read.

## What type of information is used to teach ChatGPT?

As noted above, ChatGPT and our other services are developed using (1) information that is publicly available on the internet, (2) information that we license from third parties, and (3) information that our users or human trainers provide. This article focuses on the first set: information that is publicly available on the internet.

For this set of information, we only use publicly available information that is freely and openly available on the Internet – for example, we do not seek information behind paywalls or from the "dark web." We apply filters and remove information that we do not want our models to learn from or output, such as hate

speech, adult content, sites that primarily aggregate personal information, and spam. We then use the information to teach our models.

As mentioned in the previous section, ChatGPT does not copy or store training information in a database. Instead, it learns about associations between words, and those learnings help the model update its numbers/weights. The model then uses those weights to predict and generate new words in response to a user request. It does not “copy and paste” training information – much like a person who has read a book and sets it down, our models do not have access to training information after they have learned from it.

## Is personal information used to teach ChatGPT?

A large amount of data on the internet relates to people, so our training information does incidentally include personal information. We don't actively seek out personal information to train our models.

We use training information only to help our models learn about language and how to understand and respond to it. We do not and will not use any personal information in training information to build profiles about people, to contact them, to advertise to them, to try to sell them anything, or to sell the information itself.

Our models may learn from personal information to understand how things like names and addresses fit within language and sentences, or to learn about famous people and public figures. This makes our models better at providing relevant responses.

## How does the development of ChatGPT comply with privacy laws?

We use training information lawfully. Large language models have many applications that provide significant benefits and are already helping people create content, improve customer service, develop software, customize education, support scientific research, and much more. These benefits cannot be realized without a large amount of information to teach the models. In addition, our use of training information is not meant to negatively impact individuals, and the sources of this training information are already publicly available. For these reasons, we base our collection and use of personal information that is included in training information on legitimate interests according to privacy laws like the GDPR. To fulfill our compliance obligations, we have also completed a data protection impact assessment to help ensure we are collecting and using this information legally and responsibly.

We respond to objection requests and similar rights. As a result of learning language, ChatGPT responses may sometimes include personal information about individuals whose personal information appears multiple times on the public internet (for example, public figures). Individuals in certain jurisdictions can object to the processing of their personal information by our models by filling out [this form](#). Individuals also may have the right to access, correct, restrict, delete, or transfer their personal information that may be included in our training information. You can exercise these rights by reaching out to [dsar@openai.com](mailto:dsar@openai.com). Please be aware that, in accordance with privacy laws, some rights may not be absolute. We may decline a request if we have a lawful reason for doing so. However, we strive to prioritize the protection of personal information and comply with all applicable privacy laws. If you feel we have not adequately addressed an issue, you have the right to lodge a complaint with your local supervisory authority.

We protect training information and limit how it is used and shared. To keep this information safe, we use commercially reasonable technical, physical, and administrative measures like access controls, audit logs, read-only permissions, and encrypting stored data. For more information on our security practices, please visit <https://www.openai.com/security>.

We also take steps to reduce the processing of personal information when training our models. For example, we remove websites that aggregate large volumes of personal information and we try to train our models to reject requests for private or sensitive information about people.

We do not sell training information to third parties, and only disclose portions of the information when necessary and consistent with our [Privacy Policy](#).

We only keep this information for as long as we need it to serve its intended purpose. How long we keep this information hinges on factors like its quantity, type, and sensitivity, the risk of harm from unauthorized

use or sharing, whether the information is still necessary or useful to train or update our models, and any legal requirements.

Our data controller under the GDPR is OpenAI OpCo, LLC at 3180 18th Street, San Francisco, CA, United States. For information about our EEA and UK representative for data protection matters, please see our [Privacy Policy](#). Our Data Protection Officer can be contacted at [privacy@openai.com](mailto:privacy@openai.com).

---

## Some Definitions provided by ChatGPT

User

How are neural networks related to large language models

ChatGPT

Neural networks are a fundamental component of large language models like GPT-3.5, including the previous GPT versions. Large language models leverage deep neural networks to process and generate text.

A neural network is a computational model inspired by the structure and function of the human brain. It consists of interconnected nodes or "neurons" organized in layers. Neural networks can learn from data through a process called training, where they adjust the weights and biases of the connections between neurons based on the patterns and relationships in the training data.

