Creative Problem Solving in Large Language and Vision Models – What Would it Take?

Lakshmi Nair
Georgia Institute of Technology
Atlanta, GA, USA

Evana Gizzi
Tufts University
Medford, MA, USA

Jivko Sinapov
Tufts University
Medford, MA, USA

Abstract

We advocate for a strong integration of Computational Creativity (CC) with research in large language and vision models (LLVMs) to address a key limitation of these models, i.e., creative problem solving. We present preliminary experiments showing how CC principles can be applied to address this limitation. Our goal is to foster discussions on creative problem solving in LLVMs and CC at prestigious ML venues.

1 Introduction

Creativity is "*...the ability to come up with an idea which, relative to the pre-existing domain-space in one's mind, one could not have had before. Whether any other person (or system) has already come up with it on an earlier occasion is irrelevant.*" (Boden, 1998), p.216. For artificial agents, Computational Creativity (CC) is a multi-disciplinary field (spanning Philosophy, Psychology, Neuroscience, and Computer Science) that seeks to develop computational methods capable of generating creative outcomes reminiscent of creative processes in humans (Gizzi et al., 2022). Within CC, *creative problem solving* is a sub-area that requires an agent to discover – from *its* perspective – novel and previously unseen ways to accomplish a task. For example, in the absence of a ladle to scoop ingredients, an agent might creatively choose to substitute a bowl in place of the ladle. In this sense, creative problem solving encompasses creativity that is specifically task-oriented, as opposed to the generation of creative artifacts e.g., music or images.

While recent state-of-the-art large language models (LLMs) and vision-language models (VLMs) have demonstrated competency in artistic endeavours (Rombach et al., 2021; Copet et al., 2023), creative problem solving continues to be a shortcoming of these models (we use LLVM to denote the umbrella of *both* LLMs and VLMs). For instance, in Bubeck et al. (2023), the authors point out that

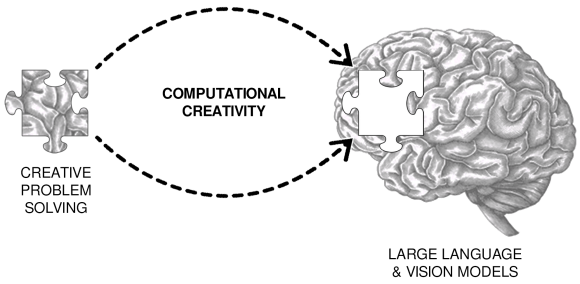

Figure 1: Computational Creativity can help address a gap in the intelligence of present-day LLVMs, elevating their ingenuity through creative problem solving.

"discontinuous tasks" that require a certain "Eureka" idea, i.e., creative problem solving, is currently a limitation of models like GPT-4. Similar observations have been made in follow up work showing that state-of-the-art LLMs inherently possess poor creative problem solving capabilities compared to humans (Tian et al., 2023; Naeini et al., 2023).

Given this obvious limitation, ongoing research in Machine Learning should seek to address the gap between LLVMs and creative problem solving, to further enhance the intelligent capabilities of these models. We believe that a discussion of Computational Creativity is essential to addressing this limitation. It is our position that **Machine Learning and Computational Creativity should be strongly integrated in research to enable effective creative problem solving in LLVMs and push the frontiers of their ingenuity.**

Creative problem solving can be a resourceful skill for artificial agents. As defined in prior work, "*Intelligence is the ability to work and adapt to the environment with insufficient knowledge and resources.*" (Pennachin and Goertzel, 2007), p.10. Demonstrated in hallmark examples of human ingenuity, like the makeshift CO_2 filter built onboard the Apollo-13 (Cass, 2005), or the makeshift medical devices used to offset equipment shortages during COVID-19 (Turner et al., 2020), creative prob-

Findings of the Association for Computational Linguistics: EACL 2024, pages 11978–11994

lem solving is especially important when dealing with resource-critical scenarios. Since humans tend to "choke" under high pressure situations (DeCaro et al., 2011) often limiting their problem solving skills, autonomous agents equipped with LLVMs that have similar capabilities would be highly assistive and transformative to humans in high-stake environments. These include situations like rescue missions (BBC, 2012) or autonomous operation in human-inaccessible environments (e.g., space or underwater exploration) with limited resources (Atkeson et al., 2018). However, the exceptional degree of creative problem solving necessary for such assistance remains beyond the scope of LLVMs today, limiting their intelligence (See Appx. B.1).

2 Two Cultures Problem: Why does CC not receive a wider reception in ML?

Even though creative problem solving (CPS) is a shortcoming of existing LLVMs, Computational Creativity seldom finds its way into mainstream ML research. We believe this discrepancy aligns with the "two cultures" problem (Hammond et al., 2013) (also corroborated in Van Heerden and Bas (2021); Lahikainen et al. (2024)), and is motivated by three aspects of CC literature as it relates to creative problem solving: a) the lack of a precise definition of CPS makes it challenging to identify how existing approaches in LLVMs are deficient in CPS skills; b) the somewhat "abstract" computational descriptions of CPS in Computational Creativity is challenging to connect to practical algorithms in LLVMs; and c) the lack of standardized benchmarks make it harder to evaluate LLVMs for CPS. In our discussions relating to a) in Section 3.1, b) in Section 4, and c) Section 5, we hope to address these gaps and encourage the ML community to think about how LLVMs can be augmented with creative problem solving skills *through a deeper discussion of Computational Creativity.*

To emphasize the applicability of principles from CC for creative problem solving in LLVMs, we discuss the seminal work of Margaret A. Boden from CC literature that introduces three forms of creativity, namely, "*exploratory*", "*combinational*", and "*transformational*" (Boden, 1998). Prior work has discussed the extension of Boden's forms of creativity to creative problem solving in AI (Gizzi et al., 2022), however, their work does not include recent advances in LLVMs nor how Boden's principles can be extended to specific approaches for LLVMs.

Ongoing discussions by leading ML experts like Dr. Shane Legg, co-founder of DeepMind, have suggested that "search" could help such models perform creative problem solving, quote, "*... these foundational models are world models of a kind, and to do really creative problem solving, you need to start searching*" (Patel, 2023). There has also been speculation that OpenAI's Q^* search (described as a "significant breakthrough" in popular media) could be targeting a similar approach (Wang, 2023; Anna Tong and Hu, 2023). Interestingly, we note that "search" as described here, can be linked to Boden's proposed "exploratory" approach (Section 4.1.1). However, in Section 4, we posit that "combinational" and "transformational" modes should be equally emphasized to achieve creative problem solving in LLVMs.

Although we choose to expand on Boden's work as the focal point to drive our arguments in the main paper, it is not the only theory in CC that is relevant to this discussion. For completeness, we elaborate on additional CC theories and their applicability to creative problem solving in LLVMs in Appx. B.

3 From Task Planning to Creative Problem Solving

Creative problem solving can be broadly described as the process through which agents *discover* novel ways of accomplishing a task that was unsolvable prior to the discovery. Computationally, creative problem solving can be achieved through planning, learning, or hybrid approaches (Gizzi et al., 2022). Following a review of the different definitions of creative problem solving that have been proposed (Appx. A), we believe the following most closely connects to existing formalisms in ML.

3.1 Definition of Creative Problem Solving

Gizzi et al. (2022) define the notion of a *concept*, as a state (of the environment and/or agent) or action. More generally, the authors denote C_X as the set of all concepts relating to X (X denotes environment states S or actions A). Hence, C_S denotes the set of all environmental states, and C_A denotes the set of agent actions. Formally, the authors state their definition as (Page 7, (Gizzi et al., 2022)):

Given an un-achievable goal due to an insufficient conceptual space, CPS refers to the process by which the agent discovers a new conceptual space $C'_X \nsubseteq C_X$*, such that* $C'_X = f(C_X)$ *is the result of applying some function* f *on the current conceptual*

space, enabling the agent to solve the previously unsolvable task by using C'_X.

