Large Language Models and Games: A Survey and Roadmap

Roberto Gallotta[1] *Graduate Student Member, IEEE*, Graham Todd[2] *Graduate Student Member, IEEE*, Marvin Zammit[1] *Graduate Student Member, IEEE*, Sam Earle[2] *Graduate Student Member, IEEE*, Antonios Liapis[1] *Member, IEEE* Julian Togelius[2] *Senior Member, IEEE,* and Georgios N. Yannakakis[1], *Fellow, IEEE*
[1]: Institute of Digital Games, University of Malta, Msida, Malta
[2]: Tandon School of Engineering, New York University, New York, USA
roberto.gallotta@um.edu.mt, gdrtodd@nyu.edu, marvin.zammit@um.edu.mt, se2161@nyu.edu, antonios.liapis@um.edu.mt, julian@togelius.com, georgios.yannakakis@um.edu.mt

Abstract—Recent years have seen an explosive increase in research on large language models (LLMs), and accompanying public engagement on the topic. While starting as a niche area within natural language processing, LLMs have shown remarkable potential across a broad range of applications and domains, including games. This paper surveys the current state of the art across the various applications of LLMs *in* and *for* games, and identifies the different roles LLMs can take within a game. Importantly, we discuss underexplored areas and promising directions for future uses of LLMs in games and we reconcile the potential and limitations of LLMs within the games domain. As the first comprehensive survey and roadmap at the intersection of LLMs and games, we are hopeful that this paper will serve as the basis for groundbreaking research and innovation in this exciting new field.

Index Terms—Large Language Models, Digital Games, Video Games, Survey, Generative Text, Gameplaying, Procedural Content Generation, Generative AI.

1 Introduction

Five years ago, autoregressive language modeling was a somewhat niche topic within natural language processing. Training models to simply predict text based on existing text was considered of primarily theoretical interest, although it might have applications as writing support systems. This changed drastically in 2019 when the Generative Pre-trained Transformer 2 (GPT-2) model was released [1]. GPT-2 demonstrated convincingly that transformer models trained on large text corpora could not only generate surprisingly high-quality and coherent text, but also that text generation could be controlled by carefully prompting the model. While not the first autoregressive model [2], [3], GPT-2 was the first of the "large" models, and as such we use it here as cutoff mark (see also Section 2). Subsequent developments, including larger models, instruction fine-tuning, reinforcement learning from human feedback [4], and the combination of these features in ChatGPT in late 2022, turbocharged interest in large language models (LLMs). Capabilities of LLMs were seemingly unbounded, as long as both problem and solution could be formulated as text.

LLMs are currently a very active research field. Researchers are focused on improving the capabilities of LLMs while reducing their compute and memory footprint, but also on understanding and learning to harness the capabilities of existing LLMs. Informed opinions on the ultimate capabilities of LLM technology vary widely, from the enthusiastic [5] to the pessimistic [6], [7]. Our aim is to approach the topic from somewhere in-between these two perspectives: optimistic with respect to the potential of LLMs and realistic with respect to their technical, theoretical, and ethical shortcomings.

Games, including board games and video games, serve both as a source of important benchmarks for research in Artificial Intelligence (AI) and as an important application area for AI techniques [8]. Almost every game utilizes some kind of AI technology, and we are currently in an exploratory phase where both developers and researchers try to figure out how to best make use of recent advances in this field [9].

In this paper, we set out to chart the impact LLMs have had on games and games research, and the impact they are likely to have in the near- to mid-term future. We survey existing work from both academia and (mostly independent) game creators that use LLMs *with* and *for* games. This paper does not set out to capture modern advances in LLM technology or algorithms for training LLMs. Not only do such resources exist [10], but the breakneck speed of technical advances in this field will likely make our writeup obsolete in a year or so. Instead, we focus on work that leverages LLMs in games and propose a range of roles that the LLM can take in the broader ecosystem of games (both within the game and beyond). We lay out promising future directions for efforts to use LLMs in games, and discuss limitations (both technical and ethical) that should be addressed for a brighter future of LLM research in games.

It is important to note that this survey emerges from the top down, based on our expertise in AI and games [8], and extensive work on most topics covered by this paper. The focus of the paper, in Section 3, is built from our own

typology and supported where possible by academic and non-academic work. While a bottom-up approach via e.g. keyword search through general paper repositories is valuable [11], [12], [13], the process would lead to a very different type of paper. Indicatively, this approach was followed by Yang *et al.* [14] who investigated current uses of GPT models in video games by searching articles in ACM, IEEE Xplore, Springer, and AAAI with keywords "game" and "GPT". Instead, we have attempted to conduct a comprehensive manual review of all recent proceedings from the major conferences in AI and games[1], and the IEEE Transactions on Games for work relevant to the themes of this paper.

2 A Note on Terminology

This paper concerns the intersection between games (board games, video games, or other), and large language models. But what exactly is an LLM?

Broadly speaking, an LLM is a model that is trained on text in order to be able to reproduce text in response to other text. But this definition is overly broad, as it would include Shannon's original n-gram models from 1946 [15], rudimentary recurrent neural networks from the early 1990s [16], and the Tegic T9 text prediction system that would help you write text messages on your Nokia 3210.

What distinguishes LLMs from other text generative models is mainly that they are *large*. But which model size is considered large enough? In 2019 emerging models such as BERT [2] and ERNIE [3] showcased significant advances in language modeling, but LLMs became a well-recognized term with the introduction of OpenAI's GPT-2 [1], whose various versions have between 117 millions and 1.5 billions of parameters. Because of the association between the term LLM and the GPT-class of models, we will use the size of GPT-2 as a soft cutoff on the type of models we consider LLMs; we are concerned with models of few hundred million parameters or more. Each subsequent iteration of the GPT family features an increased number of parameters, and larger and more diverse training corpora.

Another distinct trait of LLMs is their architecture. While language models could in principle be based on various architectures, including Long Short-Term Memory (LSTM) networks [17], the current LLM landscape is dominated by variants of the transformer architecture, a type of neural network introduced in 2017 [18]. This model became very influential because of what was perceived as a quantum leap in output quality compared to previous models. In this survey we rely primarily on LLMs employing this architectural basis.

The last feature of LLMs we consider is their versatility across a wide range of tasks with minimal or no fine-tuning or retraining. This capability represents a significant shift; since the release of GPT-3.5, LLMs have evolved from primarily autoregressive predictive text models to pre-trained, general-purpose conversational models.

It is important to note that LLMs are by no means limited to the GPT family of models. There is by now a large variety of LLMs of varying size and capabilities, including open-source models such as Mistral [19] and the Llama [20] family, which can be fine-tuned, run locally, and even be embedded in games' runtimes.

One could also argue that the definition is somewhat narrow, as many modern LLMs are multimodal models, meaning that they can take as input and/or produce as output modalities other than text. In particular, many modern LLMs can process and produce images. This is often achieved through combining the core transformer network with a visual encoder network for input and a latent diffusion model for output. Examples include GPT-4V [21] and the open-source Llava [22]. In this paper, we consider large multimodal models (LMMs) [21] as long as they retain their ability to both consume and produce text.

This survey will not concern itself with AI and machine learning techniques that are not LLMs as defined above. In particular, we will not be covering the large literature on game playing and content generation using machine learning methods [23] that does not use textual input and output. We will, however, occasionally mention some of that work where relevant, in particular to help provide historical context.

3 Roles of LLMs in Games

Past attempts at a typology for AI in games focused on three roles the AI can take in a game: to play a game, to design a game, or to model the (human) players [8]. LLMs are typically presented as conversational agents, which often invites the public to give them anthropomorphic qualities —such as reasoning and creativity. We follow these trends when considering the roles an LLM can be called to play within the game or within the game development process. An LLM can operate within the game as a player (replacing a human player while imitating their goals), as a non-player character such as an enemy or interlocutor, as an assistant providing hints or handling menial tasks for a human player, as a Game Master controlling the flow of the game, or hidden within the games' ruleset (controlling a mechanic of the game). There are however other roles an LLM can play outside of the game's runtime, such as a designer for the game (replacing or assisting a human designer) or as analyst of the gameplay data of the playerbase. Finally, the LLM can interface with a player or an audience in different ways, acting as a commentator of an ongoing play session (during runtime) or a reteller of past game events in some narrative form (outside runtime). Some of these roles (autonomous player, autonomous designer) are prominent in the broader AI and games research [8] and LLM research has targeted them extensively, while some of the other roles have been toyed with in exploratory research. The following sections present the roles themselves, surveying research undertaken for each role, while we identify gaps and opportunities for future research in Section 4.

3.1 Player

How can an LLM play a game? Fundamentally, LLM players require some transformation from their typical output space (i.e. sequences of tokens) into the input space of the game. In addition, aspects of the game and its current state must

1. IEEE Conference on Games, Foundations of Digital Games conference, Artificial Intelligence and Interactive Digital Entertainment conference.

be provided to the LLM in some form in order for it to play at a reasonable level. Depending on the game itself, these mappings might be intuitive or complex. We identify three general classes of games to which LLM players are well suited: (a) games where states and actions can be compactly represented as sequences of abstract tokens, (b) games where the main input and output modalities are natural language, and (c) games for which external programs can control player actions via an API.

The first class of games mostly includes turn-based board games (e.g. *Chess*), since the discrete set of board positions and moves is more easily transformed into a compact representation (e.g. Portable Game Notation [24]) than, for instance, a first-person shooter. By tokenizing sequences of moves taken from a game database, the problem of action selection can then be mapped to the standard autoregressive learning objective on which LLMs are trained —predicting the next move given the context of those that preceded it. *Chess* [25], [26], [27], *Go* [28], and *Othello* [29] have all been used as testbeds for LLM players in this way. This approach allows even more complex game states to be reasoned upon by an LLM player. In Bateni and Whitehead's work [30], for example, the LLM plays the popular video game *Slay the Spire* (Mega Crit, 2017), understanding synergies between cards based solely on their description and adapting to changes in gameplay rules. However, board games are not the only kind of game that can be represented as token sequences: the generalist GATO [31] agent can play a variety of Atari games at human or near-human levels by processing visual inputs as sequences of pixel values in raster order. Pixel values are interleaved with separator tokens and previous actions, allowing the model to accurately predict the appropriate game action in a dataset of human play traces. It is possible that continued improvement in transformer models that capture both spatial and visual dynamics [32], [33] could allow for a similar approach to scale to even more complex games. However, such approaches require large datasets of gameplay videos that may be comparatively more difficult to collect. In addition, we note that reliance on human gameplay traces as the basis for learning may make it more difficult for an LLM player to reach superhuman performance without leaps in terms of reasoning and generalization (see Section 5).

The second class of games most obviously includes text adventure games such as *Zork* (Infocom, 1977), where game states are presented as natural language descriptions and the game is already equipped with a parser to handle natural language responses. This means that LLMs can be queried for game actions in a way that still leverages their large-scale pre-training on natural language text. The earliest application of LLMs to these kinds of text games is CALM [34], a GPT-2 system finetuned on a dataset of human gameplay transcripts collected from a variety of text adventure games. The model is trained to predict the natural language string provided by human players given the context of previous states, actions and information about the avatar (e.g. their inventory). To actually play a game, the trained language model generates multiple candidate actions and deep reinforcement learning (RL) is used to optimize a policy that selects actions from among the candidates. At the time of its publication, this RL component was necessary because the LLM alone was not capable of generalizing well to unseen games or situations [34]. However, a more recent investigation of ChatGPT as a *Zork* player has indicated that LLM performance is improving [35]. In a preliminary experiment, Yao *et al.* [34] show that the performance of ChatGPT can approach that of existing algorithms for text game playing, as long as a human interlocutor remains in the loop to assist the model (e.g. by reminding it of actions it has already tried). However, there is obviously much room for improvement in directly applying LLMs to text games in this way. Additionally, the ability for LLMs to play entirely novel, niche, or unseen text games (especially important given the likelihood that such systems encounter walkthroughs or playtraces of popular text games during their training) remains largely unexplored.

In a similar vein, inductive biases from large language models can be applied to help guide the policies of agents trained with other methods. For instance, the GALAD system [36] uses a pre-trained LLM to guide an agent towards morally acceptable actions in text games from the Jericho suite [37], while the MOTIF system [38] learns an intrinsic reward function for *NetHack* NetHack, 1987) by mining preferences between game states from an LLM.

Text adventure games are not the only cases where natural language input and output are used for playing: many board games operate via player negotiation. *CICERO* [39] leveraged LLMs for playing the deal-making and subterfuge game *Diplomacy* (Avalon Hill Games, 1976). CICERO builds from a pre-trained LLM and is fine-tuned on a large corpus of *Diplomacy* transcripts. Throughout the game, samples from the model are sent to other players and the various dialogue transcripts are collected to condition the potential action. CICERO is further trained to condition its outputs on specific game intents (inferred from the transcripts and added as additional context during training). In order to select an action, CICERO uses a "strategic reasoning module" that predicts the actions of other players, using a value and policy function learned from self-play. *Diplomacy* is an interesting game in part because the action space is split between natural language utterances and a more standard set of moves on a discrete game board, and CICERO demonstrates how an LLM can be integrated as part of a larger system for high-level play.

