

CARE - Controlled AI Reasoning for Enterprises

Harshil Lodhiya

lodhiyaharshil@gmail.com

SlicedHealth, Inc. <https://orcid.org/0009-0005-5798-9176>

Research Article

Keywords: Ethical AI, responsible AI, enterprise AI, large language models, LLM governance, model routing, text-to-SQL, schema-aware reasoning, AI safety, human-in-the-loop systems, AI workforce impact

Posted Date: September 1st, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-10763331/v2>

License:  This work is licensed under a Creative Commons Attribution 4.0 International License.
[Read Full License](#)

Additional Declarations: The authors declare no competing interests.

CARE - Controlled AI Reasoning for Enterprises

Benchmark-Grounded Runtime Governance for Enterprise LLM Systems

Harshil Lodhiya

August 11, 2026

Abstract

Enterprise adoption of large language models is moving faster than the governance systems designed to control them. Many organizations now use LLMs for analytics, document generation, customer support, software development, decision support, and workflow automation. Yet ethical AI is still often discussed as a set of high-level principles: fairness, transparency, privacy, accountability, and human oversight. These principles matter, but they are not enough on their own. In production environments, ethical AI must be translated into technical controls that decide what data a model may see, which model should handle a request, whether an action should be executed, when a human should intervene, and how the outcome should be audited.

This paper introduces and evaluates CARE, Controlled AI Reasoning for Enterprises, as a runtime governance architecture for enterprise LLM systems. CARE treats ethical governance as a runtime control problem. It combines request risk classification, data boundary enforcement, schema-aware context construction, dynamic model routing, constrained generation, execution validation, audit logging, and workforce impact review. Using the open BIRD Mini-Dev text-to-SQL benchmark, the paper evaluates CARE as a dataset-grounded governance simulation over 500 database-access tasks across 11 SQLite databases. The benchmark shows that oracle schema-aware pruning reduces average table exposure from 7.26 to 2.06 tables per request and average column exposure from 76.86 to 31.74 columns per request, corresponding to average reductions of 65.94% and 56.45%, respectively. Deterministic CARE policy rules are then used to simulate risk classification, routing, and governance outcomes over the same benchmark tasks.

The central argument is simple: ethical AI is not achieved by asking models to behave responsibly; it is achieved by engineering the conditions under which models reason and act.

Keywords

Ethical AI; responsible AI; enterprise AI; large language models; LLM governance; model routing; text-to-SQL; schema-aware reasoning; AI safety; human-in-the-loop systems; AI workforce impact.

1. Introduction

The ethical risks of artificial intelligence are often described in human terms: unfair decisions, privacy violations, opaque reasoning, unsafe automation, and the displacement of meaningful work. These

descriptions are valid, but they create an implementation gap. A chief information officer, machine learning engineer, compliance lead, or enterprise architect cannot deploy "fairness" or "accountability" directly. They must deploy access controls, routing policies, validation layers, review workflows, logging systems, and measurable thresholds.

That gap is becoming more urgent as large language models move from experimental assistants into enterprise workflows. LLMs are no longer limited to drafting emails or summarizing documents. They increasingly generate SQL queries, call APIs, classify customers, assist with hiring workflows, review contracts, triage support tickets, write code, and recommend operational decisions. In this setting, ethical AI cannot remain a document attached to a model deployment. It must become part of the execution path.

CARE, short for Controlled AI Reasoning for Enterprises, is designed for that execution path. CARE does not assume that a single model can be made safe by prompt instructions alone. Instead, it assumes that enterprise AI systems are socio-technical systems: they involve people, models, databases, policies, interfaces, logs, approvals, and organizational incentives. The ethical behavior of such systems depends less on one model's general alignment and more on the controls surrounding the model at the moment it is asked to reason or act.

The architecture connects three practical observations.

First, automation affects people before it affects balance sheets. Hyper-automation can create insecurity, progression collapse, and a loss of perceived agency when employees see AI systems absorbing visible tasks while leaving unclear paths for advancement. Ethical AI must therefore include workforce impact, not only model performance.

Second, not every task deserves the same model. Some requests are low-risk and can be routed to small, low-cost models. Others involve sensitive data, regulated decisions, or complex reasoning and require stronger controls, private deployment, or human review. Ethical AI therefore requires model selection as a governance function, not only an optimization function.

