



EMAIL

**Eduardo Pignatelli¹, Johan Ferret², Tim Rocktäschel^{1,2}, Edward Grefenstette^{1,2},
Davide Paglieri¹, Samuel Coward³, Laura Toni¹**

¹ University College London
e.pignatelli@ucl.ac.uk

² Google DeepMind

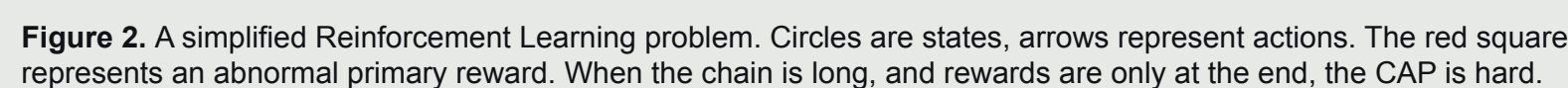
³ University of Oxford

UCL

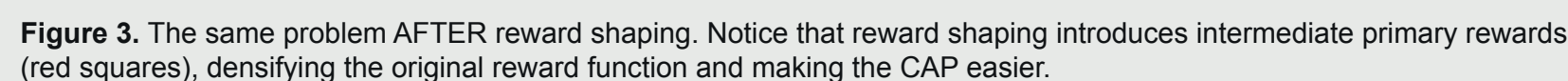
1. Context

- The **TEMPORAL CREDIT ASSIGNMENT PROBLEM** is central in RL:
- To attribute the appropriate **INFLUENCE** to actions in a trajectory, for their ability to achieve a goal.

- Canonical methods, such as **Reward Shaping** and **Options DO NOT SCALE**. Why?
- Because they require **human intervention**, which becomes a bottleneck



- How is this usually solved?
 - Reward shaping works by **DENSIFYING** the **reward function**
 - by using **artificial, intermediate primary rewards**
 - a good rule of thumb is to give rewards for **intermediate subgoals**



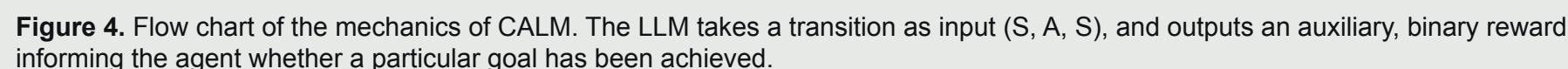
$$\begin{aligned} \tilde{r}_{t+1} &= \tilde{R}(s_t, a_t, s_{t+1}) \longleftarrow \text{Shaping function} \\ r_{t+1} &= R(s_t, a_t, s_{t+1}) + \tilde{R}(s_t, a_t, s_{t+1}) \longleftarrow \text{Augmented primary reward} \end{aligned}$$

Let's **SCALE** reward shaping, **RL** would be **SO SIMPLE!**

- Reward shaping is **EXPENSIVE**:
 - It requires **specialistic domain knowledge**
 - **Hand crafted** for each task
 - Acquiring **labels from human DOES NOT SCALE!** (for a model learning from scratch)

How can we **SCALE REWARD SHAPING**?

- We propose to use LLMs:
 - **CALM: Credit Assignment with Language Models**



$$LLM : desc(\mathcal{M}) \times desc(\mathcal{S} \times \mathcal{A} \times \mathcal{S}) \rightarrow \mathbb{B}.$$

- The LLM takes:
 - **A description of the task**
 - **A description of the transition**
- And returns:
 - **A set of useful options (subgoals)**
 - A Boolean for each option, telling whether a subgoal has been achieved or not

SUMMARY:

- We explored whether LLMs can be a useful means to transfer human knowledge into the value function of RL agents
- Human data is expensive
- We mitigate the costs by replacing humans with LLMs

LIMITATIONS:

- Results are limited to OFFLINE settings and to a single environment
- The environment must provide observations in a textual form (not necessarily natural language)
- Open question: does CALM scale with environment complexity?

