Releasing the Source Code of Large Language Models via Conversational Access

Rosco Hunter
Univeristy of Warwick
rosco.hunter@warwick.ac.uk

Abstract

Large language models (LLMs) have rapidly matured into sophisticated reasoning machines. As a result, the code that determines an LLM's capabilities has since become a powerful asset. Currently, developers are divided on whether to release (i.e., open-source) or restrict access to the inner workings of their most capable models. Proponents of open-sourcing argue that it promotes democratisation and innovation in AI, while critics contend that it facilitates misuse. This paper introduces an intermediate level of model release called Conversational Access (CA). When a model is CA-released, users cannot access its code directly; instead, they engage in a dialogue with the LLM, which can modify its behaviour on their behalf. In this sense, the LLM acts as a helpful intermediary, allowing users to develop and customise a model while retaining the safeguards of more restrictive approaches. The main focus of this paper is further exploring the benefits and drawbacks of CA models, including existing examples like OpenAI's GPT store.

1. Introduction

In recent months, large language models (LLMs) have demonstrated rapid gains in performance and popularity [12]. Their ability to approximate, or even outperform, human responses across a broad spectrum of intellectual tasks has garnered the world's attention. Consequently, their code[1] has become a powerful asset, as this determines what an LLM can and cannot do. The success of these models raises an important question: Should the general public be able to access a model's code, given its potential impact on society?

There is disagreement among AI researchers about whether to expand or limit public access to their most capable LLMs [57]. Some developers favour open-sourcing, publishing every detail of their model so that anyone can utilise, scrutinise, and customise the code. Proponents of this approach argue that it fosters innovation and distributes the economic and epistemic value that their models generate. In contrast, some developers prefer to keep their code private, limiting access to a select group of trusted collaborators. They argue that withholding their code from the public domain is the only way to guarantee that it won't be fine-tuned for malicious applications, from writing fake news to designing malware [3].

In Section 3, we propose an alternative framework for releasing LLMs that we call Conversational Access (CA). Rather than a binary choice between a model's code being fully transparent or inaccessible, it introduces a middle ground. When an LLM is CA-released, users cannot directly access its code but instead engage in a dialogue with the model, which can modify its behaviour on their behalf. The main benefit of this approach is that it (ideally) permits users to tailor a model to their individual needs, without jeopardising the security of the underlying code.

Despite these advantages, Section 3 also examines the unique vulnerabilities of CA models, including their capacity to produce unexpected or uncontrollable harms. In particular, we observe that early CA models, like OpenAI's GPT-store [30], are already prone to leak information or follow harmful (albeit seemingly innocuous) instructions. Moreover, we discuss cases where a CA-released LLM might misinterpret a prompt, resulting in undesirable changes to the model's source code which are hidden from the user. In the worst case, conversational access doesn't prevent harms, it simply hides the underlying causes from external scrutiny.

This paper acknowledges and attempts to address these legitimate concerns. In particular, we accept that the current generation of LLMs are opaque and occasionally harmful. As such, in this early stage of technological progress, we recommend a cautious approach to CA-release, providing users with a relatively limited capacity to customise their models. However, as safeguards improve, users could be granted more freedom to tailor a model to their individual needs. The paper concludes by emphasising this distinc-

[1] For brevity, this paper uses the word 'code' to refer to a broad set of the model's programs and parameters, including its architecture; weights; inference procedure; hyper-parameters; and system prompts.

tive advantage of CA models, uniquely positioning them to match the ever-changing demands for a customisable yet safe LLM ecosystem.

2. Background on Model Access

Broadly speaking, LLMs typically fall into three tiers of accessibility [43]. In this section, we will examine the main benefits and drawbacks of each tier.

- Closed models, like Gopher [35], are entirely inaccessible beyond the lab or company that produced them.
- Queryable models, like Chat-GPT [9], can be prompted, with responses appearing on a gated[2] user interface, but the code and weights are kept from the public domain.
- Open-Source models, like GPT-J [54], make both the code and model weights publicly available.

2.1. Closed Models

Closed models minimise the likelihood of misuse by restricting access to a small group of trusted individuals. This approach has been applied to control other dangerous technologies where legislator's main concern is preventing misuse. For instance, countries that restrict firearm ownership tend to have lower rates of firearm-related homicides [25]. On a more existential scale, nuclear non-proliferation treaties prevent countries without nuclear weapons from acquiring them [36]. The logic is that an LLM, firearm, or nuclear weapon can't be misused by someone who can't even access it.

However, the secrecy of these models blunts their effectiveness and social value. Firstly, there is a loss in terms of the model's quality and objectivity. The development and evaluation of a closed model can only draw from the skills and perspectives of a limited number of individuals. As a result, its design might be naive, and the resulting outputs could exhibit bias [39]. Secondly, there is a loss in terms of the public interest. Being unable to access the model in any form, the general public is excluded from the knowledge that these models offer. This leads to an epistemic inequality in favour of the highly-resourced AI companies.

