Formal Explainability for Artificial Intelligence in Dynamic Environments

Jime Cuartas Granada

Supervisors:
Alexey Ignatiev
Peter J. Stuckey
Julian Gutierrez

A Progress Review report at
Monash University in 2026
School of Information Technology

Abstract

In dynamic environments, the goal of Artificial Intelligence (AI) is to build intelligent agents capable of addressing sequential decision-making settings. Reinforcement Learning (RL) is a branch of Machine Learning that addresses sequential decision-making by agents to perform tasks. In this context, there are two important challenges for humans to understand decisions made by agents: (1) the sequential decisions are connected, and (2) the agents may use opaque black-box models (e.g., neural networks) for each decision.

Despite the success of RL in sequential decision-making, the lack of transparency in understanding their decisions can make the agents hard to validate. To address the need for transparency, there are efforts to develop Explainable Artificial Intelligence (XAI) and its subfield, Explainable Reinforcement Learning. XAI is a set of methods designed to make AI models easier to comprehend. Despite the importance of Explainable Reinforcement Learning in developing trustworthy intelligent agents, there are gaps in current research to make sequential decision-making explainable.

This project proposes to explain sequential decision-making using formal reasoning. To achieve this goal, the proposal focuses on (1) Formal Explainability for Finite Automata, to address sequential actions in deterministic environments, and (2) Formal Explainability for Reinforcement Learning, where the agent's behaviour is non-interpretable.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The deplyoyment of Artificial Intelligence (AI) algorithms has necessitated the need for eXplainability AI (XAI) methods in order to ensure transparency, trust, and accountability. While much of the field has focused on heuristic explanations for opaque models, there is an interest in formal approaches that provide rigorous guarantees about the explanations generated [1, 2].

A fundamental challenge in dynamic environments is explaining sequential decision-making, To address this, we model these processes using Automata, which provide a symbolic and tractable representation of sequential decision functions. This approach allows us to generate formal explanations, why a specific sequence of actions leads to a particular outcome. Automata are widely used in software verification [3], design of communication protocols [4],and sintax parsing in compiler [5]. When a computational model, such as a Finite Automaton (FA) or a Pushdown Automaton (PDA), accepts or rejects an input string, the reasoning behind that decision can be non-trivial. Understanding why a specific input was accepted or rejected is crucial for debugging, and refinement purposes.

This research project investigates the formalization of explanations for sequential decision-making. Having addressed an approach to deliver formal explanations for Finite Automata (FA) in the first stage of this research, and submitting it to a ICALP 2026. We now move to address explanations for Context-Free Languages (CFG) using Pushdown Automata (PDA).

1.1 Refined scope - problem statement

While standard XAI focuses on feature attribution in classifiers, the ”features” in formal languages are sequential and structural. Since the confirmation report, the research scope has been refined to address three primary gaps:

- **Research Problem 1 (Completed): Explaining Finite Automata.** Finite Automata are often assumed to be interpretable. However, large FA are cognitively inaccessible to humans. We have developed a framework to compute formal explanations for the acceptance and rejection of inputs in FA, providing a rigorous foundation for automaton-based explainability.
- **Research Problem 2: Explaining Pushdown Automata (PDA).** Context-free languages, recognized by PDAs, introduce a stack-based memory that allows to represent more complex behaviors. My objective is the generation of Minimal Contrastive Explanations (CXPs) the minimal sets of modifications required to turn a rejected word into an accepted one; and Minimal Abductive Explanations (AXps) the minimal sets of tokens such that the word is going to be rejected.

  Finally, there is a lack of quantitative metrics that assign a “degree of responsibility” to specific indices in a rejected string.
- **Research Problem 3: Explaining Markov Decision Processes.** How can the Feature Attribution Score be extended to explain failure states in Reinforcement Learning policies modeled as MDPs? This problem explores sequential decision-making, where actions influence future states and rewards.

1.2 Contributions to knowledge - achieved and projected

This research provides both theoretical and practical contributions to the field of Computer Science:

Achieved Contributions:

- Development of an theoretical and practical approach to explain Finite Automata decisions.