Finally, we consider games for which a robust API exists. This is less of a *kind* of game in the sense of its style or mechanics, and more a fact about its popularity or its ease of implementation. An API is an important attribute because it allows LLMs to act as players not by directly generating actions, but by producing *programs* that act as *policies*. Improvements in the code generation abilities of LLMs have allowed them to write small programs that can produce actions given game states without further intervention from the model. For instance, the VOYAGER system [40] leverages the code generation abilities of GPT-4 to play *Minecraft* (Mojang Studios, 2011) by interacting with the popular Mineflayer API. Using a sophisticated chain of prompts, VOYAGER generates blocks of code that leverage calls to the API in order to execute high-level "skills" (e.g. "Attack nearest zombie") that are automatically converted into low-level game inputs (e.g. mouse movements and key presses). GPT-4 is also used as a high-level goal generator

Fig. 1. Screenshot of the promotional video for *AI people*, where a player can interact via their avatar's chat with other NPCs and watch how their text has consequences between NPC relationships, in this case. Used with permission from [41].

and planner, which in turn informs the code generation. This approach proved to be very successful, with VOYAGER being the first automated system to complete a variety of in-game *Minecraft* challenges. The results are impressive and indicate that generating action-producing programs may be a more efficient way to leverage latent LLM knowledge than direct action sampling. However, VOYAGER does benefit substantially from the availability of a robust API and vast amounts of internet discussions around *Minecraft*. As with the analysis of ChatGPT on *Zork*, the ability of this approach to generalize to less popular or entirely unseen games remains to be seen.

3.2 Non-Player Characters

Non-player characters (NPCs) are agents which exist in virtual game worlds but whose actions are not directly controlled by the players. NPCs exist to enrich the player's experience and deepen immersion by adding to the world's ambiance and making it more believable [42]. NPCs may serve as pets, allies, enemies[2], merchants, quest givers, or bystanders. Therefore, they have different agency even from AI-controlled players, and their goal is never to win. This makes designing AI for NPCs interesting [8], and LLMs are uniquely advantaged in this task since they can "understand" gameworld settings and adapt their responses accordingly. It has been shown that LLMs are able to role-play through different scenarios [43], [44], thereby highlighting their potential to provide a more flexible and apt tool to emulate human behavior. We identify two ways in which LLMs can control NPCs: (a) through their *dialogue*, and (b) through their *behavior*. Behavior relates to in-game action selection, discussed in Section 3.1; however, we note that the heuristics and goals of such behavior is different than an AI player trying to win the game.

LLMs are naturally suited for natural language conversation, and as NPC dialogue systems they can generate dynamic and contextually appropriate responses based on player input [45], [46], [47], [48]. This makes interactions with NPCs more engaging and realistic, reduces repetitive discourse and provides a more explorative experience within the game [49]. LLMs can engage the players in the gameworld's narrative as *foreground* NPCs, *background* NPCs, or *narrator* NPCs. We discuss narrator LLMs as commentators in Section 3.4 whereas we cover the other two NPC types here. Foreground NPCs form part of the overarching narrative of the game, or one of its sub-narratives. They may be enemies, allies, information-givers, quest-givers, or item-providers. Their dialogue is heavily constrained by the scope of the narrative, their role within it, and the player actions. Foreground NPCs' text generation process via LLMs must consider the overall context of the game and the interaction with the player, and keep track of events transpiring in the playthrough. This raises concerns regarding the memory capacity of LLMs, as well as the impact of possible hallucinations, i.e. plausible but false statements [50]. We revisit these limitations in Section 5. The purpose of background NPCs is to make the environment more believable and act independently of the players [49]. Such NPCs' presence is purely decorative and their dialogue is essentially small talk. Thus their dialogue generation is less constrained, perhaps bound only by the identity of the speaker and their background. That said, their believability hinges on their ability to maintain the illusion that they have their own agency in the world and can interact with it [51]. Park *et al.* [45] explored NPC interactions within limited environments, where a number of LLM-based agents simulated social interactions in a sandbox environment. Within the constraints of their environment and the social affordances, the agents behaved in a believable manner, following their goals, planning for new ones, and even recalling past events as they interacted.

Other studies have shown that multiple LLM-based agents are able to follow game rules and engage in game playing [52], [53], with different models consistently exhibiting their own aptitudes and weaknesses when applied to specific roles. This ability to interact within constraints is useful to instill believable behaviors in foreground and background NPCs, grounding their actions and dialogue within the rules of the game environment. Some work has focused more on the conversational and story-writing abilities of LLMs, such as the creation of dialogue between multiple characters, each having their unique personality, whilst following a consistent plot. One such example is the use of LLMs to generate a *South Park* (Comedy Central, 1997) episode [54] with multiple characters within a well-known setting. There are limitations to this approach, primarily that LLMs perform something like a theatrical improvisation, rather than acting as an actor studying a part [43]. Through this unconstrained process the LLM is prone to hallucinations which do not fit the desired scenario. This volatility can be mitigated by providing the LLM not only with the conversation history but also with the current state of the environment, such as the items within it and their affordances, as well as the other characters and their corresponding actions. Urbanek *et al.* [55] used a configurable multi-user dungeon text-based environment in a fantasy setting to allow for both human and language model-based players. The latter were able to use an updated game state, including descriptions of local environments,

2. We use enemies and allies here for in-game agents with different skills and ways to affect the world than e.g. game playing AI such as opponents in *Chess*.

objects, and characters, to take better actions and engage in coherent dialogue. This approach may also be extended to other scenarios or to cover the use of LLMs as active or interactive narrators. Ubisoft showcased LLM-based NPCs in their Neo NPC demo [48], where the player can freely converse with the in-game characters. Each NPC was given a carefully hand-crafted persona, and all NPCs could respond within the constraints of the game narrative and their prescribed personality whilst generating realistic responses. The players were able to engage in game-specific activities with these NPCs, such as planning a heist, or even attempt an unrelated course of dialogue altogether. Care was taken to minimize toxicity and social biases inherent in the LLMs through the prompt defining each NPC, which also had a bearing on how the latter would react to any player's offensive or unruly discourse.

3.3 Player Assistant

A somewhat less explored role for LLMs in games is that of a player assistant: an interactive agent intended to enrich or guide the player experience in some way. This could be in the form of a sequence of tutorial-style tips, a character that does not causally interact with the game world at all, or an agent able to interact within the game environment at a similar level as the player. Existing games make use of player assistants in different ways. For example, in *The Sims* (Electronic Arts, 2000) a disembodied assistant provides tips specific to the game context via dialogue boxes. *Civilization VI* (Firaxis Games, 2016) uses different assistants (with different portraits) to suggest the best build option according to their idiosyncratic heuristic, alleviating some decision-making from the player. In management games, AI may automate menial tasks such as assigning jobs to a planet's population in *Stellaris* (Paradox Interactive, 2016); this assistance reduces cognitive load from the player, but the player can always micro-manage this task if they wish.

LLMs are appealing as player assistants given their expressive and conversational capacities. An LLM-based player assistant could plausibly choose an action to suggest to the player, and—more importantly—form the explanation for this suggestion as a natural language utterance delivered by a disembodied or embodied agent. This utterance could even be accompanied by a corresponding sentiment, and manifested through the assistant's body stance, gestures and facial expression (in the case of an embodied assistant). The choice of action to suggest to the player could be based on either LLM-powered or heuristic-based methods for finding the best policy or action given the current game context (see Section 3.1). While not intended as a player assistant, this type of exposition is realized by embodied agents in *AI people* [41] (see Section 3.7). Similarly, LLMs may assist the player by undertaking some minor tasks in the game via a tailored smaller role as "player" within that smaller task description (see Section 3.1). The LLM could extrapolate the policy for such a minor task through conversation with the player, parsing the natural language chat similar to [56].

A special case of player assistant comes from an "inner voice", uttered by the player's own avatar. Hints given in the player avatar's own voice are a trope in classic point-and-click adventure games, such as Guybrush Threepwood commenting "A rubber chicken with a pulley in the middle. . . What possible use could that have?" when picking up the item in *The Secret of Monkey Island* (Lucasfilm Games, 1990). This specific concept of "inner voice" was explored as an application of LLMs in the work of Rist [57], although the LLM's freedom was limited. Using a hand-crafted game environment with pre-scripted events and location-based triggers meant to occur when the player would either interact with the world or observe something, Rist authored where the LLM is prompted to generate short comments in different styles (e.g. in neutral or sarcastic tone) when certain predefined in-game events are met. These comments are generated based on text descriptions of what the user sees and can do in that moment. The LLM still requires human-authored knowledge (e.g. *where* the commentary occurs, *what* the hints are, and *why* in terms of designer goals), and is thus not a fully-autonomous player assistant.

Despite the work of Rist [57], which focused more on immersion than assistance, the potential of LLM-powered player assistants is not explored in current research. We highlight the potential of this application in Section 4.

3.4 Commentator/Reteller

LLMs are also ideally suited as commentators or retellers. Here, we identify these roles as an agent that produces and narrates a sequence of events, for the benefit of either human players or spectators. Such an agent may consider only in-game events and in-game context, acting as an in-game entity such as a sports commentator in *FIFA* (EA Sports, 1993) or also consider out-of-game events and context such as the player (their actions, strategies, motivations, etc.). The *reteller* [58] exclusively narrates past events —often grouped into a concise "chunk" such as a game session (i.e. based on out-of-game context) or a quest (i.e. based only on in-game context). The *commentator* may be narrating current, ongoing events which have not been concluded, similar to a streamer concurrently discussing their current actions (including out-of-game context) or a sportscaster in an in-progress sports game such as *FIFA*.

The vision of automated "let's play"-style commentary generation is not new. It was proposed by Guzdial, Shah and Riedl *et al.* [59] and implemented via classical machine learning methods, with limited success. Ishigaki *et al.* [60] trained an LSTM with text, vision and game state input to generate characters for a commentary script in the racing game *Assetto Corsa* (Kunos Simulazioni, 2013). Results of this approach featured repetitive and context-irrelevant generated text. LSTMs were also used by Li, Gandhi and Harrison [61] to generate text, at a character level, for *Getting Over It With Bennett Foddy* (Bennett Foddy, 2017), a challenging side-scrolling climbing game.

LLMs for commentary were also explored by Renella and Eger [62], who argue that LLMs could assist game streamers (e.g. on Twitch) while the streamer multitasks gameplay with audience interaction. The authors developed a pipeline for automatically commenting upon *League of Legends* (Riot Games, 2009) games. They took a multi-phased approach, training a model on hand-annotated data to recognize key events, then prompting ChatGPT to generate zero-shot commentary on these events in the style of a

particular (known) fictional character, and finally sending the generated text through the FakeYou[3] API to be voiced in the timbre of this same character. For example, once the event detection model has identified an enemy double kill in a particular frame, ChatGPT responds in the style of Rick Sanchez from *Rick and Morty* (Cartoon Network, 2013): "What the heck?! That enemy team just got a double kill! I can't believe it! They must be pretty good! I better watch out for them!" An additional loop buffers detected events —for example, delaying commentary on a double kill in case it should escalate into a triple kill, or prioritizing among a quick barrage of events —and prompts ChatGPT to generate random fillers, such as thanking (fictional) new subscribers.

Despite the existence of the aforementioned studies, research on LLMs as game commentators remains rather limited. The appeal is obvious: simulation-based games of emergent narrative already generate rich narrative histories, and are remixed by human players to produce secondary content that is often popular in its own right. In principle, LLMs could be used to generate more succinct retellings or highlight reels of these game events. Prompting current LLMs for stories, without any further specification of style or substance, tends to produce output that feels generic. Past events recorded in simulation games could ultimately provide specificity and narrative coherence to these outputs. Exploring more concepts beyond automating streamer commentary, such as assisting streamers via LLM commentary of the audiences' reactions rather than the in-game actions, remains unexplored. We revisit this along other future applications in Section 4.

3.5 Analyst

Another role for LLMs is that of a data analyst. Here, we take this role to primarily analyze player experience and behavior, rather than the broader big-data job title for which some LLMs are naturally suited for [63], [64], [65]. While *player modeling* is an important aspect of AI research [8] and game development practice [66], LLMs have not received much attention for this purpose. However, the ability of LLMs to make sense of structured data such as code [67], [68] can be a boon for this type of work.

So far, language models (both large and small) have been used for clustering player behaviors. Player2vec [69] trained the Longformer transformer architecture [70], with up to 121 million parameters, on a corpus of game events stored in JSON format which captured player interactions with a casual mobile game. Using dimensionality reduction methods on the latent vectors of the last layer of a pre-trained transformer, the authors discovered eight clusters with player traits that could be useful for market research, such as "lean-in casual economy aware" [69]. In such work, the strength of LLMs is their ability to process structured or unstructured data without the need for extensive data pre-processing. However, as with previous experiments in bottom-up clustering of players [71], [72], the resulting clusters need to be interpreted by expert game analysts (or the game's designers) to derive some meaningful player types. In this form, current research points to LLMs acting as *analyst assistants* rather than independent analysts; we revisit future directions in this vein in Section 4.

Representing game logs as text interpreted by an LLM need not be the end-goal for analysis. The LLM representations of game logs can also be used to find common patterns that can be used for other purposes such as gameplay (or gameplay footage) similarity. Indicatively, Rasajski *et al.* [73] used LLMs on recorded gameplay action logs to establish action similarity. This similarity was in turn applied to train video encodings on gameplay footage of these recorded actions, in order to align the visual latent vectors to the action logs. While the work focused on the computer vision task of representation learning for gameplay footage, better (learned) representations of gameplay pixels would be more generalizable across games in downstream tasks. Such latent representations would be useful for general action recognition or general affect modeling [74]; we expand on these downstream tasks in Section 4.

