A Critical Survey on LLM Deployment Paradigms: Assessing Usability and Cognitive Behavioral Aspects

Anonymous ACL submission

Abstract

Over the last decade, a wide range of training and deployment strategies for Large Language Models (LLMs) have emerged. Among these, the prompting paradigms of Auto-Regressive LLMs (AR-LLMs) have catalyzed a significant surge. This paper embarks on a quest to unravel the underlying factors behind the triumph of AR-LLMs' prompting paradigm. This study summarizes and focuses on six distinct task-oriented channels, e.g., numeric prefixes and free-form text, across diverse deployment paradigms By pivoting our focus onto these channels, we can assess these paradigms across crucial dimensions, such as task customizability, transparency, and complexity to gauge LLMs. The results emphasize the significance of utilizing free-form contexts as user-directed channels for downstream deployment. Moreover, we examine the stimulation of diverse cognitive behaviors in LLMs through the adoption of free-form, verbal outputs and inputs as contexts. We detail four common cognitive behaviors to underscore how AR-LLMs' prompting successfully imitates human-like behaviors under the free-form modality and channel.

1 Introduction

ChatGPT has emerged as the most popular AI application, with a vast user base. The success of GPT models can be attributed to the scaling of transformer-based neural networks and the extensive pre-training data, as explored in previous studies (Radford et al., 2019; Brown et al., 2020). The scope of this paper is directed towards Large Language Models (LLMs) that are sufficiently large to acquire world knowledge, commonsense, and the linguistic capabilities required to attain high performance on benchmarks such as GLUE (Wang et al., 2019).

Although LLMs are commonly perceived as general-purpose language intelligence models, the practice often diverges from employing a singular, all-encompassing model for every task. Instead, the deployment frequently entails developing a suite of specialized models tailored to specific tasks. This specialization is facilitated through the introduction of task-specific channels, modifying the model's structure or its pre-trained parameters to better suit the nuances of individual tasks. This highlights a departure from the ideal of a universal, one-size-fits-all model, while the broad capabilities of LLMs suggest they could serve as jack-of-all-trades in language processing. This trend towards creating task-specific models may stem from the tradition of evaluating linguistic intelligence through a variety of distinct tasks and benchmarks (Wang et al., 2019), with researchers striving to excel in these tasks independently to set new benchmarks. In this paper, we delve into the mechanisms behind prevalent deployment paradigms including AR-LLMs' prompting, which underpins ChatGPT's operation, and highlight several critical observations: 1) Models tailored with optimized task-specific channels often suffer from issues related to task customizability, transparency, and user-level complexity during deployment, affecting their overall usability; 2) Anticipated to mimic human-like intelligence, they often exhibit slow thinking through shortcuts (Kahneman, 2011); 3) They frequently fall short in showcasing advanced cognitive behaviors, which we contend are vital for convincing users of the models' intelligence. Conversely, AR-LLMs' prompting paradigms introduce a more natural, human-like channel (verbal free-form context) for representing a wide array of real-life tasks and employ form-form output modalities to showcase cognitive behaviors in complex scenarios.

Specifically, in this paper, we commence by examining the foundational principles of language modeling, revisiting the notable split in language modeling approaches that emerged in the late 2010s: auto-encoding LMs (AE-LMs) exemplified by BERT (Jin et al., 2020) and auto-regressive LMs (AR-LMs) exemplified by the GPT series (Radford et al., 2018; Brown et al., 2020). Rather than delve into an extensive array of deployment paradigms, we introduce and discuss the concepts of modalities and channels to investigate the usability of the deployment paradigms (§2). Upon evaluating different deployment paradigms for LLMs, it becomes clear that aside from the AR-LLMs' prompting approach, other paradigms struggle to demonstrate advanced human-like cognitive behaviors. This shortfall is attributed to the constraints within modalities and channels, coupled with a tendency towards superficial learning, i.e., slow thinking (§3 and §4.1). In contrast, via specified context in the free-from text, the AR-LLMs' prompting strategy imitate human-like cognitive behaviors, such as reasoning, planning, and feedback learning, which are elucidated in Table 2 (§4).

2 Deploying Large Language Models

This section elucidates the dual objectives underlying language models, which both aim to model the joint probability distribution of text sequences through self-supervised learning techniques and generate text that is relevant to the given context. After this introduction, we present a novel framework that facilitate the characterization of various deployment paradigms through two types of data modalities, which support language comprehension, coupled with six unique channels for processing these modalities.

