Efficient RAG Framework for Large-Scale Knowledge Bases

Karthik Meduri
University of the Cumberlands
9 PUBLICATIONS **75** CITATIONS
SEE PROFILE

Geeta Sandeep Nadella
University of the Cumberlands
13 PUBLICATIONS **70** CITATIONS
SEE PROFILE

Hari Gonaygunta
University of the Cumberlands
15 PUBLICATIONS **169** CITATIONS
SEE PROFILE

Mohan Harish Maturi
Institute of Electrical and Electronics Engineers
2 PUBLICATIONS **0** CITATIONS
SEE PROFILE

IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

Efficient RAG Framework for Large-Scale Knowledge Bases

[1] Karthik Meduri, [2] Geeta Sandeep Nadella, [3] Hari Gonaygunta, [4] Mohan Harish Maturi,
[5] Farheen Fatima
[1,3,4,5] IEEE Member, [2] IEEE ENCS Computer Society Chair -R3
[1,2,3,4,5] Dept. of Information Technology,
[1,2,3,4,5] University of the Cumberlands, Williamsburg, KY, USA.

Abstract: This research paper explores the nuances of optimizing large models of languages (LLMs) for the effective creation and retrieval of information. The current research investigation focuses on two main approaches: Knowledge Distillation (KD) and Retrieval-Augmented Generation (RAG), in addition to quantization and pruning strategies. KD reduces the size of LLMs without compromising functionality to maximize LLM efficiency and resource usage, whereas RAG combines external knowledge sources with LLMs to allow contextually relevant replies. LLMs are further optimized for constrained resource contexts through the use of quantization and trimming algorithms. By conducting a thorough assessment of the querying procedure, the research demonstrates the capacity of the model to produce precise answers and pinpoint areas in need of improvement. This study advances the architecture of the RAG Framework. It investigates its possibilities, providing large-scale scalable knowledge and practical solutions for knowledge creation and retrieval across a variety of fields, so opening the door for improved information access and human-machine interaction. This research will support the advancement of knowledge retrieval in all fields in the future.

***IndexTerms* – LLM (Large Language Model), RAG Model, Knowledge Distillation, Quantization and Pruning Techniques, NLP (Natural Language Processing).**

1. INTRODUCTION

Knowledge retrieval systems and natural language processing (NLP) have seen tremendous advancements in recent years. The amount of textual material that is available on the internet is growing, making the effective administration and use of large-scale knowledge bases essential tasks [1]. The Retrieval-Augmented Generation (RAG) model is one of the ways that has been created to address this difficulty. It is promising in that it can seamlessly integrate generators and retrievers to handle complicated queries across large-scale knowledge repositories. With a mix of retrieval and generation processes, the RAG model can explore large information repositories, including whole Wikipedia articles, which is an essential achievement in NLP. RAG provides a comprehensive framework for knowledge retrieval and synthesis by utilizing generators to generate coherent replies and retrievers to find pertinent portions [2]. Scaling the RAG model to handle large knowledge bases poses significant computing challenges, notwithstanding its potential. Large-scale and intricate repositories such as Wikipedia provide considerable computing resource constraints, which hinder the real-world implementation of RAG-based systems.
The creation of effective frameworks designed especially for massive knowledge stores is urgently needed. This research addresses this difficulty by presenting a new framework that maximizes the performance of RAG models in large-scale knowledge domains [3]. Our approach is designed to maintain or even improve performance measures like retrieval accuracy and response coherence while simultaneously reducing the computational costs related to managing large-scale knowledge bases. We want to fully realize the promise of RAG models for functional implementation in real-world applications that need access to enormous information stores by taking on these computational issues head-on.

1.1 Background and Motivation

The background and impetus for creating effective frameworks to manage massive knowledge bases are rooted in the growing need for advanced natural language processing (NLP) solutions across several sectors. Effective methods to extract, analyze, and provide insights from textual data are critical given the exponential growth in textual data volume, especially with the rise of online content sources such as Wikipedia [4]. In reaction, scholars and industry professionals have investigated a number of strategies to improve NLP systems' capabilities. The Retrieval-Augmented Generation (RAG) model is one well-known strategy that has gained popularity. RAG is a combination of generation and retrieval methods that allows systems to provide well-reasoned answers to user queries and extract pertinent information from vast knowledge stores. This method has great potential for applications where access to extensive knowledge bases is necessary, such as conversational bots, content summarization, and question-answering. There are several obstacles to the actual use of RAG prototypes, especially when dealing with large information bases like Wikipedia articles [4-5]. The processing and analysis of such large amounts of data may be computationally intensive, which limits the scalability and practicality of RAG-based systems.

1.2 Problem Statement

The main issue this research attempts to solve is how scalable the RAG model is for handling large-scale knowledge bases, such as whole Wikipedia articles. Current RAG implementations find it challenging to handle the high amount of computation that comes with such extensive archives, which limits their value to users [6]. Thus, methods for effective manufacturing and retrieval across these large knowledge bases that do not compromise on speed are desperately needed.

1.3 Research Contribution

This paper makes numerous contributions to the field of large-scale knowledge retrieval and generation:

- It proposes novel techniques for efficiently scaling the RAG model to handle massive knowledge bases.
- It introduces efficient retrieval algorithms designed precisely for large-scale repositories, addressing the computational challenges associated with retrieval.
- It provides empirical evaluations demonstrating the efficiency of the proposed approach in decreasing computational costs while sustaining high-quality generation results.

This research offers up new possibilities for utilizing vast knowledge bases in different NLP applications, such as content creation and question answering, by solving the scalability limits of current RAG implementations.

2. Literature Review: RAG Model Process

A significant development in natural-language processing is the Retrieval Augmented-Generations (RAG) paradigm, which provides a comprehensive approach to managing intricate queries across large knowledge libraries [7]. Retrieval and generation are the two fundamental components that the RAG model integrates.

