OutageGPT: Multi-Agent Retrieval-Augmented Generation Framework for Power Outage Analysis and Prediction

Charles Alba, *Graduate Student Member, IEEE*, Fei Ding, *Senior Member, IEEE*, Karthik Kumar, Kumar Utkarsh, *Member, IEEE*, Seong Lok Choi, *Member, IEEE*, and Benjamin Kroposki, *Fellow, IEEE*
{charles.alba, fei.ding, karthik.kumar, utkarsh.kumar, seong.choi, benjamin.kroposki}@nlr.gov; alba@wustl.edu

***Abstract*—Power system outage prediction models remain limited by data fragmentation, interpretability challenges, and operational deployment difficulties. With the rise of large language models (LLMs), we explore their potential to assist utilities in outage analysis and prediction—specifically through retrieval-augmented generation (RAG), which augments the model's context with retrieved historical records. "OutageGPT", a multi-agent RAG framework, is introduced. It integrates a mixture-of-experts architecture and advanced prompting strategies to address diverse outage-related queries. In retrospective 2021 severe-weather test cases, OutageGPT outperformed a state-of-the-art open-source LLM queried directly without retrieval, with ground-truth values more frequently within its predicted ranges due to contextual grounding from historical data. While it posses some limitations, like underestimating extreme events and producing broad prediction intervals, it demonstrates promise while highlighting future needs, including multimodal integration and domain-specific foundation models for energy systems.**

***Index Terms*—Large Language Models, Retrieval-Augmented Generation, Power Grid, Outage Analysis, Outage Prediction**

I. Introduction

The frequency and intensity of power system outages are rising, driven by extreme weather events, aging infrastructure, and increasing cyber threats. These outages affect grid reliability and resilience, disrupting electricity users' critical energy services. Consequently, analyzing historical outage information and preparing proactively for future disruptions has become imperative for utilities, regulators, and researchers.

In the past decade, many research efforts have been done to model the relation between power outages and weather and asset conditions (e.g., wind speed, precipitation, vegetation, equipment age) [1], [2]. Besides, a large body of research work uses machine learning methods to predict power outages. For instance, graph convolutional networks were introduced in [1] to advance outage prediction models. A generative adversarial network was used in [2] to augment existing datasets and then enable a two-step classification approach for outage prediction. However, despite extensive efforts to improve outage modeling and prediction accuracy, several key challenges have received limited attention. First, outage-related information is scattered across structured databases, unstructured text, images, and proprietary vendor systems. Integrating these sources requires significant manual data engineering. Second, outages, especially those from extreme events, are rare and inconsistently labeled, limiting the effectiveness of purely supervised learning models. Moreover, existing models suffer from limited interpretability, wherein they lack clear, human-understandable reasoning, making it difficult for utility operators to trust and act upon them.

Large language models (LLMs) offer a powerful new paradigm for addressing these limitations. Unlike traditional artificial intelligence (AI) models that specialize in structured numerical data, LLMs are capable of understanding, synthesizing, and reasoning across diverse and unstructured data types. Their ability to extract entities, infer relationships, and generate coherent summaries or recommendations makes them well-suited to outage-related applications where data variety and ambiguity are dominant challenges [3], [4].

When integrated with retrieval-augmented generation (RAG) pipelines and domain-specific knowledge bases, LLMs can reason over utility procedures, historical outage patterns, and geospatial data [5]. This paper proposes a multi-agent RAG framework that leverages the reasoning, comprehension, and orchestration capabilities of LLMs to unify disparate outage data sources, enhance predictive performance, and deliver interpretable, actionable insights. As one of the first works exploring the application of LLMs for power outage analysis and prediction, this paper presents our initial findings on the promise of LLMs. Additional research is underway to further improve the model performance.

