



Using Generative AI to implement the discrepancy checker for a Nearly Autonomous Management and Control System for Advanced Reactors

June 2024

Changing the World's Energy Future

Paridhi Athe, Nam Dinh, Linyu Lin



DISCLAIMER

This information was prepared as an account of work sponsored by an agency of the U.S. Government. Neither the U.S. Government nor any agency thereof, nor any of their employees, makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness, of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the U.S. Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the U.S. Government or any agency thereof.

Using Generative AI to implement the discrepancy checker for a Nearly Autonomous Management and Control System for Advanced Reactors

Paridhi Athe, Nam Dinh, Linyu Lin

June 2024

**Idaho National Laboratory
Idaho Falls, Idaho 83415**

<http://www.inl.gov>

**Prepared for the
U.S. Department of Energy
Under DOE Idaho Operations Office
Contract DE-AC07-05ID14517**

Using Generative AI to Implement the Discrepancy Checker for a Nearly Autonomous Management and Control System for Advanced Reactors

Paridhi Athe^{1*}, Linyu Lin², Nam Dinh¹

¹North Carolina State University, Raleigh, North Carolina; ²Idaho National Laboratory, Idaho Falls, Idaho

[leave space for DOI, which will be inserted by ANS – do not highlight this line]

ABSTRACT

Developments related to generative artificial intelligence (AI) have brought a major breakthrough in AI. These developments are rapidly accelerating developments in different science and engineering applications. Nearly Autonomous Management and Control (NAMAC) system provides recommendations to the operator for maintaining the safety and performance of the reactor. The discrepancy checker (DC) is an important component of the NAMAC) system, whose goal is to determine if the plant is moving towards the expected system state after the control actions are injected. In this work, we explore generative AI methods, particularly, a generative pretrained transformer (GPT) for implementing the DC function in NAMAC. The GPT-based DC aims to alert the operator in situations outside NAMAC’s scope and act as a chatbot the operator can use to retrieve relevant information.

This study involves two versions of GPT developed by OpenAI: GPT-3.5 and GPT-4. These GPTs are trained on huge amounts of undisclosed general domain datasets. We explored two methods to adapt GPTs for DC implementation in NAMAC: fine-tuning and retrieval augmented generation. A small knowledge base (information file) that encompasses rules for DC implementation and some general information related to NAMAC has been created to support DC implementation using GPT. In this work, the GPT-based DC implementations have been tested for their reasoning abilities, comprehension, information retrieval, and extraction abilities. It should be noted that this paper only presents a preliminary study to test the feasibility of DC implementation using generative AI technology. Given the potential risks and severe consequences associated with nuclear reactor applications, combined with the black-box nature of AI, extensive offline and online testing and reliability analyses of GPT-based DCs are needed for further developing such capabilities.

Keywords: Discrepancy checker, autonomous control system, generative AI, advanced reactor, generative pretrained transformer

1. INTRODUCTION

Over the past few decades, advances in computational methods and artificial intelligence (AI) have brought significant advances in science and engineering. In nuclear engineering (NE), data-driven modeling techniques based on AI algorithms have been increasingly used for different nuclear reactor applications [1], including improving predictive capability and efficiency by supporting closure development [2], and high-to-low modeling error corrections [3]; developing diagnostic and prognostic models for reactor health monitoring and control [4], [5]; fuel cycle optimization [6], etc. In our previous work [7], [8], we focused

*pathe@ncsu.edu

on developing and assessing a Nearly Autonomous Management and Control (NAMAC) system for advanced reactors. NAMAC involves different case studies with different levels of complexity related to loss-of-flow accidents in a sodium fast reactor (SFR). It provides recommendations to the operator for the safe operation and performance of the reactor. AI algorithms are used to develop diagnostic and prognostic digital twins (DTs) in NAMAC. An important component of the NAMAC system is the discrepancy checker (DC). The DC acts as an uncertainty manager and anomaly detector for the NAMAC system and reactor. It detects anomalies and alerts the operator in situations that are outside the scope of NAMAC (beyond the training domain). In our previous work, we used a simple DC implementation that detects discrepancies based on the deviation of reality from NAMAC predictions. The DC periodically checks whether NAMAC recommended actions, when administered, are leading to the intended outcome or not. In another implementation, the answer set programming (ASP) approach was used to implement the DC [9].

Generative AI algorithms, particularly the Transformer-based large language models (LLMs) have demonstrated advanced reasoning capabilities (e.g., actual causality, pairwise causal discovery, and counterfactual reasoning [10]) in a variety of tasks. In our previous work [11], we tested the knowledge abstraction, reasoning, and information retrieval abilities of the generative pretrained transformer (GPT) for nuclear reactor design and safety application. Our previous results indicated that GPT can support expert-driven elements in developing and assessing modeling and simulation tools used in the NE domain. They can be used as proxy experts, which can substantially reduce the time and resources needed to conduct expert elicitation.