2.1 The Fundamental Dichotomy in Language Modeling

Objective of Language Modeling The goal of language modeling is to estimate the joint probability distribution of sequences of text (Bengio et al., 2003). This involves developing two distinct yet relaxed formulations for constructing LLMs that leverage self-supervised learning from vast quantities of unlabeled text data. The self-supervised approach enables the training of LLMs on extensive text corpora, a practice that has been thoroughly investigated in various studies (Liu et al., 2019; Wei et al., 2022a). This paper focuses on how the intrinsic design of language models impacts their usability and potential to express cognitive behaviors.

Auto-Regressive (Left-to-Right) Language Modeling Typically, language modeling is approached by predicting the subsequent token in a sequence based on the preceding tokens. This prediction is quantified as the product of conditional probabilities for each subsequent token, considering its previous tokens, in accordance with the chain rule (Bengio et al., 2003).

$$P\left(w_1, \ldots, w_N\right)=\prod_{t=1}^N P\left(w_t \mid w_0, \ldots, w_{t-1}\right) \tag{1}$$

Here, w_0 serves as a marker for the beginning of text.

Auto-Encoding (Denoising) Language Modeling In the context of auto-encoding language modeling, noise is intentionally introduced to an input sequence $w_1, w_2, ...w_N$. The primary aim is to optimize

$$\max \prod_{t=1}^N P\left(w_t \mid \hat{w}_1, \ldots, \hat{w}_N\right) \tag{2}$$

where $\hat{w}_1, \hat{w}_2, ...\hat{w}_N$ represents the altered, noise-added version of the input sequence. The approach of masking specific tokens in the text at random, known as token-level masked language modeling (Devlin et al., 2019), is a widely adopted strategy. This involves substituting original tokens with a special token, such as "[MASK]", and training the model to predict these original tokens based on the context of the surrounding, unmasked tokens. The discrepancy between the original and reconstructed sequences is quantified through a reconstruction loss:

$$L_{reconstruction}=-\sum_{t=1}^N \log P\left(w_t \mid \hat{w}_1, \ldots, \hat{w}_N\right) \tag{3}$$

This denoising methodology also includes other variants such as span-level masked language modeling (Joshi et al., 2020), text infilling (Lewis et al., 2020), among others.

2.2 Exploring the Modalities within Large Language Models

This section delves into the concept of "modalities" within LLMs, a term often implicitly associated with research on multimodal systems to describe diverse, human-like channels of communication, such as text, speech, gestures, and visual inputs (Bartneck et al., 2020). Here, "modalities" specifically refer to the various forms of input and output data utilized in LLM deployment.

In the operation of both AR-LLMs and AE-LLMs, we identify three primary modalities: a unique textual modality for both the input and output in AR-LLMs (unrestricted text), a distinct textual modality for AE-LLMs (masked text or contextualized n-grams), and a shared modality of intermediate dense representations applicable to both models: 1) **Intermediate Dense Representations**: Fundamentally, LLMs convert each word (or subword) in a sequence into dense vector embeddings. These embeddings are generated through a series of mathematical operations, such as the self-attention mechanism, at every layer of the neural network, and are represented as $\{h_i^l\}$ for every position i within the sequence and for every layer l in the model. Here, i ranges from 1 to N, with N indicating the total number of elements in the sequence, and l spans from 1 to L, where L represents the complete count of layers within the model. 2) **Textual Modalities**: AE-LLMs feature an input modality of masked text, with the output modality being contextualized n-grams designed to reconstruct the masked sections. Conversely, due to their auto-regressive design, AR-LLMs are capable of encoding any text as context and generating free-form text outputs, thereby employing unrestricted text for both input and output. These modalities are inherently linked to their respective language modeling strategies.