2.2. Open-Source Models

On the other extreme, open-source models provide unfettered access to the widest possible audience. Proponents argue that open-sourcing facilitates external scrutiny from a global community of researchers, helping to rectify any lingering biases or limitations of the original model [18, 38]. Furthermore, it fosters innovation, as every developer has the opportunity to read, reuse, or build upon the code and weights of an open-source model. This is particularly consequential for the most capable models, as their training requires a scale of computational resources that only a few companies possess.

The momentum and scale of open-source innovation is staggering. Meta's core family of open-source language models, LLaMA [49], became available in February 2023. By September, Meta reported that LLaMA-based models had been downloaded through Hugging Face over 30 million times, with tens of thousands of startups using their latest model [45]. In this same report, Meta claimed that the open-source community had produced variants of LLaMA with "improvements of up to 46% for benchmark datasets".

However, the openness of these models also poses risks, as anyone can access and misuse them. In July 2023, members of the Senate Subcommittee on Privacy, Technology, and the Law expressed their concerns in a letter to Meta's CEO [7]. In this letter, they warned of the potential for LLaMA to facilitate "misuse in spam, fraud, malware, privacy violations, harassment, and other wrongdoing and harms." They followed this by noting that "even in the short time that generative AI tools have been available to the public, they have been dangerously abused."

Not only do open-source models lack robust safeguards to prevent direct misuse, but their weights can even be fine-tuned specifically for malicious applications. Fine-tuning tailors the weights of a model by training it to perform well on a specific task [26]. This is often employed to improve a model's performance for tasks, such as grammar correction or medical diagnostics [50]. However, it can also make a model more dangerous, especially if the training data is harmful [33]. By fine-tuning LLaMA for 100 GPU hours on adversarial data, Gade et al. [22] almost completely removed all safeguards, observing a 97% drop in the number of harmful instructions that the model refused to answer.

Even more concerning, after an open-source model is released, there is no way to retract it from the public domain[3] if it causes excessive harm. They remain perpetually accessible for direct misuse or malicious fine-tuning, with a digital footprint that is impossible to erase. This perpetual accessibility would be catastrophic to society in the event that an open-source model could describe novel toxins [42, 52] or ransomware [56], when appropriately prompted and fine-tuned. While we are yet to witness any truly catastrophic capabilities of open-source LLMs [27], there are already concerning signs [1, 19]. Moreover, the likelihood of such risks increase with each successive generation of language mod-

[2] Gating a model involves putting controls in place (i.e., through an API) so that access to the model can be restricted or removed.

[3] This irreversibly is of particular concern given the inherent difficulty in accurately anticipating the (long-term) impact of a technology before it's released.

els. Given the potential for harm with open-source models and the over-centralisation of closed models, is there a compromise?

2.3. Queryable Models

Straddling the extremes, queryable models mitigate undesirable aspects of both approaches. Unlike closed models, anyone can elicit their knowledge and scrutinise their outputs through prompting. However, unlike open-sourcing, the activity of queryable models can be monitored and constrained. Even in the event of unexpected harm, model access can be temporarily rolled back while patches are applied. Unfortunately, despite these advantages, queryable models still exhibit a number of concerning faults.

Firstly, queryable models don't quite offer the same safety guarantees as closed models. While language models are improving at solely producing benign responses, outputs that are harmful or inaccurate still occur [1, 56]. Given a queryable model, members of the public are bound to actively seek or inadvertently encounter these harmful outputs. Secondly, these models can't be scrutinised to quite the same extent as an open-source approach. Although a queryable model's outputs can be statistically tested for various biases [53], they fall short of providing all of the relevant information that external auditors might require.

However, perhaps the largest deficit of queryable models is that their code can't be modified. In order to prevent misuse, the code that underlies a queryable model is inaccessible and unaltered between prompts. This stifles the public's capacity to improve the model's performance or tailor its code for a specialised task. There is a danger that these safe and thoughtful models will be outpaced by innovative (but perhaps reckless) open-source alternatives whose code can be directly altered by the public [29].

Table 1 summarises this dilemma and the other trade-offs that we've discussed in the paper so far. Closed, queryable, and open-source models each exhibit a significant drawback with respect to one aspect of a model's successful release. In the next section, we introduce a novel solution, called Conversational Access (CA), that attempts to retain the safety of queryable models while allowing for increased user customisation.

3. Conversational Access

So far, this paper has considered a binary choice between a model's code being entirely accessible (open-source) or inaccessible (closed and queryable). We propose CA models as a semi-accessible alternative that tries to mitigate the drawbacks of existing approaches. CA models still possess the standard prompt-response capabilities of a queryable model, but, crucially, they also allow the public to indirectly alter their code through natural language interactions.

	Capacity to Prevent Misuse and Rollback Access	Public Capacity to Scrutinise and Prompt the Model	Public Capacity to Develop and Customise the Model
Closed	✓✓	✗✗	✗✗
Queryable	✓	✓	✗✗
Open-Source	✗✗	✓✓	✓✓

Table 1. This table summarises the key distinctions between existing approaches to model access. The most desirable condition for each metric is denoted by two ticks, and the least by two crosses. Adequate solutions are denoted by one tick.

