

محمد الغزواني

Chapter2 - AskMe Chatbot.pdf

 Midocean University

Document Details

Submission ID

trn:oid::2945:405082362

Submission Date

Aug 6, 2026, 7:40 PM GMT+3

Download Date

Aug 9, 2026, 4:13 AM GMT+3

File Name

Chapter2 - AskMe Chatbot.pdf

File Size

1.8 MB

28 Pages

5,593 Words

35,494 Characters

*% detected as AI

AI detection includes the possibility of false positives. Although some text in this submission is likely AI generated, scores below the 20% threshold are not surfaced because they have a higher likelihood of false positives.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Disclaimer

Our AI writing assessment is designed to help educators identify text that might be prepared by a generative AI tool. Our AI writing assessment may not always be accurate (it may misidentify writing that is likely human generated as AI generated and likely AI generated as human generated) so it should not be used as the sole basis for adverse actions against a student. It takes further scrutiny and human judgment in conjunction with an organization's application of its specific academic policies to determine whether any academic misconduct has occurred.



AskMe Educational RAG System

Prepared By

Name: Amro Ismail Mohamed

ID No.: 261001995

Name: Mohamed Alghazwani

ID No.: 261002006

Submitted in fulfillment of the requirements for the degree Master of
Science in Artificial Intelligence

At

Collage of Informatics

Midocean University

UAE

Supervisor Name: Hager Saleh

Collage of Informatics, Midocean University, Comoros

Email: hager.saleh@midocean.com

Contents

Chapter 2: Background and Related Work	1
2.1 Theoretical Background	1
2.1. Natural Language Processing (NLP)	1
2.1.1. Basic NLP Tasks and Taxonomy.....	1
2.1.3 End-to-End NLP Pipeline Architecture	1
2.2. Text Pre-processing & Normalization	2
2.3. Text Representation (Feature Extraction)	3
2.4. Deep Learning Models	5
2.5. Large Language Models (LLMs).....	7
2.2 Summary of Related Work	17
Paper 1:	17
Paper 2:	17
Paper 3:	18
Paper 4:	18
Paper 5:	19
Paper 6:	19
Paper 7:	19
Paper 8:	20
2.3 References.....	23

Chapter 2: Background and Related Work

2.1 Theoretical Background

2.1.1. Natural Language Processing (NLP)

Natural Language Processing (NLP) is an interdisciplinary area of computer science, artificial intelligence and computational linguistics. It is about computers' ability to process, analyze and generate human language in a semantically rich way. Modern NLP tasks vary from simple text classification to complex generative and reasoning workflows [Chowdhury, Gobinda G., 2003].

2.1.1.1. Basic NLP Tasks and Taxonomy

Modern NLP tasks can be roughly classified into the following distinct functional areas:

- **Natural Language Understanding (NLU):** Involves the assessment of semantic intent, syntactic structure, and contextual context from text. Main applications are text classification, sentiment analysis, named entity recognition (NER) and intent detection [Chowdhury, Gobinda G., 2003; Manning, Christopher D., et al., 2008].
- **Natural Language Generation (NLG):** Concentrates on creating coherent, grammatically correct and contextually relevant human text. Applications of interest include abstractive summarization, dialogue generation and automated report generation [Chowdhury, Gobinda G., 2003].
- **Sequence-to-Sequence Processing:** It processes an input sequence and produces a corresponding output sequence in a different domain or modality, e.g., neural machine translation, speech recognition [Cho, Kyunghyun, et al., 2014; Vaswani, Ashish, et al., 2017].

2.1.1.3 End-to-End NLP Pipeline Architecture

The standard NLP workflow translates unstructured human discourse to numerical representations as below:

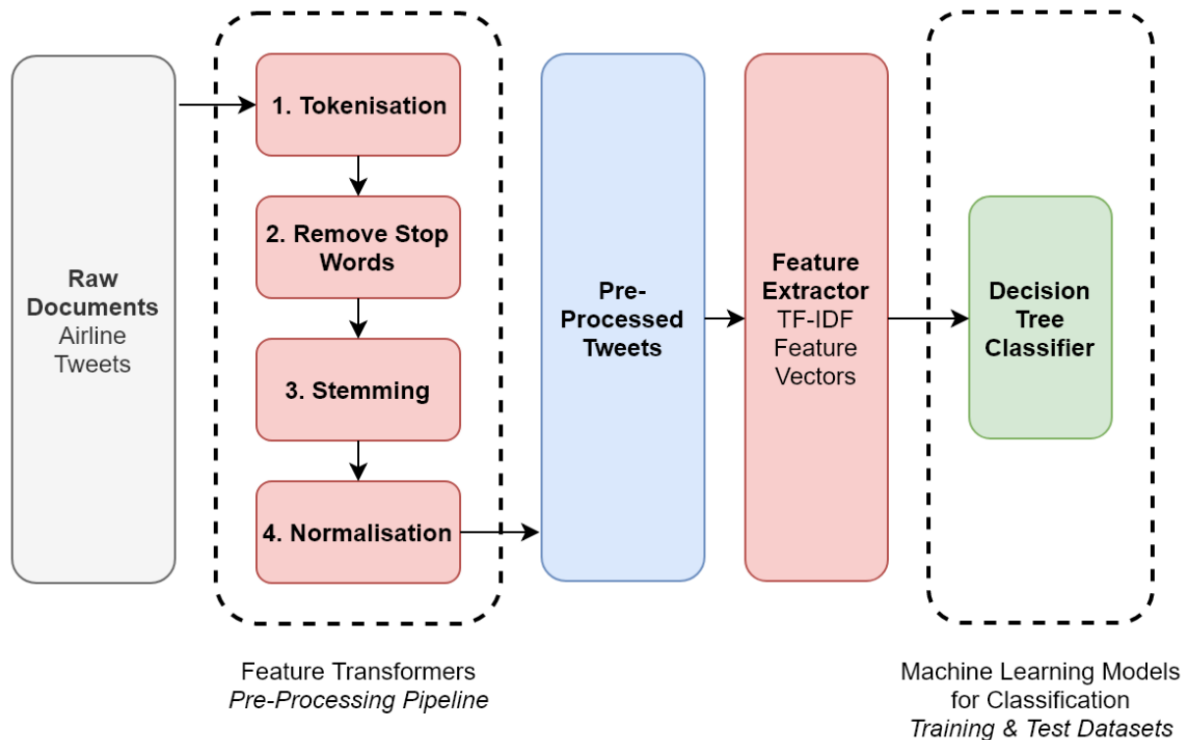


Figure 2.1.3.1: End-to-end NLP pipeline

[Machine Learning with Apache Spark Quick Start Guide]

2.2. Text Pre-processing & Normalization

Raw text data from real-world environments is inherently noisy; Pre-processing of text constitutes a mandatory preparatory phase of the pipeline that purifies raw corpora, reduces vocabulary redundancy and ensures uniform input structures for downstream machine learning algorithms [Kaur, Jaspreet, et al., 2016; Vijayarani, S., et al., 2015].

