http://doi.org/10.35784/iapgos.6694

received: 12.10.2024 | revised: 20.11.2024 | accepted: 15.03.2025 | available online: 31.03.2025

LLM BASED EXPERT AI AGENT FOR MISSION OPERATION MANAGEMENT

Sobhana Mummaneni, Syama Sameera Gudipati, Satwik Panda
Velagapudi Ramakrishna Siddhartha Engineering College, Department of Computer Science and Engineering, Vijayawada, India

Abstract. *Mission operation management is the coordination and control of various activities related to the operation of a satellite. This critical function involves planning, monitoring, controlling, and coordinating all aspects of the mission, ensuring the spacecraft achieves its objectives. Current mission operation management faces limitations in terms of data transmission relying solely on ground control, manual analysis procedures, and not capitalizing on technology for optimizing routine mission operations. This research proposal aims to develop a Large Language Model (LLM) based Expert AI agent for performing mission operation management. The proposed LLM based AI assistant will perform tasks such as data analysis for pattern recognition, operational planning, and document summarization. The system is designed to operate offline, providing flexibility in deployment. Integrating AI with mission operation management can benefit mission controllers and engineers, scientists and researchers, space agencies and organizations. AI offers opportunities to reduce mission costs, improve success rates, and enhance the efficiency of space exploration programs.*

Keywords: artificial intelligence, expert AI agent, Large Language Model (LLM), mission operation management, telemetry data, Retrieval Augmented Generation (RAG)

EKSPERT AGENTA AI Z OPCJĄ LLM DO ZARZĄDZANIA OPERACJAMI MISJI

Streszczenie. *Zarządzanie operacjami misji to koordynacja i kontrola różnych działań związanych z eksploatacją satelity. Ta krytyczna funkcja obejmuje planowanie, monitorowanie, kontrolowanie i koordynację wszystkich aspektów misji, zapewniając osiągnięcie przez statek kosmiczny swoich celów. Obecne zarządzanie operacjami misji napotyka ograniczenia w zakresie transmisji danych, opierając się wyłącznie na kontroli naziemnej i ręcznych procedurach analitycznych, a nie wykorzystując technologii do optymalizacji rutynowych operacji misji. Celem tej propozycji badawczej jest opracowanie agenta Expert AI opartego na modelu dużego języka (LLM) do zarządzania operacjami misji. Proponowany asystent AI oparty na LLM będzie wykonywał zadania takie jak analiza danych w celu rozpoznania wzorców, planowanie operacyjne i podsumowywanie dokumentów. System zaprojektowano do pracy w trybie offline, co zapewnia elastyczność we wdrażaniu. Integracja sztucznej inteligencji z zarządzaniem operacjami misji może przynieść korzyści kontrolerom i inżynierom misji, naukowcom i badaczom, agencjom i organizacjom kosmicznym. Sztuczna inteligencja oferuje możliwości obniżenia kosztów misji, poprawy wskaźników powodzenia i zwiększenia efektywności programów eksploracji kosmosu.*

Słowa kluczowe: sztuczna inteligencja, ekspert AI, model wielkojęzyczny, zarządzanie operacjami misji, dane telemetryczne, generowanie rozszerzone wyszukiwania

Introduction

Satellites have been launched into space for diverse purposes, gathering essential data related to Earth’s climate and other phenomena. The missions conducted by these satellites are crucial for various reasons, and managing them effectively is essential for ground control operations. The data concerning the satellite’s own sub systems is called telemetry data, and this data includes unstable time series with thousands of sensor measurements that indicate the state of every sub part of the spacecraft. These time series are characterized by being multivariate, diverse in nature, and incorporating multiple modes of data. The complex design of spacecraft and the changing space environment, with sudden temperature shifts, direct radiation, and the danger of space debris collisions, make control difficult. Enhancing the reliability of space system components alone is insufficient to prevent potential failures. Moreover, the vast distance between the spacecraft and the ground station frequently makes it impractical to inspect or repair malfunctioning parts [4]. Any complex system like a satellite produces volumes of telemetry data to be analyzed and visualized by systems on the Earth. For a successful mission in space, it is essential to monitor this data during the operations [10]. Mission operation management refers to the comprehensive process of planning, coordinating, and controlling the activities and resources required to achieve the objectives of a space mission. When considering telemetry data, the anomaly detection is one of the various functions in mission operation management. An anomaly is a kind of event which deviates from the normal behavior of system data. A satellite is a complex system with many interdependent components. Compared to other complex systems, hardware monitoring in satellites is challenging, and a simple component or subsystem failure can be catastrophic. Therefore, it is vital to monitor the satellite's behavior all the time to detect and address change in behaviors as early as possible [29].

Data-driven methods are used to handle many parameters to determine whether the current behavior of a satellite system or subsystem is expected. Ideally, these methods do not make any assumptions about parameter behavior during anomalies or normal satellite operations [29]. In the data feature extraction step, there is a requirement for specialized knowledge when considering traditional data-processing techniques, making the process time-consuming and labor-intensive. The deep-learning techniques have led to the development of new models for analysis of time-series data and prediction using them, which reduced the reliance on them for feature extraction. These models use learning algorithms for data features extraction. Recently, few methods and their embeddings have been applied, such as convolutional neural networks (CNN), recurrent neural networks (RNN), long short-term memory (LSTM), autoencoders (AE), generative adversarial networks (GAN), and variational autoencoders (VAE) [1]. In LSTMs, we can observe gates that take care of flow of information, Cell State – the cell state acts as the memory of the network, flowing through the entire sequence and preserving information over time. Forget Gate – the forget gate determines which information in the cell state should be discarded. It processes the current input along with the previous hidden state and applies a sigmoid function, generating a value between 0 and 1 for each cell state element. A value of 0 indicates complete forgetting, while a value of 1 indicates full retention. Input Gate – the input gate identifies which new information should be incorporated into the cell state. It consists of two parts: a sigmoid layer that selects the values to update and a tanh layer that generates new candidate values. Updating Cell State – the cell state is updated by combining the previous cell state, modified by the forget gate, with the new candidate values, adjusted by the input gate. Output Gate – the output gate determines the next hidden state, which also serves as the output. It combines the previous hidden state and the input, passing the result through a sigmoid function. With the rise of AI, we are getting closer to making what once seemed like science fiction a reality. As astronauts embark on longer missions further from Earth, resulting in increased communication delays, AI frameworks trained on Earth could offer required solutions. Such technology could be especially valuable for handling medical emergencies in distant, challenging space environments [28]. The implementation of AI, particularly with deep learning techniques, enhances the efficiency of information extraction in remote sensing applications.