3.6 Game Master

A Game Master (GM) in tabletop role-playing games (TTRPGs) is the person who creates the plot of a game, its characters, and narrative. GMs wear many hats during the course of the game session [75]; they prepare and adapt stories before sessions, guide gameplay during, and follow up with players afterward [76]. Digital games have mostly pre-scripted stories or level progressions and their players have a restricted range of affordances, compared to TTRPG players whose actions are only limited by their imagination. Similarly, the story told around the table can take any direction. Since human GMs mostly communicate about the gameworld, story, game state and action resolutions via natural language (although props such as maps, miniatures, hand-outs are also common), the potential of LLMs as a GM is often mentioned both in research circles and TTRPG discussion boards. LLMs as GMs also open the potential for solo play, while a TTRPG requires at least one player and a human GM.

One of the first notable text adventures managed by a fine-tuned version of GPT-2 is *AI Dungeon* [77]. It is an online[4] interactive chat-based storytelling application where the player takes actions through language input alone. The LLM continues the story based on the player's input, in the fashion of a human GM. The game has evolved since its creation to make use of more recent LLM models, which the player can choose from before starting a play session. Different gameworld settings are also offered, and players are also able to share the stories they create. Similar games have emerged online ever since[5], and a freely available code repository, *Kobold AI Client*[6], allows a local or remote installation of a client for such LLM-run games. Some of these games also use Stable Diffusion text-to-image models [78] to generate visuals accompanying different parts of the narrative. More recent work [79] investigates how different characterizations of the GM and their way of presenting events to the players impacts the overall experience in TTRPGs. This study presented an online interface for a

3. https://fakeyou.com/

4. https://play.aidungeon.com/
5. https://koboldai.net, https://www.hiddendoor.co
6. https://github.com/KoboldAI/KoboldAI-Client

Fig. 2. In *1001 Nights* [83], the player uses free-form text to trick the king (role-played by an LLM) into uttering the name of a particular weapon (which will then materialize, allowing the player to defeat him). Image used with permission.

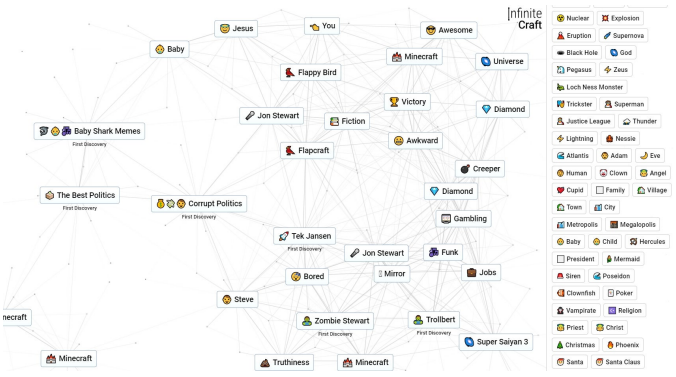


Fig. 3. In *Infinite Craft*, the player combines what begins as a simple set of atomic elements into increasingly complex entities, with an LLM dictating the product resulting from arbitrary combinations. Image used with permission.

custom game based on a *Dungeons & Dragons* (TSR, 1974) fantasy setting. LLMs were ascribed different GM roles through a number of prompts, and human players participated in 45-minute games (split into 3 sub-sessions), with an overall positive response to the automated GM.

In lieu of replacing a human GM, LLMs have also been employed as GM assistants. CALYPSO [80] is a set of tools running on a Discord server which the GM can query either to generate random encounters, brainstorm ideas, or alternatively chat with a fictional character in a *Dungeons & Dragons* setting. CALYPSO highlights that hallucinations of GPT-3 can have both positive effects when it generates plausible details not included in descriptions published in the original game manual (e.g. the shapes of creatures' eyes) and also negative effects when the created details are outright incorrect (e.g. describing the wings of a canonically wingless creature). In addition, the model's preconditioning to avoid racial bias was found to occasionally prevent it from generating racial details of fantasy creatures in the game. Other work used smaller GPT models to improvise in-game conversations [81] by monitoring and transcribing verbal exchanges between the GM and the players, and attempting to generate appropriate responses. This example was integrated into *Shoelace* [82], which is itself a GM assisting tool helping with content lookup by creating a node-based plan of the game narrative and encounters. The versatility of LLMs given their ability to rapidly process text input paves the way for their integration into the multitude of existing tools and aids for human GMs.

3.7 Game Mechanic

Games can also be built around a specific mechanic that relies on LLMs, similar to the AI-based game design patterns identified by Treanor *et al.* [84]. An obvious mechanic revolves around the social interactions facilitated by LLM-powered conversational NPCs. In this vein, the *Generative Agents* project [45] has employed LLMs to populate a virtual village with 25 characters, enabling them to communicate and engage in social behavior within a sandbox environment. Players were able to interact with these agents via text. The environment state and actions of each agent were stored in a language-based format and summarized in order to retain knowledge for each agent when prompting for its actions. This led to emerging believable social interactions, such as the agents spontaneously inviting other agents to a party which one of them was organizing. Similarly, GoodAI are developing the *AI people* video game which operates as a sandbox simulation where LLM-powered NPCs "interact with each other and their environment, forming relationships and displaying emotions" [41]. The player can interact with the agents via natural language chat, triggering reactions and potentially disrupting the relationship between NPCs (see Fig. 1).

Natural language interactions form a natural pool of mechanics to build games around, such as gamifying users' attempts at jailbreaking LLMs [85]. The game *1001 nights*, depicted in Figure 2, exemplifies this by having an LLM co-create a story from human prompts, where the player's objective is to try and steer the story to include specific keywords in order for the main character, Scheherazade, to turn these into tangible items in aid of her escape [83]. Similarly, *Gandalf*[7] challenges the player to trick an LLM into revealing a password. The game increases the difficulty of the task as levels progress by adjusting the prompt specifications, such as forcing the LLM to re-examine its generated response to ensure it does not include the password.

Another strength of LLMs is language synthesis. Huang and Sun [86] used GPT-3 to generate new words from the combination of two user-selected words. These are used to progress in a text-based game scenario, where the player wins by unlocking goal words. Similarly, language synthesis is leveraged by *Infinite Craft*[8], an "alchemy" game, in which the player combines elements to produce new ones (see Fig. 3). In *Infinite Craft*, the player begins with a set of core elements (water, fire, wind and earth). But while the former have a set of interactions defined manually by the designer, *Infinite Craft* prompts Llama 2 [20] to imagine the product of the combination of these elements [87]. Judging from gameplay, it appears that for each distinct combination, Llama is prompted to produce the result only once, with the product stored in a database for future reference. Thus seemingly anything in the language model's vocabulary

7. https://gandalf.lakera.ai/
8. https://neal.fun/infinite-craft/

might "emerge" from the combination of these elements, including all 50 states[9], "Dream"[10], and the fictional "Super Stonedosaurus Tacosaurus Rex"[11]. On occasion, the model can choose to return one of the combined elements, or refuse to combine (e.g. very lengthy or complex) elements.

3.8 Automated Designer

A key role of AI in games [8] is the algorithmic generation of game content such as levels and visuals, or even entire games. Unlike a Game Master who creates a game via natural language—meant to exist in the "theater of the mind" of the players—the aim of procedural content generation (PCG) is to create content intended for use in a digital game and thus it is required to satisfy certain constraints such as playability and aesthetic quality.

Any PCG method that is trained on available content corpora fits under the Procedural Content Generation via Machine Learning (PCGML) paradigm [23]. Strictly speaking the original PCGML framework of 2018 did not consider LLMs; instead it relied on machine learning methods such as autoencoders and LSTMs. However, important challenges of PCGML remain when considering LLMs for PCG: notably, the reliance on high-quality, machine-readable datasets from human-authored levels. While some datasets exist for arcade game levels [88], for most games the content remains both unavailable and protected by intellectual property (IP) laws. We revisit this issue in Section 6. Prior work in PCG has demonstrated that tile-based game levels can be reliably generated with sequence-based prediction models (e.g. LSTMs) from a modest set of examples, by treating such levels as linear sequences of tile types in raster order [89], [90].

More recently, this approach has been extended to modern LLMs pre-trained on natural language (instead of sequence models trained from the ground up). Todd *et al.* [91] fine-tuned a GPT-2 model on a large dataset of *Sokoban* (Thinking Rabbit, 1982) levels and, at test time, sampled from the model to produce novel puzzles (see Figure 4). Interestingly, their results indicate that while the GPT-2 model struggles when the fine-tuning dataset is restricted in size, GPT-3 (and, presumably, larger models released since then) are better able to accommodate limited training sets.

Using a similar approach, MarioGPT trains a GPT-2 model on a relatively small dataset of *Super Mario Bros* (Nintendo, 1985) levels [93]. MarioGPT overcomes the issue of data scarcity by using the initial dataset as the starting point for an evolutionary algorithm. Existing levels are selected and then sections of the level are mutated by sampling from the GPT model and then correcting the border between the re-generated section and the rest of the level with a similarly-trained BERT (i.e. bi-directional) model [2]. This approach produces a large and diverse set of playable levels, despite starting from less than 20 levels.

The above GPT-based level generation approaches also show the promise of incorporating natural language instructions to produce *conditional* level generators, either by prefixing game levels in the training dataset with desired level characteristics [91] or by embedding user instructions and allowing the model to attend to the embedding during generation [93]. A recent example of the latter is *Cardistry* [94], which used GPT-3.5 to transform a short personal narrative into a set of playing cards. Along with text information, the LLM creates prompts for DALL-E [95], completing the generation pipeline with playing card art.

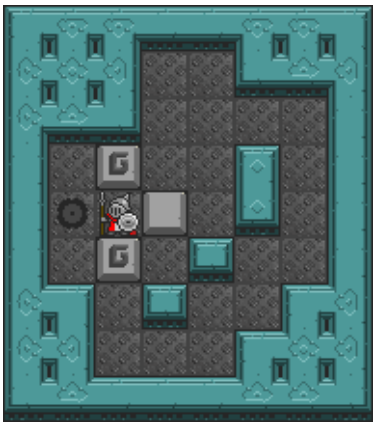


Fig. 4. A level for the puzzle game *Sokoban* generated by GPT-3, visualized with the Griddly tileset [92]. Image used with permission.

User requests however are typically fuzzy and can be misinterpreted by the LLM. Hu *et al.* [56] first "refine" user requests for the generation of *Minecraft* structures using an LLM, adding knowledge specific to the domain (such as block palette and building dimensions). Then, a second LLM is employed to generate the description of the structure that can be interpreted and placed in the game.

While most applications of LLMs as automated designers involve some mapping from the LLM's natural language output to other game-specific formats, LLMs can naturally be applied to games where the content is itself just natural language. In this vein, LLMs have been leveraged for the generation of New York Times *Connections* puzzles [96], which involve having a user group 16 words into 4 groups of 4 words each, with the challenge being for the user to infer what common semantic theme unites each group.

Unlike the above examples, LLMs can also produce instructions for other LLMs or Foundation Models (FMs) without a user request in the first place. In CrawLLM [97], the Mixtral 8x7B LLM [98] generates the theme, visual style, and even enemy descriptions which act as blueprints to guide additional LLM queries for producing player-facing narrative (e.g. introductory text) or textures and animations via Stable Diffusion [78] for re-theming a dungeon crawler game with card-based combat mechanics. In CrawLLM, the game design (code and card details) are pre-authored by the human, while LLMs and foundation models re-theme the assets to provide a visually and narratively novel experience every time (see Fig. 5).

Finally, LLMs can generate new games by writing game code directly. For instance, the GAVEL system [99] combines evolution with LLMs to generate board games. In GAVEL, a language model fine-tuned on a dataset of board game programs written in the Ludii description language [100] acts as the mutation operator within the evolutionary loop.

9. https://x.com/FeralFlex/status/1758332430136615298
10. https://x.com/slutzsmp/status/1760394123243135169
11. https://x.com/pvtspicy/status/1759316982984237139

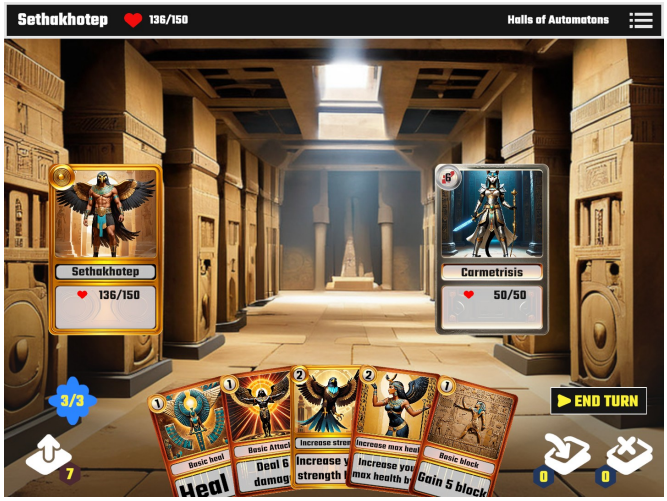


Fig. 5. A screenshot of a *CrawLLM* game instance, generated for the theme of Ancient Egypt [97]. In *CrawLLM*, the LLM generates themes, stories, characters, and locations for a card-based dungeon crawler game while Stable Diffusion generates the visuals. Image used with permission.