Third, enterprise data is structured, permissioned, and context-dependent. An LLM that generates SQL over a relational database is not merely answering a question; it is crossing a boundary into operational data. Ethical AI therefore requires schema awareness, context minimization, permission checks, and execution validation.

CARE combines these observations into a runtime architecture for pre-deployment analysis, system design, and empirical evaluation. This draft positions CARE as a benchmark-grounded architecture, with measured schema-exposure results clearly separated from simulated routing and governance outcomes.

2. Background and Motivation

Responsible AI frameworks have helped organizations name important values, including fairness, explainability, transparency, privacy, robustness, and accountability. However, many enterprise failures do not occur because these values are unknown. They occur because the values are not translated into enforceable system behavior.

For example, a policy may state that sensitive employee information should not be exposed to external AI services. But unless the system classifies requests, tags sensitive fields, routes restricted prompts away from public models, and logs the decision, the policy remains dependent on user judgment. A policy may state that humans should remain accountable for high-impact decisions. But unless the workflow has explicit approval gates and traceable review records, accountability can dissolve into interface design. A policy may state that AI should not make unauthorized database queries. But unless generated SQL is checked before execution, a model can hallucinate joins, expose restricted columns, or run expensive queries.

The shift from principle to control is especially important for LLMs because they blur boundaries that older software kept separate. A single natural language request can combine search, reasoning, code generation, data access, and action execution. This flexibility is the reason LLMs are useful, but it is also the reason they require runtime governance.

CARE is motivated by the need to make ethical AI operational at the point of use. It asks a practical question: before an enterprise AI system answers or acts, what controls should it pass through?

3. Related Work and Positioning

CARE is positioned at the intersection of AI governance, selective model routing, constrained generation, enterprise data access, and human-centered automation. It does not replace these research streams. Its purpose is to connect them into a runtime architecture that enterprises can reason about and evaluate.

3.1 AI Governance and Risk Management

The most direct governance foundation for CARE is the NIST Artificial Intelligence Risk Management Framework. NIST frames trustworthy AI through characteristics such as validity, reliability, safety, security, resilience, accountability, transparency, explainability, privacy enhancement, and harmful-bias management, and it organizes AI risk management around govern, map, measure, and manage functions (NIST, 2023). CARE is aligned with this view, but it narrows the implementation question: how should those risk-management functions appear inside the execution path of an LLM application?

The EU AI Act similarly takes a risk-based approach, with stricter obligations for high-risk systems, including risk management, documentation, logging, transparency, human oversight, robustness, cybersecurity, and accuracy (European Union, 2024). CARE is not written as a legal compliance framework, but its architecture maps naturally onto these obligations. Risk classification, audit logging, human review, and validation are not optional add-ons in high-impact systems; they are the mechanism by which governance becomes observable.

CARE also draws from socio-technical critiques of fairness and responsible AI. Selbst, Boyd, Friedler, Venkatasubramanian, and Vertesi argue that technical abstractions can fail when they ignore the social context in which systems operate (Selbst et al., 2019). This is important for CARE because enterprise LLM systems are not merely model pipelines. They are embedded in job roles, access rules, organizational incentives, review processes, and power relationships. A governance system that controls data and models but ignores the humans who absorb verification work is incomplete.

3.2 Documentation, Evaluation, and Auditability

Prior work on Model Cards and Datasheets for Datasets established that responsible AI requires structured documentation of model behavior, intended use, limitations, data provenance, and evaluation context (Mitchell et al., 2019; Gebru et al., 2021). CARE extends this documentation logic into runtime governance. Instead of treating documentation as a static artifact produced before release, CARE treats logs, routing decisions, validation results, and review outcomes as living governance evidence.

The HELM benchmark further shows why single-metric evaluation is insufficient for language models (Liang et al., 2023). Accuracy alone does not capture robustness, fairness, bias, toxicity, calibration, or efficiency. CARE adopts the same spirit at the system level. A governed enterprise AI system should be evaluated not only by answer quality, but also by data exposure, routing appropriateness, blocked unsafe actions, escalation behavior, cost, latency, and human review burden.

3.3 Model Routing and Selective Inference

Model routing research supports CARE's view that model selection is itself a system-design problem. FrugalGPT shows that LLM cascades can adaptively select models to reduce cost while preserving or improving performance (Chen et al., 2024). RouteLLM similarly shows that routers can dynamically select between stronger and weaker LLMs to reduce cost while preserving quality (Ong et al., 2025). Confidence-based routing work studies when a system should accept an answer, route to another expert, or fall back to safer behavior (Chuang et al., 2025).