II. Methods

A. Multi-Agent RAG Model Design

Unlike ordinary chatbots of pretrained models, which directly answer the question posed by a user, RAGs works by retrieving data relevant to the user's question from a provided external database [5]. Most RAGs traditionally operate as stand-alone models, typically designed for specific applications. In such systems, a basic retriever (e.g., BM25 or cosine similarity) is used to identify data most relevant to the user's query, which is then fed into a language model

This work was authored by the National Laboratory of the Rockies for the U.S. Department of Energy (DOE), operated under Contract No. DE-AC36-08GO28308. Funding provided by U.S. Department of Energy Office of Critical Minerals and Energy Innovation Solar Energy Technologies Office under Agreement 40385. Fei Ding, Utkarsh Kumar, Karthik Kumar, Seong Choi, and Benjamin Kroposki are with the Power Systems Engineering Center, National Laboratory of the Rockies, Golden, CO, USA (contact: fei.ding@nlr.gov).

as context to generate a response [5]. But these traditional stand-alone approaches are only minimally effective for our use case for several reasons. First, our application is inherently more complex, particularly due to its reliance on tabular data rather than purely textual sources. Second, users may pose a wide range of queries that a traditional stand-alone RAG cannot adequately handle, and these queries can range from simple factual questions to complex analytical or predictive queries requiring metric calculations, contextual assessments, and knowledge recommendations.

To address the aforementioned challenges, we created a multi-agent RAG model, called **OutageGPT**, to address diverse outage-related queries. Fig. 1 shows the overall architecture. Unlike a stand-alone RAG, a multi-agent RAG comprises multiple modules and RAGs, each designed with a specific task. These multiple agents work hand-in-hand or adjacently to accomplish a complex task, which in our case is to answer a wide variety of outage-related user queries, thereby creating a functioning chatbot to assist power grid operators. Within each agent, we leverage advanced prompt engineering to enable our model to reason coherently and generate accurate responses.

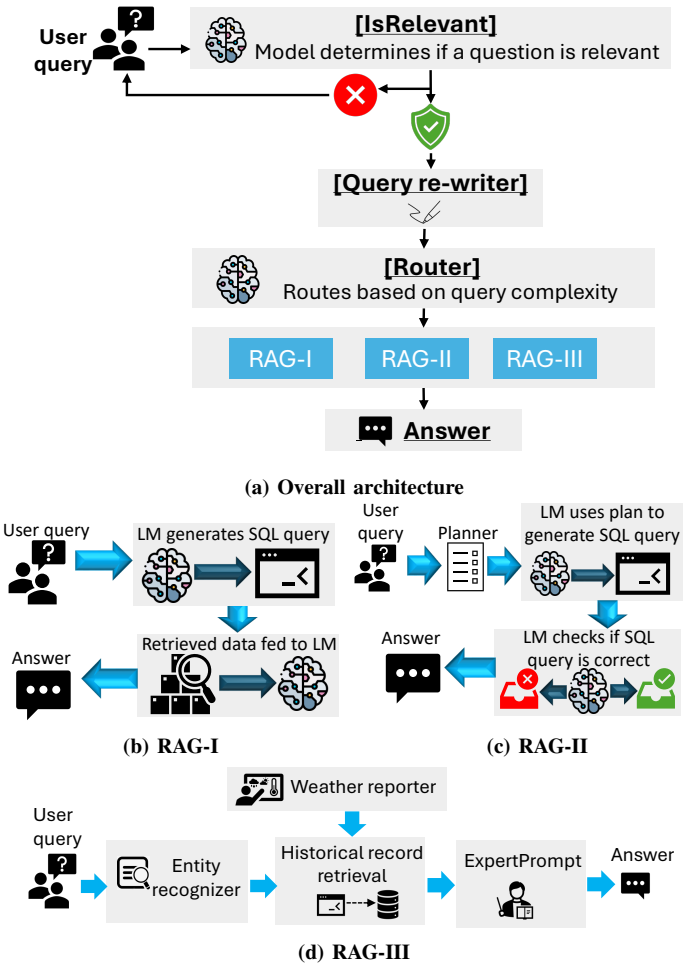


Fig. 1: Architecture of our proposed multi-agent RAG model.

B. Query Preprocessing

To handle the tabular nature that is different from the text-based modality of most LLM applications, our retriever primarily consists of a coder language model [6] that generates an SQL query from user's question [7]. Specifically, OutageGPT uses `wizardcoder:33b` [6] model to generate all SQL queries. For example, if the user asks, "What is the total duration of outages where $>$ 1000 customers were affected in Texas in 2021?", the coder model will convert it to an SQL query "*SELECT SUM(duration) FROM county_outages AND year=2021 AND customers_affected>1000 AND state='Texas'*." The resulting SQL query is then executed to retrieve and filter the relevant data from the historical power outage dataset, which is subsequently provided as context for the generation phase [7].