- **QUESTION:** Can the **prior knowledge** encoded in **LLMs** serve as a useful prior for CA in RL?"
- **AIM:** To understand if **CREDIT ASSIGNMENT** is in the spectrum of capabilities of open-weight LLMs

- LLMs can **understand goal specifications** and **verify option termination**
- LLMs can also **suggest effective options** to use, and **verify their own goal specifications**

- 1. AIM:** to understand if credit assignment is in the **SPECTRUM OF CAPABILITIES** of current open-weights LLMs
- 2. ENVIRONMENT:** We focus on the **KeyRoom** environment from **MiniHack**
- 3. DATASET:** We collect **256 one-step transitions**.
- 4. LABELS:** Given a set of subgoals, we **classify each transition** as either achieving a subgoal or not
 - This produces labels $C = \{c_i : 0 \leq i \leq |\mathcal{G}| + 1\}$, **one for each subgoal** and an additional one for no-subgoal achieved
- 5. MODELS:** we **test pretrained, open-weights LLM** that can fit a 80Gb A100 Nvidia GPU (4 or 16 bits)

THIS TRANSLATES INTO A CLASSIFICATION PROBLEM

		HUMAN
	Goal achieved	Goal NOT achieved
LLM	Goal achieved	True Positive
	Goal NOT achieved	MISS
		HALLUCINATION
		True Negative

Example prompt

The environment is Minimalist.

I will present you with an short extract of a gameplay. At each timestep, symbols represent the following items:

- "x" represents a floor tile.
- "+" represents either a wall, a vertical wall, an open door.
- "<+" represent either the bottom left corner (of a room), bottom right corner (of a room), wall, horizontal wall, wall, top left corner (of a room), or right corner (of a room).
- "+" represents a closed door. Doors can be locked, and require a key to open.
- "(+)" represents a useful item (pick-axe, key, lamp...)
- "<+" represents a ladder or staircase up.
- ">+" represents a ladder or staircase down.

The task of the agent is to win the game.

First, based on your knowledge of NetHack, break down the task of the agent into subgoals.

Then, consider the following game transition, which might or might not contain these subgoals.

Determine if any of the subgoals is achieved at Time: 1 or not.

Report your response in a dictionary containing the name of the subgoals as keys and booleans as value. For example:

```
{
  "Name of goal": {bool},
}
```

Observation Sequence:

```
<gameplay>
Time: 0
Current message:
+-----+
| . . . |
| . . . |
+-----+
+-----+ <
| . . . |
| . . . |
+-----+
```

Time: 1

Current message:

```
+-----+
| . . . |
| . . . |
+-----+ <
| . . . |
| . . . |
+-----+
```

</gameplay>

I will not consider anything that is not in the dictionary.

You have only one shot at this, and you cannot ask for clarifications.

Mistral-7B-Instruct-v0.2

Based on my understanding of the given environment and the task, the subgoals for the agent are as follows:

1. Exploration: Reach all the unvisited parts of the map.
2. Collection: Gather all the useful items (pick-axe, key, lamp, etc.)
3. Door Opening: Open all the doors (locked or not) to progress through the map.

Now, let's analyze the given gameplay sequence and determine if any of the subgoals have been achieved at Time: 1.

```
{
  "Exploration": False,
  "Collection": False,
  "Door Opening": False
}
```

At Time: 1, the agent has moved to a new location and has not interacted with any items or doors. Therefore, none of the subgoals have been achieved at this point.

gemma-1.1-7b-it

```
## Subgoals:

--Find a way out:## Locate and reach the exit of the map.
--Gather useful items## Collect any available items for future use.
--Explore the environment## Discover the layout of the map and potential hazards.

## Subgoal Achieved at Time: 1

{
  "Find a way out": False,
  "Gather useful items": True,
  "Explore the environment": True,
}
```

Llama-3-70B-Instruct

Based on my knowledge of NetHack, I break down the task of the agent into the following subgoals:

- Reach the staircase/ladder: The agent needs to find a way to move up or down the levels to progress in the game.
- Collect useful items: The agent needs to collect items such as pick-axe, keys, and lamps to overcome obstacles and access new areas.
- Unlock doors: The agent needs to find keys to unlock closed doors and access new areas.
- Avoid obstacles: The agent needs to avoid walls, locked doors, and other obstacles that can hinder its progress.
- Reach the goal: The agent needs to reach the final goal, which is not specified in this extract but is typically a treasure or the exit.