2.3 Task-specific Channels for Deployment

To tailor the core capabilities of LLMs for specific downstream tasks, both input and intermediate modalities can be altered directly (for instance, by appending prefixes or incorporating verbal context) or indirectly through the use of parametric modules such as neural networks, including adapters and output layers as described subsequently. It's worth noting that direct modifications, such as prefixes, can also be achieved using parametric modules. These parametric modules undergo optimization via task-specific supervised learning. In this context, we describe the means for modality transformation aimed at specific tasks as task-specific channels. For clarify, modalities are the types of data or the form in which data is processed, while channels are the pathways or methods through which these data modalities are adapted or transformed for specific tasks. Task-specific channels encompass: 1) **Adapter**: Adapters are compact neural networks that can be embedded between an LLM's layers. A well-known approach, adapter tuning (Houlsby et al., 2019), involves optimizing the adapter's parameters while leaving the original LLM parameters intact. These adapters are designed to adjust the intermediate layer representations to better align with task-specific needs. 2) **LLMs Themselves**: An alternative strategy involves modifying the LLM directly to produce task-specific representations by fine-tuning the model's weights across all or selected layers. This method of fine-tuning is prevalent for AE-LLMs (Jin et al., 2020) and has also been applied to AR-LLMs in early use of GPT-like models (Radford et al., 2018). 3) **Output Layers**: Once task-specific representations are produced by either adapters or the LLM directly, the function of the output layers is to translate these representations into a designated output space. These layers typically consist of one or several linear layers. For example, linear functions are frequently used for tasks involving classification, while tasks that involve extractive question answering often necessitate the use of two linear functions to determine the beginning and concluding positions of the answer within a text passage. 4) **Activation Prefixes**: Within the scope of deploying LLMs via task-specific supervised learning, where training neural networks is common, prefix tuning (Li and Liang, 2021) presents an innovative method that employs prefixes to directly modify intermediate representations. These prefixes are essentially embeddings that are added at various layers, with dimensions identical to those of token embeddings, functioning as virtual tokens. Introducing these prefixes at earlier stages in the model allows for the infusion of task-specific information into more advanced layers, thereby improving the model's alignment with the desired task objectives.

Beyond the four channels previously outlined, verbal channels offer a unique approach for articulating the task context in which LLMs can identify and execute the intended tasks. These channels include: 5) **Verbal Free-form Context**: In this approach, a context is articulated using free-form text, such as task instructions and few-shot demonstrations, which can activate complex cognitive functions. By merely incorporating task instructions within the context, AR-LLMs are enabled to undertake a multitude of tasks through zero-shot prompts. Another widely adopted method is few-shot prompting (Radford et al., 2019; Brown et al., 2020), which involves learning from a limited number of examples for in-context learning without the need for gradient updates, showcasing a human-

Channels	Relevant Paradigms	Customizability	Transparency	Complexity
Adapter	Adapter tuning	✗	✗	T
Output layers	LLM fine-tuning; Adapter tuning	✗	✗	T
LLMs	LLM fine-tuning; PET	✗	✗	T
Activation prefixes	Prefix tuning	✗	✗	T
Verbal free-form context	AR-LLMs' prompting	✓	✓	0
Contextual text patterns	PET; Auto-prompt	✗	✓(PET); ✗(Auto-prompt)	$N \times T$

Table 1: Evaluation of deployment channels for language models: A comparative analysis of task customizability, transparency and complexity from the users' perspective. PET: Pattern exploitation training; T: the total number of task; N: the number of patterns per task.

like efficiency in acquiring new tasks. This method is particularly effective in eliciting cognitive behaviors akin to those observed with few-shot demonstrations, with further details discussed in Section 4. It's important to recognize that, in contrast to channels that are easily differentiated by input-side modalities (such as task-specific examples), this channel (e.g., task instructions) can intertwine with model inputs, e.g., task-specific examples. This allows for the seamless integration of the models' world knowledge into tasks, for instance, "summarize deep learning technology". 6) **Contextual Text Patterns**: Given their training on a denoising language model objective, AE-LLMs excel in completing texts by filling in missing words, a trait that can be leveraged for downstream tasks. Task-specific patterns, in this regard, serve as a mechanism to alter given task-specific examples. Typically, this involves appending the examples with a cloze-style phrase or sentence (text with missing words) tailored to the task, allowing the model to predict the intended task outcomes based on the placeholders filled within the text. Pattern Exploitation Training (PET) (Schick and Schütze, 2021) involves the creative design of task-specific patterns and the fine-tuning of LLMs to these patterns. Conversely, auto-prompt methods (Shin et al., 2020) seek to optimize task-specific patterns to better fit the models, enhancing their ability to interpret and respond to the given tasks effectively.

3 Evaluation of Modalities and Channels

3.1 Evaluating Usability of Deployment Channels

This section introduces a framework for assessing the usability of language model deployment channels, focusing on their customizability, transparency, and complexity, as summarized in Table 1.

