

When Large Language Models Meet Vector Databases: A Survey

Zhi Jing^{*1}, Yongye Su^{*2}, Yikun Han^{*3}, Bo Yuan^{*4}, Chunjiang Liu⁵, Haiyun Xu⁶, Kehai Chen³

¹Carnegie Mellon University

²Purdue University

³University of Michigan

⁴Harbin Institute of Technology

⁵National Science Library (Chengdu), Chinese Academy of Sciences

⁶Shandong University of Technology

zjing2@cs.cmu.edu, su311@purdue.edu, yikunhan@umich.edu, 23s051006@stu.hit.edu.cn,
liucj@clas.ac.cn, chenkehai@hit.edu.cn, xuhy@sdut.edu.cn

Abstract

The recent burst in Large Language Models has opened new frontiers in human-like text processing and generation. However, alongside their remarkable growth, Large Language Models have encountered critical challenges including issues of hallucination, bias, real-time knowledge updates, and the high costs of implementation and maintenance in commercial settings. Vector Databases, another increasingly popular tool, offer potential solutions to these challenges. These databases are adept at handling high-dimensional data and are crucial for tasks such as efficient information retrieval and semantic search. By integrating with Large Language Models, they significantly enhance AI systems' ability to manage and utilize diverse data more effectively. This survey paper provides an in-depth and unique analysis of the intersection between Large Language Models and Vector Databases.

1 Introduction

The concept of Artificial Intelligence (AI) was envisioned decades ago, but it is not until ChatGPT unlocked the power of interactive smart AI assistants that people are offered with the unprecedentedly tangible experience of talking to a machine program that "seems" more knowledgeable than an ordinary person. The phenomenal heat of ChatGPT inherits from the success of Large Language Models (LLMs), one major element of AI. LLMs, exemplified by systems like GPT [Brown and et al., 2020; Achiam and et al., 2023], BERT [Devlin *et al.*, 2018], and Llama [Touvron and et al., 2023], can process, understand, and generate human-like text. Thanks to the power of pre-training which involves training a language model on a massive corpus of text data, LLMs can capture the complexities of human languages, including context, idioms, and even cultural references. Their applications range from simple text completion to complex tasks like translation, understanding, and generation, making them invaluable assets in both commercial and research domains.

However, despite the impressive capabilities of LLMs, they

are still under the shadows of doubt in many aspects. One major shortcoming is the problem of hallucination, where LLMs generate plausible but factually incorrect or faithfully nonsensical information [Huang *et al.*, 2023]. Causes behind this problem include 1. Lack of domain knowledge. The fact that LLMs are primarily trained on public datasets [Penedo *et al.*, 2023] inevitably leads to a limited ability to answer domain-specific questions that are out of the scope of their internal knowledge. 2. Real-time knowledge updates. Even if the questions are within the learning corpus of LLMs, their answers may still exhibit limitations because the internal knowledge may be outdated when the outside world is dynamic and keeps changing [Onoe *et al.*, 2022]. 3. Biases. The datasets used to train LLMs are large which may introduce systematic errors [Bender *et al.*, 2021]. And essentially every dataset can be questioned with biases issues, including imitative falsehoods [Bender *et al.*, 2021], duplicating biases [Lee *et al.*, 2022], and social biases [Venkit *et al.*, 2023]. Moreover, a disadvantage of incorporating LLMs commercially is the expensive cost of maintenance. For an average business entity, applying LLMs for business use is barely feasible. It is almost impossible for a non-tech company to customize and train a GPT model of its own because they don't have the resources and talents to conduct such big of a project [Musser, 2023], while frequent API calls to third-party LLM providers like OpenAI can be extremely expensive, not to mention there is a very limited number of such providers in certain areas. Additionally, the oblivion problem of LLMs has been of controversial because LLMs are proven to tend to forget.

Another increasingly popular sub-field of AI is AI databases that support vector data storage and its efficient retrieval at scale, also called vector databases (VecDBs). Whatever kind of multi-modal data that LLMs deal with, searching over many vectors is time-consuming, it is recommended by OpenAI and other LLM vendors to use VecDBs, which always provide LLMs and their applications with cost-effective retrievals and scalable data management.

The intersection of Large Language Models and Vector Databases has been and will still be a burgeoning area of study and application, offering exciting possibilities. This synergy allows for the creation of more powerful, efficient,

Structure	Encoder Only	Encoder-Decoder	Decoder Only
LLMs	BERT [Devlin <i>et al.</i> , 2018], RoBERTa [Liu <i>et al.</i> , 2019], XLNet [Lample and Conneau, 2019], ALBERT [Lan and <i>et al.</i> , 2019], ELECTRA [Clark and <i>et al.</i> , 2020], DeBERTa [He and <i>et al.</i> , 2020]	T5 [Raffel <i>et al.</i> , 2020], BART [Lewis and <i>et al.</i> , 2019], mT5 [Xue and <i>et al.</i> , 2020], M2M-100 [Fan and <i>et al.</i> , 2021], BigBird [Zaheer and <i>et al.</i> , 2020], ChatGLM [Zeng and <i>et al.</i> , 2022]	GPT-2 [Radford and <i>et al.</i> , 2019], GPT-3 [Brown and <i>et al.</i> , 2020], OPT [Zhang and <i>et al.</i> , 2022], PaLM [Chowdhery and <i>et al.</i> , 2022], BLOOM [Workshop and <i>et al.</i> , 2022], MT-NLG [Smith and <i>et al.</i> , 2022], GLaM [Du and <i>et al.</i> , 2022], Gopher [Rae and <i>et al.</i> , 2021], chinchilla [Hoffmann and <i>et al.</i> , 2022], LaMDA [Thoppilan and <i>et al.</i> , 2022], LLaMA [Touvron and <i>et al.</i> , 2023], GPT-4 [Achiam and <i>et al.</i> , 2023], BloombergGPT [Wu and <i>et al.</i> , 2023]

Table 1: Summary of LLMs

and versatile AI systems, capable of handling a broad spectrum of tasks with enhanced accuracy and speed. It also paves the way for innovative applications in fields like healthcare, finance, education, and entertainment, where AI can deliver tailored solutions and insights.