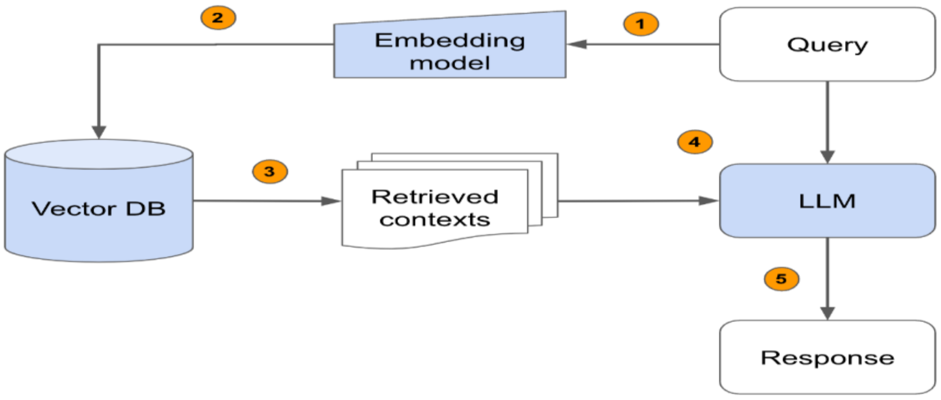


Figure 1: RAG model overview process

The above figure represents the process of RAG retrieval with LLM. The model uses complex algorithms to search through substantial knowledge bases, including collections of articles similar to those on Wikipedia, during the retrieval phase. The aim is to find appropriate passages or publications with information relevant to a specific question. Efficient access to the vast amount of knowledge contained in these repositories is contingent upon the retrieval procedure [7-8]. The RAG paradigm is unique in that it seamlessly integrates retrieval and generation techniques. RAG utilizes the advantages of both retrieval and generation techniques, in contrast to standard question-answering systems that only use one or the other, to produce more thorough and accurate responses. Through the utilization of retrieval-based evidence in the generating process, RAG generates replies that are rich in background and factual. The most significant vector database and query response system, the RAG model, is made to be highly flexible and scalable, able to process queries from a wide range of knowledge fields [9]. Its design makes it adaptable to a diversity of NLP responsibilities, enabling modification besides fine-tunings to fit particular use cases and applications.

2.1 Existing Approaches for Large-Scale Knowledge Bases

Current methods for managing large-scale knowledge bases include a variety of approaches and strategies, each designed to tackle particular difficulties related to processing enormous amounts of textual data [10]. These methods can be generally divided into a number of essential tactics.

- **Searching and indexing algorithms:** To effectively find pertinent documents inside sizable knowledge repositories, traditional information retrieval approaches rely on indexing techniques like inverted indexes and search algorithms like TF-IDF (Term Frequency-Inverse-Document-Frequency). These procedures serve as the basis for many information retrieval systems, enabling the quick and accurate retrieval of information in response to user requests [11].
- **Computing framework:** Large-scale knowledge base processing and analysis are now easier to handle because of distributed computing frameworks like Apache Hadoop and Spark. These frameworks make it possible to process data in parallel across several nodes, which makes it possible to compute complicated tasks like indexing, querying, and analysis in a scalable and effective manner.
- **NLP and ML Techniques:** Deep learning architectures such as transformers, along with other advanced machine learning and NLP models, are being used more and more to extract insights from massive amounts of text data. Large knowledge bases that may be effectively handled in models are B-E-R-T (Bidirectional-Encoder-Representations from Transformer)

and GPT (Generative Pre-trained Transformer), which are excellent at tasks like text categorization, summarization, and question answering. Knowledge graphs are organized representations of knowledge that use a graph-based structure to store entities, connections, and properties [11-12]. Knowledge graphs facilitate the effective traversal and retrieval of pertinent knowledge from extensive sources by arranging data into interconnected nodes and edges. Knowledge graphs are more useful for knowledge retrieval tasks when they are filled and refined with techniques like entity linking and relation extraction.

- **Mixed Approaches:** To maximize the benefits of various methodologies, a number of current approaches include components of the tactics mentioned above. For instance, hybrid systems might use distributed computing frameworks to improve the scalability of knowledge retrieval and analysis operations, or they can combine machine learning models with conventional indexing techniques.

The current state of large-scale knowledge base methods is varied and dynamic, with a combination of high-tech technology and established methods [9-13]. Researchers and practitioners may successfully navigate the intricacies of huge knowledge repositories and extract insightful information to serve a variety of applications and use cases by combining these techniques.

2.2 Scaling RAG Challenges

Scaling the Retrieval-Augmented Generation (RAG) model to handle large-scale knowledge bases poses several significant challenges, which include:

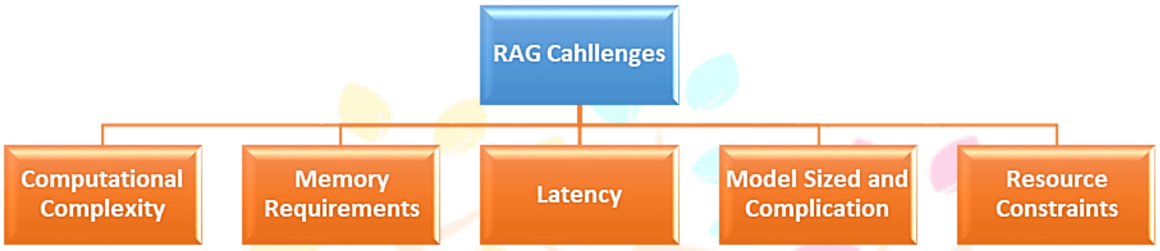


Figure 2: RAG Challenges

Computational Complexity: Significant computational complexity is introduced when the Retrieval-Augmented Generation (RAG) paradigm is scaled to large-scale knowledge bases. Finding pertinent portions requires sifting through enormous stores of text data, including whole collections of articles, during the retrieval step [14]. For this method to traverse and evaluate the text effectively, complex algorithms and substantial computer resources are usually needed. The retrieval phase's computing needs rise with the amount of the knowledge base, resulting in longer processing times and higher resource consumption. Controlling RAG model computational complexity is essential to provide scalable and effective knowledge production and retrieval systems [15].

Memory Requirement: RAG-based systems require much memory to store and manage indexed representations of substantial knowledge bases. Building and maintaining indexes or data structures is a typical retrieval phase task that enables quick and effective information retrieval [16]. These indexes have the potential to use up large amounts of memory, especially when processing large text corpora. Memory limitations and scalability problems may arise as a result of the memory needed to store and maintain these indexes as the knowledge base gets more significant. Effective memory management is crucial to maximizing RAG models' scalability and performance.

Latency: Latency is challenging to achieve in real-time responsiveness in RAG-based systems. Large knowledge bases are searched in order to find pertinent sections during the retrieval phase. This procedure might cause delays and affect how responsive the system is. Extended retrieval durations may result in heightened delay while producing answers to customer inquiries, hence impacting the user experience in its entirety. RAG-based systems must have minimal latency in order to provide accurate and timely information creation and retrieval, especially in scenarios involving interactive applications and real-time situations.