In this paper, we make an initial attempt to test GPT for implementing the DC function in NAMAC. In particular, we analyzed the knowledge abstraction, reasoning, and information retrieval abilities of GPT to support implementing the DC in NAMAC. The conversational ability (question and answer) of GPTs also makes them useful as a chatbot that the operator can use to retrieve relevant information from operating manuals, safety analysis reports, technical specifications, emergency operating procedures, maintenance manuals, records, etc. The GPT is trained using general domain information. We analyzed two methods that may be used to adapt GPT for DC implementation. The first method requires fine-tuning the GPT while the second uses the retrieval augmented generation (RAG) [12] approach to augment the knowledge of GPT using external sources of information.

In this paper, Section 2 provides a brief description of NAMAC, its components, and the DC; Section 3 describes the generative AI methods (related to GPT) used in this study for implementing the DC; Section 4 presents the results and discussion; and Section 5 includes our conclusions in this study.

2. NAMAC AND DC DESCRIPTION

The NAMAC system development involved different case studies with different levels of complexity in the issue space and was centered around three basic elements [7]:

- Knowledge base—It encompasses the issue space, scenario-based plant model based on the issue space and training database created by sampling the issue space, which defines the scenarios in mathematical formulations. We used the Experimental Breeder Reactor II as the reference SFR and used GOTHIC to create a scenario-based model of it, focusing on the key components in the primary loop: two primary pumps, low-pressure lower plenum, high-pressure lower plenum, upper plenum, active core, and intermediate heat exchanger. The model involves some simplification and assumptions related to the number of channels, initial conditions, boundary conditions, etc. However, we deemed them sufficient for the NAMAC “proof of concept” demonstration and used the RAVEN-GOTHIC interface as a data generation engine to generate training data by sampling the issue space.
- DT—DTs in NAMAC are knowledge acquisition systems that support different NAMAC functions (e.g., diagnosis, strategy planning, prognosis, and strategy assessment). Within NAMAC, the diagnosis

and prognosis DTs are implemented using neural networks (see Table I for a basic description of NAMAC components).

- Operation workflow—This workflow consists of the systematic organization of DTs to support decision-making for operational recommendation.

In addition to these elements, the DC is also an important element of NAMAC. The DC acts as an uncertainty manager and anomaly detector for the NAMAC system and reactor. It detects anomalies and alerts the operator in situations that are outside the scope of NAMAC (beyond the training domain situation) [7]. The DC implementation is governed by a set of rules formulated during the phase of development and assessment of NAMAC. These rules are based on the range of key parameters in the training domain, constraints related to DTs, and specific safety or operational criteria that require immediate attention and intervention. In our previous work [7], a simple rule-based DC implementation was used to detect discrepancies based on the deviation of reality from NAMAC predictions. In this case, the DC checks whether NAMAC recommended actions, when administered, are leading to the intended outcome or not. Another implementation used the ASP approach to implement the DC. In this work [9], reliability metrics based on different criteria were used: similarity between the training and actual scenarios, discrepancy between the NAMAC prediction and actual scenario, and action effect discrepancies. These metrics were employed in a rule-based reasoning framework constructed using ASP.

Table I. Basic description of NAMAC DTs and components [7], [8].

DT	Function	Modeling
Diagnosis	Recover full reactor states by assimilating plant sensor data with the knowledge base	Neural nets (feedforward & recurrent); logic programming (ASP)
Strategy Inventory	Find all available control and mitigation strategies	Linear models (ensembled with measurements)
Prognosis	Predict the transients of state variables over a time range	Neural nets (feedforward & recurrent)
Strategy Assessment	Rank possible mitigation strategies and make recommendations considering the preference structure (user’s preference for safety, reliability, and performance)	Single-attribute; multi-attribute reward-based
Discrepancy Checker	Detect unexpected transient during operations considering DT trustworthiness for current conditions	Distance metrics; logic programming (ASP)