3.9 Design Assistant

An AI for design assistance can provide several benefits to the creative process. Depending on the type of tool, type of AI, and type of creative process, the AI can minimize development time and cost, reduce human effort, support collaboration among members of a design team, or elicit a user's creativity [101]. So far, in games, most of the AI-powered design assistant tools focus on autocompleting a human's in-progress design [102] or providing many possible suggestions for the designer to consider [103], [104], [105], [106], [107], [108]. Ideally, we would want an LLM that can act as a human colleague that we can bounce off ideas to and collaborate with. Such LLMs are still beyond the current state of the art [109]. Existing tools implement co-creating [110] LLMs at different levels of control, which we can explore under the existing Co-Creative Framework for Interaction Design [111]. Focusing on the *interaction* over the artifact, the LLM can be of *conceptual assistance*, providing high-level guidance which is not game-ready. This would require that the designer adapts and curates the AI output in a way that fits their own vision and the constraints of the game. This implies that the LLM contributes very little in the actual artifact generation, merely providing new suggestions. By allowing the LLM to also *refine* or *transform* the artifact, we see a *procedural assistance* by the LLM. Interacting with the designer, the LLM can produce increasingly more final versions of the intended artifact. The LLM is expected to understand the context of the game for which the content is intended, in order to provide meaningful assistance. However, the LLM does not need to produce a final, playable artifact but could instead simply provide the next creative step for discussion with the designer [112]. Moreover, the designer is ultimately responsible for curating and adapting the generated content, as well as deciding when the co-creative process is completed [113], [114], [115]. Finally, if the LLM is allowed to *directly* create and alter the artifact based on user requirements, we say it provides *production assistance*. This is the closest level to PCG (see Section 3.8), but is different in that the designer remains in control and can refine their specification or reject a created artifact (versus an autonomous generator which directly sends content to the player). As expected, however, the AI operates in a much more constrained space in this scenario as it must account for all other game mechanics (the design of which are presumed finalized) and designer goals which are somehow encoded or presumed via learned designer models [116].

One can argue that existing interfaces with LLMs and Large Multimodal Models (LMMs) act as design assistants. The designer provides their specifications and receives one (in LLMs) or multiple (in AI image generators) suggestions that they can further refine. Many creatives report using such interfaces for brainstorming and concept development [117], including game developers [118]. However, the applicability of LLMs as design assistants is somewhat limited, reverting only to conceptual assistance. Similarly, their potential for refining an existing idea (i.e. offering procedural assistance) is underexplored, as we discuss in Section 4.

Conceptual assistance is thus the easiest for LLMs, and is the first case explored in games. Charity *et al.* [107] envision design assistance as a tool that combines the game description provided by the user with existing knowledge of similar games to suggest possible game features back to the user. The suggested features are fairly generic, few-word guidelines (e.g.: "learn new combat") which would need extensive design effort and creativity to transform into an implementable and coherent game design. When asking an LLM for specific game features to implement in a digital game, players found them less compelling than human-designed ones [119]. The suggestions, however, were still useful for game designers, as they provided a different perspective that could kickstart their creation process. A thorough analysis of strengths and weaknesses of LLMs in this role can be drawn from *Project AVA* [120], a non-commercial digital game developed at Keywords Studios with the assistance of LLMs and LMMs for multiple aspects of the typical game development pipeline. LLMs are shown to help greatly in giving inspiration to the designers (albeit not being creative themselves), provide simple starting code for the game logic, and even assist developers by revealing errors in the code during development. LLMs however often fall short on anything more involved or requiring further domain knowledge, such as requesting code for specific game logic or understanding functionality in UI elements. Similarly, LMMs required a lot of tuning by human artists, but provided a solid foundation for concept art and proof-of-concept user interfaces.

Since production assistance is also close to traditional PCG pipelines, it is also understandably explored for games. Nasir and Togelius [121] used GPT-3 to generate levels for the *Metavoidal* (Yellow Lab Games, 2022) brawler game from a prompt that describes the level's features (e.g. width and height) while a human curates and edits the results to ensure playability. This curated set of levels is then used for further fine-tuning, potentially automating the generative process. Kelly *et al.* [122] instead use GPT-4 to generate stories in natural language while abiding to logic constraints, assisting story writers. Instead of generating the final artifact directly,

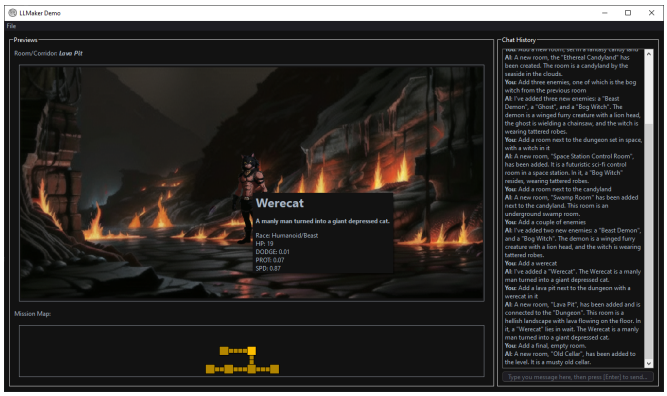


Fig. 6. A screenshot of the *LLMaker* digital game content design assistant tool [125]. In *LLMaker*, a human designer makes requests in natural language to alter a dungeon level for a video game, editing both the level layout and the entities within. The LLM satisfies the designer requests by calling the appropriate function, ensuring that the content adheres to domain constraints. Image used with permission.

Kumaran *et al.* [123] use the LLM to parse game level parameters from the user request. The natural language input to the LLM contains information about both what the user is looking for and properties that the level of *Future Worlds*—a game-based learning exhibit about environmental sustainability [124]—should posses, such as difficulty, size, type of sustainability problem, and simulation goal metrics. A collection of candidate game levels are generated from a single user request. Each candidate is then evaluated by a RL agent, and the best game is presented to the designer via the Unity game engine.

Examples of procedural assistance are very few: *LLMaker* [125] (see Figure 6) is a chat-only level editor where the user can chat with the LLM and request changes to be made to the game level and its content. Natural conversation with the LLM allows the designer to explore the current state of the level (for example, the user may ask "What are the enemies in the entrance room?") or draw inspiration for the next changes to make (for example, by asking "What kind of enemies should I add to this corridor?"). When the user requests a change to be made to the level, the LLM translates their request into a valid function call, ensuring content consistency and adherence to domain constraints [126]. Parameters for the function call that have not been specified by the user are generated by the LLM itself, biased by the existing level and overall user preferences. For example, if the user asks "Change the name of the banshee to Scary Ghost" the LLM will not only change the enemy's name but also their description to reflect the change, even though this was not specified explicitly by the user (see Fig. 6) In addition to the level and its content, *LLMaker* employs Stable Diffusion models to generate the graphical assets for the artifacts, based on descriptions generated by the LLM itself. As noted above, so far research has mostly focused on either LLMs for conceptual assistance (putting significant onus on a human designer) or as production assistance (leveraging a human designer as curator). The conversational nature of LLMs, however, seems particularly well-suited for procedural assistance when designing content; we revisit this underexplored area of research in Section 4.

4 A Roadmap for Future Applications of LLMs in Games

The previous section attempted to group current research in LLMs and games into a typology focused on the roles an LLM is asked to play. As part of this exercise, we identified a number of roles that have been heavily researched. Unsurprisingly, the role of player and automated designer have received most attention: this matches the general trends within AI and games research more broadly [8]. Following general trends in Game AI for playing or generating content, LLM-based approaches are likely to flourish via community events, benchmarks and competitions, with first steps already being taken in this direction [127]. Based on the roles listed in Section 3, we identify below some gaps found in the literature, and lay out possible research directions that leverage the power of LLMs in new ways.

While academic interest in design assistance within games has blossomed in the last decade, we find that the potential of LLMs has so far been underutilized. LLM design assistants either ask too much of a human designer in terms of creative interpretation and actual development [107] or too little, demoting them to content curator [121]. Past research in mixed-initiative systems [110] assumes a more co-creative initiative from both human and machine, and the power of LLMs as conversational agents matches the original vision of a creative dialogue between initiatives [115]. Therefore, a promising unexplored direction lies in a more procedural assistance (see Section 3.9) where the LLM not only produces output but also reasons about it to the human designer. LLMs seem especially well-suited for this task, as the context is retained and the designer can iteratively refine past products that the LLM has generated. However, concerns of LLMs' limited memory may arise (see Section 5) in long-term design processes. On the other hand, iterative refining is not as straightforward for other state-of-the-art technologies such as LMMs, despite some promising results via e.g. *InstructPix2Pix* [128]. It is expected that such applications will raise new challenges in terms of hallucinations, explainability [129], capturing or modeling designer intent [116], and more. We discuss such challenges further in Section 6.

While we identified player assistance as an important role that LLMs can play, we have found little work that targets any aspect of this beyond re-theming designer-defined hints [57]. The conversational ability of LLMs make them ideally suited for tutorial writing or hint-giving, especially in short snippets as provided e.g. by a conversational agent. However, it is important to note that LLMs often hallucinate or overfit to the corpus they have been trained on, and may be challenged, for instance, to summarize or lookup specific rules given a game manual. Similar limitations were identified when using an LLM as assistant to a human Game Master [81], where the LLM could not find information in the pre-written adventure when asked a question about the scene. Other technologies (as simple as a database search query) could be used instead, with the LLM undertaking only the task of converting the found information into a natural language utterance. Beyond mere hint-giving, however, an LLM could also act as a more hands-on player assistant, taking over more trivial tasks (such as managing minutiae of

one city in a strategy game). This is also powerful for Game Master assistance, as the LLM can keep track of locations visited and NPCs met, or look up rules. In both cases, addressing the issue of hallucinations and consistency will need to be addressed, which we review in Section 5.

Another role seemingly well-suited for LLMs that has received limited attention is that of commentator or reteller. Work so far has focused on automating the commentary of streamers or eSport casters [62]. While this direction is still largely uncharted, there are more directions that could leverage LLMs for streamer assistance rather than automation (and replacement). Rather than narrate events occurring within the game (or video stream), LLMs can summarize the audience interactions and engagement levels —thus acting as a commentator not of the game but of the audience watching it. This could allow a human streamer to better keep track of topics discussed in the chat, and engage as needed without having to read every comment. While this has been identified as a research direction for AI already [130], it has yet to be implemented. Under the role of streamer assistance, issues of explainability of the LLM's commentary would become pertinent (e.g. to address one audience member by name); we revisit this in Section 5. It is worth noting that streamers have already begun to explore AI assistance to their streams. To the best of our knowledge, the most relevant example is YouTube user Criken who plays alongside an AI assistant as a conversational agent, having a dialogue with an otherwise out-of-the-box LLM either embedded within the game[12] or as part of the stream[13]. This example is not explicitly targeted for reacting to in-game events or audience interaction (and acts more as NPC than commentator) but indicates that some streamers are open to the use of such technology for their craft.

Despite a few attempts to leverage LLMs for games user research (see Section 3.5), there is much unexplored potential in this direction. LLMs so far are used to cluster gameplay logs [69], [73], but they could also explain the groupings in the form of e.g. play personas [131], [71] described in natural language. Such a task would raise issues of explainability and privacy more broadly (see Section 5), and would likely still involve a game designer or user researcher in the loop for quality assurance. More importantly, moving from these log-based clusters towards capturing the player's experience or emotional state [132] remains an open challenge. In principle, an LLM could predict affective state transitions such as "the game is more engaging now" and thereby adapt the game environment to elicit a supposedly more engaging experience for the player. Learning such transitions builds on the experience-driven procedural content generation paradigm [133] but with an LLM acting as the player experience model. Future research could explore how LLMs can be fine-tuned to represent and infer player experience transitions based on in-game observations and demonstrations of experience. Two challenges need to be addressed for this: (a) representations of game states as natural language and (b) hallucinations of human experience. For the former challenge, potentially leveraging work of LLM players (see Section 3.1) and how they pass the game state to the LLM seems a promising first step. For the latter challenge, however, current LLMs struggle to capture user intent during conversation —let alone more ill-defined concepts such as players' emotion or engagement [134]. Current datasets on affect in games are formatted as continuous or categorical variables, often fluctuating over time [135], which would be challenging to format as text without processing. While perhaps using language as input or output for the player model requires some innovative pre-processing or more advanced LLM technologies, the underlying GPT architecture shows promise already. Broekens *et al.* showed that ChatGPT could detect emotion in English text [136], although admittedly games include many more modalities (e.g. visuals, audio) than pure text, which is mainly relegated to narrative [137]. We expect more research on player modeling powered by transformers, if not LLMs directly, such as leveraging behavior transformers [138] to imitate human playtraces grouped by playstyle [139].

To wrap up, we believe that every role an LLM could be called to play in (or around) a game identified in Section 3 could benefit from additional attention. This technology remains nascent and changes are forthcoming which may address several limitations we identified above and more extensively in Section 5. The natural language capabilities (especially for text generation) make LLMs ideal conversational assistants (for a player, a designer, a GM, or a streamer). The ability of LLMs to consume and reason from text corpora also opens new possibilities for automated design moving beyond tile-based level generation (which needs carefully crafted corpora) and more towards open-ended content such as game narratives [140], [141], [142], [143], [144], [145] or even game design documents. The potential of LLMs in that regard is already voiced by many evangelists in the field, but research on actual implementation of such ideas and on addressing the IP concerns they may raise (see Section 6) are still forthcoming.