In the light of lacking papers that introduce LLMs in the view of Vector Databases, this survey aims to picture how Vector Databases can be potential solutions to refine Large Language Models’ known shortcomings in previous works and hope to offer a unique perspective of future directions in the intersection that is fertile of research potentials. This paper develops as below:

- Section 2 is presented as introduction to the background knowledge of the two major characters of this paper, LLMs and Vector Databases. Section 2.1 offers a brief view of the popular LLMs and accents on their underlying shortcomings, while Section 3.1 depicts the picture of Vector Databases and their unique capabilities in integration with LLMs.
- In Section 3, we provide a comprehensive summary of how previous research has effectively combined LLMs with Vector Databases. Section 3.2 delves into a detailed blueprint of the Retrieval-Augmented Generation paradigm in which a unique role that Vector Databases play. Section 3.3 and 3.4 showcases other memory-wise benefits that vector databases can bring for LLMs.
- Section 4 is dedicated to our analysis of the limitations and unexplored areas at the intersection of LLMs and Vector Databases and highlights the potential future research opportunities in this domain.

2 Background

2.1 Large Language Models

Over the last half-decade, we have witnessed the groundbreaking success of Large Language Models (LLMs), marking a significant milestone in the field of Natural Language Processing (NLP). LLMs have revolutionized our views on

the approaches to understanding and generating human languages by machines. There are two key factors in the evolution of Large Language Models, the development of Neural Network architectures and the superpower of pre-training models on extensive data.

Development of Language Models

With the advent of neural networks, the field of NLP has undergone a transformative shift, which began with the introduction of Recurrent Neural Networks (RNNs) [Zaremba *et al.*, 2014]. RNNs provide a way to process sequences of words and capture temporal dependencies in text. And 2 of its famous variants, Gated Recurrent Units (GRUs) [Chung *et al.*, 2014] and Long Short-Term Memory networks (LSTMs) [Shi *et al.*, 2015] address the limitations of RNNs in handling problems of vanishing and exploding gradients.

The next pivotal milestone for NLP is the widespread adoption of the Transformer architecture [Vaswani *et al.*, 2017] which has set a new standard for language models. The multi-head self-attention modules and cross-attention modules in the encoders and decoders enable the model to capture long-range dependencies, parallel processing, and contextual understanding. Furthermore, the model starts to rapidly grow in size in the Transformer era. More importantly, the Transformer represents a paradigm shift in NLP field, and the success of the Transformer architecture becomes the significant backbone of LLMs, upon which the *encoder-only*, *encoder-decoder*, and *decoder-only* models are built.

Encoder-Only Models These models are designed to analyze and understand input text. They process the input to create representations, in the form of embedding vectors, that capture the nuances and context of the language. Examples of the models in this category include BERT [Devlin *et al.*, 2018] and its derivatives, such as RoBERTa [Liu *et al.*, 2019]. They excel in tasks that require a deep understanding of language context, such as sentiment analysis, named entity recognition, and question answering where the answers are contained within the given text.

Encoder-Decoder Models This architecture consists of two parts: an encoder that processes the input text and a de-

coder that generates the output. These models are particularly effective in tasks that involve transforming an input into a different output, such as machine translation or text summarizing. The Transformer model itself, as we have discussed before, is an example of this category, which has brought astonishing breakthroughs in machine translation. Another significant model is T5 [Raffel *et al.*, 2020], which frames all language tasks as a text-to-text problem, showcasing the versatility of the encoder-decoder framework.

Decoder-Only Models Decoder-only architectures are optimized for generating text. These models take an input and then continue to generate language based on that input. GPT-3 [Brown and et al., 2020] and its successor, GPT-4 [Achiam and et al., 2023], are quintessential examples of this category. They are particularly adept at tasks that require creative and coherent text generation, such as content creation, storytelling, and even code generation.

We show the LLMs that have received notable attention during the development of language models in Table 1.

3 LLM+VectorDB

Power of Pre-training

The scaling laws proposed by OpenAI [Kaplan and et al., 2020] highlight a critical trend: the scaling of Pretrained Language Models (PLMs), in terms of both model size and data volume, leads to significant improvement on downstream tasks. This is evidenced by the development of increasingly larger PLMs, such as the 1.76-trillion-parameter GPT-4 [Achiam and et al., 2023] and the 540-billion-parameter PaLM [Chowdhery and et al., 2023]. Unlike their smaller predecessors, like the 330-million-parameter BERT [Devlin *et al.*, 2018] and 1.5-billion-parameter GPT-2 [Radford and et al., 2019], these large-scale models demonstrate unique behaviors and emergent abilities [Wei and et al., 2022] in various tasks like zero-shot learning, which is extremely challenging to the small-scale models. Researchers started to call the large-scale models in the NLP field “Large Language Models” or “LLMs” to denote these significantly scaled PLMs, in order to distinguish the outstanding performance and gain more traction in academia.

The superpower of pre-training lies in its ability to provide models with a general understanding of the languages, which is then tailored through additional training, known as fine-tuning, for specific tasks unlike traditional deep learning methods [Dong *et al.*, 2021; Liu and et al., 2024]. The advantage is that the model doesn’t start from scratch when learning a new task; it builds upon a rich, pre-existing foundation of language understanding. And this enables what is known as transfer learning [Yosinski and et al., 2014]. Knowledge gained during this initial phase can be transferred to a wide range of tasks, even those that the model was not explicitly trained on.

Are we there yet? (Challenge faced by pure LLMs)

However, the success of LLMs does not come without challenges. Issues such as data biases [Raffel *et al.*, 2020], which can lead to skewed or unfair outcomes [Li and et al., 2023], are of significant concern. Ethical considerations, including

privacy problems [Yao *et al.*, 2024] and the potential for misuse of generative content [Ganguli *et al.*, 2022]. Training language models to follow instructions with human feedback, like toxic content, also receives quite a bit of debate. Researchers are arguing that the power of LLMs is very superficial, and it is the large pre-training data [Schaeffer, 2023] that helps the models gain excellent performance on most of the benchmarks. Hallucinations [Ji and et al., 2023] are also found to be a major problem with LLMs, casting doubt on the reliability of their outputs. Additionally, the computational resources required for training and running these models are substantial, posing environmental and accessibility questions [Strubell and et al., 2019]. As the sizes of both LLMs and the training datasets keep getting bigger, the cost of training and inferring is heavily influenced. It is estimated that training the 11-billion parameter version of T5 [Raffel *et al.*, 2020] costs over 1.3 million dollars for a single run, whereas one round of training GPT-3 [Brown and et al., 2020] of 175 billion parameters using a Tesla V100 cloud instance requires costs 4.6 million dollars. On top of that, training a BERT-based language model [Devlin *et al.*, 2018] using 8 V100 GPUs for 36 hours and used a total of 37.3 kWh which is estimated to be more energy-consuming than a gallon of gasoline in terms of CO_2 emission [Dodge and et al., 2022].