3. GENERATIVE AI FOR DC IMPLEMENTATION

Recently, developments related to generative AI have brought a breakthrough in AI. These developments are rapidly accelerating developments in different science and engineering applications. A major driving factor behind the recent developments in generative AI is the transformer architecture introduced by Vaswani et al. in 2017 [13]. The attention mechanism is the core of the transformer architecture. It allows the model to weigh the importance of different parts of the data in a sequence relative to each other (e.g., words in a sentence). By computing attention scores, the model can capture long-range dependencies and complex relationships within the input data. The transformer architecture has led to significant improvements in natural language processing (NLP) performance across a wide range of tasks, including language translation, text summarization, sentiment analysis, and question-answering. Its ability to effectively capture complex relationships in data has revolutionized generative AI and provided the foundation for LLMs and large models. These models have vast parametric memory that encapsulates large volumes of data. Notably, the LLMs developed by big corporations, for example, GPTs developed by OpenAI [14] and Llama developed by Meta AI [15], stand out. The NLP abilities of these models are

remarkable with inherent capabilities to understand the semantic and syntactic rules that govern the human language. GPT-4 is the latest version of GPT released in 2023. It has demonstrated a remarkable ability to comprehend and perform a range of tasks that require understanding complex instruction, advanced reasoning abilities, and creativity. These capabilities have inspired us to use it as a reasoning engine for DC implementation in NAMAC. We use the GPTs (GPT-3.5 Turbo and GPT-4) developed by OpenAI in this study. The information regarding their GPT's training database is not openly available due to proprietary reasons. However, OpenAI has specified that it used a vast amount of general domain data of different articles, science and engineering textbooks, webpages, etc. [14].

LLM development begins with the pretraining phase. During this phase, the LLM model is trained with massive datasets of texts and codes. The pretraining is self-supervised (i.e., the LLM does not require labeled data). During this phase, the model learns grammar rules, syntax, patterns, and facts from the natural language. In the subsequent training phase, human supervision and feedback further improves the LLM response. The pretrained LLM can be a foundation model for different downstream applications [16], [17]. It can be used directly if the LLM has appropriate domain knowledge. Otherwise, the LLM needs to be customized for use in specific domains. Our previous testing of GPT for NE applications indicates it has knowledge and understanding of nuclear reactor systems, including SFRs [11]. However, its knowledge is not comprehensive, and there are deficiencies. Therefore, it is important to customize the GPTs for their use in specific nuclear reactor applications. In this study, we consider two methods to customize the GPT for their use as a DC in NAMAC: fine-tuning and RAG [12].

Fine-tuning involves adjusting the parameters of the pretrained GPT model to adapt it to the new data. It should be noted that, during fine-tuning, only part of the model is available for weight adjustment. Usually, the initial layers that capture general language patterns, features, rules, and facts are frozen to keep the base capabilities of the model intact. Pretraining involves self-supervised learning. However, since fine-tuning involves supervised learning (i.e., for fine-tuning), we need labeled data. For instance, we used three labels in our training data: system, user, and assistant. The label “*system*” carries general instructions for the GPT for use in a specific domain (e.g., DC in NAMAC), the label “*user*” refers to the expected user's prompt, and the label “*assistant*” refers to the system (i.e., GPT) response. We uploaded the data set to the OpenAI platform on the cloud for fine-tuning. The data was trained with three epochs (i.e., the entire data was passed through the model three times). The fine-tuning duration took around 6 minutes for the dataset with over 7,500 tokens (i.e., a total of around 23,000 trained tokens). GPT-4 was not available for fine-tuning. Therefore, we only used GPT-3.5 Turbo for the fine-tuning job.

RAG does not involve any sort of additional GPT model training. Instead, it uses an external information retrieval system to augment the knowledge of GPT with external sources of information. In this context, the pretrained GPT is often considered a parametric memory (embedded within GPT) while the external source is considered a nonparametric memory that GPT accesses using the information retrieval system [12], [18]. The general scheme of RAG is shown in Figure 1.

As evident from Figure 1, for RAG, the external document is first broken down into chunks of data with a certain number of tokens. According to Manning et al. [19], “A token is an instance of a sequence of characters in some particular document that is grouped together as a useful semantic unit for processing.” It can be whole words, parts of words or subwords, individual characters, etc., depending on the type of tokenizer used. The tokenized data chunks are transformed into a numerical vector representation called embeddings using an embedding model. Embedding tends to capture the semantics of the words, by placing words with similar attributes closer in the embedding space [20]. The user query is first transformed into embedding using the embedding model (In this work, we used the embedding model developed by OpenAI called ADA-002. It has a dimension size of 1536.) Next, a similarity search is performed over the vector database to identify relevant information matching the user's query. The matched information is passed to the GPT to generate an appropriate response for the query. In this work, RAG is performed using two GPT versions: GPT-3.5 and GPT-4.