Model sized and complication: Scaling the RAG model to accommodate huge knowledge bases is challenging due to its intrinsic size and complexity. New methods for generating language are used in RAG models, which adds to their size and complexity [15-16]. When these models are scaled to handle large volumes of text, problems with model size, training time, and inference speed become more severe. Robust infrastructure and optimization strategies are necessary to manage and deploy large-scale models efficiently, minimize resource restrictions, and guarantee scalability.

Resource Constraints: There are extra difficulties when deploying RAG models in areas with limited resources, including edge devices or low-power computer systems. Scaling RAG models can be challenging because of their high computational and memory needs, which can be beyond the capability of such systems [17]. Modeling layouts, inference procedures, and resource use must all be optimized to satisfy operational requirements and preserve acceptable performance levels in order to strike a balance between efficiency and performance in contexts with limited resources.

The task of creating suitable assessment measures for scaled RAG models' performance evaluation is not easy. Careful thought is needed when balancing elements like retrieval accuracy, generation coherence, and computing economy. The subtleties of scaled RAG models may not be sufficiently captured by traditional evaluation measures, requiring the creation of innovative evaluation frameworks designed to meet the particular difficulties presented by large-scale databases of knowledge [18].

2.3 Previous Work on Model Compression

Prior research on the model's effectiveness and compression has concentrated on creating methods to lower the memory and processing demands of large-scale language models such as RAG. The goal of these initiatives is to preserve or even improve performance while making such models more manageable and lightweight for deployment in contexts with limited resources [19]. Many important strategies have been investigated.

1. **Knowledge Distillation**: In knowledge distillation, a minor, lighter model-(student) stands trained to emulate the actions of a more sophisticated, bigger model (teacher). Information distillation can dramatically lower the computational and memory footprint of the model while maintaining performance to an acceptable degree by transferring the information embedded in the teacher model onto the student model.

2. **Quantization**: Quantization methods with model parameters may be made less precise, usually from 32-bit floating-point values to lower precision forms like bfloat16 or 8-bit integers [20]. Without noticeably degrading performance, quantization can lower memory consumption and increase computational efficiency by encoding model weights and activations with fewer bits.
3. **Pruning**: Pruning reduces the size and processing needs of the model by locating and eliminating unnecessary connections or parameters [21]. Techniques for pruning can be used to target superfluous or less informative components while maintaining performance in the RAG model's retrieval and generation components.
4. **Sparsity and Sparsity Patterns**: In order to lower memory footprint and computational complexity, sparsity approaches take advantage of the natural sparsity found in neural network parameters. Sparsity patterns can be introduced into the model's parameters by applying sparsity constraints during training or post-training pruning, which will result in more effective computation and storage [22].
5. **Knowledge Representation Compression**: In order to reduce the amount of memory needed to store knowledge graphs while maintaining their semantic richness, techniques like knowledge graph compression are used to express structured information more compactly. Large knowledge bases can be stored and accessed with less overhead if compression methods designed specifically for knowledge graphs are used.
6. **Model Architecture Optimization**: Enhancing the RAG model's architecture itself can help increase efficiency. Model distillation, network slimming, and architecture search are a few of the techniques that are used to create more effective and lightweight model architectures that are suited to specific objectives and limitations [23].
7. **Hybrid Retrieval-Generation Approaches**: Improved efficiency and scalability can be obtained by hybrid systems that more closely integrate retrieval and generating processes. Through combined retrieval and generation component optimization, these strategies seek to lower redundancy and boost overall system efficiency.

Through the utilization of these approaches and the investigation of new strategies for model efficiency and compression [24]. The investigators want to increase the usability and accessibility of large-scale models for case RAG for a broader range of applications and deployment situations.

METHODOLOGY

3.1RAG Model Framework

The proposed framework, Retrieval-augmented generation (RAG), extracts the data from Wikipedia and other sources, designs the framework for retrieval of data, and addresses a critical issue faced by large language models (LLMs) known as illusions in large-scale knowledge-based systems. Visions occur when an LLM generates random or irrelevant facts from its training data, even when it lacks a genuine connection to the user's query. With the help of using LLMs, users often struggle to acknowledge when they lack information, making it challenging for users to discern the accuracy of their responses.

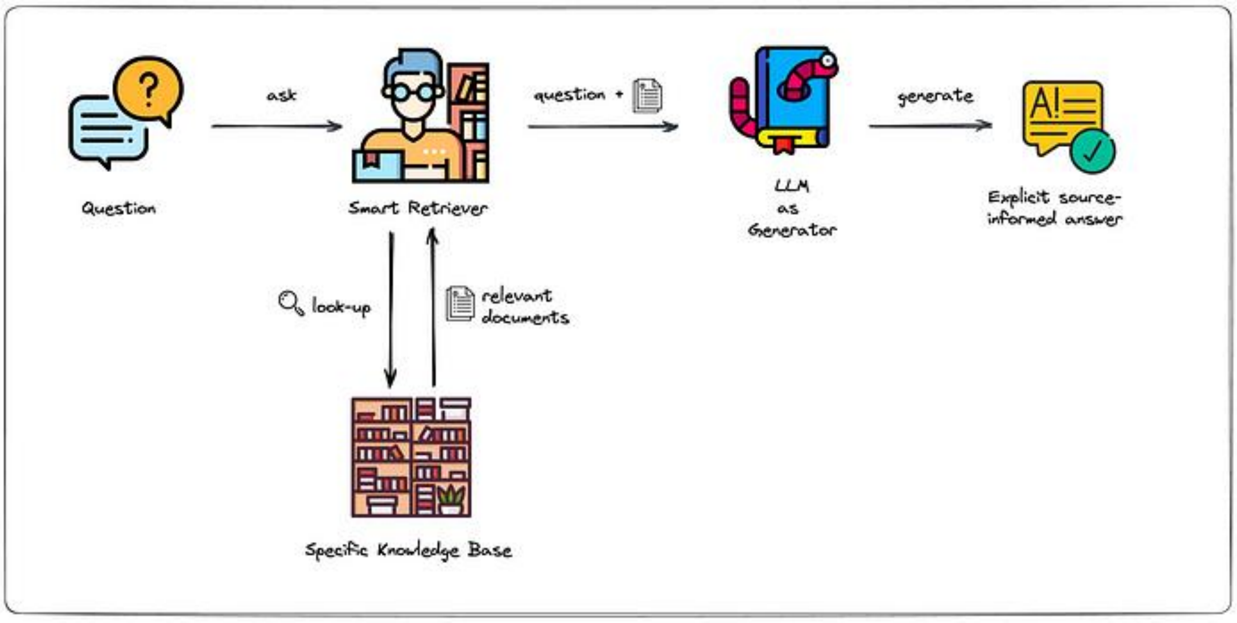


Figure 3: RAG Proposed Framework

Through the integration of the model with an external knowledge source and the provision of the model's reference materials for fact-checking, RAG seeks to improve the quality of LLM-generated replies. While using a retriever component, this system gets relevant documents from a knowledge base in response to user queries. The LLM is then given these documents and the query so that it may provide contextually aware replies that are in line with the data that was taken out of the papers that were supplied.