Large language models utilize neural networks, particularly deep neural networks known as deep learning models. These models consist of multiple layers of neurons, enabling them to capture complex patterns and dependencies in language data.

In the case of GPT-3.5, the model employs a transformer architecture, which is a type of deep neural network specifically designed for natural language processing tasks. Transformers are composed of multiple self-attention layers and feed-forward neural networks. The self-attention mechanism allows the model to focus on different parts of the input text while generating the output.

During training, large language models like GPT-3.5 are exposed to massive amounts of text data to learn the statistical properties and semantic relationships of language. The trained neural network can then generate coherent and contextually relevant text based on the input provided to it. This process is known as "conditional text generation."

In summary, neural networks form the backbone of large language models like GPT-3.5, enabling them to understand and generate human-like text through the power of deep learning and pattern recognition.

---

How are large language models different than artificial general intelligence?

ChatGPT

Large language models and artificial general intelligence (AGI) are distinct concepts, although they both fall under the umbrella of artificial intelligence (AI). Here are the key differences between the two:

**Scope and Capability:** Large language models, such as GPT-3.5, are specialized models designed for language-related tasks. They excel at understanding and generating text but do not possess a comprehensive understanding of the world or exhibit general intelligence across multiple domains. They are highly proficient in processing and generating human-like text but lack the broader cognitive abilities associated with AGI.

AGI, on the other hand, refers to highly autonomous systems that possess general intelligence comparable to human intelligence. AGI systems would have the ability to understand, learn, and apply knowledge across a wide range of tasks and domains, exhibiting human-like cognitive capabilities. Unlike large language models, AGI aims to encompass general problem-solving skills and adaptability to various contexts.

**Versatility:** Large language models are predominantly focused on natural language processing tasks. They excel at tasks like language translation, text summarization, question-answering, and text completion. While they can provide impressive text-based outputs, they lack versatility beyond language-related applications.

AGI, in contrast, is envisioned as a flexible and adaptable intelligence that can handle diverse tasks and contexts. AGI systems would be capable of learning and performing various tasks across multiple domains, surpassing the limitations of specialized models like large language models.

**Understanding and Reasoning:** Large language models primarily rely on statistical patterns and associations in the training data to generate text. While they can mimic human-like responses, they lack true understanding and reasoning. They lack a deep understanding of context, lack commonsense reasoning, and may generate plausible but incorrect or nonsensical responses.

AGI, as a comprehensive intelligence, would possess a deeper understanding of the world, context, and exhibit reasoning abilities. AGI systems would be able to infer relationships, make logical deductions, exhibit abstract thinking, and apply knowledge to solve complex problems.

**Goal-Directed Behavior:** Large language models are not inherently goal-directed. They generate text based on the provided input and training data patterns but lack autonomous decision-making or intentionality. They don't possess a purpose or motivation beyond language processing.

AGI, in contrast, would exhibit goal-directed behavior and have the ability to formulate and pursue goals autonomously. AGI systems would actively seek to achieve objectives, plan and execute actions, and exhibit intentionality in their behavior.

In summary, large language models like GPT-3.5 are powerful language processing models but are limited to language-related tasks and lack the broader cognitive abilities, understanding, versatility, and autonomy associated with artificial general intelligence. AGI represents a more comprehensive form of intelligence that encompasses general problem-solving skills, adaptability across domains, true understanding, reasoning abilities, and goal-directed behavior.