CARE builds on this line of work but adds governance constraints. In CARE, the router is not only optimizing cost and performance. It also considers data sensitivity, vendor eligibility, regulatory exposure, human-impact risk, and the need for auditability. A cheaper model is not acceptable if it violates a data boundary. A more capable external model is not acceptable if the request contains restricted employee or customer data. This turns model routing from an inference optimization problem into a policy-governed control point.

3.4 Text-to-SQL, Schema Control, and Constrained Generation

The text-to-SQL literature is central to CARE because enterprise LLMs increasingly reason over structured data. Spider established a difficult cross-domain benchmark for translating natural language into SQL over complex schemas (Yu et al., 2018). Later systems such as PICARD showed that constrained decoding can reduce invalid formal-language outputs by rejecting inadmissible tokens during generation (Scholak et al., 2021). DIN-SQL demonstrated that decomposing text-to-SQL into subproblems and adding self-correction can improve LLM-based SQL generation (Pourreza & Rafiei, 2023).

CARE uses these lessons in a broader governance context. SQL generation is not only a semantic parsing problem. In an enterprise setting, it is also a permission, privacy, and execution-risk problem. A generated query can be syntactically valid and still be unauthorized, overly expensive, privacy-invasive, or operationally misleading. CARE therefore combines schema-aware context construction with authorization checks, execution validation, and audit logging.

3.5 Human Oversight, Automation Bias, and Work

Human oversight has a long history in automation research. Parasuraman and Riley's work on automation use, misuse, disuse, and abuse remains highly relevant to LLM systems (Parasuraman & Riley, 1997). Their distinction helps explain why "human-in-the-loop" is not automatically sufficient: humans may over-rely on automation, underuse it, or be assigned poorly designed supervisory roles after automation decisions have already constrained the outcome.

Research on automation bias and complacency shows that people may insufficiently verify automated recommendations, especially when systems appear reliable (Bahner et al., 2008). CARE responds to this by making verification explicit and measurable. Reviewers need evidence, authority, escalation paths, and time; otherwise, human oversight becomes symbolic.

Labor economics and organizational research also matter. Acemoglu and Restrepo's task-based framework describes how automation displaces some tasks while creating productivity effects and new task demand (Acemoglu & Restrepo, 2018). Kellogg, Valentine, and Christin's work on algorithmic control shows how algorithms can direct, evaluate, discipline, and reshape workers (Kellogg et al., 2020). CARE therefore includes workforce impact review as a first-class layer. Ethical enterprise AI should measure not only whether a model is accurate, but also whether deployment creates hidden verification labor, role compression, or progression collapse.

4. Author Research Foundation

This paper synthesizes ideas from the author's prior research into a single governance architecture.

The first foundation is the study of hyper-automation anxiety and progression collapse in prompt-driven engineering (Lodhiya, 2026a). That work examines how AI-enabled automation can change employee perceptions of security, value, and career development. Its relevance to CARE is direct: a technically safe AI system can still be ethically incomplete if it accelerates task displacement without redesigning accountability, training, and progression pathways.

The second foundation is LLM-Advisor, a dynamic model selection and query routing approach for heterogeneous multi-LLM architectures (Lodhiya, 2026b). LLM-Advisor frames model choice as a decision problem involving cost, latency, capability, and task fit. CARE extends this idea by adding ethical and governance dimensions to routing, including data sensitivity, regulatory exposure, and human-impact risk.

The third foundation is schema-aware query translation and tabular reasoning for enterprise databases (Lodhiya, 2026c). That work addresses the risks of full schema injection, hallucinated SQL, and unsafe query execution by using metadata extraction, schema relationship graphs, context pruning, and execution-guided validation. CARE uses this as the basis for its data boundary and execution validation layers.

The fourth foundation is the author's JCSTS article associated with DOI 10.32996/jcsts.2026.8.8.17. The exact integration of this article should be finalized after its title and abstract are confirmed in the manuscript record. In the current draft, it is reserved as a supporting pillar rather than assumed into the argument.

5. CARE Architecture Overview

CARE is a runtime governance architecture for enterprise LLM systems. It is not a single model, product, or compliance checklist. It is a system pattern that places control points around AI reasoning.

Figure 1 presents the CARE runtime governance architecture. The figure shows the request moving through intent classification, data boundary control, schema-aware context construction, model routing, constrained generation, execution validation, human escalation, audit logging, and workforce impact review. This visual matters because CARE is not just a list of principles; it is an execution path.