3.1 Vector Databases, the V-factor

While LLMs like ChatGPT are relatively new concepts, database management systems (DBMS) have been thoroughly developed and applied in many aspects in the last 60 years of history, well-recognized for their consistent stability and universality for structured data with fixed formats that excel well with computer storage. However, the development and wide application of deep learning models such as convolutional neural networks [He and et al., 2016] and transformers [Devlin *et al.*, 2018; Su and et al., 2022], enables the embedding of unstructured multi-modal data like images and text mapped into corresponding fixed-length vector representations, which include the high-dimensional semantic features of original data and the semantic similarities are natively represented by distances between vectors, requiring a new type of DBMS that is specifically designed for handling vector data operations, especially vector search and storage.

There are many kinds of databases, as shown in Table 2, whereas vector databases are the only category of databases that natively support diverse unstructured data with efficient storage, indexing, and retrieval. All these operations could be done based on the vector indexes, which are optimally designed collections of vectors deployed together for ANN search. As a result of various blooming applications on the cloud, various data sources are in many different formats and diverse places. Unlike traditional databases that require structured or semi-structured data that must convey a few restrictions and formats, vector databases are purpose-designed for storing the deep learning embedding of various unstructured data that emerged in real-world applications. On the other hand, distinct from traditional DBMS that searches for exact values within databases, vector databases heavily rely on approximate nearest neighbor (ANN) search for vectors, which searches approximate top-k nearest distance neighbors within

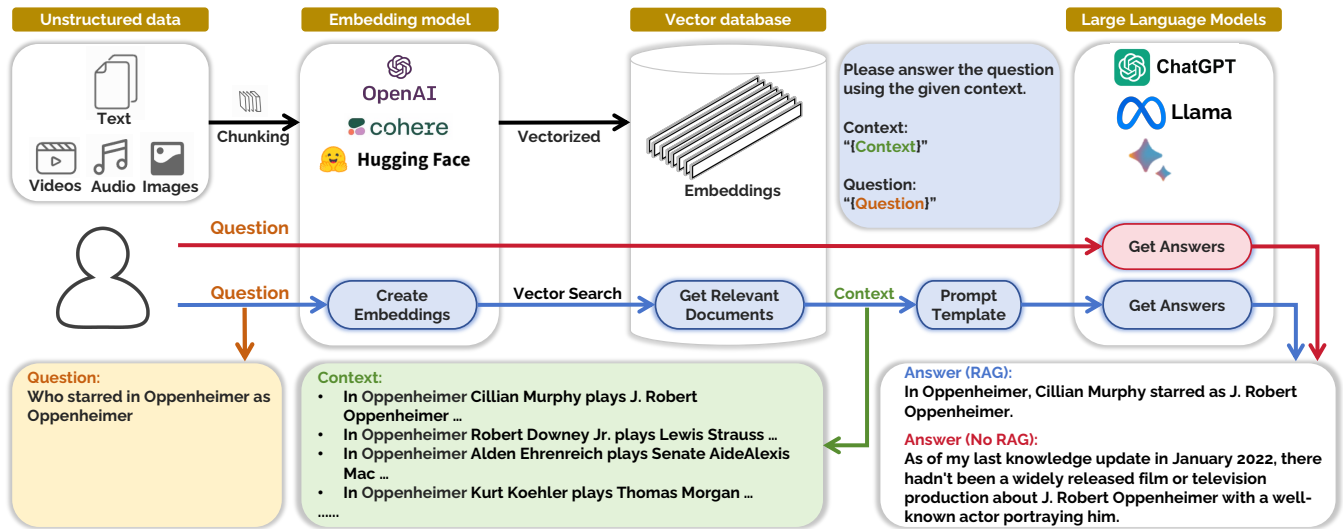


Figure 1: Sample: A sample RAG framework that uses vector databases.

the high-dimensional base vector data space that do not necessarily require exact matches.

With the unstructured base data embedded into vectors with high dimensionality, calculating k-nearest neighbors of a given query vector data can be expensive, since it requires distance computation to every point in the dataset and maintaining the top-k results. Such operation would direct to a time complexity of $O(dN + N \log k)$, where d is the dimensionality and N is the number of vectors, for searching top-k results using pair-wise distance calculation and a heap to keep top-k results.

This calls for a more efficient search technique with satisfying accuracy, the ANN Benchmarks¹ has showcased the great performance gap between brute-force ANN search and index-enhanced ANN search. Since brute force search is both time-consuming and computationally expensive, vector indexing can solve both of these problems, which optimized for ANN search within VDBMS include tree-based [Muja and Lowe, 2009; Tao *et al.*, 2009], hash-based [Andoni and Indyk, 2008], Product Quantization (PQ) [Jégou *et al.*, 2011], and graph-based methods [Malkov and Yashunin, 2018]. Among all these indexes, the graph-based Hierarchical Navigable Small Worlds (HNSW) provides state-of-the-art performance and capability with great universality and is widely used within most vector database management systems, including the examples we mentioned in Table 3.

Traditional full-text search engines, such as Elastic Search and Amazon OpenSearch, are based on term frequency metrics like BM25, which has proved practical for many search applications. However, such search techniques require significant investment in time and expertise to tune them to account for the meaning or relevance of the terms searched. On the other hand, vector databases successfully solved the aforementioned problem, thus the emergence of vector databases (as shown in Table 3) has greatly influenced machine learn-

ing systems and their multi-modal applications. An intuitive application using vector search is searching images with an image on search engines built on vector databases [Wang and et al., 2021] like Google Image Search², which uses one input image to find similar images on the internet. While vector databases boast their unique search capability and efficiency for unstructured data, they are just a backend factor for manifold applications. To maximize their abilities, this leads us to an interesting question, since LLMs use embeddings to represent text as vectors, can developers combine vector databases and LLMs to overcome the aforementioned challenges that inherit in LLM applications? The answer is yes.

