Performance Evaluation of Vector Embeddings with Retrieval-Augmented Generation

Sanjay Kukreja
Department of Machine Learning
SP Jain School of Global Management
Mumbai, India
sanjay.ds18dba008@spjain.org

Tarun Kumar
COE AI-ML
eClerx Services Ltd.
Chandigarh, India
tk7ua1678@gmail.com

Vishal Bharate
COE AI-ML
eClerx Services Ltd.
Pune, India
vishalbharate@gmail.com

Amit Purohit
COE AI-ML
eClerx Services Ltd.
Mumbai, India
amit.iitgn@gmail.com

Abhijit Dasgupta
Department of Data Science
SP Jain School of Global Management
Mumbai, India
abhijit.dasgupta@spjain.org

Debashis Guha
Department of Machine Learning
SP Jain School of Global Management
Mumbai, India
debashis.guha@spjain.org

2024 9th International Conference on Computer and Communication Systems (ICCCS) | 979-8-3503-5021-0/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICCCS61882.2024.10603291

***Abstract*—Vector embeddings form the basis of sophisticated language models. These language models were developed with the advent of developments in natural language processing (NLP) and aid in a variety of downstream tasks. Contextually relevant responses are improved using a combination of generation-based models and information retrieval, which is combined in the Retrieval-Augmented Generation (RAG) framework. The state-of-the-art research focuses on the RAG framework. Performance evaluation of vector embeddings in context with the RAG framework for data querying from documents is presented in this paper. The research encompasses a comparative analysis of various vector embeddings and their average, weighted average ensemble, evaluating their effectiveness in easing information retrieval and subsequent generation activities. The investigations focus on the impact of alternative embedding approaches on the overall performance of context generation across the NCERT books dataset using a systematic evaluation. The capabilities of ChatGPT and Llama2 are employed for evaluating the performance of embedding models. NCERT books form the underlying database, and LLM models are used to rank the contexts derived from the database. The optimized prompt is utilized to achieve ranking of the results. The same LLM is also utilized to generate the response for all the embedding models employing generated contexts. The variety in vector embedding approaches is exhibited by the experimental results.**

Keywords—Natural Language Processing, Vector Embeddings, Retrieval-Augmented Generation, Ensemble embedding, Data querying, Performance Evaluation, Information Retrieval, Response Generation.

I. INTRODUCTION

The evolution of sophisticated language models and natural language processing (NLP) techniques has undergone dramatic developments in recent years. Vector embeddings form the basis of these methods that represent documents, sentences and words as continuous numerical vectors. These vectors form the underlying database for efficient capture and language processing. The challenges in understanding the language are efficiently handled by the Retrieval-Augmented Generation (RAG) [34] framework that has the capability of utilizing generation and retrieval-based approaches.

Natural language processing experienced a paradigm shift with the emergence of the RAG framework. RAG models utilize the processes of generation based as well as retrieval based models. The RAG mechanism is able to generate keyword-sensitive and coherent responses. The execution of RAG framework depends on the embedded representation of textual data in vector form and it is directly related to the variability and appropriateness of vector embeddings.

Different types of vector embeddings and their ensembles are studied thoroughly in current research and the evaluation of their performance is presented using the RAG framework. This study examines the impact of different forms of embedding systems in the overall RAG model system and their performance by adopting systematic experimentation and analysis.

In the current research we have determined the impact of vector embeddings on information retrieval and response generation component of RAG. We have evaluated and compared each embedding model and its ensemble systems to get the better insights and utilize it to make an appropriate response in context formation.

We have provided a thorough comparative study of attributes that are involved in the evaluation process as have used various evaluation metrics suitable for NLP task evaluation. In this paper, we have used the dataset from NCERT books and query RAG system in experiments to find the benefits and drawbacks of each embedding model. We have measured the performance of RAG system using metrics like speed of retrieval [41], throughput, precision, recall, f1-score [42], semantic answer similarity and Rouge score.

The major contribution of this research work is identification of better performing embedding models in retrieval and generation tasks. The hypothesis of this research work comprises of a) Ensemble method may outperform in retrieval task. b) Ensemble method can better perform in generation task. The novelty of this research work is formation of ensemble embedding model and their performance testing with the state of the art embedding models. Additionally, we aimed to employ innovative solution for ranking of the generated context using semantic search for open and closed source LLMs.

The subsequent sections of this paper deals with the literature review, experimental methodologies, system architecture, system setup, performance evaluation, insights derived from the findings and business applicability. The utilization of RAG framework is also viewed in context of business applicability across domains and presented in this paper. Ultimately, we have aimed in this research to extend information using embedding models, to give the RAG, which may offer the best business benefits.

II. Literature Review

Natural language processing means scaffolding computers and helping them to understand the human language which is very much like the way humans interact. It functions as a structured building block for numerous applications like translation, sentiment analysis, document querying and so forth. Vector embeddings, numerical representations of words, phrases or sentences in multidimensional spaces, plays a crucial role in NLP success [29]. These embeddings comprehend the semantic relatedness which can help machines to better interpret language[43].

