Learning to Follow Instructions in Text-Based Games

Mathieu Tuli[1,2] Andrew C. Li[1,2] Pashootan Vaezipoor[1,2] Toryn Q. Klassen[1,2,3] Scott Sanner[1,2] Sheila A. McIlraith[1,2,3]

[1]Department of Computer Science, University of Toronto [2]Vector Institute [3]Schwartz Reisman Institute

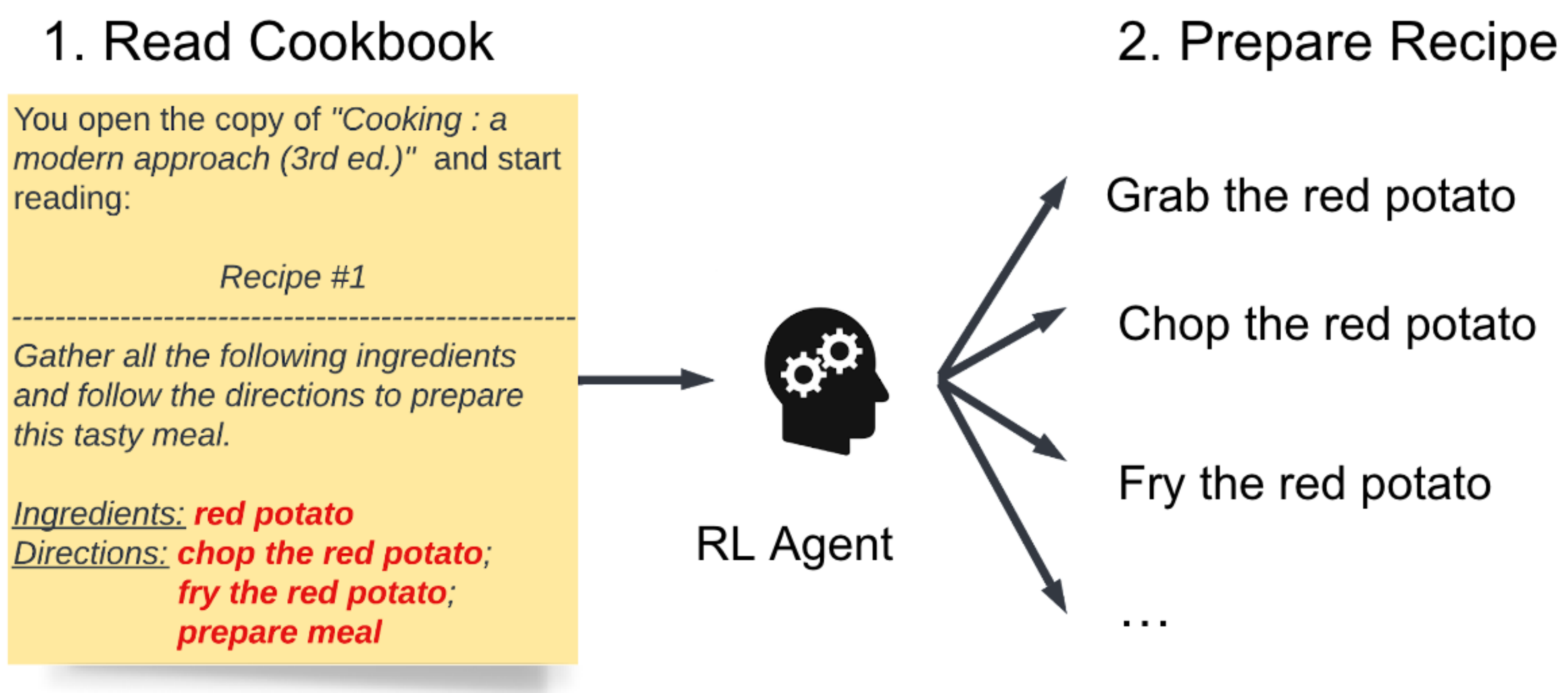
{mathieutuli, andrewli, pashootan, toryn, sheila}@cs.toronto.edu, ssaner@mie.utoronto.ca

Takeaways

TL;DR we study RL agents' ability to follow instructions in text-based games (TextWorld) and outperform the SoTA by leveraging formal language.

- ⚠ State-of-the-art Reinforcement Learning (RL) agents for text-based games are impervious to instructions.
- ⚙ We equip RL agents with a structured representation of instructions using the formal language, **linear temporal logic (LTL)**.
- ✓ LTL expresses complex instructions compactly, offers compositional syntax and semantics, and supports progress monitoring towards instruction completion.
- ✓ We achieve superior performance on 500+ TextWorld games.

TextWorld



You open the copy of *"Cooking : a modern approach (3rd ed.)"* and start reading:

Recipe #1

Gather all the following ingredients and follow the directions to prepare this tasty meal.