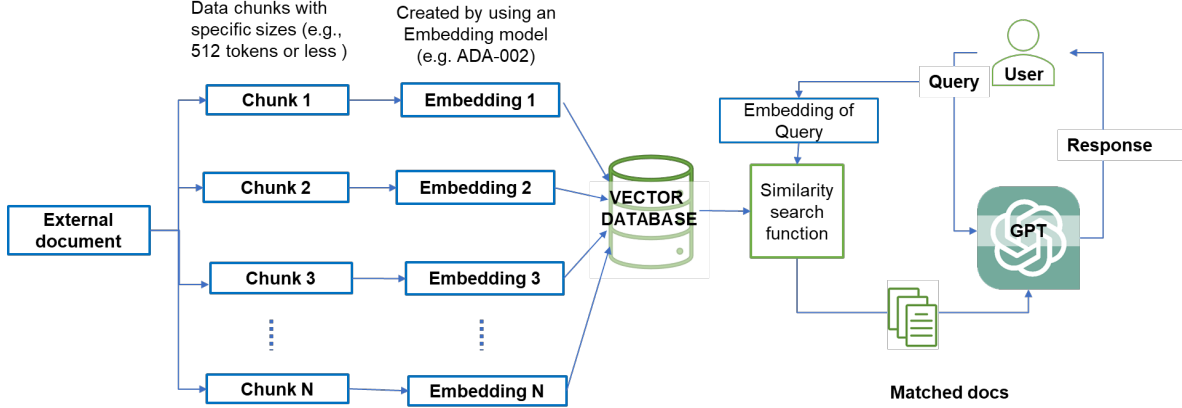


Figure 1: RAG using GPT-3.5.

For the DC implementation, we use a set of rules. These rules are formulated based on known constraints, safety criteria, testing results, and estimated bounds of uncertainties of the DTs and NAMAC components learned during the NAMAC development phase. The earlier DC implementations were also based on similar rules. For a DC implementation using GPT, we used the following two rules:

$$\{x(t) \in \mathbb{R} | X_{min} \leq x(t) \leq X_{max} \} \quad (1)$$

$$\{y_{DT-D}(t) \in \mathbb{R} | y_{DT-D}(t) < Y_l\} \quad (2)$$

Rule #1 (Eq. 1) is specified with respect to all the observable variables in the NAMAC training database. Here, $x(t)$ refers to the value of the variable x at time t and X_{min} and X_{max} refer to the respective minimum and maximum limit for the variable x . $y_{DT-D}(t)$ refers to diagnosis prediction at time t . Y_l refers to the safety limit for y_{DT-D} . Though we have specified Rule #1 for all the variables, it can only be used in the context of observable variables (that are available through sensor measurement). The minimum and maximum values of selective variables used in the tests included in this paper are shown in Table II.

Table II. Description of selective variables with minimum and maximum limits [7].

Variable Name	Notation	Minimum	Maximum
Top of the active core rod centerline temperature	TA21s1	600°C	685°C
High-pressure lower plenum sodium temperature	TL8s1	380°C	389°C
Low-pressure lower plenum sodium temperature	TL9s1	380°C	389°C
Upper plenum sodium temperature	TL14s1	480°C	580°C

In NAMAC, the diagnosis DT provides the real-time prediction of the top of the active core rod centerline temperature (TA21s1) based on the sensor measurement of different plenum temperatures (TL8s1, TL9s1, and TL14s). Rule #2 (Eq. 2) is specified in the context of safety-significant factors like the active core rod centerline temperature (TA21s1) or cladding temperature. These variables are not directly observable, so predictions from the diagnosis DT are used as a reference to identify situations where Rule #2 gets violated. Rule #2 helps in identifying situations where NAMAC predictions obtained from prognosis DT are not trustable, that is, the NAMAC recommended action does not move towards the expected outcome. We transformed these rules into a text form to create a knowledge base (information file) that can be used to support DC implementation using GPT. Apart from these rules, we also added general information regarding NAMAC (a few definitions from a previous NAMAC paper [7]) to the knowledge base for the DC. This knowledge base is different from the NAMAC knowledge base mentioned in Section 2. It is also fairly small with only a few pages of information containing DC rules, recommended actions for rule violations, and a few definitions related to NAMAC. The recommendation in case of all the rule violations

is to scram the reactor. This information is included in the information file as the message, “Alerts the operator that the situation is outside NAMAC’s training domain. Recommend to immediately SCRAM the reactor.”

We make use of the same knowledge base (information file) for both fine-tuning and RAG. In the case of fine-tuning, this information becomes a part of the parametric memory of GPT, while in the case of RAG, it stays as an external knowledge base or information source.