---

# ChatGPT and Large Language Models: Six Evolutionary Steps

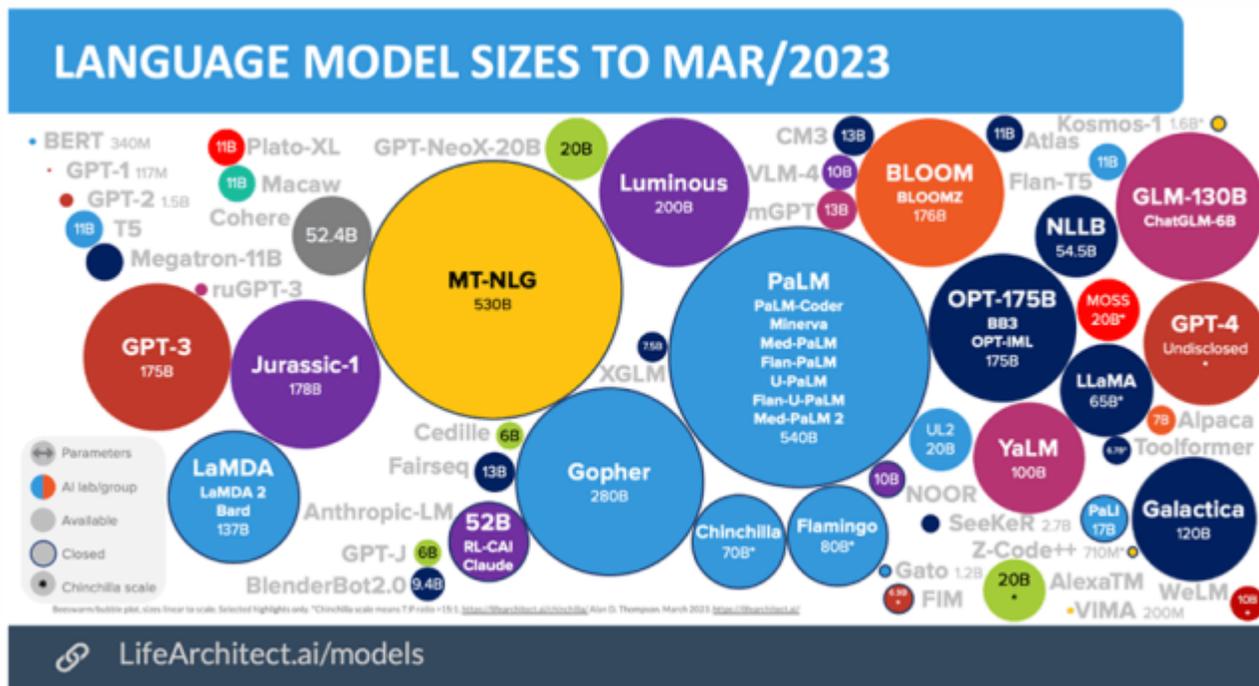
By *Dan Philps, PhD, CFA* and *Tillman Weyde, PhD*

...

## GPT: The Big Guy or the Smart Guy?

The ChatGPT and ChatGPT Plus chatbots sit on top of GPT-3 and GPT-4 neural networks, respectively. In terms of model size, Google's PaLM 2, NVIDIA's [Megatron-Turing Natural Language Generation \(MT-NLG\)](#), and now GPT-4 have eclipsed GPT-3 and its variant GPT-3.5,

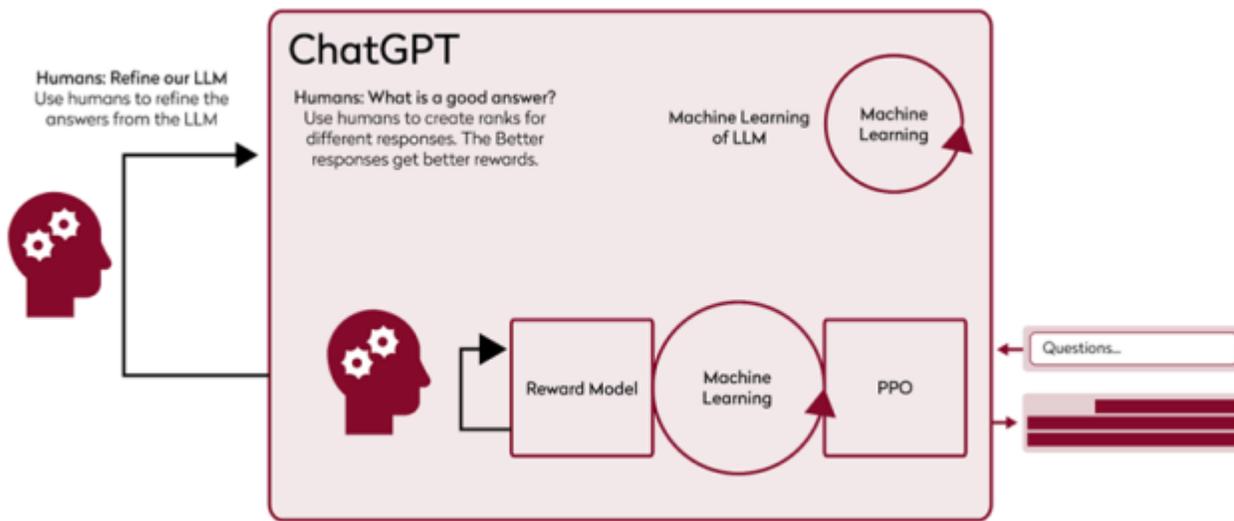
which is the basis of ChatGPT. Compared to its predecessors, GPT-4 produces smoother text of better linguistic quality, interprets more accurately, and, in a subtle but significant advance over GPT-3.5, can handle much larger input prompts. These improvements are the result of training and optimization advances — additional “smarts” — and probably the pure brute force of more parameters, but OpenAI does not share technical details about GPT-4.