One possible natural language interface would allow users to submit instructions (i.e., system prompts) which the model reads before it starts future sessions with the user. Alternatively, the user might engage in an open-ended conversation with the model which would modify its own code on their behalf. However, the exact details are unimportant; what matters for CA access is that users can modify the model through natural language instructions but without direct access to its code[4]. Now that we've defined a CA model, in the rest of this section, we explore their capabilities, benefits, and drawbacks.

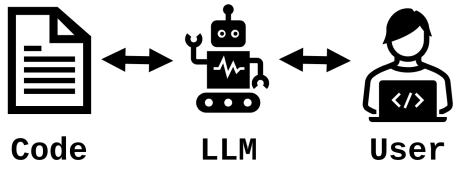


Figure 1. This illustrates how an LLM mediates interactions between the user and the source-code in a CA model.

3.1. Capabilities of a CA Model

How much freedom should a CA model be given to customise itself for the user? On the more conservative side, the model would have no active role in responding to a user's instructions. In this case, the model would simply read the system prompt and documents provided. For example, if a user wanted to design a poker-playing LLM, they would provide the model with a description of its role and documents about the rules of poker.

Slightly more broadly, the model might also be allowed to gather information in real time by freely browsing the internet. For our scenario, this would involve the LLM actively searching for information related to poker, rather than solely

[4] Unlike the unrestricted access of open-source models, natural language interactions are the only way that an individual can access or modify a CA model's code.

relying on the content which the user has explicitly provided. OpenAI's custom GPT [30] is an existing framework with these capabilities. They encourage users to customise their model by "giving it instructions and extra knowledge, and picking what it can do, like searching the web, making images or analyzing data."

These examples demonstrate how a CA model can read contextual information, from user-specified or self-selected sources, to improve its specificity. If the model was also allowed to modify its own weights, then it could digest this information through fine-tuning instead of re-reading the same content before each conversation. OpenAI already provides a platform [31] which enables users to fine-tune their GPTs. However, this still relies on user-selected datasets, as OpenAI doesn't currently allow models to fine-tune themselves on data that they select autonomously.

At its extreme, a CA model could even modify its own architecture or inference procedure. Suppose that a user instructed an LLM to become a poker-player, leveraging the most extensive self-modification capabilities. In response, the model might adapt its architecture [48] for poker-based fine-tuning, automatically improve its prompts [59], or even engage in self-play [16] to boot-strap its capabilities. These abilities sit beyond what the current generation of language models can reliably and autonomously achieve [27].

3.2. Capacity to Prevent Misuse and Rollback Access

While LLM safeguards are improving, even the most advanced models can still produce harmful or inaccurate outputs. Strengthening a model's safeguards can help to reduce these risks, but its functionally impossible to guarantee their safety. Given this inherent unreliability, it might appear dangerous to extend a user's influence over a model beyond simple prompt-response dialogues. Providing users with a broad capacity to modify CA models might increase the risk of them finding ways to circumvent its safeguards. Even with the most conservative modifications, in which users are merely allowed to provide system prompts, there are numerous risks.

A study by Yu et al. [60] demonstrated the ease with which users can extract confidential information from customised GPTs. They found that when prompted, over 97% of models revealed their (potentially sensitive) system prompt or context files. Alongside these risks of information leakage, Tao et al. [46] outlined how custom-GPTs could be specifically created with malicious intent. They described system prompts that could lead custom-GPTs to harm users by stealing their data, spreading misinformation, promoting harmful websites, or co-opting their devices for unlawful activities.

All of these risks result from a conservative level of customisation where system prompts prepend conversations between a model and a user. The potential for undesirable outcomes increases as the model is given a broader capacity to modify its own code. As such, many scholars [8, 17, 24, 58] are concerned that an LLM with an extensive capacity for self-modification could take actions which are unpredictable and uncontrollable. Given all of these risks, it may appear that providing the public with conversational access to a model's code is unacceptably dangerous, with the wisest option being for developers to take closed or queryable approach.

In addressing concerns regarding the misuse of CA models, we emphasise that granting the public any level of access to LLMs inevitably results in some degree of misuse. Nevertheless, 'the genie is out of the bottle' and increasingly capable open-source models with minimal safeguards are now certain to emerge. The real danger is that the most popular models are also harmful. Given this reality, it is naive to indiscriminately prevent users from personalising a model, as this approach risks losing them to more customisable (but potentially reckless) open-source alternatives. Instead, developers must seek to produce models which are both popular and safe. We propose that CA models at least provide a possibility of achieving this lofty, but perhaps necessary, ambition.