Customizability of User-level Tasks: Extent of User Control over Channels Essentially, any task can be articulated in human languages, such as English, using free-form context. This adaptability is a testament to the evolution of human language over thousands of years, which has been refined to describe a vast array of everyday and complex scientific problems. Typically, in a zero-shot learning context, the channel consists solely of task instructions within the prompts, capable of encompassing a wide range of tasks. For instance, Wang et al. (2022) have converted standard NLP datasets designed for optimized channels into instruction-based formats for 76 different tasks. Moreover, free-form task instructions allow for nuanced control mechanisms, including explicit directives (such as specifying output formats or initiating reasoning processes) and subtle cues (such as inducing cognitive behaviors through few-shot examples). These aspects will be further explored in Section 4 and summarized in Table 2. In contrast, since other channels are set during the optimization process for specific tasks, they lack the flexibility for user-directed modifications. Channels that require adjustments, such as fine-tuning the LLM, adapter tuning, or prefix tuning, rely on supervised learning methods for configuration. Although prompting in AE-LLMs could, in theory, facilitate task adjustments at inference time without prior task-specific fine-tuning—akin to AR-LLMs' prompting approach—it often requires task-specific optimization to achieve effective channel performance. For example, techniques like Pattern Exploitation Training (PET) (Schick and Schütze, 2021) utilize mathematical optimization to adapt models to specific patterns, whereas Auto-prompt (Shin et al., 2020) optimizes text patterns for language models. The question of whether this need for optimization arises from the inherent complexities of auto-encoding language models invites further research.

User-level Transparency: Can Channel Formulation Be Easily Understood by Users? The focus here is on the understandability of the channels themselves to lay users, rather than their functional effectiveness, as this greatly influences the user experience. For example, the objective of an output layer is clear — transforming LLM representations into a specific output format. However, the process involving dense representations through matrix multiplication is not intuitively understandable to the non-specialist. Moreover, text patterns refined through AE-LLMs' Auto-prompting often lack the straightforwardness found in manually created prompts.

User-level Complexity: Assessing the Number of Conceptual Components This analysis evaluates the conceptual load required to deploy T tasks using various channels, moving away from the parameter size metric, which is more pertinent to researchers and developers. Assuming each task is accommodable across all channels, we quantify the complexity as follows: For fine-tuned LLMs, prefixes, adapters and output layers, each task-specific adjustment equates to a complexity of T, with T denoting the total number of tasks. Additionally, N text patterns are devised per task, resulting in a complexity of $N \times T$, where N represents the number of patterns per task. The complexity for verbal free-form context is considered negligible, as these are formulated spontaneously by users at the time of use. From this framework, we can deduce the complexity inherent to each deployment paradigm. For instance, LLM fine-tuning, which necessitates one LLM and one output layer per task, carries a complexity of $2 \times T$.

3.2 Evaluating Expressiveness of Modalities

During LLM fine-tuning and adapter tuning, the task-specific output layers strictly limit the range of possible outputs, hindering the potential for detailed expressiveness and, by extension, advanced cognitive behaviors. The output space is tightly defined, with actions or labels being pre-determined and given specific meanings through task-specific supervised learning. Nonetheless, certain probing techniques allow us to uncover the thought processes behind their predictions, a topic we will explore further in Section 4.1. When it comes to AE-LLMs prompted with text patterns, these models are limited to generating only specific tokens or words, constrained by the patterns set in advance. These constraints, such as token positions and quantities dictated by the input patterns, along with the need for grammatical and coherent text completion, restrict the models' ability to articulate complex ideas, plans, and actions. On the other hand, AR-LLMs' prompting capitalizes on their auto-regressive nature to produce unbounded, free-form text, influenced solely by the given input context. This capability is further demonstrated in Section 4 and summarized in Table 2, showcasing the open-ended expressiveness unique to the AR-LLM prompting paradigm.

4 Cognitive Behaviors Under AR-LLMs' Prompting Paradigm

This section elucidates the capability of AR-LLM prompting paradigms to exhibit cognitive behaviors expressed by the free-form modalities by mainpulating the free-form channels. It's important to clarify that not every AR-LLM demonstrates cognitive behaviors—smaller models like GPT-2 (Radford et al., 2019) may not. Specifically, we analyze four cognitive behaviors: thinking, reasoning, planning, and feedback learning, leaving the examination of their interrelationships for future research.

4.1 Thinking, Fast And Slow

At the core of cognitive behavior lies thinking. The Kahneman's framework (Kahneman, 2011) divides thinking into two distinct systems: the fast system operates through intuitive shortcuts for quick navigation of daily situations without extensive analysis. Conversely, the slow system, or System 2, involves conscious, detailed and methodical examination of information, necessitating logical deliberation to arrive at decisions and address challenges.