Before 2017, the word embeddings alone were used to defeat the language simplification challenge, using human in the loop. Word embeddings [20] and LLMs have been used consistently for tackling language simplification challenge since 2017. Various word embeddings and LLMs, for example Word2vec, Sense2vec [1], Fasttext [2] and Bert were appropriately used for word substitutions. Alarcón et al. [3] investigated different word embedding models for producing Spanish candidate substitutions.

Surprisingly, this contradicts Paetzold and Specia's [4] forecast that word embedding models would perform better due to their cutting-edge reputation. During error analysis, it was revealed that these word embedding models commonly produced antonyms of the target complex word as probable candidate substitutions. This is due to the fact that word embedding approaches compute word similarity between vectors in different ways.

Seneviratne et al. [5] used a word embedding model and a pre-trained LLM: XLNet [6] to construct an embedding similarity score and a prediction score for SG. They utilized a technique similar to Arefyev et al. [7]. Arefyev et al. [7] encoded the context of the target complex word and generated a probability distribution for each word in that context using context2vec (Melamud et al., [8]) and ELMo [9]. This probability distribution was further used to assess the probability or appropriateness of a potential candidate substitution replacing the target complicated word. This score was combined with an LLM prediction score from BERT, RoBERTa, or XLNet to get a final list of top-k candidate substitutions. Seneviratne et al. [5] and Arefyev et al. [7] used combination of embedding models with pre-trained LLM model to get fair result.

A word embedding model is numerous portrayal of the word relatedness [10, 43] in a corpus from co-events in context [28]. This has been served as an essential asset with the consideration of a large number of researchers.

Like some other linguistic models, the word embeddings have demonstrated progress in different fields of NLP like named entity recognition [14, 21], part-of-speech tagging [16, 17, 21] and so forth. Word embeddings have also demonstrated potential in machine translation [15], search [7], document querying and recommendation [18, 19]. Similarly, there are various academic applications for embeddings, including enhancing search engines, improving NLP tasks for academic texts, and providing journal recommendations for publications. Business applications include document querying of large repository of documents pertaining to various business domains.

The majority of published studies has concentrated on generic content such as Wikipedia [15, 20] or informal writing such as reviews [13, 22] and tweets [19, 23]. To validate word embedding methods for academic texts that contain technical, scientific, or domain-specific information such as accurate definitions, acronyms, or chemical/mathematical formulas, authors linked articles to their appropriate publications and evaluated the embeddings. To quantify the match, they used the ranks created by ordering the similarity of embeddings between each article and all journals [4, 10].

Word2vec, Glove, and FastText vector embeddings learn from large amounts of text input to encode semantic similarities and relationships. Word2Vec, for example, employs word contexts to generate embeddings, whereas Glove is concerned with global word-to-word co-occurrence data. The NLP task is made easier with the help of embeddings that helps algorithms to understand language semantics[11].

However, difficulties remain in selecting the best embeddings for certain NLP tasks within RAG. Some embeddings may excel at capturing word-level semantics, whilst others may excel at capturing larger context [12]. Understanding these distinctions is crucial for optimizing the performance of vector embeddings inside the RAG framework.

III. System Architecture

The system architecture is presented in Fig. 1. The working of the system is divided into two parts. The large corpus from the selected dataset is fed as an input in the first part. The input is divided into smaller chunks (documents). These chunks are provided to the vector embedding model. This pre-trained vector embedding model will convert chunks into vectors. The generated vectors are stored in a vector database. The choice of selecting a vector database depends on the system requirements.

In the second part of the system architecture, the user will raise a query and expect an appropriate response concerning the context generated from the query. The query is also converted into vectors by the same vector embedding techniques used while creating the database through the embedding model.

Based on generated vectors, semantic results are searched in the vector database. Top k documents which share a semantic meaning are retrieved from the vector database. These documents act as a context for the large language model. The LLM will act as a brain that takes user queries regarding embedding vectors and top k document embedding vectors as input. Based on the query input and context provided by the system, LLM will generate meaningful responses.

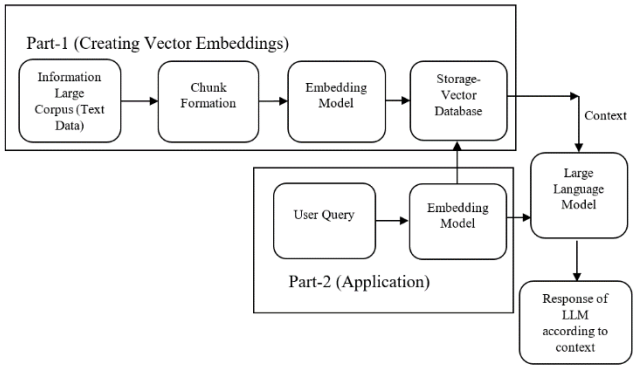

Fig. 1. System architecture

IV. METHODOLOGY

As per the system architecture discussed in the previous section, we chose different modules to carry out the experimentation smoothly. These modules and their selection criteria are discussed in this section.