Linguistic Pre-processing Methods

- **Tokenization:** The fundamental operation of segmenting a continuous stream of text into discrete, indexable semantic units called tokens:
 - Word-Level Tokenization: Tokenizes text on white spaces and explicit punctuation marks. It is simple, but suffers from out-of-vocabulary words and complex compound words [Kaur, Jaspreet, et al., 2016].
 - Subword-Level Tokenization: Rare words are divided into common subword units

(e.g., Byte-Pair Encoding [BPE], WordPiece, Unigram). With this method, models can preserve small vocabulary sizes and yet be able to construct arbitrary or out-of-vocabulary tokens [Bojanowski, Piotr, et al., 2017].

- **Stop-Word Removal:** filter and remove the stop words (e.g. “the”, “is”, “at”, “which”) using standardized domain stop-word lists. Removing these words reduces the dimensionality and concentrates model parameters on semantically salient words [Vijayarani, S., et al., 2015].
- **Normalization Strategies (Stemming vs. Lemmatization):**
 - Stemming: A heuristic process that it is a rule-based (e.g. Porter Stemmer, Lancaster Stemmer) that chops off ends of words to reduce inflected words to a common base form. Stemming is a fast computational process, but it often leads to non-words, due to over-stemming or under-stemming [Kaur, Jaspreet, et al., 2016; Vijayarani, S., et al., 2015].
 - Lemmatization: A morphologically valid approach (e.g., WordNet Lemmatizer) which considers the Part-of-Speech (POS) tag and the grammatical context of a token to map it to its corresponding root form (or lemma) in the valid dictionary [Kaur, Jaspreet, et al., 2016; Vijayarani, S., et al., 2015].
- **Text Regularization:** It covers the conversion of characters to lowercase, removal of HTML tags, removal of non-alphanumeric symbols, expansion of contractions and handling of numerical entities [Kaur, Jaspreet, et al., 2016; Vijayarani, S., et al., 2015].

2.3. Text Representation (Feature Extraction)

Text inputs have to be converted into numerical representations as mathematical and machine learning models can only operate with numerical tensors. In modern NLP, the feature extraction methods can be divided into sparse traditional representation and dense continuous representation [Manning, Christopher D., et al., 2008; Mikolov, Tomas, et al., 2013].

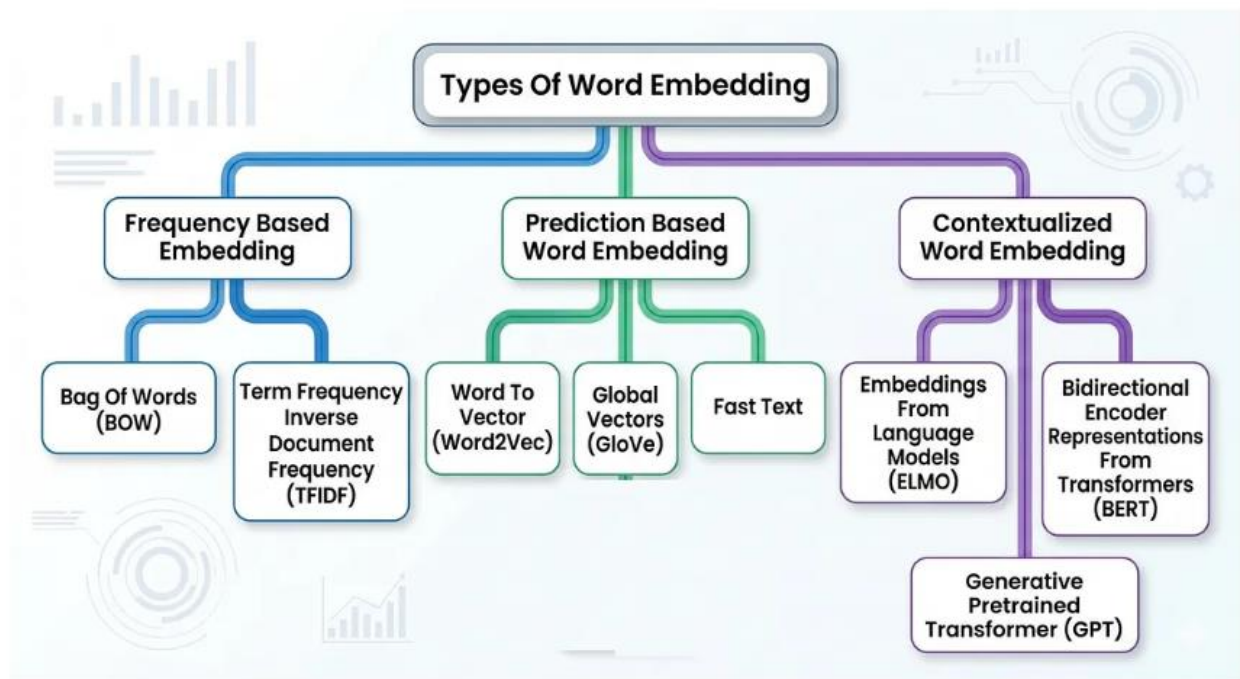


Figure 2.3: Text Representations (Word Embedding)

2.3.1. Traditional Methods (Sparse Representations)

- **Bag of Words (BoW):** Maps a document to a high dimensional vector corresponding to the size of the vocabulary. Each vector index reflects the frequency of a word within the document. BoW ignores completely the order of tokens and the grammatical syntax [Manning, Christopher D., et al., 2008].
- **N-grams:** Preserves local context by looking at a contiguous sequence of items (e.g. Unigrams, Bigrams, Trigrams). With n-grams models can distinguish between different phrases such as “not bad” vs. “bad” [Manning, Christopher D., et al., 2008].
- **Term Frequency-Inverse Document Frequency (TF-IDF):** A statistical weighting scheme for the relative importance of a term within a given document, relative to its global frequency within an entire corpus. TF-IDF

penalizes common generic words and emphasizes domain-specific keywords [Manning, Christopher D., et al., 2008].

2.3.2. *Distributed Representations (Static Word Embeddings)*

Sparse models are high dimensional and do not capture semantics similarities between words (e.g. “cat” and “feline” are treated as completely orthogonal vectors). Distributed representations project words into small continuous real-valued vector spaces where semantically similar words are close to each other [Mikolov, Tomas, et al., 2013; Pennington, Jeffrey, et al., 2014].

- **Word2Vec:** a neural framework proposed by Mikolov et al. which learns word embeddings over two different shallow architectures:
 - Continuous Bag-of-Words (CBOW): Predicts a target word from the context tokens [Mikolov, Tomas, et al., 2013; Pennington, Jeffrey, et al., 2014].
 - Skip-gram: Predicts the tokens in the context of a single target word [Mikolov, Tomas, et al., 2013; Pennington, Jeffrey, et al., 2014].
- **GloVe (Global Vectors):** An unsupervised embedding algorithm that utilizes the global co-occurrence matrix to learn the embeddings, and thus combines the advantages of global matrix factorization methods and local context window methods [Pennington, Jeffrey, et al., 2014].
- **FastText:** An extension of Word2Vec by Facebook AI Research, which represents words as bags of character n-grams. Rare, misspelt or out-of-vocabulary words are represented by aggregating subword vectors in FastText [Bojanowski, Piotr, et al., 2017].

2.4. Deep Learning Models

Traditional Feedforward Neural Networks treat text instances as independent vectors, and do not capture sequential context and temporal dependencies. Recurrent architectures are explicitly designed for maintaining the state

memory over sequential inputs [Hochreiter, Sepp, et al., 1997; Cho, Kyunghyun, et al., 2014].