While the focus of this paper is on what LLMs can do for games, we do not underestimate what games can do for LLMs. One of the watershed moments for AI and games research was the article by Laird and Van Lent naming games as the "killer app" for human-level AI [146]. This remains true for LLMs today: games are ideally poised for LLM research. Not only do games produce rich multimodal data (ideal for e.g. LMMs), but there also exist rich corpora of text and multimodal data produced by players, viewers, fans, etc. Game text data, such as transcripts, have already been used to train LLM players [39], [40]. On the other hand, LLMs struggle with both spatial reasoning and planning by their very nature, while most games rely heavily on both aspects. From strategy board games and digital games (where long-term planning is crucial) to first-person shooters (which hinge on precision in spatial reasoning and a reactive plan for reaching the enemy base), such games remain state-of-the-art testbeds for gameplaying AI [147], [148] and will likely be fraught arenas for LLM research. Games also hinge on long-term interactions, especially in the case of LLM-based GMs (see Section 3.6). Games can thus form testbeds or benchmarks to explore the limits of recollection under different context lengths, a critical limitation of LLMs detailed in Section 5. In terms of game design tasks, we also note that games are complex constrained problems, with

12. https://youtu.be/dQ-7-r5aM1U
13. https://youtu.be/KhE9NhUqtBc

hard constraints on e.g. levels that can be completed [149], but also soft constraints regarding game balance between competing players in multi-player games [150], [139], or the progression and pacing of a single-player experience [102]. While some LLMs can handle some hard constraints via, say, function calling [126], this may not be possible for more complex or more constrained game domains. Moreover, soft constraints would need to be conveyed to the LLM in more nuanced ways. Game benchmarks specific to LLMs have already started to emerge [127], but identifying critical game-based challenges for LLMs, appropriate and interesting benchmarks, and (ethically sourced) data for training or fine-tuning LLMs remains an open question.

5 Limitations of LLMs in games

Large language models have exciting potential for games, but they also come with inherent limitations. Mainly, LLMs suffer from hallucinations [50], [151], meaning that they will output plausible but false statements simply because they are a probable sequence of words. Hallucinations are inevitable, given how the world is described to the machine [152]; LLMs lack grounding, so the text they generate is detached from constraints of reality. Yet LLMs always "act" confidently in their responses, even when wholly mistaken. Indeed, Hicks *et al.* argue that the term *AI hallucinations* misrepresents how "the models are in an important way indifferent to the truth of their outputs" [7]. LLMs are shown to also output responses that are wrong even though the LLM has access to information that proves otherwise [153], [154], [155]. In the context of digital games, these limitations affect certain applications of LLMs more than others, for example NPCs may hallucinate quests that do not exist in the game, or a player assistant may provide suggestions to the user based on wrong assumptions.

Another limitation is that LLMs sometimes struggle to capture user intent. This is especially evident with expressions of sarcasm [156]. The ability to capture user intent is important for applications of LLMs that converse directly with the player. Many LLMs misunderstand user requests [157], and clarifying to the LLM multiple times leads to frustration. This limitation is most relevant to cases where the LLM is in direct conversation with the user, e.g. as design assistant, player assistant, or Game Master. Depending on how much the LLM output controls the user experience (e.g. as Game Master or offering production assistance to a human designer), the inability to capture user intent can be a frustrating experience.

On a larger scale, LLMs suffer from losing context, and struggle with continuity. This is because the "memory" of an LLM is constrained by its context size, which limits the extent of its inputs and outputs, as well as its response time due to the attention mechanism [18]. The longer the conversation, the less likely it is that the LLM will recall early events [158]. In digital games, it is possible to separately summarize the game events (see Section 3.4) and process them as part of the input to the LLM. As a game progresses past a few game sessions, however, this summary may still be too long, or details of increasing significance will be omitted, thus leading to a degraded performance. This is especially relevant for roles requiring long-term engagement, such as LLM-powered retellers or Game Masters. In *Infinite Craft* (see Section 3.7), this is handled by an external database that stores and looks up past combination rules, ensuring consistency in future uses of the same mechanic. However, LLMs could theoretically tackle this issue directly.

Recent models have tried to address this recollection issue by increasing the context length, with some of the larger models encompassing 128K or even 10M tokens [159]. Despite this being adequate for a wide range of applications, it may still fall short when applied to long-term tracking of game states. In particular, massive multiplayer online games offer a simulation space with a large intricate domain of actions and interactions, which scales exponentially with the number of agents (players or otherwise) participating. Researchers have also tried to address the context limit by including compressive memory into the attention mechanism of the LLM [160], in an attempt to create a seemingly infinite context length. The authors of [160], however, acknowledge its current limitation, partly due to the difficulty of selecting and compressing the data which should be "memorized". A different approach proposed by Fountas *et al.* [161] draws inspiration from cognitive science to equip LLMs with episodic memory, greatly reducing context length limitations during information retrieval.

A Retrieval-Augmented Generation (RAG) system [162] could address this limitation, drawing from a database containing vector representations or other latent representations of pertinent text or data. When the text generator processes a sequence, the RAG system would retrieve similar entries from this external data source. This would hypothetically provide a streamlined archive of game events and actions for the LLMs to consult in order to generate a consistent narrative progression.

Another challenge is that currently LLMs are trained to be highly compliant to the users' requests. For an LLM assistant, this is not a cause for concern, but in the role of a Game Master this can create issues. Human GMs frequently curb the more exotic player requests which could drastically diverge from the game narrative or which would result in an unrecoverable disruption of a required sequence of game events. An LLM Game Master would try to accommodate for even the most bizarre requests, with little consideration for the consequential impacts to any predetermined game events.

Yet another limitation of LLMs that prevents their application in mainstream media is their cost. Running AAA games and LLMs in parallel on consumer hardware is infeasible [163] due to their computation requirements. If one wants to integrate LLMs in games, they would have to host the models on their own servers or access existing models via APIs. Additionally, the cost of querying LLMs is a recurring cost, and cannot be properly estimated beforehand. This kind of problem is also affected by the scale of LLMs-powered games or tools. Similarly to how server costs increase with the number of active players in massive multiplayer online games, the more players use a LLM over multiple play sessions, the more the game developers or publishers will have to bear the financial burden. The monetary cost of this approach can be prohibitive or difficult to estimate for real-world applications. The game

need not even be played by other players: to evaluate the performance of their simulations with multiple LLM-based NPCs, Park *et al.* ran simulations for several days with a cost of "thousands of dollars in token credits" [45]. While promising techniques to reduce the costs of running LLMs exist [164], [165], these are not yet widespread and require further engineering to set them up properly.

Perhaps due to the above limitations, the implementation and deployment of LLMs in digital game applications is still very limited. A digital game is a domain where responsiveness is vital for players, so it follows that LLMs should also be able to provide their responses quickly. Unfortunately, while research on more efficient and faster architectures is being carried out [166], the real-time application of LLMs is still not plausible. This is especially evident in other domains such as design applications, where "real time" responses are generated in around 30 seconds to over a minute [167].

6 Ethical Issues with LLMs in games

With the improvement of AI methods applied to games over the recent years, many questions regarding their ethics and real-world impact have been raised [168]. Using LLMs raises ethical issues regarding sustainability, copyright, explainability, privacy, and biases. Naturally, each of these issues has serious implications in the field of games.

The reliance of LLMs on training data and training time raises concerns regarding their carbon footprint. Beyond training costs, inference over the model's lifespan has a greater environmental impact due to constant querying [169], [170]. Factors like renewable and local energy, better model architectures, and more meaningful (and thus less wasteful) training data can mitigate this. In the context of LLMs for digital games, sustainability remains crucial, considering the carbon footprint of frequent queries during gameplay (e.g. for Game Master or NPC responses, or for LLM-powered players). This is especially pertinent if the LLM is intended to run locally, on consumer-level hardware which are usually powered by non-renewable sources.

When it comes to copyright, issues apply to the input data, the output data, and the model itself. LLMs trained on data under copyright is an unfortunate common practice [171], deservedly raising public outrage [172], [173]. The models themselves have different copyright licenses applied, which can also lead to artifacts they generate to fall under the public domain [19], [174]. For the game industry, matters of IP and copyright are extremely important. This is as much a concern regarding having the company's copyrighted content somehow used as training data by competitors, as it is about LLMs producing material that the company cannot copyright. It is important to note here that, at least when it comes to the latter concern, the role the LLM takes is very pertinent. If an LLM or LMM produces content automatically (see Section 3.8), past legal consensus in the USA indicates that the material can not be copyrighted [175]. If an LLM or LMM acts as an "assistive tool" [176] to a designer (especially for conceptual assistance, see Section 3.9) then the extensive and impactful human effort needed to transform these concepts into game design and game art likely makes the final product eligible for copyright [176]. The limited rulings in copyright courts regarding this, however, and the "likely" caveat we include in our own text, understandably would make game companies hesitant to tread in untested waters for major game IPs beyond e.g. small-scale indie productions [120], [83]. For researchers, however, the ethical issues of copyright breach and exploitation by large corporations, and the public outcry for the above, leave a bad taste and make research in LLMs less palatable [177].

In applications, understanding how a final result or product is reached is extremely crucial, particularly when a product is iteratively refined as with design assistants (see Section 3.9). This is a problem of explainability [129], whereas LLMs are inherently opaque in their generation process. Liu *et al.* [178] highlight different methods to improve the explainability of language models, such as concept-based explanations or saliency maps. Particularly for LLMs, the self-explanation applied via the chain-of-thought [179] reasoning has received attention by the research community [180], [181]. While this method adds a layer of explained reasoning to the generated output, there are multiple examples in the literature that demonstrate how this reasoning may just be an illusion of reasoning capabilities. Such examples include disregarding the provided reasoning in the final output [182], or reaching the correct solution via incorrect steps in math problems [183]. In the domain of games, explainability is paramount across roles, ensuring gameplay coherence and user engagement.

Replicability of an application's behavior is equally crucial. When applying LLMs to digital games (especially as game mechanics, NPCs, or automated designers) one would expect their output quality to not change over time. This is not the case for closed-source LLMs: even when using the same model name, the same request at one time can generate content that is vastly different from a past iteration [184]. In this case, developers may have to consider switching to open-source models with open weights, such as those hosted on the popular HuggingFace Transformers library [185]. An additional benefit of switching to self-hosted models, model size notwithstanding, is the additional level of privacy that is guaranteed to the users. Querying local models ensures all messages remain within the application, whereas interacting with models hosted via APIs entails that conversations are exchanged over third-party websites. A developer might be willing to share conversation logs with an API provider[14] for model improvement. However, users may not be aware of this practice or its implications. Local deployment of LLMs has been democratized by making models more accessible on lower-end hardware, relying on the widespread adoption of the GGUF format[15] and the release of different versions of the same model at varying degrees of quantization [186]. The quantization of an LLM usually results in a loss of performance, but this is usually considered a valid trade-off for the reduced model size to load on VRAM. Combined with friendly APIs for running

14. Such as ChatGPT, which shares conversations by default unless opted out

15. Details of this file format are available at https://github.com/ggerganov/ggml/blob/master/docs/gguf.md

LLMs locally, such as Open WebUI[16] and LM Studio[17], it is possible to run LLMs in a more controlled fashion. The more pertinent technological breakthroughs lie in compacting size and carbon footprint while retaining high-quality LLM outputs.

Finally, biases emerge as LLMs are trained on a large corpus, usually scraped from the (Western-focused part of the) internet. This allows models to capture a current reality snapshot, which is advantageous for a conversational or question-answering model, though it requires curating this data from different kinds of biases. Some biases, such as social stereotypes, could be targeted and alleviated; others, such as exclusionary norms, pose greater challenges. In games, we identify two main concerns when interacting with an LLM: toxic behavior, and stereotypes or incorrect notions. Toxic behavior is a harmful property that a language model may learn from its training corpus, which often contains text from community-based fora or social platforms. Tools that combat toxic language in digital games are constantly evolving, with some even blocking chat messages before they are delivered to the user [187], [188]. Therefore, similar applications could theoretically be developed to target toxic outputs from language models. Unlike human players, however, when an LLM plays the role of an NPC, it should align with the game themes and avoid any kind of toxic language or racial slurs. This requires developers to ensure proper behavior of the model through data cleaning, if the model is trained from scratch, or supplying tailored data if finetuning it to their needs. Addressing *prejudices* such as stereotypes and incorrect notions is complex, as they are not necessarily related to single words or expressions, but instead present themselves as a collection of ideals that can be wrong at best, and harmful at worst. An NPC LLM may exhibit real world stereotypes that can negatively impact the player experience, although we argue that the impact of prejudices from an LLM commentator or Game Master is much stronger and disturbing due to their perceived authority.

7 Conclusions

As discussed in this paper, LLMs can take up many different roles that can improve the experience of players in digital games, or enhance the ability of game designers to bring their ideas to life. However, we also highlighted many different challenges specific to the applications of LLMs and intrinsic to the nature of LLMs and the ecosystem that surrounds them. Despite technical, ethical, and legal challenges posed by LLMs, it is not realistic to ignore the impact that this research will likely have on both Game AI research and the game industry. We expect to see many new technical innovations from LLM researchers and corporations. Anticipating this, we propose promising directions where LLMs could be applied to games in the future.

Acknowledgments

This work has been supported by the European Union's Horizon 2020 research and innovation programme from the AI4media project (Grant Agreement No. 951911), and by the US National Science Foundation under the Graduate Research Fellowship Program.

16. https://openwebui.com/
17. https://lmstudio.ai/

References

[1] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, 2019.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*, 2019.

[3] Z. Zhang, X. Han, Z. Liu, X. Jiang, M. Sun, and Q. Liu, "ERNIE: Enhanced language representation with informative entities," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2019.

[4] T. Kaufmann, P. Weng, V. Bengs, and E. Hüllermeier, "A survey of reinforcement learning from human feedback," *arXiv preprint arXiv:2312.14925*, 2023.

[5] S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee, Y. Li, S. Lundberg, H. Nori, H. Palangi, M. T. Ribeiro, and Y. Zhang, "Sparks of artificial general intelligence: Early experiments with GPT-4," *arXiv preprint arXiv:2303.12712*, 2023.

[6] R. T. McCoy, S. Yao, D. Friedman, M. Hardy, and T. L. Griffiths, "Embers of autoregression: Understanding large language models through the problem they are trained to solve," *arXiv preprint arXiv:2309.13638*, 2023.

[7] M. T. Hicks, J. Humphries, and J. Slater, "Correction: ChatGPT is bullshit," *Ethics and Information Technology*, vol. 26, no. 46, 2024.