$$p_{\text{RAG-Token}}(y|x) \approx \prod_{i}^{N} \sum_{z \in \text{top-}k(p(\cdot|x))} p_\eta(z|x) p_\theta(y_i|x, z, y_{1:i-1})$$

The Querying Retrieval Model, known as the Dense Passage Retriever (DPR), operates by encoding documents into dense representations using a document encoder, represented as 'd'. In queries, they are encoded into symbols using a query encoder, represented as 'q(x).' Employing these encodings, DPR directories passages in a low-dimensional and continuous space

LLMs can use bespoke datasets for information retrieval thanks to RAG's facilitation of integration with private data sources. RAG improves the model's capacity to produce pertinent and correct replies in a variety of areas by linking LLMs to a wide range of knowledge bases, including proprietary datasets.
The workflow of RAG is depicted below:

- The user submits a query to the LLM.
- The query is passed to a retriever component.
- The retriever recovers related documents from the knowledge base founded on the query.
- The retrieved documents, along with the query-encoder, are forwarded to L-L-M.
- The LLM produces a response informed by the content extracted from the provided documents.

To summarize, RAG utilizes the advantages of both LLMs and external knowledge sources to produce replies that are more accurate and contextually relevant, all the while enabling users to double-check the information the model provides.

3.2 Knowledge Distillation for Model Size Reduction

Knowledge-Distillation (KD) is a well-known method that has surfaced to solve the issues with large language models (LLMs). Large-scale and sophisticated learning modules (LLMs) can become computationally demanding, requiring significant resources for training and implementation [25]. Their enormous memory footprint makes deployment difficult in contexts with limited resources.
Knowledge distillation reduces the size of LLMs without sacrificing performance, which helps to address these problems. Transferring information from a bigger, more complicated model frequently referred to as the "teacher" ideal to a naiver, well-lit models-habitually referred to as the "student" model is the basic concept of KD. KD makes it possible to create concise, effective learning outcomes by transferring the crucial knowledge acquired by the instructor to the student models.
The workflow of Knowledge Distillation for Model Size Reduction within the RAG framework can be outlined as follows:

1. **Teacher Model Selection**: A suitable pre-trained LLM, proficient in generating high-quality replies, is selected as the teacher model.
2. **Student Model Design**: A lesser, extra lightweight model architecture is intended, tailored to the requirements of the retrieval-augmented generation task.
3. **Knowledge Transferred**: The teacher model knowledge is transferred to the student prototypical through a process of distillation. This involves exercising the student model to mimic the performance and production of the teacher model using methods such as corresponding soft target probabilities or feature representations.
4. **Fine-tuning**: The model of std is fine-tuned on a specific task of retrieval-augmented generation, utilizing the knowledge distilled from the teacher model to enhance its enactment in producing contextually informed responses.
5. **Evaluation and Deployment**: Model performance should be assessed from time to time to make sure the distilled student model continues to perform comparably to the instructor model in producing pertinent and correct replies [26]. After it has been verified, the student model may be implemented into the RAG framework, which will minimize the system's total size and computational complexity without compromising its usefulness and performance.

Organizations may achieve notable enhancements in resource efficiency and scalability through the utilization of Knowledge Distillation for Model Size Reduction in the RAG framework. This allows retrieval-augmented generation systems to be implemented in a more extensive array of settings and applications.

3.3 Quantization and Pruning Techniques for Model Compression

Quantization and pruning approaches have become effective tactics for model compression in the endeavor to optimize large language models (LLMs) for optimal deployment and resource consumption. By reducing LLM size and eliminating computational overhead, these strategies increase LLM suitability for deployment in situations with limited resources [27]. Let's examine the applications of quantization and trimming for LLMs:

3.3.1 Quantization-Techniques

The process of quantization entails lowering the precision of the model's parameters, usually after 32 bits, moving point values toward 8-bit integers or other lower precision forms. The model is easier to store and run because of this decrease in accuracy, which also significantly reduces the computational complexity and memory footprint [28]. Quantization may be used for the weights and activations of the model in LLMs during inference.

- **Post-training Quantization:** After training is over, this approach quantizes the activations and weights of a pre-trained LLM. Significant compression is made possible without compromising model performance using post-training quantization, which maps the floating-point data to a discrete collection of quantized values.
- **Dynamic Quantization**: During inference, dynamic quantization quantizes the weights and activations of the model according to the observed range of values encountered. This method is especially well-suited for implementation in memory-constrained contexts as it minimizes the requirement for maintaining quantization parameters.

3.3.2 Pruning-Techniques:

Pruning is the process of eliminating unnecessary or unimportant connections (weights or neurons) within the model in a directive to reduce the number of parameters and scope of the model overall [29]. A variety of granularities, from individual weights to whole neurons or layers, can be pruned.

- **Magnitude-based Pruning:** This method efficiently prunes connections that add very little to the model's output by identifying and eliminating weights whose magnitudes fall below a predetermined threshold. Magnitude-based pruning delivers large compression with low-performance loss by continually thinning and fine-tuning the model.
- **Structured Pruning:** This technique removes whole neurons, channels, or layers based on how crucial they are to the functionality of the model. Structured pruning allows for more aggressive compression while maintaining the structural integrity of the model by removing entire structures compared to particular weights.

- **Iterative-Pruning and Fine Tunings**: The iterative method alternates between pruning and fine-tuning the model to offset the enactment loss from pruning [30]. Iterative pruning and fine-tuning enable better compression rates while retaining performance by progressively deleting and retraining the model.

Even further compression and efficiency advantages can be obtained by combining the approaches of quantization and pruning [31-32]. Organizations may further minimize the memory footprint and computational complexity of LLMs, increasing their suitability for deployment in a range of applications and situations by applying quantization to the trimmed model.

3.4Statistical tools and econometric models

This section elaborates on the proper statistical/econometric/financial models that are being used to forward the study from data towards inferences. The details of the methodology are given below.