2.4.1 Long Short-Term Memory (LSTM) Networks

Standard Recurrent Neural Networks (RNNs) suffer from severe vanishing and exploding gradients during backpropagation through time preventing them from learning long-range context dependencies. To solve this problem, Hochreiter and Schmidhuber introduced the (LSTM) architecture. The LSTM architecture introduces a dedicated Cell State controlled by three specialized gating mechanisms:

- **Forget Gate:** It takes the input and previous hidden state and decides what information to be thrown away from cell state [Hochreiter, Sepp, et al., 1997].
- **Input Gate:** Determines which new candidate information to add to update the current cell state [Hochreiter, Sepp, et al., 1997].
- **Output Gate:** Determines the fraction of the internal cell state to be emitted as the current hidden state output to downstream layers [Hochreiter, Sepp, et al., 1997].

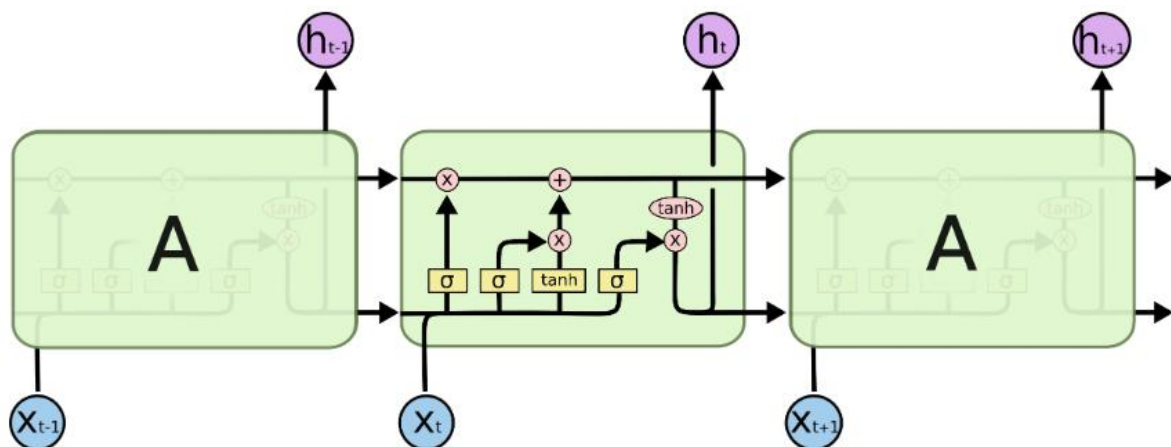


Figure 2.4.1: Long Short-Term Memory (LSTM) Networks - contains four interacting layers

Understanding LSTM Networks - Christopher Olah (colah blog).

2.4.2 Gated Recurrent Unit (GRU)

GRUs is a simplified version of LSTM architecture. The GRU merges the cell and hidden state into a single hidden state. It has two gates:

- Reset Gate: Decides to what extent the previous hidden state memory needs to be dropped while calculating the candidate state values [Cho, Kyunghyun, et al., 2014].
- Update Gate: It functions as an input and forget gate at the same time and determines the amount of new information to be added to the current memory state [Cho, Kyunghyun, et al., 2014].

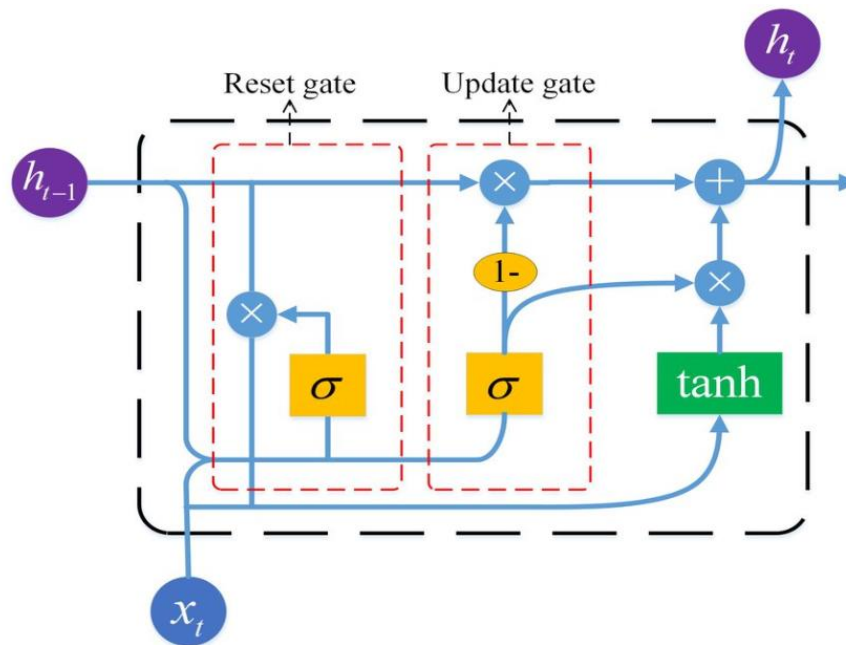


Figure 2.4.2: Gated Recurrent Unit

GRUs have fewer parameters than LSTMs, so they can be trained faster and require less computational memory, while performing comparably on many tasks.

2.5. Large Language Models (LLMs)

LLMs constitute a significant milestone in NLP, transitioning from narrow, task-specific models to broad, unified foundational architectures trained over enormous quantities of unstructured text. Rather than training separate pipelines for classification, translation and summarization, a single LLM learns general language capabilities, contextual reasoning and broad world knowledge that can be adapted to a wide range of downstream tasks [Zhao, Wayne Xin, et al., 2023].

2.5.1. Architectural Foundations of Large Language Models

2.5.1.1 The Transformer Paradigm

Modern LLMs are based on Transformer architecture. Prior to Transformers, long text sequences suffered from severe performance bottlenecks when processed sequentially by recurrent networks (e.g., LSTMs and GRUs). The Transformer removed recurrent connections and instead used Self-Attention Mechanisms which process all the words in a sequence at the same time in parallel [Vaswani, Ashish, et al., 2017].

This model uses self-attention to determine dynamically how much any token in a text is related to any other token. This allows the model to capture long-range contextual dependencies, resolve pronouns and understand complex syntactic relations irrespective of the distance between words in a document [Vaswani, Ashish, et al., 2017].

2.5.1.2 Transformer Taxonomy and Variations

The Transformer architectures can be divided into three main classes depending on the implementation of the attention mechanism [Vaswani, Ashish, et al., 2017; Zhao, Wayne Xin, et al., 2023]:

- **Decoder-Only (Autoregressive Models):** Great at generating text. The model predicts the next word in a sequence only from the previous words (causal masking). These models are at the heart of large generative assistant architectures such as GPT-4, LLaMA and Mistral [Vaswani, Ashish, et al., 2017; Zhao, Wayne Xin, et al., 2023].
- **Encoder-Only (Autoencoding Models):** For understanding text in both directions. The model considers both the words before and after a target token. These models (BERT, RoBERTa) are good for document classification, named entity recognition and dense search retrieval but not text generation [Devlin, Jacob, et al., 2019].
- **Encoder-Decoder (Sequence-to-Sequence Models):** The encoder processes the input prompt, and the decoder generates the output sequence. Such architectures (e.g., T5, BART) are useful for structural transformations such as machine translation and abstractive document summarization [Vaswani, Ashish, et al., 2017; Zhao, Wayne Xin, et al., 2023].

