A Hierarchical Multi-Agent LLM Framework for Explainable Financial Risk Assessment and Decision Support

Nimishakavi Vignesh[1], Manasvi Niwalkar[2], Shubhangi Kurkute[3], Sadiya Pansare[4]
[1,2,3,4]*MIT ADT University*

***Abstract*—Personal financial assessment tools usually report a score or a risk label and leave the reader to guess what produced it. This paper presents FinRisk AI, a decision-support framework that keeps the arithmetic of risk assessment apart from its explanation. A Python component validates a user's income, expenses, loan obligations, savings and credit score, and computes four indicators: the debt-to-income ratio, the expense-to-income ratio, savings coverage and remaining monthly income. A rule-based engine turns those numbers into a risk level. Two LLM agents, a Financial Analyst Agent and a Risk Analyst Agent, then read the computed results in sequence, describe the financial pattern and write recommendations that refer back to the figures. A dashboard presents the outcome. Because the language models never compute an indicator or choose a risk level, every number in an explanation can be traced to a deterministic step. We describe the architecture, the formulas, the agent roles and a worked example with a hypothetical profile. We report no experimental results. Instead, the paper proposes an evaluation plan that covers numerical correctness, explanation faithfulness and comparison with simpler baselines.**

***Index Terms*—financial risk assessment, explainable AI, large language models, multi-agent systems, decision support, debt-to-income ratio**

I. INTRODUCTION

Most people make borrowing and saving decisions with a partial view of their own finances. A loan calculator reports an instalment, a credit bureau reports a score and a bank statement reports a balance. Nothing puts these together and says which number is the problem. A person who is told that their credit position is "fair" learns very little about whether to repay a loan, trim spending or build a buffer first.

Machine learning has improved the prediction side of this problem. Lessmann et al. benchmarked 41 classifiers on eight real-world credit scoring data sets [4]. A better predictor does not, on its own, explain a result to the person it affects. Explainable AI (XAI) research covers part of this gap. Adadi and Berrada note that the black-box nature of many AI systems prevents their outputs from being explained directly [1], and methods such as LIME [2] and SHAP [3] attribute a single prediction to individual input features. Hadji Misheva et al. point out that what counts as a good explanation depends on who is reading it [5]. A feature-attribution plot suits a model validator better than a salaried employee deciding whether to take a second loan.

Large language models (LLMs) can write the plain-language explanation that those methods leave out. Finance-specific models such as BloombergGPT [7] and the open-source FinGPT [8] show that language models can be adapted to financial text. The difficulty is reliability. LLMs produce fluent text that is sometimes unfaithful to its input, a problem surveyed by Ji et al. [12]. For that reason, FinRisk AI does not ask a language model to calculate anything.

The framework is built around that decision. Python code validates the user's data and computes the indicators. A rule-based engine assigns the risk level. Two LLM agents sit above this layer and only interpret what it produced. Dividing work among specialised agents is a familiar pattern in LLM research [9], [10], and Xiao et al. have applied it to trading [11]. We apply it to personal financial risk, where the explanation has to be readable and the numbers have to be right.

The paper makes three contributions. It describes a five-component hierarchical architecture that separates calculation and rules from LLM reasoning. It gives the formulas and the scoring rule used by the

deterministic layer. And it proposes an evaluation protocol, because no study has been run yet and no experimental results are claimed.

II. LITERATURE REVIEW

1. Credit and risk modelling:
Lessmann et al. update an earlier comparison of classification algorithms for credit scoring with newer learners, ensemble selection and several accuracy measures, and they also assess the financial impact of different scorecards [4]. The study measures how well scorecards predict and what they are worth. Explaining a score to the applicant lies outside its scope.

2. Explainable AI in finance:
Adadi and Berrada survey XAI and identify missing transparency as a key obstacle to adopting AI systems [1]. LIME explains a prediction by learning an interpretable model locally around it [2], and SHAP assigns each feature an importance value for a particular prediction [3].
Yeo et al. review explainability methods for deep learning in finance [6]. Hadji Misheva et al. survey explanations for AI-based risk management and stress that the right explanation depends on the audience [5]. These methods explain a model. In FinRisk AI the scoring rules are already transparent, so the open question is communication: turning known numbers and known rules into advice that a non-specialist can act on.