- A paper submitted to ICALP 2026 tittled "A Formal Framework for the Explanation of Finite Automata Decisions"

Projected Contributions:

- Explaining Pushdown Automata decisions: (In Progress) Extending the formal explanation framework to PDAs, which recognize context-free languages. This involves developing algorithms to identify the minimal contrastive explanations (CXPs) and Abductive Explanations (AXPs), and quantifying the contribution of specific tokens to the decision (acceptance and/or rejection).
- Explainable Reinforcement Learning via MDPs: Extending the formal explanation framework to Markov Decision Processes (MDPs). The goal is to provide verifiable explanations for failure states in Reinforcement Learning policies treating the policy as a stochastic process and identifying the specific environmental factors or decision points that lead to a particular outcomes.

Chapter 2

Maturing Theoretical Constructs and Frameworks

2.1 Model Proposal: Context-Free Grammar (CFG) Explanations

The research has evolved from the study of Finite Automata (FA) to more expressive computational models. While FA provided a baseline for explaining sequential behaviors, they are insufficient for dynamic environments requiring memory or stochastic reasoning.

- **From FA to PDA:** We propose the use of Pushdown Automata (PDAs) to model decision-making processes with memory.

  Unlike FA, the addition of a stack allows the description of more complex languages. challenging explanations to explain recursive behaviors and long-range dependencies in agent traces

Formal Framework for Explanations

We have matured our theoretical framework by defining and identifying two distinct types of formal explanations within the PDA context:

Abductive Explanations: These identify a minimal sufficient set of features (or stack operations) that guarantee the observed outcome (rejection). It answers: "What specific parts of this input were enough to cause the failure?"

Contrastive Explanations: These identify the minimal necessary changes to the input or trace that would result in a different outcome (acceptance). It answers: "What is the smallest change that would have fixed the failure?"

Chapter 3

Progress Since Confirmation

3.1 Preliminaries

Context-Free Grammars and Pushdown Automata

To provide a foundation for the proposed explanation extraction methods, I establish the formal definitions of the underlying structures.

Definition 3.1 (Context-Free Grammar)**.** A Context-Free Grammar (CFG) is defined as a 4-tuple $G = (V, \Sigma, R, S)$, where:

- V (Variables/Non-terminals) is a finite set of variables (non-terminal symbols).
- Σ (Terminals) is a finite set of terminal symbols, disjoint from V.
- R is a finite set of production rules of the form $A \to \alpha$, where $A \in V$ describes a variable and $\alpha \in (V \cup \Sigma)^*$ is a string of variables and terminals.
- $S \in V$ is the start variable.

Definition 3.2 (Pushdown Automaton)**.** A Pushdown Automaton (PDA) extends the capabilities of a Finite Automaton by incorporating an infinite memory stack. A PDA is formally defined as a 7-tuple $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, where:

- Q is a finite set of states.
- Σ is the input alphabet, equivalent to the terminals in a Context-Free Grammar.

- Γ is the stack alphabet. A finite set of symbols that can be pushed onto or popped from the stack.
- $\delta : Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma \to Q \times \Gamma^*$
- is the transition fuction. It dictates how the machine transitions between states and modifies the stack based on the current state, input symbol, and the top symbol of the stack.
- $q_0 \in Q$ is the initial state.
- $z_0 \in \Gamma$ is the initial stack symbol.
- $F \subseteq Q$ is the set of accepting states.

Need citation

It is a well-known result that for every PDA M, there exists an equivalent CFG G that generates the same language $L(M) = L(G)$. This equivalence allows us to utilize grammar-based parsing algorithms, such as CYK, to analyze the behavior of stack-based automata.

$$
\begin{aligned}
\text{Balanced} &\to \text{Open Suffix} \\
\text{Suffix} &\to \text{Balanced Suffix (Nested content)} \\
\text{Suffix} &\to \text{Close Balanced (Close current, start new)} \\
\text{Suffix} &\to \text{')' (End block)} \\
\text{Open} &\to \text{'('} \\
\text{Close} &\to \text{')'}
\end{aligned}
\tag{3.1}
$$

Transition to Explainability for Pushdown Automata decisions Following the confirmation of candidacy, a natural progression from the study of Finite Automata is to consider computational models with greater expressive power. the research scope has expanded to include Pushdown Automata (PDA).