4. RESULTS AND DISCUSSION

This section presents the results of the DC implementation obtained by using fine-tuning and RAG. In this initial testing and exploration phase, we did not integrate GPT with NAMAC and the plant simulator. We tested the GPT in isolation by creating different prompts that encompass the rules and information included in the knowledge base we created for DC implementation (i.e., the information file used in fine-tuning and RAG). The GPT configuration can be altered by varying different hyperparameters related to the model, for example, “*Top-p*” and “*Temperature*” parameters can be modified to alter the degree of stochasticity of GPT’s response. In this work, we used the default values for these parameters suggested by OpenAI for a balanced response.

For Rule #1, we tested the GPT by prompting it with different values of the observable variables in NAMAC’s training data (e.g., plenum temperatures, TL8s1, TL9s1, TL14s1). For Rule #2, we prompted GPT with TA21s1. TA21s1 is an unobservable variable obtained from the diagnosis DT during NAMAC operation. Lastly, we queried the GPT with general information related to NAMAC that was included in the knowledge base (i.e., the information file we used for fine-tuning and RAG exercises).

Table III shows a set of responses obtained by querying the fine-tuned GPT-3.5 model. The fine-tuned GPT-3.5 seems to provide the same alert message for all the cases of rule violation. This alert message is based on the action recommendation for the rule violation specified in the knowledge base (information file). On additional testing, we found that the fine-tuned GPT-3.5 did not adequately adapt to the new information during fine-tuning and seems to provide the same message irrespective of the variable value and name. On querying GPT about the minimum and maximum limit of variables, it mostly provided wrong answers (see highlights in red color). It did provide correct answers sometimes, but those instances were few. When we asked the GPT some general questions related to NAMAC, it did provide relevant responses. However, in some cases, the response was different from the knowledge base (information file used in fine-tuning) and mixed with information from the pretraining phase or just hallucinations (see Response # 8 in Table III).

Table III. Response of fine-tuned GPT-3.5.

1	User: TL8s1=400 C ChatGPT: Alert! Alert! The situation is outside NAMAC's training domain. Recommend to immediately SCRAM the reactor.
2	User: TL14s1=600 C ChatGPT: Alert! Alert! The situation is outside NAMAC's training domain. Recommend to immediately SCRAM the reactor.
3	User: TA21s1=695 C in diagnosis time ChatGPT: Alert! Alert! The situation is outside NAMAC's training domain. Recommend to immediately SCRAM the reactor.
4	User: TL7s1=660 C ChatGPT: Alert! Alert! The situation is outside NAMAC's training domain. Recommend to immediately SCRAM the reactor.
5	User: TL8s1=385 C within the allowed range? ChatGPT: Yes, TL8s1=385 C is within the allowed range. User: Please explain. ChatGPT: The allowed range for TL8s1 is 400 C to 610 C. Therefore, TL8s1=385 C is within the allowed range.
6	User: What is the minimum and maximum limit for TL14s1s? ChatGPT: The minimum limit for TL14s1 is 400 C and the maximum limit is 580 C
7	User: What is NAMAC? ChatGPT: NAMAC stands for Nearly Autonomous Management and Control.
8	User: What are the different elements of NAMAC?

ChatGPT: NAMAC has three elements: **neural network (NN) for anomaly detection, NN for control-oriented surrogate model (COSM),** and discrepancy checker

9 **User:** what is the role of discrepancy checker?

ChatGPT: The discrepancy checker determines whether the situation is within NAMAC's training domain.

Table IV compares the RAG results using GPT-3.5 and GPT-4. For GPT-3.5, we used an external retrieval system for RAG (shown in Figure 1) created using Langchain [21] and the OpenAI developer platform. For GPT-4, we used the inbuilt functionality on OpenAI web interface for customizing GPT-4 and RAG. The specifics related to this RAG system and customization of GPT-4 are not openly available. We used the same knowledge base (information file) for RAG in both cases.

For RAG with GPT-3.5, we obtained mixed responses, with both positive and negative results. In most cases, GPT-3.5 was able to recall the minimum and maximum limit of variables. However, in some cases, it failed with the logic regarding a variable exceeding or not exceeding the permissible range (for reference, see Response #2 in Table IV). When we prompted GPT-3.5 with a wrong variable name (not existing in the knowledge base), it was able to identify the situation and responded with a message stating, it doesn't know the answer (see Response #4 in Table IV). When we asked GPT-3.5 general questions regarding NAMAC, it retrieved the correct information from the external knowledge base (see Table V).

Table IV. Results of RAG using GPT-3.5 and GPT-4.