3. LLMs in finance:
BloombergGPT is a 50-billion-parameter model trained on a mix of financial and general data [7]. FinGPT pursues a similar goal through an open-source, data-centric approach [8]. Both address financial language tasks.
Neither is a framework for explaining one person's risk assessment, and both inherit the hallucination problem that Ji et al. describe for neural text generation [12].

4. Multi-agent LLM systems:
Guo et al. survey LLM-based multi-agent systems and describe how agents are profiled, how they communicate and which structures they use, layered ones included [9].
MetaGPT encodes standard operating procedures into prompt sequences to reduce the logical inconsistencies that appear when LLMs are chained naively [10]. TradingAgents gives LLM agents analyst, researcher, trader and risk-management roles for stock trading [11].

These works show that role specialisation is workable. They address software development and trading, not the explainable assessment of a household budget.

Gap in research:
In the sources reviewed here we found strong predictive models, general explanation methods, finance-adapted language models and multi-agent patterns, but no design that combines them to explain personal financial risk in plain language while keeping computed facts clearly apart from generated text. That combination is what FinRisk AI proposes.

III. PROBLEM STATEMENT AND RESEARCH MOTIVATION:

Rule-based assessment is easy to build and easy to audit, which is why debt-to-income cut-offs are still common. Its weakness is what it leaves for the user to work out.

Five limitations motivate this work.

a) Conventional indicators are read one at a time against fixed cut-offs.
b) A label such as "high risk" arrives without its reasons.
c) Advice is generic and not tied to the user's own figures.
d) Interactions between indicators are hard to see. A tolerable debt-to-income ratio with thin savings is a different situation from the same ratio with a large buffer.
e) Few designs use specialised AI agents for financial decision support in a form that separates computed facts from generated text.

FinRisk AI responds with a hierarchical multi-agent architecture intended to improve interpretability, modularity and accessibility. Table I sets out how it differs from a conventional rule-based assessment. The comparison describes the design. It does not rest on measured results.

TABLE I Traditional assessment and FinRisk AI

Aspect	Traditional approach	FinRisk AI
Output	A score or a label	Risk level, main drivers and a written explanation
Explanation	Generic text, or none	Written from the user's own figures
Advice	General tips	Tied to the indicators that were flagged
Arithmetic	Deterministic code	Deterministic code, kept out of the LLMs
Structure	One scoring routine	Separate processing, rules, agents and display
Maturity	Established and widely used	Prototype design, not yet validated

IV. PROPOSED SYSTEM AND METHODOLOGY:

1. Financial data collection:
The user supplies monthly income (*I*), monthly expenses excluding loan instalments (*E*), the total of monthly loan instalments (*L*), liquid savings (*S*) and a credit score (*C*). The input schema can hold further fields, such as dependants or insurance cover. The rules described in this paper do not use them.

2. Data processing and validation:
Before anything is computed, the backend checks that every field is present, numeric and within a plausible range, and that all amounts share a monthly basis. A yearly bonus, for instance, is spread over twelve months.

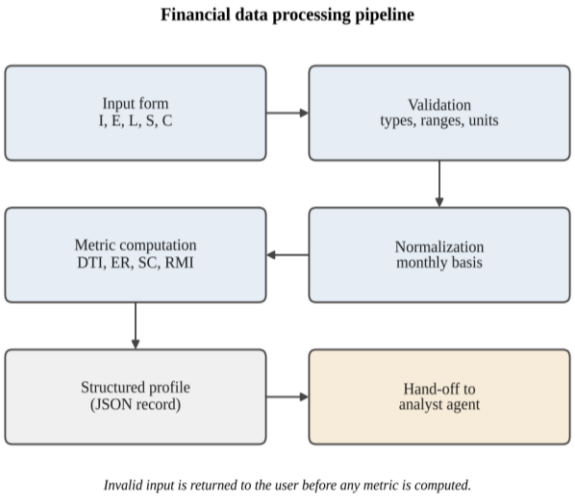

Fig. 3. Financial data processing pipeline.

Invalid input goes back to the user with a message that names the field. Every later stage trusts these values, so this is the only place where bad data can be stopped. Fig. 3 shows the pipeline.