As a simplified example, let us assume a robot that has a goal G of transferring beans from a jar to a cooker: G ={in(beans, cooker)}. Here, the initial state is defined as C_S ={in(beans, jar), $hasContainability$(spoon)}. Let the actions be defined as C_A ={$scoop$(beans, X, loc_s, loc_d)}. Here, X refers to an object that satisfies $hasContainability(\cdot)$, for example a spoon, to scoop beans from loc_s to loc_d. If the robot has access to a spoon, the robot can use it to scoop the beans from the jar to the cooker to meet the goal. However, what if the robot did *not* have a spoon, but had a *glass* instead? By the definition of C_S, the agent is unaware that $hasContainability$(glass) is true, making the goal un-achievable. By our definition, creative problem solving is the process by which the agent uses some function $f(\cdot)$ to discover a new conceptual space: $f(C_S) = C'_S = C_S \cup \{hasContainability$ (glass)$\}$. This would allow the agent to solve the previously unsolvable task by using the glass to scoop the beans instead.

In essence, CPS arises when the agent uses what it knows, to discover something new and the newly discovered knowledge is applied to solve a previously impossible task. Boden's three forms of creativity that we focus on in this paper, denote three plausible functions for $f(C_X)$. We revisit the notion of conceptual spaces in Section 3.

In the remainder of this section, we discuss how typical task planning is achieved with LLVMs. We divide the discussion into three subsections based on the level of task planning abstraction where LLVMs are applied: a) high-level task planning, b) low-level task planning, and c) hybrid task planning. While not exhaustive, our review is meant to offer a general insight into how LLVMs are used for task planning, to identify entry points for introducing creative problem solving capabilities.

3.2 LLVMs for high-level task planning

Approaches for high-level task planning often involve using LLVMs to identify high-level goals for accomplishing a task. Some approaches to task planning with LLMs often take a user input specifying the task, and generate high-level task plans for accomplishing it. These approaches often use LLMs as a form of "knowledge base", to extract actionable task plans from the models via appropriate prompting (Huang et al., 2022), further iterating over the generated task plan with repeated calls to the LLM as needed (Prasad et al., 2023).

In the context of Reinforcement Learning (RL), prior work has focused on using LLMs to suggest high-level goals for an RL agent (Du et al., 2023). Dubbed as ELLMs (Exploring with LLMs), an RL agent provides its current state to an LLM via a prompt, and receives a goal suggestion from the LLM that is then used to shape the reward and the agent exploration. Further work has extended this approach to incorporate the use of experience memory (Zhang et al., 2023a). Existing approaches have also used LLMs to generate directed acyclic graphs composed of sub-goal states to aid the exploration of an RL agent (Shukla et al., 2023).

3.3 LLVMs for low-level task planning

Approaches for low-level task planning involve using LLMs to generate low-level code for performing a task. In contrast to high-level planning, where high-level goals and sub-goals are generated, these approaches use LLMs to directly generate low-level execution code via appropriate API calls (Liang et al., 2023). Other approaches have also investigated the capacity of LLMs to generate task plans via a low-level planning language such as PDDL (Silver et al., 2023), including iterating over the generated plan descriptions in case of errors (Guan et al., 2023). In terms of low-level planning using VLMs, prior work has introduced an approach that uses a diffusion model to generate robot trajectories conditioned on language and the current visual state of the robot (Chen et al., 2023).

3.4 Hybrid high and low-level planning with LLVMs

Hybrid approaches use LLVMs both for high-level goal generation as well as low-level planning. For instance, in Li et al. (2023), user inputs are passed as LLM prompts to generate high-level plans. The high-level plans are then converted to low-level plans for robot execution via LLMs specialized for coding. Other approaches have used a high-level LLM planner, a VLM perceiver, and a low-level LLM planner for re-planning with both visual and language inputs (Skreta et al., 2024).

3.5 Summary

Given this overview, we see that LLVMs both at the high-level and low-level, can be modified to incorporate creative problem solving into task planning. For instance, the high-level task plans generated can encompass a novel substitution for a missing

object, whereas the low-level task plan can generate an appropriate trajectory for creatively using the object. While the above approaches could, in principle, be studied within the framework of creative problem solving, that is not usually how the problem is formulated; there is a lack of paradigms for studying creative problem solving beyond just, *"do you solve the problem or not?"*. Creative problem solving needs a fundamental rethinking of the typical problem formulations and approaches in ML. The next section is aimed at ways in which ML approaches in LLVMs can be reformulated from the perspective of CC.

4 Augmenting LLVM embedding spaces for creative problem solving

In this section, we discuss how principles from CC can be extended to LLVMs for creative problem solving. We begin with Boden's definition of "conceptual spaces" as "*[conceptual space] is the generative system that underlies the domain and defines a certain range of possibilities: chess moves, or molecular structures, or jazz melodies*" (Boden, 2005), p.18 and "*... in short, any reasonably disciplined way of thinking*" (Boden, 1998), p.214. By this definition, the embedding space of an LLVM describes its conceptual space or "*its way of thinking*". Some evidence for this also comes from existing work that introduces an approach for enabling LLMs to interpret continuous embedding spaces via natural language. Given an embedding vector representing an interpolation of different concepts, the model is able to interpret a text prompt in the context of the supplied embedding (Tennenholtz et al., 2023). The embedding thus determines the model's way of thinking. Hence, a discussion of enabling creative problem solving in LLVMs should target their embedding space. To this end, we explore two questions: a) *how* can LLVM embedding spaces be augmented to achieve creative problem solving, and b) *what* information should they be augmented with? Aligning with our original position, we show that CC literature can offer insights into these questions.

4.1 How can LLVM embedding spaces be augmented?

In this section, we draw parallels between Boden's three forms of creativity and existing approaches in LLVMs. We further elaborate on how the three forms of creativity may enhance the potential of LLVMs to perform creative problem solving. We note that the ML approaches discussed in this section do not specifically perform creative problem solving. However, we discuss how they could potentially be extended to do so, by leveraging references from the CC literature.

4.1.1 Exploratory Creativity

Exploratory approaches involve exploration within the conceptual or equivalently, the embedding space of the model, and most closely relates to "search". Note that the term "exploration" here differs from its usage in RL, instead referring to exploration through the model's *embedding space*. Several existing approaches in the ML literature involve searching the *output space* of LLMs with the goal of improving the performance of these models. The "tree-of-thought" model generates a "tree" of next possible LLM outputs, and searches through the states via Breadth-first or Depth-first search to reach the desired goal state, often guided by heuristics (Yao et al., 2023). Numerous other approaches have built upon a similar strategy, such as using Monte-Carlo Tree Search (MCTS) (Zhou et al., 2023; Feng et al., 2023), beam search (Zhang et al., 2023b) or integrating pruning to remove subpar candidates (Golovneva et al., 2023).

Extension of exploratory creativity to LLVMs: An important point to note here is that these approaches involve searching exclusively within the *output* "solution space" of the LLMs rather than *directly* operating in the *embedding space* itself. In contrast to operating in the solution space of the LLM, exploratory approaches directly within the LLMs' embedding space would not be limited by what the LLM can generate as output – "*Some exploration merely shows us the nature of the relevant conceptual space that we had not explicitly noticed before*" (Boden, 2005), p.18. To effectively reveal the full extent of the conceptual space for creative problem solving, the approach should not be limited by the outputs the LLVM can generate. Rather, the generated (creative) outputs itself should be the result of heuristic or non-heuristic based search within the model's embedding space. However, to the best of our knowledge current approaches have not focused on LLVMs from this perspective, and have also not applied search to embedding spaces of Vision-LMs. Regardless, exploratory approaches are still limited by the dimensions of the model's embedding space. "*To overcome a limitation in the conceptual space, one must change it in*

some way" (Boden, 2005), p.18 - this leads us to combinational and transformational creativity.

4.1.2 Combinational Creativity

Combinational approaches involve combining two concepts to create something new - "*A novel combination of two familiar ideas is something which did not happen before.*" (Boden, 1998), p.213. We can broadly translate this to a function that takes in multiple concepts within an LLVM's embedding space to output a novel concept.

