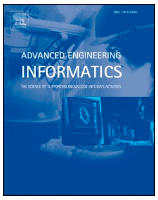
Advanced Engineering Informatics 64 (2025) 103007

Contents lists available at ScienceDirect

Advanced Engineering Informatics

journal homepage: www.elsevier.com/locate/aei



An advanced retrieval-augmented generation system for manufacturing quality control

José Antonio Heredia Álvaro *, Javier González Barreda

Industry 4.0 Chair, Universitat Jaume I, Castellón, Spain

ARTICLE INFO

Keywords:
Knowledge-based systems
Quality control
Ceramic tile
Retrieval-augmented generation
Large language models

ABSTRACT

The rise of Large Language Models (LLMs) with generative artificial intelligence has revolutionized the development of knowledge-based systems, enabling intuitive interactions through natural language. This paper explores the implementation of an advanced Retrieval-Augmented Generation (RAG) system, designed to improve manufacturing quality control by utilizing the capabilities of LLMs, particularly OpenAI's GPT models. We focus on the ceramic tile manufacturing process, where the system retrieves and analyzes specialized bibliographic sources to diagnose defects and propose solutions. In addition to core RAG functionalities, the system incorporates tailored pre-processing and post-processing mechanisms to optimize document retrieval and response generation. The system's effectiveness in solving quality issues is demonstrated through its application in identifying defect causes and generating actionable solutions, significantly improving non-conformities management. This approach not only streamlines troubleshooting but also enhances the quality control system, providing a comprehensive, scalable tool for manufacturers.

1. Introduction

1.1. Context and motivation

Quality control refers to the procedures and processes that ensure products meet specific quality standards, often relying on both human expertise and automated systems. In manufacturing, quality control focuses on identifying defects early and implementing corrective measures to reduce the risk of recurrence.

The digitization of production systems, along with the progressive integration of generative artificial intelligence in industry, has the potential to accelerate continuous improvement cycles and enhance productivity by leveraging documented information. Recent research emphasizes the rising trend of digitalizing quality control processes in SMEs as a means to boost efficiency and reliability [1]. In addition to boosting efficiency, the digitization of quality control processes fosters greater interdepartmental collaboration and sustainability, aligning with modern manufacturing goals of reducing waste and optimizing resource usage.

1.2. State of the art and limitations

Knowledge-based systems are computational systems designed to use structured data and rules to emulate human decision-making, particularly in complex domains such as engineering and quality management. In Knowledge-based systems, domain knowledge is represented explicitly, it is separated from the reasoning algorithm used for processing or inference engine [2]. These systems have been extensively applied in industries where expert knowledge is necessary to automate decision processes and improve operational efficiency. Zhang and Lobov (2024) [3] discuss how Semantic Web Rule Language can be employed to enhance Knowledge-based systems, illustrating their importance in domains that require rule-based automation such as engineering design.

Traditional quality assurance and control practices often rely heavily on human expertise, overlooking the valuable internal knowledge accumulated in company records [4]. To address this limitation, modern quality management information systems have introduced search functionalities that leverage historical data from digitally stored nonconformance reports. By entering keywords, users can retrieve related information when a quality issue arises. However, the effectiveness of these search engines remains constrained, as they are often unable to accurately and directly provide the most relevant insights for

* Corresponding author.
E-mail address: heredia@uji.es (J.A. Heredia Álvaro).

https://doi.org/10.1016/j.aei.2024.103007
Received 8 May 2024; Received in revised form 24 November 2024; Accepted 1 December 2024
Available online 4 December 2024

resolving specific quality problems [5].

The study of how to apply artificial intelligence to the resolution of quality problems, particularly the identification of causes and the proposal of solutions, is a well-established area of research [6] but now takes a new approach with the incorporation of automated text processing. To better process the data in text form contained in quality logs, various automated learning methods have been proposed. For instance, Xu et al. [5] proposed a system that includes tools to build an ontology of quality problems and various text mining and classification algorithms that allow the generation of a digital fishbone diagram. The same authors in another paper [4] developed a text vectorization method for a quality system based on word embedding and proposed using the well-known k-means clustering algorithm to recommend possible solutions for each quality problem. This system provides solutions with effectiveness metrics rather than just similarity measures, improving upon traditional recommender systems. However, with this methodology, expert knowledge of the context is lost when processing quality log reports. Ma et al. [7] proposed a framework with three phases. In the first phase, a library of characteristics related to quality problems is generated partly manually by analyzing the existing documentation in the systems and interviewing the company's technical staff. In the second phase, a multiclass classification (using K-nearest neighbor and a multilayer perceptron network) is performed to relate the characteristics in the library to the root causes. Finally, the authors suggest using a multicriteria group decision method involving several departments of the company to generate possible solutions and select the most appropriate solutions according to various criteria, considering expert and context knowledge. In our opinion, this type of approach suffers from being not very generalizable, requires much manual intervention, as key processing functions are not automated, and therefore requires many resources and long lead times for its implementation in real cases.

The current paradigm of quality management information systems, which represents the state of the art, limits the knowledge base to samples collected from an organization's internal records, excluding the broader body of knowledge found in technical publications relevant to manufacturing process quality issues. In line with this paradigm, Zhou et al. [8] propose an inspection assistant that employs a large language model for root cause analysis of quality defects in aerospace part manufacturing, drawing on defect survey sheets, quality inspections, and maintenance reports from the past three years within a specific company. A key limitation of this approach, and others like it, is the inability to generate reasoned explanations that inspire sufficient confidence for personnel to adopt the recommended solutions.

Despite advancements in integrating artificial intelligence into quality management, existing systems remain limited in their ability to leverage external, domain-specific knowledge effectively. This paper addresses this gap by introducing a Retrieval-Augmented Generation (RAG) system that bridges the divide between generalized language models and the specific needs of manufacturing quality control.

1.3. Objective and contributions of this work

In this paper we explore how the capabilities of LLMs, particularly OpenAI's GPT (Generative Pretrained Transformer), can be harnessed to analyze specialized bibliographic sources and extract information on the causes and potential solutions to manufacturing defects. LLMs are advanced models trained on extensive datasets to perform a wide range of natural language processing tasks and generate human-like text.

Transformer-based LLMs have demonstrated their effectiveness in understanding natural language and generating coherent, contextually accurate text. These models have already been applied successfully in various industries beyond manufacturing, such as in VR rehabilitation, where Zhang, Li, and Chang [9] demonstrated the adaptability of LLMs to provide tailored support and enhance automation in entirely different contexts. This shows the versatility of LLMs and their potential for broad applications across industries. The transformer architecture [10], which underpins models like GPT, employs attention mechanisms that allow it to better interpret dependencies between distant texts than earlier neural network models, such as long short-term memory (LSTM) networks [11]. This enables it to capture syntactic patterns and understand the semantics of the content [12]. Additionally, GPT's scalability allows for training on increasingly large datasets.

Significant potential benefits have been identified from the use of LLMs, such as OpenAI's new GPT models for quality management. By using the text-based communication capabilities provided by an LLM, companies can ensure effective information sharing among employees [13,14], and employees can access quick and accurate information on the steps needed to improve product or service quality, helping to make processes more efficient [15]. Another potential application is the use of chat applications in these models to support the training of new employees and ensure continuous staff training. Thus, for example, ChatGPT can become a practical tool for providing training material, answering questions, and accessing learning resources for management. This helps to increase competence in quality management processes, and in this way, companies can take more effective and focused improvement measures.

LLMs, such as OpenAI's new GPT models, are trained with general information for the purpose of providing generic answers to virtually any problem that is posed to them. However, they are not capable of giving precise answers or with a level of knowledge of a well-trained and up-to-date subject matter specialist on very specific or highly topical domains of knowledge. To achieve this level of precision in the answers, it is necessary to complement the LLMs' capacity with specialized knowledge bases so that they truly represent a useful and effective application.

One way to achieve this level of performance in domain-specific applications is to specialize the language model in the corresponding field of knowledge by developing a system that interacts with the LLM in such a way that, when faced with a user request, it performs a previous search in a specialized knowledge base that allows the LLM to retrieve the information of the specific field needed to accurately generate the response required by the user, as suggested by Kamnis [16]. This approach is known as Retrieval-Augmented Generation [17], whose research and development are rapidly evolving and increasing its ability to improve its usefulness in professional practice.

1.4. Retrieval-augmented generation

We can define a Retrieval-Augmented Generation (RAG) as a hybrid AI technique that combines the retrieval of domain-specific information with generative language models. This approach allows for more accurate and context-aware responses by integrating knowledge from a specialized database. The seminal work by Lewis et al. (2020) [17] introduced RAG as a key technique for handling knowledge-intensive tasks in natural language processing.

In an RAG system, the domain knowledge is stored in vector databases or vector stores, which are designed to store individual text units called splits or chunks in which the dataset or corpus to be used as an external source of knowledge to specialize the model is divided. These splits are stored and indexed by their corresponding embeddings belonging to the same vector space [18] (Fig. 1).

Since embeddings are representations that capture the semantic meaning of text in dense multidimensional vectors [19,20], the representation of splits in a vector space allows fast and efficient retrieval of relevant information when faced with a query encoded in the same latent space (bi-encoder). This retrieval is based on comparing all embeddings corresponding to the splits of the vector store with the embedding of the query using distance metrics, which reflects the semantic similarity between the query and the external information [21,22]. For example, in word embeddings, words similar in meaning will have similar vectors, i.e., vectors close in latent space. Thus, during a query, the splits that are vectorially closest to the query are retrieved

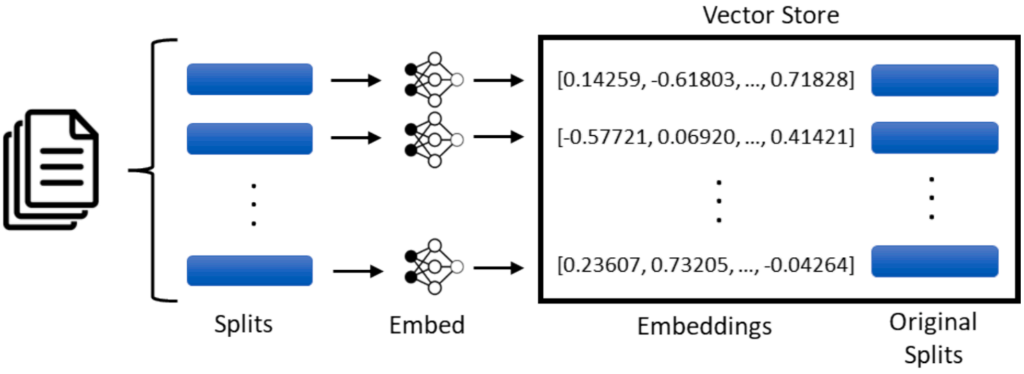


Fig. 1. Indexing process.

for use in the answer synthesis module (Fig. 2).