A. Dataset Selection

We collected datasets from National Council of Educational Research and Training (NCERT) books [25] in order to retrieve context to produce a cohesive response from the RAG framework. The dataset contains multiple pdf files of different chapters for a subject. Such 2 subjects are selected for experimentation. We also collected solution sets [26], which contain questions and their solutions. With the aid of solution set we examine the quality of the response produced by the RAG framework.

B. Vector Embeddings

For experimentation, we choose variety of vector embedding models, including BAAI/be-base-en, thenlper/gte-base, intfloat/e5-base-v2. These embeddings record several levels of semantic and syntactic information, allowing for a more thorough assessment of their suitability within the RAG framework. Based on the performance on leader-board of Hugging Face [24], embedding models are chosen for experimentation.

- BAAI/bge-base-en [31]- This model can run on local machine. It exhibits input sequence length of 512 tokens and 768 embedding dimensions. It is having relatively smaller size of 0.44 GB. This model can be used for various NLP applications.
- thenlper/gte-base [32]- It is state of the art model which offers a much higher performance. It competes number of models available in the sentence transformer library. The embedding dimensions of 768 and 0.22GB size makes it a suitable choice for numerous NLP applications.
- intfloat/e5-base-v2 [33]- It is one of the pre-trained model which is specifically trained for English language. The model has a size of 0.44 GB with embedding dimensions of 768.
- Average Ensemble [38, 39]- We specifically selected these state of the art vector embedding models as the size of each model is the same. We made an ensemble of these models using the averaging technique. We used the addition of all three embedding vectors and divided it by the length of the embedding vector.
- Weighted Average Ensemble [40]- We also created weighted average ensemble by applying weights to selected embedding models. These weights are decided based on the performance of individual embedding models for all tests. We applied [0.4, 0.3, 0.3] weight for BAAI/bge-base-en, thenlper/gte-base and intfloat/e5-base-v2 respectively.

C. Selection of Vector Database

We surveyed a variety of vector databases, as various available databases are categorized as self-hosted, cloud-based, serverless, or client-server-based. We selected the Chroma vector database [27], as it is self-hosted and serverless. It enables us to experiment with different vector embeddings on a single platform. The multiple collections are created in order to test the performance of all vector embeddings and their ensemble in tandem.

For semantic search Chroma vector database offers these distance metrics namely, Squared Euclidian (L2), Cosine and Inner Product. Mathematically L2 is computed using (1). Where q is the query vector and d is the vector present in vector database.

$$L2 = \sum_{k=0}^{n} (q_k - d_k)^2 \quad (1)$$

Equation (2) represents cosine metrics for semantic search. Mathematically it is represented as shown in (2).

$$\cos(q, d) = \frac{q.d}{\|q\|\|d\|} \quad (2)$$

Another powerful tool to search for semantic results is dot product. Mathematically it is presented in (3).

$$q.d = \sum_{k=1}^{n} q_k d_k \quad (3)$$

We used these metrics to get more insights in selected embedding models.

D. RAG Model Configuration

We created the RAG model configuration by incorporating the chosen vector embeddings into the retrieval and creation modules. We configure the RAG architecture to get relevant

information from a knowledge source, which is the Chroma database, efficiently and provide coherent responses based on the context retrieved from the LLM model.

- GPT 3.5 [35]– We selected the GPT 3.5 LLM model, an LM with a decoder-only architecture. It can handle a 16k token context. It supports more than 50 languages such as English, French, Italian, German, Spanish and more. It can be fine-tuned into an instruction-following model. It matches or beats various LLMs, thus we chose it for our RAG model configuration.

Llama2 [36]- It is open source large language model. Llama2 excels in various natural language processing tasks, including text completion, question answering, and content generation. We consciously selected llama70b-v2-chat model for experimentation.

V. System Setup and Evaluation MEtrics

We ran extensive experiments in which we trained and evaluated RAG models with various vector embeddings and their ensembles. To ensure robustness and scalability, experiments are run on a high-performance computing cluster. The system architecture presented in previous sections discusses the experimental setup in depth.

As illustrated in system architecture, we selected the Chroma vector database to store unstructured data. The specialty of the selected vector database is that it can be self-hosted (in-premise) and embedded (serverless). In order to save unstructured data into a vector database, embeddings are required. The embeddings are formed using chunks created. These chunks of text data are fed to the embedding model to get vector embeddings (numerical representation). The various embedding models namely BAAI/bge-base-en, thenlper/gte-base, intfloat/e5-base-v2 and their ensemble are used for the experimentation. Using these embedding models, embedding vectors are created and stored in the Chroma vector database. The dataset used for experimentation is the NCERT books dataset. The GPT 3.5 LLM model is used for experimentation. The details of the hardware and software used for the entire experimentation are presented below.

A. Hardware

For the experimentation, we used a laptop with specifications 11th Gen Intel(R) Core(TM) with an i7-1165G7 @ 2.80GHz processor. It has 16 GB RAM, which enables it to handle large-scale computational tasks.

B. Software

The software details used for experimentation are as: Programming Language- Python 3.11, Database- Chroma 0.4.22 [30].