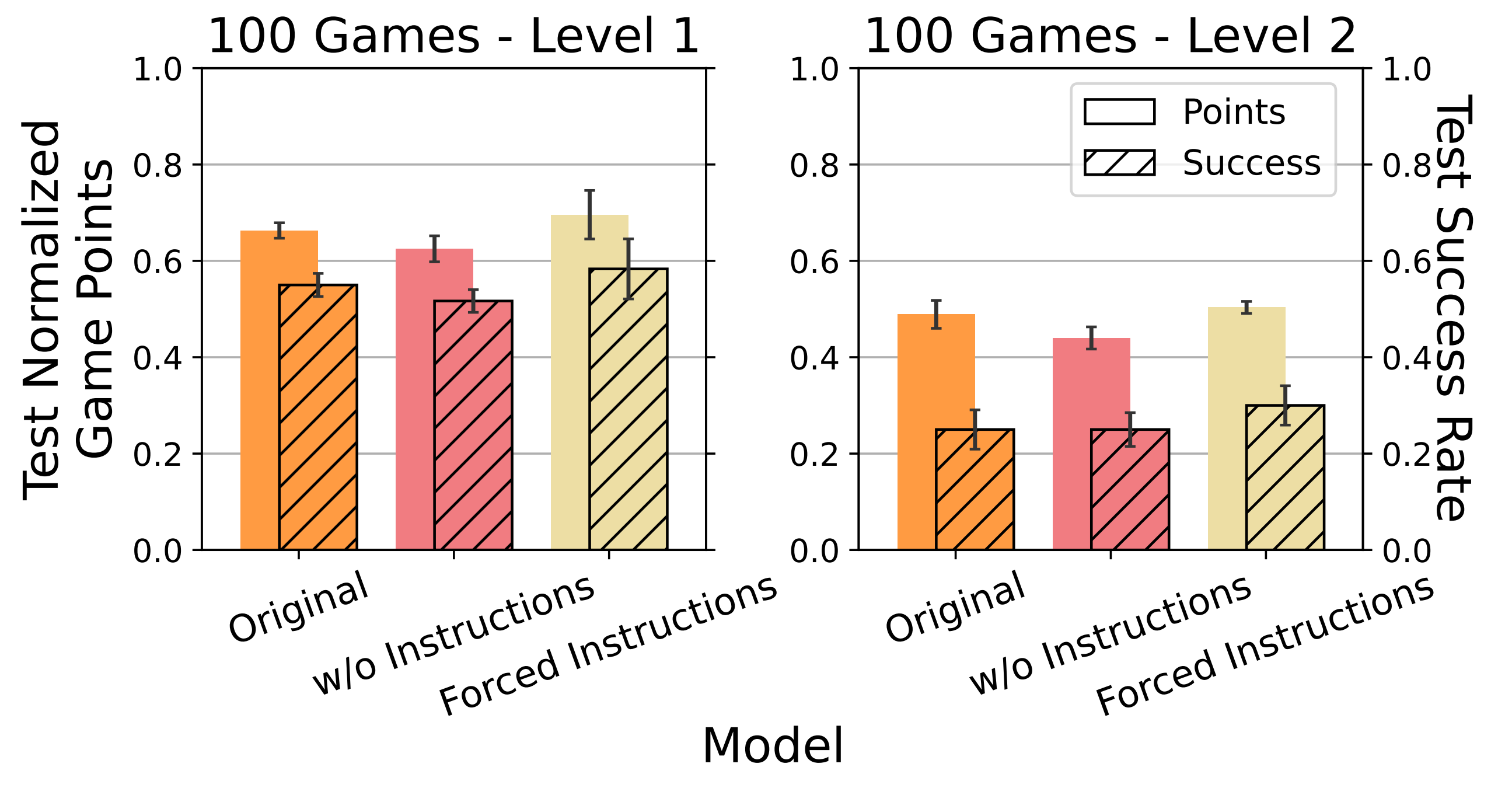
Ingredients: ***red potato***
Directions: ***chop the red potato****;*
fry the red potato*;*
prepare meal

Observations and actions are in natural language. Challenges include **partial observability**, **long-term memory**, and **language understanding**.

Can SoTA agents follow instructions?

GATA (Adhikari et al., 2020) augments transformer-based agents with dynamic long-term memory.

- ⚠ **Largely ignores instructions critical to success.** Performance does not change when instructions (e.g. the cookbook recipe) are removed from observations, or forcibly given to the agent.



Linear Temporal Logic (LTL)

LTL is a temporal logic classically used for verification, program synthesis, and recently, for non-Markovian reward specification in RL.

