Poisoning the RAG : misleading generative AI applications

Short Paper

Abstract

With the commercial availability of large transformers based language models (LM), they are increasing being used in many SaaS applications. Among these, the most popular use-cases are summarization, Q&A and chat based customer support. Due to high computation cost of training such models, most enterprises use *off-the-shelf* LMs (either open or proprietary) which are trained on text crawled from the Internet. This study puts forward a hypothesis that given that the data which is used to train such model is unverified, such models are susceptible to subterfuge. It is possible for a determined actor to systematically poison data, to ensure that the LM responds with a certain output for a given input. This has been demonstrated by published works which have been shown to be successful for models with both open and closed weights. The implication for information systems using such Gen AI capabilities is that they can be misled to become vehicles of surrogate advertisement or even misrepresent facts. This becomes even more pertinent for the organization as these applications are mostly publicly accessible and are used to automate customer related decisions. Any such attack would be public knowledge and can be detrimental for the orgainization's public image.

Keywords: *large language model, RAG, privacy*

Introduction

The International Monetary Fund estimates that approximately forty percent of the global workforce has been exposed to machine intelligence (AI) (Cazzaniga et al. 2024). Such an exposure would either be direct, as users of AI enabled workspace tools or as consumers of such services. This increase has been accelerated post the use of transformers (Vaswani 2017) which improved the performance of deep learning models on benchmarks tasks. A very popular pattern, Retrieval-augmented generation (RAG) (Lewis et al. 2020) has allowed embedding of reasoning capabilities within SaaS applications. It is estimated that by 2030, seventy percent of all all business activity could be powered by this approach. (Durth et al. 2023).

Contribution

This study attempts to look at the most popular pattern of use of language models in the industry with a focus on cyber-security and threat hunting. Specifically,

- This paper describes how Retrieval-augmented generation or RAG works.
- It details the typical use-cases in the industry.
- The kind of attacks these use-cases are susceptible to.
- Finally, best practices and potential solutions which can be used in Gen AI applications to reduce the possibility of such attacks.

The following sections details the concept of Retrieval-augmented generation, Following with a description of the kinds of attacks and ending with possible countermeasures.

Background

Retrieval-augmented generation

RAG is a very popular pattern of using LM in production. It essentially consists of three stages as observed in (Lewis et al. 2020) namely,

Embedding

This refers to the model used for generating embedding or numeric representation the input. The input itself could be multi-modal. e.g. text, images, IMU data, audio etc.

Retrieving

The retriever is the module which finds the most similar items in the knowledge base for a given input.

Generating

Using the retrieved output and the user query, the generator summarizes the input.

For the purpose of this study, the above described components can be considered as a black boxes. Mostly, they consists of a deep learning model with varied architecture. A detailed study can be found in (Weng 2020).

Use cases

RAG is a very popular pattern of using machine intelligence to automate business processes. Below are few of the most common use-cases that are observed in modern IT systems.

Question Answering / fact verification

Given an user query as a free form text, the system answers it using a knowledge base. The knowledege base could consist of multiple data sources with different file formats.

Summarization

For a give input (e.g.a book, or transcript of a meeting), a short summary is generated optionally confirming to a specific style of writing.

Dialogue Systems

A LM based agent interacts with application users in the form of multi-turn conversation. It attempts to provide information, answer queries and makes business decisions e.g. deciding if a refund is required or not.

Concerns

This ubiquitous pattern has been observed to have certain problems (Wu et al. 2024) which can be categorized in the following heads namely,

Relevance versus semantic similarity

The retriever use a mathematical operation on the numeric representation of the input (mostly a high dimension vector). The mathematical operation fails to measure relevance (the importance of a context for answering the question). Instead, it looks at similarity which measures if question and context are regarding common topics. As a substitute of relevance, similarity especially using off-the-shelf retrievers is *generally correct*. There could be edge cases where the measure may fail to retrive relevant documents. As a stopgap, these approach are occasionally refined using a keyword based approach like BM25 and TF-IDF, however they too fail to completely capture relevance. The reason lies in the training approach which is used to create such models. The training metrics look at accuracy over a large corpus of training examples which ensure that the model output *generally* agrees with the training data.

Noisy input corpus

Most real-world applications have a knowledege base containing tables, numbers or charts in various file formats. The lack of a common representation requires that all such file formats be converted to text (mostly UTF-8 encoding). This conversion is not perfect and often leads to presence of extraneous special characters. Further, the LM has a limited vocabulary and therefore the encoding performed by the tokenizer of such text is not perfect.