As a kind of database encapsulated with vector search in vector data that represents real-world information within high dimensionalities, vector databases are well-capable for retrieval applications [Asai and et al., 2023] incorporating LLMs because LLM applications are generally read-intensive, not requiring many write-related changes, especially data deletes [Pan *et al.*, 2023]. On the other hand, vector databases can efficiently manage and warehouse vector data required and generated by LLMs, thus providing a solid data cornerstone for both LLMs and their applications. While LLMs are often limited by domain knowledge that cannot be uploaded or distributed due to security and privacy concerns, domain knowledge is often unstructured data that can easily be embedded into vector databases for efficient local retrieval and further integration with generative AIs. Moreover, the computing and storage resources of vector databases are way cheaper than LLMs, since it does not require costly GPUs and TPUs, thus achieving a cost-effective way of fast retrieval and durable (non-volatile) storage.

¹<https://ann-benchmarks.com/>

²<https://cloud.google.com/vertex-ai/docs/vector-search/overview>

Category	Supported Data type	Examples
Relational Databases (RDBMS)	Structured	MySQL, PostgreSQL, Oracle
Document Databases	Semi-Structured	MongoDB, Couchbase
Time-Series Databases	Structured	InfluxDB, TimescaleDB
Graph Databases	Structured	Neo4j, Amazon Neptune
Vector Databases (VDBMS)	Unstructured	Pinecone, Milvus, Jina, Qdrant

Table 2: Different kinds of mainstream business-level databases.

Name (Version with year)	Supported Data		Supported Query		Vector Index	
	Vec. Dim.	Type	Filter	Multi-Vector	Graph	IVF
ChormaDB (2022)	1536	Vec.	✓		✓	✓
Manu (2022)	32768	Vec.	✓	✓	✓	✓
Milvus (2021)	32768	Vec.	✓	✓	✓	✓
Pinecone (2019)	20000	Vec.	✓	✓	✓	✓
Weaviate (2019)	65535	Vec.	✓	✓	✓	✓
Qdrant (2021)	4096	Vec.	✓	✓	✓	✓
Amazon OpenSearch (v2.9, 2023)	16,000	Ftx.	✓	✓	✓	✓
Elastic Search (v8.0, 2022)	4096	Ftx.	✓	✓	✓	✓
AnalyticDB-V (2020)	≥ 512	Rel.	✓		✓	✓
PostgreSQL-pgvector (2021)	2000	Rel. + Ftx.	✓	✓	✓	✓
MongoDB Atlas (v6.0, 2023)	2048	NoSQL	✓		✓	
MyScale (2023)	1536	Rel.	✓		✓	

Table 3: Comparison of mainstream all-level databases supporting vector data (the version number represents the first version that supports vector search and its release date)

3.2 Retrieval-Augmented Generation (RAG): VecDB as an External Knowledge Base

Development of RAG Paradigm

As the release of ChatGPT casts spotlight on LLMs to the public world, there is an increasing need of using AI chatbots as query and retrieval agents. But simply loading users’ private data as input to LLMs is found to be incompetent in real-world applications. Because LLMs have been constrained by their limited token counts and the high costs associated with training and fine-tuning for every alterations of data, especially pronounced when dealing with personalized or business-specific responses that require real-time data updates. To address such concerns and need, retrieval-augmented generation (RAG) emerges as a novel solution that addresses the challenges faced by LLMs in integrating and processing large and dynamic data in external databases, where vector databases offer a solution by acting as external memory for LLMs. They allow for the segmentation of private data by converting them into vectors and storing these vectors efficiently for quick retrieval process. Integrating with vector databases enables LLMs to access and synthesize enormous amount of data without the need for constant re-training, thereby overcoming their inherent limitations.

The concept of RAG is devised to be a paradigm, and a common workflow of RAG is illustrated in Figure 1. There are essentially 3 main parts to complete one full run of the systems: data storage, retrieval, and generation.

Data storage means establishing reliable external memory like vector databases, and there are detailed recipes for this process [Han *et al.*, 2023]. It starts with the data preprocessing, during which the original data is collected, cleaned, integrated, transformed, normalized and standardized. The processed data is chunked into smaller segments because of the contextual limit of LLMs. These segments of data are then converted by an embedding model into vectors, the semantic representations of the data, which are stored in a vector database and will be used for vector search in later steps. A well-developed vector database will properly index the data and optimize the retrieval process. The retrieval part starts by a user asking a question in the form of prompts to the same embedding model, which has generated the vector representation of the stored data, and gaining the vector embeddings of the question. The next step of the process is vector searching inside the vector databases. The vector search is essentially computing similarity scores among vectors, and the database then identifies and retrieves the data segments that have the highest similarity scores (top K in most RAG systems) compared to the query vector. These retrieved segments are then converted back from their vector format to their original form, which is typically text of documents. In the generation part, LLMs are involved to generate the final answers. The retrieved documents, along with the user’s question, are incorporated into a specifically chosen and designed prompt template. This template selection is based on the task type and aims to effectively merge the context with the question to

form a coherent prompt. The selected LLM is provided with the prompt and generates the final answer.

RAG with Vector Databases

In the prototype of the RAG paradigm, we use such an idea to overcome the hallucination problems of LLMs by providing domain-specific data with accurate instruction, where the vector database serves as an external knowledge base to warehouse domain-specific data, then LLMs can easily handle massive data owned by users.

Even though LLMs such as ChatGPT now have user-specified GPTs for specific uses, with just prompt engineering, their knowledge bases are still limited to the training data provided by OpenAI. However, by using a vector database, users can pre-filter whatever they want it to look at, whereas that is difficult with prompt-engineering GPT assistants.

Expanding Horizons: Multi-modal Integration and Retrieval Innovations in RAG Systems

Numerous studies have been undertaken to enhance the performance and functionality of RAG systems.

Datasets RAG was originally developed and evaluated on text-based question-answering (QA) tasks. There are numerous datasets and benchmarks available for evaluating the performance of various RAG systems, where different prompt templates need to be designed for different tasks. Therefore, we classified datasets based on the type of tasks and summarized them below.