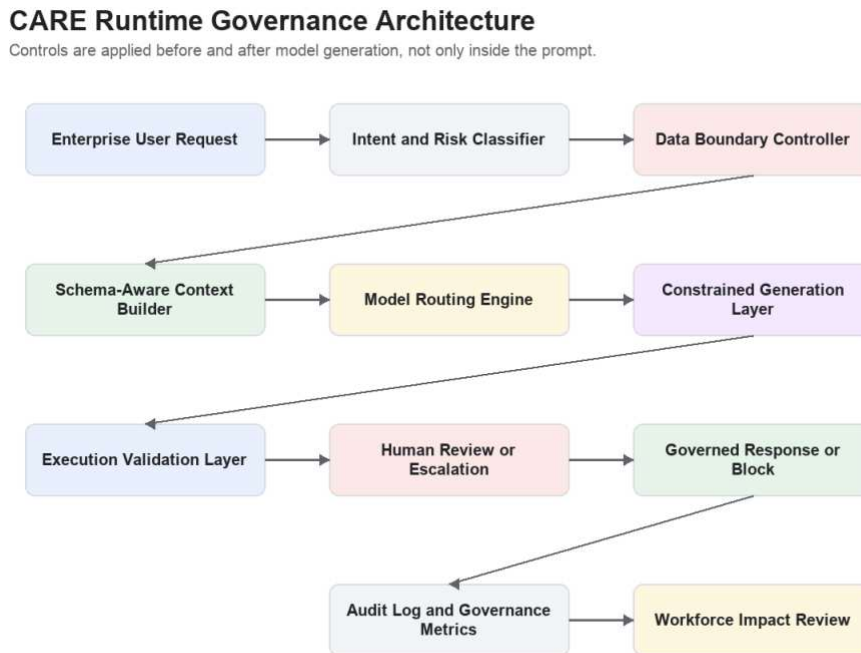


Figure 1. CARE runtime governance architecture.

At a high level, CARE follows this sequence:

1. A user submits an enterprise request.
2. The system classifies the intent and risk level.
3. The system determines what data, if any, the request is allowed to access.
4. The system builds a minimal context package instead of exposing broad data.
5. The system routes the request to an appropriate model or model class.
6. The model generates a response, query, structured object, or proposed action.
7. The system validates the generated output before it is returned or executed.
8. High-risk outputs are escalated to human review.
9. The system logs the decision path and updates governance metrics.

This architecture reflects a central design principle: model output should not be the first enforceable event in an enterprise AI workflow. The enforceable events should occur before and after generation through policies, constraints, and validation.

Figure 2 shows CARE as a layered runtime governance stack. **Figure 3** shows the control flow among the user, classifier, data boundary controller, model router, selected LLM, validator, reviewer, and audit log.



Figure 2. CARE runtime governance stack.

CARE Runtime Control Flow

The model response is treated as a proposal that passes through validation and escalation logic.

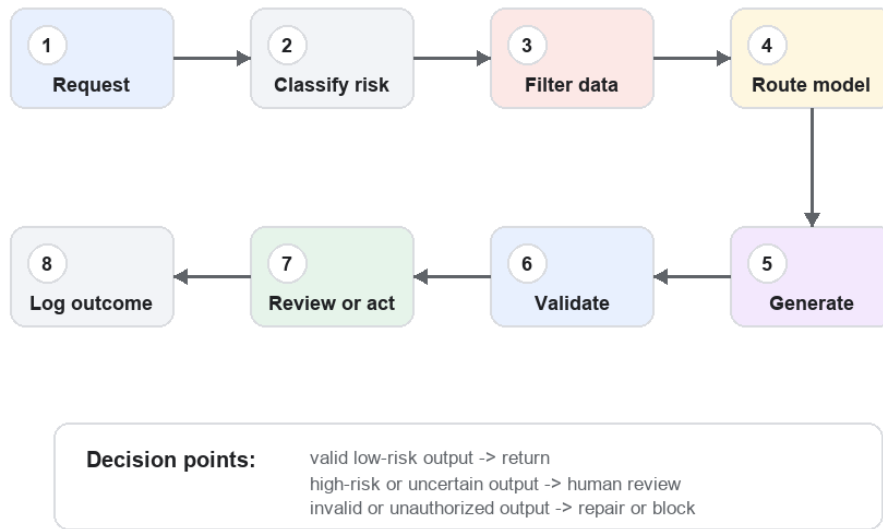


Figure 3. CARE runtime control flow.