## ChatGPT Training: Half Machine, Half Human

ChatGPT is an LLM that is fine-tuned through reinforcement learning, specifically [reinforcement learning from human feedback \(RLHF\)](#). The process is simple in principle: First humans refine the LLM on which the chatbot is based by categorizing, on a massive scale, the accuracy of the text the LLM produces. These human ratings then train a reward model that automatically ranks answer quality. As the chatbot is fed the same questions, the reward model scores the chatbot’s answers. These scores go back into fine-tuning the chatbot to produce better and better answers through the [Proximal Policy Optimization \(PPO\)](#) algorithm.

## ChatGPT Training Process



Source: Rothko

## Investment Strategies

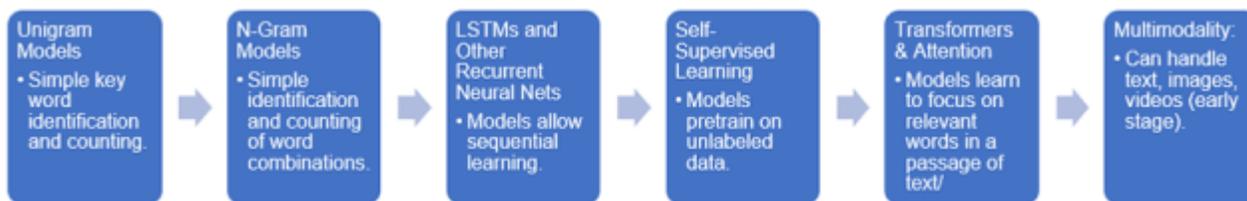
### The Machine Learning behind ChatGPT and LLMs

LLMs are the latest innovation in natural language processing (NLP). A core concept of NLP are language models that assign probabilities to sequences of words or text —  $S = (w_1, w_2, \dots, w_m)$  — in the same way that our mobile phones “guess” our next word when we are typing text messages based on the model’s highest probability.

### Steps in LLM Evolution

The six evolutionary steps in LLM development, visualized in the chart below, demonstrate how LLMs fit into NLP research.

#### The LLM Tech (R)Evolution



### 1. Unigram Models

The unigram assigns each word in the given text a probability. To identify news articles that describe fraud in relation to a company of interest, we might search for “fraud,” “scam,” “fake,” and “deception.” If these words appear in an article more than in regular language, the article is likely discussing fraud. More specifically, we can assign a probability that a piece of text is

about. More specifically, we can assign a probability that a piece of text is about fraud by multiplying the probabilities of individual words:

$$P(S) = \prod_{i=1}^m P(w_i)$$

In this equation,  $P(S)$  denotes the probability of a sentence  $S$ ,  $P(w_i)$  reflects the probability of a word  $w_i$  appearing in a text about fraud, and the product taken over all  $m$  words in the sequence, determines the probability that these sentences are associated with fraud.

These word probabilities are based on the relative frequency at which the words occur in our corpus of fraud-related documents, denoted as  $D$ , in the text under examination. We express this as  $P(w) = \text{count}(w) / \text{count}(D)$ , where  $\text{count}(w)$  is the frequency that word  $w$  appears in  $D$  and  $\text{count}(D)$  is  $D$ 's total word count.

A text with more frequent words is more probable, or more typical. While this may work well in a search for phrases like “identify theft,” it would not be as effective for “theft identify” despite both having the same probability. The unigram model thus has a key limitation: It disregards word order.