In particular, the customisability of CA models can be continually adjusted to ensure that they are both popular and safe. At this early stage of technological progress, we recommend that users are given relatively limited capacity to customise their models. This may simply allow the model to read system prompts. As safeguards become more effective, it might also become tenable for the model to be tailored in more radical ways, for example by letting it fine-tune itself or directly modify its own architecture [48]. Its even possible that CA models themselves could drive the safeguarding improvements that they require to become more customisable.

With CA-release, unlike open-source approaches, modifications to the model are conducted through a user interface, which the company can monitor, control, and restrict[5]. Consequently, every prompt that is found to induce harmful modifications can be flagged and leveraged to improve the company's ability to detect and mitigate malicious activity [46, 55]. Specifically, flagged users can be barred from the platform and their prompts could be used to train a classifier for identifying (and blocking) other malicious actors. By leveraging this wealth of data, CA providers might outpace their competitors at developing broadly effective safe-

[5] Developers' ability to restrict model access post-release is important, as it allows them to retract problematic model capabilities if their harm exceeds expectations.

guards.

In summary, we acknowledge the concerns outlined above and the significant work required to improve LLM safeguards. Nevertheless, we argue that the controlled customisability of CA models uniquely enables their functionality to match the sophistication of their safeguards. Moreover, by using harmful prompts as training data, CA models can gradually improve at detecting and mitigating malicious user activity. Therefore, we reassert that CA models can form a central part of a popular and safe LLM ecosystem.

3.3. Public Capacity to Prompt and Scrutinise the Model

A user's ability to prompt and scrutinise a CA model is arguably comparable to their ability to do so with a queryable model. Specifically, they afford users the same capacity to engage in prompt-response dialogues, which can elicit helpful information or test the model for various biases. Unfortunately, examining a model for biases solely via prompt-response pairs is a cumbersome and imperfect science. As such, by preventing users from directly probing their code, CA models (much like queryable approaches) don't quite facilitate the scrutiny that's offered by open-source approaches[6].

Moreover, an individual user's ability to prompt and scrutinise a CA model is contingent on their 'good behaviour', as defined by the AI company which controls the API. Therefore, some may argue that CA models produce the illusion of distributed power, while private companies maintain control. Perhaps open-sourcing is the only way to truly 'democratise' a language model. Addressing these concerns, we reiterate our claim that the decentralised nature of open-source models raises the risk of misuse. At least for this stage of technological development, a centralised authority is well-placed to safeguard, monitor, and restrict a model's actions.

Nevertheless, companies which provide highly capable LLMs still have a duty seek external input and scrutiny, a goal they can pursue without resorting to open-sourcing. In order to 'democratise' [39] the control of AI, we support the suggestions made by Serger et al. [38], three of which we highlight below. Firstly, AI companies could elicit public input for complex normative decisions, giving the end-user a seat at the table. Secondly, they could incorporate more democratic organizational structures, such as public benefit corporations. This would provide a "clearer legal standing to make decisions about institutional structure that aim to maximize public benefit." Thirdly, there is a greater role for democratic governments in restricting and auditing the development and release of highly capable language models [11].

The suggestions outlined above provide an alternative means for external input and scrutiny when open-source access isn't feasible. Yet, there remains a subtle danger to CA models which these suggestions cannot solve. In particular, as CA models block users from directly scrutinising their code, it might be difficult to determine whether modifications are truly aligned with a user's intentions. For example, a user could instruct their model to "Read through law reports and become an accurate judge." The model might respond by incorporating racial classifiers into their architecture, which could improve accuracy by perpetuating systematic biases.

In this scenario, the model has changed its behaviour in response to the user's instruction, but not in alignment with their intentions. This misalignment is especially difficult to detect when using a CA model, since their code is hidden from the user. How might we ensure that CA models are well-aligned if we can't scrutinise their code? One approach [21] that's emerged for improving alignment involves developing models which are uncertain, and even skeptical, about whether a user's input accurately describes their true intentions. The intuition is that this uncertainty might incentivise a CA model to consult users when it's unsure about whether its modifications match their intentions. In our legal example, one would hope that the model identifies (and attempts to clarify) that the user might also have an unobserved desire to avoid prejudice.

In summary, we observed that CA models can be prompted and scrutinised in much the same way as a queryable approach. Furthermore, despite the absence of publicly released code, companies can still 'democratise' a CA model via public involvement or alternative organisational structures. Finally, we examined the alignment problem for LLMs, highlighting the challenge of scrutinising whether their behavior matches the user's intentions. In response, we suggested that CA models could act with uncertainty about their user's intentions, and might even be prudent to scrutinise the users themselves.

3.4. Public Capacity to Develop and Customise the Model

CA models are at least as customisable as closed and queryable approaches. Closed models prevent users from even prompting them, let alone modifying their code. Queryable models can adapt (in-context) to a specific prompt; however, these changes are transient and the underlying code remains the same. In contrast, conversations with CA models can permanently alter aspects of their code,

[6] An important caveat is a mechanism (which CA models could incorporate) for semi-directly probing a model's code, as proposed by Shevlane [41], where "the external researcher uploads to the developer a tool that automates the interpretability analysis, then the developer runs that tool on the model and sends back the feedback."

including their system prompts, weights, or inference procedure. There is even an argument that CA models are more customisable than open-source approaches. In particular, CA-release lowers the barrier to entry for model customisation.