Response synthesis or response generation is the process in which the model generates a response based on the input it receives (prompt). Response synthesis in a RAG system involves the direct incorporation of all the splits retrieved as domain-specific context along with the query into the user's input to generate the final response (Fig. 2).

LangChain [23] has been used to implement the entire infrastructure of the RAG system. This framework provides tools for creating indexed databases such as Chroma [24] that facilitate the storage of the model input context for easy access, fast insertion, handling of input message limitations when the context is too large, and management in corpus partitioning, among other utilities [23].

In addition, when adapting an LLM with an explicit external knowledge source containing domain-specific information, it is crucial to define a set of rules for the LLM to prioritize the external knowledge context, as it may contain details relevant to the task at hand that could contradict the intrinsic knowledge or behavior of the model. Specialization of the model along with these rules ensures that the model's predictions are anchored in context, allowing refinement or correction of predictions about specific knowledge fields that would be made by the model in isolation, avoiding hallucinations [25] and increasing performance and accuracy.

Therefore, domain specialization can be understood as the process of adapting general LLMs, such as GPTs, to the specific knowledge pertaining to the field in which a given task is to be performed.

Approaches to LLM domain specialization can be categorized primarily into two classes: external knowledge augmentation and model fine-tuning. The external augmentation approach involves augmenting LLM knowledge by incorporating external information directly into the model input, which generally indicates that one does not have access to the internal structure of the model (e.g., GPT-3.5 and later); the model fine-tuning approach implies that one has full access to the LLM (e.g., LLaMA and its variants [26]), including parameter settings, training data, and model architecture [27,28].

Therefore, external augmentation does not necessarily require access to the LLM's internal parameter space, making it more accessible to users with limited resources (e.g., computational resources or domain-specific data). By using external resources or tools, domain-specific knowledge is incorporated into the input prompt, the generated output, or both, effectively tailoring LLM performance without modifying its internal structure, i.e., providing a targeted injection of domain-specific information while employing *prompt* engineering to shape the model inference process.

Model fine-tuning requires more access and resources, as it involves updating LLM parameters to directly incorporate domain-specific knowledge into the model, which, in turn, leads to more profound changes in model behavior.

This paper focuses on the application of the external augmentation strategy using an advanced RAG system to obtain the specialization of an expert language model in instruction processing [29], specifically, the gpt-3.5-turbo-instruct model of the OpenAI GPT-3.5 series.

Employing the gpt-3.5-turbo-instruct model as the basis for the RAG system offers several significant advantages over other models because it is specially trained to follow instructions or meet specific targets detailed in the prompt. This means that the gpt-3.5-turbo-instruct is best for tasks that require a detailed understanding of instructions and the generation of responses that closely align with the provided guidelines.

Testing and evaluation of the results is a fundamental part of the development of a text assistant based on LLMs with external augmentation of the knowledge domain to ensure that the assistant works effectively and reliably. The main purpose of this stage is to verify that the software meets the established objectives by ensuring the functionality, performance and robustness of the tool. In addition, the aim is to identify possible failures, errors or limitations to make the necessary

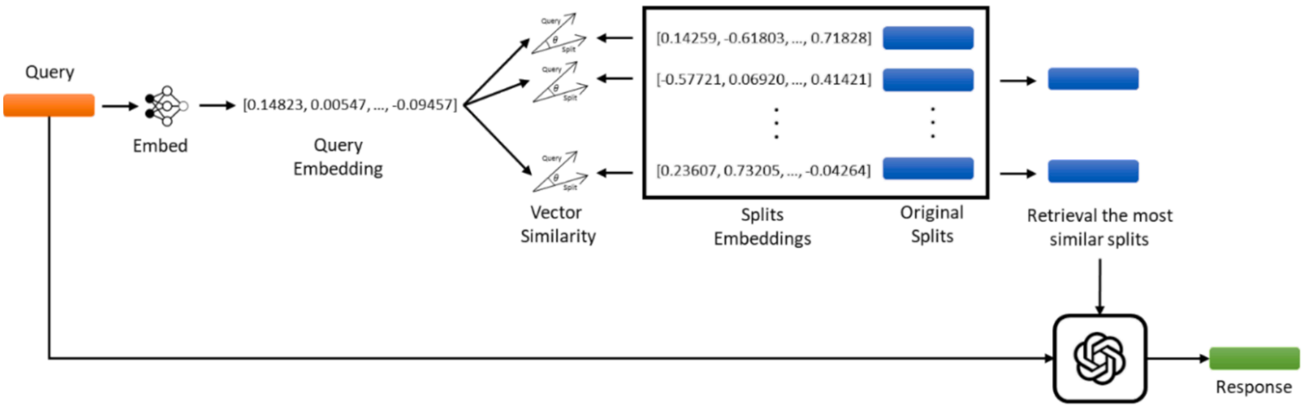


Fig. 2. Retrieval by indexing and response synthesis.

corrections and guarantee the quality of the final answers and to justify the improvement that the use of this type of specialized language model entails, as opposed to the isolated use of an LLM. For this purpose, two different evaluation methodologies are used: one in the retrieval phase and one in the generation phase.

The RAG system developed in this study represents a significant advancement over existing methods for quality control in manufacturing. Unlike traditional information retrieval systems that only fetch relevant documents, our RAG system integrates the retrieval of domain-specific knowledge with generative text capabilities. This enables the system to provide not only accurate information but also well-reasoned explanations tailored to the user's query. Additionally, the system automates critical pre-processing and post-processing tasks, reducing manual intervention and significantly shortening the time required to resolve quality issues. This combination of retrieval and generation provides a more comprehensive solution to address the complex challenges in quality control, particularly in industries like ceramic tile manufacturing, where defect identification and solution generation are highly context-dependent.

1.5. Structure of the paper

Then, in the remainder of the paper, we describe the application scenario selected to validate the approach: ceramic tile manufacturing. Next, we explain the approach of the RAG system we designed and the logic of the system describing its workflow. Then, we evaluate the responses of our system, and finally, we state the conclusions of this work. The results obtained show how the use of LLMs is enhanced with a small selection of academic texts on ceramic manufacturing defects and their causes, as domain knowledge augmentation works exceptionally well, far outperforming more advanced general LLMs.

2. Application scenario

The application scenario used to validate the developed RAG approach is within the manufacturing process of ceramic tiles. The primary stages in this process include spray-drying, milling, pressing, drying, enameling, printing, firing, finishing, and classification. Fig. 3 provides an illustration of these stages, highlighting some of the critical control variables essential for quality control.

Excessive variation in any of the variables involved in these factors and their interactions can generate defects and thus products non-conformities [30]. The purpose of a quality management system is to control this complex set of variables to minimize the defect rate and detect problems in the process as early as possible to limit the loss of productivity. In addition, the intrinsic variability of the raw materials and of the ceramic manufacturing process makes it necessary to inspect the final product to classify the products into different quality categories and to avoid any defective products reaching the customer.

The manufacturing of ceramic tiles involves the control of numerous factors at each stage of the process, including the composition of the raw materials, environmental conditions such as humidity and temperature, rheological properties of the glazes and colors, and conditions and control parameters of the production equipment such a presses and kilns.

The specialized knowledge base on defects in ceramic manufacturing consists of data compiled from a comprehensive handbook that catalogs main ceramic manufacturing defects [31] and a curated selection of academic articles that focus on specific defects and analyze their causes [32–42]. The documents were selected based on their relevance to common defects in ceramic tile manufacturing including research on the defect causes. We prioritized high-quality sources, including industry-recognized handbooks and peer-reviewed academic articles, to ensure the reliability and applicability of the information. The selected documents cover a wide range of defects, from material composition issues to process control factors, providing a comprehensive knowledge base that