Additionally, the advantages of deep learning extend beyond a single payload's results, potentially increasing the overall flexibility of the entire satellite [8]. While AI is successfully used in space, it remains largely restricted to offline data processing and has not been implemented on the edge within spacecraft. The primary challenge is adapting deep learning networks to older hardware, which does not have the performance capacity for even basic inferencing [8]. There are no large language models which were combined with deep learning models for mission operation management and that is what we tried to achieve through our research. Generally, any machine learning or deep learning model's output is not explainable. Especially for a non-technical person or someone without the context of what a model is doing it is hard and impossible for them to understand the output. By developing an LLM, we tried to bring a solution to this problem so that the output being generated by the deep learning model, and any satellite mission jargon is understandable to anyone as the output of the LLM being a readable text.

1. Literature review

The satellite telemetry data consists of the information collected and transmitted by satellites to ground stations. This data includes various measurements and parameters related to the satellite's health, status, and environment. The research with Artificial Intelligence is very limited in this field. A comparative analysis for AI-enhanced lunar rover missions, such as Chandrayaan, against conventional missions, has shown that AI-driven approaches significantly improved success rates, navigation precision, and data acquisition efficiency [13]. The authors listed out a few conventional algorithms that can be used for various functionalities, SLAM (Simultaneous Localization and Mapping) can be used for real-time mapping, precise navigation in lunar terrains, Neural networks can be used for learning patterns, terrain classification, obstacle avoidance, Reinforcement learning can be used for adaptive navigation, Trial-and-error learning in dynamic environments, Computer Vision techniques can be used for Geological feature identification, image analysis. [26] revolves around the usage of Adaptive Piecewise Constant Approximation (APCA), Dynamic Time Wrapping and Probabilistic methods like Gaussian Mixture Model and Hidden Markov Model of finding patterns in the satellite telemetry datasets. This is done for a user selected interval in a time series. Finally, the ensembling of these models was done, for better accuracy and predicting capacities. A semi-supervised machine learning technique is always helpful when dealing with time variant data. Few pattern recognition techniques in time series data were discussed in [22] and their evaluation is done. A broad discussion of how Long Short-Term Memory (LSTM) can be useful, Orbit Correction Maneuvers with cross-correlation. The performances are evaluated for artificially generated data and real time data. Steepened Waves were found to be more in the artificial one. German Space Operation Center (GSOC) has developed a data analysis framework for monitoring and analyzing satellite telemetry. This data analysis service uses file watcher service, csv file parser service, telemetry database writer service, deep learning module [21].

Major issues in space missions are the failure of the spacecraft or its deteriorating health, both of which can be identified using its telemetry data. There can be correlations between data if some multivariate telemetry time series data is considered. The anomaly identification is done in the telemetry data and they can be categorized as Point anomalies – an individual data instance which is anomalous with respect to the rest of the data, Collective anomalies - consecutive data instances considered as anomalous, Univariate anomalies – an individual data instance which is anomalous in a particular context. [23] uses sparse representations and dictionary learning, which is a signal vector representation of data for categorizing the anomalies. Based on the sparse values predicted, the classification of whether it is a discrete or continuous anomaly is found. Predicting point anomalies is done easier compared to fragmented anomalies, because point anomalies deviate significantly from the rest of the data where-as fragmented anomalies do not deviate much but are still anomalies.

Prediction or analysis of these anomalies becomes hard, and [15] proposes a detection strategy using Least Square Support Vector Machine (LS-SVM) with performance evaluation for fragmented anomalies. When considering supervised techniques in satellite telemetry data for anomaly detection is hard because there is less to no possibility of finding truth sets, so unsupervised algorithms form the mainstream in this segment. [19] shows a study of already existing algorithms like LSTM, and how they are used in the detection of anomalies, along with their evaluation using Dice Coefficient, Confusion matrix at sequence level, how early is an anomaly spotted or how late is it spotted. In anomaly detection one of the other problems faced is when the rate of false positives is higher, [30] proposes few approaches like using casual network and feature attention-based LSTMs. This also included setting of dynamic thresholds which will finally predict the anomalies being present in the data. [20] proposes a deep clustering based local outlier probabilities approach (DCLOP) for anomaly detection and extracting patterns which are realistic from the operational telemetry data. This approach involves utilizing a dynamically weighted loss function along with a modified version of local outlier probabilities, derived from the outcomes of deep clustering.

One of the essential steps in designing an engineering system is the selection of right parts for it. The most important information could be presented in unstructured documents and its retrieval becomes difficult. [18] explains the usage of information extraction system from data sheets of a space part. This is done by generating all relevant concepts for data and a deep analysis of what is exactly useful for this data. [7] examines the role of Information Retrieval (IR) in Retrieval-Augmented Generation (RAG) frameworks, which enhances LLMs by retrieving relevant external information. It highlights that both the relevance and placement of retrieved passages are crucial for LLM performance.