3.4.1 Descriptive Statistics

Descriptive statistics has been used to find the maximum, minimum, standard deviation, mean, and regular distribution of the data of all the variables of the study. The normal distribution of data shows the sensitivity of the variables to periodic changes and speculation. When the data is not normally distributed, it means that the data is sensitive to periodic changes and speculations, which creates the chances of arbitrage. The investors have the chance to earn above the expected profit. However, the APT assumes that there should not be arbitrage in the market, and the investors can only earn the expected profit. Jarque Bera test is used to test the normality of data.

3.4.2 Fama-Mcbeth two-pass regression

After the test statistics, the methodology follows the next step in order to test the asset pricing models. When testing asset pricing models related to risk premium on assets to their betas, the primary question of interest is whether the beta risk of a particular factor is priced. Fama and McBeth(1973)develop a two-pass methodology in which the beta of each asset with respect to a factor is estimated in a first-pass time series regression, and estimated betas are then used in second-pass cross-sectional regression to estimate the risk premium of the factor. According to Blum (1968), testing two-parameter models presents an unavoidable errors-in-the-variables problem. It is important to note that portfolios (rather than individual assets) are used to make the analysis statistically feasible. Fama McBeth regression is used to attenuate the problem of errors-in-variables (EIV) for two parameter models (Campbell, Lo, and MacKinlay, 1997). If the errors in the β (beta)of individual security are not perfectly positively correlated, the β of portfolios can be much more precise estimates of the truth.

The study follows Fama and McBeth's two-pass regression to test these asset pricing models. The Durbin Watson is used to check serial correlation and measure the linear association between adjacent residuals from a regression model. If there is no serial correlation, the DW statistic will be around 2. The DW statistic will fall if there is a positive serial correlation (in the worst case, it will be near zero). If there is a negative correlation, the statistics will lie somewhere between 2 and 4. Usually, the limit for non-serial correlation is considered to be DW, which ranges from 1.8 to 2.2. A robust positive serial correlation is considered at DW lower than 1.5.

To make the model more effective and efficient, the selection criteria for the shares in the period are Shares with no missing values in the period, Shares with adjusted $R^2 < 0$ or F significant (p-value) >0.05of the first pass regression of the excess returns on the market risk premium are excluded. Furthermore, shares are grouped alphabetically into a group of 30 individual securities.

3.4.2.1 Model for CAPM

In the first pass, the linear regression is used to estimate beta, which is the systematic risk.

$$R_i - R_f = (R_m - R_f)\beta \qquad (3.1)$$

Where $R_i$isMonthly return of the security, R_f isMonthly risk-free rate, R_m isMonthly return of the market, and β is systematic risk (market risk).

The excess returns R_i - R_f of each security are estimated from a time series share prices of KSE-100 index listed shares for each period under consideration. For the same period, the market Premium Rm - Rf is also estimated. After that, the excess returns Ri - Rf on the market premium Rm - Rf are regressed to find the beta coefficient (systematic risk).

Then, cross-sectional regression or second-pass regression is used to calculate the average excess returns of the shares and estimated betas.

$$\hat{R}_i = \gamma_0 + \gamma_1\beta_1 + \epsilon \qquad (3.2)$$

Where $ƛ_0$= intercept, $\hat{R}_I$is average excess returns of security i,$\beta_I$isestimated be coefficient of security I, and Є is the error term.

3.4.2.2 Model for APT

In the first pass, the betas coefficients are computed using regression.

$$R_i - R_f = \beta_i f_1 + \beta_{i2} f_2 + \beta_{i3} f_3 + \beta_{i4} f_4 + \epsilon \qquad (3.3)$$

Where Ri is the monthly return of stock i, R_f is the risk-free rate, β_i is the sensitivity of stock *i* with factors, and ϵ is the error term.
Then, a cross-sectional regression or second-pass regression is used to calculate the average excess returns of the shares on the factor scores.

$$\hat{R} = \gamma_0 + \gamma_1\beta_1 + \gamma_2\beta_2 + \gamma_3\beta_3 + \gamma_4\beta_4 + \epsilon_i \qquad (3.4)$$

Where$\hat{R}$ is the average monthly excess return of stock I, ƛ = risk premium, β_1 to β_4 are the factors scores, and ε_i is the error term.

3.4.3 Comparison of the Models

The next step of the study is to compare these competing models to evaluate which one of these models is more supported by data. This study follows the methods used by Chen (1983), the Davidson and Mackinnon equation (1981), and the posterior odds ratio (Zellner, 1979) for comparison of these Models.

3.4.3.1 Davidson and MacKinnon Equation

CAPM is considered the particular or strict case of APT. These two models are non-nested because by imposing a set of linear restrictions on the parameters, the APT cannot be reduced to CAPM. In other words, the models do not have any common variables. Davidson and MacKinnon (1981) suggested the method to compare non-nested models. The study used the Davidson and MacKinnon equation (1981) to compare CAPM and APT.
This equation is as follows;

$$R_i = \alpha R_{APT} + (1-\alpha)R_{CAPM} + e_i \qquad (3.5)$$

Where$R_{i=}$ the average monthly excess returns of the stock i, $R_{APT=}$ expected excess returns estimated by APT, $R_{CAPM=}$ expected excess returns estimated by CAPM, and α measures the effectiveness of the models. The APT is the accurate model to forecast the returns of the stocks as compared to CAPM if α is close to 1.

3.4.3.2 Posterior Odds Ratio

A standard assumption in theoretical and empirical research in finance is that relevant variables (e.g., stock returns) have multivariate normal distributions. Given the assumption that the residuals of the cross-sectional regression of the CAPM and the APT satisfy the IID (Independently and identically distribution) multivariate typical assumption (Campbell, Lo, and MacKinlay, 1997), it is possible to calculate the posterior odds ratio between the two models. In general, the posterior odds ratio is a more formal technique than the DM equation and has sounder theoretical grounds.
The second comparison is done using a posterior odd radio. The formula for posterior odds is given by Zellner (1979) in favor of model 0 over model 1.
The formula has the following form;

$$R = [ESS_0/ESS_1]^{N/2} N^{K_0 - K_1/2} \qquad (3.6)$$

WhereESS_0iserror sum of squares of APT, ESS_1iserror sum of squares of CAPM, Nisnumber of observations, $K_0$is number of independent variables of the APT and K_1 isnumber of independent variables of the CAPM. According to the ratio, when;
R> 1 means CAPM is more strongly supported by the data under consideration than APT.
R < 1 means the data under consideration more strongly supports APT than CAPM.