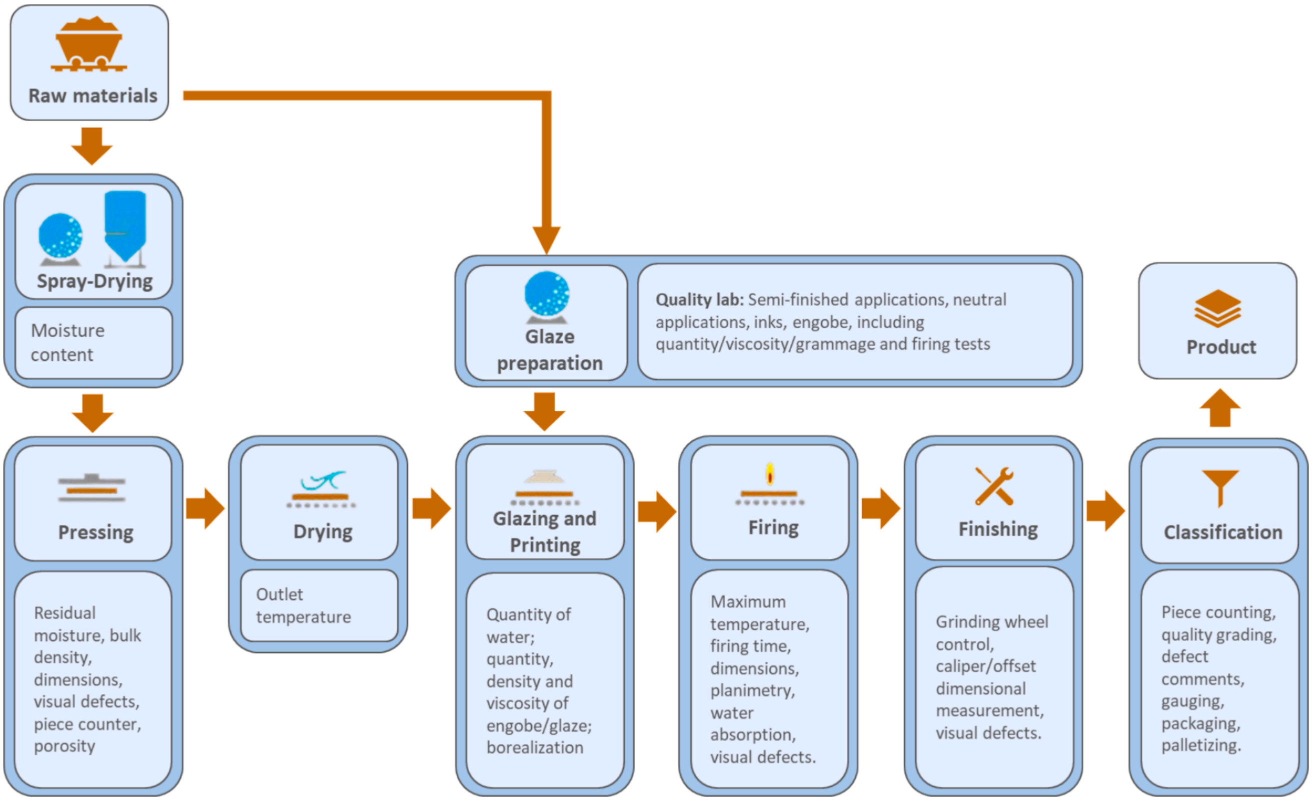

Raw materials
Spray-Drying
Moisture content
Glaze preparation
Quality lab: Semi-finished applications, neutral applications, inks, engobe, including quantity/viscosity/grammage and firing tests
Product
Pressing
Residual moisture, bulk density, dimensions, visual defects, piece counter, porosity
Drying
Outlet temperature
Glazing and Printing
Quantity of water; quantity, density and viscosity of engobe/glaze; borealization
Firing
Maximum temperature, firing time, dimensions, planimetry, water absorption, visual defects.
Finishing
Grinding wheel control, caliper/offset dimensional measurement, visual defects.
Classification
Piece counting, quality grading, defect comments, gauging, packaging, palletizing.

Fig. 3. Ceramic tile process phases and main control factors.

supports the RAG system's ability to diagnose defects and recommend corrective actions. This careful curation of documents ensures that the extracted information is both relevant and trustworthy for addressing real-world quality issues in the manufacturing process.

Table 1 shows the defects and process variables studied as potential causes in the reference articles included in the knowledge base in addition to the more than one hundred defect types included in the handbook [31].

We can envision several use cases where a virtual assistant powered by the RAG system could enhance quality management in a manufacturing plant. The following are illustrative examples:

Use case 1.- Customer claim resolution

Upon receipt of a complaint from a customer regarding the appearance of defects in tiles installed on the façade of a building (i.e. black spots and dark brown clusters with yellow halos), the virtual assistant helps identify the possible cause (i.e. glaze grain size) and therefore focusing the investigation process and faster resolution of the claim.

Use Case 2.- Non-conformities reporting.

The computer vision system detects variations in the tone of the finished product during a production batch. With the virtual assistant's support, the quality manager identifies the root cause: a resolution mismatch in the inkjet printer. After resolving the issue, the virtual assistant assists in drafting the non-conformities report in compliance with ISO 9001 standards.

Use case 3.- Continuous improvement action

On a porcelain stoneware production line, the virtual assistant identifies that tile deformation during firing, caused by pyroplastic deformation, resulted from inadequate control of the bulk density of pressed bodies. The issue is addressed by implementing quality control checks before and after pressing to ensure this characteristic remains within acceptable limits.

3. Methods

In accordance with the taxonomy of Gao et al. [43], RAG systems can be classified according to the technologies and methods used to implement the semantic representation, align user queries with the documents used as knowledge sources, and combine retriever outputs with the text generated by the LLM.

The solution adopted in this work can be framed as illustrated in the flowchart (Fig. 4) as an advanced RAG [43], which is structured around pre-processing, indexing and retrieval, and post-retrieval and generation components to meet the objective of improving non-conformities management in the ceramic tile manufacturing process.

Details of the implementation of each of the components are described below.

3.1. Pre-processing

The first step consists of transforming both the queries made by the user and the documents used as the source of the domain augmentation. The purpose of this first step is to standardize the text, improve consistency and enrich the context by removing irrelevant information and ambiguities to provide more focused and accurate retrieval information.

3.1.1. Query pre-processing

Query processing involves the application of a process known as query rewriting [43], which results in the generation of standardized queries that consistently refer to the specific domain in question.

In our case, we have implemented two options for the user. One option is through a user interface that enables the selection of a specific defect from a list; the tool automatically formulates the necessary query to collect relevant information about the chosen defect. The second is through an open query. In this case, the query text is processed and subjected to a validation process executed by the language model to ensure that the query is related to the field of quality management in the

Table 1
Defects and causes in the reference papers.

Article	Defect studied	Process analyzed	Process variables studied
[32]	blob, crack, edge, eclipse glaze, pinhole, scratch	firing, pressing, firing	humidity levels, temperature changes, pressure applied during forming, glaze application method, glaze amount, kiln temperature
[33]	color, white dots, stains	inkjet printing, glass application and firing	temperature of green body, amount of glaze, printer resolution, glaze type
[34]	negative curvature, denaturation of exterior surface, wrong flatness, cracks	firing, glazing, pressing, drying	temperature change, human error, machines state
[35]	burr, crack, drip, deformation, pinhole, pitting, glaze blistering, color tone, black spot, plucking, tear, surface defect	firing, press, drying, glazing	firing temperature, press surface moisture, post-drying moisture, firing time, firing temperature, glaze weight, engobe weight, glaze weight, drying temperature, glaze density, engobe density, engobe density
[36]	flaking of ceramic tile glazes	glazing lines, kiln	thermal expansion of body, glazes, and engobes, engobe and glaze layer thickness, initial densification temperature of engobe, porosity of engobe, kiln temperature variation
[37]	glaze accumulation at tile edges, surface flatness and curling, differences in layer thickness, tile color variation, drying rate	glazing and firing	quantity of glaze, surface roughness, body type, tile temperature during glazing, glaze-body thickness ratio, temperature of the item to be glazed, glaze conditions (type, rheology, and additives), temperature of the tile to be glazed, amount of applied glaze, roughness of the surface to be glazed, layer thickness, additives in engobe and transparent glaze
[38]	Overfiring	Firing	heating rate, holding time, green bulk density, temperature gradient, thermal conductivity, characteristics of the green compact, microstructural development
[39]	spots, pits, scratches, cracks, color differences, defects, impurities, bubbles, uneven distribution	mixing, firing, handling, cooling, raw materials, pressing	material composition, sintering temperature, sintering duration, handling procedures, cooling rate, material homogeneity, additives, mixing duration, forming pressure, raw material quality, forming technique, material viscosity
[40]	surface defect detection	mechanical finishing	tool forces, angles and cutting conditions, tool wear condition
[41]	black spots and dark brown clusters with yellow halos	Firing	composition and size of glaze material grains

(continued on next page)

Table 1 (*continued*)

Article	Defect studied	Process analyzed	Process variables studied
[42]	pyroplastic deformation	Firing	additive proportions, raw material particle size, max. temperature, unfired body bulk density

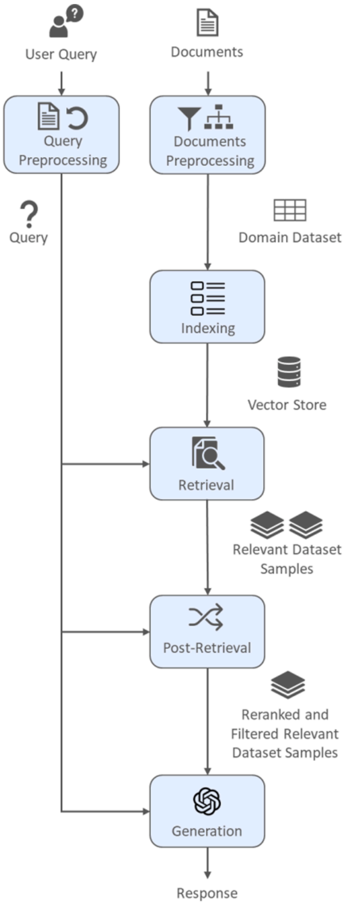


Fig. 4. Flowchart of the developed RAG system.

ceramic industry. This validation mechanism allows us to maintain control over the queries processed by the tool, thus preventing the entry of incorrect data that could cause inadequate performance of the tool.

3.1.2. Document pre-processing

A prior task to achieve an increase in domain knowledge is the collection of documents representing the state of the art in a single repository and making them available for processing.

By pre-processing the documents, all the information available in the repository is filtered and structured to ensure good performance in the retrieval phase. This involves cleaning, extracting, organizing and formatting the raw information from the various sources into suitable knowledge samples that make up the dataset to ensure that the input data are in a format suitable for adapting the model.

The pre-processing of these documents was performed as described in more detail in the flowchart in Fig. 5. The documents are classified into short documents and long documents according to a predefined token threshold.

If a document contains a number of tokens that does not exceed this threshold, it is entered as context in the LLM input (gpt-3.5-turbo-16 k).

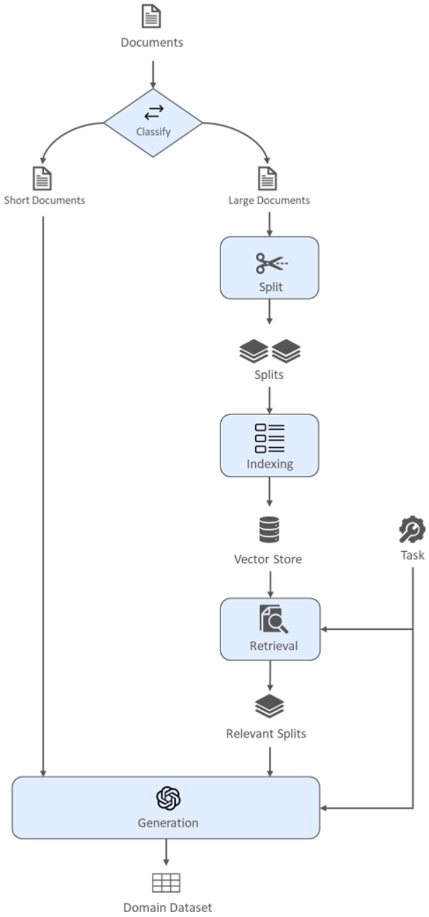


Fig. 5. Document pre-processing flowchart.