It found that even random or noisy documents can improve accuracy by up to 35% when well-positioned, challenging conventional IR assumptions. [14] develops and evaluates a Retrieval-Augmented Generation (RAG) pipeline to be utilised healthcare, more in preoperative medicine. Using 35 guidelines, the LLM-RAG model, particularly GPT4.0-RAG, outperforms the base GPT4.0 model and matching human-generated responses across 14 clinical scenarios. [24] analyses application of open-source LLMs, specifically Llama-3-8B and Mixtral 8x7B models, in Retrieval Augmented Generation (RAG) tasks using enterprise-specific datasets. The Llama3-8B model outperformed the Mixtral 8x7B due to broader training and effective instruction-tuning. The study found that beyond a certain top-k value, additional retrieved documents do not enhance retrieval effectiveness, and smaller context windows handle reasoning-dense questions more efficiently. [12] surveys Automatic Text Summarization (ATS) methods and text-preprocessing, with a focus on integrating LLMs. It introduces an alignment of ATS and practical applications using "Process-Oriented Schema", reviewing recent LLM-based works that bridge a two-year gap in the literature. The survey covers ATS applications across domains like news, scientific papers, novels, blogs, dialogues, and medical texts, each with unique challenges. Key findings highlight the superior quality and flexibility of LLM generated summaries, the importance of prompt design, domain-specific training, and methods for ensuring consistency in outputs. [17] also reviews recent advancements in Automatic Text Summarization (ATS) with more research on optimization-based approaches. It emphasizes high accuracy of these techniques, particularly in metrics like ROUGE scores, and compares them to machine learning and deep learning methods. The review categorizes ATS methods by their ability to handle single/multiple documents and multi-lingual texts, highlighting the significant impact of Large Language Models (LLMs) on ATS.

Performance improvement of a pre-trained model includes fine-tuning, which is the process of adapting the model to a specific task by reducing the dataset to a task-specific one. [27] details the fine-tuning of LLMs, specifically LLaMA, using proprietary documents and code from an enterprise repository to generate domain-specific knowledge. It offers practical guidance on GPU requirements, data formatting, and dataset preparation.

The study also includes experiments with different data preparation techniques and manual evaluation of model responses using various LORA configurations. [25] categorizes multiple prompt engineering techniques used to enhance LLMs and vision language models (VLMs) by guiding model behavior with task-specific prompts. It reviews methodologies, applications, and datasets, while highlighting challenges. This research also explores future trends, such as meta learning and hybrid prompting. [3] examines the critical role of prompt engineering in optimizing LLMs, covering techniques like role-prompting, one-shot, few-shot prompting, and chain-of-thought approaches. It also discusses plugins for reducing hallucinations and improving performance. It also highlights applications in education and programming, underscoring the transformative impact of prompt engineering, with the future research directions to focus on understanding agent roles in AI-Generated Content (AIGC) tools.

Expert Systems utilize domain specific knowledge and rules to analyze solutions for problems. [11] reviews Expert Systems, detailing their core components – knowledge base, inference engine, and user interface and their role in problem-solving. It discusses knowledge representation methods like semantic networks and rule-based systems, and inference mechanisms such as forward and backward chaining. It emphasizes the usage of Expert Systems in structural designs and engineering for solving complex problems. [6] examines the integration of AI tools with domain expertise, focusing on explainability challenges in machine learning (ML) tools for seismic data analysis. The study suggests involving ML professionals as end-users to better address explainability concerns and emphasizes the need for context-specific AI explanations with future research on the social aspects of explainability and the impact of training ML experts in domain knowledge to improve AI design and effectiveness. [2] explores enhancing space mission design through an Expert System (ES) using AI, Knowledge Representation and Reasoning (KRR), and a Knowledge Graph (KG). It details the development of a Design Engineering Assistant (DEA) that utilizes NLP, ML, and Ontology Learning to structure unstructured knowledge. The study highlights the importance of multi-word extraction and plans to refine linguistic methods and integrate structured data. Future work aims to validate DEA components and improve query handling for effective space mission design.

Retrieval-augmented generation is a well-established method which answers user questions across entire datasets. It is particularly effective in scenarios where the answers are found within specific sections of text, and retrieving these sections provides adequate grounding for the generation task. [5] focuses on building a graph-based text index using LLM, such that it can derive knowledge graph from source documents and can generate summaries for closely related entity groups. Here, for every question a community summary is generated which is combined and summarized together with all such partial responses. [9] revolves around Dynamic-Relevant Retrieval-Augmented Generation (DR-RAG), for retrieval recall and maintenance of accuracy of answers. The DR-RAG works by concatenating the query documents, after specific retrieval of tags called tokens, then select or deselect documents based on relevance. The result depended on whether relevant context is provided to any LLM or not, because of the hallucination problem with LLMs. Enhancing of robustness of various response scenarios is discussed in [16], with dialogue generation. The response quality degrades with the increase in irrelevance of the input documents to the context. Addressing this they developed a RAG based response generator system utilized GPT-3.5, Jurassic, Claude2 and advanced prompting, provided through Mixtral-7B.

2. Proposed methodology

The anomaly detection for telemetry data is done for the SMAP (Soil moisture active passive) and MSL (Mars science laboratory) datasets. This telemetry data was available as time series data on the internet. The data itself is divided into channels to be fed into the model. This data is used for training the Long Short-Term model, and once the model is fed with the data, at particular time series for each channel the model will start predictions. The predicted values are compared with actual data, and error between them is generated. These errors based on the dynamic thresholding concept, where errors greater than a certain threshold will be classified as anomalies, and are printed out in csv format. This happens in a loop for every channel, until there is user intervention or when all the channels are covered.

The results of the LSTM model, containing anomalies are fed to the LLM including with relevant documents related to the mission. This will serve as a knowledge base to the LLM. They will be converted to vector embeddings, and stored in a vector database. Whenever a user asks a question, the LLM answers the question using the knowledge base, or replies that it cannot answer the question. This is how the whole architecture works, and the final output is derived, to understand more we divided them into various sections with relevant explanations as given below, LSTM – for anomaly detection, LLM – for text output generation, UI – for interacting easily.