Anyone can change the code of a CA model simply by talking to it, but only an expert can modify the code of an open-source model. Unfortunately, the argument that CA-release is at least as customisable as open-sourcing is damaged by the following asymmetry. An adept programmer can transform an open-source model into a CA one, but the reverse is clearly impossible. As such, it's evident that open-source models are strictly more customisable tha CA approaches. Nevertheless, by convincingly challenging open-source models in this regard, CA-release shows itself to be highly customisable approach that isn't far behind open-sourcing.

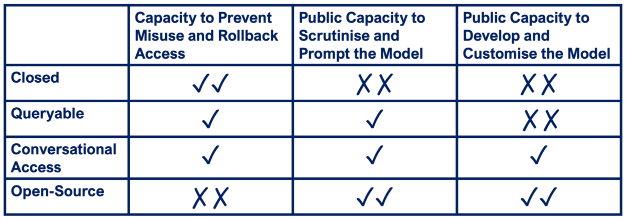
We have just observed that CA models fall between queryable and open-source approaches with regards to customisability. However, the open-source community not only excels at customisation but also has a strong tradition of research and development. For instance, they were central to recent breakthroughs, such as Low Rank Adaptation[7] [26], a method that reduces the cost of fine-tuning. Developers inability to directly tinker with the code of a CA model would hinder similar research efforts. As such, it's hard to imagine CA models generating the same culture of research and development that open-source approaches so often produce.

In summary, while queryable models exhibit a certain level of rigidity, CA-release not only facilitates, but also reduces the barrier to entry for user customisation. Nevertheless, CA models are clearly not quite as flexible, customisable, and research-conducive as an open-source approach. For this reason, we suggest that they provide the public with an adequate capacity to develop and customise their models, albeit one that is surpassed by open-source models.

3.5. Summarising CA Models

In this section, we introduced CA-release, an approach where users can modify a model's underlying behaviour through natural language interactions alone. The controlled customisability of CA models allow them to (at least in theory) meet an ever-changing set of ethical expectations and market demands. Therefore, in Table 2, we assign no critical failures to CA models, suggesting that they mostly retain the desirable properties of a queryable approach while enabling a degree of customisation closer to open-source approaches.

Assigning all 'ticks' to CA-release does not suggest that we believe it is always the best approach. Although CA models find their niche in situations that require a trade-off between customisation and safety, we accept that in many cases such trade-offs might not be appropriate. When developing an LLM that is trained on classified information, a model's propensity to leak this data to malicious actors may take priority over other factors, making a closed approach the most suitable. Alternatively, smaller 'plug-and-play' LLMs that enable researchers to experiment with novel ideas are perhaps most suited for open-sourcing.



	Capacity to Prevent Misuse and Rollback Access	Public Capacity to Scrutinise and Prompt the Model	Public Capacity to Develop and Customise the Model
Closed	✓✓	✗✗	✗✗
Queryable	✓	✓	✗✗
Conversational Access	✓	✓	✓
Open-Source	✗✗	✓✓	✓✓

Table 2. This table adds CA to the list of existing approaches for model accessibility given in Table 1. It suggests that CA models can perform adequately across all metrics.

4. Related Literature

The main focus of this paper has been 'Conversational Access to an LLM's Code'. In this section, we consider the literature regarding three related topics:

1. Access to an LLM's Code: In Section 2, we simplified the existing body of work on LLM release into three categories: open-source, queryable, and closed. Below, we consider some important proposals which sit outside of this simple picture.

2. Conversational Access to an Algorithm: There already exists a large body of research into conversation-mediated human-computer interactions, albeit not with regards to an LLM's source code. In this section, we explore the case of conversational recommendation systems.

3. Copilots for an LLM's Code: CA models turn an LLM into a intermediary between the user and its source code. We now consider an alternative possibility where LLM assistants (i.e., Copilots) help users to write and understand their code.

4.1. Access to an LLM's Code

Downloadable models [43], such as Craiyon, allow users to download the model's code through a user interface,

[7] LoRa involves learning a pair of low-dimensional (i.e., cheap to calculate or fine-tune) weight matrices which complement a model's original high-dimensional (i.e., expensive to calculate or fine-tune) weight matrix, thereby improving its performance.

subject to approval. This differs from both closed and queryable models, which restrict source code access to a single company or lab. It also differs from open-souring, where the code is made publicly accessible, without exception. In theory, downloadable release can help to stress-test an LLM before it's released more broadly. Nevertheless, we express our concern that downloadable release is inherently unstable and can prematurely collapse into open-sourcing. Specifically, it rests on the assumption that none of the model-holders will disseminate, or even accidentally leak, the code. For this reason, in the main body of this paper we instead focused on closed, queryable, and open-source models, all three of which we view as more stable.