3. Structured profile:
The result is a single JSON record that holds the cleaned inputs, the computed indicators and a timestamp. The agents and the dashboard all read from this record. A figure shown on screen and the same figure quoted in an explanation therefore come from one source.

V. SYSTEM ARCHITECTURE:

1. Layered overview:
Fig. 1 shows the five components in the order in which a result is assembled. The first two collect and process the data. The Financial Analyst Agent examines the processed profile and writes notes on patterns, for example that expenses and loan payments together take most of the income, or that savings would cover under two months of outgoings.

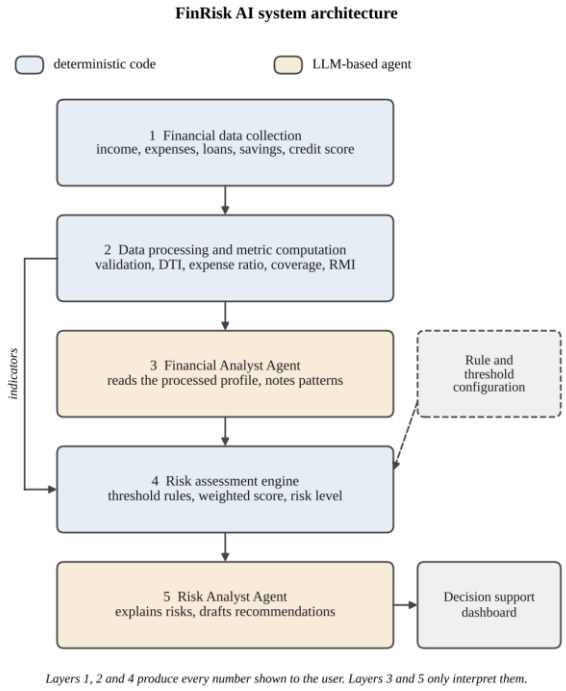

Fig. 1. Overall architecture of the proposed FinRisk AI framework.

The risk assessment engine applies predefined rules to the computed indicators and returns a risk level with

flags. It reads the indicators directly and not the analyst's text, so an odd output from the language model cannot change the risk level. The Risk Analyst Agent receives the metrics, the flags and the analyst notes, explains the main risks and drafts recommendations. The dashboard presents the outcome.

2. Why a hierarchy:
Control flows in one direction, from deterministic code to the analyst agent to the risk agent. This has three practical consequences.
Each part can be tested alone: the processing code with unit tests, the engine with boundary cases and the agents with fixed input records.

A part can be replaced, such as a different LLM or a different rule set, without touching the others. And every number in an explanation can be traced upward to a deterministic step.

Mathematical formulation and financial risk indicators:
Let I be monthly income, E monthly expenses excluding loan instalments, L the total of monthly loan instalments, S liquid savings and C the credit score. Income must be positive, and $E + L$ must be positive for the coverage ratio to be defined.
The debt-to-income ratio gives the share of income already committed to loans:

$$DTI = \frac{L}{I} \quad (1)$$

The expense-to-income ratio gives the share of income spent on living costs:

$$ER = \frac{E}{I} \quad (2)$$

Remaining monthly income is the surplus, or deficit, left after expenses and loan payments:

$$RMI = I - E - L \quad (3)$$

Savings coverage is the number of months that savings could carry total outgoings if income stopped:

$$SC = \frac{S}{E+L} \quad (4)$$

The surplus is also expressed as a share of income, so that households of different size can be compared:

$$\rho = \frac{RMI}{I} \quad (5)$$

Table II summarises the indicators.

TABLE II Financial indicators and formulations

Indicator	Formula	Reading
DTI	L / I	Share of income used for loan payments
ER	E / I	Share of income used for living costs
RMI	$I - E - L$	Monthly surplus or deficit
SC	$S / (E + L)$	Months of outgoings the savings cover
ρ	RMI / I	Surplus as a share of income

Each indicator, together with the credit score, is mapped to a sub-score $s_k \in \{0, 1, 2\}$, where 0, 1 and 2 stand for low, moderate and high risk. Table III gives the cut-offs and the weights w_k. They are illustrative defaults chosen by the authors, they are configurable, and they have not been calibrated against lending outcomes. The credit score is assumed to run from 300 to 900. A different scale needs different cut-offs.