Limited reasoning capabilities of the LM

Finally, LMs are not first-class reasoners. Their performance depends on how articulated the context is. The problem is exacerbated when such pipeline runs for low resources languages, for which the LM never got the opportunity to capture the nuances of the language.

Finally, the corpus is a snapshot of the world at a given point in time. To keep it updated, additional context has to be provided using traditional search engines.

Possible attacks on RAG

Poisoning the knowledge base

Across different studies, LMs have been observed to be susceptible to misinformation. This is because they lack agency and when presented with untrue information (in form of context), they tend to follow that lead. (Qian et al. 2024) observed that LLM can be easily fooled by asking leading questions. The training process does not impart an explicit ability to reason, unless the inference pipeline explicitly attempts to break down the task as a set of sub tasks and reflects on it as is done by OpenAI for openai-o1-preview .

(OWASP n.d.) describes the most popular types of attacks currently known to happen against LMs. These approaches include methods which roughly can be categorized in approaches which either attempt subversion during the training process or during the inference process. The attack described in this paper is a combination of both, where based on the behaviour of the trained model, certain text sequences are generated which are further inserted in the knowledege base of this RAG pipeline. The insertion could be an active one, where attacker purposely modifies the corpus or could a done using a passive approach, where the sequence could be added as a discussion or post on prominent websites like Reddit or StackOverflow. (Carlini et al. 2024) reported that 6.5 percent of Wikipedia can be edited, especially for topics which are not commonly referred to. Hence such an attack is quite possible, especially due to the practice of periodic refreshes performed by crawler, where such a context is expected to be crawled, scanned and indexed by the system. This text is used either as a knowledege base or as a pretraining database for LMs. (Chaudhari et al. 2024) has described detailed steps that could be used to execute such an attack.

It is pertinent to be noted that such attacks cannot be carried on entities which have large presence on the Internet as the malicious context would be retrieved in addition to other relevant text items.

Prompt injection

Another approach to use subterfuge to confuse a language model is by prompting the model to change the internal stage of the LM. This allows attacker to override the default behaviour just by conducting more turns of conversations.

Others

Other approaches include denial of service, passage ex-filtration or harmful behaviour which use combination of above techniques.

Reasons why such attacks are possible

As observed by (Zou et al. 2024), the RAG framework introduces a new and practical attack surface. If an attacker is able to insert malicious text in the input corpus then the LM outputs specific sentences for a given input. (Zou et al. 2024) showed instances where a sentence (generated through and adversarial text) was retrieved for a given question. However, the text did not make semantic sense but was a perfectly legitimate string which can easily bypass OWASP based guardrails.

Model trained on data collected from the wild

The practice of training the LM using next token prediction is an easy way of exposing the model to real world interactions without the need of labelled data. Text,images or audio data is used because of the ease of their production and their availability. The concern with the approach is that none of the data is verified for bias, correctness or relevance. Even datasets which are used to benchmark LM are part of the training corpus. e.g the canary strings from BigBench were found to be reproducible by models trained by Anthropic. (optimalsolver n.d.).

LM gives equal importance to all top K contexts

The retriever forms the second part of the puzzle. Most retrievers, rely on an embedding model which itself is a deep learning model trained on open source data. The unsupervised learning of the embedder does not optimize for a specific kind of similarity. e.g. is the word Apple related to the word fruit more or the word smartphone. This problem trickles down to the retriever as well, where setting a hard cut off on similarity ensures that the top K items generally refer to the topics discussed in the question. The items retriever by the retriever are sorted in decreasing order of the similarity metric and using a hard cutoff (mostly a count), a top-K selection is made. However all the top K items are added to the context,before asking the question. It is generally excepted that LM tend to focus on text at the starting of the context, but this focus is too less to differentiate between the different selected context. Ideally, they should have been weighted by the similarity metric to bias the LM towards their relative importance.

Similarity is a poor metric

Sometimes for answering a question, mere similarity is not useful. e.g. as observed in (Berglund et al. 2023) LM suffer from a reversal curse. Embedding similarity optimizes for the presence of tokens and not semantics.

Possible remediation

The solutions for the above cited problems have been classified by (Neel Jain 2023) in three main groups namely, Detection of an malicious text in the knowledge base, Pre-processing Defenses and adversarial training.