The various metrics used to evaluate selected embedding techniques are presented in the subsequent section.

C. Evaluation Metrics

The performance of the RAG models employing different vector embeddings is evaluated using a set of evaluation measures. As performance evaluation metrics, we employ retrieval speed, throughput, precision at k, etc., for semantic search and Precision, Recall, and F1-score for overall RAG model configuration. The detailed description of various metrics used to evaluate performance follows:

- Upload Time- It is the time required to upload the documents present in the entire dataset.
- Index (Build) Time- It is the time required to index the documents in the entire dataset.
- Retrieval speed- Retrieval speed refers to how quickly a system finds and returns information when prompted. It determines how long a system can search its stored data and deliver pertinent results. It is desirable to have less retrieval speed.
- Throughput- It is defined as the quantum of queries that a database can handle per given amount of time. It is used to check how much amount of data, a system can handle effectively. It is desirable to have large throughput.
- Precision- Precision is measure of true positive observations out of all observations detected as positive. It is desired to have high value of precision.
- Recall- Recall, also known as true positive rate or sensitivity, examines how accurately positive observations are recorded among all the observations for a hypothesis.
- F1 score- It is the harmonic mean of precision and recall. It should be of higher value.
- Semantic Answer Similarity [37] – In a given context the similarity of answers with respect to overall contextual meaning is measured using semantic answer similarity. In NLP it plays a crucial role to measure the performance of the system.
- Rouge score- It measures the matching of n-grams between machine generated language and human provided reference. It computes precision, recall and f1 score. By default, it computes precision, recall and f1 score for n=1 (unigram), n=2 (Bigram) and n=l (longest common subsequence). It is better to have high rouge score. This is useful for performance evaluation of various NLP applications like, summarization, question answering, machine translation.

VI. Results And Discussions

This section presents results obtained after systemic experimentation on selected dataset. Initially we presented metrics for creating database using different embedding techniques, followed by metrics for retrieval from the dataset and generating response from the extracted context. Further we presented insights found from experimental results.

Also we tried to rank the context retrieved by embedding models using LLMs. We specifically used ChatGPT and Llama2 open source model to verify the performance. To rank the models based on context we used following prompt. *{"role": "system", "content": """Suppose you are the best content evaluator and advisor. Perform a holistic evaluation and rank*

the following contexts: Context A {context1} Context B {context2} Context C {context3} Context D {context4} based on their effectiveness in providing the most accurate and comprehensive answer to the question: Question {question}. Consider factors such as clarity, relevance, depth of information, and overall accuracy. Provide a numerical ranking for each context, with 1 being the highest and 4 being the lowest, along with brief justifications for your rankings. # """}. The same prompt is used for generating rankings from Llama2 model.

The document upload time and dividing into chunks is 3.9413 seconds. From Fig. 2, it is observed that weighted average ensemble takes more time to index the chunks followed by average ensemble, while BAAI/bge-base-en performs best over all other models.

The queries are selected from NCERT solutions for each chapter. Around 100 queries were selected for experimentation. When we manually analyzed it is found that for 60% of queries performance is similar as that of the sample query results presented. The sample query is: Who were Marianne and Germania? What was the importance of the way in which they were portrayed? The performance evaluation for this query is presented below.

A. Retrieval-Based Metrics

The retrieval based metrics, retrieval speed and throughput, are presented in Fig.3, and Fig. 4 respectively. For selected embedding models and distance metrics results are presented. We queried Chroma vector database to get retrieval speed and throughput.

Fig. 3 reveals noteworthy performance variations among the models in assessing retrieval speed. BAAI/bge-base-en exhibits a retrieval at 0.14 seconds, closely followed by intfloat/e5-base-v2 and thenlper/gte-base with 0.11 and 0.10 seconds, respectively. However, the average ensemble and weighted average ensemble model outshines them all, achieving an exceptional retrieval speed of merely 0.0075 seconds. Also average ensemble takes 0.0078 seconds to retrieve semantic documents. This impressive speed of retrieval makes the Ensemble a better choice for efficient and rapid information retrieval tasks.