TABLE III Sub-score cut-offs and weights

Indicator	w_k	$s = 0$	$s = 1$	$s = 2$
DTI	0.30	≤ 0.30	0.30 to 0.43	> 0.43
ER	0.20	≤ 0.50	0.50 to 0.70	> 0.70
SC	0.25	≥ 6	3 to < 6	< 3
C	0.15	≥ 750	650 to 749	< 650
ρ	0.10	≥ 0.20	0.10 to < 0.20	< 0.10

The weighted composite score divides by its largest possible value, so it always lies between 0 and 1:

$$R = \frac{\sum_{k=1}^{5} w_k s_k}{2 \sum_{k=1}^{5} w_k} \quad (6)$$

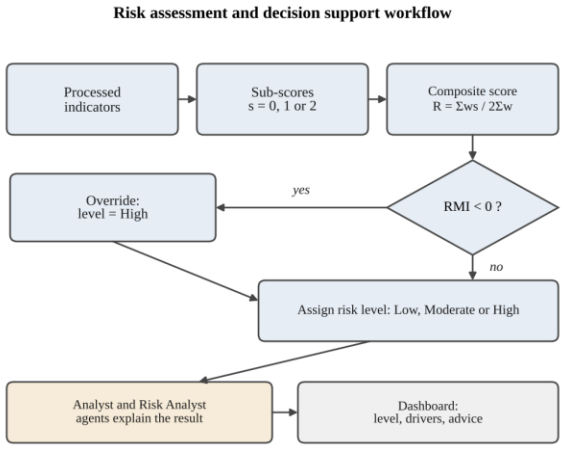

Fig. 4. Risk assessment and decision support workflow.

The risk level is High if RMI < 0 or $R \geq 0.66$, Moderate if $0.33 \leq R < 0.66$, and Low otherwise. The first condition is a hard rule. A household that spends more than it earns is at risk whatever its other numbers say, so no weighting is allowed to hide that. Fig. 4 shows the sequence.

VI. MULTI-AGENT FRAMEWORK:

1. Why more than one agent:

One prompt that carries the whole task has to analyse the profile, interpret the rules and write advice in a single pass. Guo et al. list the specialisation of LLMs into distinct agents with different capabilities as a main attraction of multi-agent designs [9]. Two agents are enough here. The Financial Analyst Agent describes and the Risk Analyst Agent explains and advises. Each prompt gets a narrow job, and the second agent receives structured analyst notes that its own text can be checked against. MetaGPT reports that chaining LLMs naively can cascade hallucinations [10]. A fixed, structured hand-off between roles is one way to limit that.

2. Roles and responsibilities:

Table IV lists the components. Only two of the five are language models.

TABLE IV Roles and responsibilities of the components

Component	Type	Responsibility
Data processing	Code	Validates input and computes DTI, ER, RMI, SC and ρ
Financial Analyst Agent	LLM	Reads the processed profile and notes patterns and how indicators relate
Risk assessment engine	Code	Applies cut-offs and weights, returns risk level and flags
Risk Analyst Agent	LLM	Explains the risks, ranks the drivers, drafts recommendations
Dashboard	Code	Shows level, indicators, explanation and advice

3. Communication:

Fig. 2 shows how messages move. The data processing component hands the metrics to the analyst agent as JSON, and the same metrics to the rule engine. The engine sends its level and flags to the Risk Analyst Agent. The analyst agent sends short structured notes, each naming the indicator it rests on. The agents do not call each other freely and there are no loops. The pipeline is sequential, so its behaviour is predictable and each step can be logged.

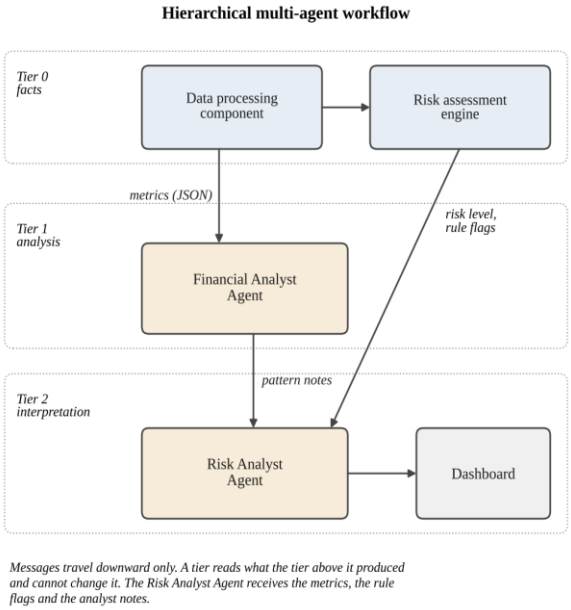

Fig. 2. Hierarchical multi-agent workflow.