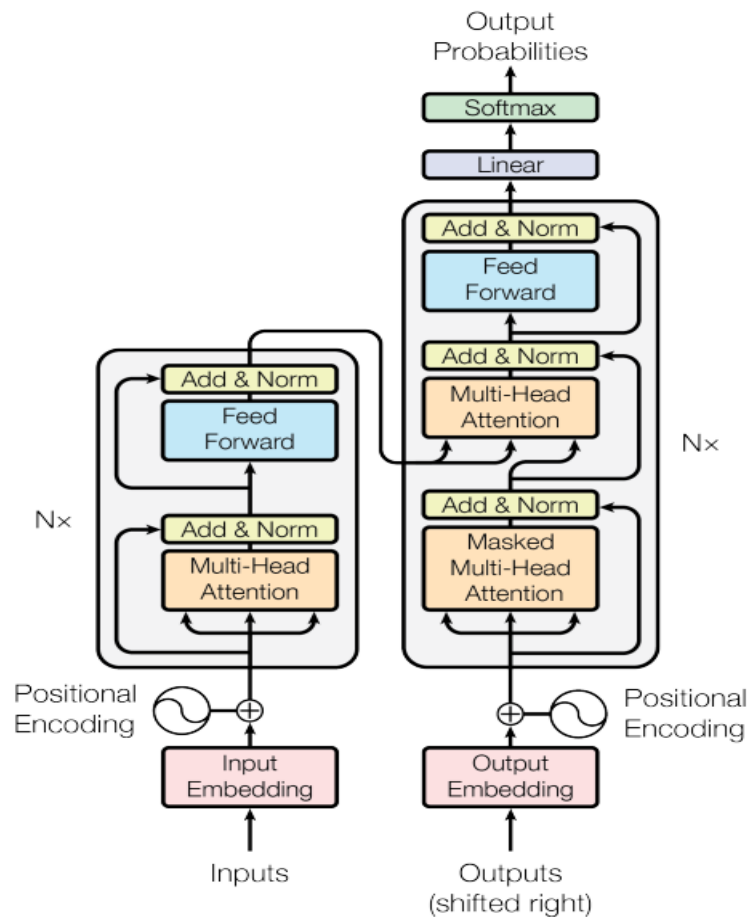


Figure 2.5.1.2: The Transformer - model architecture

[Attention Is All You Need Paper (ArXiv:1706.03762)]

2.5.2. Complete LLM Training & Alignment Lifecycle

Developing a production-ready LLM involves a structured multi-stage training pipeline, transforming raw, unstructured text into an aligned conversational system [Zhao, Wayne Xin, et al., 2023; Rafailov, Rafael, et al., 2024].

2.5.2.1 Pre-training in a self-supervised manner

In the pre-training phase the base model ingests vast amounts of data, including trillions of words from books, academic papers, code repositories and web content. Pre-training involves self-supervised learning, where the model learns by trying to predict missing or next tokens, and does not require human annotated data. At this point the model accumulates a basic vocabulary, grammar, contextual understanding and knowledge of the world [Zhao, Wayne Xin, et al., 2023].

2.5.2.2 Supervised Fine-Tuning (SFT)

The base model is good at predicting what word is likely to come next, but not so good at following instructions or directly answering a user's questions. Supervised Fine-Tuning fine-tunes the base model on curated datasets of high-quality instruction-response pairs, to teach the model to act as a helpful conversational assistant [Zhao, Wayne Xin, et al., 2023].

2.5.2.3 Parameter Efficient Fine Tuning (PEFT/LoRA)

Memory and computation intensive is fine-tuning a multi-billion parameter model by updating all the parameters. This is addressed by Low-Rank Adaptation (LoRA), which freezes the main weights of the base model and instead injects small, lightweight adapter layers into key attention modules. During fine-tuning only these small adapters are trained. This reduces the number of trainable parameters by more than 99 % while matching the performance of fine-tuning the full model [Hu, Edward J., et al., 2021].

LLM fine-tuning

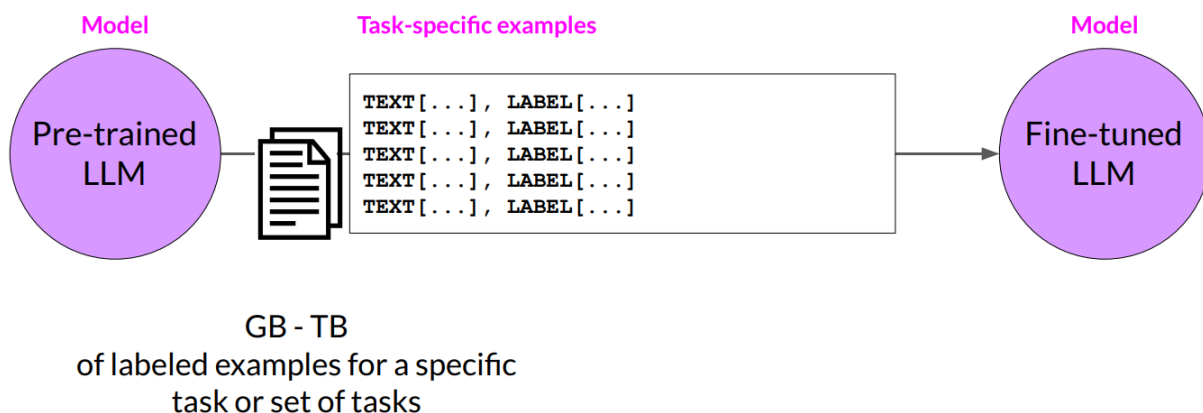


Figure 2.5.1.2: LLM fine-tuning at a high level

[Deeplearning.ai]

2.5.2.4 Approaches for Human Alignment

Models are aligned using data of human preferences. This helps keep model outputs helpful, truthful, and safe on of them is:

- **RLHF (Reinforcement Learning from Human Feedback):** - A separate Reward Model is trained on preference rankings from humans [Zhao, Wayne Xin, et al., 2023; Rafailov, Rafael, et al., 2024].

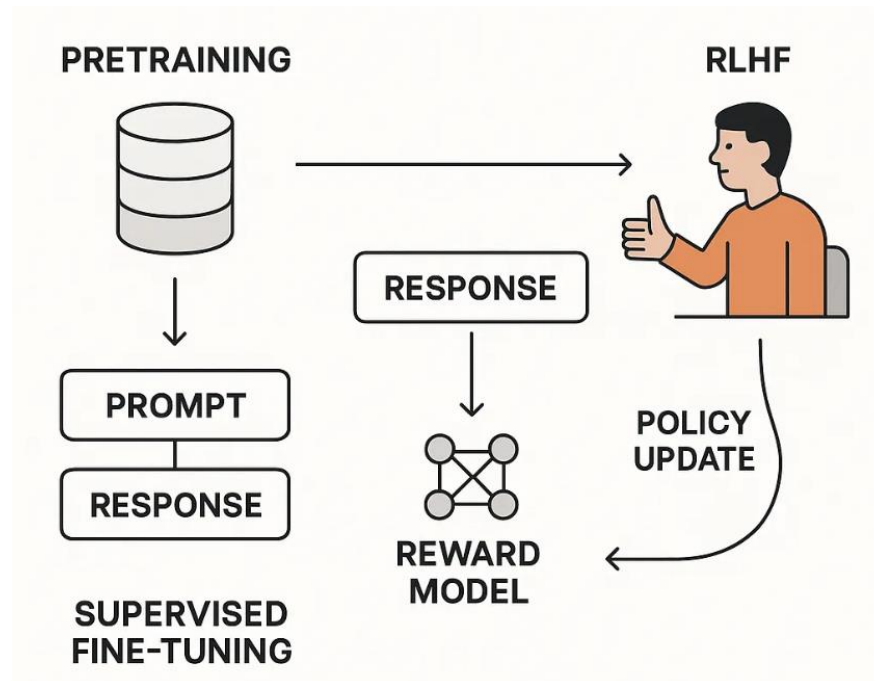


Figure 2.5.2.4: LLM fine-tuning at a high level

[Medium]