6. Technical Components

6.1 Intent and Risk Classifier

The intent and risk classifier determines what kind of request the user is making and what level of control is required. A request to summarize a public marketing document does not require the same governance path as a request to analyze employee performance, generate SQL over customer records, or modify access permissions.

CARE classifies requests across at least six risk categories:

- Low-risk productivity requests
- Business-sensitive information requests
- Regulated or legally sensitive requests
- Data-access and analytics requests
- Action-bearing workflow requests
- Human-impacting requests involving employees, candidates, customers, or patients

The classifier can be implemented using a combination of rules, metadata, and model-based classification. In early deployments, a conservative hybrid approach is preferable: deterministic rules catch obvious restricted cases, while an LLM classifier handles ambiguous natural language. Low-confidence classifications should escalate rather than silently downgrade risk.

6.2 Data Boundary Controller

The data boundary controller decides what information the AI system may access. This component is where privacy and confidentiality become technical controls.

In enterprise systems, data sensitivity is rarely uniform. One table may contain public product names, confidential margins, customer identifiers, and regulated personal information. CARE therefore assumes that governance must operate at the field, table, document, and source level.

The controller may use:

- Role-based access control
- Attribute-based access control
- Column-level sensitivity tags
- PII and secret detection
- Data retention rules
- Tenant or customer boundary checks
- Vendor and deployment restrictions

The output of this layer is not simply "allow" or "deny." It can produce a filtered context, a redacted context, a private-model-only route, a human-review requirement, or a full block.

6.3 Schema-Aware Context Builder

For enterprise analytics, one of the most important CARE components is the schema-aware context builder. LLMs often perform poorly when given large relational schemas without structure or constraints. Full schema injection can waste tokens, increase latency, expose unnecessary metadata, and cause hallucinated joins.

CARE uses schema-aware context construction to minimize exposure while preserving task relevance. The system extracts metadata from the database, builds a schema relationship graph, identifies relevant tables and columns, prunes unrelated joins, and passes only the necessary schema subgraph to the model.

This layer supports both technical efficiency and ethical data minimization. The model should not see every possible field simply because the user asked one business question. It should see the minimum authorized structure required to answer the request.

Key metrics for this layer include:

- Prompt token reduction
- Schema precision
- Join-path accuracy
- Invalid SQL rate
- Unauthorized column reference rate
- Execution success rate

6.4 Model Routing Engine

The model routing engine selects the model or model class used for a request. In many enterprise systems, model selection is treated mainly as a cost and performance decision. CARE treats it as a governance decision as well.

Routing should consider:

- Request complexity
- Data sensitivity
- Required reasoning depth
- Latency tolerance
- Cost constraints
- Regulatory exposure
- Vendor eligibility
- Need for private or local inference
- Human-impact risk

A low-risk formatting request may be routed to a small model. A complex analytical task over non-sensitive data may be routed to a high-capability external model. A request involving restricted customer or employee data may be routed to a private model or blocked if no approved model is available. A high-impact recommendation may require human review regardless of model choice.

The routing layer allows ethical AI to become proportional. CARE does not slow every request with the same controls. It applies heavier controls where the risk justifies them.

6.5 Constrained Generation Layer

Free-form generation is useful for conversation but risky for enterprise action. CARE encourages constrained outputs whenever the model's response may be executed, stored, audited, or used in a decision workflow.

Examples include:

- SQL represented as a validated query object
- API calls represented as typed JSON
- Policy decisions represented with reason codes
- Summaries linked to source citations
- Recommendations paired with confidence and escalation flags

The purpose is not to eliminate natural language. The purpose is to prevent natural language from becoming an uncontrolled execution interface.

6.6 Execution Validation Layer

The execution validation layer checks generated outputs before they affect enterprise systems. This is especially important for generated SQL, API calls, workflow actions, code patches, and security operations.

Validation may include:

- Syntax checking
- Permission checking
- Dry-run execution
- Query cost estimation
- Destructive action detection
- Restricted field detection
- Result-size limits
- Consistency checks
- Policy conformance checks

If validation fails, CARE can repair the output, ask for clarification, escalate to a stronger model, route to human review, or block execution. The important point is that the model's first answer is treated as a proposal, not an authority.