Now, considering the given game transition, I determine if any of the subgoals have been achieved at Time: 1 or not:

```
{
  'Reach the staircase/ladder': False,
  'Collect useful items': False,
  'Unlock doors': False,
  'Avoid obstacles': True,
  'Reach the goal': False,
}
```

The agent has avoided obstacles by moving north and not hitting the wall, but none of the other subgoals have been achieved at Time: 1.

2. Can LLMs detect option termination?

- We provide the LLM with a set of useful subgoals \mathcal{G} , and a transition, (s_t, a_t, s_{t+1})
- We prompt it to ask if the action in the transition has made progress to achieve the goal

Annotator	F1 \uparrow	Accuracy \uparrow	Precision \uparrow	Recall \uparrow
Human	1.00	1.00	1.00	1.00
Mixtral-8x7B-Instruct-v0.1*	0.74	0.67	0.77	0.73
gemma-1.1-7b-it	0.73	0.70	0.91	0.61
Meta-Llama-3-70B-Instruct	0.66	0.65	0.97	0.50
Meta-Llama-3-8B-Instruct	0.64	0.64	0.95	0.49
c4ai-command-r-v0.1	0.60	0.57	0.80	0.49
Mistral-7B-Instruct-v0.2	0.48	0.54	0.96	0.32
gemma-1.1-2b-it	0.00	0.33	0.00	0.00
Random	0.33	0.33	0.33	0.33

Table 1. Performance with **game screen** observations and with subgoals **provided** in the prompt.

Annotator	F1 \uparrow	Accuracy \uparrow	Precision \uparrow	Recall \uparrow
Human	1.00	1.00	1.00	1.00
Mixtral-8x7B-Instruct-v0.1*	0.78	0.70	0.78	0.77
gemma-1.1-7b-it	0.76	0.69	0.79	0.73
gemma-1.1-2b-it	0.76	0.68	0.76	0.77
c4ai-command-r-v01*	0.75	0.69	0.81	0.70
Meta-Llama-3-70B-Instruct	0.63	0.58	0.76	0.54
Meta-Llama-3-8B-Instruct	0.61	0.61	0.92	0.46
Mistral-7B-Instruct-v0.2	0.61	0.62	0.96	0.45
Random	0.33	0.33	0.33	0.33

Table 2. Performance with **cropped** observations and with subgoals **provided** in the prompt.

FINDINGS:

1. LLMs are generally effective in **recognising when an instruction has been successfully completed**.
2. **Restricting the field of view** of the agent improves performance

Annotator	F1 ↑	Accuracy ↑	Precision ↑	Recall ↑
Human	1.00	1.00	1.00	1.00
Meta-Llama-3-70B-Instruct	0.82	0.72	0.71	0.96
Meta-Llama-3-8B-Instruct	0.80	0.70	0.72	0.89
gemma-1.1-7b-it	0.77	0.66	0.71	0.85
Mixtral-8x7B-Instruct-v0.1*	0.74	0.64	0.71	0.76
Mistral-7B-Instruct-v0.2	0.57	0.48	0.63	0.53
c4ai-command-r-v01*	0.56	0.52	0.71	0.47
gemma-1.1-2b-it	0.00	0.33	0.00	0.00
Random	0.33	0.33	0.33	0.33

Table 3. Performance with **game screen** observations and with **autonomously discovered** subgoals.

Annotator	F1 ↑	Accuracy ↑	Precision ↑	Recall ↑
Human	1.00	1.00	1.00	1.00
Meta-Llama-3-70B-Instruct	0.83	0.75	0.75	0.93
gemma-1.1-7b-it	0.81	0.70	0.71	0.95
Mixtral-8x7B-Instruct-v0.1*	0.72	0.62	0.71	0.74
Mistral-7B-Instruct-v0.2	0.65	0.54	0.66	0.64
c4ai-command-r-v01*	0.60	0.52	0.68	0.54
gemma-1.1-2b-it	0.47	0.52	0.89	0.32
Meta-Llama-3-8B-Instruct	0.45	0.39	0.57	0.37
Random	0.33	0.33	0.33	0.33

Table 4. Performance with **cropped** observations and with **autonomously discovered** subgoals.

FINDINGS:

1. The **prior knowledge** of these LLMs is **effective** to break down the task into **subgoals** that are both meaningful and generic enough to be useful in many occasions
2. LLMs can verify subgoals from their own suggestions