- **Single-hop QA** involves simple questions where the answer can be obtained through a single-step reasoning process, including TriviaQA [Joshi and et al., 2017], Natural Questions [Kwiatkowski and et al., 2019], WebQuestions [Berant and et al., 2013]. **Multi-hop QA** requires multiple intermediate reasoning steps to reach the final answer. For example, for a question like "In which state did Obama go to high school?", one needs to first infer which school Obama attended and then determine the location of that school to obtain the final answer. Datasets for multi-hop QA include HotpotQA [Yang and et al., 2018], 2WikiMultiHopQA [Ho and et al., 2020], MuSiQue [Trivedi and et al., 2022], StrategyQA [Geva and et al., 2021]. In **Multiple-choice QA**, the model does not need to generate an answer independently but rather select the correct answer from provided options. Multiple-choice QA datasets include MMLU [Hendrycks and et al., 2020], SQuAD [Rajpurkar and et al., 2016]. **Open-domain QA** involves the task of searching for answers to information-seeking queries within a large pool of knowledge sources. Relevant datasets for open-domain QA include MS MARCO [Nguyen and et al., 2016], PopQA [Mallen and et al., 2022], CommonsenseQA [Talmor and et al., 2018].
- **Fact-checking** requires the model to first select relevant sentences from a given document as evidence and then validate statements based on that evidence. Datasets include FEVER [Thorne and et al., 2018], FM2 [Eisenschlos and et al., 2021].

- **Embedding evaluation** comprehensively assess all essential capabilities of text embeddings, including BEIR [Thakur and et al., 2021], C-MTEB [Xiao and et al., 2023].

Multimodality of RAG RAG has now evolved to handle a wide range of data types by lending the power of multimodal models.

The impressive achievements of LLMs have inspired significant advancements in vision-and-language research. DALL-E from OpenAI introduced a Transformer-based approach for converting text to images, treating images as sequences of discrete tokens. Subsequent improvements in the text-to-image area [Zhang *et al.*, 2023] have been achieved through methods like model scaling, pretraining, and enhanced image quantization models. BLIP-2 [Li *et al.*, 2023] uses static image encoders with LLMs for efficient visual language pre-training, facilitating direct image-to-text transformations. Flamingo [Alayrac *et al.*, 2022] presented a visual language model for text generation, showcasing remarkable adaptability and leading performance across various vision-and-language tasks. CM3 [Aghajanyan *et al.*, 2022] trained a randomly masked model on a large HTML corpus, and showed that the model is capable of generating images and text. FROMAGE [Koh *et al.*, 2023] gains robust multimodal capabilities for few-shot learning solely from image-caption pairs, unlike other models that necessitate large-scale, interwoven image-text data from the websites.

To import speech data to RAG systems, Wav2Seq [Wu *et al.*, 2022] allows for efficient pre-training without the need for transcriptions, using techniques like k-means clustering and byte-pair encoding to generate pseudo subwords from speech. The Large Language and Speech Model (LLaSM) [Shu *et al.*, 2023] is an end-to-end trained, large multi-modal speech-language model equipped with cross-modal conversational skills and proficient in understanding and responding to combined speech-and-language directives. Videos are also made available to certain types of RAG systems. Vid2Seq [Yang *et al.*, 2023] enhances language models with specific temporal indicators for predicting event limits and textual descriptions in a single output sequence.

Retrieval Optimizations of RAG To better harness the knowledge from various types of data, kNN-LMs [Khandelwal *et al.*, 2020] explores how incorporating nearest neighbor search into language models can enhance their ability to generalize by effectively leveraging memorized examples. EASE [Nishikawa *et al.*, 2022] is distinctive in its use of entities as a strong indicator of text semantics, providing rich training signals for sentence embedding. In-context RALM [Ram *et al.*, 2023] proves that with the language model architecture unchanged, simply appending grounding documents to the input will improve the performance. SURGE [Kang *et al.*, 2023] enhances dialogue generation by incorporating context-relevant sub-graphs from a knowledge graph. Another work that combines knowledge graphs and LLMs is RET-LLM [Modarressi *et al.*, 2023], which is designed to equip large language models with a general write-read memory unit. These studies have focused on retrieval granularity and data structuring levels, with coarse granularity providing

more, but less precise, information. Conversely, structured text retrieval offers detailed information at the cost of efficiency.

To utilize both internal knowledge and external resources, SKR [Yu *et al.*, 2023] improves LLMs’ ability by enabling them to assess what they know and identify when to seek external information to answer questions more effectively. Selfmem [Cheng *et al.*, 2023] enhances retrieval-augmented text generation by creating an unbounded memory pool using the model’s own output and selecting the best output as memory for subsequent rounds. FLARE [Jiang *et al.*, 2023] uses predictions of upcoming sentences to anticipate future content and retrieve relevant documents for regenerating sentences, especially when they contain low-confidence tokens. Atlas [Izacard *et al.*, 2022] demonstrates impressive performance in tasks like question-answering and fact-checking, outperforming much larger models despite having fewer parameters. Many other works like these also aim to make RAG system more efficient and competent.

3.3 VecDB as a Reliable Memory of GPTs

Oblivion When using LLM Q&A applications like ChatGPT, LLMs are likely to completely forget the content and information of your previous conversation, even within the same chat tab.

The integration of Large Language Models (LLMs) such as the Generative Pre-trained Transformer (GPT) with vector databases (VecDB) heralds a transformative leap in the field of information retrieval and artificial intelligence. This subsection delves into the compelling synergy between VecDB as a persistent, reliable memory layer and the dynamic, contextually aware computation layer provided by LLMs. In recent works of vector databases that integrate with LLMs, [Zhang and *et al.*, 2023] showcases that vector databases’ capability of storing vectors for GPT’s memory.

Vector databases serve as a robust memory layer that addresses one of the intrinsic limitations of LLMs: the static nature of their knowledge. While LLMs excel in generating human-like text based on patterns learned during training, they cannot inherently update their knowledge base dynamically. VecDBs bridge this gap by offering a storage solution that can be continually updated with new information, ensuring that the LLM’s responses are informed by the most current and relevant data available.

The VecDB and LLM combination brings forth a synergy where the LLM provides context and understanding for user queries, while the VecDB offers a precise mechanism for storing and retrieving the relevant vectors. This collaborative approach allows for more accurate, relevant, and efficient responses to complex queries, which would be challenging for either system to address independently.

A VecDB integrated with an LLM facilitates real-time learning and adaptation. As new data is ingested into the VecDB, the LLM can immediately leverage this updated repository to refine its responses. This capability is pivotal for applications requiring up-to-the-minute accuracy, such as financial analysis, news dissemination, and personalized recommendations.