Structured Access [13,41] allows users to engage in "controlled, arm's length interactions with their AI systems," subject to approval. This arm's length approach prevents users from obtaining the information required to recreate and disseminate the model themselves, which was our concern with downloadable release. An example of structured access is GPT-3 [10], which allows approved users to fine-tune a GPT-3 model or integrate it into their own application using OpenAI's API. More broadly, many of the approaches that we've examined so far could be considered as a form of structured access, including queryable and CA models. Notably, CA-release is on this list as it mediates (i.e., structures) a user's access to the model's source code through natural language conversations, a rather stark example of an arm's length interaction.

Staged Disclosure [41, 43, 44] involves gradually releasing the model's components over time. Examples of this include GPT-2 [34], whose parameters were originally released in four stages, with the number of released parameters increasing at each stage. This gradual deployment allows developers to test their safeguards in a low-capability regime before releasing their most capable models. We view staged disclosure as a coarse tactic that can be applied alongside the methods that we've discussed so far. In particular, for almost any level of access, companies might be wise to initially release smaller versions of their model. As such, we view staged disclosure as orthogonal to the approaches explored in the main body of this paper and express our hope that it can be employed in tandem with CA-release.

4.2. Conversational Access to an Algorithm

The public's capacity to modify algorithms via natural language has the potential to transform software release. One particularly active area of research is whether LLMs can mediate natural language interactions between a user and their recommendation system (RS). Typically, an RS selects the content that a user consumes whilst online, based on factors like their social network and search history [15]. They are perhaps most recognisable for their role in social media algorithms, where they are trained to maximise user engagement, keeping our attention for as long as possible. This can produce an undesirable [47,51] (yet addictive [2]) experience for the user, over which they have little recourse for control [14].

In theory, by conversing with an LLM, individuals could continually modify the incentives of their RS to align with their current mood or interests. For example, a user could request more scientific content when they feel inquisitive and less negative content when they feel vulnerable, taking back control over their digital experience. In the words of Lazar [28] "They could provide recommendation and filtering without surveillance and engagement-optimisation." But is there evidence that conversations with an LLM can actually improve user's recommendations?

A recent paper by researchers at Netflix [23] demonstrated that GPT-4 can generate highly accurate movie recommendations when provided with a system prompt that details its role as an RS. Similarly, Google [20] recently explored whether the YouTube RS could be improved by incorporating ”an LLM to match preferences extracted from the context of the conversation to item metadata." In this example, a user's conversation with an LLM (effectively) modifies the functionality of an RS, demonstrating that the tenets of CA models can be applied to software release more broadly.

4.3. Copilots for an LLM's Code

The main body of this paper examined whether LLMs could *mediate* modifications to a model's source code. However, with the advent of products like GitHub Copilot [32], there is a growing appetite for LLMs designed to *assist* users in modifying their code. In particular, assistance is distinct from mediation in that the user maintains direct control over their code while also having the option to converse with their copilot for suggestions and explanations [40]. As such, among the approaches covered in this paper, copilots could only be applied to models that are directly modifiable, namely those that are downloadable or open-source. While open-sourcing the code of a highly-capable LLM might pose unacceptable risks, copilots are well-suited for assisting users to modify smaller (low-capability) models intended for research or education [6,37].

5. Conclusion

Closed, queryable, and open-source approaches all fail to produce a customisable yet safe release strategy. In this paper we proposed conversational access as an approach that (ideally) permits users to tailor a model to their individual needs, without jeopardising the security of the underlying code. In particular, a user's capacity to modify a CA model can be controlled and continually adjusted to match the ever-changing demands for a customisable yet safe LLM ecosystem. Due to this flexibility, conversational access appears to be an appropriate release strategy for a broad spectrum of highly-capable LLMs.

Regardless of these benefits, one thing is certain: There is no silver bullet for ensuring that language models are both helpful and harmless in every situation [4, 5]. When developing or regulating LLMs, practitioners and policymakers must thoughtfully consider the balance between democratisation, innovation, and safety, with no simple solutions. Rather than a comprehensive solution, we hope that conversational access can serve as one tool out of many for navigating the release of highly capable language models.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Hunt Allcott, Matthew Gentzkow, and Lena Song. Digital addiction. *American Economic Review*, 112(7):2424–2463, 2022.

[3] Markus Anderljung and Julian Hazell. Protecting society from ai misuse: When are restrictions on capabilities warranted? *arXiv preprint arXiv:2303.09377*, 2023.

[4] Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, et al. A general language assistant as a laboratory for alignment. *arXiv preprint arXiv:2112.00861*, 2021.

[5] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

[6] Brett A Becker, Paul Denny, James Finnie-Ansley, Andrew Luxton-Reilly, James Prather, and Eddie Antonio Santos. Programming is hard-or at least it used to be: Educational opportunities and challenges of ai code generation. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*, pages 500–506, 2023.