For this purpose, a basic LangChain chain (LLMChain) is used, which is responsible for including the document content as prompt context in a template that tells the language model what information to extract from the document and the format in which to present it. Then, the chain inserts the template into the input of the model that generates the response.

Otherwise, a simple RAG system (naive RAG [43]) is implemented. In this system, the document is first fragmented into splits (usually paragraphs). These splits are transformed into OpenAI embeddings and stored in a Chroma vector store. Subsequently, a Question-Answering Retrieval Chain from LangChain (RetrievalQA) is used as a retriever, which employs the gpt-3.5-turbo-instruct as the LLM, the vector store and an input template. This retriever receives the information being searched for in the documents (task), retrieves the most relevant splits for those indications, and adds them to the input template together with the task and the instructions for the content and form of the response to be generated by the language model. Finally, the chain sends the template as input to the LLM, which is in charge of generating the final response.

The domain-specific dataset obtained after performing all the document pre-processing steps consists of our example of 221 samples of ceramic defect information structured as follows:

- <defect type > is the type or shape of the observed defect, e.g., ’Bubbles’, ’Pinholes’, ’Cracks’…
- <identification > is the method for identifying a defect or confirming its identification.
- <causes > are the possible causes that have produced the defect and/or the explanation of the defect.
- <solutions > are the possible solutions for the defect.

- <defect > refers to the class to which the defect belongs, e.g., 'Pinholes' is a defect of (or due to) 'Insufficient Grinding' (often coinciding with the causes).
- <origin > are the possible areas where the defect originated.

The resulting dataset was manually reviewed and corrected to ensure the accuracy of the information, eliminating erroneous samples.

3.2. Indexing and retrieval

The main objective of the indexing and retrieval process is to identify the appropriate context for answering a certain query.

For this purpose, a bi-encoder is used, as mentioned in the introduction, with which the information samples that make up the dataset are encoded on the one hand and the query on the other. OpenAI embeddings are used to index all external domain knowledge previously preprocessed in a Chroma vector store. The Euclidean distance is used as the semantic similarity metric between the query embeddings and the external information samples to select the relevant information in relation to the user's query.

The embedding model used is OpenAI's text-embedding-3-large model, which is characterized by the dynamic generation of embeddings, i.e., this model adapts to the context in which the words are used, unlike static models, which use a single vector for each word regardless of the context [44]. In other words, in dynamic embedding, the same word can have different embeddings depending on the other surrounding words, allowing contextual understanding to be captured. This process is carried out in a single step (once retrieval [43]), which means that a certain number of samples of relevant information for a given query are selected in a single retrieval before moving on to the generation of the final answer. Some studies suggest that by employing more sophisticated augmentation processes, better retrieval results can be obtained. These include iterative retrieval [45,46], recursive retrieval [47], adaptive retrieval [48] and learning to rank retrieval [49]. According to Gao et al. [43], these methods arise as a solution to the drawbacks that have been detected in some once-retrieval applications, such as how to produce redundant or irrelevant content or retrieve information in an inappropriate order, which may dilute or contradict essential information, thus degrading the quality of generation [50,51]. Additionally, this single retrieval may be insufficient for complex problems that require multistep reasoning, as it provides a limited scope of information [43]. However, the effort made in this work in the preprocessing phase, in which the augmentation data are optimized, and in the post-retrieval phase, in which the retrieved information is optimized, circumvents these drawbacks and makes it unnecessary to implement methods to refine the retrieval process.

To summarize, at this stage, the system divides the domain dataset obtained in the previous stage into its component samples. Then, it extracts the embeddings of the samples and stores them in the vector store indexed by their corresponding embeddings. Subsequently, it performs the semantic retrieval process by comparing the Euclidean distance between each of the embeddings of the different samples of the dataset with the embedding of the user's query, and selecting the samples closest to the query. That is, the system retrieves samples that are relevant to answer the given query.

3.3. Post-retrieval

The performance of linguistic models usually decreases when excessive context is introduced, so we included a reranking process after retrieval to solve this problem. The central idea is to reorder the information records to prioritize the most relevant records, thus reducing the total number of context documents. This not only solves the problem of widening the context window during retrieval but also improves efficiency and responsiveness. The reordering process provides more efficient and accurate input for further processing of the linguistic model [52].

To perform this post-retrieval process, our system includes a reranker cross-encoder, specifically the sentence transformers cross-encoder ms-frame-MiniLM-L-6-v2, which improves the selection of data based on their relevance and semantic distance, complementing the retrieval via Euclidean similarity.

A cross-encoder is a specific type of transformer-based natural language processing model used to understand the relationship between two texts. Unlike classical models that embed each text independently (also called bi-encoders), cross-encoders take pairs of texts as a single input. In this way, the representations of the two texts interact with each other from the first layer of the model, which allows the model to capture more complex and deeper contextual relationships between texts as they capture these relationships from the beginning and throughout all layers of the model, unlike traditional models that first generate independent embeddings of each text and then compare them with some semantic similarity metric. Consequently, cross-encoders tend to be more accurate in understanding the relationship between two texts than are bi-encoders.

However, the disadvantage of cross-encoders is that they are computationally more intensive than methods using distance metrics, especially when comparing one text with a large set of candidate texts, since each pair of texts needs to be processed by the model separately, whereas with traditional methods, the texts are only processed once by the embedding model and then compared using distance-based metrics.

Given the design requirements of this case, where a query text needs to be compared with many candidate texts, we have chosen to employ a bi-encoder model—which is less accurate but much more efficient—in the retrieval stage (seen in the previous point) followed by a reranking process of the preselected information with a cross-encoder model. In this way, the cross-encoder only processes a small number of text pairs, scoring each information sample according to its semantic relationship with the user's query and retaining only the most relevant ones.

In this way, starting from the relevant samples of the dataset extracted in the previous stage, the system passes each of these samples together with the query through the cross-encoder, obtaining a ranking of the samples ordered by their relevance as a context to answer a given query. Then, these samples are filtered keeping only the most relevant ones for the generation phase, using a relevance threshold as we will see in the next section.

Accordingly, we can obtain the most relevant contextual information samples to answer a given query from our dataset. In other words, in this phase we extract the context that the language model will later need in the next stage to answer our query accurately.

3.3.1. Retrieval evaluation

During the development of the system, an evaluation of the retrieval should also be carried out to verify and improve its efficiency. For this purpose, an evaluation process has been meticulously designed to not only measure the goodness of information retrieval by the model but also to optimize the two critical hyperparameters that regulate this system: the number k of samples retrieved as relevant and the threshold for the filtering of these retrieved samples.

To carry out this task, first, a validation dataset was generated consisting of pairs of input queries and labels, the latter representing the set of identifiers of the relevant information samples for each query, ordered from most to least relevant. The selection of these samples was based on two essential criteria: the content coverage and the accuracy and precision of the information provided.

To determine the final configuration of the hyperparameters, an extensive testing process was followed to explore all possible combinations of both hyperparameters, in which the range of k was set between 1 and 20, while the threshold varied between different filtering criteria: from taking samples with a score higher than the average of all scores through samples with a score higher than the average of the positive scores to including only those samples with positive scores, the first n samples with the highest score (n being an integer from 1 to k), and even all samples without applying a threshold.

In addition, to evaluate the retrieval performance, i.e., the accuracy of the samples retrieved by the model for each hyperparameter configuration, three different evaluation metrics were employed: the Jaccard similarity, the F1-score, and the Kendall-Tau distance, which compare the retrieval output (model predictions) with the labels of the *dataset*. Jaccard similarity measures the intersection over the union of the retrieved samples and the labels, providing a clear view of the accuracy in terms of matched elements (Appendix A). The F1 score combines precision and sensitivity to provide a balanced measure of model accuracy, penalizing both omissions and erroneous inclusions (Appendix B). The Kendall-Tau distance, on the other hand, evaluates the concordance in the ordering of the samples and is particularly useful for measuring the quality of the reranking performed by the system (Appendix C).

Thus, after processing the entire validation *dataset* for each hyperparameter combination, the mean of each evaluation metric is calculated for all queries, thus identifying the hyperparameter configurations that maximize the mean Jaccard similarity, the F1-score, and the Kendall–Tau distance. Therefore, at the end of this process, three outstanding hyperparameter configurations are obtained, each corresponding to the best means of the three metrics.

Finally, the final hyperparameter configuration is selected after a manual comparison of these three optimal configurations, considering the specific strengths and weaknesses of each metric in relation to the objective of retrieving and generating relevant information.

This approach allows us not only to optimize the performance of the RAG system for the identification and analysis of ceramic defects but also to provide a detailed and robust evaluation of this performance, ensuring that the system is not only able to retrieve relevant information efficiently but also to generate accurate and relevant answers, generating greater confidence for the system user.

3.4. Generation

The information retrieved from the vector store must be merged with the user query and form an input to the LLM (gpt-3.5-turbo-instruct) while addressing the drawback posed by the language model context window limit. The simultaneous input of all relevant retrieved data to the LLM without applying a post-retrieval process may exceed the context window limit, introduce noise, and make it difficult to focus on crucial information.

This is why the reranking carried out in the previous stage together with the filtering of the information performed in the pre-processing phase are techniques that guarantee that the limit of the context window is never exceeded.

Thus, making use of a basic LangChain chain, the final response synthesis is performed by directly incorporating the retrieved, reordered and filtered samples and the user's query into the LLM prompt using a standardized template that organizes all the input information and describes the instructions to be followed by the LLM, directing it to generate the response correctly.