[8] G. N. Yannakakis and J. Togelius, *Artificial Intelligence and Games*. Springer, 2018.

[9] J. Gwertzman and J. Soslow, "The generative AI revolution in games," https://a16z.com/the-generative-ai-revolution-in-games/, 2022, accessed 25 Feb 2024.

[10] S. Minaee, T. Mikolov, N. Nikzad, M. Chenaghlu, R. Socher, X. Amatriain, and J. Gao, "Large language models: A survey," *arXiv preprint arXiv:2402.06196*, 2024.

[11] P. García-Sánchez, A. M. García, P. A. Castillo, and I. J. Pérez, "A bibliometric study of the research area of videogames using Dimensions.ai database," in *Proceedings of the International Conference on Information Technology and Quantitative Management*, 2019.

[12] A. Liapis, "10 years of the PCG workshop: Past and future trends," in *Proceedings of the FDG Workshop on Procedural Content Generation*, 2020.

[13] A. Jordanous and B. Keller, "Modelling creativity: Identifying key components through a corpus-based approach," *Public Library of Science*, vol. 11, pp. 1–27, 2016.

[14] D. Yang, E. Kleinman, and C. Harteveld, "GPT for games: A scoping review (2020-2023)," in *Proceedings of the IEEE Conference on Games*, 2024.

[15] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, 1948.

[16] J. L. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, 1990.

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, pp. 1735–1780, 1997.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the Conference on Neural Information Processing Systems*, 2017.

[19] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed, "Mistral 7B," *arXiv preprint arXiv:2310.06825*, 2023.

[20] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[21] Z. Yang, L. Li, K. Lin, J. Wang, C.-C. Lin, Z. Liu, and L. Wang, "The dawn of LMMs: Preliminary explorations with GPT-4V(ision)," *arXiv preprint arXiv:2309.17421*, 2023.

[22] H. Liu, C. Li, Q. Wu, and Y. J. Lee, "Visual instruction tuning," in *Proceedings of the Neural Information Processing Systems Conference*, 2023.

[23] A. Summerville, S. Snodgrass, M. Guzdial, C. Holmgård, A. K. Hoover, A. Isaksen, A. Nealen, and J. Togelius, "Procedural content generation via machine learning (PCGML)," *IEEE Transactions on Games*, vol. 10, no. 3, 2018.

[24] S. J. Edwards, "Standard: Portable game notation specification and implementation guide," https://ia802908.us.archive.org/26/items/pgn-standard-1994-03-12/PGN_standard_1994-03-12.txt, 1993, accessed 12 June 2024.

[25] D. Noever, M. Ciolino, and J. Kalin, "The chess transformer: Mastering play using generative language models," *arXiv preprint arXiv:2008.04057*, 2020.

[26] A. Stöckl, "Watching a language model learning chess," in *Proceedings of the Recent Advances in Natural Language Processing International Conference*, 2021.

[27] S. Toshniwal, S. Wiseman, K. Livescu, and K. Gimpel, "Chess as a testbed for language model state tracking," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 10, 2022.

[28] M. Ciolino, J. Kalin, and D. Noever, "The Go transformer: Natural language modeling for game play," in *Proceedings of the Artificial Intelligence for Industries International Conference*, 2020.

[29] K. Li, A. K. Hopkins, D. Bau, F. Viégas, H. Pfister, and M. Wattenberg, "Emergent world representations: Exploring a sequence model trained on a synthetic task," in *Proceedings of the International Conference on Learning Representations*, 2023.

[30] B. Bateni and J. Whitehead, "Language-driven play: Large language models as game-playing agents in Slay the Spire," in *Proceedings of the International Conference on the Foundations of Digital Games*, 2024.

[31] S. Reed, K. Zolna, E. Parisotto, S. G. Colmenarejo, A. Novikov, G. Barth-Maron, M. Gimenez, Y. Sulsky, J. Kay, J. T. Springenberg *et al.*, "A generalist agent," *arXiv preprint arXiv:2205.06175*, 2022.

[32] M. Xu, W. Dai, C. Liu, X. Gao, W. Lin, G.-J. Qi, and H. Xiong, "Spatial-temporal transformer networks for traffic flow forecasting," *arXiv preprint arXiv:2001.02908*, 2020.

[33] H. Chang, H. Zhang, L. Jiang, C. Liu, and W. T. Freeman, "MaskGIT: Masked generative image transformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[34] S. Yao, R. Rao, M. Hausknecht, and K. Narasimhan, "Keep CALM and explore: Language models for action generation in text-based games," in *Proceedings of the Empirical Methods in Natural Language Processing Conference*, 2020.

[35] C. F. Tsai, X. Zhou, S. S. Liu, J. Li, M. Yu, and H. Mei, "Can large language models play text games well? current state-of-the-art and open questions," *arXiv preprint arXiv:2304.02868*, 2023.

[36] P. Ammanabrolu, L. Jiang, M. Sap, H. Hajishirzi, and Y. Choi, "Aligning to social norms and values in interactive narratives," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2022.

[37] M. Hausknecht, P. Ammanabrolu, M.-A. Côté, and X. Yuan, "Interactive fiction games: A colossal adventure," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[38] M. Klissarov, P. D'Oro, S. Sodhani, R. Raileanu, P.-L. Bacon, P. Vincent, A. Zhang, and M. Henaff, "Motif: Intrinsic motivation from artificial intelligence feedback," *arXiv preprint arXiv:2310.00166*, 2023.

[39] Meta Fundamental AI Research Diplomacy Team (FAIR), A. Bakhtin, N. Brown, E. Dinan, G. Farina, C. Flaherty, D. Fried, A. Goff, J. Gray, H. Hu *et al.*, "Human-level play in the game of diplomacy by combining language models with strategic reasoning," *Science*, vol. 378, no. 6624, 2022.

[40] G. Wang, Y. Xie, Y. Jiang, A. Mandlekar, C. Xiao, Y. Zhu, L. Fan, and A. Anandkumar, "Voyager: An open-ended embodied agent with large language models," in *Proceedings of the NeurIPS Workshop on Foundation Models for Decision Making*, 2023.

[41] GoodAI, "Introducing our work on general-purpose LLM agents," https://www.goodai.com/introducing-general-purpose-llm-agents/, 2023, accessed 24 Feb 2024.

[42] M. Ç. Uludağlı and K. Oğuz, "Non-player character decision-making in computer games," *Artificial Intelligence Review*, vol. 56, no. 12, 2023.

[43] M. Shanahan, K. McDonell, and L. Reynolds, "Role play with large language models," *Nature*, vol. 623, 2023.

[44] F. Gao, K. Fang, and W. K. Victor Chan, "Chemical life: Knowledge-based personality, emotion and action cues in educational games," in *Proceedings of the IEEE Conference on Games*, 2023.

[45] J. S. Park, J. O'Brien, C. J. Cai, M. R. Morris, P. Liang, and M. S. Bernstein, "Generative agents: Interactive simulacra of human behavior," in *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, 2023.

[46] M. Kreminski, "Toward better gossip simulation in emergent narrative systems," in *Proceedings of the IEEE Conference on Games*, 2023.

[47] M. Müller-Brockhausen, G. Barbero, and M. Preuss, "Chatter generation through language models," in *Proceedings of the IEEE Conference on Games*, 2023.

[48] L. O'Brien, "How Ubisoft's New Generative AI Prototype Changes the Narrative for NPCs," https://news.ubisoft.com/en-us/article/5qXdxhshJBXoanFZApdG3L/how-ubisofts-new-generative-ai-prototype-changes-the-narrative-for-npcs, 2024, accessed 12 June 2024.

[49] H. Warpefelt and H. Verhagen, "A model of non-player character believability," *Journal of Gaming & Virtual Worlds*, vol. 9, no. 1, 2017.

[50] J. Duan, H. Cheng, S. Wang, A. Zavalny, C. Wang, R. Xu, B. Kailkhura, and K. Xu, "Shifting attention to relevance: Towards the uncertainty estimation of large language models," *arXiv preprint arXiv:2307.01379*, 2023.

[51] A. Mehta, Y. Kunjadiya, A. Kulkarni, and M. Nagar, "Exploring the viability of conversational AI for non-playable characters: A comprehensive survey," in *Proceedings of the International Conference on Recent Trends in Computer Science and Technology*, 2022.

[52] E. Akata, L. Schulz, J. Coda-Forno, S. J. Oh, M. Bethge, and E. Schulz, "Playing repeated games with large language models," *arXiv preprint arXiv:2305.16867*, 2023.

[53] Y. Xu, S. Wang, P. Li, F. Luo, X. Wang, W. Liu, and Y. Liu, "Exploring large language models for communication games: An empirical study on Werewolf," *arXiv preprint arXiv:2309.04658*, 2023.

[54] P. Maas, F. Carey, C. Wheeler, E. Saatchi, P. Billington, and J. Yaffa Shamash, "To infinity and beyond: SHOW-1 and Showrunner agents in multi-agent simulations," https://fablestudio.github.io/showrunner-agents/, 2023, accessed 27 Feb 2024.

[55] J. Urbanek, A. Fan, S. Karamcheti, S. Jain, S. Humeau, E. Dinan, T. Rocktäschel, D. Kiela, A. Szlam, and J. Weston, "Learning to speak and act in a fantasy text adventure game," in *Proceedings of the Empirical Methods in Natural Language Processing Conference and the Natural Language Processing International Joint Conference*, 2019.

[56] S. Hu, Z. Huang, C. Hu, and J. Liu, "3D building generation in Minecraft via large language models," in *Proceedings of the IEEE Conference on Games*, 2024.

[57] T. Rist, "Using a large language model to turn explorations of virtual 3d-worlds into interactive narrative experiences," in *Proceedings of the IEEE Conference on Games*, 2024.

[58] M. P. Eladhari, "Re-tellings: The fourth layer of narrative as an instrument for critique," in *Proceedings of the Interactive Digital Storytelling International Conference*, 2018.

[59] M. Guzdial, S. Shah, and M. Riedl, "Towards automated let's play commentary," in *Proceedings of the AIIDE workshop on Experimental AI in Games*, 2018.

[60] T. Ishigaki, G. Topić, Y. Hamazono, H. Noji, I. Kobayashi, Y. Miyao, and H. Takamura, "Generating racing game commentary from vision, language, and structured data," in *Proceedings of the Natural Language Generation International Conference*, 2021.

[61] C. Li, S. Gandhi, and B. Harrison, "End-to-end let's play commentary generation using multi-modal video representations," in *Proceedings of the International Conference on the Foundations of Digital Games*, 2019.

[62] N. Renella and M. Eger, "Towards automated video game commentary using generative AI," in *Proceedings of the AIIDE workshop on Experimental AI in Games*, 2023.

[63] G. Jaimovitch-López, C. Ferri, J. Hernández-Orallo, F. Martínez-Plumed, and M. J. Ramírez-Quintana, "Can language models automate data wrangling?" *Machine Learning*, 2023.

[64] L. Cheng, X. Li, and L. Bing, "Is GPT-4 a good data analyst?" in *Findings of the Association for Computational Linguistics*, 2023.

[65] P. Maddigan and T. Susnjak, "Chat2VIS: Generating data visualisations via natural language using ChatGPT, Codex and GPT-3 large language models," *IEEE Access*, vol. 11, 2023.

[66] M. S. El-Nasr, A. Drachen, and A. Canossa, *Game Analytics: Maximizing the Value of Player Data 2013th Edition*. Springer, 2013.

[67] Z. Cheng, T. Xie, P. Shi, C. Li, R. Nadkarni, Y. Hu, C. Xiong, D. Radev, M. Ostendorf, L. Zettlemoyer, N. A. Smith, and T. Yu, "Binding language models in symbolic languages," in *Proceedings of the International Conference on Learning Representations*, 2023.

[68] Z. Chen, H. Mao, H. Li, W. Jin, H. Wen, X. Wei, S. Wang, D. Yin, W. Fan, H. Liu, and J. Tang, "Exploring the potential of large language models (LLMs) in learning on graph," in *Proceedings of the NeurIPS Workshop on New Frontiers in Graph Learning*, 2023.

[69] T. Wang, M. Honari-Jahromi, S. Katsarou, O. Mikheeva, T. Panagiotakopoulos, S. Asadi, and O. Smirnov, "player2vec: A language modeling approach to understand player behavior in games," *arXiv preprint arXiv:2404.04234*, 2024.

[70] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The long-document transformer," *arXiv preprint arXiv:2004.05150*, 2020.

[71] A. Drachen, A. Canossa, and G. N. Yannakakis, "Player modeling using self-organization in Tomb Raider: Underworld," in *Proceedings of the IEEE Symposium on Computational Intelligence and Games*, 2009.

[72] A. Canossa, J. B. Martinez, and J. Togelius, "Give me a reason to dig: Minecraft and psychology of motivation," in *Proceedings of the IEEE Conference on Computational Intelligence and Games*, 2013.

[73] N. Rasajski, C. Trivedi, K. Makantasis, A. Liapis, and G. N. Yannakakis, "BehAVE: Behaviour alignment of video game encodings," in *Proceedings of the ECCV Workshop on Computer Vision For Videogames*, 2024.

[74] J. Togelius and G. N. Yannakakis, "General general game AI," in *Proceedings of the IEEE Conference on Computational Intelligence and Games*, 2016.

[75] A. Tychsen, M. Hitchens, T. Brolund, and M. Kavakli, "The game master," in *Proceedings of the Australasian conference on Interactive entertainment*, 2005.