One way of extending this definition to LLVMs involves applying cross-attention layers. The attention operation is defined as (Vaswani et al., 2017):

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V,$$

where, Q, K and V denote query, keys and values respectively, and d_k denotes the dimensionality of the keys. Cross-attention involves passing K and V from a *different* model, e.g., in Flamingo (Alayrac et al., 2022), the keys and values represent visual input (from a separate vision encoder) and queries represent a language input. By applying cross attention in this manner, the embedding space of a model can be extended with capabilities of another model. In Bansal et al. (2024) the authors show that using cross-attention layers can help augment an *anchor* LLM with an *augmenting* LLM's capabilities to perform a task that the anchor LLM was incapable of achieving before - hinting at some creative possibilities of this method.

Other approaches in LLVMs, while using "combinations" in some way, do not conform to the notion of *combinational creativity*. This includes, for instance, approaches that perform arithmetic combination of LLM weights to enhance the model performance (Matena and Raffel, 2022; Ilharco et al., 2022). Or approaches that combine image and text embeddings via concatenation (Kim et al., 2021) or a scaled dot product at the output (Radford et al., 2021). While these approaches may be useful in imparting multi-modal capabilities, however, they do not lead to combinational creativity since the combination occurs *external* to the models as opposed to within the model's embedding space.

Extension of Combinational Creativity to LLVMs: The ML approaches described here involve combining embedding spaces across models. Existing approaches have not looked at combining concepts *within* the *same* model's embedding space. The extension of combinational creativity to LLVMs is much more apparent in the sense of *conceptual blending* (Fauconnier and Turner, 2003) for generation of creative artifacts, e.g., via blending of artistic styles. However, the extension of combinational creativity to creative problem solving is less obvious, and CC literature offers us further insights for making this connection. Typical conceptual blending corresponds to a form of "aesthetic combination", whereas creative problem solving would benefit from "functional combinations" (Chen et al., 2018). Functional combination combines the functions (as opposed to aesthetic) of two components, e.g., a coin combined with pliers could function as a makeshift screwdriver. The authors extend this framework to a combination of two nouns with a "base" noun (e.g., "pliers") and "additive" noun (e.g., "coin"). An interesting possibility stems from this notion: Can a combination of embeddings of the same LLVM, corresponding to "base" and "additive" nouns (perhaps with some prior denoting the task), enable the LLVM to generate creative combinations of objects for solving a task? This question remains unexplored, and points to a potential research direction for LLVMs inspired by CC.

4.1.3 Transformational Creativity

Transformational approaches involve transforming existing conceptual spaces to produce new ones. Transforming conceptual spaces can involve "*altering existing rules*" (Boden, 1998), p.216. One way of transforming the embedding space of a model involves fine-tuning or training (Franceschelli and Musolesi, 2023). However, additional insight into transformational creative problem solving comes from prior work in CC, which describes creative problems as those with a poorly defined structure where a solution is not immediately apparent (Olteteanu, 2014). And in such cases, "*... re-representation being the process which transforms an ill-structured problem into a well-structured one with direct inference to a problem solution*" (Olteteanu, 2014), p.1. The notion of "re-representing" or "redefining" the problem can be best captured in the input prompts provided to an LLVM. This most closely connects to *prompt engineering* and *in-context learning* (ICL).

Prompt engineering augments LLVMs with task-specific hints, called prompts, to adapt the LLVM to new tasks (Gu et al., 2023). Relatedly, in-context learning is a prompting method that provides the LLVM with instructions to solve a new task with-

out requiring additional training. Previous work has shown that in-context learning and gradient-based optimization are equivalent (Von Oswald et al., 2023), thus connecting ICL to training or fine-tuning.

Extension of transformational creativity to LLVMs: Task re-representations for creative problem solving, through prompting or ICL, have not been well explored within ML. Prompt engineering and ICL is a challenging task, since model performance is strongly dependent on the chosen prompts (Rubin et al., 2021), further compounded by the fact that creative problems are inherently poorly defined (Olteteanu, 2014). However, useful insights can be derived from CC literature. For instance, regarding problems that require creatively re-purposing objects, the *Object-replacement-object-composition* (OROC) framework (Olteţeanu and Falomir, 2016) illustrates re-representations of tasks, which can be translated into prompts. The paper defines three different types of creative tasks involving objects, and their task re-representations as (from (Olteţeanu and Falomir, 2016), p.16):

1. Replace an unfound object needed for a task with other objects present in the environment: *"If I do not have an object X, which I would normally use because of its affordance*[1] Af_X *, what other object Y could I use, so that I can get a similar affordance,* $Af_X \approx Af_Y$*?"*

2. Compose objects. *"If I do not have object X with affordance* Af_X *, which objects* $Y_1; Y_2; ...; Y_n$*, could I use to construct* X *or an object* X' *with an equivalent or similar affordance,* $Af_X \approx Af_{X'}, Af_X \approx Af_{Y1} + Af_{Y2} + ... + Af_{Yn}$*?"*

3. Decompose objects. *"If I do not have an object* X *with affordance* Af_X *, which objects* $Y_1; Y_2; ...; Y_n$ *which are components of object* Y *could I use to obtain an object* Y_i' *with an equivalent or similar affordance,* $Af_X \approx Af_{Y'i}$*?"*

For task re-representation, affordances can refer to object properties that are relevant to the task, e.g., in some cases the shape may be relevant and in other cases, the material (Olteţeanu and Falomir, 2016). Within LLVMs, the affordances Af_X or Af_Y can be defined via natural language or other modalities such as images. In the following section, we present preliminary experiments on using LLVMs for object replacement, with prompts that are inspired by the above task re-representations. However, an in-depth application of these re-representations as defined in CC to in-context learning in LLVMs remains unexplored.

[1]Affordance is defined as the relation between an agent, action and object, e.g., bowls have the "contain" affordance for humans.

Model	Acc. % (no creativity)
CLIP-B-32	100.0%
CLIP-B-16	92.0%
CLIP-L-14	98.0%
CLIP-H-14-laion	98.0%
ViLT-B-32	68.0%
LLaVA	98.0%

Table 1: Accuracy of the models in predicting the nominal use of objects with no creativity involved.

4.1.4 Summary

In the previous sections, we drew parallels between Boden's three forms of creativity and approaches in LLVMs, further emphasizing how principles from CC can potentially help enable creative problem solving skills in these models.

Integration with task planning: Given the three methods, we see that transformational and combinational approaches may be especially aligned with LLVMs for high-level task planning. In contrast, exploratory methods may be suited to low-level planning, e.g., trajectory generation.

Creative problem solving as a combination of the three methods: An effective approach to creative problem solving may require all the three methods described in this section. While papers have explored chaining of LLMs within frameworks (often via prompts) (Karpas et al., 2022; Ling et al., 2023), the individual LLMs themselves do not exhibit the characteristics described here. Existing frameworks in CC have shown that achieving creative problem solving would take a combination of all three methods, each of which is triggered in different contexts (Olteteanu, 2014). This presents potential opportunities for ML approaches that develop frameworks using multiple LLVMs, e.g., extending CC frameworks such as "*CreaCogs*" (Olteţeanu and Falomir, 2016) can be highly beneficial for productive developments in ML.

4.2 What information should LLVM embeddings be augemented with?

In the previous section, we discussed three methods for augmenting LLVM embedding spaces. In this section, we explore the question: "What information should be targeted by the three methods when augmenting the embedding space for creative problem solving?". In the previous section, we discussed this in the context of OROC. According to the OROC framework (Olteţeanu and Falomir, 2016), information about object affordances could enable models to re-represent the task, such that the solution becomes evident. We propose a small experiment to validate whether the principles of transformational creativity of OROC are useful to LLVMs. We note that creativity can occur in various contexts, e.g., creatively solving a math problem or creatively playing a chess move, each of which would require different information. However, to facilitate the discussion in this paper, we focus our scope on tasks that require innovatively replacing missing objects (OROC Task #1).

