Full Application Template — AISI Challenge Fund (Stage 2)

Section 1: Project Overview

Research Title	Verifiably-Safe Reinforcement Learning via (Soft) Linear Temporal Logic
Lead Applicant Name	Samuel Gélineau
Affiliation / Organisation	Glasgow Lab for AI Verification Ltd, a.k.a. Glaive Research
Country Location	UK
Assigned AISI Research Sponsor	David Africa
Priority Research Area	Alignment
Project Duration (in months)	6 months
Total Funding Requested (£)	104 140£

Section 2: Research Objectives & Summary

2.1 Research Statement

Our prior work at https://gelisam.com/range-analysis has shown that if we convert weights to code, we can use static analysis to verify that a classification model has robustly learned its safety property and will satisfy it on *all* inputs. Does this result generalize:

1. to AI agents?
2. to user-defined safety properties?

Guaranteeing a safe output for every input is promising for high-stakes settings where, to quote AISI, even a single unsafe output could cause safety and security risks. In this new context, the agent's input is the sequence of environment states which the agent observes as a result of its actions, so the definition of "all inputs", or "all circumstances", depends on our choice of world model, which we assume to be known. See the example in the appendix.

This project is a step towards scaling our approach to frontier models. Adding support for AI agents is important because frontier models are agentic, and because the most dangerous scenario is the one in which a misaligned agent is actively performing actions in the world in response to human actions, that is, the AI is fighting back.

User-defined safety properties are also important, as researchers are still working on outer alignment and specification gaming. Leaving the safety property open-ended makes it possible for other researchers to fill it in once a robust formulation has been found.

2.2 Key Research Objectives

1. Train an agent on an objective defined by an arbitrary predicate expressed in Linear Temporal Logic.
2. Use code generation and static analysis to conservatively verify that the agent will satisfy this property under all circumstances.
3. Train an agent on the objective of being accepted by the verifier. Note that this does not follow from (1) and (2), because our prior work found that backprop tends to find the solutions which satisfy the property in practice but fail verification.

2.3 Research Contribution

This project contributes a technique for verifying whether an agent has robustly learned its safety property or still needs more training. Depending on how it is used, this approach can address either of these aspects of AISI's alignment plan:

1. showing that training has converged, so we can rely on asymptotic guarantees, or
2. proving a worst-case guarantee, in case the training has not converged.

The distinction between those two cases depends on whether the conservative verification is applied to the agent's entire objective or only to a component of that objective. For example, it is possible to express "connect all the stars but do not get any blocks stuck" (Gridworld's stand-in for an irreversible action) as a predicate in Linear Temporal Logic (LTL). The conservative verification can thus tell us whether the agent will accomplish this entire objective in all circumstances and thus has converged, or if we need to train it more. Opting for a soft, differentiable version of LTL instead of the sharp, boolean-valued default enables penalizing the agent for failing to satisfy user-specified properties, explicitly pushing it towards formal well-behavedness throughout its training.

By contrast, "collect all the stars in as few moves as possible" cannot be expressed, so we will use regular RL for this part of the objective, and verification will prove the worst-case guarantee that even if the agent's solution is not the shortest, at least we know that it does irreversibly push boxes onto walls.

Section 3: Proposed Methodology

3.1 Research Approach

We will implement the algorithm, tweaking metaparameters and simplifying the problem if needed, then add complications and polish until we run out of time. This allows us to adapt to unexpected difficulties while still demonstrating the promise of this line of research.

3.2 Data Sources and Collection

All the code will be publicly-available on github. Weights are expected to be ephemeral, but can be made publicly-available if re-training from scratch proves time-consuming.

3.3 Analysis Methods

1. Visualize the environment, paths, and directed graphs (see appendix).
2. Measure whether training an agent to pass verification in a variety of environments increases its chances of success on unseen (and thus unverified) environments, compared to directly training on the safety property.

3.4 Technical Resources Required

We plan to use small neural networks and thus expect the training cost to be negligible.

