Exercise caution when building off LLMs

Large Language Models are an exciting technology, but our understanding of them is still 'in beta'.

David C

Since the release of ChatGPT in late 2022, Large Language Models (LLMs) have attracted global interest and curiosity. Whilst initially this saw unprecedented numbers of user signups to ChatGPT[1], in recent months we've seen products and services built with LLM integrations for both internal and customer use. Organisations in all sectors report they are investigating building integrations with LLMs into their services or businesses.

As a rapidly developing field, even paid-for commercial access to LLMs changes rapidly. With models being constantly updated in an uncertain market, a startup offering a service today might not exist in 2 years' time. So if you're an organisation building services that use LLM APIs, you need to account for the fact that models might change behind the API you're using (breaking existing prompts), or that a key part of your integrations might cease to exist.

In addition to the risks of working in a very rapidly changing and evolving market, LLMs occupy an interesting blind spot in our understanding. Only a few years ago, when asked to think of machine learning (ML) and artificial general intelligence (AGI), people could understand the difference. ML was perceived as *'good at things like classifying whether an image had a cat in it'*. AGI as a concept didn't really exist yet (but whatever it was, *'we'd know it when we saw it because it would act like us'* and then possibly lock the pod bay doors).

So amongst the understandable excitement around LLMs, the global tech community still doesn't yet fully understand LLM's capabilities, weaknesses, and (crucially) vulnerabilities. Whilst there are several LLM APIs already on the market, you could say our *understanding* of LLMs is still 'in beta', albeit with a lot of ongoing global research helping to fill in the gaps.

The challenge with LLMs is that, although fundamentally still ML, LLMs (having being trained on increasingly vast amounts of data) now show some signs of more general AI capabilities. Creators of LLMs and academia are still trying to understand exactly how this happens and it has been commented that it's more accurate to say that we 'grew' LLMs rather than 'created' them. It may be indeed more useful to think of LLMs as a third entity that we don't yet fully understand, rather than trying to apply our understanding of ML or AGI.

Researchers and vendors have already found some concerning issues. Research is suggesting that an LLM inherently cannot distinguish between an instruction and data provided to help complete the instruction. In one example, the prompt used to create an organisation's LLM-powered chatbot (with appropriate coaxing from a hostile user) was subverted to cause the chatbot to state upsetting or embarrassing things, which then quickly appeared on social media. Whilst the above represents only a small reputational risk, it's easy to imagine a more dangerous scenario. Consider a bank that deploys an ‘LLM assistant' for account holders to ask questions, or give instructions about their finances. An attacker might be able send a user a transaction request, with the transaction reference hiding a prompt injection attack on the LLM. When the user asks the chatbot “am I spending more this month?” the LLM analyses transactions, encounters the malicious transaction and has the attack reprogram it into sending user’s money to the attacker’s account. Early developers of LLM-integrated products have already observed attempted prompt injection attacks.

As a technical community, we generally understand classical attacks on services and how to solve them. SQL injection is a well-known, and far less-commonly seen issue these days. For testing applications based on LLMs, we may need to apply different techniques (such as social engineering-like approaches) to convince models to disregard their instructions, or find gaps in instructions.

Whilst research is ongoing into prompt injection, it may simply be an inherent issue with LLM technology. Research is also ongoing into possible mitigations, and there are some strategies that can make prompt injection more difficult, but as yet there are no surefire mitigations.

One of the most important approaches is ensuring your organisation is architecting the system and data flows so that you are happy with the 'worst case scenario' of whatever the LLM-powered application is permitted to do. There is also the issue that more vulnerabilities or weaknesses will be discovered in the technologies that we haven't foreseen yet.

The emergence of LLMs is undoubtedly a very exciting time in technology. This new idea has landed - almost completely unexpectedly - and a lot of people and organisations (including the NCSC) want to explore and benefit from it. However, organisations building services that use LLMs need to be careful, in the same way they would be if they were using a product or code library that was in beta. They might not let that product be involved in making transactions on the customer's behalf, and hopefully wouldn't fully trust it yet. Similar caution should apply to LLMs.

Dave C
Tech Director for Platforms Research

[1] Other ChatBots are available

WRITTEN BY

David C
NCSC Technical Director for Platforms Research

PUBLISHED

30 August 2023

WRITTEN FOR

Cyber security professionals