The CYK-Based Explanation Engine

To extract formal explanations from these Context-Free structures, we have developed a novel adaptation of the CYK (Cocke-Younger-Kasami) algorithm.

While standard CYK is primarily used for parsing, we utilize the resulting triangular parsing table as a search space for explanations.

By analyzing partial trees within the CYK table, for a a rejected string w and a PDA $\mathcal{A}$ we can identify:

- *Minimal Abductive Explanations:* minimal subsets of input tokens (leaves in the parse tree) such that is impossible for the start symbol S to be derivable (does not appear in the top cell)
- *Minimal Contrastive Explanations:* minimal set of edits (substitutions) required to "repair" the CYK table such that the start symbol S is derivable (appears in the top cell).

Minimal Contrastive Explanations: In the case of rejection, the Definition environment.

The Grammar $\mathcal{A}$ PCFG is defined as a tuple $G = (V, \Sigma, R, S, P)$, where:

- V is a finite set of non-terminal symbols.
- Σ is a finite set of terminal symbols (the alphabet).
- R is a finite set of production rules.
- $S \in V$ is the start symbol.
- $P : R \to [0, 1]$ is a probability function such that for each $A \in V$, $\sum_{A \to \alpha \in R} P(A \to \alpha) = 1$.

The Rejected WordLet $w = \sigma_1 \sigma_2 ... \sigma_n$ be a string in Σ^*.We say w is rejected if $w \notin L(G)$, where $L(G)$ is the language generated by the grammar.

Definition: Contrastive Set. For a rejected word w of length n, a set of indices $I \subseteq \{1, \ldots, n\}$ is a Contrastive Explanation if:

- Feasibility: There exists a word $w' \in L(G)$ such that w and w' differ only at indices $i \in I$. Formally, $\forall j \notin I, \sigma_j = \sigma'_j$.
- Minimality: No proper subset $I' \subset I$ satisfies the feasibility condition.

Example: For $w = ()))$, the index set $I = \{2\}$ is a contrastive explanation because changing index 2 to) results in $w' = (()) \in L(G)$, and the empty set $\emptyset$ is not feasible.

Introducing Probabilistic Preference. In a PCFG, not all accepted words w' are equal. We can rank explanations by the likelihood of the correction they enable.

The Scoring Function For any word $w' \in L(G)$, the score $P(w')$ is the maximum probability among all possible parse trees T that yield w' (Viterbi Algorithm):

$$P(w') = \max_{T \in \text{Trees}(w')} P(T)$$

Optimal Contrastive Explanation Given a rejected word w, an explanation I_1 is probabilistically superior to I_2 if the best correction enabled by I_1 is more likely than that of I_2. We define the Explanation Weight as:

$$\text{Weight}(I) = \max\{P(w') \mid w' \text{ matches } w \text{ except at indices } I, w' \in L(G)\}$$

The Extended CYK Table

Standard CYK populates a 3D table $T[i, j, A]$, representing the maximum probability that non-terminal A derives the substring from index i to j.

For a word $w = \sigma_1\sigma_2...\sigma_n$ and an index set I:

- **Base Case (Length 1)** For each position $i \in \{1, \ldots, n\}$ and each non-terminal A:
  - If $i \notin I$ (Fixed):
    $$T[i, i, A] = P(A \to \sigma_i)$$
    (If no such rule exists, the probability is 0).
  - If $i \in I$ (in cxp):
    $$T[i, i, A] = \sum_{\sigma \in \Sigma} P(A \to \sigma)$$
    This selects the most likely terminal that A can produce at that "broken" index.

- **Recursive Step (Length $l > 1$)** For each length l from 2 to n, each starting position i from 1 to $n - l + 1$, and each non-terminal A:

$$T[i, l, A] = \max_{A \to BC \in R} \left(\max_{1 \le k < l} \{P(A \to BC) \cdot T[i, k, B] \cdot T[i + k, l - k, C]\} \right)$$

Heatmap for Contrastive Explanations

Theorem: Given a word $w = \sigma_1\sigma_1 \dots \sigma_n$, and grammar G, such that $w \notin L(G)$. Let $\mathcal{E}$ be the set of all contrastive explanations for w. If H is a hitting set of all contrastive explanations $\mathcal{E}$, then no word w' that keeps the indices in H fixed to their original values in w can be accepted by the grammar. Mathematically:

If $\forall i \in H, \sigma'_i = \sigma_i$, then $w' \notin L(G)$.