2.5.3. External Knowledge Integration: Retrieval-Augmented Generation (RAG)

2.5.3.1 The Need for Grounded Retrieval

Despite their strong reasoning capabilities, standalone LLMs face inherent limitations: static training cutoffs, lack of access to private institutional data, and a tendency to generate plausible-sounding but incorrect information (hallucinations). **Retrieval-Augmented Generation (RAG)** addresses these issues by pairing the generative LLM with an external document retrieval pipeline [Lewis, Patrick, et al., 2020].

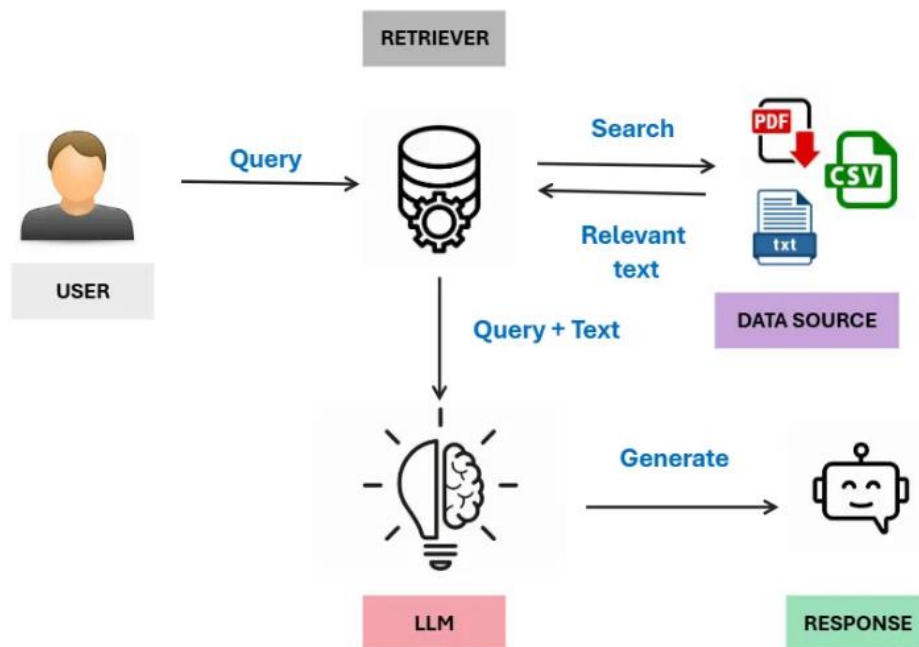


Figure 2.5.3.1: RAG Example

[Medium]

2.5.3 Incorporating External Knowledge: Retrieval-Augmented Generation (RAG)

2.5.3.1 Why Grounded Retrieval is Required

Standalone LLMs, while powerful in their reasoning abilities, have inherent limitations tendency to generate plausible-sounding but incorrect information (hallucinations). Retrieval-Augmented Generation (RAG) solves these issues by coupling the generative LLM with an external document retrieval pipeline.

2.5.3.2 Advanced Institutional Document Pipeline

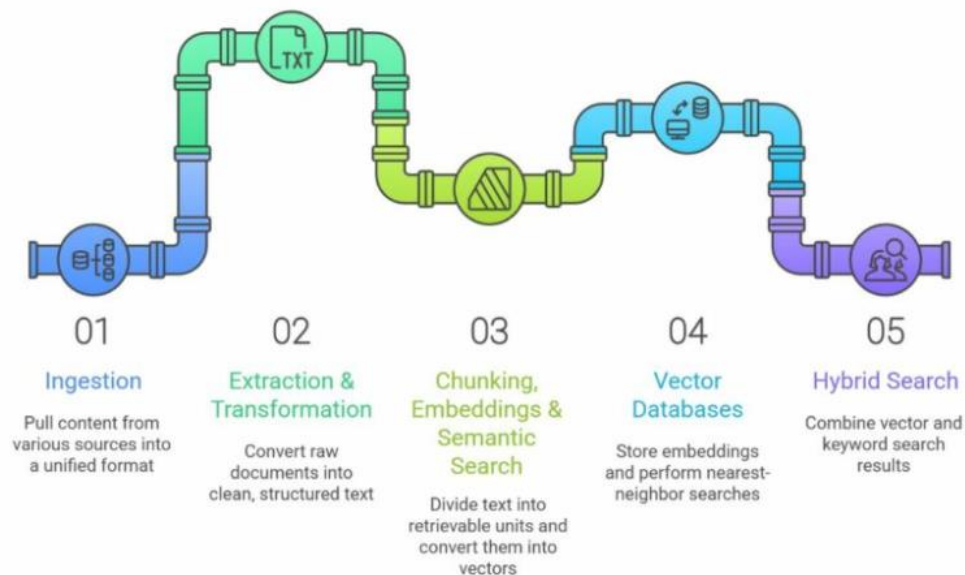


Figure 2.5.3.3: RAG Pipeline Components

[wearefram.com]

- Document Parsing and Semantic Chunking:** Complex lists and formal clauses are found in structured institutional documents (academic regulations, bylaws, course catalogues, PDFs). Fixed size chunking can destroy meaningful context. Semantic Chunking Strategies split documents into sections according to headings, paragraphs, and logical boundaries, and thus each chunk contains a full administrative rule or concept [Lewis, Patrick, et al., 2020].
- Vector Indexing & Dense Retrieval:** Text chunks are transformed into high dimensional vector spaces, and then indexed in vector databases (e.g. Chroma DB, FAISS, Qdrant). When a user asks a question, the system retrieves the most relevant passages according to their semantic similarity (e.g. Cosine Similarity) [Manning, Christopher D., et al., 2008; Lewis, Patrick, et al., 2020].
- Re-Ranking Mechanisms for Accurate Retrieval:** The initial retrieval stage frequently produces multiple candidate passages with keyword overlaps but no precise relevance. A secondary Cross-Encoder Reranker scores and re-orders the retrieved chunks, so that the exact regulatory passage is at the very top of the context window before it reaches the LLM [Lewis, Patrick, et al., 2020].

- **Contextual Prompt Building:** The top-ranked passages are directly injected into the prompt, along with the user query, giving the LLM the exact information it needs to generate the response [Lewis, Patrick, et al., 2020].