Fig. 1 illustrates the methodology diagram, providing a clear overview of the research based on the above-mentioned approach.

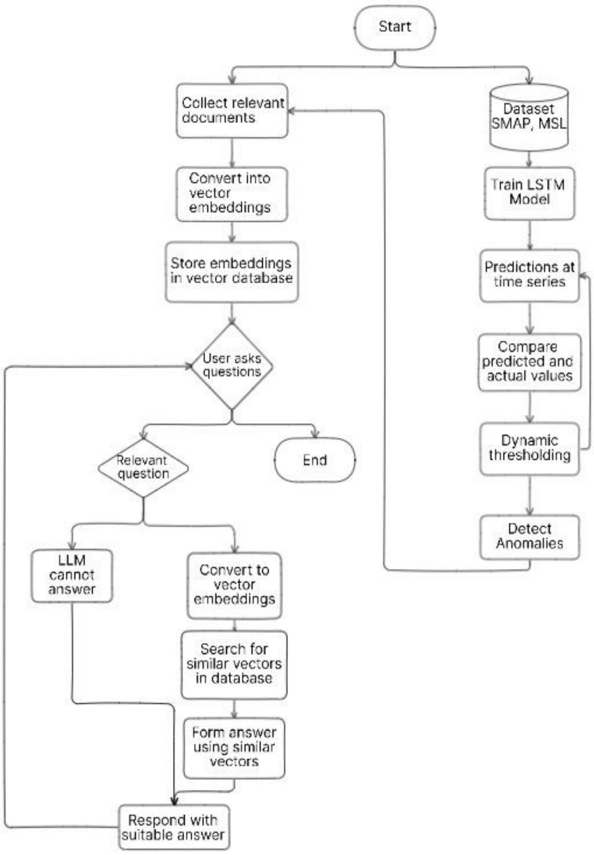

Start
Collect relevant documents
Dataset SMAP, MSL
Convert into vector embeddings
Train LSTM Model
Store embeddings in vector database
Predictions at time series
User asks questions
Compare predicted and actual values
Relevant question
End
Dynamic thresholding
LLM cannot answer
Convert to vector embeddings
Detect Anomalies
Search for similar vectors in database
Form answer using similar vectors
Respond with suitable answer

Fig. 1. Architecture diagram

2.1. LSTM

Satellite Telemetry data of SMAP and MSL data sets has been used to develop a long short-term model to detect anomalies in the telemetry data. Long Short-Term Memory (LSTM) networks are a specialized type of recurrent neural network (RNN) developed to address the vanishing gradient issue that often arises in traditional RNNs. LSTMs excel in capturing long-term dependencies, making them ideal for tasks that involve sequential data. An LSTM network is composed of repeating cells, each containing four key components: an input gate, output gate, cell state, and forget gate. These gates control the movement of information within the cell, ensuring efficient data processing.

The architecture of the model consists of hidden layers 2, units in a hidden layer being 80, the number of training iterations being 35. This architecture gave enough capacity. A single model is created and used for every channel. This model itself predicts values for every channel. For a time-series X = $\{x_1, x_2, x_3 \ldots . x_n\}$ where x_T is an m-dimensional vector, with elements being our input variables. Here, we are using single channel as input and are trying to get single channel outputs to reduce computational overhead. The input x_t being fed into the LSTM model will consist of previous telemetry values for a channel and encoded information of the satellite. After the values are predicted, let the predicted value be y_t, for every t, an error is calculated which is called the prediction error. Error $e_t = |y_t - Y_t|$ where Y_t is x_{t+1}. Every e_t is combined to form a one-dimensional vector. Eq. 1 constructs an error vector by assembling the current prediction error alongside a window of past errors, capturing temporal dependencies.

$$e = [e_{t-d}, \ldots \ldots e_{t-l}, \ldots \ldots . e_{t-1}, e_t] \tag{1}$$

LSTM based predictions often generate spikes in errors, so smoothing them is required which is done here by taking the averages. Eq. 2 shows how the raw error vector is then smoothed by averaging, resulting in a new vector that mitigates the impact of transient spikes.

$$eS = [eS_{t-d}, \ldots \ldots eS_{t-l}, \ldots \ldots . eS_{t-1}, eS_t] \tag{2}$$

Now, these values need to be checked if they are normal or actual anomalies. This is done by setting a threshold for the smoothed errors and all those values above the threshold are classified as anomalies.

Threshold (E) can be determined using Eq. 3.

$$\epsilon = \mu(e_s) + z\sigma(e_s) \tag{3}$$

where ϵ is determined by

$$\epsilon = argmax(\epsilon) = \frac{\frac{\Delta\mu(e_s)}{\mu(e_s)} + \frac{\Delta\sigma(e_s)}{\sigma(e_s)}}{|e_a| + |E_{seq}|^2} \tag{4}$$

Eq. 4 determines optimal threshold by maximizing a function that considers changes in the mean and standard deviation of the smoothed errors, normalized by the magnitude of specific error terms.

False or fake anomalies are those which are predicted by the model as anomalies, but they are not. These false anomalies can be prevented by learning from history. Learning from the history includes trying to feed labelled data, that is data which true or actual anomalies. Assuming that anomalies of same magnitude are not often repeated in the same channel, we can introduce a minimum most score, s_{min}, where future anomalies with s_{min} will be classified as nominal. This threshold is set for channels that generate anomalies frequently, and s_{min} is tailored individually for each channel. Previous anomaly scores of a channel can inform the setting of s_{min} balancing the need for precision and recall. Also, if the system allows users to label anomalies, these labels can help adjust s_{min} for specific streams. For instance, if a set of values has many confirmed false positives, s_{min} can be set close to the highest of these false positive scores. These strategies have been vital in improving the precision of system implementations by accounting for normal but rare spacecraft behaviors. For a stream of data with one or more anomalies, evaluation of all telemetry values in a near-by timeframe is done. The model is trained with each different and unique set of values. Each labelled sequence of anomalies of the telemetry values are evaluated with a final set of predicted anomalous sequences.