Proof by Contradiction

Step 1: Assume the negation. Assume there exists a word $w' \in L(G)$ such that w' agrees with the original rejected word w on all indices in the hitting set H.

$$\forall i \in H : \sigma'_i = \sigma_i$$

Step 2: Let J be the set of indices where w' differs from w ($J \cap H = \emptyset$).

$$J = \{j \mid \sigma'_j \neq \sigma_j\}$$

Step 3: Relate to Contrastive Explanations.

Since $w' \in L(G)$ and it was formed by changing indices J in w, then by definition, J is a "feasible" contrastive set. It must either be a minimal contrastive explanation or a superset of one.

$$\exists I \in \mathcal{E} \text{ such that } I \subseteq J$$

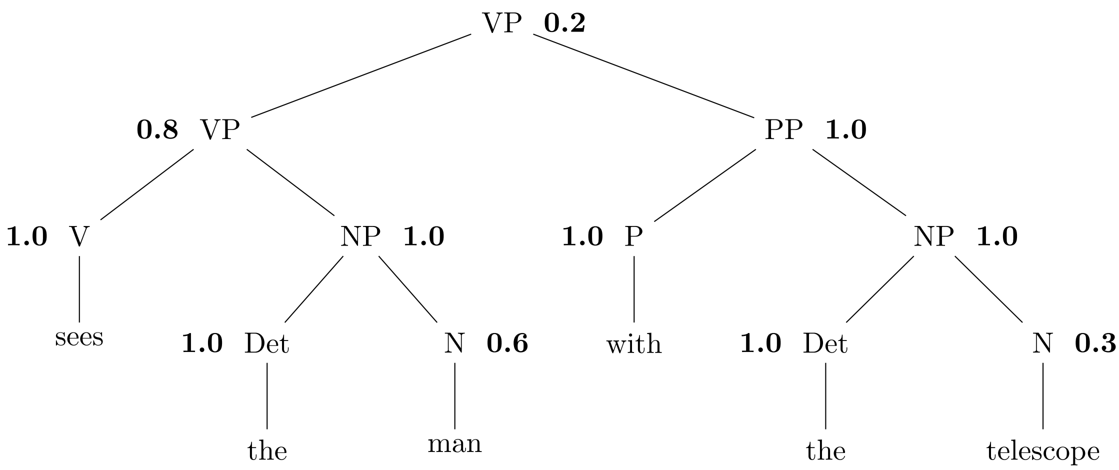
Step 4: The Contradiction.

We know from Step 2 that $J \cap H = \emptyset$. Since $I \subseteq J$, it follows that $I \cap H = \emptyset$. However, by definition, H is a hitting set of $\mathcal{E}$, meaning it must have a non-empty intersection with every $I \in \mathcal{E}$.