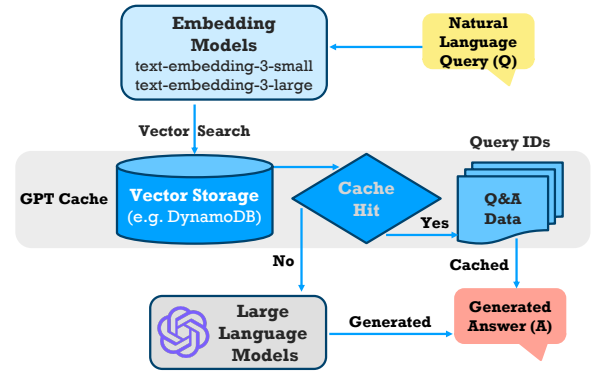


Figure 2: An overview of semantic cache for GPTs that utilizes vector database

3.4 VecDB as a Cost-effective Semantic Cache

The combination of vector databases and LLM not only facilitates the profound application of LLM with RAG but also provides a new frontier for cost-effective end-to-end LLM applications:

Outrageous API Costs LLM-based chatbots and agent systems heavily rely on LLM’s output from API vendors, considerable repeated or similar inquiries may lead to outrageous API costs.

API Bandwidth Limitations Such chatbots and agent systems could also experience bursty inquiries workload that may drown the system’s bandwidth with explosive API calls coming within seconds, leading to the system’s outage and reconfiguration.

One of the primary benefits of integrating vector databases with LLMs is the significant reduction in data operational costs. GPT-Cache, as shown in Figure 2, for instance, stores responses to previously asked queries, and works as a cache before calling LLM APIs. This caching mechanism means that the system doesn’t really need to have API calls to wait for generated responses from scratch every time, reducing the number of costly API calls to the LLM. Moreover, this approach also speeds up the response time, enhancing user experience.

Vector databases enhance the LLMs’ ability to retrieve and utilize relevant information by indexing vast amounts of previous Q&A data and mapping them into a vector space, instead of caches in computer systems that require exact hash-match, this allows more precise semantic matching of queries with existing knowledge and results in responses that are not only generated based on the LLM’s training data but also informed by the most relevant and recent information available in the vector database.

4 Discussion: Challenges and Future Work

Are vector searches all you need?

Although vector databases offer a cost-efficient modality for information retrieval within LLM frameworks, their utility in traditional relational database operations remains limited. Specifically, vector-based search methodologies are often not

optimized for operations such as post-query filtering, comprehensive full-text searches, and nuanced keyword search mechanisms that are fundamental to conventional database systems. This distinction underscores a potential gap in the functional alignment of vector databases with established data retrieval paradigms, necessitating the development of hybrid search algorithms that can seamlessly integrate vector search with traditional relational database capabilities.

VDBMS with multi-modality

While uni-modal data may lack the prospect of providing informative and more contextually appropriate search results, the multi-modal data and its hybrid processing also present a non-trivial challenge for vector databases' storage and retrieval [Wang and et al., 2023], since integrating and merging multi-modal data require efficient preprocessing with multi-modal encoders, multi-modal storage indexes, but also weight assignment and multi-modal data fusion.

Data preprocessing

While **text embedding** is considered efficient for processing long text and its retrieval, it comes with a dazzling challenge for building every text-based knowledge database. It is observed that vector retrieval would have fundamentally better performance when precise chunks of text with clear meaning are applied, this raises an interesting question: how to have a proper (probably unified) embedding methodology for every raw text?

On the other hand, **dimension reduction** for vector data is also considered important due to *the curse of dimensionality* theory. The current vector data dimension may be not cost-effective, as vectors are large: e.g., a 1536 dimensional vector is about 6 kilobytes, with scaling up to 1 billion, the data size would be 6 terabytes. This requires a more efficient vector data dimension reduction algorithm that matches vector databases and search algorithms.

Knowledge conflict

Conflicting knowledge from the same or different knowledge bases presents a non-trivial challenge for both humans and LLMs to distinguish the correct piece of knowledge. This conflict becomes particularly pronounced when integrating multiple knowledge bases, each potentially carrying its own biases or inaccuracies. Resolving such conflicts requires robust conflict resolution strategies that can assess the reliability of sources and the context of data to determine the most accurate information.

Data management systems for LLMs with multi-tenancy

This challenge is particularly in maintaining the isolation and security of distinct tenants' data while ensuring efficiency, where isolation is crucial for privacy and security, yet it must be balanced with the need for resource sharing to ensure cost-effectiveness.

5 Conclusion

In this paper, we present a systematic review of recent advances in combinations of LLMs and vector databases. We also introduce recent VecDB+LLMs applications with distinct prototypes that categorize existing works from various

perspectives and interdisciplinary studies. Our study also demonstrates both the research and engineering challenges in this fast-growing field and suggests directions for future work.

References

- [Achiam and et al., 2023] Josh Achiam and Steven Adler et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
- [Aghajanyan et al., 2022] Armen Aghajanyan, Bernie Huang, Candace Ross, Vladimir Karpukhin, Hu Xu, Naman Goyal, Dmytro Okhonko, Mandar Joshi, Gargi Ghosh, Mike Lewis, and Luke Zettlemoyer. Cm3: A causal masked multimodal model of the internet, 2022.
- [Alayrac et al., 2022] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andrew Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning, 2022.
- [Andoni and Indyk, 2008] Alexandr Andoni and Piotr Indyk. Near-Optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions. *Communications of the ACM*, 51(1):117–122, 2008.
- [Asai and et al., 2023] Akari Asai and Sewon Min et al. Acl 2023 tutorial: Retrieval-based language models and applications. *ACL 2023*, 2023.
- [Bender et al., 2021] Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency, FAccT '21*, page 610–623, New York, NY, USA, 2021. Association for Computing Machinery.
- [Berant and et al., 2013] Jonathan Berant and Andrew Chou et al. Semantic parsing on freebase from question-answer pairs. In *Proc. 2013 Conf. on Empirical Methods in Natural Language Processing*, pages 1533–1544, 2013.
- [Brown and et al., 2020] Tom Brown and Benjamin Mann et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.
- [Cheng et al., 2023] Xin Cheng, Di Luo, Xiuying Chen, Lemao Liu, Dongyan Zhao, and Rui Yan. Lift yourself up: Retrieval-augmented text generation with self memory, 2023.
- [Chowdhery and et al., 2023] Aakanksha Chowdhery and Sharan Narang et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