[7] Richard Blumenthal and Josh Hawley. Meta llama model leak letter. `https://www.blumenthal.senate.gov/imo/media/doc/06062023metallamamodelleakletter.pdf`, 2023.

[8] Nick Bostrom. The superintelligent will: Motivation and instrumental rationality in advanced artificial agents. *Minds and Machines*, 22:71–85, 2012.

[9] Greg Brockman, Atty Eleti, Elie Georges, Joanne Jang, Logan Kilpatrick, Rachel Lim, Luke Miller, and Michelle Pokrass. Introducing chatgpt and whisper apis. *OpenAI Blog*, 2023.

[10] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[11] Miles Brundage, Shahar Avin, Jasmine Wang, Haydn Belfield, Gretchen Krueger, Gillian Hadfield, Heidy Khlaaf, Jingying Yang, Helen Toner, Ruth Fong, et al. Toward trustworthy ai development: mechanisms for supporting verifiable claims. *arXiv preprint arXiv:2004.07213*, 2020.

[12] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

[13] Benjamin S Bucknall and Robert F Trager. Structured access for third-party research on frontier ai models: Investigating researchers' model access requirements. 2023.

[14] Justin Bullock and Anton Korinek. @ elonmusk and@ twitter: The problem with social media is misaligned recommendation systems, not free speech. 2022.

[15] Meta Transparency Center. Our approach to facebook feed ranking. `https://transparency.fb.com/en-gb/features/ranking-and-content/`, 2023.

[16] Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-play fine-tuning converts weak language models to strong language models. *arXiv preprint arXiv:2401.01335*, 2024.

[17] Adrien Ecoffet, Jeff Clune, and Joel Lehman. Open questions in creating safe open-ended ai: tensions between control and creativity. In *Artificial Life Conference Proceedings 32*, pages 27–35. MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info . . . , 2020.

[18] Alex Engler. How open-source software shapes ai policy. 2021.

[19] Richard Fang, Rohan Bindu, Akul Gupta, Qiusi Zhan, and Daniel Kang. Llm agents can autonomously hack websites. *arXiv preprint arXiv:2402.06664*, 2024.

[20] Luke Friedman, Sameer Ahuja, David Allen, Terry Tan, Hakim Sidahmed, Changbo Long, Jun Xie, Gabriel Schubiner, Ajay Patel, Harsh Lara, et al. Leveraging large language models in conversational recommender systems. *arXiv preprint arXiv:2305.07961*, 2023.

[21] Iason Gabriel and Vafa Ghazavi. The challenge of value alignment: From fairer algorithms to ai safety. *arXiv preprint arXiv:2101.06060*, 2021.

[22] Pranav Gade, Simon Lermen, Charlie Rogers-Smith, and Jeffrey Ladish. Badllama: cheaply removing safety fine-tuning from llama 2-chat 13b. *arXiv preprint arXiv:2311.00117*, 2023.

[23] Zhankui He, Zhouhang Xie, Rahul Jha, Harald Steck, Dawen Liang, Yesu Feng, Bodhisattwa Prasad Majumder, Nathan Kallus, and Julian McAuley. Large language models as zero-shot conversational recommenders. In *Proceedings of the 32nd ACM international conference on information and knowledge management*, pages 720–730, 2023.

[24] Dan Hendrycks. Natural selection favors ais over humans. *arXiv preprint arXiv:2303.16200*, 2023.

[25] Lisa M Hepburn and David Hemenway. Firearm availability and homicide: A review of the literature. *Aggression and Violent behavior*, 9(4):417–440, 2004.

[26] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[27] Megan Kinniment, Lucas Jun Koba Sato, Haoxing Du, Brian Goodrich, Max Hasin, Lawrence Chan, Luke Harold Miles, Tao R Lin, Hjalmar Wijk, Joel Burget, et al. Evaluating language-model agents on realistic autonomous tasks. *arXiv preprint arXiv:2312.11671*, 2023.

[28] Seth Lazar. Frontier ai ethics. 2024.

[29] Dan Milmo. Google engineer warns it could lose out to open-source technology in ai race. *The Guardian*, 2023.

[30] OpenAI. Introducing gpts. `https://openai.com/blog/introducing-gpts#OpenAI`, 2023.

[31] OpenAI. Welcome to the openai platform. `https://platform.openai.com`, (visited January 17, 2020).

[32] Sida Peng, Eirini Kalliamvakou, Peter Cihon, and Mert Demirer. The impact of ai on developer productivity: Evidence from github copilot. *arXiv preprint arXiv:2302.06590*, 2023.

[33] Xiangyu Qi, Yi Zeng, Tinghao Xie, Pin-Yu Chen, Ruoxi Jia, Prateek Mittal, and Peter Henderson. Fine-tuning aligned language models compromises safety, even when users do not intend to! *arXiv preprint arXiv:2310.03693*, 2023.

[34] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[35] Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.

[36] Thomas Reed and Danny Stillman. *The nuclear express: A political history of the bomb and its proliferation*. Zenith press, 2010.