Queries first undergo a [IsRelevant] module, inspired by [8], where it filters and determines if the the query is outage related or within the scope of the data. This is then followed by a [Query re-writer] module, which aims to rewrite queries with spelling errors or poor grammatical syntax or replaces references with broad events with those of specific dates and entities [9], [10]. By standardizing and clarifying user queries, this step reduces the cognitive and computational load on subsequent RAG modules, which would otherwise attempt to infer or correct such ambiguities during retrieval and reasoning. Both modules leverage the simple yet effective technique of "few-shot learning," where the models learn from a provided set of carefully curated examples. We adopted `Llama3.3:70b` model [11] for this agent.

C. Mixture-of-RAGs

The preprocessed query is then fed into the RAG model tasked with retrieving the relevant data pertaining to the preprocessed query and generating a coherent response. To deal with the fact that users may ask a wide variety of queries with different levels of complexity, we innovate a "mixture-of-RAGs" architecture. Adapted from the "mixture-of-experts" architecture model [12], the mixture-of-RAGs consists of many sub-RAGs—three, in our case—each designed to answer distinct types of queries. This architecture contrasts with having a single, overly large RAG that attempts to handle every query type, which risks producing incorrect or inconsistent results due to being overwhelmed or its inability to adapt its reasoning to different query complexities, resulting in misguided or misinterpreted reasoning surrounding the user query. Rather, having smaller "specialized" RAGs, each highly capable of answering a query type of distinct complexity, could result in a comprehensive yet more reliable model. Upon preprocessing the query per the aforementioned section, the query is passed to a [Router], which assists in determining which RAG is best suited to answer the provided query and routes it accordingly.

a) RAG-I: RAG-I is designed to answer basic user queries, such as outage records of specific regions or aggregated statistics of historical outages (e.g., "*How many outages were in the State of Texas in 2021?*" or "*How long was the outage in Dallas County starting 2020-10-29?*"). To do

so effectively, the system leverages the ReAct (Reasoning + Acting) framework [13], which enables LLMs to interleave logical reasoning with predefined actions rather than directly generating an output. Unlike traditional RAG systems that attempt to produce an answer in a single step—such as returning either an SQL query or a textual response—ReAct allows the model to *think* through intermediate steps and *decide* which actions to take.

In our implementation, three actions are defined for the retriever: `_get_schema`, which retrieves the database schema for interpretation; `_run_sql`, which constructs and executes an SQL query based on the user's request; and `_fmt`, which formats and presents the final output.

For example, when presented with the query "*What is the total duration of outages where more than 1000 customers were affected in Texas in 2021?*", the model may first reason, "*I need to understand how the data is structured*", and invoke the `_get_schema` action. After obtaining the schema, it may reason, "*Now I can construct an SQL query using the relevant columns*", and call the `_run_sql` action. This stepwise reasoning–action process ensures that the model retrieves accurate information while maintaining transparency and interpretability. These retrieved data are then fed into an LLM as context to generate a coherent response.

b) RAG-II: RAG-II is designed to compute complex metrics, such as those defined in [14]. Unlike RAG-I, which is tasked with filtering or aggregating outages from the dataset, these metrics require multistep and careful calculations; hence, instead of using ReAct, we use the "plan-and-solve" [15] prompting approach, where a model first composes a step-by-step plan from the provided query. This plan is then used to guide the coder model in devising the proper SQL query that computes the respective metrics. For instance, if the user asks "*Compute average impact level across Texas during 2021, which is defined as the maximum number of customers that are affected during a power outage, with a power outage defined as periods more than 5% of customers affected*", the model will compose a plan such as:

1) Filter records of Texas in 2021.
2) Filter periods where $> 5\%$ of customers are affected.
3) Group continuous periods as outages.
4) Find the maximum customers affected for each outage.
5) Aggregate average across all outages.