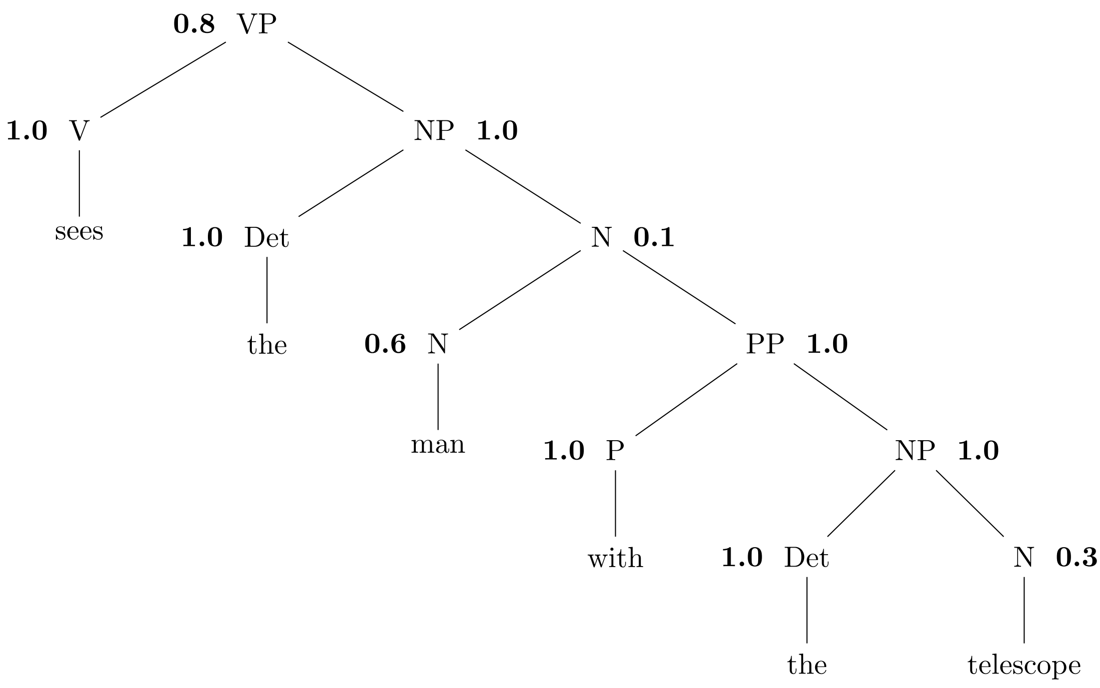
$$H \cap I \neq \emptyset \quad \text{(Contradiction)}$$

Conclusion: Our assumption that an accepted word w' exists is false. Therefore, the indices in H effectively "block" all possible paths to acceptance. They are the necessary components of the error.

0.8	$VP \to V\ NP$	1	$NP \to Det\ N$	0.1	$N \to N\ PP$	1	$Det \to$ the	0.6	$N \to$ man
0.2	$VP \to VP\ PP$	1	$PP \to P\ NP$	1	$V \to$ sees	1	$P \to$ with	0.3	$N \to$ telescope



P(t1) = 0.6 * 0.8 * 0.2 * 0.3 = 0.0288



P(t2) = 0.6 * 0.8 * 0.1 * 0.3 = 0.0144

p(VP,sees the man with the telescope) = 0.0288 + 0.0144 = 0.0432

If “telescope” were “man”

P(t3) = 0.6 * 0.8 * 0.2 * 0.6 = 0.0576

P(t4) = 0.6 * 0.8 * 0.1 * 0.6 = 0.0288

sees the man with the ? = P(t1) + P(t2) + P(t3) + P(t4) = 0.1296

Given the grammar G and the rejected word $w = \sigma_1\sigma_2\dots\sigma_n \notin L(G)$. Let $\mathcal{E} = \{I_1, I_2, \dots, I_k\}$ the collection of all Minimal Contrastive Explanations

Each $I \in \mathcal{E}$: “The error happened exactly in the set of index I”. and $P(I)$ Is the sum up of probabilities of all possible trees that derivates accepted words $w' = \sigma'_1\sigma'_2\dots\sigma'_n$ where $\forall_{i \notin I}\sigma'_i = \sigma_i$

$P(A)$: Probability to get an accepted Tree with a minimal cxp. Every tree is disjoint. $P(A) = P(I_1) + P(I_2) + \cdots + P(I_k)$

$$P(I \mid A) = \frac{P(I)}{\sum_{J \in \mathcal{E}} P(J)}$$

$$P(\text{Error in } i \mid A) = \sum_{I \in \mathcal{E}} P(\text{Error in } i \mid I, A) P(I \mid A)$$

$$P(\text{Error in } i \mid I, A) = \mathbb{F}(i \in I) = \begin{cases} 1 & \text{if } i \in I \\ 0 & \text{if } i \notin I \end{cases}$$

3.2 Future Work and Timeline to Completion

Appendix A

An Appendix

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus at pulvinar nisi. Phasellus hendrerit, diam placerat interdum iaculis, mauris justo cursus risus, in viverra purus eros at ligula. Ut metus justo, consequat a tristique posuere, laoreet nec nibh. Etiam et scelerisque mauris. Phasellus vel massa magna. Ut non neque id tortor pharetra bibendum vitae sit amet nisi. Duis nec quam quam, sed euismod justo. Pellentesque eu tellus vitae ante tempus malesuada. Nunc accumsan, quam in congue consequat, lectus lectus dapibus erat, id aliquet urna neque at massa. Nulla facilisi. Morbi ullamcorper eleifend posuere. Donec libero leo, faucibus nec bibendum at, mattis et urna. Proin consectetur, nunc ut imperdiet lobortis, magna neque tincidunt lectus, id iaculis nisi justo id nibh. Pellentesque vel sem in erat vulputate faucibus molestie ut lorem.