[37] Advait Sarkar. Will code remain a relevant user interface for end-user programming with generative ai models? In *Proceedings of the 2023 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software*, pages 153–167, 2023.

[38] Elizabeth Seger, Noemi Dreksler, Richard Moulange, Emily Dardaman, Jonas Schuett, K Wei, Christoph Winter, Mackenzie Arnold, Seán Ó hÉigeartaigh, Anton Korinek, et al. Open-sourcing highly capable foundation models: An evaluation of risks, benefits, and alternative methods for pursuing open-source objectives. *arXiv preprint arXiv:2311.09227*, 2023.

[39] Elizabeth Seger, Aviv Ovadya, Divya Siddarth, Ben Garfinkel, and Allan Dafoe. Democratising ai: Multiple meanings, goals, and methods. In *Proceedings of the 2023 AAAI/ACM Conference on AI, Ethics, and Society*, pages 715–722, 2023.

[40] Abigail Sellen and Eric Horvitz. The rise of the ai co-pilot: Lessons for design from aviation and beyond. *arXiv preprint arXiv:2311.14713*, 2023.

[41] Toby Shevlane. Structured access: an emerging paradigm for safe ai deployment. *arXiv preprint arXiv:2201.05159*, 2022.

[42] Emily H Soice, Rafael Rocha, Kimberlee Cordova, Michael Specter, and Kevin M Esvelt. Can large language models democratize access to dual-use biotechnology? *arXiv preprint arXiv:2306.03809*, 2023.

[43] Irene Solaiman. The gradient of generative ai release: Methods and considerations. In *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency*, pages 111–122, 2023.

[44] Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, et al. Release strategies and the social impacts of language models. *arXiv preprint arXiv:1908.09203*, 2019.

[45] Joe Spisak and Sergey Edunov. The llama ecosystem: Past, present, and future. `https://ai.meta.com/blog/llama-2-updates-connect-2023/`, 2023.

[46] Guanhong Tao, Siyuan Cheng, Zhuo Zhang, Junmin Zhu, Guangyu Shen, and Xiangyu Zhang. Opening a pandora's box: Things you should know in the era of custom gpts. *arXiv preprint arXiv:2401.00905*, 2023.

[47] Petter Törnberg. How digital media drive affective polarization through partisan sorting. *Proceedings of the National Academy of Sciences*, 119(42):e2207159119, 2022.

[48] Alexander Tornede, Difan Deng, Theresa Eimer, Joseph Giovanelli, Aditya Mohan, Tim Ruhkopf, Sarah Segel, Daphne Theodorakopoulos, Tanja Tornede, Henning Wachsmuth, et al. Automl in the age of large language models: Current challenges, future opportunities and risks. *arXiv preprint arXiv:2306.08107*, 2023.

[49] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[50] Tao Tu, Anil Palepu, Mike Schaekermann, Khaled Saab, Jan Freyberg, Ryutaro Tanno, Amy Wang, Brenna Li, Mohamed Amin, Nenad Tomasev, et al. Towards conversational diagnostic ai. *arXiv preprint arXiv:2401.05654*, 2024.

[51] Jean M Twenge. Increases in depression, self-harm, and suicide among us adolescents after 2012 and links to technology use: possible mechanisms. *Psychiatric Research and Clinical Practice*, 2(1):19–25, 2020.

[52] Fabio Urbina, Filippa Lentzos, Cédric Invernizzi, and Sean Ekins. A teachable moment for dual-use. *Nature machine intelligence*, 4(7):607–607, 2022.

[53] Yuxuan Wan, Wenxuan Wang, Pinjia He, Jiazhen Gu, Haonan Bai, and Michael R Lyu. Biasasker: Measuring the bias in conversational ai system. In *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 515–527, 2023.

[54] Ben Wang and Aran Komatsuzaki. Gpt-j-6b: A 6 billion parameter autoregressive language model. `https://github.com/kingoflolz/mesh-transformer-jax`, 2021.

[55] Zeming Wei, Yifei Wang, and Yisen Wang. Jailbreak and guard aligned language models with only few in-context demonstrations. *arXiv preprint arXiv:2310.06387*, 2023.

[56] Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, et al. Ethical and social risks of harm from language models. *arXiv preprint arXiv:2112.04359*, 2021.

[57] Jess Whittlestone and Aviv Ovadya. The tension between openness and prudence in ai research. *arXiv preprint arXiv:1910.01170*, 2019.

[58] Roman V Yampolskiy. Analysis of types of self-improving software. In *Artificial General Intelligence: 8th International Conference, AGI 2015, AGI 2015, Berlin, Germany, July 22-25, 2015, Proceedings 8*, pages 384–393. Springer, 2015.

[59] Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. *arXiv preprint arXiv:2309.03409*, 2023.

[60] Jiahao Yu, Yuhang Wu, Dong Shu, Mingyu Jin, and Xinyu Xing. Assessing prompt injection risks in 200+ custom gpts. *arXiv preprint arXiv:2311.11538*, 2023.