In other words, in this stage we obtain the final answer to a given user query constructed by the language model from the query itself and the context extracted in the previous stage.

3.4.1. Generation evaluation

Finally, during the tool design process, an evaluation of the generation process was carried out with several objectives:

- The software must meet the established requirements to ensure adequate functionality, performance and robustness.
- Possible failures, errors or limitations were identified to make the necessary corrections and ensure the quality of the final answers.
- To quantify the improvement obtained by using this type of language model with domain-specific specialization, as opposed to using it in isolation.

To carry out this evaluation, we selected the ROUGE-L metric (recall-oriented understudy for gisting evaluation—longest common subsequence), a specific validation metric widely used in the field of natural language processing, which is a special case of the ROUGE metric.

ROUGE was devised for the automatic evaluation of the quality of text summaries. That is, it was designed to measure the similarity between reference texts (human summaries) and texts generated by automatic summarization systems. ROUGE is considered a good indicator of similarities in coherence, conciseness, grammaticality, readability and content in text summarization tasks [53]. This metric is based on the comparison of n-grams (sequences of n consecutive words) between human references and those produced automatically by the system under evaluation.

ROUGE-L is a variant of ROUGE that focuses on the length of the longest sequence of matching words between the automatic summary and the human references. That is, unlike ROUGE-N metrics, which are based on the matching of individual n-grams, ROUGE-L evaluates similarity in terms of longer contiguous word sequences. Thus, to counteract the disadvantages of a pure recall metric such as in ROUGE-N, ROUGE-L calculates the weighted harmonic mean (or f-mean) by combining the precision score and the recall score (the calculation methodology to obtain the ROUGE-L score is explained in detail in Appendix D).

The advantage of ROUGE-L is that it does not require consecutive matches but rather sequence matches that reflect the word order at the sentence level as n-grams. In addition, it automatically includes the longest common n-grams in sequence so that no predefined n-gram length is necessary.

Scores range from 0 to 1, where higher scores suggest that there is greater similarity in terms of word sequence and word order between the system-generated response and the human reference, indicating a better quality of the generation in terms of fidelity and coverage of relevant content. Lower scores indicate that there is less agreement between the word sequences of the generation and those of the reference, which could point to problems of relevance, accuracy or even coherence in the generated text.

Since the task performed by the language model in this study is very similar to that of summarization, since it involves rewriting the retrieved information samples according to the indications made by the user in his or her query, ROUGE-L represents a suitable metric for validating the tool generation process.

To verify that a good ROUGE-L score is obtained on the entire dataset, the metric is applied as explained below. First, a query is performed so that the system deliberately returns as a response all the information concerning a defect as close as possible to the samples in the dataset. Next, this response is defined as the hypothesis, and the samples from the *dataset* that the system has used to generate the response are defined as the reference. Finally, this process is repeated for all the defects available in the knowledge base. This allows us to perform an analysis on all the information contained in the dataset in a few minutes and in a fully automatic way. To determine the goodness of the obtained scores, for each defect, in addition to applying ROUGE-L to the hypothesis and its respective reference, it is also applied in parallel to the hypothesis and a random reference. That is, samples from the *dataset* other than those related to the given query are randomly chosen and compared with the hypothesis. In this way, on the one hand, the actual score is obtained, and on the other hand, a random score is obtained. If the real score is significantly higher than the random score, a good score is obtained, and consequently, the tool performs well.

4. Results

4.1. Retrieval evaluation and hyperparameter optimization

This section discusses the results of the system retrieval evaluation carried out as explained in section 3.3.1. As seen, promising results are obtained, highlighting the effectiveness of the model in retrieving relevant information in the face of a possible query. The key results of this process are presented below and are structured in a table summarizing the values obtained for the Jaccard similarity metric, F1-score and Kendall-Tau distance under the optimal configurations for each metric (Table 2).

An analysis of the results revealed that the configuration that maximizes both the Jaccard similarity and F1-Score shares the same values for both metrics, with a Jaccard similarity of 92.68 %, an F1-Score of 85.81 %, and a Kendall-Tau distance of 0.3631. This configuration was achieved with a value of $k = 7$ and using the mean of the relevance scores as the threshold. On the other hand, the configuration optimizing the Kendall-Tau distance achieved with a value of $k = 4$ and using the positive scores as a threshold improved this specific metric (0.2931), although it decreased the other two metrics by approximately 5 % (Jaccard similarity of 87.13 % and F1-Score of 80.22 %).

Given these results, the choice of the optimal hyperparameter configuration for the RAG system is one that simultaneously maximizes the Jaccard similarity and F1 score. This decision is based on the importance of prioritizing both the accuracy in retrieving information samples (indicated by Jaccard similarity) and the balance between accuracy and sensitivity (reflected in the F1-Score). The efficiency of these two metrics suggests that the system can identify with high accuracy the most relevant samples against a query.

Although the improvement in the Kendall-Tau distance indicates a greater correlation in the ordering of the retrieved samples with respect to the relevance labels, the compromise in the other two metrics leads to the decision to prioritize a configuration that guarantees higher accuracy and precision in the identification of relevant information. Furthermore, the hyperparameter configuration that maximizes both the Jaccard similarity and F1-score offers a Kendall-Tau distance very similar to that offered by the hyperparameter configuration that maximizes the Kendall-Tau distance; however, this configuration offers a 5 % lower Jaccard similarity and F1-Score, which can considerably decrease model performance.

To support the results of the information retrieval evaluation of the model, a three-dimensional visual representation of the embedding space of the dataset samples was generated to illustrate how they are organized and related to each other (Fig. 6).

To reduce the dimensionality and visualize the set of high-dimensional embeddings, the uniform manifold approximation and projection (UMAP) algorithm [54] was used. First, this unsupervised learning algorithm has been trained with the embeddings of our vector store so that it can understand the underlying structure of the high-dimensional space of our embeddings and how to project it into a lower-dimensional space. Then, the trained model transforms the high-dimensional embeddings into low-dimensional projections, i.e., returns their representation in the low-dimensional space learned during fitting.

In the graph in Fig. 6, each point represents an embedding of a sample of information or a specific query. The colors are used to differentiate between the different types of embeddings:

- The orange points represent the embeddings of all the information samples present in the dataset.
- The red point indicates the embedding of the specific query made by the user (in the case shown, concerning solutions to bubble defects on the surface).
- The blue points represent the embeddings of the information samples that the model has retrieved as relevant to the query in question.

The main purpose of this graph is to provide an intuitive representation of how the model processes and responds to a specific query. By visualizing the proximity of points in three-dimensional space, one can better understand the model's ability to identify and retrieve relevant information in response to a query. Ideally, the blue points (retrieved samples) should be closer to the red point (the query) than most of the orange points, indicating that the model has correctly identified the most relevant information samples among all those available in the dataset.

In the specific example shown in Fig. 6, the visual output of the embedding map clearly demonstrates the effectiveness of the model in the information retrieval task. The fact that the three points closest to the red point (the query) are colored blue suggests that the model can accurately identify the most relevant information samples for the bubble defect solution query. This implies high relevance and accuracy in the retrieval process since the samples closest in the embedding space are usually the most similar or relevant to the query, according to the features learned by the model during training.

This type of visualization not only helps to validate the effectiveness of the model in terms of retrieving relevant information but also provides valuable insights into the structure of the embedding space and the distribution of the data. It allows researchers to better understand the internal dynamics of the model, identify possible areas for improvement, and helps to adjust the hyperparameters or architecture of the model to optimize its performance on specific retrieval and response generation tasks.

4.2. Generation evaluation

This section analyzes the results of the system generation evaluation. This analysis is based on the procedures previously described in section 3.4.1, which details the method used to evaluate the ability of the specialized model (*gpt-3.5-turbo-instruct*) to generate adequate and accurate responses to domain-specific queries.

The results obtained from this evaluation are presented through a statistical analysis represented in a boxplot, as shown in Fig. 6. This boxplot illustrates the distribution of the scores obtained from the responses generated by the system (called real scores), as opposed to the scores derived from randomly generated responses (called random scores). In this graphical representation, the actual scores are shown in blue, while the random scores are shown in red.

The quantitative analysis (Table 3) revealed a mean of 0.6108 for the real scores, with a standard deviation of 0.1371, which contrasts significantly with the random scores, whose mean is 0.2300, accompanied by a standard deviation of 0.05697. This comparison highlights that the mean of the real scores exceeds the mean of the random scores by more than twice the standard deviation of the former, which indicates a marked superiority in the quality of the responses generated by the model. Moreover, as shown in the Figure above, the minimum non-outlier value of the real scores is very close to the third quartile of the random scores, reaffirming this disparity.

This difference in the distribution of scores between real and random responses, clearly visible in the box plot (Fig. 7), underscores the model's ability to generate responses that are not only consistent and relevant but also closely aligned with the expectations and domain-

Table 2
Results of the retrieval evaluation and hyperparameter optimization.