- [Chung *et al.*, 2014] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.
- [Clark and et al., 2020] Kevin Clark and Minh-Thang Luong et al. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*, 2020.
- [Devlin *et al.*, 2018] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [Dodge and et al., 2022] Jesse Dodge and Taylor Prewitt et al. Measuring the carbon intensity of ai in cloud instances, 2022.
- [Dong *et al.*, 2021] Hongyuan Dong, Jiaqing Xie, Zhi Jing, and Dexin Ren. Variational autoencoder for anti-cancer drug response prediction, 2021.
- [Du and et al., 2022] Nan Du and Yanping Huang et al. Glam: Efficient scaling of language models with mixture-of-experts. In *Int. Conf. on Machine Learning*, pages 5547–5569. PMLR, 2022.
- [Eisenschlos and et al., 2021] Julian Martin Eisenschlos and Bhuwan Dhingra et al. Fool me twice: Entailment from wikipedia gamification. *arXiv preprint arXiv:2104.04725*, 2021.
- [Fan and et al., 2021] Angela Fan and Shruti Bhosale et al. Beyond english-centric multilingual machine translation. *Journal of Machine Learning Research*, 22(107):1–48, 2021.
- [Ganguli *et al.*, 2022] Deep Ganguli, Liane Lovitt, Jackson Kernion, Amanda Askell, Yuntao Bai, Saurav Kadavath, Ben Mann, Ethan Perez, Nicholas Schiefer, Kamal Ndousse, Andy Jones, Sam Bowman, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Nelson Elhage, Sheer El-Showk, Stanislav Fort, Zac Hatfield-Dodds, Tom Henighan, Danny Hernandez, Tristan Hume, Josh Jacobson, Scott Johnston, Shauna Kravec, Catherine Olson, Sam Ringer, Eli Tran-Johnson, Dario Amodei, Tom Brown, Nicholas Joseph, Sam McCandlish, Chris Olah, Jared Kaplan, and Jack Clark. Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned, 2022.
- [Geva and et al., 2021] Mor Geva and Daniel Khashabi et al. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361, 2021.
- [Han *et al.*, 2023] Yikun Han, Chunjiang Liu, and Pengfei Wang. A comprehensive survey on vector database: Storage and retrieval technique, challenge, 2023.
- [He and et al., 2016] Kaiming He and Xiangyu Zhang et al. Deep residual learning for image recognition. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pages 770–778, 2016.
- [He and et al., 2020] Pengcheng He and Xiaodong Liu et al. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*, 2020.
- [Hendrycks and et al., 2020] Dan Hendrycks and Collin Burns et al. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.
- [Ho and et al., 2020] Xanh Ho and Anh-Khoa Duong Nguyen et al. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*, 2020.
- [Hoffmann and et al., 2022] Jordan Hoffmann and Sebastian Borgeaud et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.
- [Huang *et al.*, 2023] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions, 2023.
- [Izacard *et al.*, 2022] Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Atlas: Few-shot learning with retrieval augmented language models, 2022.
- [Jégou *et al.*, 2011] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product Quantization for Nearest Neighbor Search. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 33(1):117–128, 2011.
- [Ji and et al., 2023] Ziwei Ji and Nayeon Lee et al. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.
- [Jiang *et al.*, 2023] Zhengbao Jiang, Frank F. Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation, 2023.
- [Joshi and et al., 2017] Mandar Joshi and Eunsol Choi et al. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*, 2017.
- [Kang *et al.*, 2023] Minki Kang, Jin Myung Kwak, Jinheon Baek, and Sung Ju Hwang. Knowledge graph-augmented language models for knowledge-grounded dialogue generation, 2023.
- [Kaplan and et al., 2020] Jared Kaplan and Sam McCandlish et al. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.
- [Khandelwal *et al.*, 2020] Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models, 2020.
- [Koh *et al.*, 2023] Jing Yu Koh, Ruslan Salakhutdinov, and Daniel Fried. Grounding language models to images for multimodal inputs and outputs. 2023.

- [Kwiatkowski and et al., 2019] Tom Kwiatkowski and Jen-nimaria Palomaki et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.
- [Lample and Conneau, 2019] Guillaume Lample and Alexis Conneau. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*, 2019.
- [Lan and et al., 2019] Zhenzhong Lan and Mingda Chen et al. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.
- [Lee et al., 2022] Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. Deduplicating training data makes language models better, 2022.
- [Lewis and et al., 2019] Mike Lewis and Yinhan Liu et al. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.
- [Li and et al., 2023] Yingji Li and Mengnan Du et al. A survey on fairness in large language models. *arXiv preprint arXiv:2308.10149*, 2023.
- [Li et al., 2023] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models, 2023.
- [Liu and et al., 2024] Chunjiang Liu and Yikun Han et al. A community detection and graph-neural-network-based link prediction approach for scientific literature. *Mathematics*, 12(3):369, 2024.
- [Liu et al., 2019] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.
- [Malkov and Yashunin, 2018] Yury A. Malkov and Dmitry A. Yashunin. Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 42(4):824–836, 2018.
- [Mallen and et al., 2022] Alex Mallen and Akari Asai et al. When not to trust language models: Investigating effectiveness and limitations of parametric and non-parametric memories. *arXiv preprint arXiv:2212.10511*, 7, 2022.
- [Modarressi et al., 2023] Ali Modarressi, Ayyoob Imani, Mohsen Fayyaz, and Hinrich Schütze. Ret-llm: Towards a general read-write memory for large language models, 2023.
- [Muja and Lowe, 2009] Marius Muja and David G Lowe. Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration. *VISAPP*, 2(331-340):2, 2009.
- [Musser, 2023] Micah Musser. A cost analysis of generative language models and influence operations, 2023.
- [Nguyen and et al., 2016] Tri Nguyen and Mir Rosenberg et al. Ms marco: A human generated machine reading comprehension dataset. *Choice*, 2640:660, 2016.
- [Nishikawa et al., 2022] Sosuke Nishikawa, Ryokan Ri, Ikuya Yamada, Yoshimasa Tsuruoka, and Isao Echizen. EASE: Entity-aware contrastive learning of sentence embedding. In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz, editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3870–3885, Seattle, United States, July 2022. Association for Computational Linguistics.
- [Onoe et al., 2022] Yasumasa Onoe, Michael Zhang, Eunsol Choi, and Greg Durrett. Entity cloze by date: What LMs know about unseen entities. In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz, editors, *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 693–702, Seattle, United States, July 2022. Association for Computational Linguistics.
- [Pan et al., 2023] James Jie Pan, Jianguo Wang, and Guoliang Li. Survey of vector database management systems. *arXiv preprint arXiv:2310.14021*, 2023.
- [Penedo et al., 2023] Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The refinedweb dataset for falcon llm: Outperforming curated corpora with web data, and web data only, 2023.
- [Radford and et al., 2019] Alec Radford and Jeffrey Wu et al. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019.
- [Rae and et al., 2021] Jack W Rae and Sebastian Borgeaud et al. Scaling language models: Methods, analysis and insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.
- [Raffel et al., 2020] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.
- [Rajpurkar and et al., 2016] Pranav Rajpurkar and Jian Zhang et al. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.
- [Ram et al., 2023] Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331, 2023.
- [Schaeffer, 2023] Rylan Schaeffer. Pretraining on the test set is all you need. *arXiv preprint arXiv:2309.08632*, 2023.