6.7 Human Accountability and Workforce Impact Layer

Many AI governance systems include "human-in-the-loop" as a phrase but do not define the loop. CARE treats human accountability as a measurable layer.

The system should specify:

- Which decisions require human approval
- What evidence the reviewer sees
- Whether the reviewer can override the AI
- How overrides are logged
- Which roles absorb verification work
- Whether automation changes progression pathways
- Whether employees receive training for new verifier or orchestrator responsibilities

This layer connects technical governance to organizational ethics. If AI systems remove entry-level work but create only senior-level review responsibilities, the organization may create progression collapse. If employees become responsible for verifying AI outputs without time, authority, or recognition, the organization may create hidden labor rather than productivity. CARE therefore includes workforce impact as part of ethical deployment, not as a separate HR afterthought.

7. Proposed Evaluation and Statistics

This draft does not claim that CARE has already been deployed across a production enterprise. Instead, it reports an initial open-benchmark evaluation and proposes additional evaluation paths for future empirical work.

7.1 Representative Request Simulation

A synthetic workload can be created with 500 to 1,000 representative enterprise AI requests across productivity, analytics, HR, finance, customer support, security, and workflow automation. Each request can be labeled by risk class, data sensitivity, action type, and expected review requirement.

The simulation would estimate:

- Distribution of risk categories
- Percentage of requests routed to small, standard, premium, or private models
- Percentage requiring human review
- Percentage blocked due to policy restrictions
- Estimated cost reduction from routing
- Estimated latency impact from context pruning
- Unsafe action interception rate

These statistics should be reported explicitly as simulation results, not production outcomes.

7.2 Open Benchmark Evaluation Using BIRD Mini-Dev

The initial dataset-grounded evaluation uses BIRD Mini-Dev. BIRD is a large-scale database-grounded text-to-SQL benchmark designed to reduce the gap between academic text-to-SQL tasks and real-world database interfaces (Li et al., 2023). The Mini-Dev SQLite split provides a manageable evaluation setting with 500 question-SQL pairs over 11 databases.

The current evaluation does not call an LLM. Instead, it uses benchmark schemas and gold SQL to measure how much schema exposure could be reduced by an oracle schema-aware context builder. It also applies deterministic CARE policy rules to classify benchmark requests by risk tier and simulate model routing and governance outcomes. This distinction is important: schema exposure reduction is directly measured from benchmark artifacts, while routing and review outcomes are policy simulation results.

In this first pass, CARE evaluates all 500 Mini-Dev examples. The benchmark contains 11 SQLite databases, with an average of 6.82 tables and 72.55 columns per database. Across requests, the average full-schema context contains 7.26 tables and 76.86 columns. An oracle CARE pruning policy based on gold SQL references reduces this to 2.06 tables and 31.74 columns per request. This corresponds to an average table exposure reduction of 65.94% and an average column exposure reduction of 56.45%.

Metric	Value
Examples evaluated	500
Databases used	11
Average full-schema tables per request	7.26
Average CARE-pruned tables per request	2.06
Average full-schema columns per request	76.86
Average CARE-pruned columns per request	31.74
Average table exposure reduction	65.94%
Average column exposure reduction	56.45%

The CARE risk simulation classifies 40 examples as low risk, 21 as business-sensitive, 83 as regulated, and 356 as human-impacting. The high human-impact rate reflects a conservative first-pass policy over

schemas containing person-like, student-like, customer-like, or user-like fields. In a production deployment, these rules would be calibrated using organization-specific sensitivity tags and review policies.

CARE risk tier	Count
Low risk	40
Business-sensitive	21
Regulated	83
Human-impacting	356

Under the corresponding CARE routing policy, 40 requests are routed to a small low-cost model, 21 to a standard enterprise model, 83 to a private or local restricted model, and 356 to a human-first review path. This result should not be interpreted as a universal enterprise distribution. It shows that CARE routing can be calculated reproducibly from explicit risk rules over an open benchmark.

CARE route	Count
Small low-cost model	40
Standard enterprise model	21
Private/local restricted model	83
Human-first review path	356

These results support a limited but useful claim: open database-grounded benchmarks can be used to evaluate ethical AI controls as measurable system behavior. CARE can report schema minimization, risk classification, routing, and governance outcomes without relying on private enterprise data or unverifiable production claims.

The generated artifact package includes four result figures derived from this benchmark: CARE context reduction, CARE risk distribution, CARE routing distribution, and CARE governance outcomes. These figures are generated from the benchmark output files rather than edited manually.