## 2. N-Gram Models

“You shall know a word by the company it keeps!” — [John Rupert Firth](#)

The n-gram model goes further than the unigram by examining subsequences of several words. So, to identify articles relevant to fraud, we would deploy such bigrams as “financial fraud,” “money laundering,” and “illegal transaction.” For trigrams, we might include “fraudulent investment scheme” and “insurance claim fraud.” Our fourgram might read “allegations of financial misconduct.”

This way we condition the probability of a word on its preceding context, which the n-gram estimates by counting the word sequences in the corpus on which the model was trained.

The formula for this would be:

$$P(w_i | w_{i-(n-1)}, \dots, w_{i-1}) = \frac{\text{count}(w_{i-(n-1)}, \dots, w_{i-1}, w_i)}{\text{count}(w_{i-(n-1)}, \dots, w_{i-1})}$$

This model is more realistic, giving a higher probability to “identify theft” rather than “theft identify,” for example. However, the counting method has some pitfalls. If a word sequence does not occur in the corpus, its probability will be zero, rendering the entire product as zero.

As the value of the “n” in n-gram increases, the model becomes more precise in its text search. This enhances its ability to identify pertinent themes, but may lead to overly narrow searches.

The chart below shows a simple n-gram textual analysis. In practice, we might remove “stop words” that provide no meaningful information, such as “and,” “in,” “the,” etc., although LLMs do keep them.

---

### Understanding Text Based on N-Grams

Unigram	<b>Modern-slavery</b> practices including <i>bonded-labor</i> have been identified in the <u>supply-chain</u> of Company A
Bigrams	<b>Modern-slavery practices</b> including <i>bonded-labor have</i> been identified in <u>the supply-chain</u> of Company A
Trigrams	<b>Modern-slavery practices including</b> <i>bonded-labor have been</i> identified <u>in the supply-chain</u> of Company A
Fourgrams	<b>Modern-slavery practices including bonded-labor</b> have been identified <u>in the supply-chain of</u> Company A

---

### 3. Neural Language Models (NLMs)

In NLMs, machine learning and neural networks address some of the shortcomings of unigrams and n-grams. We might train a neural network model  $N$  with the context  $(w_{i-(n-1)}, \dots, w_{i-1})$  as the input and  $w_i$  as the target in a straightforward manner. There are many clever tricks to improve language models, but fundamentally all that LLMs do is look at a sequence of

words and guess which word is next. As such, the models characterize the words and generate text by sampling the next word according to the predicted probabilities. This approach has come to dominate NLP as deep learning has developed over the last 10 years.

#### **4. Breakthrough: Self-Supervised Learning**

Thanks to the internet, larger and larger datasets of text became available to train increasingly sophisticated neural model architectures. Then two remarkable things happened:

First, words in neural networks became represented by vectors. As the training datasets grow, [these vectors arrange themselves according to the syntax and semantics of the words](#).

Second, simple [self-supervised](#) training of language models turned out to be unexpectedly powerful. Humans no longer had to manually label each sentence or document. Instead, the model learned to predict the next word in the sequence and in the process also gained other capabilities. Researchers realized that pre-trained language models provide great foundations for text classification, sentiment analysis, question answering, and other NLP tasks and that the process became more effective as the size of the model and the training data grew.

This paved the way for sequence-to-sequence models. These include an encoder that converts the input into a vector representation and a decoder that generates output from that vector. These neural sequence-to-sequence models outperformed previous methods and were incorporated into [Google Translate in 2016](#).

#### **5. State-of-the-Art NLP: Transformers**

Until 2017, recurrent networks were the most common neural network architecture for language modeling, long short-term memory (LSTM), in particular. The size of LSTM's context is theoretically infinite. The models were also made bi-directional, so that also all future words were considered as well as past words. In practice, however, the benefits are limited and the recurrent structure makes training more costly and time consuming: It's hard to parallelize the training on GPUs. For mainly this reason, transformers supplanted LSTMs.

Transformers build on the attention mechanism: The model learns how much weight to attach to words depending on the context. In a recurrent model, the most recent word has the most

direct influence on predicting the next word. With attention, all words in the current context are available and the models learn which ones to focus on.