In doing so, the logical and computational reasoning burden is offloaded from the coder model, which primarily excels at code generation rather than abstract or multistep reasoning. By having two separate agents—one dedicated to decomposing the problem into a structured plan and another focused on translating that plan into an executable SQL code—each agent is able to specialize in its respective task, resulting in more reliable and efficient query generation. As errors may naturally arise from the coder model, particularly when handling complex metrics with more detailed plans, the produced SQL and its corresponding plan are fed into a `self-reflect` module, where the model reviews and verifies the SQL output for logical consistency and correctness prior to execution [16]. Specifically, the `self-reflect` module either returns a "PASS" to indicate that no issues detected, or it generates targeted feedback highlighting discrepancies between the plan and the produced SQL code, which is then fed back to the coder model for correction and regeneration of the query [8], [16].

c) RAG-III: Finally, RAG-III is tasked with answering complex queries that assess or predict. These include questions involving retrospective assessments of outages or grid vulnerabilities (e.g., "*Assess and compare the vulnerability of outages between Colorado and California in 2021*"), as well as predictive queries where users may request the model to forecast potential outage scenarios under specified weather conditions for anticipatory outage management (e.g., "*What is the likely outage scenario in Franklin County (FL) given an anticipated hurricane with potential wind speeds of 90 km/h expected between 08-06-2028 and 12-06-2028?*").

Unlike RAG-I and RAG-II, which are designed to provide relatively concise answers, RAG-III is intended to reason in depth and generate detailed, report-like responses grounded in the retrieved contextual data. Consequently, RAG-III employs a set of specialized approaches tailored to produce outputs resembling those of a domain expert in power systems. The process begins with an `Entity Recognizer` agent that identifies key entities within the user query (e.g., "Franklin County (FL)" $\rightarrow$ location; "08-06-2028 and 12-06-2028" $\rightarrow$ dates). These extracted entities assist in SQL generation by constraining the retrieval of relevant historical outage data. The recognized entities are then passed to the coder model, which retrieves the corresponding outage records from the database.

Because the queries that RAG-III handle are often highly correlated with weather patterns, we include a dedicated `Weather Reporter` agent tasked with generating weather summaries spanning the temporal and geographical scope of each retrieved historical outage record. These summaries may include, but are not limited to, precipitation, wind speed, snow or ice accumulation, and notable weather events (e.g., hurricanes occurring during or in the vicinity of the outage). The resulting weather reports will later assist the generator model in identifying and reasoning about patterns relevant to the user query where necessitated, as demonstrated in the example above. In our current implementation, the `Weather Reporter` relies on the pretrained knowledge embedded within existing LLMs, which we have verified to be sufficiently accurate. Nonetheless, due to the agentic nature of our architecture, this component can be readily replaced or augmented with an agent connected to external data sources (e.g., NOAA) to obtain verified meteorological records corresponding to the spatial and temporal frame of each outage.

With the historical outage records now augmented by the corresponding weather data, this combined information is fed into the generator. Unlike the preceding RAG-I and RAG-II, the key innovation of RAG-III lies within this generation stage. Specifically, the retrieved data serve as contextual input to an `ExpertPrompt` model, which guides the language model to reason as a domain expert when performing assessments or predictions [17]. In summary, the expert prompting strategy conditions a model to think, reason, and produce responses analogous to those of a domain-specific expert—in this case, a power systems expert. This is achieved by constructing

an "expert identity" through in-context demonstrations using, which is then augmented into the system of the generator model, effectively conditioning it to emulate the reasoning and response patterns of a power systems expert. Consequently, the model produces expert-like recommendations, assessments, and predictions, enabling it to handle complex analytical and predictive queries with greater reliability.

III. Case Studies

A. Dataset and Model Setup

In this paper, the open-source Environment for Analysis of Geo-Located Energy Information (EAGLE-I) dataset [14] is used to develop and validate the proposed multi-agent RAG model. EAGLE-I records eight years of county-level power outage information from 2014 to 2022 at 15-minute intervals, for 3044 of 3226 U.S. counties and county equivalents. Although we use EAGLE-I data to develop the RAG model, the pipeline of using LLM to retrieve historical information and use the information to analyze and predict power outages is applicable to any dataset.

B. Results on Basic Power Outage Analysis

In addition to developing the RAG model, we also developed a chatbot that enables Q&A. Fig. 2 shows a screenshot of the functioning OutageGPT chatbot answering three different types of questions: One simply retrieves historical outage data using RAG-I, one relies on RAG-II to do some analysis, and one responds to an irrelevant question. Results prove that OutageGPT handles different user queries effectively.