4. RESULTS AND DISCUSSION

A number of important metrics are evaluated while evaluating the quantization and pruning methods used on large language models (LLMs) in order to compress the model. First and foremost, it is necessary to quantify the effect on model size reduction and computational efficiency, taking into account decreases in memory footprint and inference delay. Furthermore, the assessment needs to take into account the compromise between the compression ratio and the performance of the model, scrutinizing any decline in precision or efficacy of the compressed model in contrast to the original. To guarantee generalizability, it is also necessary to assess the compressed model's stability and resilience over a range of tasks and datasets. Lastly, evaluating the overall efficacy and viability of the quantization and pruning strategies for LLM compression requires careful consideration of practical factors, including deployment compatibility and ease of implementation.

4.1 Dataset Loading/Inspection

A number of processes are involved in importing and examining the dataset in order to comprehend the structure and content of the Wikipedia Crypto Articles dataset. Two columns are shown when the dataset is first imported into a pandas DataFrame: "title" and "article." To maintain data integrity, nine items that were found to contain empty articles have been removed. The length of the data frame is then verified both before and after the empty items are removed, yielding 227 rows at the beginning and 218 rows at the end of the cleanup process.

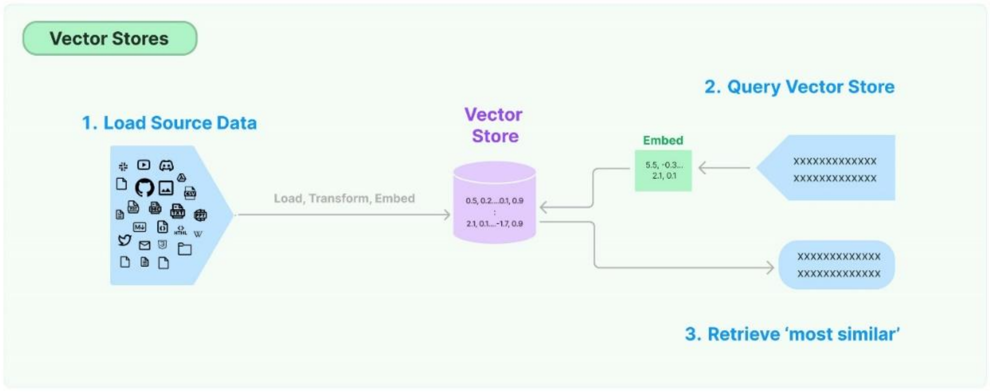


Figure 4: Data loading and inspection
When the contents of the dataset are examined, the last record related to the NEO cryptocurrency is shown, which includes comprehensive details on its technical characteristics, background, and references. This data sets the stage for further processing and analysis by offering insightful information on the kind and extent of the articles included in the dataset. Hugging-Face-Embedding is used to load the sentence-transformers/all-Mini-LM-L6-v2-model, which maps sentences and paragraphs towards a dense vector space. The purpose of this model is to generate embedded data for text data, simplifying similarity-based retrieval and analysis.

4.2 Loading Mistral 7b Model

Using state-of-the-art language models aimed at text generation tasks requires loading the Mistral 7B model, which is a crucial step. By using the Hugging Face Pipeline class, we may have access to an extensive collection of more than 120,000 free models that are stored on Huggingface. In particular, we choose the mistrial/Mistral-7B-v0.1 model, which is well-known for its outstanding output in the text-generation field. By far the most potent open-source model as of January 2024, the Mistral-7B model outperforms the Meta-Llama 2-13-B model on all benchmarks, boasting an astounding 7.3 billion parameters.

1. Querying Model

During the querying technique, a series of questions designed to retrieve pertinent data are used to assess the model's capabilities. The QnA chain is rummage-sale in combination with the get answers functions near ask-model questions and receive answers that demonstrate the model's comprehension and expertise. Through this exchange, the model shows that it can adequately answer inquiries on the history of cryptocurrencies, well-known frauds, prominent members of the crypto community, and other topics. Although the model has impressive performance in particular domains, such as accurately identifying the founders of Ethereum and Bitcoin, there are certain areas where it needs to be improved, like the names being repeated in answer to specific requests.
In order to retrieve responses from the model in response to user questions, the get_answers function with the established QnA chain is used in the querying process. The following table lists the questions the model was asked and the responses it received:

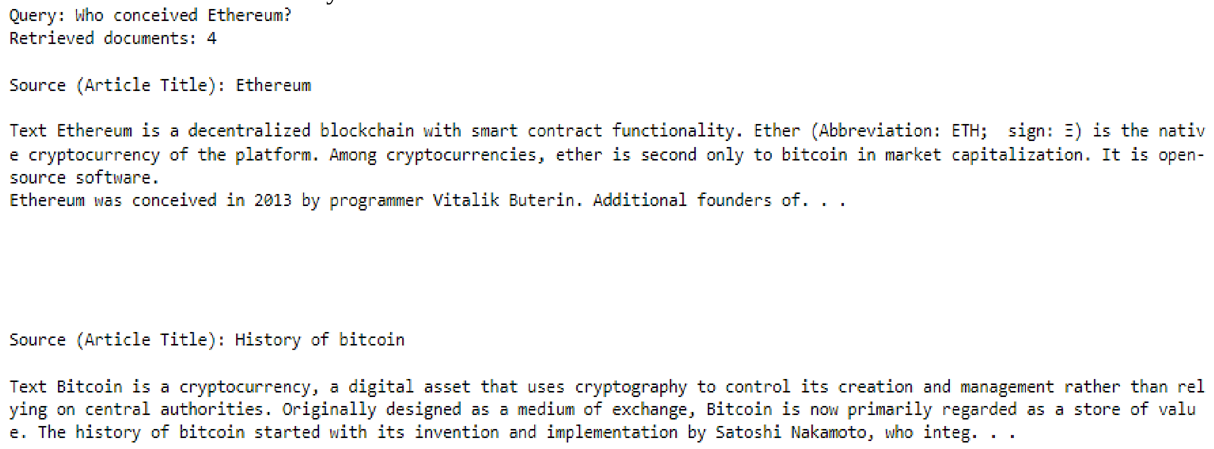
Table 1: Program demonstration of query retrieval process

Query	**Answer**
Who created the Bitcoin? When was it created?	Bitcoin was produced by an unidentified individual or group of persons using the designation Satoshi-Nakamoto. It was created in 2009.
What was the biggest scam in the history of crypto-currencies?	The biggest scam in the history of cryptocurrencies was the 2021 Squid Game cryptocurrency scam.
How much will one Bitcoin cost in 2030?	I do not know.
Cite the names of five relevant people in crypto.	1. Andreas Antonopoulos 2. Brian Armstrong 3. Changpeng Zhao 4. Andreas Antonopoulos 5. Andreas Antonopoulos
What exchanges can I use to buy crypto?	Crypto.com is a cryptocurrency exchange that offers an initial exchange offering (IEO) for various cryptocurrencies. It is available in Europe and other parts of the world.
Who conceived Ethereum?	Vitalik Buterin conceived Ethereum.