- [Shi *et al.*, 2015] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems*, 28, 2015.
- [Shu *et al.*, 2023] Yu Shu, Siwei Dong, Guangyao Chen, Wenhao Huang, Ruihua Zhang, Daochen Shi, Qiqi Xiang, and Yemin Shi. Llasmm: Large language and speech model, 2023.
- [Smith and et al., 2022] Shaden Smith and Mostofa Patwary et al. Using deepspeed and megatron to train megatron-turing nlge 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*, 2022.
- [Strubell and et al., 2019] Emma Strubell and Ananya Ganesh et al. Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243*, 2019.
- [Su and et al., 2022] Yongye Su and Qian Liu et al. Yolo-10: A transformer-based yolo segmentation model for breast mass detection and segmentation in digital mammograms. *Computer Methods and Programs in Biomedicine*, 221:106903, 2022.
- [Talmor and et al., 2018] Alon Talmor and Jonathan Herzig et al. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*, 2018.
- [Tao *et al.*, 2009] Yufei Tao, Ke Yi, Cheng Sheng, and Panos Kalnis. Quality and efficiency in high dimensional nearest neighbor search. In *Proceedings of the International Conference on Management of Data (SIGMOD)*, SIGMOD ’09, page 563–576, New York, NY, USA, 2009. Association for Computing Machinery.
- [Thakur and et al., 2021] Nandan Thakur and Nils Reimers et al. Beir: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. *arXiv preprint arXiv:2104.08663*, 2021.
- [Thoppilan and et al., 2022] Romal Thoppilan and Daniel De Freitas et al. Llama: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*, 2022.
- [Thorne and et al., 2018] James Thorne and Andreas Vlachos et al. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*, 2018.
- [Touvron and et al., 2023] Hugo Touvron and Thibaut Lavril et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.
- [Trivedi and et al., 2022] Harsh Trivedi and Niranjan Balasubramanian et al. Musique: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.
- [Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- [Venkit *et al.*, 2023] Pranav Narayanan Venkit, Sanjana Gautam, Ruchi Panchanadikar, Ting-Hao ‘Kenneth’ Huang, and Shomir Wilson. Nationality bias in text generation, 2023.
- [Wang and et al., 2021] Jianguo Wang and Xiaomeng Yi et al. Milvus: A purpose-built vector data management system. In *Proc. of the 2021 Int. Conf. on Management of Data*, pages 2614–2627, 2021.
- [Wang and et al., 2023] Mengzhao Wang and Xiangyu Ke et al. Must: An effective and scalable framework for multimodal search of target modality. *arXiv preprint arXiv:2312.06397*, 2023.
- [Wei and et al., 2022] Jason Wei and Yi Tay et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.
- [Workshop and et al., 2022] BigScience Workshop and Teven Le Scao et al. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*, 2022.
- [Wu and et al., 2023] Shijie Wu and Ozan Irsoy et al. Bloomberggpt: A large language model for finance. *arXiv preprint arXiv:2303.17564*, 2023.
- [Wu *et al.*, 2022] Felix Wu, Kwangyoun Kim, Shinji Watanabe, Kyu Han, Ryan McDonald, Kilian Q. Weinberger, and Yoav Artzi. Wav2seq: Pre-training speech-to-text encoder-decoder models using pseudo languages, 2022.
- [Xiao and et al., 2023] Shitao Xiao and Zheng Liu et al. C-pack: Packaged resources to advance general chinese embedding. *arXiv preprint arXiv:2309.07597*, 2023.
- [Xue and et al., 2020] Linting Xue and Noah Constant et al. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*, 2020.
- [Yang and et al., 2018] Zhilin Yang and Peng Qi et al. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*, 2018.
- [Yang *et al.*, 2023] Antoine Yang, Arsha Nagraani, Paul Hongsuck Seo, Antoine Miech, Jordi Pont-Tuset, Ivan Laptev, Josef Sivic, and Cordelia Schmid. Vid2seq: Large-scale pretraining of a visual language model for dense video captioning, 2023.
- [Yao *et al.*, 2024] Yifan Yao, Jinhao Duan, Kaidi Xu, Yuanfang Cai, Zhibo Sun, and Yue Zhang. A survey on large language model (llm) security and privacy: The good, the bad, and the ugly, 2024.
- [Yosinski and et al., 2014] Jason Yosinski and Jeff Clune et al. How transferable are features in deep neural networks? *Advances in Neural Information Processing Systems*, 27, 2014.
- [Yu *et al.*, 2023] Wenhao Yu, Dan Iter, Shuohang Wang, Yichong Xu, Mingxuan Ju, Soumya Sanyal, Chenguang Zhu, Michael Zeng, and Meng Jiang. Generate rather than retrieve: Large language models are strong context generators, 2023.

- [Zaheer and et al., 2020] Manzil Zaheer and Guru Guruganesh et al. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33:17283–17297, 2020.
- [Zaremba *et al.*, 2014] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*, 2014.
- [Zeng and et al., 2022] Aohan Zeng and Xiao Liu et al. Glm-130b: An open bilingual pre-trained model. *arXiv preprint arXiv:2210.02414*, 2022.
- [Zhang and et al., 2022] Susan Zhang and Stephen Roller et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.
- [Zhang and et al., 2023] Yi Zhang and Zhongyang Yu et al. Long-term memory for large language models through topic-based vector database. In *2023 International Conference on Asian Language Processing (IALP)*, pages 258–264, 2023.
- [Zhang *et al.*, 2023] Chenshuang Zhang, Chaoning Zhang, Mengchun Zhang, and In So Kweon. Text-to-image diffusion models in generative ai: A survey, 2023.