



why not change the world?®

Explainability-Driven Quality Assessment for Rule-Based Systems

Oshani Seneviratne, Brendan Capuzzo, and William Van Woensel

Web of Data Quality Workshop at the Web Conference'25 Sydney, Australia

nsselaer



Why Rule Quality and Explainability matter

- Rules drive reasoning in knowledge-based systems, but errors, gaps, and biases often go unnoticed.
- Traditional validation requires manual checking or labor-intensive dataset labeling.
- Low-quality rules compromise system trustworthiness, fairness, and transparency.
- Explainability offers a human-centered, data-driven path to debug, refine, and improve rules without retraining from scratch.



The Challenge: Ensuring Trustworthy Rule-Based Systems

Knowledge-Based Systems rely on high-quality rules for reliable and trustworthy reasoning.

Key Challenges:

- Labor-intensive **data labeling** requirements
- Incomplete or inaccurate rules affecting outcomes

Our Strategy: Use Explainability to drive quality assessment and refinement.

Types of Explanations to Support Rule Quality:

- **Trace-Based**: Step-by-step rule derivations
- **Contextual**: Immediate rule and fact insights
- **Contrastive**: Comparative reasoning between outcomes
- Counterfactual: "What-if" scenarios to suggest changes



Rensselaer



Explainability-Driven Quality Assessment for Rule-Based Systems



Use Case: Loan Eligibility Reasoning

Scenario: Assess loan eligibility based on:

Credit Score

nsselaer

• Debt-to-Income (DTI) Ratio

Applicants:

- O Alex: Credit Score 680, DTI 0.40 \rightarrow X Not Eligible (High DTI)
- **O D Beth**: Credit Score 605, DTI 0.30 → X Not Eligible (Low Credit Score)
- \bigcirc **Charlie**: Credit Score 700, DTI 0.20 \rightarrow **C** Eligible

Goal:

• Use explanations to understand and debug eligibility decisions.



Four Types of Explanations for Rule Quality Assessment

Rensselaer

Туре	What It Shows	Why It Helps
Q Trace-Based	Full derivation chain	Debug unexpected rule interactions
P Contextual	Immediate rule and facts	Quickly validate local rule behavior
Contrastive	Differences between two cases	Refine thresholds, detect unfairness
Counterfactual	Minimal changes to flip outcome	Suggest actionable improvements



Demo – Scenario Setup

7:27 ₾ ₲ ₫ ☶ • 🕸 ♥∠i 0	8:25 🖱 G 📾 🔺 · 🔌 👻 🖓 🖄	8:28 🖱 G 🔮 🍝 • 🕸 👻 🖗 🗭 ∠/ 🕅	8:29 🖱 G 🖻 🏊 🔸 🔌
Mobile AI Explainability	Mobile AI Explainability	Mobile AI Explainability	Mobile AI Explainability
Model Inspection	Explanations	http://example.com/loanEligibility	http://example.com/loanEligibility
Select Model:	Select Knowledge Model:		Salast Object
transitive 👻	Ioan-eligibility		Select Object.
Select View:	Select Subject:	Not Eligible	Not Eligible
base-model	http://example.com/applicant1	Select Explanation Type:	Select Explanation Type:
buse model		counterfactual -	counterfactual
VIEW MODEL	Select Predicate:	GENERATE EXPLANATION	GENERATE EXPLANATION
	http://example.com/loanEligibility		
Explanations	Select Object:	Output:	Output:
• Select Knowledge Model:	Not Eligible 👻	applicant schema:monthlyDebt ?debt) (?applicant schema:monthlyIncome ?income) quotient(?debt, ?	applicant2 -> Ioanchigonity -> Not Engible applicant2 -> dtiRatio -> XMLSchema#double applicant1 -> IoanEligibility -> Not Eligible
Ioan-eligibility		income, ?dti) -> (?applicant ex:dtiRatio ?dti)] [EligibilityRule: (?applicant rdf:type foaf:Person) (? applicant and itpatic 2dti) (?applicant protectific parts	applicant1 -> dtiRatio -> XMLSchema#double applicant3 -> loanEligibility -> Eligible applicant2 -> dtiBatic -> XMLSchema#double
	trace-based	score, /620 ³⁴ xsd.int) -> (?applicant ex.loanEligibility	applicant2 -> monthlyIncome -> XMLSchema#float applicant2 -> monthlyIncome -> XMLSchema#float
Select Subject:		'Eligible')] [NotEligibleDTIRule: (?applicant rdf.type foaf.Person) (?applicant ex:dtiPatio 2dti) greaterThan(2dti	applicant2 -> creditScore -> XMLSchema#int applicant2 -> name -> Beth annlicant2 -> 22-rdf.svntax-ns#tyne -> Person
http://example.com/applicant1	GENERALE EXPLANATION	(0.349999 ⁴ /vsd:double) -> (?applicant ex:loanEligibility 'Not Eligible')]	applicant1 -> monthlyIncome -> XMLSchema#float applicant1 -> monthlyDebt -> XMLSchema#float
Select Predicate:	Output:	[NotEligibleCreditRule: (?applicant rdf:type foaf/Person) (?applicant ex:creditScore ?score) lessThan(?score,)'621**xsd:int) -> (?applicant ex:loanFilipibility:Not	applicant1 -> creditScore -> XMLSchema#int applicant1 -> name -> Alex applicant1 -> 22-rdf-svntax-ns#type -> Person
http://example.com/loanEligibility		Eligible)]	applicant3 -> monthlyIncome -> XMLSchema#float applicant3 -> monthlyDebt -> XMLSchema#float
Oslast Object			applicant3 -> creditScore -> XMLSchema#int applicant3 -> name -> Charlie applicant3 -> 22-rdf-svntax-ns#type -> Person
Not Eligible			
odel Inspection Dickers	Explanation Input Pickers	Loan Eligibility Model Rules	Inference Mo
Donccoloor			
S REUSSEIAEL	Explainability-Driven Quality Ass	essment for Rule-Based Systems	