In their aptly titled paper, “[Attention is All You Need](#),” Google researchers introduced Transformer sequence-to-sequence architecture, which has no recurrent connections except that it uses its own output for context when generating text. This makes the training easily parallelizable so that models and training data can be scaled up to previously unheard of sizes. For classification, the [Bidirectional Encoder Representations from Transformers \(BERT\)](#) became the new go-to model. For text generation, the race was now on to scale up.

## 6. Multimodal Learning

While standard LLMs are trained exclusively on textual data, other models — GPT-4, for example — include images or audio and video. In a financial context, these models could examine chart, images, and videos, from CEO interviews to satellite photography, for potentially investable information, all cross-referenced with news flow and other data sources.

### Criticism of LLMs

Transformer LLMs can predict words and excel at most benchmarks for NLP tasks, including answering questions and summarization. But they still have clear limitations. They memorize rather than reason and have no causal model of the world beyond the probabilities of words. [Noam Chomsky](#) described them as “high tech plagiarism,” and [Emily Bender et al.](#) as “stochastic parrots.” Scaling up the models or training them on more text will not address their deficits. [Christopher D. Manning](#) and [Jacob Browning and Yann LeCun](#), among other researchers, believe the focus should be on expanding the models’ technology to multimodality, including more structured knowledge.

LLMs have other scientific and philosophical issues. For example, to what extent can neural networks actually *learn* the nature of the world just from language? The answer could influence how reliable the models become. The economic and environmental costs of LLMs could also be steep. Scaling up has made them expensive to develop and run, which raises questions about their ecological and [economic sustainability](#).

# Artificial General Intelligence (AGI) Using LLMs?

Whatever their current limitations, LLMs will continue to evolve. Eventually they will solve tasks far more complex than simple prompt responses. As just one example, LLMs can become “controllers” of other systems and could in principle guide elements of investment research and other activities that are currently human-only domains. Some have described this as “Baby AGI,” and for us it is easily the most exciting area of this technology.

-----

## Large Language Model – technology references

<https://github.com/databricks-academy/large-language-models> (Databrick Notebooks)

### 0 - Natural Language Processing

- [Stanford Online Course on NLP](#)
- [Hugging Face NLP course](#)

### Language Modeling

- [TF-IDF](#)
- [Bag of Words](#)
- [LSTMs](#)
- [Language Modeling](#)

### Word Embeddings

- [Word2vec](#)
- [Tensorflow Page on Embeddings](#)

### Tokenization

- [Byte-Pair Encoding](#)
- [SentencePiece](#)
- [WordPiece](#)

## 1 - NLP tasks

- [Hugging Face tasks page](#)
- [Hugging Face NLP course chapter 7: Main NLP Tasks](#)
- Background reading on specific tasks
  - Summarization: [Hugging Face summarization task page](#) and [course section](#)
  - Sentiment Analysis: [Blog on “Getting Started with Sentiment Analysis using Python”](#)
  - Translation: [Hugging Face translation task page](#) and [course section](#)
  - Zero-shot classification: [Hugging Face zero-shot classification task page](#)
  - Few-shot learning: [Blog on “Few-shot learning in practice: GPT-Neo and the 😊 Accelerated Inference API”](#)

## [Hugging Face Hub](#)

- [Models](#)
- [Datasets](#)
- [Spaces](#)

## Hugging Face libraries

- [Transformers](#)
  - Blog post on inference configuration: [How to generate text: using different decoding methods for language generation with Transformers](#)
- [Datasets](#)
- [Evaluate](#)

## Models

- Base model versions of models used in the demo notebook
  - [T5](#)
  - [BERT](#)
  - [Marian NMT framework](#) (with 1440 language translation models!)
  - [DeBERTa](#) (Also see [DeBERTa-v2](#))
  - [GPT-Neo](#) (Also see [GPT-NeoX](#))
- [Table of LLMs](#)

## Prompt engineering

- [Best practices for OpenAI-specific models](#)
- [DAIR.AI guide](#)
- [ChatGPT Prompt Engineering Course](#) by OpenAI and DeepLearning.AI
- [Awesome ChatGPT Prompts](#) for fun examples with ChatGPT

---

## Research papers on increasing context length limitation

- [Pope et al 2022](#)
- [Fu et al 2023](#)