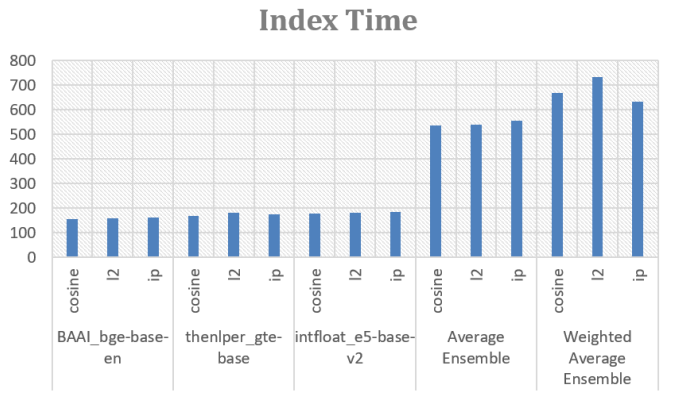

Fig. 2. Index time

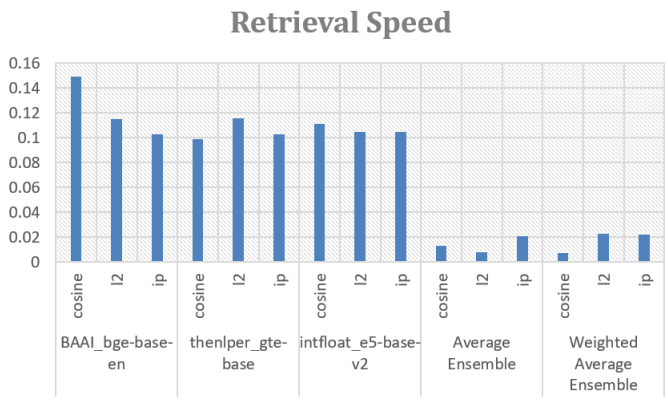

Fig. 3. Retrieval speed

Analyzing the throughput metric from Fig. 4, it is clear that average ensemble and weighted average ensemble outperforms overall compared to state of the art models by handling a remarkable 127 queries per second. Among individual models, thenlper/gte-base takes the lead with 10 queries per second, followed by intfloat/e5-base-v2 at 9, queries per second. This indicates that Ensemble is exceptionally efficient in rapidly processing a high volume of queries, making it an optimal choice for tasks requiring quick response times over all the distance metrics. In average ensemble l2 performs better as compared to other distance metrics, whereas in weighted average ensemble cosine outshines.

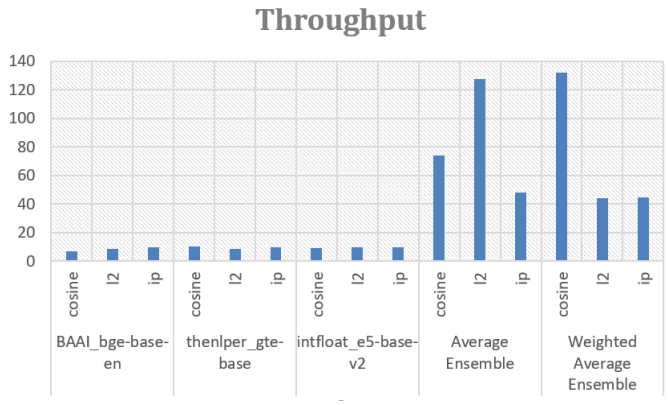

Fig. 4. Throughput

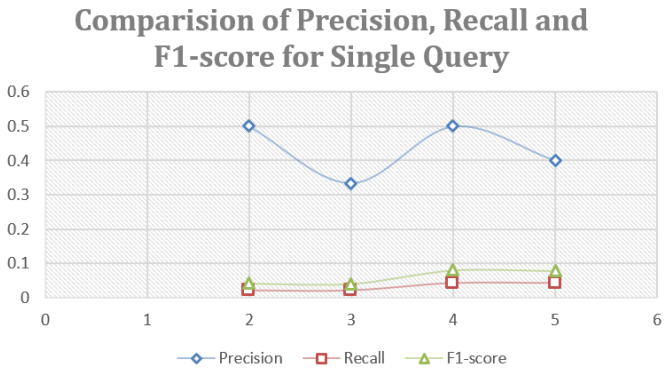

Fig. 5. Comparision of Recall, Precision and F1-score@k for single query

Fig.5 shows the performance of single query for different metrics like precision, recall and F1-score at various values of k for single query. From Fig. 5 it can be seen that at k=4 we are getting optimum results for all metrics.

Fig. 6 reveals various embedding models with selected distance metrics. From Fig. 6, it can be seen that promising result for all ensemble models compared to state of the art models for all metrics namely precision, recall and f1-score@k. Overall, from precision point of view, cosine and l2 provides optimum results over all the embedding models.

B. Generation-Based Metrics

For response generation we specifically used ChatGPT model. We used following prompt to generate response. *"role": "system", "content": {"""Use the following pieces of information to answer the user's question. If you don't know the answer, just say that you don't know, don't try to make up an answer. Only return the helpful answer below and nothing else. Helpful answer: # """}, {"role": "user", "content": context[0]}, {"role": "user", "content": query}.* The response generated by ChatGPT and standard answer from question answer dataset, we computed semantic answer similarity (SAS) score.

The generation based metrics, semantic answer similarity and rouge score, are presented in Table I. The response is generated using ChatGPT LLM model. Using machine generated text and human answers semantic similarity and rouge score are computed.

Considering the SAS (Semantic Answer Similarity) metric, the results from Table I showcase that thenlper/gte-base achieves the highest score of 0.96, closely followed by BAAI/bge-base-en at 0.95. The model intfloat/e5-base-v2 also demonstrates a strong performance with a score of 0.93. Both ensemble techniques, while slightly lower at 0.81 and 0.82, maintains a commendable level of semantic similarity. This implies that thenlper/gte-base and BAAI/bge-base-en are particularly effective in generating responses closely aligned with the semantics of the ground truth.