Fast Thinking via Task-specific Channels Using channels trained through task-specific supervised learning can achieve performances that rival or exceed human performance. Nonetheless, they often struggle with generalizing to data from natural domain shifts, adversarial perturbations and debiased data, as summarized by Li et al. (2023). This limitation is consistently attributed to shortcut learning, such as classifying sentences containing the word "No" as "contradiction" in text entailment tasks (Wallace et al., 2019; Du et al., 2021). The intriguing question arises whether task-specific channels can also develop System 2 — the

Behaviors	Context	Relevant Works
Reasoning	CoT triggers, e.g., “Let’s think step by step.”	Zero-shot CoTs (Kojima et al., 2022), Auto-CoTs (Zhang et al., 2023)
	Few-shot demos with CoTs	Few-shot CoTs (Wei et al., 2022b), CoTs-SC (Wang et al., 2023b), Auto-prompt (Zhang et al., 2023), ToT (Yao et al., 2023a)
Planning	Zero-shot instruction	Wang et al. (2023a)
	Few-shot demos with planning steps	Huang et al. (2022)
Feedback Learning	Observations from external environments	Reflexion (Shinn et al., 2023)
	Outputs from LLM-Profiled Evaluators	Self-refine (Madaan et al., 2023), Reflexion (Shinn et al., 2023), RAP (Hao et al., 2023)
	Feedback from Tools	Guan et al. (2023), CRITIC (Gou et al., 2024)

Table 2: Cognitive behaviors enabled by free-form context. For the “Feedback Learning” sections, we illustrate the contexts utilized to produce feedback. It’s worth noting that the methods for feedback adaptation might not always employ free-form context; for instance, they may involve advanced search techniques as outlined in our study. The final column presents examples of tasks for demonstration purposes, though the list is not comprehensive.

fast system. While the limited expressiveness of task-specific outputs does not offer straightforward evidence, Li and Liu (2023) employ a technical probe (Sundararajan et al., 2017) to reveal that indulgence in shortcut learning during task-specific training impedes the development of the slow system. While the mentioned research primarily examines the LLM fine-tuning paradigm, it’s our contention that shortcut learning and the fast thinking are likely prevalent across all the parametric channels, including prefixes and adapters, trained on supervised datasets to some degree. This is attributed to the inherent characteristics of gradient descent optimization, as demonstrated by empirical findings in Li and Liu (2023). Another empirical evidence shows that methods like prefix and adapter tuning, although more resilient, still notably falter under distribution shifts and adversarial attacks (Han et al., 2021; Yang and Liu, 2022). The mitigated impact observed in prefix and adapter tuning is attributed to the fact that the underlying LLMs are not directly engaged as task-specific channels, as explored by (Han et al., 2021). While we draw parallels between reliance on shortcuts and fast thinking within human cognition, some research within the NLP field argues that such dependency on shortcuts (dataset biases) detracts from the models’ relevance to human-level cognition (Zhong et al., 2023). This perspective arises from the view that the shortcuts might not reflect genuine human cognitive activities within the field of NLP.

Minimal Fast Thinking Evident with AR-LLMs Prompting Research findings (Si et al., 2023; Zhang et al., 2022) consistently indicate the difficulty of inducing fast thinking in AR-LLMs through prompting techniques. These models typically remain unfazed by various distributional shifts, such as domain shift and adversarial perturbations. Min et al. (2022) demonstrate that, even with few-shot demonstrations for in-context learning, the models tend to leverage the structure of these demonstrations to organize the generation rather than relying on simplistic input-to-label mappings for predictions. Additionally, Raman et al. (2023) show that PET prompting improve the AE-LLMs’ ability to withstand adversarial attacks. Nonetheless, this enhanced robustness is somewhat restricted. The constrained effectiveness could be attributed to the dependency on task-specific channels inherent during the deployment of the PET prompting.

Slow Thinking in Prompting Paradigms The remainder of this section will illustrate the capacity of AR-LLMs’ prompting to replicate the human slow thinking process through the exhibition of effortful mental activities, as encapsulated in Table 2.

4.2 Reasoning

Reasoning is a thinking process to conclusions or decisions with the sequential and interconnected nature, i.e., chain-of-thoughts (CoTs) (Wei et al., 2022b). This is the most common definition in the NLP/LLM are to investigate the LLMs’ reasoning ability. With a reasoning path in free-form modality, models can better solve complicated tasks requiring multi-step reasoning compared to the conclusion without CoTs. As an illustration, Wei et al. (2022b) substantially boosts model efficacy in solving mathematical reasoning bechmarks.