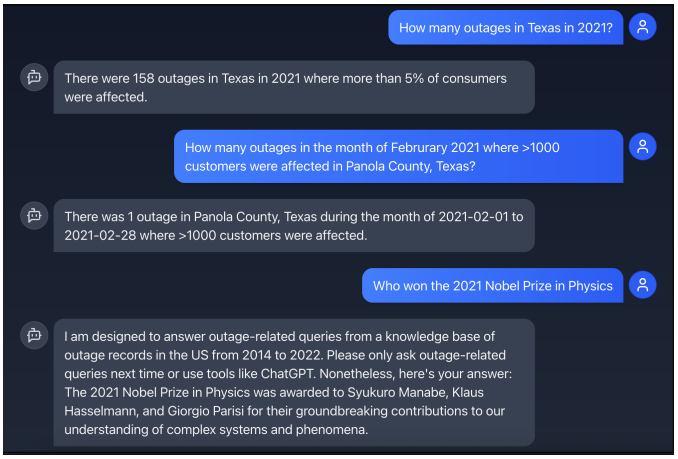


Fig. 2: OutageGPT chatbot answering different user queries.

C. Results on Advanced Power Outage Analysis

To validate the performance of OutageGPT on conducting advanced power outage analysis, we studied two cases. In the first case, we asked OutageGPT to analyze which state in the United State should take priority for maintenance based on historical outage records. Fig. 3 shows the output from the full RAG-III model compared to the answer generated without including the `ExpertPrompt` agent—the key innovation of RAG-III. It clearly shows that the addition of the expert