1. A true positive occurs when the $|e_s| > 0$, that is true positive occurs when a portion of sequence which is predicted for anomalies falls under a true labeled sequence being considered.
2. If not, even a single predicted sequence overlaps with a positive labeled telemetry anomaly from the labeled sequence, it becomes a false negative.
3. Any predicted sequence that does not overlap with the actual anomalies becomes a false positive.

Telemetry values are combined and are processed in batches. Every batch is evaluated with 2100 prior values that are used to calculate threshold's error and to find the performance of the present batch.

2.2. LLM

LLMs are playing a huge role in today's development in the AI landscape. LLM stands for Large Language Models. They find applications in diverse fields such as automated customer support, content creation, and language translation, where they can generate coherent responses and facilitate communication. Additionally, LLMs are used in research and development to enhance tools like chatbots, virtual assistants, and educational software by providing contextually relevant and nuanced interactions. LLMs use a deep learning architecture known as transformers at the core. Transformers are neural networks specifically designed for processing sequential data like text. They excel at identifying relationships between words and understanding the overall context of a sentence. During the training process, LLMs are exposed to massive amounts of text data, including books, articles, code, and web pages. By analysing these vast datasets, transformers within the LLM learn to recognize patterns in language usage and statistical relationships between words. Once trained, LLMs can perform various tasks related to language processing. They can generate different creative text formats like code, scripts, poems, musical pieces.

Additionally, they can answer questions in an informative and comprehensive way, following the context of the conversation. Essentially, LLMs leverage their learned understanding of language to predict the upcoming word or phrase in a sequence, allowing them to generate human-like text or complete a user's query. The LLM we used in the project is the Mixtral 8x7B model. Mixtral 8x7B is a powerful open-source decoder-only sparse mixture-of-experts (SMoE) language model. It stands out for its ability to achieve high performance with efficient use of parameters. Unlike traditional models that use all parameters for every token, Mixtral employs an SMoE architecture. This is a technique where multiple smaller models (experts) compete to handle a task, and a separate "router" network selects the most suitable expert(s) for each input. In Mixtral, there are 8 expert models. There are 8 experts participating in the SMoE architecture, and each expert is trained on 7 billion parameters. Hence the name, Mixtral 8x7B. SMoE architecture allows Mixtral to achieve performance comparable to much larger models (e.g., GPT-3.5) while requiring less computational power. Additionally, Mixtral is pre-trained on a large dataset consisting of text and code, enabling it for various tasks like text generation, code completion, and question answering. LLMs are limited by their training data. They lack domain specific data. They cannot answer questions of highly specialised domains. One of the key challenges associated with LLMs is their tendency to produce outputs that are factually incorrect, nonsensical, or irrelevant to the prompt – a phenomenon known as hallucinations. They can lack specific knowledge, struggle with factual accuracy, and may not be suited for specialised domains. Retrieval-Augmented Generation (RAG) is a kind

of technique designed to improve the performance of Large Language Models (LLMs). It works by combining information retrieval with LLM generation to make the responses more accurate and relevant. When you provide a prompt or query, RAG first retrieves useful information from an external source, such as factual passages or documents. This additional information is then given to the LLM as extra context. By using this external knowledge, the LLM can produce more detailed and precise answers tailored to specific topics. Our approach uses Retrieval-Augmented Generation (RAG) to improve the Large Language Model's (LLM) responses. By providing the LLM with relevant documents, such as documents related to specific missions, it can better answer user questions based on the content of those documents. Besides answering questions, the LLM can also summarize the documents, making it easier to understand and utilize the information. A critical component within the RAG process is parsing, the process of analyzing and structuring information from retrieved documents. Effective parsing plays a pivotal role in better extraction of data, thus ensuring the success of RAG by providing the LLM with well-organized and relevant context for generating accurate and informative responses. We used LlamaParse to parse our data because it is specifically designed for Large Language Model (LLM) use cases. LlamaParse excels in handling a variety of data formats, including tables and JSON, and supports over 10 different file formats. This robust support makes it the ideal choice for efficiently parsing and organizing complex documents, ensuring that our LLM
can process and utilize the information effectively. To optimize our data processing workflow, we stored the parsed data in a .pkl file.

This approach prevents the need to reparse the same data multiple times, which helps in conserving computational resources and reducing processing time. When new documents are added, we reparse only the updated or new entries and update the .pkl file accordingly. This method ensures that our system remains efficient by minimizing redundant computations and speeds up the overall data handling process. After parsing the data, we convert the documents into vector embeddings. Vector embeddings are a powerful technique for representing words or other linguistic units as numerical vectors in a high-dimensional space. These vectors capture the semantic relationships and similarities between words. Words with similar meanings tend to be positioned close together in this vector space, while words with distinct meanings reside farther apart. Nomic embeddings is an open-source text embedding model that converts semantic data into vector embeddings, which are then stored in a vector database. A vector database is specifically designed to manage these vector embeddings by using distance metrics to identify and retrieve similar data points based on their semantic relationships, rather than relying solely on keyword matches. This approach enhances the efficiency of data retrieval, allowing for similarity searches across multiple embeddings, and offers significant advantages such as scalability and flexibility. For our purposes, we use Qdrant as our vector database. Qdrant supports high-performance indexing and querying of vector embeddings, making it well-suited for managing and searching large volumes of semantic data efficiently. After generating the embeddings using Nomic, we store them in Qdrant vector database. The LLM can be run locally using Ollama or be using online services like Groq. Groq is an online service which uses Language Processing Unit (LPU). An LPU has a greater compute capacity when compared to a GPU and CPU in the context to LLMs. This will reduce the amount of time for every word in calculation, which allows the sequences of text to be generated much faster.