Note on embeddings vs. concepts: Our work connects "conceptual spaces" (or "concepts") as defined in Computational Creativity literature, to "embedding spaces" (or "embeddings") as defined in typical LM literature. We use "concepts" and "embeddings" interchangeably in this context. We make this connection to note that existing methods in Computational Creativity that operate on conceptual spaces translate to ML algorithms that operate on the LM's embedding space. In this section, we connect the concept of "affordances" to the "embeddings" of the LLVMs in our experiments. Our goal is to show how the model can be prompted via an approach inspired by transformational creativity, to connect affordances of two seemingly distinct objects, e.g., a bowl and a spoon that appear distinct, but share the containability affordance.

4.2.1 Experiment Setup

We create a simple experiment setup that tests the "object replacement" principle from OROC, where we create test sets composed of images of objects for replacing one of five core objects: "Scoop", "Hammer", "Spatula", "Toothpick", and "Pliers". We create two groups of tests: a) a nominal group where the actual object itself is available in each test set and requires no replacement (which serves as a form of baseline) and b) an object replacement group, where the nominal tool is missing, and a creative replacement object should be chosen.

For each group, we create test sets with 4 objects each, chosen from a set of RGB images of 16 objects (Appendix Figure 3). We create 10 test sets per core object (total 50 samples per model). Each test set only includes one ground truth object, along with three other random objects that will not suit as an appropriate replacement. In the nominal group, the ground truth is the actual object itself. In the object replacement group, the replacements are chosen based on self-assessment of the authors as (core object → replacement): "Scoop" → "Bowl"; "Hammer" → "Saucepan"; "Spatula" → "Knife"; "Toothpick" → "Safety pin"; "Pliers" → "Scissors". For each test case, we pass the images in the test set along with a prompt. We record whether the ground-truth object image was chosen by the model for the prompt (i.e., assigned highest output probability)[2].

The nominal group is subjected to one type of prompt: "*Can this object be used as a* $\langle core\_object \rangle$?". In the object replacement group, each test case is subjected to four types of prompts:

1. Baseline (regular) prompt: the same prompt as used in the nominal cases to obtain a baseline.

2. Prompt prepended with affordance information: the prompt includes additional information about the desired object affordances specified as object features.

3. Prompt prepended with task information: the prompt includes additional information about the desired task.

4. Prompt prepended with task and affordance information: the prompt includes additional information on the task and object affordance.

Case #2 aligns with task re-representations of OROC, and we explore cases #3 and #4 for comparison. We formulate our affordance prompts as brief versions of OROC's task re-representations. According to Olteţeanu and Falomir (2016) affordances can be defined using shape features, which we apply to the prompts here. The full set of prompts is shown in Appendix Table 2. The models that we explore include versions of CLIP (Radford et al., 2021), LLaVA (Liu et al., 2024), and ViLT (Kim et al., 2021) obtained

[2]CLIP generates probabilities that given images correspond to a text. ViLT and LLaVA respond with a text, and we assess if the model responded "yes" with a high probability for the ground truth.

from HuggingFace. We use different model sizes (Base, Large, Huge) and patch sizes (14, 16, 32). The open-source code for reproducing our experiment results (including our dataset and test cases) is available at: https://github.com/lnairGT/creative-problem-solving-LLMs. Appendix C includes more details on the experiments.

4.2.2 Results

In Table 1, we see the performances of the different models in the nominal test group, where the object requires no creative replacement. The models perform $> 90\%$ in such cases (except for ViLT). In Figure 2, we see the performances (accuracy shown on a $0.0 - 1.0$ scale) of the models in the object replacement test cases, where the object requires a creative replacement. For reference, a model that randomly picks an object achieves about 30% overall accuracy. Figure 2 shows average accuracies for the different prompting strategies across random test sets. From Table 1 to Figure 2 ("regular"), the models perform poorly when they need to creatively reason about object replacements, highlighting their limitation. Comparing the "Regular" tab in Figure 2 to "Affordance", we see a general improvement in model performances, when object affordance information is provided, consistent with description of the OROC framework (Olteţeanu and Falomir, 2016). However, information about the task (Figure 2, "Task") leads to mostly detrimental results. Information about task *and* affordances (Figure 2, "Task + Affordance") does not lead to substantial improvements either, and is also detrimental in certain cases. We note that there is quite a variance in performances across the different models, which may be partially attributed to the original training datasets of the models. These observations warrant further exploration beyond the scope of this paper. Appendix D includes a detailed, class-wise breakdown of the results.

4.2.3 Summary

While the experiments that we conducted are only preliminary, they offer some validity that the extension of principles in Computational Creativity can help overcome limitations of LLVMs in creative problem solving. The notion of task re-representation via improved prompting warrants further investigation in LLVMs, with regards to how the prompts can be generated automatically based on the creative task.

The models used in our experiments have all been trained jointly in visual and text domains. Multi-modal prompting capabilities may be useful for achieving creative problem solving. It can be quite challenging to describe affordances in words (example of "hammers" in our tests) and they may be better described through other means, e.g., images or depth maps or spectral data for material properties (Erickson et al., 2020). This would require the application of multi-modal LLVMs that can process a variety of data types (Girdhar et al., 2023; Han et al., 2023). Computational creativity can offer insight into meaningful representations of these different modalities that would help achieve creative problem solving, e.g., whether object material or shape matters more for one task vs. another (Olteţeanu and Falomir, 2016).

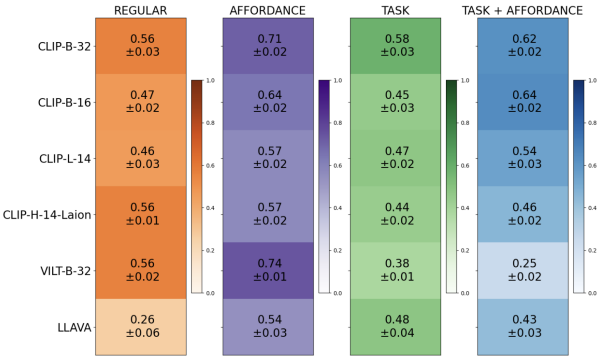

Figure 2: Object replacement group: Average accuracies and standard deviations of the models across ten different sets of randomly chosen objects.

It is also worth noting that the creative problem solving examples in our experiments are human-centric. For instance, robots may not have similar capabilities as humans to manipulate bowls for scooping. In such cases, LLVMs need to account for the affordances as described *with respect to the agent*, to derive creative solutions. However, that adds another level of complexity, yet to be explored, since these models are typically trained on human-centric data.

5 Evaluation of Creativity

An important discussion in the context of creative problem solving is *how can creative problem solving be evaluated?*. Prior work has proposed that creativity necessitates both *novelty* and *value* (Boden, 1998; Runco and Jaeger, 2012), where the former guarantees that the generated outputs of a creative process are original, and the latter ensures that the generated outputs are useful. In the context of CPS, novelty refers to the discovery of new con-

cepts (as defined in section 3.1), while value insists that the newly discovered concepts successfully solve the task. Hence, CPS benchmarks should specifically evaluate *how* the task was solved (novelty and value) rather than the typical ML evaluation of whether *the task was successful or not* (value only). Some existing approaches that make this distinction describe problem settings that can be used to measure CPS skills of LLMs through the implicit integration of novelty and value measurements (Tian et al., 2023; Naeini et al., 2023; Bisk et al., 2020; Talmor et al., 2022). In Tian et al. (2023), the authors create a dataset of 1600 real-world problems that necessarily involve creative reasoning abilities. Their proposed benchmark involves identifying novel approaches that can accomplish the given task (value). Similarly, in Naeini et al. (2023), the authors introduce the Only-Connect-Wall (OCW) dataset to measure CPS capabilities of LLMs. The authors in (Bisk et al., 2020) explore physical commonsense reasoning that is more generally applicable, beyond object-based creative problems. The authors introduce Physical Interaction: Question Answering, or PIQA consisting of 16,000 QA pairs where each question is paired with two possible common-sense solutions with a ground truth. In (Talmor et al., 2022), the authors introduce CommonSenseQA 2.0 (CSQA2) dataset consisting of both object-based and non-object based creative problems. The dataset consists of 14,343 questions distributed across 1,868 distinct topics. Currently, to the best of our knowledge, there are no standard benchmarks available to measure CPS skills of VLMs, although our preliminary experiments show one way to measure this using the task of object substitution.