Demo – Different Explanations

× ⊽∠i 0

8:27 🖱 G 🖻 🌥 🔸
Mobile AI Explainability
http://example.com/loanEligibility
Select Object:

Not Eligible

Select Explanation Type:

trace-based

GENERATE EXPLANATION

Output:

Conclusion: applicant 1 has Loan Eligibility: Not Eligible used the following matches: Match: applicant1 has Type: Person Conclusion: applicant1 has Type: Person Match: applicant1 has Monthly Debt: 2000 0 And paired them with the following rule: [[DTIRule: (Tapplicant type Person) (Tapplicant monthlyDebt) (Tapplicant durige the monthlyDebt) to reach this conclusion.

And paired them with the following rule: [[NotEligibleDTIRule: (?applicant type Person) (?applic ditRatio ?dt) greaterThan(?dti 10.349999' -> (?applicant ioanEligibility Not Eligible)]]] to reach this conclusion.

☜ ♥∠! 🛛
•
•
*

conclusion: applicant1 has Loan Eligibility: Not Eligible sed the following matches: Match: applicant1 has Type: Person Conclusion: applicant1 has Type: Person Match: applicant1 has Type: Person Match: applicant1 has Type: Person Match: applicant1 has Monthly Debt: 2000.0 Match: applicant1 has Monthly Debt: 2000.0 And paired them with the following rule: [[DTRube: (?applicant1 has Monthly Income: 5000.0 And paired them with the following rule: [[DTRube: (?applicant1 monthly/income?income?income) puotient(?debt (?applicant monthly/income?income) puotient(?debt ?income ?dti) ~ (?applicant dtiRatio ?dti)]

And paired them with the following rule: [] NotEligibleDTRule: (?applicant type Person) (?applicant dtiRatio ?dti) greaterThan(?dti '0.349999' > (?applicant loanEligiblity' NotEligible')]] to reach this conclusion. 8:28
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C

Select Explanation Type:

counterfactual

GENERATE EXPLANATION

Output:

To change the outcome for applicant1 has Loan Eligibility: Not Eligible, you could look at these examples:

applicant3 has Loan Eligibility: Eligible because: Their applicant3 has DT Ratic: 0.2 while yours is applicant1 has DT Ratic: 0.4 Their applicant1 has Monthly Debt: 1000 0 while yours is poplicant1 has Monthly Debt: 2000.0 Their applicant3 has Credit Score: 700 while yours is applicant1 has Credit Score: 680



Trace-Based Explanation

Rensselaer

Contextual Explanation

Contrastive Explanation

Counterfactual Explanation

Explainability-Driven Quality Assessment for Rule-Based Systems



Conclusion

- Rule quality is critical for trustworthy knowledge-based systems.
- Explainability transforms rule validation from manual checking to systematic, human-centered refinement.
- Our framework supports **trace-based**, **contextual**, **contrastive**, and **counterfactual** explanations.
- Integrated into the MIT App Inventor Punya platform for practical, lightweight reasoning.
- Future Directions:
 - Improved **UI/UX** for explanation readability
 - **User studies** to evaluate trust, usability, and impact



More Information

- Paper: <u>https://arxiv.org/abs/2502.01253</u>
- Code: <u>https://github.com/brains-group/androidReasoningPunya</u>
- MIT App Inventor: <u>https://appinventor.mit.edu</u>
- Punya Project: <u>https://punya.mit.edu</u>

Email: <u>senevo@rpi.edu</u> Website: <u>https://oshani.info</u>

Questions?



why not change the world?®