2.5.4. *Hallucination mitigation, guard-rail and multi-language support*

2.5.4.1 *Prompting and Mitigating Hallucinations*

Providing wrong information in an institutional setting can cause serious confusion. To prevent model hallucinations, strict guardrails are embedded into the system:

- **Strict Context Guard Rails:** System prompts explicitly instruct the LLM to respond only based on the retrieved passages of context. If the answer is not in the context the model is strictly conditioned to decline politely (e.g. "This information is not available in the provided documents") [Lewis, Patrick, et al., 2020].
- **Citation & Source Attribution:** Modern RAG architectures provide explicit metadata (file name, section title, page number) to every retrieved passage. The LLM is prompted to cite these sources directly in its answer, allowing users to verify the validity of the response against the official document [Lewis, Patrick, et al., 2020].

2.5.4.2 *Multilingual NLP for Various User Requests*

Institutional settings often require bilingual support (e.g. Arabic and English) and have to cope with a variety of language registers from formal terminology to colloquial expressions:

- **Multilingual Embedding Models:** Uses specialized vector space models that can map different languages into shared semantic spaces, matching queries in one language to document passages in another [Zhao, Wayne Xin, et al., 2023].
- **Cross-Lingual Language Understanding:** Advanced multilingual LLMs can handle enquiries posed in local dialects or informal formats and accurately associate them with formal institutional documents [Zhao, Wayne Xin, et al., 2023].

2.5.5. *Evaluation Frameworks for Knowledge-Grounded Systems*

Evaluating document-grounded, conversational AI systems requires specialized metrics beyond standard classification measures [Sokolova, Marina, et al., 2009; Es, Shahul, et al., 2023]. Frameworks such as **RAGAS** evaluate the pipeline across several key dimensions [Es, Shahul, et al., 2023]:

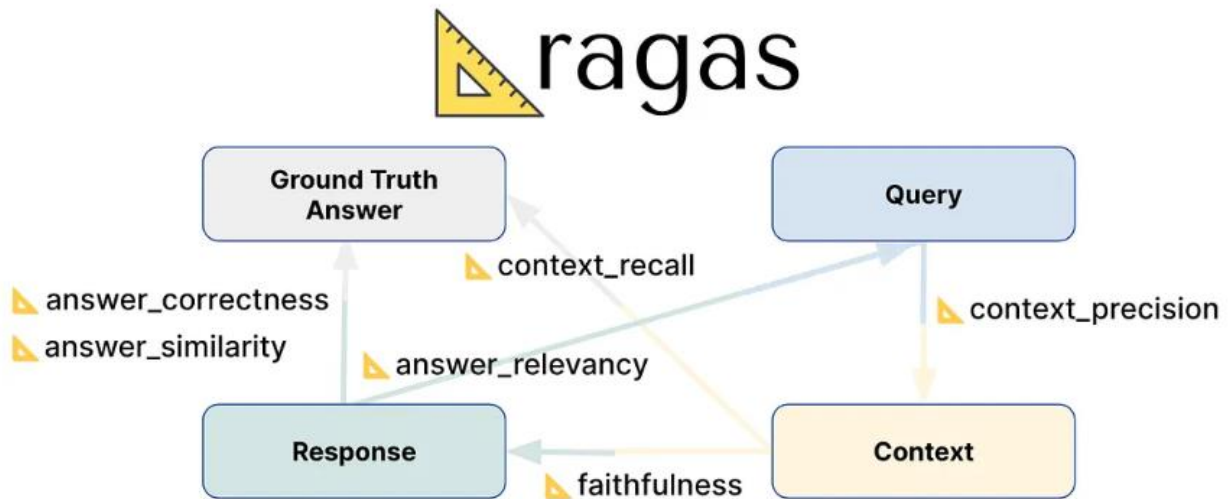


Figure 2.5.5: RAGAS

[Meduim.com]

Evaluation Metric	Evaluated Pipeline Stage	Objective & meaning
Faithfulness (Groundedness) [Es, Shahul, et al., 2023]	Generated Response vs. Retrieved Context	Verifies that every claim in the response is directly supported by the retrieved official documents, guaranteeing zero hallucinations.

Evaluation Metric	Evaluated Pipeline Stage	Objective & meaning
Answer Relevance [Es, Shahul, et al., 2023]	Generated Response vs. User Query	Ensures the response directly answers the user's specific question without adding off-topic information.
Context Precision [Es, Shahul, et al., 2023]	Retrieved Passages vs. User Query	Measures whether the vector store fetched relevant document sections while filtering out background noise.
System Latency & Performance [Sokolova, Marina, et al., 2009]	End-to-End Pipeline	Tracks end-to-end response time to ensure fast and responsive performance for user interaction.

2.2 Summary of Related Work

Paper 1:

URAG: Implementing a Unified Hybrid RAG for Precise Answers in University Admission Chatbots – A Case Study at HCMUT

Long Nguyen and Tho Quan proposed a unified hybrid retrieval-augmented generation method, the Unified RAG (URAG) framework, to tackle the high operational costs and the risk of hallucination in university chatbots. They integrated a two-stage retrieval architecture (consisting of FAQ search and document search stages) with an in-house lightweight generator Large Language Module (LLM) to deliver precise and accurate answers to questions about university admission processes, academic programs, and campus guidelines. Instead of naive RAG or advanced RAG methods that consume huge resources, their method is to collect 300 domain-related documents from official university sources, segment them with semantic chunking, enrich the database with URAG-D (document rewrite) and URAG-F (FAQ enrichment) mechanisms, and use the dangvantuan/vietnamese-embedding model for vector representation. They compared their light-weight model (ura-hcmut/ura-llama-7b) embedded in the URAG pipeline with the top commercial models such as GPT-4o, Claude 3.5 Sonnet, and Gemini 1.5 Pro. The systems were quantitatively assessed using an accuracy metric on a dataset consisting of 500 genuine student questions. The results showed that the URAG-powered HCMUT Chatbot achieved the best performance, outperforming the commercial models with an accuracy of 0.628, and provided a highly feasible solution for real-world academic deployments.

Paper 2:

From Questions to Insightful Answers: Building an Informed Chatbot for University Resources

Subash Neupane investigated an informed chatbot system built using Retrieval-Augmented Generation (RAG) pipelines to enhance information access and user experience for university resources. To overcome their limitations of standalone pretrained large language models (LLMs) that cannot answer domain-specific queries and hallucinate outputs, they proposed the BARKPLUG V.2 framework. Methodology: They built an external knowledge corpus by creating a multi-threaded web-scraper to collect and curate text data from 42 campus resources at Mississippi State University (MSU). The corpus was preprocessed by a recursive chunking strategy of 8000 characters and overlap of 1200 characters. Using the text-embedding-3-large model through API calls, the text was transformed to vector representations and stored in an in-memory ChromaDB. For the retrieval phase, they used a vector store backed similarity score threshold retrieval mechanism using LangChain. In the completion phase, we created accurate responses using the generator model of OpenAI, i.e., the generator model of gpt-3.5-turbo. The authors quantitatively evaluated BARKPLUG V.2 using the Retrieval Augmented Generation

Assessment (RAGAS) framework on several categories, including engineering programs, general enquiries, research opportunities, and university resources. In addition, 50 students performed a subjective usability assessment using the System Usability Scale (SUS). The results showed that mean score of 0.96 and acceptance score of 67.75, indicating that the chatbot is practically feasible for real-world academic settings.