Quisque tristique urna in lorem laoreet at laoreet quam congue. Donec dolor turpis, blandit non imperdiet aliquet, blandit et felis. In lorem nisi, pretium sit amet vestibulum sed, tempus et sem. Proin non ante turpis. Nulla imperdiet fringilla convallis. Vivamus vel bibendum nisl. Pellentesque justo lectus, molestie vel luctus sed, lobortis in libero. Nulla facilisi. Aliquam erat volutpat. Suspendisse vitae nunc nunc. Sed aliquet est suscipit sapien rhoncus non adipiscing nibh consequat. Aliquam metus urna, faucibus eu vulputate non, luctus eu justo.

Donec urna leo, vulputate vitae porta eu, vehicula blandit libero. Phasellus eget massa et leo condimentum mollis. Nullam molestie, justo at pellentesque vulputate, sapien velit ornare diam, nec gravida lacus augue non diam. Integer mattis lacus id libero ultrices sit amet mollis neque molestie. Integer ut leo eget mi volutpat congue. Vivamus sodales, turpis id venenatis placerat, tellus purus adipiscing magna, eu aliquam nibh dolor id nibh. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Sed cursus convallis quam nec vehicula. Sed vulputate neque eget odio fringilla ac sodales urna feugiat.

Phasellus nisi quam, volutpat non ullamcorper eget, congue fringilla leo. Cras et erat et nibh placerat commodo id ornare est. Nulla facilisi. Aenean pulvinar scelerisque eros eget interdum. Nunc pulvinar magna ut felis varius in hendrerit dolor accumsan. Nunc pellentesque magna quis magna bibendum non laoreet erat tincidunt. Nulla facilisi.

Duis eget massa sem, gravida interdum ipsum. Nulla nunc nisl, hendrerit sit amet commodo vel, varius id tellus. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc ac dolor est. Suspendisse ultrices tincidunt metus eget accumsan. Nullam facilisis, justo vitae convallis sollicitudin, eros augue malesuada metus, nec sagittis diam nibh ut sapien. Duis blandit lectus vitae lorem aliquam nec euismod nisi volutpat. Vestibulum ornare dictum tortor, at faucibus justo tempor non. Nulla facilisi. Cras non massa nunc, eget euismod purus. Nunc metus ipsum, euismod a consectetur vel, hendrerit nec nunc.

Bibliography

[1] João Marques-Silva. Logic-based explainability in machine learning. In Leopoldo E. Bertossi and Guohui Xiao, editors, *Reasoning Web. Causality, Explanations and Declarative Knowledge - 18th International Summer School 2022, Berlin, Germany, September 27-30, 2022, Tutorial Lectures*, volume 13759 of *Lecture Notes in Computer Science*, pages 24–104. Springer, 2022. doi: 10.1007/978-3-031-31414-8_2. URL https://doi.org/10.1007/978-3-031-31414-8_2.

[2] Adnan Darwiche. Logic for explainable AI. In *LICS*, pages 1–11. IEEE, 2023.

[3] Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT Press, 2008. ISBN 978-0-262-02649-9.

[4] Gerard J. Holzmann. The model checker SPIN. *IEEE Trans. Software Eng.*, 23 (5):279–295, 1997. doi: 10.1109/32.588521. URL https://doi.org/10.1109/32.588521.

[5] Alfred V. Aho, Ravi Sethi, and Jeffrey D. Ullman. *Compilers: Principles, Techniques, and Tools*. Addison-Wesley series in computer science / World student series edition. Addison-Wesley, 1986. ISBN 0-201-10088-6. URL https://www.worldcat.org/oclc/12285707.