XGBoost as a Surrogate Model for Testing Deep Reinforcement Learning Agents

1. BACKGROUND

Deep Reinforcement Learning (DRL) has emerged as a powerful paradigm for training agents which are capable of learning complex tasks by trial-and-error interaction with an environment. However, the testing of these DRL agents remains computationally expensive and inefficient.
To address this problem, Biagiola et al. [1] proposed the use of surrogate models to approximate whether the DRL agent will fail in a certain environment.

While their results are promising, their study focuses exclusively on neural networks and performs minimal hyperparameter tuning, leaving open the question whether other ML models may perform better and whether hyperparameter optimisation techniques like grid search could improve the performance of these models

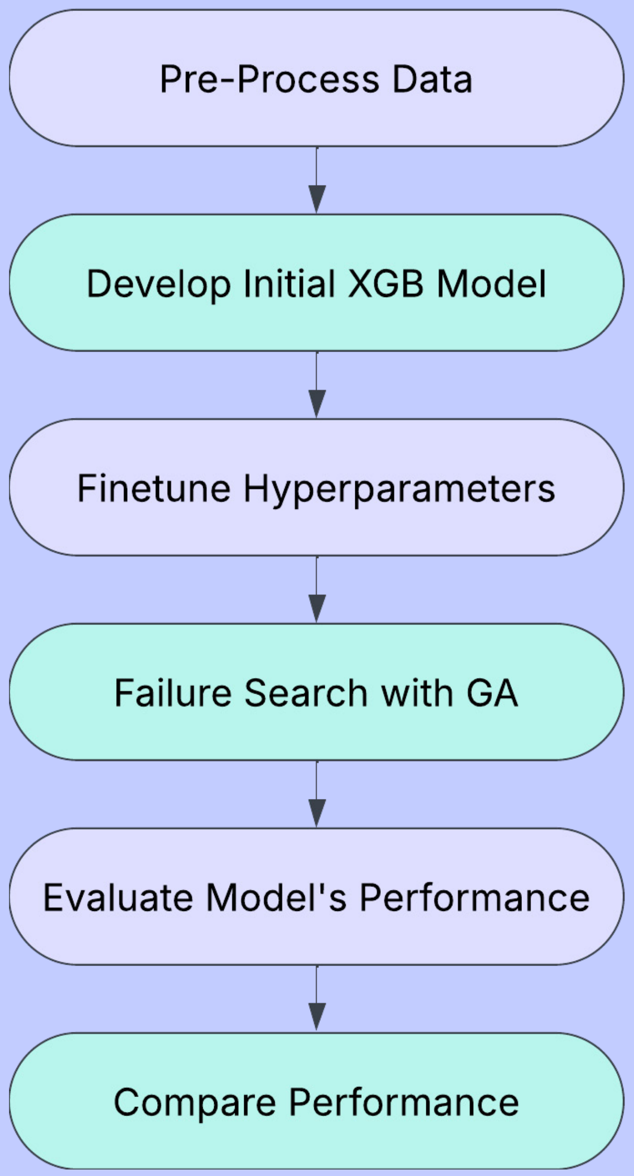
2. RESEARCH QUESTIONS

RQ1:
How effective is XGBoost as a surrogate model to identify failing environments for a DRL agent compared to a baseline Multi-Layer-Perceptron?

- **RQ1.1**: What is the performance of XGBoost compared to the MLP in classifying failing environments?
- **RQ1.2**: How effective is XGBoost compared to the MLP in guiding a Genetic Algorithm to find failing environments?

3. METHODOLOGY

1. **Data Preprocessing**: We omit the first 25% of training data and split it up into a training, validation and test set.
2. **XGBoost Model Development**: We develop an initial model of an XGBoost Classifier by extending the existing Indago framework developed by Biagiola et al. [1].
3. **Hyperparameter Tuning**: We perform Grid Search over several model parameters (e.g. learning rate, regularisation terms, sampling ratio's) and techniques to handle data imbalance (oversampling, class-weights).
4. **Failure Search**: We use the model as a fitness function to guide a Genetic Algorithm to generate failing environments. The DRL agent is then run on these environments.
5. **Evaluation**: We evaluate the model's classification performance (Accuracy, F1-score, AUC-ROC) and the performance in failure search (Amount of failing environments produced, coverage, entropy).
6. **Comparison**: We compare the performance of XGBoost against the pre-trained baseline MLP used in the work by Biagiola et al. [1] using the mentioned evaluation metrics.



4 RESULTS & DISCUSSION

Classification Performance

Metric	XGBoost	MLP
Accuracy	**0.944** ± 0.010	0.788 ± 0.027
Precision	**0.441** ± 0.063	0.099 ± 0.015
Recall	**0.525** ± 0.036	0.424 ± 0.070
F-score	**0.475** ± 0.034	0.161 ± 0.022
AUC-ROC	**0.820** ± 0.021	0.687 ± 0.023

XGBoost consistently outperformed the MLP baseline in predicting failing environments:

- **Achieved a better trade-off** between precision and recall, leading to more reliable failure detection.
- **Demonstrated lower variability** across multiple runs, indicating stable and robust performance.
- **Showed stronger discriminative capability** in separating failure from success cases

Performance in Failure Search

Metric	XGBoost	MLP
Failing environments	**17.66** ± 3.37	14.98 ± 3.24
Coverage	**82.52** ± 16.63	43.49 ± 9.03
Entropy	**67.69** ± 18.59	32.12 ± 26.94

As a fitness function for the Genetic Algorithm, XGBoost improved the quality and diversity of generated test cases:

- **Discovered a higher number of unique failing configurations**, making failure testing more effective.
- **Achieved broader coverage** of the configuration space, exploring more diverse scenarios.
- **Produced more uniformly distributed failures**, highlighting its ability to identify varied edge cases.

4 Conclusion & Future Work

This study shows that XGBoost outperforms MLP in both failure classification and guided failure search for DRL agents:

- More accurate and consistent at predicting failures, making it a stronger surrogate model.
- Guided the Genetic Algorithm to discover more diverse and meaningful failing environments.

Thus, this work demonstrates that XGBoost is a strong candidate for surrogate modelling in the context of DRL testing.

Future Work

- Evaluate other models like LightGBM and CatBoost.
- Tune MLP hyperparameters extensively for fairer comparison.
- Test across different DRL agents and environments to assess generalizability.

REFERENCES

[1] Matteo Biagiola and Paolo Tonella. Testing of deep reinforcement learning agents with surrogate models. ACM Transactions on Software Engineering and Methodology, 33(3):1-33, March 2024.

AUTHOR: Aadesh Ramai (aramai@student.tudelft.nl)

TU Delft

SUPERVISOR & RESPONSIBLE PROFESSOR: Antony Bartlett & Annibale Panichella