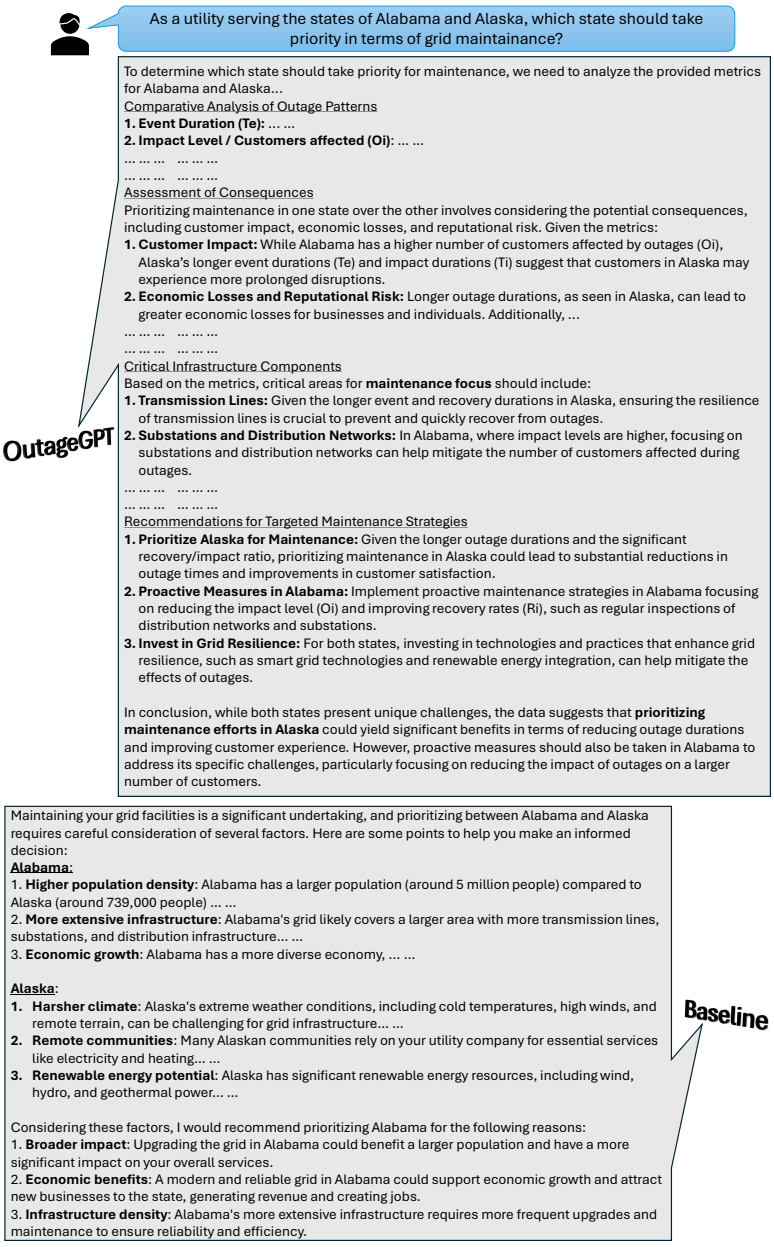


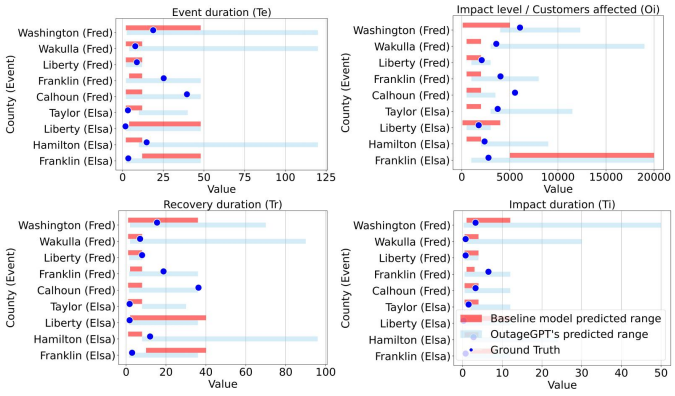
Fig. 3: The comparison of advanced power outage analysis between full RAG-III (top) and the one without expert prompt (bottom).

prompt technique can help construct the answer to provide more domain knowledge.

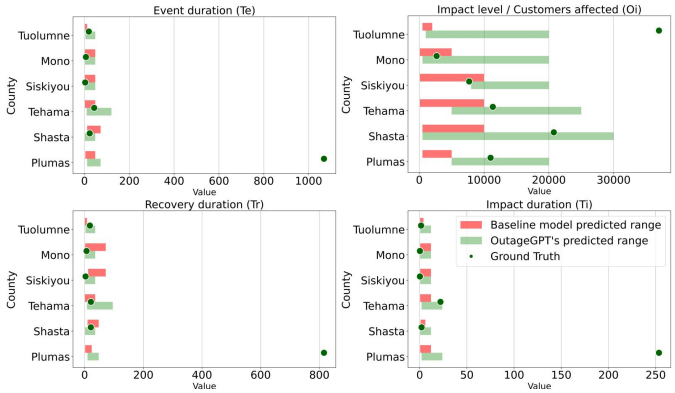
Then, we designed the second test case to evaluate the effectiveness of OutageGPT on predicting outages. Specifically, we provided OutageGPT with historical outage records from 2014 to 2020, and then we asked OutageGPT to predict outages in 2021 by providing real weather information extracted from the following four actual events.

1) Californian Wildfires, specifically those listed in [18].
2) Florida's outages on Hurricane or Thunderstorm season.
3) New York's 2021–2022 winter storm, including [19].
4) The unprecedented 2021 Texas power outage [20].

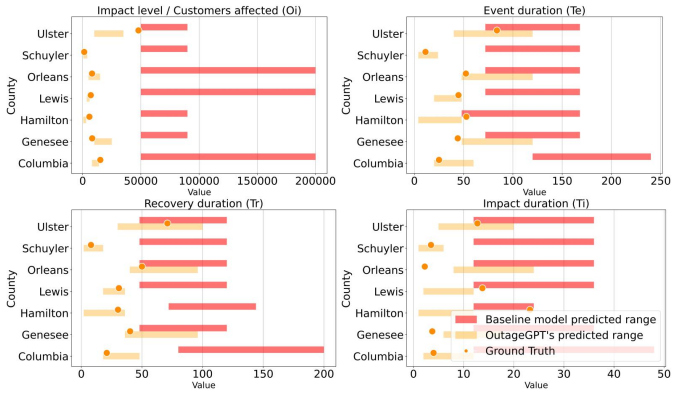
To evaluate the effectiveness of the proposed RAG model, as shown in Fig. 4, the four outage impact metrics men-



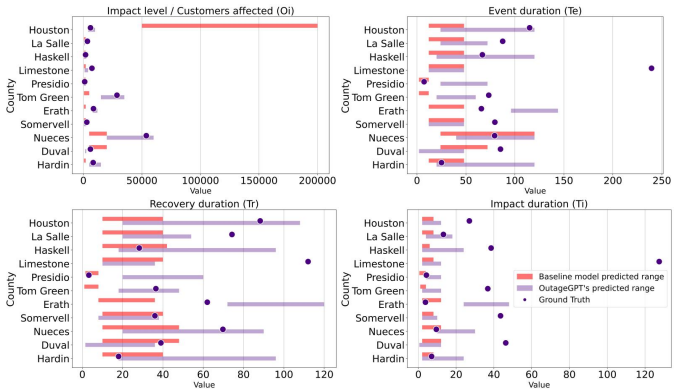
(a) Florida hurricanes



(b) California wildfires



(c) New York winter storms



(d) Texas outages

Fig. 4: Results of our retrospective test cases. Note that Texas was truncated due to the sheer number of counties affected.