Section 4: Deliverables and Outputs

Software development is often delayed, so we plan to use the last month as a buffer period. Delivery dates are thus a month later than the plan below. We will deliver earlier if we can.

Deliverable	Description	Format	Target Completion Date
D1: Scaffolding	A visual, Gridworlds-style RL framework	Initial github repo	End of month 2
D2: User-defined goals	Let users specify the objective	Code update	End of month 3
D3: Static analysis	Safe-in-{practice,theory} checks	Code update	End of month 4
D4: Verifiably-safe RL	Verifier accepts some agents	Code update	End of month 5
D5: Validation	Measure safety performance on unseen environments	Docs update	End of month 6

Section 5: Work Plan and Timeline

Month/ Week	Key Activities	Milestones	Performance Metrics / Success Criteria
Month 1	Create or adapt a visual Gridworlds-style RL framework	Scaffolding completed	Can train an agent on a hardcoded task and view its path

Month 2	Integrate with Linear Temporal Logic	User-defined goals completed	Can train on a user-defined task
Month 3	Implement code generation and static analysis	Static analysis completed	Can verify a hardcoded policy
Month 4	Implement a soft, differentiable version of the static analysis	Verifiably-safe RL completed	Can train an agent to pass verification
Month 5	Generate random environments, measure performance	Validation completed	Measures obtained and discussed
Month 6	Polish UI & readme	Project completed	Clear & easy to use

Section 6: Key Project Risks and Mitigation Strategies

Risk	Likelihood	Impact	Mitigation Plan
Overfitting objective or safety property	High	Low	Balance the relative importance of those two conflicting goals.
RL stuck in local minima	High	Low	Re-train with a different seed.
RL always gets stuck	Low	High	Simplify the environment even further, give the agent access to more information.
Reward Misspecification	Med	Low	The objective is configurable, tweak it.
Static analysis too slow	Low	Low	Make the state space even smaller.
Unexpected blocker	Low	High	Earlier deliverables still provide value.

Some limitations are known and will need more research to overcome. If we finish all of our deliverables early, those are directions we could explore.

Limitation	Future work
Extra training needed from safe-in-all-cases to verifiably-safe	Conservative static analysis algorithms are fast but reject some correct programs. Near convergence, we reject many programs which are safe in all cases but not yet provably-so. We can stop early by occasionally using an SMT solver (slower, fewer false positives).
Frontier models are big / neurons are too low-level to analyze	A static analysis with good asymptotics scales to large programs, but frontier models are more complex, not just bigger. Proper analysis requires a higher-level representation. We want to fine-tune a coding model to convert weights to higher-level code which is amenable to static analysis even if it remains incomprehensible to humans.
Real environments have too many states	This project reduces the cost from the number of trajectories to the number of states, while our previous work reduces it from the number of input values / states to the number of neurons. Combine both?

The world model must be known	Yoshua Bengio's "Scientist AI" solution is to use a trustworthy AI to infer a world model, then prove safety with respect to it. We agree!

Section 7: Team and Expertise

Name	Designation / Role	Affiliation	Relevant Experience	Key Responsibilities in this Project
Samuel Gélineau	Principal investigator	Glaive Research	Author of the prior work on verifying a classifier model.	Implementation, reporting progress.
Zanzi Mihejevs	Researcher	Glaive Research	Experience with converting weights to code, ARIA Grantee for "True Categorical Programming for Composable Systems Total".	Advising with the category theoretic approach to machine learning, pair programming.
Konstantinos Kogkalidis	Researcher	Independent researcher	Implemented a soft version of Linear Temporal Logic.	Integrating his soft Linear Temporal Logic library into the project.
Vincent Wang-Maścianica	Advisory role	HAILab, University of Oxford	Expertise with category-theoretic and interpretable AI, co-author of "On the Anatomy of Attention".	Aid with design of structured learning objectives.

Section 8: Ethical Considerations

8.1 Ethical Approvals

Not applicable, we are not collecting data from any humans during this project.

8.1 Data Privacy and Security

Not applicable, we are not collecting data from any humans during this project, and there is no data to protect since everything will be made publicly-available.

Appendix: mockup