A detailed explanation of how the functionality works:

1. User Interaction: The workflow begins with the user either submitting documents or opting not to. If no documents are submitted, the Large Language Model (LLM) will generate responses based solely on the knowledge it was trained on, which may be limited to general information or prior training data.
2. Document Submission: If the user chooses to submit documents, these documents are first processed to extract relevant content. This involves parsing the documents using tools designed to handle various file formats and structures.
3. Data Conversion and Storage: After parsing, the extracted content is converted into vector embeddings using models like Nomic. These embeddings are numerical representations that capture the semantic meaning of the text. The embeddings are then stored in a vector database, such as Qdrant. This database organizes and maintains these embeddings, allowing for efficient retrieval based on semantic similarity.
4. Query Handling: When the user poses a question, the system searches the vector database for content related to the query. This search retrieves documents or passages that are semantically similar to the question.
5. Response Generation: The retrieved content is sent to the LLM, which processes this information. The LLM then converts the relevant content into coherent and contextually appropriate responses, which are returned to the user as answers to their questions.

The Large Language Model (LLM) can be executed either locally using tools like Ollama or through online services such as Groq. Ollama allows for local deployment of LLMs, providing control over hardware and configuration but requiring substantial computational resources. Groq offers a cloud-based solution leveraging Language Processing Units (LPUs), which are specialized hardware designed for optimal performance with LLMs. LPUs surpass GPUs and CPUs in handling the parallel processing and data throughput needed for language models, leading to faster generation and processing of text.

2.3. User interface

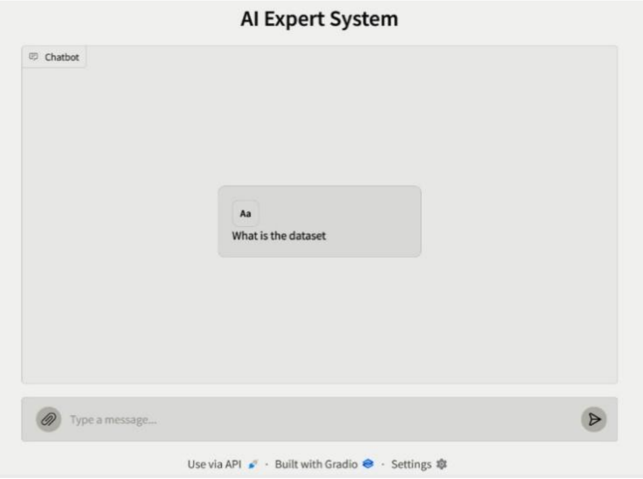
We utilized Gradio, a Python package known for its ease of use in creating interactive web interfaces, to develop the user interface for our system. Gradio simplifies the process of building functional and intuitive UIs, which was essential for ensuring a seamless user experience.

Our Gradio-based interface was designed with simplicity and functionality in mind. It features a clean layout with two main components:

1. File Upload Section: This component allows users to upload documents directly into the system. Users can drag and drop files or select them from their device. Once uploaded, these documents are automatically processed to be parsed, converted into embeddings, and stored in the vector database. This functionality enables the system to expand its knowledge base with new, user-provided content, enhancing the accuracy and relevance of the LLM's responses.
2. Question Input Field: This section is where users can input their queries. The system uses the question to search for relevant information within the knowledge base, which includes both the original training data and any additional documents provided by the user. The Gradio interface captures the user's question, sends it to the backend where it interacts with the LLM, and displays the generated response in a user-friendly manner.

Gradio’s interactive features allow users to easily switch between uploading documents and asking questions. The interface is designed to be intuitive. This setup ensures that users can effectively utilize the system's capabilities without encountering technical difficulties.

Fig. 2 illustrates the user interface, where users can interact with the system through a conversation. They can type questions into the input box, and responses generated by the LLM are displayed above. Users can view the entire conversation and also attach documents in the input bar, which are then parsed and used to answer the users’ questions.



AI Expert System

What is the dataset

Fig. 2. GUI of the application

Use this paper as reference.

Fig. 3. Attaching documents to be parsed

Fig. 3 shows how users can submit documents to the system. These documents are then parsed and used to answer user queries

3. Results

3.1. LSTM – precision metrics

The image shows a comparison graph for actual values and predicted telemetry values. This can be observed near value of 1 at y-axis. X-axis here represents the number of telemetry values under consideration, Y-axis represents the whole input being normalized to values at 1. The purple bars show few spikes of predicted values compared to actual ones.

$$Precision = \frac{TP}{TP+FP} \tag{5}$$

$$Recall = \frac{TP}{TP+FN} \tag{6}$$

$$F(0.5) = \frac{(1+0.5^2)*Precision*Recall}{0.5^2*Precision*Recall} \tag{7}$$

Table I. Precision metrics for LSTM model

Space Craft	Precision	Recall	$F_{0.5}$ *score*
SMAP	85.5%	85.5%	0.71
MSL	87.5%	80.0%	0.71

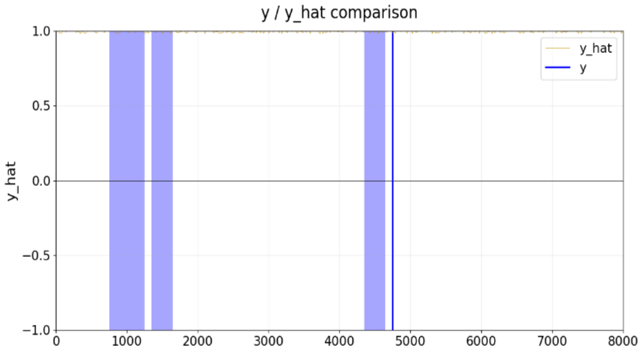


Fig. 4. Results of LSTM Model

Fig. 4 shows the comparison between the actual values (y) and the predicted values (y_hat) from the LSTM model. The plot reveals significant deviations between y and y_hat at specific points, indicating potential anomalies detected by the model. These deviations are highlighted by the blue shaded regions, where the actual values diverge substantially from the predicted values.