Optimal metrics	Jaccard similarity	F1-score	Kendall-Tau distance	k	Threshold
Best Jaccard similarity	0.9268	0.8581	0.3631	7	Mean
Best F1-score	0.9268	0.8581	0.3631	7	Mean
Best Kendall-Tau distance	0.8713	0.8022	0.2931	4	Positives

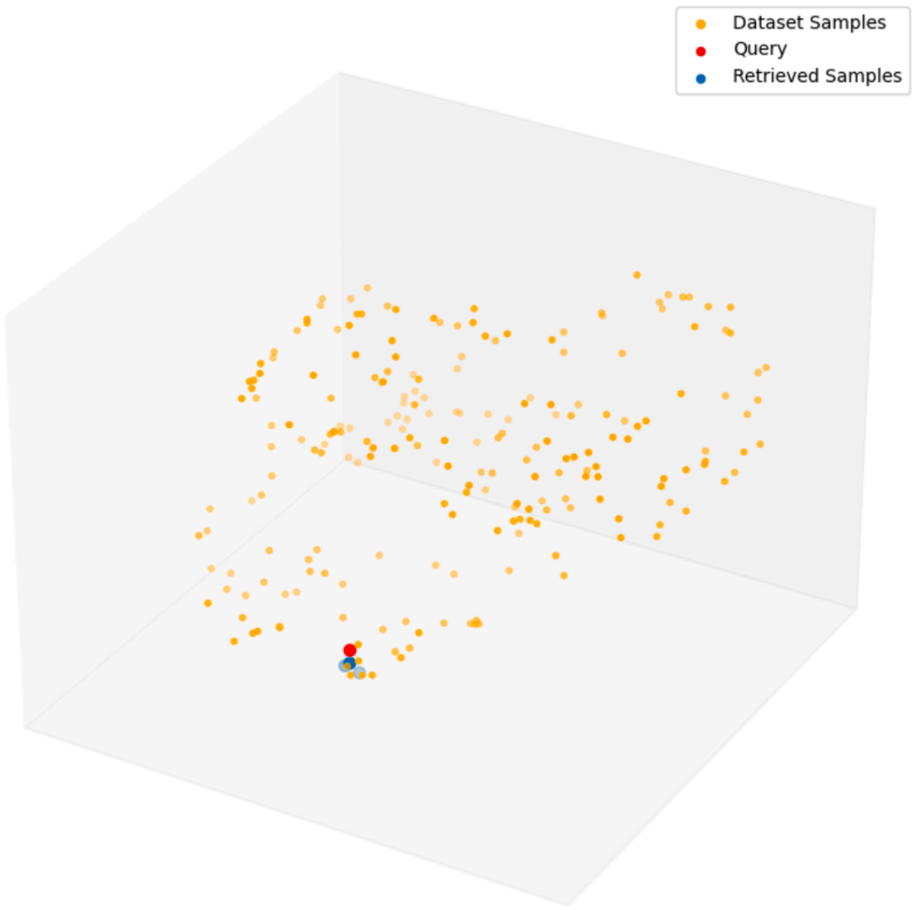


Fig. 6. Representation of the *embeddings* in three-dimensional space.

Table 3
Generation evaluation results.

	Real Scores	Random Scores
Mean	0.6108	0.2300
Standard deviation	0.1371	0.05697

specific information provided.

In addition, a ROUGE-L score of 0.61 indicates that, on average, 61 % of the longest word sequences in the system-generated responses match those in the references, taking into account word order and allowing for nonconsecutive but in-order sequences, reflecting both the quality of content coverage and the fidelity of sentence structure with respect to the references.

Therefore, these results show good performance, suggesting that the answers generated by the system have a reasonable similarity in length and content with respect to the standard references, meeting the accuracy, reliability and robustness criteria of the RAG system in the generation of answers to specific queries in the field of ceramic quality control.

5. Discussion

The infrastructure for domain adaptation of the language model built with LangChain that the developed RAG system employs makes it a highly scalable tool due to the high capacity for growth and flexibility provided by this technology, easily allowing the introduction of new knowledge into the model. Likewise, the OpenAI LLM implementing the assistant not only stands out for having one of the highest performances in the market for generation tasks but also offers the advantage of being able to be executed in any machine without having to consider its computational capacity since the model is executed in its own servers, which makes this system a versatile and adaptable tool for different application environments.

The proposed system has a low economic cost, both in its development phase and during its production phase. The cost of using the model (gpt-3.5-turbo-instruct) was 2 dollars per 1 M tokens, which is approximately 0.0012 dollars per query, or 1 dollar per 830 queries.

It is illustrative to show how the system works when faced with possible questions from a user and to compare the answers with those provided by the GPT-4 model without domain adaptation. Table 4 shows some representative examples that are not intended to constitute a formal evaluation result of our system, but rather to provide a qualitative comparison that illustrates the differences in performance and relevance of responses between a domain-adapted model and a general model such as GPT-4.

In this table, together with the responses, the evaluation of the responses is also provided using the ROUGE-L metric and a verification by a human expert who assigns a “YES” for correct responses, i.e., those that a human expert in this domain would answer or consider valid, and a “NO” for incorrect or incomplete responses, i.e., those that a human expert in this domain would never answer or does not consider valid due to imprecision, lack of coherence, lack of concreteness, lack of content, wrongness of concepts, etc.

The ROUGE-L metric in these examples is applied as described in point 3.4.1, i.e. comparing the response generated by the language model with the information retrieved from the database, since the aim is to evaluate the quality of the response, i.e. to contrast the response with reality. And this is precisely what is achieved by applying ROUGE-L in this way, comparing the similarity of the answer with the information

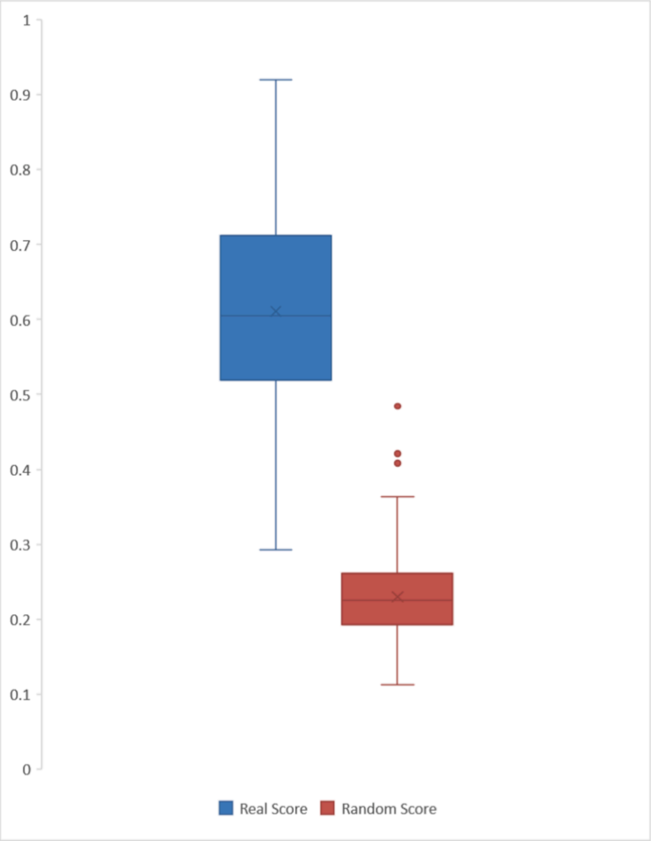


Fig. 7. Boxplots of real scores and random scores.

samples retrieved from the database, which is the real information about the query being performed.

It is important to note that the ROUGE-L metric is a standard and widely accepted tool for assessing the quality of summaries and answers generated by language models, measuring the overlap of n-grams, word strings and substrings between the generated answer and a reference. Although these metrics may not fully capture contextual relevance or semantic accuracy in some cases, they provide an objective and quantifiable basis for comparison.

In this context, the application of ROUGE-L helps to measure how well the general model, such as GPT-4, aligns with the domain-specific database information, allowing to identify areas where the domain-tailored model could offer significant improvements, as discussed below.

The first query relates to the observation of carbon particles in a batch and asks you to identify the type of defect and its solution. The disparity in the responses from the original GPT-4 model and the adjusted, data-indexed model underscores the benefits of fine tuning and domain adaptation. The response from the adjusted model is more accurate and complete, specifically mentioning what defect it is and what techniques and actions to apply to eliminate the occurrence of this defect, such as weathering the clays or screening through finer meshes, found in the indexed content. The ROUGE-L score for our adapted model in this query is 0.59, reflecting a high match with the database information. The human evaluation is also “YES”, indicating that the answer is correct and valid. In contrast, the original GPT-4 model offers a generalist and less accurate answer, incorrectly stating that the cause of the defect is given by dirt on the parts and the solution is based on a cleaning of the parts. Its ROUGE-L score is 0.12, very low in comparison, and the human evaluation is “NO”, which highlights its lack of precision and relevance.

In the second query, the reasoning ability of the model to formulate an answer to a question about a very specific type of defect is evaluated. The GPT-4 model responds in a context different from that of ceramic production, obtaining a ROUGE-L score of 0.089, reflecting a low similarity to the relevant information, and the human evaluation is “NO”. However, the RAG system accurately identifies the relevant source in the dataset and provides the correct answer. The ROUGE-L score of our adapted model is 0.26, although lower than the first query, it is lower due to the specificity of the question and the variability in the possible correct answers, which does not necessarily imply a deficiency in the model, but a diversity in the way the correct answer is expressed. The human evaluation is “YES”, confirming that the answer is correct and accurate.

Finally, for the third query, the detected defect is described in detail and the type of defect and its causes are requested to be identified. Again, the GPT-4 model reaches conclusions that are too generalist and incorrect, with a ROUGE-L score of 0.12 and a human evaluation of “NO”. On the other hand, the fitted model accurately identifies the relevant source text and provides an accurate answer to the given query, with a ROUGE-L score of 0.68, the highest among the queries, reflecting

Table 4
Examples for comparison between the GPT-4 model and our RAG system (gpt-3.5-turbo-instruct with domain specialization).

Query	GPT-4			RAG GPT-3.5		
	Response	ROUGE-L	Human Evaluation	Response	ROUGE-L	Human Evaluation
Carbon particles are observed in a batch of parts. Tell me what the defect is and how I can fix it.	This is a fouling defect. To solve it, improve the cleaning process. Make sure that the parts are properly cleaned before moving them to the next production step. Also, regular checking of the equipment can prevent contaminations.	0.12	NO	This is a punctual defect due to impurities present in the raw materials. It can be solved by weathering the impurity-bearing clays, reducing the particle size or by sieving through finer meshes.	0.59	YES
Explain to me how I can identify devitrification and in which area of the production process it originates.	Devitrification is identified by an opaque or crystalline surface on glass, generally resulting from incorrect temperatures during the cooling process. This defect originates in the annealing stage of the glass production process.	0.089	NO	Devitrification can be identified by electron microscopy with spot microanalysis. This form of defect originates in the enamel development of the production process.	0.26	YES
A small hollowness of a few tenths of a millimeter has been detected on the flat surface of the parts. Tell me what the defect is and what are its causes.	This is a “porosity” defect in the part. Possible causes may be poor casting, defective welding or inadequate cooling of the material during processing. It can also be caused by the presence of impurities in the metal.	0.12	NO	This is the defect known as “Eyes”. The possible causes are: presence in the powder that feeds the presses of lumps of paste in plastic state, formed during the wetting process, and which produce a lower point porosity and, consequently, a lower absorption; dirty punch that causes small hollows on the surface of the piece.	0.68	YES

high similarity between the fitted model answer and the expected answer, demonstrating the model's ability to extract and reformulate domain-specific information. The human evaluation is "YES", demonstrating once again the validity and accuracy of the response provided by the fitted model.