Use of guardrails

One way to prevent undesirable input and outputs from the pipeline is by using LMs based guardrails which score the content. The measured components could look at attributes like hate, sexuality and bias. However, in the context of data poisoning there measures will not be very useful and there is no consistent pattern for Toxicity, Language Polarity or Hurtful content. Benchmarks like real-toxicity-prompts, HONEST and BOLD can be used to training models to detect such content. This however, cannot be used to detect incorrect facts in the knowledge base.

Use a different patterns of RAG

Another approach to reduce the possibility of such attacks would be by using methods like ReAct (Yao et al. 2022) or self Reflection which break down the reasoning tasks in steps while evaluating the output of each such steps before proceeding.

Fine-tune embedding for custom similarity metric

As similarity search is one of the reasons for such problems, the retriever can use a custom embedding model which is more in-tune with the use case.

Generate answer using K fold validation from the retrieved context

Finally, getting the LM to generate multiple answers from the same question is one more way by which such the possibility of such attacks can be reduced.

Conclusion

The study looked at the popularly used paradigm of RAG and detailed ways it can be compromised. The suggested solutions are part of a continuous effort which developers of information systems have to make to ensure that Gen AI based IT initiatives have adequate guardrails. Compared to other technologies, Gen AI is a newer approach with little or limited institutional memory of running and maintaining such applications. The arms race associated with the need to constantly add new AI features to one's products should not be a reason for adding such a vulnerability in one's application. The conclusion from the numerous focus group discussions conducted for this study is that enterprises (both big and small) are sacrificing due-diligence for an *AI-first* approach.

References

Berglund, L., Tong, M., Kaufmann, M., Balesni, M., Stickland, A. C., Korbak, T., and Evans, O. 2023. "The reversal curse: Llms trained on" a is b" fail to learn" b is a"," *arXiv preprint arXiv:2309.12288* ().

Carlini, N., Jagielski, M., Choquette-Choo, C., Paleka, D., Pearce, W., Anderson, H., Terzis, A., Thomas, K., and Tramèr, F. 2024. "Poisoning Web-Scale Training Datasets is Practical," in: *2024 IEEE Symposium on Security and Privacy (SP),*

Cazzaniga, M., Jaumotte, M. F., Li, L., Melina, M. G., Panton, A. J., Pizzinelli, C., Rockall, E. J., and Tavares, M. M. M. 2024. *Gen-ai: Artificial intelligence and the future of work,* International Monetary Fund.

Chaudhari, H., Severi, G., Abascal, J., Jagielski, M., Choquette-Choo, C. A., Nasr, M., Nita-Rotaru, C., and Oprea, A. 2024. "Phantom: General Trigger Attacks on Retrieval Augmented Language Generation," *arXiv preprint arXiv:2405.20485* ().

Durth, S., Hancock, B., Maor, D., and Sukharevsky, A. 2023. "The organization of the future: Enabled by gen AI, driven by people," *Hg. v. McKinsey & Company* ().

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., et al. 2020. "Retrieval-augmented generation for knowledge-intensive nlp tasks," *Advances in Neural Information Processing Systems* (33), pp. 9459–9474.

Neel Jain, A. S. 2023. *Baseline Defenses for Adversarial Attacks Against Aligned Language Models.*

optimalsolver. *Claude 3.5 Sonnet Reproduces BIG-Bench Canary String.* https://news.ycombinator.com/item?id=40911350. Accessed: 2024-05-01.

OWASP. *OWASP Top 10 for Large Language Model Applications.* https://owasp.org/www-project-top-10-for-large-language-model-applications/. Accessed: 2024-05-01.

Qian, Y., Zhang, H., Yang, Y., and Gan, Z. 2024. "How easy is it to fool your multimodal llms? an empirical analysis on deceptive prompts," *arXiv preprint arXiv:2402.13220* ().

Vaswani, A. 2017. "Attention is all you need," *Advances in Neural Information Processing Systems* ().

Weng, L. 2020. "The Transformer Family," *lilianweng.github.io* () 2020.

Wu, S., Xiong, Y., Cui, Y., Wu, H., Chen, C., Yuan, Y., Huang, L., Liu, X., Kuo, T.-W., Guan, N., et al. 2024. "Retrieval-Augmented Generation for Natural Language Processing: A Survey," *arXiv preprint arXiv:2407.13193* ().

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., and Cao, Y. 2022. "React: Synergizing reasoning and acting in language models," *arXiv preprint arXiv:2210.03629* ().

Zou, W., Geng, R., Wang, B., and Jia, J. 2024. "Poisonedrag: Knowledge poisoning attacks to retrieval-augmented generation of large language models," *arXiv preprint arXiv:2402.07867* ().