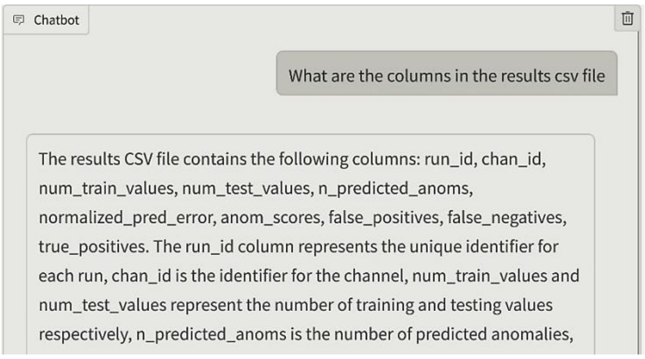
3.2. LLM – input and output

The Large Language Model (LLM) plays a central role in the system's question-answering capability. It is responsible for interpreting text-based queries, accessing relevant information stored within the knowledge base (a vector database), and generating well-structured, meaningful responses. The LLM is tightly integrated with the vector database, where pre-processed knowledge is encoded as vector embeddings. This design allows for efficient retrieval of information that is both contextually relevant and accurate. By comparing the query with the stored embeddings, the system identifies the most relevant pieces of data, ensuring that the generated responses are not only coherent but also aligned with the specific context of the query.

Input: The input consists of a user-provided text-based query. The questions must pertain to the knowledge base, specifically focusing on space missions or related topics. The LLM analyzes this input, drawing from its stored knowledge.

Output: The output is a text-based answer that accurately corresponds to the query's context. By using the most relevant information retrieved from the vector database, the LLM ensures that the response is precise, contextually appropriate, and presented in a clear, coherent manner. This ensures the system maintains both the reliability and relevance of the answers it provides.

Fig. 5 illustrates how a user interacts with the system. The user types a question in the chat interface, and the system responds with an answer derived from the provided documents. The response is displayed in a structured format and simple language, ensuring clarity and relevance to the user's query.



What are the columns in the results csv file

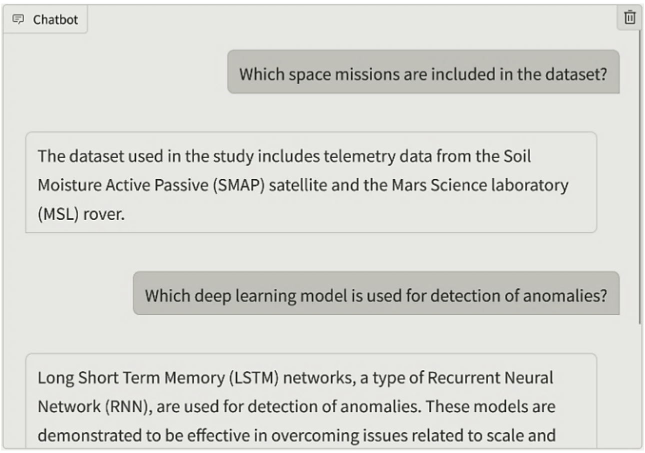
The results CSV file contains the following columns: run_id, chan_id, num_train_values, num_test_values, n_predicted_anoms, normalized_pred_error, anom_scores, false_positives, false_negatives, true_positives. The run_id column represents the unique identifier for each run, chan_id is the identifier for the channel, num_train_values and num_test_values represent the number of training and testing values respectively, n_predicted_anoms is the number of predicted anomalies,

Fig. 5. Input and output of LLM

Fig. 6. The documents constituting knowledge base when 2 files are added by user

Fig. 6 shows the various documents, stored locally, which are submitted by users and are used to answer user questions. The retrieved vector embeddings guide the LLM in generating responses that are contextually accurate and maintain narrative coherence. The model is fine-tuned to ensure that the generated sentences not only reflect factual correctness but also exhibit logical flow, making the output intelligible and relevant to the user query.

Fig. 7 provides a visual representation of the users' conversation with the system. The system allows users to ask clarifying questions or explore related topics The user can read through the whole conversation by scrolling.



Which space missions are included in the dataset?

The dataset used in the study includes telemetry data from the Soil Moisture Active Passive (SMAP) satellite and the Mars Science laboratory (MSL) rover.

Which deep learning model is used for detection of anomalies?

Long Short Term Memory (LSTM) networks, a type of Recurrent Neural Network (RNN), are used for detection of anomalies. These models are demonstrated to be effective in overcoming issues related to scale and

Fig. 7. Having a conversation with LLM based expert AI agent

4. Conclusion

This research focuses on developing a system that not only detects anomalies in spacecraft telemetry data using deep learning models, but also provides explanations for these anomalies to facilitate user understanding. This is particularly crucial for non-technical users who might require clear explanations of complex deep learning results. The system leverages in-depth analysis of telemetry data, utilising Long Short-Term Memory (LSTM) networks. These models aim to classify telemetry data from space missions into two distinct categories: "Anomaly" and "Not Anomaly". This classification aids users in making informed decisions based on the received telemetry data and the identified anomalies.

The system also integrates a Large Language Model (LLM) capable of parsing and understanding the data produced by the anomaly detection models. The LLM can answer questions related to the results, ensuring that users receive accurate information.

Additionally, users can submit documents that will be parsed and used as knowledge base for the LLM. This allows the system to answer questions based on the provided documents. Furthermore, the project implements document summarization capabilities, allowing the LLM to provide concise overviews of the analysed documents. It is crucial to emphasise that the LLM’s responses must be grounded in information derived from the processed documents. The system is designed to avoid hallucinating or generating answers based on its own knowledge or external sources. This ensures the reliability and trustworthiness of the LLM’s outputs, particularly when responding to user queries.

References

[1] Ahn H. et al.: Deep generative models-based anomaly detection for spacecraft control systems. Sensors 20(7), 2020, 1991.