This study represents one of the first implementations of a Retrieval-Augmented Generation (RAG) system specifically tailored to address challenges in manufacturing quality control. By integrating domain-specific knowledge with generative capabilities, the proposed system provides context-aware and actionable solutions, surpassing the limitations of traditional knowledge-based systems that often rely solely on internal records. These contributions not only demonstrate the feasibility of combining advanced AI technologies with domain-specific applications but also set the foundation for further research into adaptive, intelligent quality control systems across various manufacturing domains.

6. Conclusions

This work addressed the development of an advanced RAG system, leveraging LLMs with external domain augmentation, aimed at improving non-conformities management in the ceramic industry.

We conclude that it is necessary to preprocess and postprocess the information retrieved by LLMs to achieve accurate and good performance using increased domain knowledge.

The strategy of using a bi-encoder in the retrieval phase and a cross-encoder in the post-retrieval phase achieved good results both in terms of response quality and computational performance.

We propose including rigorous evaluations of both retrieval and generation during the development process to fine-tune and debug the system. The evaluation of the retrieval allows the adjustment of two hyperparameters: the number of retrieved samples and the filtering threshold. The evaluation of the generation of the final response using the ROUGE-L metric allows us to validate the quality of the system responses.

This study demonstrates that GPT models, when enhanced with a curated selection of academic texts on manufacturing defects and integrated into a well-designed RAG system, can generate coherent and reliable answers. This capability positions such systems for practical use in real-world quality management environments to support continuous improvement.

A key area for future improvement is the integration of internal records. While our current approach focuses on external, specialized knowledge sources, incorporating the ability to process and leverage internal company data, such as historical quality logs and nonconformance reports, would further enrich the system's knowledge base. This integration would allow for more comprehensive and accurate solutions, combining both internal operational insights and broader industry knowledge to provide a more personalized approach to quality control. Furthermore, adopting more sophisticated retrieval strategies, such as iterative or adaptive methods, could enhance the system's ability to address complex queries, thereby improving its accuracy and overall performance.

While this study focused on the specific application of a RAG system in ceramic tile manufacturing, its methodology and results highlight the system's potential to be adapted across other manufacturing domains. By combining domain-specific retrieval with generative capabilities, the proposed system offers a versatile framework for addressing complex quality control challenges. Future work could explore its implementation in other industries, such as automotive, aerospace, or pharmaceuticals, where the integration of external knowledge bases with internal operational data could further enhance its performance.

Author contributions

Both authors contributed to the study conception and design. State of the art review, data collection and analysis were performed by [José Antonio Heredia Álvaro]. Software development was performed by [Javier Gonzalez Barreda]. The first draft of the manuscript was written by both authors commented on previous versions of the manuscript. Both authors read and approved the final manuscript.

CRediT authorship contribution statement

José Antonio Heredia Álvaro: Writing – review & editing, Writing – original draft, Validation, Supervision, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization. **Javier González Barreda:** Writing – review & editing, Writing – original draft, Software, Formal analysis, Data curation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. . Jaccard similarity calculation methodology

A.1. Jaccard similarity definition

The Jaccard similarity is defined as the size of the intersection divided by the size of the union of the sample sets.

A.2. Jaccard similarity calculation

Given two sets A and B, the Jaccard similarity J is calculated as follows:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

Where:

- $|A \cap B|$ is the number of elements common to both sets A and B.
- $|A \cup B|$ is the total number of unique elements in both sets A and B.

Appendix B. . F1-score calculation methodology

B.1. Preliminary definitions

Before delving into the calculation of the F1-score, let us define the basic components derived from the confusion matrix:

- True Positives (TP): Number of positive instances correctly identified.
- False Positives (FP): Number of negative instances incorrectly identified as positive.
- False Negatives (FN): Number of positive instances incorrectly identified as negative.
- True Negatives (TN): Number of negative instances correctly identified.

With these definitions, we proceed to calculate the metrics of precision and recall, fundamental to the F1-score calculation.

B.2. Precision

Precision is defined as the proportion of positive instances correctly identified against the total instances identified as positive (both correct and incorrect):

$$Precision = \frac{TP}{TP + FP}$$

B.3. Recall

Recall measures the proportion of positive instances correctly identified out of the total actual positive instances:

$$Recall = \frac{TP}{TP + FN}$$

B.4. F1-score calculation

Finally, the F1-score is calculated as the harmonic mean of precision and recall, offering a balance between these two metrics:

$$F1 - score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} = 2 \cdot \frac{\left(\frac{TP}{TP+FP}\right) \cdot \left(\frac{TP}{TP+FN}\right)}{\left(\frac{TP}{TP+FP}\right) + \left(\frac{TP}{TP+FN}\right)}$$

This formula ensures that the F1-score is only high if both precision and recall are high, promoting a balance between correctly detecting positive instances (recall) and the accuracy in labeling an instance as positive.

Appendix C. . Kendall-Tau distance calculation methodology

C.1. Kendall-Tau distance definition

The Kendall-Tau distance between two sets is defined as the number of pairwise inversions needed to transform one sequence into the other. An inversion occurs when the order of two elements is different in the two sequences being compared.

C.2. Preliminary definitions

Consider two sequences of ranks A and B, both containing the same set of n integer elements. Our goal is to calculate the Kendall-Tau distance, denoted as $\tau(A, B)$, between these sequences.

For each pair of elements i, j in the sequence, we define two properties:

- Concordant: A pair i, j is concordant between A and B if the order of i and j is the same in both sequences (either $i < j$ or $i > j$).
- Discordant: A pair i, j is discordant between A and B if the order of i and j is different in the two sequences (one has $i < j$ and the other $i > j$).

C.3. Kendall-Tau distance calculation

The Kendall-Tau distance, $\tau(A, B)$, is calculated as follows:

1. Counting Concordant and Discordant Pairs: For each pair of elements i, j with $i < j$, determine if they are concordant or discordant between A and B. This can be done by comparing the relative positions of each pair in both sequences.
2. Calculation of the Distance: The Kendall-Tau distance is the total number of discordant pairs between the two sequences. Formally, if C denotes the number of concordant pairs and D denotes the number of discordant pairs, the Kendall Tau distance is calculated as $\tau(A, B) = D$.

Appendix D. . ROUGE-L calculation methodology

D.1. ROUGE-L definition

ROUGE-L assesses the similarity between generated and reference texts through precision, recall, and F1-score metrics, based on the longest common subsequence (LCS). The LCS is defined as the longest subsequence present in both texts, maintaining element order but not necessarily contiguous.

D.2. Calculation of LCS

For a generated text G of length n and a reference text R of length m, $LCS(G,R)$ denotes the LCS's length. This calculation employs dynamic programming, constructing a matrix L of size $(n+1) \times (m+1)$, with the following computation steps:

1. Initialize $L[i][0] = 0$ and $L[0][j] = 0$ for all i,j.
2. For $i,j > 0$:
   - If $G_i = R_j$, then $L[i][j] = L[i-1][j-1] + 1$.
   - Else, $L[i][j] = max(L[i-1][j], L[i][j-1])$.

The LCS length is found at $L[n][m]$.

D.3. Precision, recall, and F1-score calculation

With $LCS(G,R)$ determined, the following formulas compute the metrics:

- Precision $(P) = \frac{LCS(G,R)}{lengthofG}$
- Recall $(R) = \frac{LCS(G,R)}{lengthofG}$
- F1-score $(F1) = \frac{2 \cdot P \cdot R}{P+R}$

The F1-score harmonically balances precision and recall, offering a singular measure of generated text effectiveness relative to the reference.

Data availability

Data will be made available on request.

References

[1] G. Dutta, R. Kumar, R. Sindhwani, R.K. Singh, Digitalization priorities of quality control processes for SMEs: a conceptual study in perspective of Industry 4.0 adoption, J. Intell. Manuf. 32 (6) (2021) 1679–1698.

[2] N. Sánchez-Pi, J. Carbó, J.M. Molina, A knowledge-based system approach for a context-aware system, Knowl.-Based Syst. 27 (2012) 1–17.

[3] L. Zhang, A. Lobov, Semantic web rule language-based approach for implementing knowledge-based engineering systems, Adv. Eng. Inf. 62 (2024) 102587.

[4] Z. Xu, Y. Dang, Solution knowledge mining and recommendation for quality problem-solving, Comput. Ind. Eng. 159 (2021) 107313.

[5] Z. Xu, Y. Dang, P. Munro, Knowledge driven intelligent quality problem-solving system in the automotive industry, Adv. Eng. Inf. 38 (2018) 441–457.

[6] S. Bird, Object-oriented expert system architectures for manufacturing quality management, J. Manuf. Syst. 11 (1) (1992) 50–60.

[7] Q. Ma, H. Li, A. Thorstenson, A big data driven root cause analysis system: application of machine learning in quality problem solving, Comput. Ind. Eng. 160 (2021) 107580.

[8] B. Zhou, X. Li, T. Liu, K. Xu, W. Liu, J. Bao, CausalKGPT: Industrial structure causal knowledge-enhanced large language model for cause analysis of quality problems in aerospace product manufacturing, Adv. Eng. Inf. 59 (2024) 102333.

[9] Y. Zhang, F. Li, D. Chang, VR rehabilitation system evaluator: a fNIRS-based and LLM-enabled evaluation paradigm for Mild Cognitive Impairment, Adv. Eng. Inf. 62 (2024) 102734.

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Proces. Syst. 30 (2017) 1–11.

[11] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.

[12] Zhao, Wayne Xin, Zhou, Kun, Li, Junyi, Tang, Tianyi, Wang, Xiaolei, Hou, Yupeng, Min, Yingqian, Zhang, Beichen, Zhang, Junjie, Dong, Zican, others, A survey of large language models, arXiv preprint (2023), arXiv:2303.18223,.