6 Conclusion and Future Work

In this paper, we argued that an effective approach for enabling creative problem solving – currently a key limitation of LLVMs – should derive from Computational Creativity literature. To emphasize this at each juncture, we discussed the specific principles from CC that can be extended to achieve creative problem solving in LLVMs, describing the potential for further research with these insights.

It is rare to see special tracks or workshops targeted at Computational Creativity within more prestigious ML conferences. These programs typically focus on creative artifact generation and art (such as the NeurIPS *Workshop on Machine Learning for Creativity and Design* (NeurIPS, 2022) or the recent tutorial at EMNLP on Creative Natural Language Generation (Chakrabarty et al., 2023)), but do not discuss CPS, thus failing to bridge the gap between CC and ML. We hope to see a deeper integration of the CC communities at such strong ML venues. We hope to encourage the reader to view creative problem solving and ML holistically, through the lens of Computational Creativity.

7 Limitations

Literature outside of Computational Creativity that enables CPS is unexplored: Our paper predominantly focuses on CC literature. This work does not cover literature beyond CC that can potentially inform creative problem solving in LLVMs. Although CC literature broadly encompasses psychology, neuroscience and philosophy, our future work seeks to explore specific literature within these subdomains and discuss their applicability to creative problem solving and ML.

Lack of an explicit creative problem solving algorithm for LLVMs: Since the scope of our work aligns with a position paper, we have not focused on developing a concrete algorithm for creative problem solving in LLVMs. The prompting strategies explored in our preliminary experiments are manually specified, and our work does not elaborate on how these prompts may be automatically discovered. While our paper seeks to address some of the key gaps that prevent the application of CC literature to ML, there are still several unanswered questions when it comes to the practical implementation of an ML approach: e.g., what is a good representation for concepts that facilitate creative problem solving (symbolic, non-symbolic, or hybrid)? What is a good problem formulation for a given creative problem solving task (planning or learning)? etc. However, these questions are not directly answered within the scope of our work.

8 Ethical Considerations

The authors do not have specific ethical considerations to be highlighted with respect to this work.

References

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. 2022. Flamingo: a visual language

model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736.

Jeffrey Dastin Anna Tong and Krystal Hu. 2023. Openai researchers warned board of ai breakthrough ahead of ceo ouster, sources say. https://www.reuters.com/technology/sam-altmans-ouster-openai-was-precipitated-by-letter-board-about-ai-breakthrough-2023-11-22/. [Online; accessed 19-Jan-2024].

Christopher G Atkeson, PW Babu Benzun, Nandan Banerjee, Dmitry Berenson, Christoper P Bove, Xiongyi Cui, Mathew DeDonato, Ruixiang Du, Siyuan Feng, Perry Franklin, et al. 2018. What happened at the darpa robotics challenge finals. *The DARPA robotics challenge finals: Humanoid robots to the rescue*, pages 667–684.

Rachit Bansal, Bidisha Samanta, Siddharth Dalmia, Nitish Gupta, Shikhar Vashishth, Sriram Ganapathy, Abhishek Bapna, Prateek Jain, and Partha Talukdar. 2024. Llm augmented llms: Expanding capabilities through composition. *arXiv preprint arXiv:2401.02412*.

BBC. 2012. Us navy funds 'macgyver' robot that can create tools. https://www.bbc.com/news/technology-19902954. [Online; accessed 9-April-2024].

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Margaret A. Boden. 1998. Creativity and Artificial Intelligence. *Artificial Intelligence*, 1-2:347–356.

Margaret A. Boden. 2005. What is creativity? *Creativity in human evolution and prehistory*.

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

Stephen Cass. 2005. Apollo 13, we have a solution. *IEEE Spectrum On-line, 04*, 1.

Tuhin Chakrabarty, Vishakh Padmakumar, He He, and Nanyun Peng. 2023. Creative natural language generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts*, pages 34–40.

Lili Chen, Shikhar Bahl, and Deepak Pathak. 2023. Playfusion: Skill acquisition via diffusion from language-annotated play. In *Conference on Robot Learning*, pages 2012–2029. PMLR.

Liuqing Chen, Pan Wang, Feng Shi, Ji Han, Peter Childs, et al. 2018. A computational approach for combinational creativity in design. In *DS 92: Proceedings of the DESIGN 2018 15th International Design Conference*, pages 1815–1824.

Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez. 2023. Simple and controllable music generation. *arXiv preprint arXiv:2306.05284*.

Marci S DeCaro, Robin D Thomas, Neil B Albert, and Sian L Beilock. 2011. Choking under pressure: multiple routes to skill failure. *Journal of experimental psychology: general*, 140(3):390.

Yuqing Du, Olivia Watkins, Zihan Wang, Cédric Colas, Trevor Darrell, Pieter Abbeel, Abhishek Gupta, and Jacob Andreas. 2023. Guiding pretraining in reinforcement learning with large language models. *arXiv preprint arXiv:2302.06692*.

Zackory Erickson, Eliot Xing, Bharat Srirangam, Sonia Chernova, and Charles C Kemp. 2020. Multimodal material classification for robots using spectroscopy and high resolution texture imaging. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10452–10459. IEEE.

Gilles Fauconnier and Mark Turner. 2003. Conceptual blending, form and meaning. *Recherches en communication*, 19:57–86.

Nanyi Fei, Zhiwu Lu, Yizhao Gao, Guoxing Yang, Yuqi Huo, Jingyuan Wen, Haoyu Lu, Ruihua Song, Xin Gao, Tao Xiang, et al. 2022. Towards artificial general intelligence via a multimodal foundation model. *Nature Communications*, 13(1):3094.

Xidong Feng, Ziyu Wan, Muning Wen, Ying Wen, Weinan Zhang, and Jun Wang. 2023. Alphazero-like tree-search can guide large language model decoding and training. *arXiv preprint arXiv:2309.17179*.

Giorgio Franceschelli and Mirco Musolesi. 2023. On the creativity of large language models. *arXiv preprint arXiv:2304.00008*.

Kenneth J Gilhooly. 2016. Incubation and intuition in creative problem solving. *Frontiers in psychology*, 7:1076.

Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. 2023. Imagebind: One embedding space to bind them all.

Evana Gizzi, Lakshmi Nair, Sonia Chernova, and Jivko Sinapov. 2022. Creative problem solving in artificially intelligent agents: A survey and framework. *Journal of Artificial Intelligence Research*, 75:857–911.

Ben Goertzel. 2014. Artificial general intelligence: concept, state of the art, and future prospects. *Journal of Artificial General Intelligence*, 5(1):1.

O. Golovneva, S. O'Brien, R. Pasunuru, T. Wang, L. Zettlemoyer, M. Fazel-Zarandi, and A. Celikyilmaz. 2023. Pathfinder: Guided search over multi-step reasoning paths. *arXiv preprint arXiv:2312.05180*.

Jonathan Grudin and Richard Jacques. 2019. Chatbots, humbots, and the quest for artificial general intelligence. In *Proceedings of the 2019 CHI conference on human factors in computing systems*, pages 1–11.

Jindong Gu, Zhen Han, Shuo Chen, Ahmad Beirami, Bailan He, Gengyuan Zhang, Ruotong Liao, Yao Qin, Volker Tresp, and Philip Torr. 2023. A systematic survey of prompt engineering on vision-language foundation models. *arXiv preprint arXiv:2307.12980*.

Lin Guan, Karthik Valmeekam, Sarath Sreedharan, and Subbarao Kambhampati. 2023. Leveraging pretrained large language models to construct and utilize world models for model-based task planning. *arXiv preprint arXiv:2305.14909*.

Joy P Guilford. 1967. Creativity: Yesterday, today and tomorrow. *The Journal of Creative Behavior*, 1(1):3–14.

Adam Hammond, Julian Brooke, and Graeme Hirst. 2013. A tale of two cultures: Bringing literary analysis and computational linguistics together. In *Proceedings of the Workshop on Computational Linguistics for Literature*, pages 1–8.