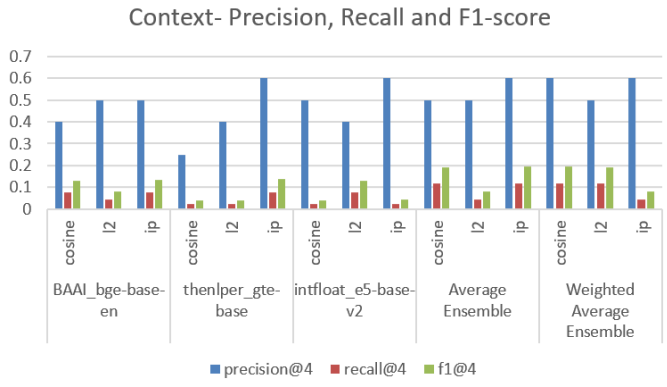

Fig. 6. Comparision of Average Recall, Precision and F1-score@k

TABLE I. SEMANTIC ANSWER SIMILARITY (SAS), ROUGE SCORE

Model / Metric	BAAI/bge-base-en	thenlper/gte-base	intfloat/e5-base-v2	Average Ensemble	Weighted Average Ensemble
SAS	0.95	0.96	0.93	0.81	0.82
rouge-1 (unigram)	r: 0.32, p: 0.61, f: 0.42	r: 0.32, p: 0.53, f: 0.40	r: 0.16, p: 0.31, f: 0.21	r: 0.25, p: 0.48, f: 0.33	r: 0.27, p: 0.40, f: 0.33
rouge-2 (bigram)	r: 0.12, p: 0.29, f: 0.17	r: 0.14, p: 0.31, f: 0.19	r: 0.05, p: 0.13, f: 0.08	r: 0.08, p: 0.18, f: 0.11	r: 0.06, p: 0.12, f: 0.08
rouge-l (Longest Common Subsequence)	r: 0.29, p: 0.55, f: 0.38	r: 0.28, p: 0.46, f: 0.35	r: 0.14, p: 0.27, f: 0.18	r: 0.23, p: 0.44, f: 0.30	r: 0.22, p: 0.31, f: 0.26

The Rouge scores assess the quality of generated answers across different models. BAAI/bge-base-en shows strong unigram precision and recall, while thenlper/gte-base excels in unigram precision. intfloat/e5-base-v2 exhibits lower performance in both unigram and bigram metrics. The average ensemble model achieves a balanced performance, demonstrating notable precision and recall in unigram evaluation. However, it still faces challenges in bigram evaluation. Weighted averaging ensemble model achieves better score as compared to average embedding model. Overall, these scores highlight individual models varying strengths and weaknesses and the collaborative power of the ensemble approach in capturing linguistic nuances.

When we asked ChatGPT to rank Context generated from all models using the prompt explained earlier, it responded as ChatGPT Ranking: Rankings: 1. Context A, 2. Context D, 3. Context B, 4. Context E and Context C. Context A is Context generated from BAAI/bge-base-en, Context B thenlper/gte-base, Context C intfloat/e5-base-v2, Context D average ensemble and Context E weighted average ensemble. From this, thenlper/gte-base outperforms the selected query. Also in order to check performance of ChatGPT, Llama2 model was also used for ranking. It gave Context A, B, C, D and E, which indicates BAAI/bge-base-en performing best. When we manually investigated ranking, we found that for 80% queries ChatGPT and for 70% queries Llama2 has given correct rankings.

We measured the performance of the selected embedding techniques using the semantic answer similarity search metric. This semantic similarity gives optimum performance as compared to all other metrics, viz. precision, recall, and F1 score, for all the experimented queries. We also experimented with an average ensemble and weighted average ensemble of embedding techniques. From the obtained results, it has been observed that the ensemble fails to give optimum performance for all the generation-related metrics, but it outperforms for retrieval-based metrics. It may be the bitter choice to select an ensemble for the selected application to generate the response for this application. It may outperform for other applications.

VII. BUSINESS UTILITIES, CHALLENGES AND WAY FORWARD

Data querying from a single or repository of documents is emerging as a vital business need for various business domains. As documents change in form and shape, document querying becomes an uphill task. Moreover, information locked in documents is also required to be reconciled with critical information present in systems. Such business initiatives are highly human and capital intensive in nature. Data querying finds its business applications across domains such as financial

markets, digital transformation, pharmacology, human resources, legal and may more. Data querying is also emerging as a moat for compliance related operations spanning numerous business domains. In financial markets, data querying is required on documents pertaining limited partnership agreements, master service and trade agreements, product catalogue of financial products and more. In the area of pharmacology, data querying from large sets of documents related to drug discovery and genome mapping is becoming a critical use case of fetching information. Human resource personnel are using data querying to train and transfer domain skills to users in an optimized manner. Data querying will enable users to have key information on their tips which is locked in document archives and will empower users to take better business decisions in current dynamic business scenarios.