CARE Context Reduction on BIRD Mini-Dev

Measured from 500 open benchmark tasks using schemas and gold SQL references.

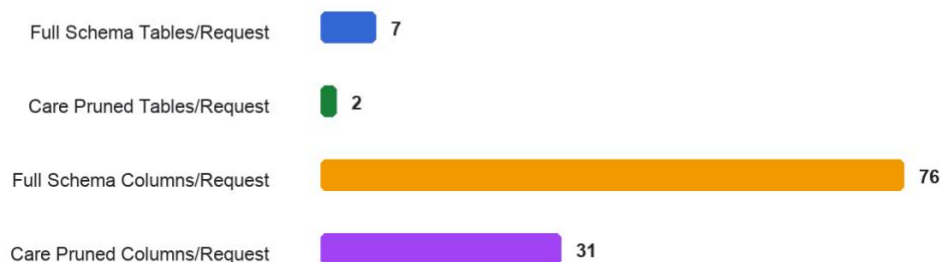


Figure 4. CARE context reduction on BIRD Mini-Dev.

CARE Risk Distribution on BIRD Mini-Dev

Deterministic CARE policy simulation over 500 benchmark tasks.

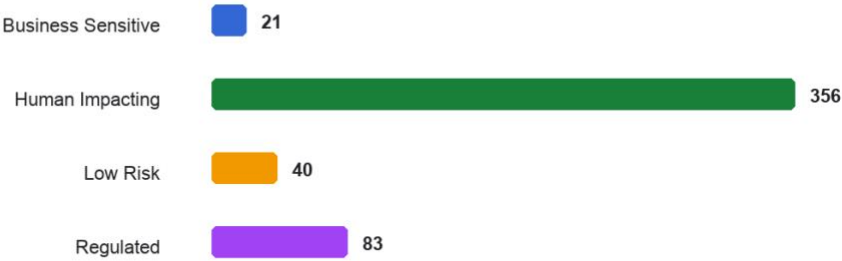


Figure 5. CARE risk distribution on BIRD Mini-Dev.

CARE Routing Distribution on BIRD Mini-Dev

Routes are simulated from explicit risk-tier rules.

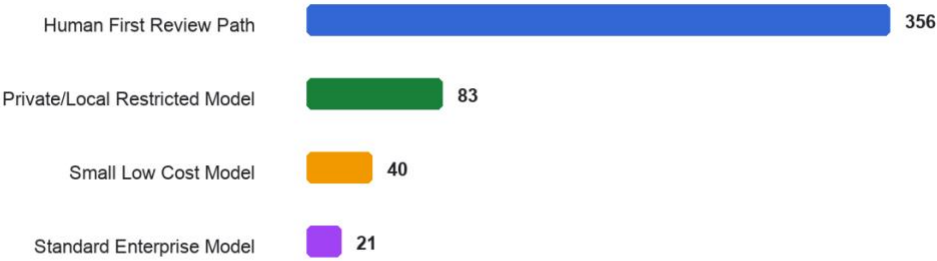


Figure 6. CARE routing distribution on BIRD Mini-Dev.

CARE Governance Outcomes on BIRD Mini-Dev

Outcomes separate direct responses, validation paths, and human review.



Figure 7. CARE governance outcomes on BIRD Mini-Dev.

7.3 Schema-Aware Query Benchmark

The schema-aware layer can be evaluated using public text-to-SQL benchmarks and representative enterprise schemas. The baseline would inject broad schema context into an LLM. CARE would use schema pruning, permission-aware context construction, and execution validation.

Metrics should include:

- Prompt token volume
- Execution accuracy
- Invalid SQL rate
- Unauthorized field reference rate
- Join hallucination rate
- End-to-end latency

This evaluation directly connects CARE to the author's prior schema-aware query translation work.

7.4 Risk-Control Ablation

CARE can also be evaluated by selectively disabling components:

- No controls
- Model routing only
- Data boundary controls only
- Schema-aware context only
- Full CARE

The purpose of ablation is to show whether ethical risk reduction comes from one component or from the combined architecture.

Expected metrics include unsafe output rate, cost per request, human review rate, blocked action rate, and execution success rate.

8. Illustrative Metrics Table

The following table defines the kinds of statistics CARE should report. Values should be generated through simulation or benchmark experiments before being presented as results.