Reasoning is defined as the process of arriving at conclusions or decisions through a sequential and interconnected series of thoughts, often referred to as a chain-of-thoughts (CoTs) (Wei et al., 2022b). This definition is widely accepted in the field of Natural Language Processing (NLP) for exploring the reasoning capabilities of LLMs. By employing a reasoning path via the modality of free-form text, models are more adept at tackling complex tasks that necessitate multi-step reasoning, as opposed to reaching conclusions without the aid of CoTs. Technically, the auto-regressive nature employs the thoughts or intermediate steps generated as the prior for generating subsequent thoughts and, ultimately, the final predictions.

Context for Eliciting Reasoning Two primary contexts are employed to facilitate the creation of intermediate reasoning steps: incorporating a Chain of Thought (CoT) triggers in task instructions (**zero-shot CoTs**), such as "Let's think step-by-step" (Kojima et al., 2022), within prompts, or integrating manually crafted reasoning steps in a few-shot learning context (**few-shot CoTs**) (Wei et al., 2022b). To circumvent the manual compilation of few-shot demonstrations with reasoning sequences, Zhang et al. (2023) developed a method to automatically generate few-shot demonstrations by choosing several queries and utilizing zero-shot CoTs to craft reasoning sequences for each query (**Auto CoTs**). Given that simple greedy decoding (producing a single chain) is prone to error accumulation in intermediate steps, Wang et al. (2023b) propose generating multiple chains and consolidating them through majority voting, thereby enhancing model accuracy in both scenarios (**CoTs-SC**).

4.3 Planning

Planning involves the forethought and organization of actions or steps to achieve a predetermined objective. This process fundamentally requires a comprehension or representation of the environment and involves breaking down tasks into smaller, manageable subgoals. It represents a key cognitive behavior modeled within the fields of AI. Typical planning methods break down tasks into subgoals through explicit symbolic representation (Russell and Norvig, 2010). For instance, partial-order planning ensures the logical sequencing of actions by modeling actions, preconditions, effects, and the relations among actions in such a way that actions are logically sequenced to meet the goal's preconditions. Differing from traditional approaches that rely on explicitly modeled knowledge and reasoning mechanisms, LLMs leverage their inherent knowledge and inferential capabilities to mimic planning. They do this by producing text sequences that suggest a logical progression of steps or actions directed towards an objective (Hao et al., 2023; Wang et al., 2023a; Huang et al., 2022). This skill stems from the models' proficiency in forecasting the subsequent most likely word sequence based on a context indicative of planning or reasoning processes.

Context to Elicit Plans Similar to the activation of reasoning processes, the process of planning can be prompted through the inclusion of specific planning cues in zero-shot scenarios, such as the prompt "let's carry out the plan" (Wang et al., 2023a), or through the demonstration of planning steps in few-shot examples (Huang et al., 2022). Experimental findings indicate that instructions tailored to tasks significantly enhance the performance of LLMs on various tasks. For instance, directives like "pay attention to calculation" (Hao et al., 2023) or "identify key variables and their corresponding figures to formulate a plan" (Wang et al., 2023a) have been shown to improve outcomes in tasks requiring numerical reasoning.

Applying Planning for Sequential Decision-making This ability is essential for addressing problems requiring a series of decisions, especially when deploying LLMs in open-world scenarios like robotics. In such environments, tasks typically need physical actions (grounded), involve translating broad objectives into actionable steps (high-level), and present a vast range of possible actions (open-ended). Research has demonstrated the effectiveness of LLMs in deconstructing complex goals into actionable sequences within such dynamic environments, as seen in projects like ALFWorld (Yao et al., 2023b), VirtualHome (Huang et al., 2022), and Minecraft (Wang et al., 2023c). An example from ALFWorld illustrates this: achieving the objective of "examining paper under desklamp" necessitates LLMs to devise practical plans (e.g., initially approaching the coffee table, then acquiring the paper and utilizing the desklamp) and subsequently generate textual instructions for execution in real-world settings.

4.4 Feedback Learning

As Kahneman (2011) elucidates, although System 1 may rush to judgments that are biased or erroneous, System 2 has the capacity to identify and rectify these mistakes through introspection on the rapid decisions made by System 1. Similarly, LLMs have shown the ability to mimic this aspect of human cognition.