A. *Applications of Data Querying*

- Financial Markets: In the financial sector, data querying plays a key role in extracting important information from legal and financial documents such as limited partnership agreements, financial product documentation and trade agreements. It enables analysis of product catalogs for financial products, legacy data pertaining to financial decisions and enables financial institutions to make informed investment decisions. Convoluted clauses pertaining to legal aspects of financial documentation can be easily inferenced.

- Pharmacology: Drug discovery is a time consuming and experiment oriented process leading to creation of tons of data sets and documentation. As AI plays a key role in drug discovery, document querying has been pivotal in enabling researchers to extract valuable insights from vast document repositories related to drug discovery and genome mapping. As a result, research processes are streamlined and pace of innovation in the pharmaceutical industry is accelerated with lesser cost.

- Human Resources: As businesses become demanding and dynamic, humans on the job need to be continuously trained and enabled to upskill their domain expertise in a timely manner. Human resource professionals utilize data querying to enhance the training and skill transfer processes within organizations. Important information from training materials and documentation can be efficiently extracted resulting in quick onboarding and upskilling of employees, enhancing overall organizational productivity.

B. *Challenges and Considerations*

- Document Variability: Unstructured, multi-lingual and variability in document templates and forms pose a challenge to data querying techniques. Training models and LLMs for handling such challenges is an ongoing subject of research.

- Information Reconciliation: The semantics and nuances of representation of information in one language differs in a way it is interpreted and written. This leads to presence of varied language styles and representation of similar information in various forms. Training LLMs to understand contextual meaning of similar information written in varied styles is an interesting research problem worked upon by various researchers across the world.

C. *Future Directions*

- Advanced Querying Techniques: Future research of advanced querying techniques include suggestive data queries and fetching information from structured and unstructured documents. Advanced machine learning techniques and neural networks look promising in this area.

- Integration with AI Systems: Integration of output from document querying systems with other systems for business decision making and recommendations is an interesting future direction of the research. Handling of contextual relevance, ranking and accuracy of understanding dense text is of prime importance.

VIII. CONCLUSION

This research work has revealed the insights using various embedding techniques and their ensemble models. A variety of applications can be developed and enhanced using appropriate selection of embedding models. The selected application realizes RAG based techniques applied in data querying which has vast business applicability across domains. We evaluated performance of selected embedding models with ensemble techniques using various metrics. From the results, we found the following in alignment with our preset hypothesis: a) is accepted as both ensemble models perform better than state of the art embedding models for retrieval tasks b) there is scope of improvement in performance optimization of ensemble models for generation tasks given LLMs are not fine-tuned for response generation. BAAI/bge-base-en embedding techniques give the optimum results over all other selected techniques in generation-based metrics. Also, it was found from the results that semantic similarity and rouge score are the best-suited metrics to measure the performance of embedding techniques as compared to precision, recall, and F1 scores. The findings highlight the importance of embedding selection in maximizing the RAG framework's effectiveness for diverse NLP applications.

REFERENCES

[1] Andrew Trask, Phil Michalak, and John Liu., “sense2vec - A Fast and Accurate Method for Word Sense Disambiguation In Neural Word Embeddings”, ArXiv, abs/1511.06388, 2015.

[2] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov, “Enriching Word Vectors with Subword Information”, arXiv preprint arXiv:1607.04606, 2016.

[3] Rodrigo Alarcón, Lourdes Moreno, and Paloma Martínez, “Exploration of Spanish Word Embeddings for Lexical Simplification”, In Proceedings of CTTS, 2021.

[4] Gustavo Paetzold and Lucia Specia, “Lexical simplification with neural ranking”, In Proceedings of EACL, 2017.

[5] Sandaru Seneviratne, Elena Daskalaki, and Hanna Suominen, “Shared Task: Investigating the Applicability of Lexical Substitution Methods for Lexical Simplification”, In Proceedings of CILS at TSAR-2022, 2022.

[6] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le, “XLNet: Generalized Autoregressive Pretraining for Language Understanding”, In Proceedings of NeurIPS, 2019.

[7] Nikolay Arefyev, Boris Sheludko, Alexander Podolskiy, and Alexander Panchenko, “Always Keep your Target in Mind: Studying Semantics and

Improving Performance of Neural Lexical Substitution", In Proceedings of COLING, 2020.

[8] Oren Melamud, Jacob Goldberger, and Ido Dagan, "context2vec: Learning Generic Context Embedding with Bidirectional LSTM", In Proceedings of SIGNLL, 2016.

[9] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer, "Deep Contextualized Word Representations", In Proceedings of NAACL, 2018.

[10] Hou, Y., "Enhanced word representations for bridging anaphora resolution", arXiv preprint arXiv:1803.04790, 2018.

[11] Bruni, E., Tran, N.K., Baroni, M., "Multimodal distributional semantics", Journal of Artificial Intelligence Research 49, 1–47, 2014.

[12] Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., Ruppin, E., "Placing search in context: the concept revisited", ACM Trans. Inf. Syst. 20, 116–131, 2001.

[13] Luong, T., Socher, R., Manning, C.D., "Better word representations with recursive neural networks for morphology", In: CoNLL, 2013.