Jiaming Han, Kaixiong Gong, Yiyuan Zhang, Jiaqi Wang, Kaipeng Zhang, Dahua Lin, Yu Qiao, Peng Gao, and Xiangyu Yue. 2023. Onellm: One framework to align all modalities with language.

Sebastien Hélie and Ron Sun. 2010. Incubation, insight, and creative problem solving: a unified theory and a connectionist model. *Psychological review*, 117(3):994.

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

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2022. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*.

Ehud Karpas, Omri Abend, Yonatan Belinkov, Barak Lenz, Opher Lieber, Nir Ratner, Yoav Shoham, Hofit Bata, Yoav Levine, Kevin Leyton-Brown, Dor Muhlgay, Noam Rozen, Erez Schwartz, Gal Shachaf, Shai Shalev-Shwartz, Amnon Shashua, and Moshe Tenenholtz. 2022. Mrkl systems: A modular, neuro-symbolic architecture that combines large language models, external knowledge sources and discrete reasoning.

Wonjae Kim, Bokyung Son, and Ildoo Kim. 2021. Vilt: Vision-and-language transformer without convolution or region supervision. In *International Conference on Machine Learning*, pages 5583–5594. PMLR.

Joonas Lahikainen, Nadia M Ady, and Christian Guckelsberger. 2024. Creativity and markov decision processes. *arXiv preprint arXiv:2405.14966*.

Boyi Li, Philipp Wu, Pieter Abbeel, and Jitendra Malik. 2023. Interactive task planning with language models. *arXiv preprint arXiv:2310.10645*.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. 2023. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500. IEEE.

Zhan Ling, Yunhao Fang, Xuanlin Li, Tongzhou Mu, Mingu Lee, Reza Pourreza, Roland Memisevic, and Hao Su. 2023. Unleashing the creative mind: Language model as hierarchical policy for improved exploration on challenging problem solving.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2024. Visual instruction tuning. *Advances in neural information processing systems*, 36.

Yuxi Ma, Chi Zhang, and Song-Chun Zhu. 2023. Brain in a vat: On missing pieces towards artificial general intelligence in large language models. *arXiv preprint arXiv:2307.03762*.

Michael S Matena and Colin A Raffel. 2022. Merging models with fisher-weighted averaging. *Advances in Neural Information Processing Systems*, 35:17703–17716.

Michael Moor, Oishi Banerjee, Zahra Shakeri Hossein Abad, Harlan M Krumholz, Jure Leskovec, Eric J Topol, and Pranav Rajpurkar. 2023. Foundation models for generalist medical artificial intelligence. *Nature*, 616(7956):259–265.

Caterina Moruzzi. 2020. Artificial creativity and general intelligence. *Journal of Science and Technology of the Arts*.

Saeid Naeini, Raeid Saqur, Mozhgan Saeidi, John Giorgi, and Babak Taati. 2023. Large language models are fixated by red herrings: Exploring creative problem solving and einstellung effect using the only connect wall dataset. *arXiv preprint arXiv:2306.11167*.

NeurIPS. 2022. Workshop on machine learning for creativity and design. https://nips.cc/virtual/2022/workshop/49965. [Online; accessed 19-Jan-2024].

Ana-Maria Olteteanu. 2014. Two general classes in creative problem-solving? an account based on the cognitive processess involved in the problem structure-representation structure relationship. *Publications of the Institute of Cognitive Science*.

Ana-Maria Olteţeanu and Zoe Falomir. 2016. Object replacement and object composition in a creative cognitive system. towards a computational solver of the alternative uses test. *Cognitive Systems Research*, 39:15–32.

Dwarkesh Patel. 2023. Llms need search for problem solving - shane legg (deepmind founder). https://www.youtube.com/watch?v=qulfo6-54k0. [Online; accessed 19-Jan-2024].

Cassio Pennachin and Ben Goertzel. 2007. Contemporary approaches to artificial general intelligence. In *Artificial general intelligence*, pages 1–30. Springer.

Archiki Prasad, Alexander Koller, Mareike Hartmann, Peter Clark, Ashish Sabharwal, Mohit Bansal, and Tushar Khot. 2023. Adapt: As-needed decomposition and planning with language models. *arXiv preprint arXiv:2311.05772*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2021. High-resolution image synthesis with latent diffusion models. 2022 ieee. In *CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10674–10685.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2021. Learning to retrieve prompts for in-context learning. *arXiv preprint arXiv:2112.08633*.

Mark A Runco and Garrett J Jaeger. 2012. The standard definition of creativity. *Creativity research journal*, 24(1):92–96.

Vasanth Sarathy and Matthias Scheutz. 2018. Macgyver problems: Ai challenges for testing resourcefulness and creativity. *Advances in Cognitive Systems*, 6:31–44.

Henry Shevlin, Karina Vold, Matthew Crosby, and Marta Halina. 2019. The limits of machine intelligence: Despite progress in machine intelligence, artificial general intelligence is still a major challenge. *EMBO reports*, 20(10):e49177.

Yash Shukla, Wenchang Gao, Vasanth Sarathy, Alvaro Velasquez, Robert Wright, and Jivko Sinapov. 2023. Lgts: Dynamic task sampling using llm-generated sub-goals for reinforcement learning agents. *arXiv preprint arXiv:2310.09454*.

Tom Silver, Soham Dan, Kavitha Srinivas, Joshua B Tenenbaum, Leslie Pack Kaelbling, and Michael Katz. 2023. Generalized planning in pddl domains with pretrained large language models. *arXiv preprint arXiv:2305.11014*.

Marta Skreta, Zihan Zhou, Jia Lin Yuan, Kourosh Darvish, Alán Aspuru-Guzik, and Animesh Garg. 2024. Replan: Robotic replanning with perception and language models. *arXiv preprint arXiv:2401.04157*.

Alon Talmor, Ori Yoran, Ronan Le Bras, Chandra Bhagavatula, Yoav Goldberg, Yejin Choi, and Jonathan Berant. 2022. Commonsenseqa 2.0: Exposing the limits of ai through gamification. *arXiv preprint arXiv:2201.05320*.

Guy Tennenholtz, Yinlam Chow, Chih-Wei Hsu, Jihwan Jeong, Lior Shani, Azamat Tulepbergenov, Deepak Ramachandran, Martin Mladenov, and Craig Boutilier. 2023. Demystifying embedding spaces using large language models. *arXiv preprint arXiv:2310.04475*.

Yufei Tian, Abhilasha Ravichander, Lianhui Qin, Ronan Le Bras, Raja Marjieh, Nanyun Peng, Yejin Choi, Thomas L Griffiths, and Faeze Brahman. 2023. Macgyver: Are large language models creative problem solvers? *arXiv preprint arXiv:2311.09682*.

MC Turner, LV Duggan, BA Glezerson, and SD Marshall. 2020. Thinking outside the (acrylic) box: a framework for the local use of custom-made medical devices. *Anaesthesia*.

Imke Van Heerden and Anil Bas. 2021. Ai as author–bridging the gap between machine learning and literary theory. *Journal of Artificial Intelligence Research*, 71:175–189.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Dan Ventura. 2014. Can a computer be lucky? and other ridiculous questions posed by computational creativity. In *Artificial General Intelligence: 7th International Conference, AGI 2014, Quebec City, QC, Canada, August 1-4, 2014. Proceedings 7*, pages 208–217. Springer.

Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. 2023. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pages 35151–35174. PMLR.

Graham Wallas. 1926. *The art of thought*. 24. Harcourt, Brace.

Brian Wang. 2023. Openai q* could be based upon a* search without expansions. https://www.nextbigfuture.com/2023/11/openai-q-could-be-based-upon-a-search-without-expansions.html. [Online; accessed 19-Jan-2024].

Geraint A Wiggins. 2006. A preliminary framework for description, analysis and comparison of creative systems. *Knowledge-based systems*, 19(7):449–458.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. 2023. The rise and potential of large language model based agents: A survey. *arXiv preprint arXiv:2309.07864*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*.