## Industry examples on using vector databases

- FarFetch
  - [FarFetch: Powering AI With Vector Databases: A Benchmark - Part I](#)
  - [FarFetch: Powering AI with Vector Databases: A Benchmark - Part 2](#)
  - [FarFetch: Multimodal Search and Browsing in the FARFETCH Product Catalogue - A primer for conversational search](#)
- [Spotify: Introducing Natural Language Search for Podcast Episodes](#)
- [Vector Database Use Cases compiled by Qdrant](#)

## Vector indexing strategies

- Hierarchical Navigable Small Worlds (HNSW)
  - [Malkov and Yashunin 2018](#)
- Facebook AI Similarity Search (FAISS)
  - [Meta AI Blog](#)
- Product quantization
  - [PQ for Similarity Search by Peggy Chang](#)

## Cosine similarity and L2 Euclidean distance

- [Cosine and L2 are functionally the same when applied on normalized embeddings](#)

## Filtering methods

- [Filtering: The Missing WHERE Clause in Vector Search by Pinecone](#)

## Chunking strategies

- [Chunking Strategies for LLM applications by Pinecone](#)
- [Semantic Search with Multi-Vector Indexing by Vespa](#)

## Other general reading

- [Vector Library vs Vector Database by Weaviate](#)
- [Not All Vector Databases Are Made Equal by Dmitry Kan](#)
- [Open Source Vector Database Comparison by Zilliz](#)
- [Do you actually need a vector database? by Ethan Rosenthal](#)

---

## LLM Chains

- [LangChain](#)
- [OpenAI ChatGPT Plugins](#)

## LLM Agents

- [Transformers Agents](#)
- [AutoGPT](#)
- [Baby AGI](#)
- [Dust.tt](#)

## Multi-stage Reasoning in LLMs

- [CoT Paradigms](#)
- [ReAct Paper](#)
- [Demonstrate-Search-Predict Framework](#)

---

## Fine-tuned models

- [HF leaderboard](#)
- [MPT-7B](#)
- [Stanford Alpaca](#)
- [Vicuna](#)
- [DeepSpeed on Databricks](#)

#### Databricks' Dolly

- [Dolly v1 blog](#)
- [Dolly v2 blog](#)
- [Dolly on Hugging Face](#)
- [Build your own Dolly](#)

#### Evaluation and Alignment in LLMs

- [HONEST](#)
- [LangChain Evaluate](#)
- [OpenAI's post on InstructGPT and Alignment](#)
- [Anthropic AI Alignment Papers](#)

---

#### Social Risks and Benefits of LLMs

- [Weidinger et al 2021 \(DeepMind\)](#)
- [Bender et al 2021](#)
- [Mokander et al 2023](#)
- [Rillig et al 2023](#)
- [Pan et al 2023](#)

#### Hallucination

- [Ji et al 2022](#)

#### Bias evaluation metrics and tools

- [NeMo Guardrails](#)
- [Guardrails.ai](#)

- [Liang et al 2022](#)

#### Other general reading

- [All the Hard Stuff Nobody Talks About when Building Products with LLMs by Honeycomb](#)
- [Science in the age of large language models by Nature Reviews Physics](#)
- [Language models might be able to self-correct biases—if you ask them by MIT Technology Review](#)

---

### General MLOps

- [“The Big Book of MLOps”](#) (eBook overviewing MLOps)
  - Blog post (short) version: [“Architecting MLOps on the Lakehouse”](#)
  - MLOps in the context of Databricks documentation ([AWS](#), [Azure](#), [GCP](#))

### LLMOps

- Blog post: Chip Huyen on [“Building LLM applications for production”](#)

### [MLflow](#)

- [Documentation](#)
  - [Quickstart](#)
  - [Tutorials and examples](#)
- Overview in Databricks ([AWS](#), [Azure](#), [GCP](#))

### [Apache Spark](#)

- [Documentation](#)
  - [Quickstart](#)
- Overview in Databricks ([AWS](#), [Azure](#), [GCP](#))

### [Delta Lake](#)

- [Documentation](#)

- Overview in Databricks ([AWS](#), [Azure](#), [GCP](#))
- [Lakehouse Architecture \(CIDR paper\)](#)