	RAG using GPT-3.5	RAG using GPT-4
1	<p>User: TL8s1=400 C ChatGPT: TL8s1 exceeds the maximum limit of 389 C. The reactor should be immediately SCRAMed as the situation is beyond NAMAC's training domain.</p>	<p>User: TL8s1=400 C ChatGPT: The reported temperature for the High pressure lower plenum sodium temperature (TL8s1) is 400°C. According to the maximum and minimum limits provided in the knowledge source, the maximum limit for TL8s1 is 389°C, and the minimum limit is 380°C. Therefore, the reported temperature exceeds the maximum limit. Alert the reactor operator that the situation is outside NAMAC's training domain. Recommend to immediately SCRAM the reactor.</p>
2	<p>User: TL14s1=600 C ChatGPT: No, TL14s1 does not exceed the maximum limit of 580 C.</p>	<p>User: TL14s1=600 C ChatGPT: The reported temperature for the Upper plenum sodium temperature (TL14s1) is 600°C. According to the maximum and minimum limits provided in the knowledge source, the maximum limit for TL14s1 is 580°C, and the minimum limit is 480°C. Therefore, the reported temperature exceeds the maximum limit. Alert the reactor operator that the situation is outside NAMAC's training domain. Recommend to immediately SCRAM the reactor.</p>
3	<p>User: TA21s1=695 C in diagnosis time ChatGPT: Yes, the TA21s1 temperature of 695 C is exceeding the maximum limit.</p>	<p>User: TA21s1=695 C in diagnosis time ChatGPT: The reported temperature for the Top of the active core rod centerline temperature (TA21s1) during the diagnosis time is 695°C. According to the safety criteria provided in the knowledge source, if the TA21s1 exceeds 685°C during the diagnosis time, it is considered beyond the safety limit. Alert the reactor operator that the reactor is in a region beyond the safety limit. Recommend to immediately SCRAM the reactor.</p>
4	<p>User: TL7s1=660 C ChatGPT: We do not have any information about the maximum limit for TL7s1 sodium temperature. User: what does the discrepancy checker suggest? ChatGPT: The given context does not provide any information about the discrepancy checker's suggestion in the situation involving TL7s1 sodium temperature.</p>	<p>User: TL7s1=660 C ChatGPT: The parameter TL7s1 is not mentioned in the provided knowledge source. Therefore, without specific information on the maximum and minimum limits for TL7s1, I cannot accurately determine if the reported temperature of 660°C is within the operational limits or if it requires any specific action. For accurate assistance, please provide the operational limits for TL7s1 or refer to a relevant parameter within the documented limits.</p>
5	<p>User: TL8s1=385 C within the allowed range? ChatGPT: Yes. User: Please explain ChatGPT: Yes, TL8s1=385 C is within the allowed range. The maximum limit for High pressure lower plenum sodium temperature (TL8s1) is 389 C and the minimum limit is 380 C. Since 385 C falls within this range, it is considered to be within the allowed range.</p>	<p>User: TL8s1=385 C within the allowed range? ChatGPT: Yes, the reported temperature for the High pressure lower plenum sodium temperature (TL8s1) at 385°C is within the allowed range. The specified maximum and minimum limits for TL8s1 are 389°C and 380°C, respectively. Therefore, a temperature of 385°C falls comfortably within these limits, indicating that no immediate action is required based on this parameter alone, as it aligns with NAMAC's operational domain.</p>

6	User: What is the minimum and maximum limit for TL14s1s? ChatGPT: The minimum limit for TL14s1 is 480 C and the maximum limit is 580 C.	User: What is the minimum and maximum limit for TL14s1s? ChatGPT: The minimum limit for Upper plenum sodium temperature (TL14s1) is 480°C, and the maximum limit is 580°C.
---	--	---

In the case of RAG using GPT-4, the responses were accurate, comprehensive, and followed the right reasoning and logic in the context of Rule #1 and Rule #2 (see Table IV). It also retrieved the right information when we queried general questions related to NAMAC (see Table V). We did encounter a few cases of hallucination and incorrect information in the case of GPT-4 as well. However, such cases were rare and mostly happened due to ambiguous information or lack of adequate context in the prompt. Compared to the scale of data used in the GPT training, the knowledge base we created for RAG (and fine-tuning) is extremely small and limited in context. So, such deviation in behavior is not surprising. However, given the high risks and consequences associated with nuclear reactor application and autonomous control, we need to have stringent requirements, comprehensive and exhaustive testing protocols, and methods for the use and adaptation of generative AI models in NE applications.

Table V. Results of RAG using GPT-3.5 and GPT-4 [22].