- ✓ **Temporal patterns** are defined via (nested) modalities such as `EVENTUALLY`, `UNTIL`, `ALWAYS` applied to propositions p, composed together using logical connectives.
- ✓ **Unambiguous semantics** allow us to automatically monitor progress towards instruction completion, unlike natural language.

Next Eventually

$$\varphi ::= p \mid \neg\varphi \mid \varphi\wedge\psi \mid \bigcirc\varphi \mid \varphi \,\mathsf{U}\, \psi \mid \Diamond\varphi \mid \Box\varphi$$

Until Always

Instructions Steps	LTL	Natural Language
Single	◊ `red-potato-in-player`	"Get the red potato"
Ordered	◊ (`red-potato-in-player` ∧ ◊ `red-potato-is-chopped`)	"Get the red potato then chop the red potato"
Unordered	◊ `red-potato-in-player` ∧ ◊ `carrot-in-player`	"Get red potato and carrot in any order"
Disjunctive	◊ `red-potato-is-fried` ∨ ◊ `red-potato-is-baked`	"Fry or bake the red potato"
Safety	◊ `red-potato-in-player` ∧ □ ¬ `knife-in-player`	"Get the red potato while not holding the knife"

More complex instructions are also supported.

LTL-GATA

1. Translate natural language observations to LTL

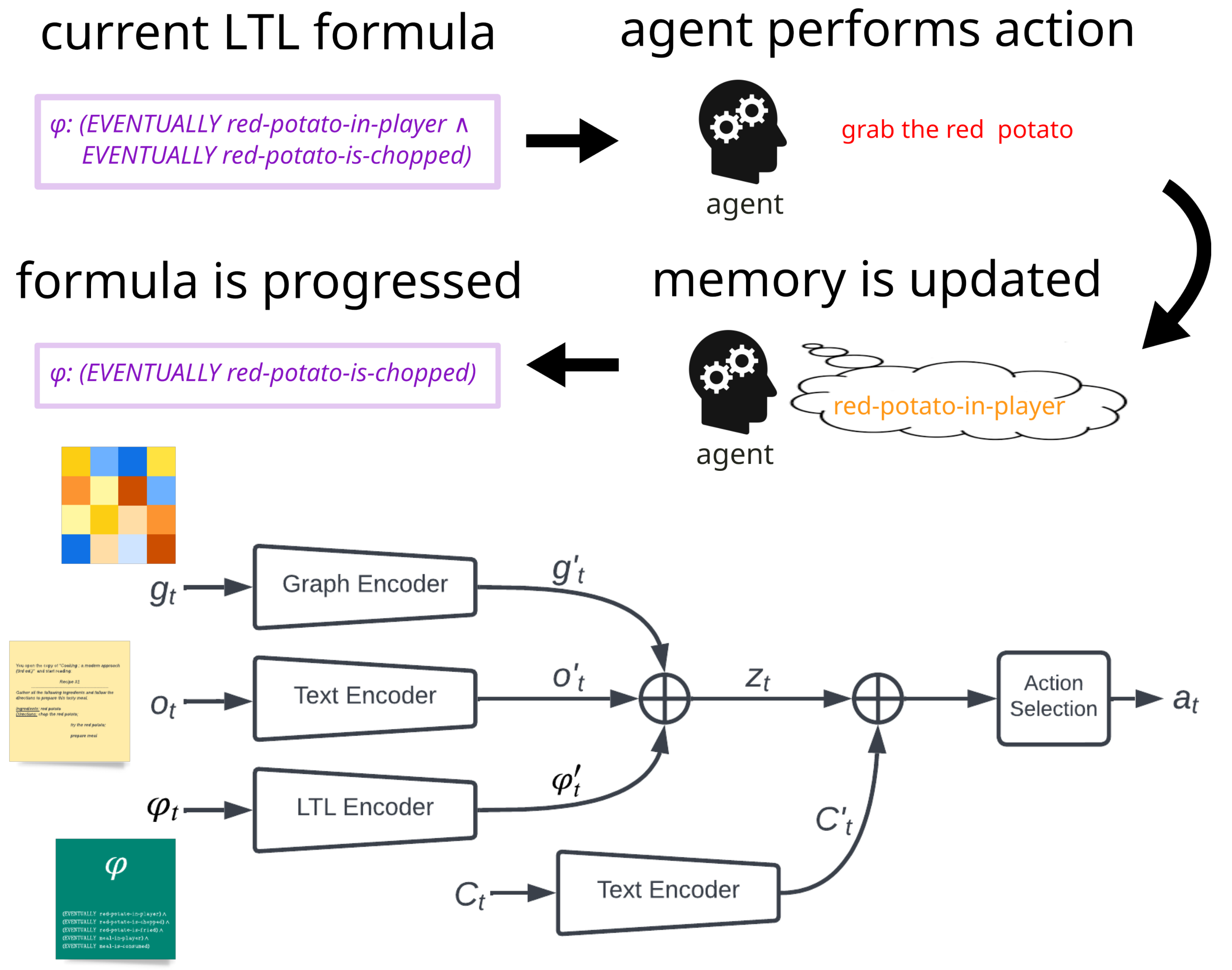
- ⚙ We build a natural-language-to-LTL translator that extracts instruction info.
- ✓ We show that GPT-3 can automatically perform this translation using as few as six examples.