4. Deterministic calculation and LLM reasoning:

The dividing line is simple. Code owns every number, every threshold and the risk level. The language models own the ordering of concerns, the wording, the links drawn between indicators and the suggestions. The framework specifies that both agents are told to use only the figures supplied, to quote them as given, to introduce no new ones and to avoid naming specific financial products. A check that compares the numbers in generated text with the record is part of the proposed evaluation. It is not a claim about the current prototype. The system supports a decision and does not make one. The user stays in control, and the output carries a notice that it is information and not professional financial advice.

VII. IMPLEMENTATION AND TECHNOLOGIES USED:

The system is organised as a Python backend that exposes the pipeline to a web dashboard (Table V). The LLM is reached through an API, so the model can be swapped. Because the numbers do not depend on the model, replacing it changes the wording of

explanations and nothing else. Prompts are assembled by application code from the structured record, so one record always yields the same prompt.

TABLE V Technologies and tools

Layer	Technology	Purpose
Calculation and rules	Python	Validation, indicators, scoring, risk level
Agents	LLM through an API	Pattern notes, explanation, recommendations
Orchestration	Python application code	Sequencing, prompt assembly, message passing
Data exchange	JSON	Structured hand-off between tiers
Presentation	Web dashboard	Indicators, risk level and written advice

VIII. RESULTS AND DISCUSSION:

No experiments have been run on this framework, and this section reports none. It does three other things. It walks through a hypothetical profile to show how the parts fit together.
It sets out the evaluation we propose. And it discusses expected benefits, kept apart from anything that has been demonstrated.

1. Illustrative example (hypothetical):
Take an invented household, with amounts in an arbitrary currency unit: I = 60,000, E = 27,000, L = 21,000, S = 90,000 and C = 700. The indicators follow from (1) to (5): DTI = 0.35, ER = 0.45, RMI = 12,000, ρ = 0.20 and SC = 90,000 / 48,000 ≈ 1.9 months. Table III gives sub-scores of 1 for DTI, 0 for ER, 2 for SC, 1 for the credit score and 0 for ρ. With the weights in Table III, (6) gives R = (0.30 + 0 + 0.50 + 0.15 + 0) / 2 = 0.475, so the level is Moderate. Savings coverage supplies 0.50 of the 0.95 weighted points, just over half, and DTI supplies 0.30.

A Risk Analyst Agent working from this record would be expected to say that the main exposure is the thin savings buffer, that the loan share is moderate, and that building savings should come before further borrowing. That is the kind of explanation the design aims for. We have not measured how well any model produces it.

2. Proposed evaluation methodology:
The evaluation would use synthetic profiles spread across the range of each indicator.
Real user data would need consent and a privacy review first.

a) Numerical correctness. Compare computed indicators with hand-calculated values, including values at each cut-off, zero savings and RMI < 0. This layer is deterministic, so every case should pass.
b) Rule consistency. Check that worsening any single indicator never lowers R.
c) Explanation faithfulness. Count the numerical statements in an agent's output and compare them with the record, giving the numeric grounding rate in (7). Reviewers would also mark each claim as supported or unsupported by the record.
d) Usefulness. Reviewers with finance or teaching experience rate clarity and actionability on a five-point scale for three conditions: the rule engine alone, a single LLM given the raw profile, and FinRisk AI.
e) Stability and cost. Run each profile several times and record how much the recommendations vary, along with latency and token use. The risk level itself should not vary.

$$NGR = \frac{N_{match}}{N_{total}} \qquad (7)$$

Here N_{total} is the number of numerical statements in the generated text and N_{match} is the number that agree with the record.