Danyang Zhang, Lu Chen, Situo Zhang, Hongshen Xu, Zihan Zhao, and Kai Yu. 2023a. Large language model is semi-parametric reinforcement learning agent. *arXiv preprint arXiv:2306.07929*.

Shun Zhang, Zhenfang Chen, Yikang Shen, Mingyu Ding, Joshua B Tenenbaum, and Chuang Gan. 2023b. Planning with large language models for code generation. *arXiv preprint arXiv:2303.05510*.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. 2023. Language agent tree search unifies reasoning acting and planning in language models. *arXiv preprint arXiv:2310.04406*.

A Alternate Definitions of Creative Problem Solving

Prior work by Olteţeanu (Olteteanu, 2014) defines CPS from an object affordance perspective, where affordances broadly refer to action possibilities for objects, e.g., cups are pour-able and doors are open-able. The authors in Olteteanu (2014) define creative problems as nominal problem solving tasks that have a poor representational structure, and as "*the ability of a cognitive, natural, or artificial system to use new objects to solve a problem, other than the ones that have been stored in its memory as tools for that specific purpose (if any), or to create those objects by putting together objects or parts of objects the system has access to. Depending on the problem, objects can be either physical or abstract/informational (concepts, problem templates, heuristics or other forms of representations)*". However, this definition is primarily object-creativity centered, and does not cover a wider range of creative problems.

Follow-up work by Sarathy and Scheutz (Sarathy and Scheutz, 2018), define "*Macgyver-esque*" creativity as a planning task that involves "*generating, executing, and learning strategies for identifying and solving seemingly unsolvable real-world problems*". They introduce the "*MacGyver Problem*" (MGP) as a planning problem with an unreachable goal state. Through the modification of the agent's domain knowledge (through *domain expansion* and *domain contraction*), the agent must discover new information and incorporate it into its existing domain knowledge, allowing the agent to accomplish the task. The domain expansion and contraction processes align with the divergent-convergent model of creative problem solving (Guilford, 1967). The definition of an MGP aligns well with the formulation of planning problems in ML, but less with learning or hybrid planning-learning approaches.

B Alternate theories on creative problem solving and their applications to ML

While there is exhaustive literature regarding theories on general creativity, we focus specifically on creative problem solving, with three well received works: **Divergent-Convergent Thinking** (Guilford, 1967), **Explicit-Implicit Interaction Theory** (Hélie and Sun, 2010), and the **Creative Systems Framework** (Wiggins, 2006). We discuss their applicability to ML in addition to the literature discussed in the main body of this paper to further

emphasize the different pathways for connecting CC to LLVMs for creative problem solving.

B.0.1 Divergent-Convergent Thinking

In Guilford (1967), the authors discuss the notion of "divergent-convergent" thinking. Divergent thinking or "divergent-production" (DP) abilities involve a more open-ended generation of a variety of ideas, whereas convergent thinking focuses on applying specific ideas to solve the problem.

Applicability to CPS in LLVMs: Prior work by Tian et al. (2023) have demonstrated the applicability of "divergent-convergent" thinking towards solving *Macgyver* problems. Similar in spirit to our experiments with VLMs in Section 4.2.1, the authors prompt LLMs with descriptions of objects to enable the LLMs to reason about solving the task. Their work is, to the best of our knowledge, the only direct example demonstrating the value of CC literature in enabling CPS in LLMs.

B.0.2 Explicit-Implicit Interaction Theory

In Hélie and Sun (2010), the authors introduce the *Explicit-Implicit Interaction* (EII) theory, building upon the seminal work in Wallas (1926), that describes four stages of creativity: Preparation, incubation, illumination (i.e., insight), and verification. Preparation refers to the initial stage of searching in many different directions, which may fail to find a solution (i.e., impasse) in case of ill-defined problems (as is the case with CPS). Following an impasse, the incubation phase begins, where attention is not devoted to solving the problem. Over a period of time, illumination is the manifestation of the solution to the problem within the *conscious* thought (i.e., *"Aha"* moment). Finally, verification involves using deliberative thinking to assess if the solution indeed solves the problem.

Applicability to CPS in LLVMs: The authors in (Hélie and Sun, 2010) incorporate the four stages via a concrete computational method into the CLARION cognitive architecture. Prior work has also introduced a CPS framework for ML approaches inspired by the four stages (Gizzi et al., 2022). In their work, "preparation" aligns with problem formulation, either task learning or planning. Incubation and illumination aligns with knowledge representation (symbolic, non-symbolic, or hybrid), and knowledge manipulation (functions that manipulate the conceptual space). Lastly, verification aligns with evaluation (via simulation, real-world platforms, or benchmarks). Although these works do not explicitly cover LLVMs and related algorithms, they demonstrate the value of integrating CC literature in ML, and can serve as useful starting points for ML approaches towards creative problem solving in LLVMs.

B.0.3 Creative Systems Framework

In Wiggins (2006), the author expands on Boden's levels further in the context of a framework that formalizes creative systems. The paper defines: a) creative system, b) creative behavior, c) novelty, and d) value. The paper also discusses formalized notion of a *universe of possibilities*, and *conceptual spaces*. Crucially, the work describes the characteristics of a creative agent, that can help distinguish modes of *failures* within a creative system, namely: a) *hopeless uninspiration* – where there are no valued concepts within the universe; b) *conceptual uninspiration* – where there are no valued concepts within the conceptual space of the agent; and c) *generative uninspiration* – where an agent is unable to find a valued concept owing to the specific method (e.g., search) employed.

Applicability to CPS in LLVMs: While the discussion of novelty, value and conceptual spaces in Wiggins (2006) aligns with our descriptions in Section 4, the different modes of uninspiration offers potential ways to assess failure modes in LLVMs. This allows agents to distinguish between systems where creative problem solving is not possible (hopeless uninspiration), as compared to systems where the conceptual space or the methodology for searching the conceptual space, may be at fault (conceptual or generative uninspiration). Although this approach has not been expanded in existing literature, it presents a promising direction for an evaluation framework that can distinguish CPS from non-CPS problems.

B.1 A potential link between creative problem solving and general intelligence

Existing literature hints at a potential link between creative problem solving and Artificial General Intelligence (AGI) - systems that are broadly capable of solving almost all tasks that humans can (Shevlin et al., 2019). For instance, in Moruzzi (2020), p.85., the author argues that there exists a strong correlation between creativity and AGI: "*... features that systems need to develop in order to achieve general intelligence are aspects that they need to possess also to earn the attribute creative*". In (Goertzel, 2014), the author compiles a list of *competencies*

deemed essential for achieving AGI, including creative capacities like "*conceptual invention*" and "*creative constructive play with objects*". The processes of "insight" or "incubation" often associated with creative problem solving (Hélie and Sun, 2010; Gilhooly, 2016) is also considered important for AGI (Ventura, 2014). Taken together, it is likely that any promising vision of AGI would be incomplete without creative problem solving.

Alongside the heavy ongoing discussion of AGI surrounding LLVMs (Bubeck et al., 2023; Fei et al., 2022; Ma et al., 2023; Xi et al., 2023; Moor et al., 2023; Grudin and Jacques, 2019), there is often little to no discussion of creative problem solving or Computational Creativity within mainstream ML. As described in Moruzzi (2020), p.96, "*The investigation on the nature of creativity and on how it manifests itself not only in human but also in animal and artificial systems should, thus, not be intended as a niche discussion but, rather, as a fundamental research which can lay the foundations for further studies in artificial intelligence and its relation to humans*". We hope that this work will encourage discussions of creative problem solving and Computational Creativity alongside discussions on AGI.

C Experiment Settings

C.1 Data: Test images

Figure 3 shows the test set of 16 RGB images of objects used for the object substitution task. From the shown image dataset, we create test sets with 4 objects each, chosen from the set of 16 object images. We create 10 such test sets per core object (total 50 samples per model). Each test set only includes one ground truth object, along with three other random objects that will not suit as an appropriate replacement. In the nominal group, the ground truth is the actual object itself. In the object replacement group, the replacements are chosen based on self-assessment of the authors as (core object → replacement): "Scoop" → "Bowl"; "Hammer" → "Saucepan"; "Spatula" → "Knife"; "Toothpick" → "Safety pin"; "Pliers" → "Scissors".