[76] A. Liapis and A. Denisova, "The challenge of evaluating player experience in tabletop role-playing games," in *Proceedings of the International Conference on the Foundations of Digital Games*, 2023.

[77] M. Hua and R. Raley, "Playing with unicorns: AI dungeon and citizen NLP," *Digital Humanities Quarterly*, vol. 14, no. 4, 2020.

[78] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[79] X. You, P. Taveekitworachai, S. Chen, M. C. Gursesli, X. Li, Y. Xia, and R. Thawonmas, "Dungeons, Dragons, and Emotions: A preliminary study of player sentiment in LLM-driven TTRPGs," in *Proceedings of the International Conference on the Foundations of Digital Games*, 2024.

[80] A. Zhu, L. Martin, A. Head, and C. Callison-Burch, "CALYPSO: LLMs as Dungeon Masters' assistants," in *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2023.

[81] J. Kelly, M. Mateas, and N. Wardrip-Fruin, "Towards computational support with language models for TTRPG game masters," in *Proceedings of the FDG Workshop on Human-AI Interaction through Play*, 2023.

[82] D. Acharya, J. Kelly, W. Tate, M. Joslyn, M. Mateas, and N. Wardrip-Fruin, "Shoelace: A storytelling assistant for GUMSHOE One-2-One," in *Proceedings of the International Conference on the Foundations of Digital Games*, 2023.

[83] Y. Sun, Z. Li, K. Fang, C. H. Lee, and A. Asadipour, "Language as reality: A co-creative storytelling game experience in 1001 Nights using generative AI," in *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2023.

[84] M. Treanor, A. Zook, M. P. Eladhari, J. Togelius, G. Smith, M. Cook, T. Thompson, B. Magerko, J. Levine, and A. Smith, "AI-based game design patterns," in *Proceedings of the International Conference on the Foundations of Digital Games*, 2015.

[85] Y. Liu, G. Deng, Z. Xu, Y. Li, Y. Zheng, Y. Zhang, L. Zhao, T. Zhang, and Y. Liu, "Jailbreaking ChatGPT via prompt engineering: An empirical study," *arXiv preprint arXiv:2305.13860*, 2023.

[86] L. Huang and X. Sun, "Create ice cream: Real-time creative element synthesis framework based on GPT3.0," in *Proceedings of the IEEE Conference on Games*, 2023.

[87] T. Litchfield, "This browser-based 'endless crafting game' starts you off with fire and water, but it quickly escalates to God, the Big Bang, and 'Yin-Yoda'," https://www.pcgamer.com/this-browser-based-endless-crafting-game-starts-you-off-with-fire-and-water-but-it-quickly-escalates-to-god-the-big-bang-and-yin-yoda/, 2024, accessed 28 Feburary 2024.

[88] A. J. Summerville, S. Snodgrass, M. Mateas, and S. Ontanon, "The VGLC: The video game level corpus," in *Proceedings of the FDG Workshop on Procedural Content Generation*, 2016.

[89] A. Summerville and M. Mateas, "Super Mario as a string: Platformer level generation via LSTMs," in *Proceedings of the Joint Conference of DIGRA and FDG*, 2016.

[90] Y. Zakaria, M. Fayek, and M. Hadhoud, "Procedural level generation for Sokoban via deep learning: An experimental study," *IEEE Transactions on Games*, vol. 15, no. 1, 2022.

[91] G. Todd, S. Earle, M. U. Nasir, M. C. Green, and J. Togelius, "Level generation through large language models," in *Proceedings of the International Conference on the Foundations of Digital Games*, 2023.

[92] C. Bamford, "Griddly: A platform for AI research in games," *Software Impacts*, vol. 8, 2021.

[93] S. Sudhakaran, M. González-Duque, M. Freiberger, C. Glanois, E. Najarro, and S. Risi, "MarioGPT: Open-ended text2level generation through large language models," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[94] B. Lyman, A. Ebrahimi, J. E. C. III, S. Chan, C. Barney, and B. D. Schutter, "Cardistry: Exploring a GPT model workflow as an adapted method of gaminiscing," in *Proceedings of the International Conference on the Foundations of Digital Games*, 2024.

[95] J. Betker, G. Goh, L. Jing, T. Brooks, J. Wang, L. Li, L. Ouyang, J. Zhuang, J. Lee, Y. Guo, W. Manassra, P. Dhariwal, C. Chu, YunxinJiao, and A. Ramesh, "Improving image generation with better captions," https://cdn.openai.com/papers/dall-e-3.pdf, 2023, accessed 13 Jun 2024.

[96] T. Merino, S. Earle, R. Sudhakaran, S. Sudhakaran, and J. Togelius, "Making new connections: Llms as puzzle generators for the new york times' connections word game," *arXiv preprint arXiv:2407.11240*, 2024.

[97] M. Zammit, A. Liapis, and G. N. Yannakakis, "CrawLLM: Theming games with large language models," in *Proceedings of the IEEE Conference on Games*, 2024.

[98] A. Q. Jiang, A. Sablayrolles, A. Roux, A. Mensch, B. Savary, C. Bamford *et al.*, "Mixtral of experts," *arXiv preprint arXiv:2401.04088*, 2024.

[99] G. Todd, A. Padula, M. Stephenson, É. Piette, D. J. Soemers, and J. Togelius, "GAVEL: Generating games via evolution and language models," *arXiv preprint arXiv:2407.09388*, 2024.

[100] C. Browne, M. Stephenson, É. Piette, and D. J. N. J. Soemers, "A practical introduction to the Ludii general game system," in *Proceedings of the International Conference on Advances in Computer Games*, 2019.

[101] A. Liapis, "Searching for sentient design tools for game development," Ph.D. dissertation, Center for Computer Games, IT University of Copenhagen, Copenhagen, Denmark, 2014.

[102] G. Smith, J. Whitehead, and M. Mateas, "Tanagra: Reactive planning and constraint solving for mixed-initiative level design," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 3, no. 3, 2011.

[103] A. Liapis, G. N. Yannakakis, and J. Togelius, "Sentient Sketchbook: Computer-aided game level authoring," in *Proceedings of the International Conference on the Foundations of Digital Games*, 2013.

[104] P. Migkotzidis and A. Liapis, "SuSketch: Surrogate models of gameplay as a design assistant," *IEEE Transactions on Games*, vol. 14, no. 2, 2021.

[105] M. Charity, I. Dave, A. Khalifa, and J. Togelius, "Baba is Y'all 2.0: Design and investigation of a collaborative mixed-initiative system," *IEEE Transactions on Games*, 2022.

[106] R. Gallotta, K. Arulkumaran, and L. B. Soros, "Preference-learning emitters for mixed-initiative quality-diversity algorithms," *IEEE Transactions on Games*, 2023.

[107] M. Charity, Y. Bhartia, D. Zhang, A. Khalifa, and J. Togelius, "A preliminary study on a conceptual game feature generation and recommendation system," in *Proceedings of the IEEE Conference on Games*, 2023.

[108] M. G. Torii, T. Murakami, and Y. Ochiai, "Lottery and sprint: Generate a board game with design sprint method on AutoGPT," in *Companion Proceedings of the Annual Symposium on Computer-Human Interaction in Play*, 2023.

[109] D. Dhamani and M. L. Maher, "The tyranny of possibilities in the design of task-oriented LLM systems: A scoping survey," *arXiv preprint arXiv:2312.17601*, 2023.

[110] G. N. Yannakakis, A. Liapis, and C. Alexopoulos, "Mixed-initiative co-creativity," in *Proceedings of the International Conference on the Foundations of Digital Games*, 2014.

[111] J. Rezwana and M. L. Maher, "Designing creative AI partners with COFI: A framework for modeling interaction in human-AI co-creative systems," *ACM Transactions on Computer-Human Interaction*, vol. 30, no. 5, 2023.

[112] A. Liapis, G. N. Yannakakis, and J. Togelius, "Sentient World: Human-based procedural cartography," in *Proceedings of Evolutionary and Biologically Inspired Music, Sound, Art and Design Conference*, 2013.

[113] R. Barth, "The convergence of AI and creativity: Introducing Ghostwriter," https://news.ubisoft.com/en-us/article/7Cm07zbBGy4Xml6WgYi25d/the-convergence-of-ai-and-creativity-introducing-ghostwriter, accessed 12 June 2024.

[114] M. Guzdial and M. Riedl, "An interaction framework for studying co-creative AI," *arXiv preprint arXiv:1903.09709*, 2019.

[115] D. Novick and S. Sutton, "What is mixed-initiative interaction?" in *Proceedings of the AAAI Spring Symposium on Computational Models for Mixed Initiative Interaction*, 1997.

[116] A. Liapis, G. N. Yannakakis, and J. Togelius, "Designer modeling for personalized game content creation tools," in *Proceedings of the AIIDE Workshop on Artificial Intelligence & Game Aesthetics*, 2013.

[117] V. Vimpari, A. Kultima, P. Hämäläinen, and C. Guckelsberger, ""An adapt-or-die type of situation": Perception, adoption, and use of text-to-image-generation AI by game industry professionals," in *Proceedings of the ACM on Human-Computer Interaction*, vol. 7, 2023.

[118] J. Boucher, G. Smith, and Y. Telliel, "Examining early professionals' use of generative AI in the game development process," in *Proceedings of the AIIDE Workshop on Experimental Artificial Intelligence in Games*, 2023.

[119] A. Anjum, Y. Li, N. Law, M. Charity, and J. Togelius, "The ink splotch effect: A case study on ChatGPT as a co-creative game designer," in *Proceedings of the International Conference on the Foundations of Digital Games*, 2024.

[120] L. W. Stephen Peacock, "Unravelling Project AVA: Insights from our research on GenAI in game development (presented by Keywords Studios)," https://www.gdcvault.com/play/1034841/Unravelling-Project-AVA-Insights-from, 2024, accessed 13 Jun 2024.

[121] M. U. Nasir and J. Togelius, "Practical PCG through large language models," in *Proceedings of the IEEE Conference on Games*, 2023.

[122] J. Kelly, A. Calderwood, N. Wardrip-Fruin, and M. Mateas, "There and back again: Extracting formal domains for controllable neurosymbolic story authoring," in *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2023.

[123] V. Kumaran, D. Carpenter, J. Rowe, B. Mott, and J. Lester, "End-to-End procedural level generation in educational games with natural language instruction," in *Proceedings of the IEEE Conference on Games*, 2023.

[124] J. Rowe, E. Lobene, B. Mott, and J. Lester, "Play in the museum: Design and development of a game-based learning exhibit for informal science education," *International Journal of Gaming and Computer-Mediated Simulations*, 2017.

[125] R. Gallotta, A. Liapis, and G. N. Yannakakis, "LLMaker: A game level design interface using (only) natural language," in *Proceedings of the IEEE Conference on Games*, 2024.

[126] ——, "Consistent game content creation via function calling for large language models," in *Proceedings of the IEEE Conference on Games*, 2024.

[127] P. Taveekitworachai, F. Abdullah, M. F. Dewantoro, R. Thawonmas, J. Togelius, and J. Renz, "ChatGPT4PCG competition: Character-like level generation for science birds," in *Proceedings of the IEEE Conference on Games*, 2023.

[128] T. Brooks, A. Holynski, and A. A. Efros, "InstructPix2Pix: Learning to follow image editing instructions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[129] J. Zhu, A. Liapis, S. Risi, R. Bidarra, and G. M. Youngblood, "Explainable AI for designers: A human-centered perspective on mixed-initiative co-creation," in *Proceedings of the IEEE Conference on Computational Intelligence and Games*, 2018.

[130] A. Liapis, M. Awiszus, A. J. Champandard, M. Cook, A. Denisova, A. Dockhorn, T. Thompson, and J. Zhu, "Artificial intelligence for audiences," in *Human-Game AI Interaction (Dagstuhl Seminar 22251)*, D. Ashlock, S. Maghsudi, D. P. Liebana, P. Spronck, and M. Eberhardinger, Eds. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023.

[131] A. Canossa and A. Drachen, "Patterns of play: Play-personas in user-centred game development," in *Proceedings of DiGRA*, 2009.

[132] G. N. Yannakakis and D. Melhart, "Affective game computing: A survey," *Proceedings of the IEEE*, 2023.

[133] G. N. Yannakakis and J. Togelius, "Experience-driven procedural content generation," *IEEE Transactions on Affective Computing*, 2011.

[134] K. Pinitas, D. Renaudie, M. Thomsen, M. Barthet, K. Makantasis, A. Liapis, and G. N. Yannakakis, "Predicting player engagement in Tom Clancy's The Division 2: A multimodal approach via pixels and gamepad actions," in *Proceedings of the ACM International Conference on Multimodal Interaction*, 2023.

[135] D. Melhart, A. Liapis, and G. N. Yannakakis, "The Arousal video Game AnnotatIoN (AGAIN) dataset," *IEEE Transactions on Affective Computing*, vol. 13, no. 4, 2022.

[136] J. Broekens, B. Hilpert, S. Verberne, K. Baraka, P. Gebhard, and A. Plaat, "Fine-grained affective processing capabilities emerging from large language models," in *Proceedings of the International Conference on Affective Computing and Intelligent Interaction*, 2023.

[137] A. Liapis, "Artificial intelligence for designing games," in *The Handbook of Artificial Intelligence and the Arts*, P. Machado, J. Romero, and G. Greenfield, Eds. Springer, 2021, in print.

[138] N. M. M. Shafiullah, Z. J. Cui, A. Altanzaya, and L. Pinto, "Behavior transformers: Cloning k modes with one stone," in *Proceedings of the Neural Information Processing Systems Conference*, 2022.