Paper 3:

From Surface Learning to Deep Understanding: A Grounded AI Tutoring System for Moodle

The authors presented the development of the AI Teaching & Learning Assistant, a modular Moodle plugin based on Retrieval-Augmented Generation (RAG) for hallucination-free education. The system's dual-centric design allows students to interact with interactive, Socratic-style tutoring, while giving educators a "Human-in-the-loop" workspace for supervised content creation. In their approach they built a presentation layer natively integrated as a PHP Moodle Activity Module using a Next.js/React frontend and a Python FastAPI backend for asynchronous parsing of PDFs, DOCX and PPTX, semantic text embedding and intent recognition. Data storage was done in a hybrid manner, using PostgreSQL, ChromaDB and MinIO. They systematically optimized the retrieval and completion phases in various disciplines, and chose the Gemini Flash 2.0 as the production model over GPT-4o-mini and Llama 3.1, as it has a higher faithfulness and speed-to-cost ratio, with an optimal retrieval context of Top-K=10. The authors quantitatively assessed the system using the RAGAS framework for STEM and Humanities disciplines and particular modes of student interaction. They also ran a usability study with 18 participants. The results showed high performance with RAGAS faithfulness scores of up to 0.97 in STEM, 0.98 in Humanities and a high user recommendation rate of 4.00 out of 5.00.

Paper 4:

Large Language Model-Based Chatbots in Higher Education

The authors investigated the integration, features, and pedagogical uses of Large Language Model (LLM) chatbots in higher education. In several generative models they contrasted various transformer architectures, training datasets, parameters and fine-tuning strategies. They discussed how LLM-based chatbots can foster highly customized instruction, provide real-time formative feedback, and support curriculum development. However, they pointed out some of the main limitations including academic dishonesty (cheating and plagiarism), data privacy breaches, algorithmic bias, and the spread of misinformation or "hallucinations". The results showed that GPT-4 is better than other models on a set of standardized human examinations. Adding an information retrieval system (IRS) to an educational question answering system improves the accuracy compared to vanilla LLM models.

Paper 5:

EduChat: A Large-Scale Language Model-based Chatbot System for Intelligent Education

The authors investigated the educational chatbot system, EduChat, based on a large-scale language model (LLM), for individualized, fair, and compassionate intelligent education. For training, they used two main datasets: (1) pre-training corpus of Chinese middle/high school textbooks, exam questions, 70k Chinese poems and 60 psychology books; (2) fine-tuning dataset of 500k customized educational instructions, including essay assessment, emotional support, Socratic teaching and open QA. Pre-processing involved semantic-level deduplication using sentence embeddings, removing one duplicate from pairs with cosine similarity above 0.7, accelerated with Compute Unified Device Architecture (CUDA) on a Graphics Processing Unit (GPU). The authors constructed structured system prompts that include personal profiles, tool usage and skill selection, and a self-check retrieval-augmented generation (RAG) mechanism. Results showed that EduChat (with retrieval) achieved the best performance on the C-Eval benchmark among other models of similar parameter scale.

Paper 6:

Reinforcement Learning for Optimizing RAG for Domain Chatbots

The authors explored a Reinforcement Learning (RL) based framework for optimizing the number of LLM tokens and reducing API costs in Retrieval-Augmented Generation (RAG) pipelines for domain-specific chatbots. They used a proprietary domain FAQ dataset of 72 FAQs relating to the credit card application process. They also used ~3.5k training, 1k validation, and 1,014 test paraphrase queries (in English and Hinglish) generated through ChatGPT and manual tagging. The pre-processing steps involved concatenating questions and answers in each FAQ to obtain vector embeddings, keeping track of conversation history by remembering the last two turns in the conversation, and formatting the inputs into specific state tokens (Classification Token [CLS] and Separator Token [SEP]) to train the external policy network. The results showed that the proposed SimThr + policy (BERT) model obtained the best performance, achieving a good balance between retrieval accuracy, generation quality, and system operating costs compared to traditional "All fetch" RAG baselines and general-purpose public embedding models.

Paper 7:

Design and Performance Evaluation of LLM-Based RAG Pipelines for Chatbot Services in International Student Admissions

Khasanova Zafar kizi,Suh investigated different pipeline configurations of LLM-based Retrieval-Augmented Generation (RAG) systems to design and evaluate an AI chatbot for international

student admissions. They used an internal university admissions corpus (a 29-page PDF guidebook, 38 website FAQs and 11 department FAQs) besides two dual evaluation reference datasets: an LLM-generated QA set (72 pairs) and a human-tagged QA set (62 pairs). The pre-processing steps consisted of loading documents with metadata labels, splitting the texts using either Recursive Character TextSplitter or Semantic Chunker, converting the chunks into 384-dimensional dense vectors using the all-MiniLM-L6-v2 embedding model, and storing them in an FAISS index. The results showed that the advanced open-source OpenChat model pipeline (openchat:7b-v3.5-0106-fp16 with Recursive splitting and MultiQuery Hybrid retriever) had the best performance, outperforming the baseline commercial GPT-4o pipeline.

Paper 8:

Large Language Model based Educational Virtual Assistant using RAG Framework

Umair Hasan Khan, Muneeb Hasan Khan, Rashid Ali investigated approach to address the complexity of university systems by developing an educational virtual assistant. They combined Large Language Models (LLMs) with a Retrieval-Augmented Generation (RAG) framework to offer accurate and current information support for university admissions, course selection, and campus facilities. They used web scraping to scrape the domain-specific data from the official university website (amu.ac.in and amucontrollerexams.com) instead of the old-fashioned administrative methods. They preprocessed the collected documents (scanned files using Optical Character Recognition (OCR) tools such as pytesseract) and created the vector representations using an embedding model (all-mpnet-base-v2). They inserted the open-source models Llama-2-7b-chat-hf and Mistral-7B-Instruct-v0.2 into the RAG pipeline and tested it. The models were quantitatively evaluated based on the bilingual evaluation understudy (BLEU) score metrics. The results showed that the RAG-based Llama-2-7b-chat-hf model performed the best, surpassing the Mistral model at all n-gram levels considered and was a very feasible solution for automating administrative support.

Paper	All Models	Datasets	Dataset Link	Limitation	Best Model	Results
[1]	GPT-4o Claude 3.5 Sonnet Gemini 1.5 Pro HCMUT Chatbot	HCMUT university admissions	huggingface.co/ura-hcmut/ura-llama-7b	performance compared to larger models or third-party APIs, especially when addressing general queries beyond the admissions domain	HCMUT Chatbot	Accuracy=0.62