	RAG Using GPT-3.5	RAG Using GPT-4
1	User: what is NAMAC? ChatGPT: NAMAC is a computerized safety case that aims to achieve an alignment of nuclear power plant safety design, analysis, operator training, and emergency management by furnishing recommendations to operators for effective actions that will achieve particular goals, based on the NAMAC's knowledge of the current plant state, prediction of the future state transients, and reflecting the uncertainties that complicate the determination of mitigating strategies.	User: what is NAMAC? ChatGPT: NAMAC is a computerized safety case that aims to achieve an alignment of nuclear power plant safety design, analysis, operator training, and emergency management by furnishing recommendations to operators for effective actions that will achieve particular goals. This is based on NAMAC's knowledge of the current plant state, prediction of the future state transients, and reflecting the uncertainties that complicate the determination of mitigating strategies.
2	User: What are the different elements of NAMAC? ChatGPT: The different elements included in NAMAC are the Knowledge base, Digital Twin (DT), and Operational workflow.	User: What are the different elements of NAMAC? ChatGPT: The development of the NAMAC system is based on three elements: Knowledge Base: A class of databases, scenarios, and models to support the control and risk management of the reactor. Digital Twin (DT): A knowledge acquisition system to support different NAMAC functions such as diagnosis, strategy planning, prognosis, strategy assessment, etc. Operational Workflow: An assembly of DTs to support the operator's decision-making or to make direct operational recommendations.
3	User: what is the role of discrepancy checker? ChatGPT: The role of the discrepancy checker in the NAMAC system is to notify the reactor operator whenever there is an anomaly and the NAMAC prediction deviates from reality. It alerts the operator when the situation is outside the NAMAC's training domain and recommends the operator to immediately SCRAM the reactor.	User: what is the role of discrepancy checker? ChatGPT: The role of the discrepancy checker in the Nearly Autonomous Management and Control (NAMAC) system is to determine if the plant is moving towards the expected system state after the control actions recommended by NAMAC are injected by the operator. It acts as an uncertainty manager and alerts the operator whenever a situation that is outside the training domain of NAMAC is encountered, ensuring the safety and proper operation of the reactor.

The challenges and methods to address different elements related to the trustworthiness of generative AI technologies have garnered significant interest and attention [23]. Huang et al. [24] suggested a citation mechanism to enhance the transparency and verifiability of LLM-generated content. A similar citation mechanism is also evident in Microsoft's "Bing AI search" [25], which assimilated an LLM-based reasoning engine with a search engine. It provides appropriate citations for the generated content, especially when the user asks for precise information. Wang et al. [26] suggest incorporating a probabilistic inference layer in the LLM to increase its accuracy and reliability in information retrieval tasks. We need to incorporate a similar approach for verifiability and fact-checking for the use of generative AI models in NE applications.

Apart from these issues, there are also AI ethics, copyright issues, fairness, and regulatory concerns [27] that need to be addressed as we start adopting generative AI technologies for engineering and science applications.

5. CONCLUSIONS

In this work, we explore generative AI methods, particularly GPT for implementing the DC function in NAMAC. The GPT-based DC aims to support two functions: alert the operator in situations that are outside NAMAC's scope and act as a chatbot that the operator can use to retrieve relevant information. This study is focused on two versions of GPT developed by OpenAI: GPT-3.5 and GPT-4. These GPTs are trained on huge amounts of general domain datasets. To adapt GPT for DC implementation in NAMAC, we explore two methods: fine-tuning and RAG. A small knowledge base (information file) that encompasses rules for DC implementation and some general information related to NAMAC has been created to support DC implementation in NAMAC.

Our testing results indicate that the fine-tuning of GPT-3.5 did not provide the expected outcome. We need further analysis with respect to the quality and quantity of training data and optimization of hyperparameters related to the fine-tuning tasks. RAG using GPT-3.5 seems better compared to the fine-tuned GPT-3.5. However, it failed to show the required reasoning ability for DC implementation. The results of testing GPT-4 with RAG indicated a positive outcome and displayed appropriate reasoning, comprehension, information retrieval, and extraction capabilities needed to support DC implementation in NAMAC. Though the results seem promising, we would like to emphasize this paper only presents a preliminary study to test the feasibility of DC implementation using generative AI technology. Given the potential risks and severe consequences associated with nuclear reactor application, combined with the black-box nature of AI, extensive offline and online testing and reliability analysis are needed for the further development of such capabilities. Furthermore, there are certain aspects related to the integration of a GPT-based DC in the NAMAC operational workflow that require additional analysis, expansion in tasks, and other considerations.

ACKNOWLEDGMENTS

This work was performed with support from of the Idaho National Laboratory through a contract 264247-14 "Uncertainty Quantification Approach for Digital Twin-based Autonomous Control", and the U.S. Nuclear Regulatory Commission through an R&D grant 31310022M0038 "Trustworthiness of Digital-Twin-based Automation Technology in Nuclear Power Plant Operation."