[139] J. Pfau, A. Liapis, G. N. Yannakakis, and R. Malaka, "Dungeons & Replicants II: Automated game balancing across multiple difficulty dimensions via deep player behavior modeling," *IEEE Transactions on Games*, 2023.

[140] S. Johnson-Bey, M. Mateas, and N. Wardrip-Fruin, "Toward using ChatGPT to generate theme-relevant simulated storyworlds," in *Proceedings of the AIIDE Workshop on Experimental Artificial Intelligence in Games*, 2023.

[141] M. Kreminski, D. Acharya, N. Junius, E. Oliver, K. Compton, M. Dickinson, C. Focht, S. Mason, S. Mazeika, and N. Wardrip-Fruin, "Cozy mystery construction kit: Prototyping toward an AI-assisted collaborative storytelling mystery game," in *Proceedings of the International Conference on the Foundations of Digital Games*, 2019.

[142] C. Fernández-Vara and A. Thomson, "Procedural generation of narrative puzzles in adventure games: The Puzzle-Dice system," in *Proceedings of the Workshop on Procedural Content Generation in Games*, 2012.

[143] J. Dormans, "Adventures in level design: generating missions and spaces for action adventure games," in *Proceedings of the Workshop on Procedural Content Generation in Games*, 2010.

[144] S. Al-Nassar, A. Schaap, M. V. D. Zwart, M. Preuss, and M. A. Gómez-Maureira, "QuestVille: Procedural quest generation using NLP models," in *Proceedings of the International Conference on the Foundations of Digital Games*, 2023.

[145] V. Kumaran, J. Rowe, B. Mott, and J. Lester, "SceneCraft: Automating interactive narrative scene generation in digital games with large language models," *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2023.

[146] J. E. Laird and M. van Lent, "Human-level AI's killer application," *AI Magazine*, 2001.

[147] The AlphaStar team, "AlphaStar: Mastering the real-time strategy game StarCraft II," https://deepmind.google/discover/blog/alphastar-mastering-the-real-time-strategy-game-starcraft-ii/, 2019, accessed 10 July 2024.

[148] S. Milani *et al.*, "Towards solving fuzzy tasks with human feedback: A retrospective of the MineRL BASALT 2022 competition," in *Proceedings of the NeurIPS 2022 Competitions Track*, 2022.

[149] A. M. Smith, E. Andersen, M. Mateas, and Z. Popovic, "A case study of expressively constrainable level design automation tools for a puzzle game," in *Proceedings of the International Conference on the Foundations of Digital Games*, 2012.

[150] D. Karavolos, A. Liapis, and G. N. Yannakakis, "Learning the patterns of balance in a multi-player shooter game," in *Proceedings of the FDG Workshop on Procedural Content Generation in Games*, 2017.

[151] P. Manakul, A. Liusie, and M. J. F. Gales, "SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models," in *Proceedings of the Empirical Methods in Natural Language Processing Conference*, 2023.

[152] A. Strasser, "On pitfalls (and advantages) of sophisticated large language models," *arXiv preprint arXiv:2303.17511*, 2023.

[153] N. Bian, H. Lin, P. Liu, Y. Lu, C. Zhang, B. He, X. Han, and L. Sun, "Influence of external information on large language models mirrors social cognitive patterns," *arXiv preprint arXiv:2305.04812*, 2023.

[154] M. Karpinska and M. Iyyer, "Large language models effectively leverage document-level context for literary translation, but critical errors persist," in *Proceedings of the Machine Translation Conference*, 2023.

[155] Z. Gekhman, J. Herzig, R. Aharoni, C. Elkind, and I. Szpektor, "TrueTeacher: Learning factual consistency evaluation with large language models," in *Proceedings of the Empirical Methods in Natural Language Processing Conference*, 2023.

[156] J. Zhou, "An evaluation of state-of-the-art large language models for sarcasm detection," *arXiv preprint arXiv:2312.03706*, 2023.

[157] Y. Liu, T. Han, S. Ma, J. Zhang, Y. Yang, J. Tian, H. He, A. Li, M. He, Z. Liu, Z. Wu, L. Zhao, D. Zhu, X. Li, N. Qiang, D. Shen, T. Liu, and B. Ge, "Summary of ChatGPT-related research and perspective towards the future of large language models," *Meta-Radiology*, vol. 1, no. 2, 2023.

[158] D. Li, A. S. Rawat, M. Zaheer, X. Wang, M. Lukasik, A. Veit, F. Yu, and S. Kumar, "Large language models with controllable working memory," in *Findings of the Association for Computational Linguistics*, 2023.

[159] Gemini Team, "Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context," *arXiv preprint arXiv:2403.05530*, 2024.

[160] T. Munkhdalai, M. Faruqui, and S. Gopal, "Leave no context behind: Efficient infinite context transformers with infini-attention," *arXiv preprint arXiv:2404.07143*, 2024.

[161] Z. Fountas, M. A. Benfeghoul, A. Oomerjee, F. Christopoulou, G. Lampouras, H. Bou-Ammar, and J. Wang, "Human-like episodic memory for infinite context LLMs," *arXiv preprint arXiv:2407.09450*, 2024.

[162] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Proceedings of the Conference on Neural Information Processing Systems*, 2020.

[163] Valve Corporation, "Steam Hardware & Software Survey: May 2024," https://store.steampowered.com/hwsurvey/steam-hardware-software-survey-welcome-to-steam, accessed 12 June 2024.

[164] M. Yue, J. Zhao, M. Zhang, L. Du, and Z. Yao, "Large language model cascades with mixture of thoughts representations for cost-efficient reasoning," *arXiv preprint arXiv:2310.0309*, 2024.

[165] Y. Yu, Q. Zhang, J. Li, Q. Fu, and D. Ye, "Affordable generative agents," *arXiv preprint arXiv:2402.02053*, 2024.

[166] X. Miao, G. Oliaro, Z. Zhang, X. Cheng, H. Jin, T. Chen, and Z. Jia, "Towards efficient generative large language model serving: A survey from algorithms to systems," *arXiv preprint arXiv:2312.15234*, 2023.

[167] F. de la Torre, C. M. Fang, H. Huang, A. Banburski-Fahey, J. A. Fernandez, and J. Lanier, "LLMR: Real-time prompting of interactive worlds using large language models," *arXiv preprint arXiv:2309.12276*, 2023.

[168] D. Melhart, J. Togelius, B. Mikkelsen, C. Holmgård, and G. N. Yannakakis, "The ethics of AI in games," *IEEE Transactions on Affective Computing*, vol. 15, no. 1, 2023.

[169] A. A. Chien, L. Lin, H. Nguyen, V. Rao, T. Sharma, and R. Wijayawardana, "Reducing the carbon impact of generative AI inference (today and in 2035)," in *Proceedings of the SCS Workshop on Sustainable Computer Systems*, 2023.

[170] S. A. Khowaja, P. Khuwaja, and K. Dev, "ChatGPT needs SPADE (sustainability, PrivAcy, Digital divide, and Ethics) evaluation: A review," *arXiv preprint arXiv:2305.03123*, 2023.

[171] Free Law Project, "Authors Guild v. OpenAI Inc." https://www.courtlistener.com/docket/67810584/authors-guild-v-openai-inc/, accessed 27 Feb 2024.

[172] J. A. Rothchild, "Copyright implications of the use of code repositories to train a machine learning model," *Free Software Foundation*, 2022.

[173] HackerNews, "Github Copilot," https://news.ycombinator.com/item?id=27676266, 2021, accessed 27 Feb 2024.

[174] P. Zhang, G. Zeng, T. Wang, and W. Lu, "TinyLlama: An open-source small language model," *arXiv preprint arXiv:2401.02385*, 2024.

[175] U.S. Copyright Office Review Board, "Decision affirming refusal of registration of "A Recent Entrance to Paradise"," 2022.

[176] ——, "Registration decision on "Zarya of the Dawn"," 2023.

[177] C. E. Lamb and D. G. Brown, "Should we have seen the coming storm? Transformers, society, and CC," in *Proceedings of the International Conference on Computational Creativity*, 2023.

[178] Y. Liu, Y. Yao, J.-F. Ton, X. Zhang, R. Guo, H. Cheng, Y. Klochkov, M. F. Taufiq, and H. Li, "Trustworthy LLMs: a survey and guideline for evaluating large language models' alignment," in *Proceedings of the NeurIPS Workshop on Socially Responsible Language Modelling Research*, 2023.

[179] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large language models are zero-shot reasoners," in *Proceedings of the Advances in Neural Information Processing Systems Conference*, 2022.

[180] S. Chen, B. Li, and D. Niu, "Boosting of thoughts: Trial-and-error problem solving with large language models," *arXiv preprint arXiv:2402.11140*, 2024.

[181] D. Mondal, S. Modi, S. Panda, R. Singh, and G. S. Rao, "KAM-CoT: Knowledge augmented multimodal chain-of-thoughts reasoning," *arXiv preprint arXiv:2401.12863*, 2024.

[182] M. Turpin, J. Michael, E. Perez, and S. R. Bowman, "Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting," in *Proceedings of the Neural Information Processing Systems Conference*, 2023.

[183] S. Frieder, L. Pinchetti, A. Chevalier, R.-R. Griffiths, T. Salvatori, T. Lukasiewicz, P. C. Petersen, and J. Berner, "Mathematical capabilities of ChatGPT," in *Proceedings of the Neural Information Processing Systems Conference*, 2023.

[184] A. M. Karkaj, M. J. Nelson, I. Koutis, and A. K. Hoover, "Prompt wrangling: On replication and generalization in large language models for PCG levels," in *Proceedings of the International Conference on the Foundations of Digital Games*, 2024.

[185] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, and A. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2020.

[186] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[187] Y. Jia, W. Wu, F. Cao, and S. C. Han, "In-game toxic language detection: Shared task and attention residuals," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

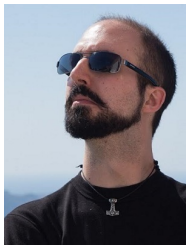
[188] J. Thomas, N. Chorakhalikar, A. Dantrey, R. R. Nalla, and P. Mehta, "Automatic classification and reporting of inappropriate language in online applications," US Patent 20 210 370 188, 2021.

Roberto Gallotta is a second-year Ph.D. student in Game Research at the Institute of Digital Games, University of Malta, researching dynamic evolutionary computation and mixed-initiative co-creation for the aide of video game designers. Prior to joining the Institute of Digital Games, he was a junior researcher at Araya Inc., Tokyo, for a year. He has published multiple papers, mostly focused on procedural content generation and evolutionary computation with games as a domain, and co-organized the "ALife for and from video games" workshop at ALIFE2023.

Graham Todd is a fourth-year PhD student in Computer Science in the Game Innovation Lab at NYU Tandon. His research focuses on the intersection of language games, and the ways in which people and algorithms can generate novel games and goals. He is particularly interested in what games can teach us about their players. He has published a number of papers on topics ranging from modeling the process of game generation with evolutionary algorithms to interrogating the ways that LLMs understand language through word games.

Marvin Zammit is a researcher at the Institute of Digital Games, University of Malta, where he is currently reading for a Ph.D. in Game Research. His research revolves around computational creativity, procedural content generation, evolutionary computation, quality diversity, and educational games. His primary interest lies in the application of machine learning algorithms in building practical pipelines for game development. He is also an experienced developer of games and interactive installations.

Sam Earle is a fifth-year PhD student in Computer Science in the Game Innovation Lab at NYU Tandon. His research focuses on open-ended learning in terms of the automatic generation of diverse playable game environments, and the training of robust embodied agents within these games. He has also investigated using foundation models to enable text-guided environment generation, with an eye toward steering ever-complexifying environment- and agent-generation loops toward human-relevant content.



Antonios Liapis is an Associate Professor at the Institute of Digital Games, University of Malta, where he bridges the gap between game technology and game design in courses focusing on human-computer creativity, digital prototyping and game development. He received the Ph.D. degree in Information Technology from the IT University of Copenhagen in 2014. His research focuses on Artificial Intelligence in games, human-computer interaction, computational creativity, and user modeling. He has published over 140 papers in the aforementioned fields, and has received several awards for his research contributions and reviewing effort. He serves as Associate Editor for the IEEE Transactions on Games, and has served as general chair in four international conferences, as guest editor in five special issues in international journals, and has co-organized 15 workshops.

Julian Togelius is an Associate Professor in the Department of Computer Science and Engineering at New York University, and director of the NYU Game Innovation Lab. He is also a co-founder and research directo of the game AI company modl.ai. Julian's research focuses on games for AI and AI for games; for example, procedural content generation via reinforcement learning, open-ended learning in generative environments, and LLM-guided game creation. Julian got his PhD in Computer Science in 2007 from the University of Essex in England, and he has also worked in Switzerland and Denmark despite being Swedish. He was Editor-in-Chief of IEEE Transactions on Games from 2018 to 2021.

Georgios N. Yannakakis (F'24) is a Professor at the Institute of Digital Games, University of Malta (UM) and a co-founder of modl.ai. He received the PhD degree in Informatics from the University of Edinburgh in 2006. Prior to joining the Institute of Digital Games, UM, in 2012 he was an Associate Professor at the Center for Computer Games Research at the IT University of Copenhagen. He does research at the crossroads of artificial intelligence, computational creativity, affective computing, advanced game technology, and human-computer interaction. He has published more than 350 papers in the aforementioned fields and his work has been cited broadly. His research has been supported by numerous national and European grants (including a Marie Skłodowska-Curie Fellowship) and has appeared in *Science Magazine* and *New Scientist* among other venues. He is currently the Editor-in-Chief of the IEEE Transactions on Games and an Associate Editor of the IEEE Transactions on Evolutionary Computation. Georgios is an IEEE Fellow.