3. Expected benefits:
We expect the separation of calculation from language to make errors easier to locate. A wrong figure in an explanation would point at an agent, and a wrong risk level would point at the rules or the data. We also expect users to find an explanation that quotes their own numbers easier to act on than a bare label. These are expectations. The proposed evaluation is how they would be tested.

IX. ADVANTAGES AND LIMITATIONS:

The main advantage is traceability. The figures are computed by code and the level by a short rule set that anyone can read. The modular structure lets a team swap the model, change a threshold or test one part alone. The cut-offs and weights sit in a configuration,

which makes them easy to adjust for a different country or a different lender.

The limitations are real. The cut-offs and weights are not calibrated against repayment data. The indicator set is small. It ignores income stability, dependants, interest rates and assets. The inputs are self-reported and unverified. The language models can still misstate or leave out a figure, even when told not to [12]. When an external LLM API is used, the user's financial data leaves the system, which is a privacy cost. And, as stated above, the framework has not been evaluated.

X. FUTURE SCOPE:

Calibrating the cut-offs and weights against repayment outcomes is the first task. A learned risk model could then be added next to the rules, with SHAP [3] or LIME [2] attributions passed to the same agents. A what-if mode would let a user change one figure, have the deterministic layer recompute the indicators and have the agents explain the difference. The numeric grounding check in (7) could run inside the pipeline and block any explanation that quotes a figure missing from the record. Locally hosted open models [8] would keep financial data on the user's side. A user study with the reviewers described above, and output in more than one language, would follow.

XI. CONCLUSION:

FinRisk AI divides the work three ways: code for the numbers, rules for the risk level and language models for the explanation. The paper has described the architecture, the formulas and the agent roles in enough detail to be reimplemented, and has walked through a hypothetical profile. The framework has not been tested. The next steps are the numerical and faithfulness tests, followed by calibration of the thresholds on real repayment data.

REFERENCES:

[1] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (XAI)," *IEEE Access*, vol. 6, pp. 52138–52160, 2018, doi: 10.1109/ACCESS.2018.2870052.

[2] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?' Explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, San Francisco, CA, USA, 2016, pp. 1135–1144, doi: 10.1145/2939672.2939778.

[3] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Adv. Neural Inf. Process. Syst. 30 (NIPS 2017)*, Long Beach, CA, USA, 2017.

[4] S. Lessmann, B. Baesens, H.-V. Seow, and L. C. Thomas, "Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research," *Eur. J. Oper. Res.*, vol. 247, no. 1, pp. 124–136, 2015, doi: 10.1016/j.ejor.2015.05.030.

[5] B. Hadji Misheva, D. Jaggi, J.-A. Posth, T. Gramespacher, and J. Osterrieder, "Audience-dependent explanations for AI-based risk management tools: A survey," *Front. Artif. Intell.*, vol. 4, Art. no. 794996, 2021, doi: 10.3389/frai.2021.794996.

[6] W. J. Yeo, W. van der Heever, R. Mao, E. Cambria, R. Satapathy, and G. Mengaldo, "A comprehensive review on financial explainable AI," arXiv:2309.11960, 2023.

[7] S. Wu et al., "BloombergGPT: A large language model for finance," arXiv:2303.17564, 2023.

[8] H. Yang, X.-Y. Liu, and C. D. Wang, "FinGPT: Open-source financial large language models," in *Proc. FinLLM Symp. at IJCAI 2023*, arXiv:2306.06031.

[9] T. Guo et al., "Large language model based multi-agents: A survey of progress and challenges," in *Proc. 33rd Int. Joint Conf. Artif. Intell. (IJCAI)*, Jeju, South Korea, 2024, pp. 8048–8057, doi: 10.24963/ijcai.2024/890.

[10] S. Hong et al., "MetaGPT: Meta programming for a multi-agent collaborative framework," in *Proc. 12th Int. Conf. Learn. Representations (ICLR)*, 2024.

[11] Y. Xiao, E. Sun, D. Luo, and W. Wang, "TradingAgents: Multi-agents LLM financial trading framework," arXiv:2412.20138, 2024.

[12] Z. Ji et al., "Survey of hallucination in natural language generation," *ACM Comput. Surv.*, vol. 55, no. 12, Art. no. 248, pp. 1–38, 2023, doi: 10.1145/3571730.