The program demonstrates an impressive level of comprehension by correctly recognizing the inventors of Ethereum and Bitcoin, as well as offering insights about prominent cryptocurrency scams and significant figures in the industry. Still, certain areas may use work, such as the reiteration of names in answers to questions and the lack of insight when questioned about potential future pricing for Bitcoin. However, the model's capacity to obtain pertinent data from the vector store in response to user inquiries improves its usefulness in offering educational answers.

4.3 Retrieval Speed

Retrieval speed, as used in this research, is the speed at which the model can access and retrieve pertinent documents from the vectorized database that has been indexed. In order to retrieve documents with relevant information, the database is queried in response to user queries. In this instance, the model's job is to find articles regarding particular cryptocurrency subjects, such as Ethereum's inception or details on cryptocurrency exchanges. The model's retrieval speed directly impacts the entire responsiveness and user experience. Adequate retrieval speed guarantees that users get answers to their questions quickly, which improves the model's usefulness and efficacy.



```
Query: Who conceived Ethereum?
Retrieved documents: 4

Source (Article Title): Ethereum

Text Ethereum is a decentralized blockchain with smart contract functionality. Ether (Abbreviation: ETH;  sign: Ξ) is the nativ
e cryptocurrency of the platform. Among cryptocurrencies, ether is second only to bitcoin in market capitalization. It is open-
source software.
Ethereum was conceived in 2013 by programmer Vitalik Buterin. Additional founders of. . .



Source (Article Title): History of bitcoin

Text Bitcoin is a cryptocurrency, a digital asset that uses cryptography to control its creation and management rather than rel
ying on central authorities. Originally designed as a medium of exchange, Bitcoin is now primarily regarded as a store of valu
e. The history of bitcoin started with its invention and implementation by Satoshi Nakamoto, who integ. . .
```

Figure 5: Source Article data retravel queries

4.4 Memory Usage
Memory utilization describes how much memory is used by the model when it is retrieving and creating data. Factors like the size of the indexed vectorized database, the intricacy of the retrieval queries, and the computing demands of producing replies all affect how much memory the model uses in the particular scenario. Maintaining peak performance and avoiding resource limitations that might cause slowdowns or system breakdowns require efficient memory utilization. The model can facilitate a good user experience by guaranteeing smooth operation and seamless user interaction through efficient memory resource management.

4.5 Generation Quality
The precision, applicability, and consistency of the responses produced by the model with the information it obtained are referred to as generation quality.

- The research presented assesses the generation quality of the model by looking at how well it generates contextually appropriate and informative responses to user questions.
- When queried about the origins of Ethereum, the model should produce a response that correctly credits Vitalik Buterin as the project's developer and offers other pertinent information.

When asked about bitcoin exchanges, the model needs to produce answers that include a list of reliable exchanges together with pertinent details about how they operate. A high-generation quality model will eventually increase its usefulness and efficacy as a tool for knowledge production and retrieval by fostering user confidence in its skills.

5 Conclusion
The report offers a thorough review of several methods and frameworks designed to improve large language models' (LLMs') capability, performance, and efficiency when applied to information creation and retrieval activities. By investigating methods such as Knowledge Distillation (KD), Quantization/Pruning, and Retrieval-Augmented Generation (RAG), businesses may explore state-of-the-art approaches to maximize LLM deployment and performance in a variety of settings and applications. By incorporating external information sources and enabling contextually informed generation, RAG emerges as a potent paradigm for enhancing LLM-generated answers. RAG improves the relevance and accuracy of replies by facilitating smooth communication between users and LLMs and giving users the ability to independently check information by accessing reference resources. Applying Knowledge Distillation provides a workable way to deal with considering the memory requirements and computational complexity of LLMs. KD makes it possible to install more effective and scalable models inside the RAG framework by reducing the size of LLMs while maintaining performance. Techniques for quantization and trimming are essential for optimizing LLMs in situations with limited resources. These strategies improve the efficiency and accessibility of LLMs while preserving excellent generation quality by lowering memory utilization and computational overhead.

6 ACKNOWLEDGEMENT
We extend our deepest gratitude to Dr. Steven Brown, Dr. Michalina Hendon, Dr. Nathan Hamblin and Dr. Machica Mcclain at the University of the Cumberlands for their invaluable support and guidance throughout our research. Their expertise, encouragement, and constructive feedback have been instrumental in shaping this work. We could not have completed our research journey without their continuous encouragement and insightful mentorship. Thank you for your unwavering commitment to our success.

REFERENCES
[1]. Suresh, Karthik, Neeltje Kackar, Luke Schleck, and Cristiano Fanelli. "Towards a\textbf {RAG}-based Summarization Agent for the Electron-Ion Collider." *arXiv preprint arXiv:2403.15729* (2024).
[2]. Gao, Yunfan, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. "Retrieval-augmented generation for large language models: A survey." *arXiv preprint arXiv:2312.10997* (2023).
[3]. Lyu, Yuanjie, Zhiyu Li, Simin Niu, Feiyu Xiong, Bo Tang, Wenjin Wang, Hao Wu, Huanyong Liu, Tong Xu, and Enhong Chen. "CRUD-RAG: A comprehensive chinese benchmark for retrieval-augmented generation of large language models." *arXiv preprint arXiv:2401.17043* (2024).
[4]. Tang, Yixuan, and Yi Yang. "Multihop-rag: Benchmarking retrieval-augmented generation for multi-hop queries." *arXiv preprint arXiv:2401.15391* (2024).
[5]. Bhardwaj, Rohit, Saurabh Srivastava, Hari Govind Mishra, and Sumit Sangwan. "Exploring micro-foundations of knowledge-based dynamic capabilities in social purpose organizations." *Journal of Knowledge Management* 27, no. 4 (2023): 1016-1041.
[6]. Wang, Huan, Yan-Fu Li, and Min Xie. "Empowering ChatGPT-Like Large-Scale Language Models with Local Knowledge Base for Industrial Prognostics and Health Management." *arXiv preprint arXiv:2312.14945* (2023).
[7]. Zheng, Mingxue, Xiangcheng Shen, Zhiqing Luo, Pingting Chen, Bo Guan, Jicheng Yi, and Hairong Ma. "A Knowledge-Based Multi-Scale Adaptive Classification Approach for Mobile Laser Scanning Point Clouds in Urban Scenes." *IEEE Access* 11 (2023): 134531-134546.
[8]. Li, Jiarui, Ye Yuan, and Zehua Zhang. "Enhancing LLM Factual Accuracy with RAG to Counter Hallucinations: A Case Study on Domain-Specific Queries in Private Knowledge-Bases." *arXiv preprint arXiv:2403.10446* (2024).
[9]. Siriwardhana, Shamane, Rivindu Weerasekera, Elliott Wen, Tharindu Kaluarachchi, Rajib Rana, and Suranga Nanayakkara. "Improving the domain adaptation of retrieval augmented generation (RAG) models for open domain question answering." *Transactions of the Association for Computational Linguistics* 11 (2023): 1-17.
[10]. Li, Jianquan, Xidong Wang, Xiangbo Wu, Zhiyi Zhang, Xiaolong Xu, Jie Fu, Prayag Tiwari, Xiang Wan, and Benyou Wang. "Huatuo-26m, a large-scale chinese medical qa dataset." *arXiv preprint arXiv:2305.01526* (2023).