[2]	gpt-3.5-turbo chroma langchain	Scrapped 42 campus resources into a JSON file from the university site	Scrapped from www.msstate.edu	No Automatic Speech Recognition (ASR) capability	gpt-3.5-turbo	RAGAS score of 0.96
[3]	Gemini Flash 2.0 GPT-4o-mini Llama 3.1	teacher-provided course PDF, DOCX, and PPTX documents	tinyurl.com/mrytc2dr	Not mentioned	Gemini Flash 2.0	RAGAS score of 0.97
[4]	GPT-1, GPT-2, GPT-43, GPT-4, BERT, ROBERTa, ALBERT, ELECTRA, XLNet, T5, LaMDA	Wikipedia Toronto Book Corpus CC News	NA	Inaccuracies generated by AI (hallucinations), misinformation, threats to data privacy, plagiarism/cheating, and dissemination of biases	GPT-4	Standardized Test Performance: - Uniform Bar Exam: 90th percentile - LSAT: 88th percentile - AP Biology, AP Macroeconomics, etc.: 5/5 - AP English Literature: 2/5
[5]	EduChat	(1) pre-training corpus of textbooks, exam questions, (2) fine-tuning dataset of 500k open QA	https://github.com/ECNU-ICALK/EduChat https://huggingface.co/ecnu-icalk	Hallucinations generation, knowledge gap in education and obsolete updates	EduChat (with Retrieval)	C-Eval Benchmark: - EduChat (with Retrieval) Avg =49.3 - EduChat (Vanilla) Avg =40.7 - Chinese-Alpaca-13B Avg =33.3

[6]	BERT GPT2 RAG	Credit Card FAQ Dataset (72 FAQs)	Not Public	Assumes the underlying RAG retriever and LLM weights are fixed. Prompt- guided evaluation is heavily dependent on GPT-4, which could introduce bias.	SimThr + policy (BERT)	Token Savings ≈31% Accuracy = 1.0
[7]	GPT-4o Ollama LLaMA3 OpenChat (latest, q6_K, fp16)	Internal University Admissions Guide (29-page PDF, FAQ1, FAQ2)	Not Public	Small size of evaluation datasets	openchat:7b- v3.5-0106-fp16	- Average RAGAS Score =0.7377 - Answer Relevancy = 0.785 - BLEU-1 = 0.5720
[8]	- Llama-2-7b- chat-hf - Mistral-7B- Instruct-v0.2 RAG	Custom university dataset built from website pdfs, scanned documents and jpegs.	Not Public	Computational constraints (reliance on Google Colab free GPU limits performance and testing).	Llama-2-7b- chat-hf (integrated with RAG)	

2.3 References

[Bojanowski, Piotr, et al., 2017]

Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov. “Enriching Word Vectors with Subword Information”. Transactions of the Association for Computational Linguistics 5, 135–146., 2017.

[Cho, Kyunghyun, et al., 2014]

Cho, Kyunghyun, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. “Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation”. arXiv preprint arXiv:1406.1078., 2014.

[Chowdhury, Gobinda G., 2003]

Chowdhury, Gobinda G. “Natural language processing”. Journal of Documentation 59 (1), 51–68., 2003.

[Devlin, Jacob, et al., 2019]

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. Proceedings of NAACL-HLT, 4171–4186., 2019.

[Es, Shahul, et al., 2023]

Es, Shahul, Jithin James, Luis Espinosa-Anke, and Steven Schockaert. “Ragas: Automated Evaluation of Retrieval Augmented Generation”. arXiv preprint arXiv:2309.15217., 2023.

[Hochreiter, Sepp, et al., 1997]

Hochreiter, Sepp, and Jürgen Schmidhuber. “Long Short-Term Memory”. Neural Computation 9 (8), 1735–1780., 1997.

[Hu, Edward J., et al., 2021]

Hu, Edward J., Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. “LoRA: Low-Rank Adaptation of Large Language Models”. arXiv preprint arXiv:2106.09685., 2021.

[Kaur, Jaspreet, et al., 2016]

Kaur, Jaspreet, and Jatinderkumar R. Saini. “Traditional and Nature-Inspired Techniques for Text Pre-Processing: A Review”. *International Journal of Computer Applications* 137 (10), 18–25., 2016.

[Lewis, Patrick, et al., 2020]

Lewis, Patrick, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks”. *Advances in Neural Information Processing Systems (NeurIPS)* 33, 9459–9474., 2020.

[Manning, Christopher D., et al., 2008]

Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schütze. “Introduction to Information Retrieval”. Cambridge University Press., 2008.

[Mikolov, Tomas, et al., 2013]

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. “Efficient Estimation of Word Representations in Vector Space”. *arXiv preprint arXiv:1301.3781.*, 2013.

[Pennington, Jeffrey, et al., 2014]

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. “GloVe: Global Vectors for Word Representation”. *Proceedings of EMNLP*, 1532–1543., 2014.

[Rafailov, Rafael, et al., 2024]

Rafailov, Rafael, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. “Direct Preference Optimization: Your Language Model is Secretly a Reward Model”. *Advances in Neural Information Processing Systems (NeurIPS)* 36., 2024.

[Ramos, Juan, 2003]

Ramos, Juan. “Using TF-IDF to Determine Word Relevance in Document Queries”. *Proceedings of the First Instructional Conference on Machine Learning* 242 (1), 29–48., 2003.

[Sokolova, Marina, et al., 2009]

Sokolova, Marina, and Guy Lapalme. “A Systematic Analysis of Performance Measures for Classification Tasks”. *Information Processing & Management* 45 (4), 427–437., 2009.

[Vaswani, Ashish, et al., 2017]

Vaswani, Ashish, 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 (NeurIPS)* 30, 5998–6008., 2017.

[Vijayarani, S., et al., 2015]

Vijayarani, S., J. Ilamathi, and C. Nithya. “Preprocessing Techniques for Text Mining”. *International Journal of Computer Science & Communication Networks* 5 (1), 7–16., 2015.

[Zhao, Wayne Xin, et al., 2023]

Zhao, Wayne Xin, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Zheng, Zelai Chen, and Ji-Rong Wen. “A Survey of Large Language Models”. *arXiv preprint arXiv:2303.18223.*, 2023.

[URAG: Implementing a Unified Hybrid RAG for Precise Answers in University Admission Chatbots -- A Case Study at HCMUT]

<https://arxiv.org/abs/2501.16276>

[From Questions to Insightful Answers: Building an Informed Chatbot for University Resources]

<https://arxiv.org/abs/2405.08120>

[From Surface Learning to Deep Understanding: A Grounded AI Tutoring System for Moodle]

<https://arxiv.org/html/2605.06963v1>

[Large Language Model-Based Chatbots in Higher Education]

<https://advanced.onlinelibrary.wiley.com/doi/epdf/10.1002/aisy.202400429>

[EduChat: A Large-Scale Language Model-based Chatbot System for Intelligent Education]

<https://arxiv.org/abs/2308.02773>

[Reinforcement Learning for Optimizing RAG for Domain Chatbots]

<https://arxiv.org/abs/2401.06800>

[Design and Performance Evaluation of LLM-Based RAG Pipelines for Chatbot Services in International Student Admissions]

<https://www.mdpi.com/2079-9292/14/15/3095>

[Large Language Model based Educational Virtual Assistant using RAG Framework]

<https://www.sciencedirect.com/science/article/pii/S1877050925000511>

[Advanced NLP models for Technical University Information Chatbots: Development and Comparative Analysis. IEEE Access, 12, 29633-29647]

<https://doi.org/10.1109/ACCESS.2024.3368382>

[Enhancing Arabic Information Retrieval for Question Answering]

<https://avesis.kocaeli.edu.tr/yayin/330f25f8-21dc-4b50-a43d-d65a9853bcda/enhancing-arabic-information-retrieval-for-question-answering>

[Improving LLM Reliability with RAG in Religious Question-Answering: MufassirQAS]

<https://arxiv.org/abs/2401.15378>