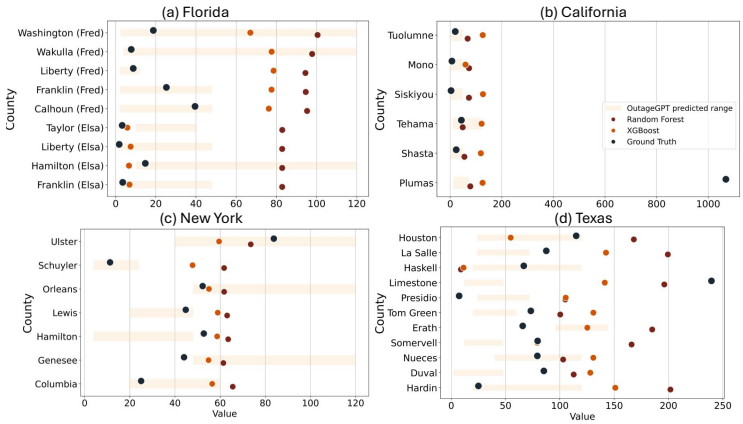


Fig. 5: Comparison between OutageGPT and other data-driven methods.

tioned in [14] are quantified to measure the accuracy. We compared the results generated from OutageGPT with both the actual data (ground-truth) and the results generated from the naive use of LLMs (`Llama 3.3:70B-instruct` in the test case) without adding RAG (baseline). Both OutageGPT and the baseline generate outage prediction results in the form of an estimated range. Overall, OutageGPT consistently outperforms the baseline model, with the ground-truth values more frequently falling within its predicted ranges. In addition, Fig. 5 compares OutageGPT against XGBoost and Random Forest, with weather, precipitation, and county-level customer counts as features. To avoid potential data leakage from customer-related outcomes (e.g., customers affected), we opted to predict event duration. While XGBoost performs well in several cases, numerical models can be inconsistent and deviate from observed outcomes. In contrast, OutageGPT's predicted ranges more consistently capture the ground truth and provide text-based interpretations and recommendations unavailable to numerical methods.

IV. Discussion and Conclusion

This paper presents a multi-agent RAG framework to address outage-related queries of relevance to utilities. Since the work presented in this paper is still at its early stage, we recognize several limitations in the current results:

1) OutageGPT has a tendency to underestimate unprecedented events (Fig. 4(d)). Our hypothesis is that these extreme cases are not observed in the past, so the retrieved historical knowledge cannot be leveraged effectively.
2) OutageGPT provides the predictions with wide ranges. In some cases, it repositions–as opposed to realigning or calibrating–its thresholds based on the historical context.

However, with the promising results presented in both Fig. 4 and Fig. 5, we can conclude that integrating the proposed multi-agent RAG framework with a pretrained generic LLM can provide a promising alternate to predicting power outages. OutageGPT can serve as a tool to assist utilities in anticipating outage impacts and enabling proactive planning and communication. Besides, our work lays the foundation for further work, including (1) exploring complementing LLMs

with other forms of AI models, physical models, and other data sources to produce predictive capabilities [3], [4] and (2) strengthening the inherent foundations of LLMs themselves to better adapt to utility-specific corpora, which includes developing an energy foundation model [21], [22].

REFERENCES

[1] M. Azizi, X. Zhang, F. Yang, K. Udeh, J. Zhao, and E. Anagnostou, "Advancing outage prediction modeling: Incorporating circuit spatial relationships with graph neural networks and lidar-derived tree risk," *Reliability Engineering & System Safety*, vol. 265, January 2026.

[2] R. Rastgoo, N. Amjady, S. Islam, I. Kamwa, and S. M. Muyeen, "Extreme outage prediction in power systems using a new deep generative informer model," *International Journal of Electrical Power & Energy Systems*, vol. 167, June 2025.