[11]. Gao, Yunfan, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. "Retrieval-augmented generation for large language models: A survey." *arXiv preprint arXiv:2312.10997* (2023).
[12]. Shi, Yucheng, Shaochen Xu, Zhengliang Liu, Tianming Liu, Xiang Li, and Ninghao Liu. "Mededit: Model editing for medical question answering with external knowledge bases." *arXiv preprint arXiv:2309.16035* (2023).
[13]. Wu, Jialin, Jiasen Lu, Ashish Sabharwal, and Roozbeh Mottaghi. "Multi-modal answer validation for knowledge-based vqa." In *Proceedings of the AAAI conference on artificial intelligence*, vol. 36, no. 3, pp. 2712-2721. 2022.
[14]. Das, Rajarshi, Manzil Zaheer, Dung Thai, Ameya Godbole, Ethan Perez, Jay-Yoon Lee, Lizhen Tan, Lazaros Polymenakos, and Andrew McCallum. "Case-based reasoning for natural language queries over knowledge bases." *arXiv preprint arXiv:2104.08762* (2021).
[15]. Jeong, Soyeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C. Park. "Adaptive-RAG: Learning to Adapt Retrieval-Augmented Large Language Models through Question Complexity." *arXiv preprint arXiv:2403.14403* (2024).
[16]. Li, Yu, Baolin Peng, Yelong Shen, Yi Mao, Lars Liden, Zhou Yu, and Jianfeng Gao. "Knowledge-grounded dialogue generation with a unified knowledge representation." *arXiv preprint arXiv:2112.07924* (2021).
[17]. Saxena, Apoorv, Aditay Tripathi, and Partha Talukdar. "Improving multi-hop question answering over knowledge graphs using knowledge base embeddings." In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pp. 4498-4507. 2020.
[18]. Marino, Kenneth, Xinlei Chen, Devi Parikh, Abhinav Gupta, and Marcus Rohrbach. "Krisp: Integrating implicit and symbolic knowledge for open-domain knowledge-based vqa." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14111-14121. 2021.
[19]. Yang, Zhengyuan, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Yumao Lu, Zicheng Liu, and Lijuan Wang. "An empirical study of gpt-3 for few-shot knowledge-based vqa." In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 3, pp. 3081-3089. 2022.
[20]. Sourty, Raphaël, Jose G. Moreno, François-Paul Servant, and Lynda Tamine. "Knowledge base embedding by cooperative knowledge distillation." In *International Conference on Computational Linguistics (COLING 2020)*, pp. 5579-5590. International Committee on Computational Linguistics, 2020.
[21]. Mohamed, Ali Wagdy, Anas A. Hadi, and Ali Khater Mohamed. "Gaining-sharing knowledge based algorithm for solving optimization problems: a novel nature-inspired algorithm." *International Journal of Machine Learning and Cybernetics* 11, no. 7 (2020): 1501-1529.
[22]. Hallinger, Philip, Sedat Gümüş, and Mehmet Şükrü Bellibaş. "'Are principals instructional leaders yet?'A science map of the knowledge base on instructional leadership, 1940–2018." *Scientometrics* 122, no. 3 (2020): 1629-1650.
[23]. Aldweesh, A., Derhab, A. and Emam, A.Z., 2020. Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues. *Knowledge-Based Systems*, *189*, p.105124.
[24]. Larkin, Aoife, Steven J. Marygold, Giulia Antonazzo, Helen Attrill, Gilberto Dos Santos, Phani V. Garapati, Joshua L. Goodman et al. "FlyBase: updates to the Drosophila melanogaster knowledge base." *Nucleic acids research* 49, no. D1 (2021): D899-D907.
[25]. Lacroix, Timothée, Guillaume Obozinski, and Nicolas Usunier. "Tensor decompositions for temporal knowledge base completion." *arXiv preprint arXiv:2004.04926* (2020).
[26]. Tang, Jiaxi, Rakesh Shivanna, Zhe Zhao, Dong Lin, Anima Singh, Ed H. Chi, and Sagar Jain. "Understanding and improving knowledge distillation." *arXiv preprint arXiv:2002.03532* (2020).
[27]. Mirzadeh, Seyed Iman, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. "Improved knowledge distillation via teacher assistant." In *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 04, pp. 5191-5198. 2020.
[28]. Chen, Defang, Jian-Ping Mei, Hailin Zhang, Can Wang, Yan Feng, and Chun Chen. "Knowledge distillation with the reused teacher classifier." In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11933-11942. 2022.
[29]. Li, Tianhong, Jianguo Li, Zhuang Liu, and Changshui Zhang. "Few sample knowledge distillation for efficient network compression." In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14639-14647. 2020.
[30]. Aguilar, Gustavo, Yuan Ling, Yu Zhang, Benjamin Yao, Xing Fan, and Chenlei Guo. "Knowledge distillation from internal representations." In *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 05, pp. 7350-7357. 2020.
[31]. Huang, Tao, Shan You, Fei Wang, Chen Qian, and Chang Xu. "Knowledge distillation from a stronger teacher." *Advances in Neural Information Processing Systems* 35 (2022): 33716-33727.
[32]. Ji, Mingi, Seungjae Shin, Seunghyun Hwang, Gibeom Park, and Il-Chul Moon. "Refine myself by teaching myself: Feature refinement via self-knowledge distillation." In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10664-10673. 2021..