[13] A.S. George, AS.H. George, A review of ChatGPT AI's impact on several business sectors, Partners Universal Int. Innov. J. 1 (1) (2023) 9–23.

[14] A.S. George, AS.H. George, AS.G. Martin, ChatGPT and the future of work: a comprehensive analysis of AI's impact on jobs and employment, Partners Universal Int. Innov. J. 1 (3) (2023) 154–186.

[15] D.Y. Pimenov, A. Bustillo, S. Wojciechowski, V.S. Sharma, M.K. Gupta, M. Kuntoglu, Artificial intelligence systems for tool condition monitoring in machining: Analysis and critical review, J. Intell. Manuf. 34 (5) (2023) 2079–2121.

[16] S. Kamnis, Generative pre-trained transformers (GPT) for surface engineering, Surf. Coat. Technol. 466 (2023) 129680.

[17] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al., Retrieval-augmented generation for knowledge-intensive nlp tasks, Adv. Neural Inf. Proces. Syst. 33 (2020) 9459–9474.

[18] O. Ram, Y. Levine, I. Dalmedigos, D. Muhlgay, A. Shashua, K. Leyton-Brown, Y. Shoham, Incontext retrieval-augmented language models, Transactions of the Association for, Comput. Linguist. 11 (2023) 1316–1331.

[19] Chen, Haochen and Perozzi, Bryan and Al-Rfou, Rami and Skiena, Steven, A tutorial on network embeddings, arXiv preprint, (2018), arXiv:1808.02590.

[20] Liu, Qi and Kusner, Matt J and Blunsom, Phil, A survey on contextual embeddings, arXiv preprint,(2020), arXiv:2003.07278.

[21] Li, Huayang and Su, Yixuan and Cai, Deng and Wang, Yan and Liu, Lemao, A survey on retrieval-augmented text generation, arXiv preprint, (2022), arXiv: 2202.01110.

[22] Mao, Yuning and He, Pengcheng and Liu, Xiaodong and Shen, Yelong and Gao, Jianfeng and Han, Jiawei and Chen, Weizhu, Generation augmented retrieval for open-domain question answering, arXiv preprint (2020), arXiv:2009.08553.

[23] O. Topsakal, T.C. Akinci, Creating large language model applications utilizing langchain: A primer on developing llm apps fast, International Conference on Applied Engineering and Natural Sciences 1 (1) (2023) 1050–1056.

[24] T. Taipalus, Vector database management systems: Fundamental concepts, use-cases, and current challenges, Cogn. Syst. Res. 101216 (2024).

[25] Zhang, Yue and Li, Yafu and Cui, Leyang and Cai, Deng and Liu, Lemao and Fu, Tingchen and Huang, Xinting and Zhao, Enbo and Zhang, Yu and Chen, Yulong and others, Siren's song in the AI ocean: a survey on hallucination in large language models, arXiv preprint (2023), arXiv:2309.01219.

[26] Touvron, Hugo and Martin, Louis and Stone, Kevin and Albert, Peter and Almahairi, Amjad and Babaei, Yasmine and Bashlykov, Nikolay and Batra, Soumya and Bhargava, Prajjwal and Bhosale, Shruti and others, Llama 2: Open foundation and fine-tuned chat models, arXiv preprint, (2023), arXiv:2307.09288.

[27] Zhao, Xujiang and Lu, Jiaying and Deng, Chengyuan and Zheng, Can and Wang, Junxiang and Chowdhury, Tanmoy and Yun, Li and Cui, Hejie and Xuchao, Zhang and Zhao, Tianjiao and others, Domain specialization as the key to make large language models disruptive: A comprehensive survey, arXiv preprint, (2023), arXiv:2305.18703.

[28] K. Ezukwoke, A. Hoayek, M. Batton-Hubert, et al., Big GCVAE: decision-making with adaptive transformer model for failure root cause analysis in semiconductor industry, J Intell Manuf (2024) 1–16.

[29] L. Ouyang, J. Wu, Xu. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, Katarina Slama, Alex Ray, et al., Training language models to follow instructions with human feedback, Adv. Neural Inf. Proces. Syst. 35 (2022) 27730–27744.

[30] J.A. Heredia, M. Gras, Análisis y modelado de la transmisión de variabilidad dimensional en un proceso de producción de baldosas cerámicas, Boletín De La Sociedad Española De Cerámica y Vidrio 48 (6) (2009) 289–296.

[31] Amorós Albero, JL and Beltán-Porcar, V and Blasco-Fuentes, A and Enrique-Navarro, JE and Escardino-Benlloch, A and Negre-Medall, F, Defectos de fabricación de pavimentos y revestimientos cerámicos, (1991), Univ. de Valencia: Ed. AICE-ITC.

[32] N.-T. Huynh, A multi-subpopulation genetic algorithm-based CNN approach for ceramic tile defects classification, J. Intell. Manuf. (2023) 1–12.

[33] M. Montorsi, C. Mugoni, A. Passalacqua, A. Annovi, F. Marani, L. Fossa, R. Capitani, T. Manfredini, Improvement of color quality and reduction of defects in the ink jet-printing technology for ceramic tiles production: A Design of Experiments study, Ceram. Int. 42 (1) (2016) 1459–1469.

[34] P.H. Tsarouhas, D. Arampatzaki, Application of Failure Modes and Effects Analysis (FMEA) of a ceramic tiles manufacturing plant, 1st Olympus International Conference on Supply Chains (2010) 1–17.

[35] V. Sevinc, M.M. Kırca, A comparison between the Bayesian network model and the logistic regression model in prevention of the defects on ceramic tiles, J. Exp. Theor. Artif. Intell. (2022) 1–17.

[36] Quinteiro, Eduardo and Caridade, Marcelo D and Menegazzo, Ana Paula M and Paschoal, Andre B and Machado, Edvaldo AG and Cesario, Ronaldo M, Key ceramic processing variables that lead to flaking of ceramic tile glazes, Qualicer (2004), VIII World Congress on Ceramic Tile Quality, 3.

[37] Corma, P and Martinez, J and Vte, J, Solutions to the problem of glaze accumulation at tile edges and associated faults, Qualicer (2000). VI World Congress on Ceramic Tile Quality, 1.

[38] F. Contartesi, F.G. Melchiades, A.O. Boschi, Anticipated Overfiring in Porcelain Tiles: Effects of the firing cycle and green bulk density, Boletín De La Sociedad Española De Cerámica y Vidrio 58 (2) (2019) 69–76.

[39] Z. Zhao, Review of non-destructive testing methods for defect detection of ceramics, Ceram. Int. 47 (4) (2021) 4389–4397.

[40] H. Shang, C. Sun, J. Liu, Xuefeng Chen, Ruqiang Yan, Defect-aware transformer network for intelligent visual surface defect detection, Adv. Eng. Inf. 55 (2023) 101882.

[41] Z. Radojevíc, A. Terzíc, M. Vasíc, M. Arsenovíc, Non-typical defects on surfaces of ceramic and roof tiles: nature and the causes, International Journal of Modern Manufacturing Technologies 7 (1) (2015) 61–66.

[42] E. Sánchez, V. Sanz, E. Cañas, J. Sales, K. Kayacı, M.U. Taskıran, U.E. Anıl, S. Turk, Revisiting pyroplastic deformation. Application for porcelain stoneware tile bodies, J. Eur. Ceram. Soc. 39 (2–3) (2019) 601–609.

[43] Gao, Yunfan and Xiong, Yun and Gao, Xinyu and Jia, Kangxiang and Pan, Jinliu and Bi, Yuxi and Dai, Yi and Sun, Jiawei and Wang, Haofen, Retrieval-augmented generation for large language models: A survey, arXiv preprint, (2023), arXiv: 2312.10997.

[44] Xie, Yu and Li, Chunyi and Yu, Bin and Zhang, Chen and Tang, Zhouhua, A survey on dynamic network embedding, arXiv preprint, (2020), arXiv:2006.08093.

[45] Z. Feng, X. Feng, D. Zhao, M. Yang, B. Qin, Retrieval-Generation Synergy Augmented Large Language Models, in: ICASSP 2024–2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2024, pp. 11661–11665.

[46] Shao, Zhihong and Gong, Yeyun and Shen, Yelong and Huang, Minlie and Duan, Nan and Chen, Weizhu, Enhancing Retrieval-Augmented Large Language Models with Iterative Retrieval-Generation Synergy, arXiv preprint, (2023), arXiv: 2305.15294.

[47] Trivedi, Harsh and Balasubramanian, Niranjan and Khot, Tushar and Sabharwal, Ashish, Interleaving Retrieval with Chain-of-Thought Reasoning for Knowledge-Intensive Multi-Step Questions, arXiv preprint, (2023), arXiv:2212.10509.

[48] C. Ye, Exploring a learning-to-rank approach to enhance the Retrieval Augmented Generation (RAG)-based electronic medical records search engines, Informat. Health 1 (2) (2024) 93–99.

[49] Asai, Akari and Wu, Zeqiu and Wang, Yizhong and Sil, Avirup and Hajishirzi, Hannaneh, Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection, arXiv preprint (2023), arXiv:2310.11511.

[50] N.F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, P. Liang, Lost in the middle: how language models use long contexts, Trans. Assoc. Comput. Linguist. 12 (2024) 157–173.

[51] Yoran, Ori and Wolfson, Tomer and Ram, Ori and Berant, Jonathan, Making Retrieval-Augmented Language Models Robust to Irrelevant Context, arXiv preprint, 2023, arXiv:2310.01558.

[52] Zhuang, Shengyao and Liu, Bing and Koopman, Bevan and Zuccon, Guido, Open-source Large Language Models are Strong Zero shot Query Likelihood Models for Document Ranking, arXiv preprint, (2023), arXiv:2310.13243.

[53] Lin, Chin-Yew, ROUGE : a package for automatic evaluation of summaries, Proceedings of the Workshop on Text Summarization Branches Out, (2004), 74-81.

[54] McInnes, Leland and Healy, John and Melville, James, UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction, arXiv preprint, (2020), arXiv:1802.03426.