[14] Do, H., Than, K., Larmande, P., "Evaluating named-entity recognition approaches in plant molecular biology", bioRxiv p. 360966, 2018.

[15] Gouws, S., Bengio, Y., Corrado, G., "Bilbowa: Fast bilingual distributed representations without word alignments", In: International Conference on Machine Learning. pp. 748–756, 2015.

[16] He, L., Lee, K., Levy, O., Zettlemoyer, L., "Jointly predicting predicates and arguments in neural semantic role labeling", arXiv preprint arXiv:1805.04787, 2018.

[17] Luong, T., Socher, R., Manning, C., "Better word representations with recursive neural networks for morphology", In: Proceedings of the Seventeenth Conference on Computational Natural Language Learning. pp. 104–113, 2013.

[18] Musto, C., Semeraro, G., de Gemmis, M., Lops, P., "Learning word embeddings from wikipedia for content-based recommender systems", In: European Conference on Information Retrieval. pp. 729–734. Springer, 2016.

[19] Ozsoy, M.G., "From word embeddings to item recommendation", arXiv preprint arXiv:1601.01356, 2016.

[20] Park, D., Kim, S., Lee, J., Choo, J., Diakopoulos, N., Elmqvist, N., "Conceptvector: text visual analytics via interactive lexicon building using word embedding", IEEE Transactions on Visualization & Computer Graphics (1), 361–370, 2018.

[21] Santos, C.D., Zadrozny, B., "Learning character-level representations for part-of-speech tagging", In: Proceedings of the 31st International Conference on Machine Learning (ICML-14). pp. 1818–1826, 2014.

[22] dos Santos, C., Gatti, M., "Deep convolutional neural networks for sentiment analysis of short texts", In: Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers. pp. 69–78, 2014.

[23] Yang, X., Macdonald, C., Ounis, I., "Using word embeddings in twitter electi ion classification", Information Retrieval Journal 21(2-3), 183–207, 2018.

[24] recipe/embeddings · Hugging Face.

[25] NCERT Class 10 History Books PDF Download - NCERT Books.

[26] Class 10 History NCERT Solutions PDF (ncrtsolutions.in).

[27] http://trychroma.com.

[28] Kusner, M., Sun, Y., Kolkin, N. & Weinberger, K., "From Word Embeddings to Document Distances", Proceedings of the 32nd International Conference on Machine Learning, in Proceedings of Machine Learning Research 37:957-966, 2015.

[29] Q. Chen, Y. Peng and Z. Lu, "BioSentVec: creating sentence embeddings for biomedical texts", IEEE International Conference on Healthcare Informatics (ICHI), Xi'an, China, 2019, pp. 1-5.

[30] https://pypi.org/project/chromadb/.

[31] BAAI/bge-reranker-base · Hugging Face.

[32] thenlper/gte-base · Hugging Face.

[33] intfloat/e5-base-v2 · Hugging Face.

[34] Yu, H., et al.: Automated assertion generation via information retrieval and its integration with deep learning. In: IEEE/ACM International Conference on Software Engineering, pp. 163–174 (2022).

[35] Xuanting Chen, Junjie Ye, Can Zu et. al., "How Robust is GPT-3.5 to Predecessors? A Comprehensive Study on Language Understanding Tasks", arXiv:2303.00293, 2023.

[36] Hugo Touvron, Louis Martin, Kevin Stone et.al., "Llama 2: Open Foundation and Fine-Tuned Chat Models", arXiv:2307.09288, 2023.

[37] Julian Risch, Timo Möller, Julian Gutsch, Malte Pietsch, "Semantic Answer Similarity for Evaluating Question Answering Models", arXiv:2108.06130, 2021.

[38] Wu, H., and Levinson, D., "The ensemble approach to forecasting: A review and synthesis", Transportation Research Part C, 132, 2021.

[39] Mohsen Shahhosseini, Guiping Hu, Hieu Pham, "Optimizing Ensemble Weights and Hyperparameters of Machine Learning Models for Regression Problems", arXiv:1908.05287, 2020.

[40] Anand, V., Gupta, S., Gupta, D., Gulzar, Y., Xin, Q.; Juneja, S.; Shah, A.; Shaikh, A, "Weighted Average Ensemble Deep Learning Model for Stratification of Brain Tumor in MRI Images", Diagnostics 13(7): 2023.

[41] Mohadese Danesh, Behrouz Minaei, and Omid Kashefi, "A Distributed N-Gram Indexing system to Optimizing Persian Information Retrieval", International Journal of Computer Theory and Engineering, vol. 5, no. 2, pp. 214-222, 2013.

[42] Patta Yovithaya and Sukree Sinthupinyo, "Using Graph Evolutionary to Retrieve More Related Tweets," International Journal of Computer Theory and Engineering, vol. 15, no.2, pp. 62-67, 2023.

[43] Nhon V. Do, TruongAn PhamNguyen, Hung K. Chau, and ThanhThuong T. Huynh, "Improved Semantic Representation and Search Techniques in a Document Retrieval System Design," Vol. 6, No. 3, pp. 146-150, August, 2015. doi: 10.12720/jait.6.3.146-150.