C.2 Model: Checkpoints

For all the models, we use pre-trained HuggingFace checkpoints, with no additional training or fine-tuning. The models are of different architecture sizes and patch sizes: "CLIP-B-32" uses the *"openai/clip-vit-base-patch32"* which is a base model with a patch size of 32. "CLIP-B-16" uses *"openai/clip-vit-base-patch16"* – a base model with patch size of 16. "CLIP-L-14" uses *"openai/clip-vit-large-patch14"* – a large model with patch size of 14. "CLIP-H-14" uses *"laion/CLIP-ViT-H-14-laion2B-s32B-b79K"* which is a "huge" model, with a patch size of 14. This model is trained with the 2 billion sample English subset of LAION-5B. For LLaVA, we use the *"llava-hf/llava-1.5-7b-hf"* with 7B parameters, version 1.5. Lastly, "VILT-B-32" uses *"dandelin/vilt-b32-finetuned-vqa"* trained for visual question answering. However, there is limited data available on HuggingFace regarding the model.

C.3 Prompts used in testing

In this section, we discuss the prompts used in the different testing conditions (see Table 2). We explore four classes of prompts for the creative object substitution task: "Regular", "Affordance", "Task" and "Task and affordance". Regular prompts involve a direct prompt as to whether a given object will suffice as a substitute for the missing object. Affordance prompts, adds information about the desired affordances that are essential for replacing the missing object. Task prompts adds additional information on the task to be performed as context for whether a given object can be used as replacement for the missing object. Lastly, task and affordance prompts combine the task and object affordance information within the prompt.

C.4 Testing Procedure

For each test case, we pass the images in the test set along with a prompt belonging to one of the four classes described in Table 2. We record whether the ground truth object image was chosen by the model for the prompt (i.e., assigned highest output probability). CLIP generates probabilities that given images correspond to a text. ViLT responds with a text, and we evaluate if the model responded "yes" with a high probability for the ground truth.

C.5 Testing Infrastructure

We used NVIDIA-A100 GPUs to run the evaluation. However, the models are not too large and we have tested and confirmed that the code can be executed on CPU only as well.

D Continued Experiment Results

In this section, we show the class-wise breakdown of the different models for the different prompting

Prompt type	Prompt
Regular	“can this object be used as a scoop?” “can this object be used as a hammer?” “can this object be used as a spatula?” “can this object be used as a toothpick?” “can this object be used as pliers?”
Affordance	“scoops must be concave and hollow. can this object be used as a scoop?” “hammers must be heavy and have a handle attached to a cylinder at the end. can this object be used as a hammer?” “spatulas must have a handle attached to a flat surface at the end. can this object be used as a spatula?” “toothpicks must have a pointed tip. can this object be used as a toothpick?” “pliers must have two-prongs. can this object be used as pliers?”
Task	“scoops can transfer beans from one jar to another jar. can this object be used as a scoop?” “hammers can hit a nail into the wall. can this object be used as a hammer?” “spatulas can spread butter onto a pan. can this object be used as a spatula?” “toothpicks can pick food caught between the teeth. can this object be used as a toothpick?” “pliers can grab a coin. can this object be used as pliers?”
Task and affordance	“scoops can transfer beans from one jar to another jar. scoops are concave and hollow. can this object be used as a scoop?” “hammers can hit a nail into the wall. hammers have a handle attached to a cylinder at the end. can this object be used as a hammer?” “spatulas can spread butter onto a pan. spatulas have a handle attached to a flat surface at the end. can this object be used as a spatula?” “toothpicks can pick food caught between the teeth. toothpicks have a pointed tip. can this object be used as a toothpick?” “pliers can grab a coin. pliers have two-prongs. can this object be used as pliers?”

Table 2: Prompts (across 4 groups) used in the experiment

Figure 3: Complete test set of objects used in the experiments.

strategies (Figures 4 - 7). We note that "hammers" present a particularly challenging case for all the models, perhaps due to the fact that correlating affordance of a hammer to a saucepan textually is difficult. In contrast, all models with the augmented prompts typically perform well in the case of creatively replacing "toothpick" with "safety pin" – presumably indicating that specifying the relevant affordance textually in this case provides sufficient information. We repeated each experiment across multiple random seeds and found similar performances, showing that our general findings hold across different random cases. Generally, specifying object affordance information in the prompts leads to improved model performance.

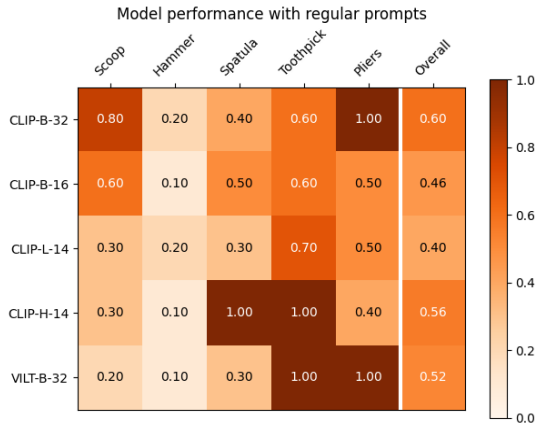

Figure 4: Object replacement test: Using the same prompts as for the nominal group. Random selection of a replacement object achieves ≈30% overall accuracy.

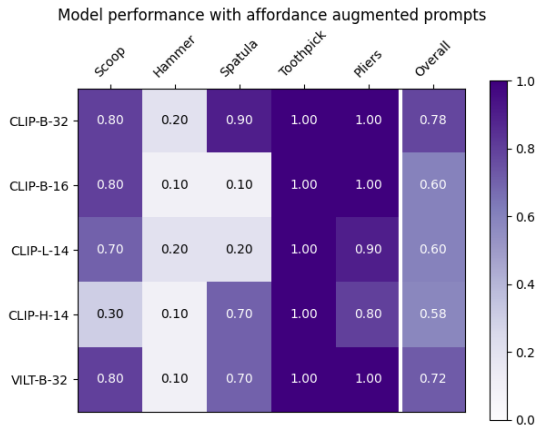

Figure 5: Object replacement test: Accuracies when the prompts are augmented with object affordance information.

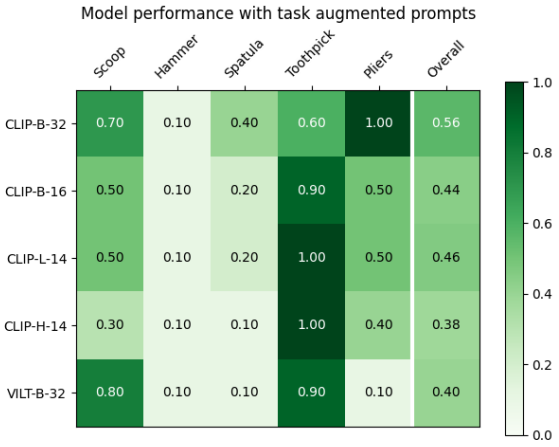

Figure 6: Object replacement test: Accuracies when the prompts are augmented with task information.

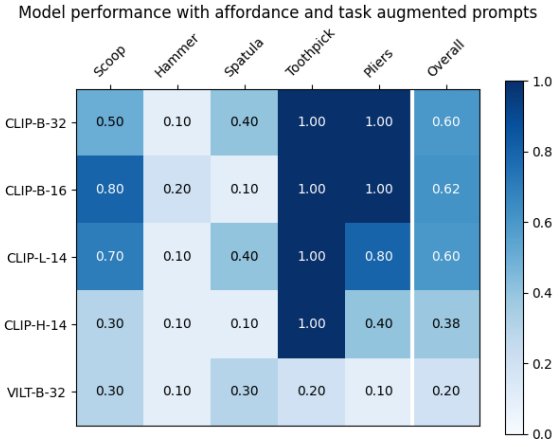

Figure 7: Object replacement test: Accuracies when the prompts are augmented with task and object affordance.