2. Track satisfaction of instruction steps with LTL progression

- 💡 **LTL Progression** (Bacchus and Kabanza, 2000) is a formally defined, semantics-preserving rewriting operation that simplifies instructions over time as parts of the task are solved.
- ⚙ We evaluate the truth/falsity of propositions using GATA's learned belief graph, in support of LTL progression.
- ⚙ We reward or penalize the agent for satisfying or violating instructions (resp.).

3. Condition policy on Transformer-encoded LTL

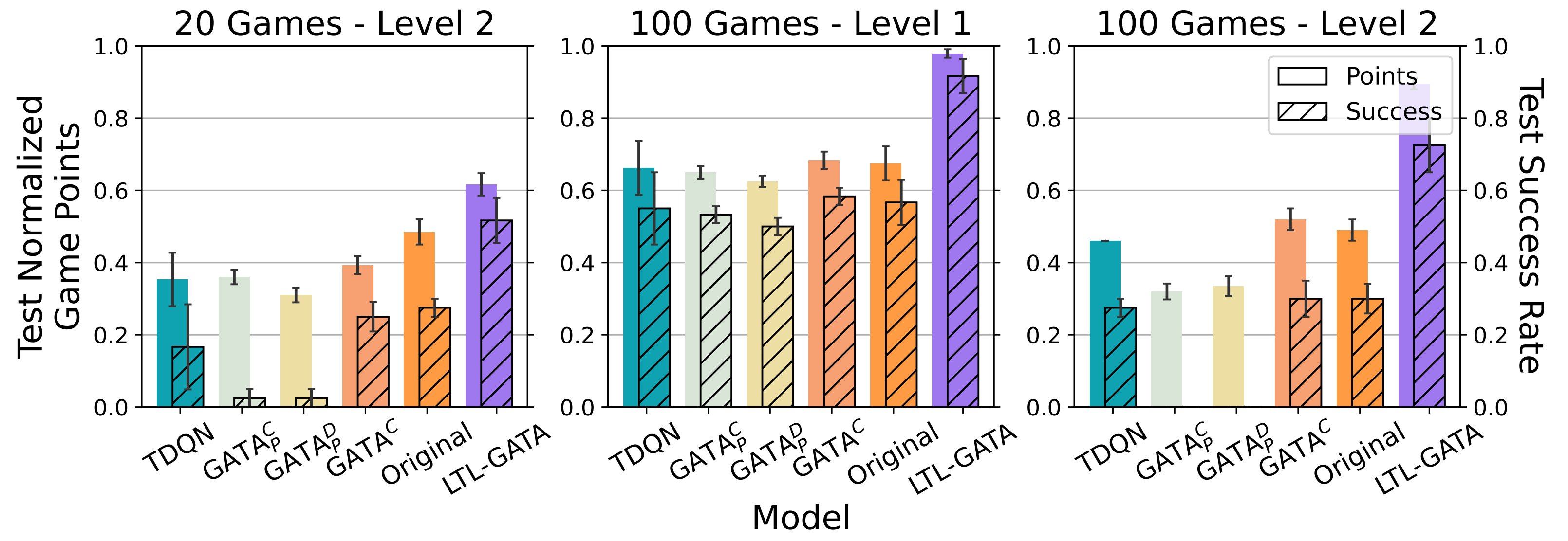
- ⚙ LTL-GATA selects actions $a_t \in C_t$ conditioned on observations o_t, belief graph (memory) g_t, and the generated LTL instructions φ_t.
- ⚙ Belief graph is encoded using graph convolutional neural networks, while text observations, actions and LTL instructions are encoded using Transformers.



Experiments

Key Results

- ✓ **Superior performance** over previous SOTA.
- ✓ **Progression matters.** Ablations show that the use of LTL and its progression operator is a critical mechanism for success.
- ✓ **Strong generalization performance** by LTL-GATA when given sufficient data.
- Code at `https://github.com/MathieuTuli/LTL-GATA`.


20 Games - Level 2
100 Games - Level 1
100 Games - Level 2
Test Normalized Game Points
Test Success Rate
Points
Success
TDQN
$GATA^C_P$
$GATA^D_P$
$GATA^C$
Original
LTL-GATA
Model