Feedback Sources There are different sources of feedback: 1) Feedback from LLM-profiled evaluators: In such cases, LLM-profiled evaluators can give feedback on previous generations. The evaluators are normally prompted to follow certain evaluation metrics, such as determining the relevance of a sub-question to the original question requiring intricate, multi-step reasoning (Hao et al., 2023). An example prompt could be: "Given a question, assess if the subquestion aids in solving the original question. Answer 'Yes' or 'No'. Question: {goal}; Subquestion: {action}. Is the subquestion useful?". The generated feedback would be appended for LLM actors to re-generate answers. 2) Feedback from task-specific environments, e.g., (simulated) embodied environments (Shinn et al., 2023). 3) Feedback from tools, e.g., error messages from Python interpreters (Gou et al., 2024). Typically, raw feedback originating from external environments and tools undergoes a process of refinement by LLM evaluators prior to being presented to LLM actors. In the work by Yao et al. (2023b), LLMs engaging with a Wikipedia API to search for entities that do not exist, such as "Search[goddess frigg]", may encounter a 404 error, delivered in JSON format. In response, an LLM evaluator can articulate feedback about the error related to their action, such as stating, "Could not find goddess frigg.".

5 Future Work: Autonomous Cognitive Behaviors

Instead of relying on explicit contextual cues to trigger advanced cognitive functions, an intelligent system is expected to independently engage in reasoning, planning, and decision-making as it interacts with the external world—for instance, by seeking input from humans or utilizing available tools. To foster such autonomous behaviors, various algorithms aim to tune LLMs for independently exhibiting behaviors that align with human cognitive processes. For instance, Liu et al., Liu et al. (2023) have developed techniques for instruction tuning that facilitates autonomous reasoning. Yet, the challenge remains in creating instructional data that encapsulates higher-order cognitive functions. A pivotal question emerges: *How can various cognitive behaviors be encapsulated within free-form text (instruction data)?* Addressing this question is crucial for ensuring that the data used for tuning mirrors human cognitive processes, thereby making the resulting model actions more human-like. Unraveling this issue might necessitate insights from both cognitive psychology and linguistics. Another approach to tuning involves the use of reliable reward models, such as reinforcement learning from human feedback (RLHF) (Ouyang et al., 2022) and behavior cloning (Nakano et al., 2021). Many studies (Ouyang et al., 2022; Nakano et al., 2021) develop reward models based on comparisons of model-generated responses, with human evaluators ranking these responses. An unresolved inquiry remains: *How can reward models be devised to truly reflect human cognitive preferences?*

6 Conclusion

In summary, our survey seeks to inspire further research in AI, within the domain of language intelligence and beyond, to move away from heavily optimized task-specific channels. Instead, we advocate for the adoption of natural and free-form modalities throughout the pretraining phase via self-supervised learning, followed by straightforward inference-time deployment that eschews the necessity for mathematically optimizing task-specific channels. We developed an analytical framework to examine the deployment of LLMs to reach the conclusion. Besides, the auto-regressive nature of free-form modalities, leveraged during pretraining, enhances the capacity for exhibiting a range of human-like cognitive behaviors by utilizing the free-form channel. It is important to clarify that we do not advocate that LLMs possess conscious thought. Rather, our findings illustrate how LLMs, such as ChatGPT, can imitate the outcomes of human cognitive activities via the free-form modality given suitable verbal context.

Limitations

This work acknowledges an omission of a significant deployment strategy: the utilization of multi-agent systems, as reviewed by Wang et al. (2024).

However, the prompting paradigm of AR-LLMs and the cognitive behaviors encapsulated herein serve as pivotal building blocks for LLM-based agents. Instances include the integration of LLM-profiled planners in recent studies by Huang et al. (2022); Wang et al. (2023c); Dasgupta et al. (2022); Wang et al. (2023a), alongside the formulation of feedback-learning workflows by Shinn et al. (2023); Gou et al. (2024).

References

Christoph Bartneck, Tony Belpaeme, Friederike Eyssel, Takayuki Kanda, Merel Keijsers, and Selma Šabanović. 2020. *Human-robot interaction: An introduction*. Cambridge University Press.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *JMLR*, 3:1137–1155.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in NIPS*, volume 33, pages 1877–1901.

Ishita Dasgupta, Christine Kaeser-Chen, Kenneth Marino, Arun Ahuja, Sheila Babayan, Felix Hill, and Rob Fergus. 2022. Collaborating with language models for embodied reasoning. In *Second Workshop on Language and Reinforcement Learning*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Mengnan Du, Varun Manjunatha, Rajiv Jain, Ruchi Deshpande, Franck Dernoncourt, Jiuxiang Gu, Tong Sun, and Xia Hu. 2021. Towards interpreting and mitigating shortcut learning behavior of NLU models. In *NAACL-HLT*, pages 915–929.