[3] S. L. Choi, R. Jain, P. Emami, K. Wadsack, F. Ding, H. Sun, K. Gruchalla, J. Hong, H. Zhang, X. Zhu *et al.*, "egridgpt: Trustworthy ai in the control room," National Renewable Energy Laboratory (NREL), Golden, CO (United States), Tech. Rep., 2024.

[4] S. L. Choi, R. Jain, C. Feng, P. Emami, H. Zhang, J. Hong, T. Kim, S. Park, F. Ding, M. Baggu *et al.*, "Generative ai for power grid operations," National Renewable Energy Laboratory (NREL), Golden, CO (United States), Tech. Rep., 2024.

[5] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel *et al.*, "Retrieval-augmented generation for knowledge-intensive nlp tasks," *Advances in neural information processing systems*, vol. 33, pp. 9459–9474, 2020.

[6] Z. Luo, C. Xu, P. Zhao, Q. Sun, X. Geng, W. Hu, C. Tao, J. Ma, Q. Lin, and D. Jiang, "Wizardcoder: Empowering code large language models with evol-instruct," in *The Twelfth International Conference on Learning Representations*, 2023.

[7] A. Mohammadjafari, A. S. Maida, and R. Gottumukkala, "From natural language to sql: Review of llm-based text-to-sql systems," *arXiv preprint arXiv:2410.01066*, 2024.

[8] A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi, "Self-rag: Self-reflective retrieval augmented generation," in *The Twelfth International Conference on Learning Representations*, 2024.

[9] X. Ma, Y. Gong, P. He, H. Zhao, and N. Duan, "Query rewriting in retrieval-augmented large language models," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023, pp. 5303–5315.

[10] A. Anand, V. Setty, A. Anand *et al.*, "Context aware query rewriting for text rankers using llm," *arXiv preprint arXiv:2308.16753*, 2023.

[11] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan *et al.*, "The llama 3 herd of models," Tech. Rep., 2024.

[12] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural computation*, vol. 3, no. 1, pp. 79–87, 1991.

[13] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. R. Narasimhan, and Y. Cao, "React: Synergizing reasoning and acting in language models," in *The eleventh international conference on learning representations*, 2022.

[14] M. Abdelmalak, J. Cox, S. Ericson, E. Hotchkiss, and M. Benidris, "Quantitative resilience-based assessment framework using eagle-i power outage data," *IEEE Access*, vol. 11, pp. 7682–7697, 2023.

[15] L. Wang, W. Xu, Y. Lan, Z. Hu, Y. Lan, R. K.-W. Lee, and E.-P. Lim, "Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, 2023, pp. 2609–2634.

[16] A. Madaan, N. Tandon, P. Gupta, S. Hallinan, L. Gao, S. Wiegreffe, U. Alon, N. Dziri, S. Prabhumoye, Y. Yang *et al.*, "Self-refine: Iterative refinement with self-feedback," *Advances in Neural Information Processing Systems*, vol. 36, pp. 46 534–46 594, 2023.

[17] B. Xu, A. Yang, J. Lin, Q. Wang, C. Zhou, Y. Zhang, and Z. Mao, "Expertprompting: Instructing large language models to be distinguished experts," *arXiv preprint arXiv:2305.14688*, 2023.

[18] Wikipedia contributors, "2021 california wildfires," https://en.wikipedia.org/wiki/2021_California_wildfires, 2025, accessed: 2025-09-10.

[19] D. R. Novak, "Big stories from the 2022-2023 winter season," in *104th Annual AMS Meeting 2024*, vol. 104, 2024, p. 428866.

[20] N. M. Flores, H. McBrien, V. Do, M. V. Kiang, J. Schlegelmilch, and J. A. Casey, "The 2021 texas power crisis: distribution, duration, and disparities," *Journal of exposure science & environmental epidemiology*, vol. 33, no. 1, pp. 21–31, 2023.

[21] H. F. Hamann, B. Gjorgiev, T. Brunschwiler, L. S. Martins, A. Puech, A. Varbella, J. Weiss, J. Bernabe-Moreno, A. B. Massé, S. L. Choi *et al.*, "Foundation models for the electric power grid," *Joule*, vol. 8, no. 12, pp. 3245–3258, 2024.

[22] C. Alba, "Benchmarking small language models in the renewable energy domain," in *Proceedings of the 18th IEEE/ACM International Conference on Utility and Cloud Computing*, 2025, pp. 1–2.