[2] Berquand A. et al.: Artificial intelligence for the early design phases of space missions. 2019 IEEE Aerospace Conference, IEEE, 2019.

[3] Chen B. et al.: Unleashing the potential of prompt engineering in large language models: a comprehensive review. arXiv, 2310.14735, 2023.

[4] Cuéllar S. et al.: Explainable anomaly detection in spacecraft telemetry. Engineering Applications of Artificial Intelligence 133, 2024, 108083.

[5] Edge D. et al.: From local to global: A graph rag approach to query-focused summarization. arXiv, 2404.16130, 2024.

[6] Ferreira J. J., de Souza Monteiro M.: Do ML Experts Discuss Explainability for AI Systems? A discussion case in the industry for a domain-specific solution. arXiv, 2002.12450, 2020.

[7] Florin et al.: The power of noise: Redefining retrieval for rag systems. Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2024.

[8] Furano G. et al.: AI in space: Applications examples and challenges. IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT). IEEE, 2020.

[9] Hei Z. et al.: Dr-rag: Applying dynamic document relevance to retrieval-augmented generation for question-answering. arXiv, 2406.07348, 2024.

[10] Herrmann L. et al.: Unmasking overestimation: a re-evaluation of deep anomaly detection in spacecraft telemetry. CEAS Space Journal 16(2), 2024, 225–237.

[11] Janjanam D. et al.: Design of an expert system architecture: An overview. Journal of Physics: Conference Series 1767(1), 2021.

[12] Jin H. et al.: A comprehensive survey on process-oriented automatic text summarization with exploration of LLM-based methods. arXiv, 2403.02901, 2024.

[13] Josphineleela R. et al.: Exploration beyond boundaries: Ai-based advancements in rover robotics for lunar missions space like chandrayaan. Int. J. Intell. Syst. Appl. Eng 11(10s), 2023, 640–648.

[14] Ke Y. et al.: Development and Testing of Retrieval Augmented Generation in Large Language Models--A Case Study Report. arXiv, 2402.01733, 2024.

[15] Liu D. et al.: Fragment anomaly detection with prediction and statistical analysis for satellite telemetry. IEEE Access 5, 2017, 19269–19281.

[16] Liu S. et al.: Towards a robust retrieval-based summarization system. arXiv, 2403.19889, 2024.

[17] Muhammad H. H. et al.: A review on optimization-based automatic text summarization approach. IEEE Access 12, 2023, 4892–4909.

[18] Murdaca F. et al.: Knowledge-based information extraction from datasheets of space parts. 8th International Systems & Concurrent Engineering for Space Applications Conference 2018.

[19] Nalepa J. et al.: Evaluating algorithms for anomaly detection in satellite telemetry data. Acta Astronautica 198, 2022, 689–701.

[20] Obied M. A. et al.: Deep clustering-based anomaly detection and health monitoring for satellite telemetry. Big Data and Cognitive Computing 7(1), 2023, 39.

[21] O'Meara C. et al.: Applications of deep learning neural networks to satellite telemetry monitoring. 2018 Spaceops Conference.

[22] Ostaszewski K. et al.: Pattern recognition in time series for space missions: A rosetta magnetic field case study. Acta Astronautica 168, 2020, 123–129.

[23] Pilastre B. et al.: Anomaly detection in mixed telemetry data using a sparse representation and dictionary learning. Signal Processing 168, 2020, 107320.

[24] Purwar A.: Evaluating the efficacy of open-source llms in enterprise-specific rag systems: A comparative study of performance and scalability. arXiv, 2406.11424, 2024.

[25] Sahoo P. et al.: A systematic survey of prompt engineering in large language models: Techniques and applications. arXiv preprint arXiv, 2402.07927, 2024.

[26] Schefels C. et al.: To Catch Them All: A Generic Approach for Pattern Detection in Time Series Satellite Telemetry Data. 2021.

[27] Raj Mathav J. et al.: Fine tuning llm for enterprise: Practical guidelines and recommendations. arXiv, 2404.10779, 2024.

[28] Waisberg E. et al.: Generative pre-trained transformers (GPT) and space health: a potential frontier in astronaut health during exploration missions. Prehospital and Disaster Medicine 38(4), 2023, 532–536.

[29] Wang Y. et al.: A deep learning anomaly detection framework for satellite telemetry with fake anomalies. International Journal of Aerospace Engineering 1, 2022, 1676933.

[30] Zeng Z. et al.: Satellite telemetry data anomaly detection using causal network and feature-attention-based LSTM. IEEE Transactions on Instrumentation and Measurement 71, 2022, 1–21.

Prof. Sobhana Mummaneni
e-mail: sobhana@vrsiddhartha.ac.in

Dr. Sobhana Mummaneni is currently working as an associate professor in the Department of Computer Science and Engineering, V. R. Siddhartha Engineering College, Vijayawada, India. She received Ph.D. degree in Computer Science and Engineering in 2018 from Krishna University. She has 16 years of teaching experience. Her research interests lie in areas such as Artificial Intelligence, Machine Learning, Data Analytics, Cyber Security, and Software Engineering. She published 35 papers in national and international journals and published 7 patents.

https://orcid.org/0000-0001-5938-5740

Eng. Syama Sameera Gudipati
e-mail: gsyamasameera2004@gmail.com

Syama Sameera Gudipati is a fourth-year B. Tech. student specializing in Computer Science and Engineering at V. R. Siddhartha Engineering College, Vijayawada, India.
She is passionate about Machine Learning, and Web Technologies.

https://orcid.org/0009-0006-7405-7570

Eng. Satwik Panda
e-mail: satwik9903@gmail.com

Satwik Panda is a Fourth-year B. Tech. student specializing in Computer Science and Engineering at V. R. Siddhartha Engineering College, Vijayawada, India.
He is passionate about Web Development and Generative AI.

https://orcid.org/0009-0005-4154-9918