This manuscript has been authored by Battelle Energy Alliance, LLC under Contract No. DEAC07-05ID14517 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for U.S. Government purposes.

REFERENCES

- [1] Q. Huang *et al*, "A review of the application of artificial intelligence to nuclear reactors: Where we are and what's next," *Heliyon*, 2023.
- [2] Y. Liu *et al*, "SAM-ML: Integrating data-driven closure with nuclear system code SAM for improved modeling capability," *Nucl. Eng. Des.*, vol. 400, pp. 112059, 2022.
- [3] M. R. Oktavian *et al*, "Preliminary development of machine learning-based error correction model for low-fidelity reactor physics simulation," *Ann. Nucl. Energy*, vol. 187, pp. 109788, 2023.
- [4] J. Lee *et al*, "Development of the machine learning-based safety significant factor inference model for diagnosis in autonomous control system," *Ann. Nucl. Energy*, vol. 162, pp. 108443, 2021.
- [5] L. Lin, A. Gurgen and N. Dinh, "Development and assessment of prognosis digital twin in a NAMAC system," *Ann. Nucl. Energy*, vol. 179, pp. 109439, 2022.

- [6] J. C. Reed, "Application of genetic algorithm and deep reinforcement learning for in-core fuel management," *PhD Thesis, Massachusetts Institute of Technology*, 2020.
- [7] L. Lin *et al*, "Development and assessment of a nearly autonomous management and control system for advanced reactors," *Ann. Nucl. Energy*, vol. 150, pp. 107861, 2021.
- [8] L. Lin *et al*, "Digital-twin-based improvements to diagnosis, prognosis, strategy assessment, and discrepancy checking in a nearly autonomous management and control system," *Ann. Nucl. Energy*, vol. 166, pp. 108715, 2022.
- [9] B. N. Hanna *et al*, "Trusting Machine Learning in Nuclear Plant Control: A Reasoning-Based Discrepancy Checker," *12th Nuclear Plant Instrumentation Control and Human-Machine Interface, Virtual Meeting Technologies*, pp. 1-13, 2021.
- [10] E. Kıcıman *et al*, "Causal reasoning and large language models: Opening a new frontier for causality," *arXiv Preprint arXiv:2305.00050*, 2023.
- [11] P. Athe and N. Dinh, "Testing generative pre-trained transformer for knowledge abstraction and reasoning for nuclear reactor design and safety applications," in *Probabilistic Safety Assessment and Management 2023 Topical Conference*, October 23-25, 2023.
- [12] P. Lewis *et al*, "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *34th Conference on Neural Information Processing Systems (NeurIPS 2020)*, 2020, pp. 9459-9474.
- [13] A. Vaswani *et al*, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.
- [14] OpenAI, "GPT-4 Technical Report," *arXiv:2303.08774*, 2023.
- [15] N. Jayaseelan, "LLaMA 2: The New Open Source Language Model," *Meta AI*, 2023.
- [16] S. Yang *et al*, "Foundation models for decision making: Problems, methods, and opportunities," *arXiv Preprint arXiv:2303.04129*, 2023.
- [17] R. Bommasani *et al*, "On the opportunities and risks of foundation models," *arXiv Preprint arXiv:2108.07258*, 2021.
- [18] H. Li *et al*, "A survey on retrieval-augmented text generation," *arXiv Preprint arXiv:2202.01110*, 2022.
- [19] H. Schütze, C. D. Manning and P. Raghavan, *Introduction to Information Retrieval*. 200839.
- [20] S. Wolfram, *What is ChatGPT Doing... and Why does it Work?* Wolfram Media, Incorporated, 2023.
- [21] Harrison Chase, "Langchain Python Library," *Langchain*, 2022.
- [22] OpenAI, "Open AI GPT-4 Chat for DC Implementation in NAMAC," 2024. Available: <https://chat.openai.com/share/90536813-3364-43a0-897d-b0af991f0e24>.
- [23] NIST, "AI Risk Management Framework," National Institute of Standards and Technology, US Department of Commerce, 2023.
- [24] J. Huang and K. C. Chang, "Citation: A key to building responsible and accountable large language models," 2023.
- [25] Microsoft, "Bing AI Search," 2023. Available: <https://www.microsoft.com/en-us/bing?ep=140&form=MA13LX&es=31>.
- [26] B. Wang, S. Wang and Q. Ouyang, "Probabilistic Inference Layer Integration in Mistral LLM for Accurate Information Retrieval," *Research Square Preprint*, 2024.
- [27] V. Freiberger and E. Buchmann, "Fairness Certification for Natural Language Processing and Large Language Models," *arXiv Preprint arXiv:2401.01262*, 2024.