Zhibin Gou, Zhihong Shao, Yeyun Gong, yelong shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2024. CRITIC: Large language models can self-correct with tool-interactive critiquing. In *The Twelfth International Conference on Learning Representations*.

Lin Guan, Karthik Valmeekam, Sarath Sreedharan, and Subbarao Kambhampati. 2023. Leveraging pre-trained large language models to construct and utilize world models for model-based task planning. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Wenjuan Han, Bo Pang, and Ying Nian Wu. 2021. Robust transfer learning with pretrained language models through adapters. In *ACL-IJCNLP*, pages 854–861.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Hong, Zhen Wang, Daisy Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. In *EMNLP*, pages 8154–8173.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *ICML*, pages 2790–2799.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. 2022. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *ICML*, pages 9118–9147.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *AAAI*, volume 34, pages 8018–8025.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *TACL*, 8:64–77.

D. Kahneman. 2011. *Thinking, Fast and Slow*. Farrar, Straus and Giroux.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in NIPS*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*, pages 7871–7880.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *ACL-IJCNLP*.

Xinzhe Li and Ming Liu. 2023. Make text unlearnable: Exploiting effective patterns to protect personal data. In *TrustNLP*, pages 249–259.

Xinzhe Li, Ming Liu, Shang Gao, and Wray Buntine. 2023. A survey on out-of-distribution evaluation of neural nlp models. In *IJCAI-23*.

Hanmeng Liu, Zhiyang Teng, Leyang Cui, Chaoli Zhang, Qiji Zhou, and Yue Zhang. 2023. Logicot: Logical chain-of-thought instruction tuning. In *EMNLP*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback. In *Advances in NIPS*.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? In *EMNLP*, pages 11048–11064.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. 2021. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Gray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in NIPS*.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Mrigank Raman, Pratyush Maini, J Kolter, Zachary Lipton, and Danish Pruthi. 2023. Model-tuning via prompts makes NLP models adversarially robust. In *EMNLP*, pages 9266–9286.

Stuart J Russell and Peter Norvig. 2010. *Artificial intelligence a modern approach*. London.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *EACL*, pages 255–269.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *EMNLP*, pages 4222–4235.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik R Narasimhan, and Shunyu Yao. 2023. Reflexion: language agents with verbal reinforcement learning. In *Advances in NIPS*.

Chenglei Si, Zhe Gan, Zhengyuan Yang, Shuohang Wang, Jianfeng Wang, Jordan Lee Boyd-Graber, and Lijuan Wang. 2023. Prompting GPT-3 to be reliable. In *ICLR*.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *ICML*, pages 3319–3328.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing nlp. In *EMNLP-IJCNLP*, pages 2153–2162.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR*. OpenReview.net.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):1–26.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023a. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *ACL*, pages 2609–2634.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023b. Self-consistency improves chain of thought reasoning in language models. In *ICLR*.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Kuntal Kumar Pal, Maitreya Patel, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Savan Doshi, Shailaja Keyur Sampat, Siddhartha Mishra, Sujan Reddy A, Sumanta Patro, Tanay Dixit, and Xudong Shen. 2022. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In *EMNLP*, pages 5085–5109.

Zihao Wang, Shaofei Cai, Guanzhou Chen, Anji Liu, Xiaojian Ma, and Yitao Liang. 2023c. Describe, explain, plan and select: Interactive planning with LLMs enables open-world multi-task agents. In *Advances in NIPS*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022a. Emergent abilities of large language models. *TMLR*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022b. Chain of thought prompting elicits reasoning in large language models. In *Advances in NIPS*.

Zonghan Yang and Yang Liu. 2022. On robust prefix-tuning for text classification. In *ICLR*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik R Narasimhan. 2023a. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in NIPS*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023b. React: Synergizing reasoning and acting in language models. In *ICLR*.

Hongxin Zhang, Yanzhe Zhang, Ruiyi Zhang, and Diyi Yang. 2022. Robustness of demonstration-based learning under limited data scenario. In *EMNLP*, pages 1769–1782.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2023. Automatic chain of thought prompting in large language models. In *ICLR*.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. 2023. Agieval: A human-centric benchmark for evaluating foundation models. *arXiv preprint arXiv:2304.06364*.