Metric group	Metric	Why it matters
Routing	Premium model usage rate	Shows whether high-cost models are reserved for high-need tasks
Routing	Private model routing rate	Shows whether sensitive requests stay within approved inference boundaries
Cost	Average model cost per request	Measures whether governance can coexist with efficiency
Latency	Median response latency	Measures operational feasibility
Data governance	Prompt token volume	Measures context minimization
Data governance	Unauthorized column exposure rate	Measures privacy and permission enforcement
SQL reliability	Execution accuracy	Measures practical usefulness
Safety	Unsafe action block rate	Measures prevention of harmful execution
Accountability	Human review rate	Measures where human judgment is preserved
Workforce impact	Verification burden index	Measures hidden labor created by AI-assisted workflows

9. Initial Simulation Seed

To make the evaluation design concrete, an initial synthetic workload can be defined across 1,000 representative enterprise AI requests. This seed is not a production result. It is a structured starting point for a later simulation.

Request class	Share
General productivity	24.0%
SQL and business analytics	26.0%
HR and workforce questions	14.0%
Finance and compliance	15.0%
Customer support	11.0%
Security and administrative actions	10.0%

Under an illustrative CARE routing policy, low-risk and routine requests would be routed primarily to small or standard models, while sensitive, regulated, or action-bearing requests would be routed to private models, premium reasoning models, or human-first review paths.

CARE route	Share
Small low-cost model	33.0%
Standard enterprise model	27.0%

Premium reasoning model	18.0%
Private/local restricted model	14.0%
Human-first review path	8.0%

This distribution illustrates a key CARE principle: governance should be proportional. The system should not force every request through the most expensive model or the slowest approval workflow. It should reserve heavier controls for requests where data sensitivity, execution risk, or human impact justify them.

The draft figure set includes three illustrative charts derived from this seed: risk classification distribution, CARE routing distribution, and governance outcomes. These charts should remain labeled as synthetic until the simulation or benchmark is executed with a finalized workload.

10. Discussion

CARE reframes ethical AI as an engineering problem without reducing it to engineering alone. The architecture recognizes that enterprise AI systems involve real people, real data, real permissions, and real organizational consequences. This is why CARE includes both technical controls and workforce impact review.

The practical advantage of CARE is proportional governance. A common failure in enterprise AI policy is treating every use case as equally risky or relying entirely on user discretion. The first approach slows adoption. The second creates unmanaged risk. CARE offers a middle path: low-risk tasks move quickly, while sensitive and action-bearing tasks receive stronger controls.

The deeper ethical argument is that AI systems should not be judged only by outputs. They should be judged by the conditions under which outputs are produced. A correct answer generated from unauthorized data is still a governance failure. A cheap answer that routes sensitive data to an unapproved model is still a privacy failure. A productivity gain that quietly converts employees into uncompensated AI verifiers is still an organizational failure.

CARE is designed to make these failures visible before they become incidents.

11. Reproducibility and Artifact Availability

The CARE benchmark is designed to be reproducible with open data and standard Python. The companion artifact is available at <https://github.com/harshillodhiya/CARE> and contains scripts for downloading BIRD Mini-Dev, computing benchmark-grounded CARE statistics, validating generated outputs, and running unit tests. The benchmark uses only the Python standard library in its current form and does not require API keys or proprietary models.

The reproducibility workflow is:

```
python3 scripts/download_bird_minidev.py --output-dir data/raw
```

```
python3 scripts/care_benchmark.py --data-root data/raw --output-dir stats/generated --figures-dir figures/generated
```

```
python3 scripts/validate_outputs.py --output-dir stats/generated --figures-dir figures/generated
```

```
python3 -m unittest discover -s tests
```

The raw BIRD dataset is not redistributed with the artifact because of size. The benchmark outputs, generated charts, and source scripts are small enough to be included with the research repository.

12. Limitations

CARE is currently presented as a runtime governance architecture with an initial benchmark-grounded evaluation. Its effectiveness depends on the quality of risk taxonomies, data classification, access-control metadata, model evaluation, and organizational adoption. Poorly maintained sensitivity tags or vague review policies can weaken the architecture.

The BIRD Mini-Dev evaluation also has limits. The schema-exposure reduction results are measured from benchmark schemas and gold SQL references, which makes them reproducible but also oracle-guided. A deployed system would need to predict the relevant schema from the user's request before seeing gold SQL. The routing and governance results are deterministic CARE policy simulations, not observations from a production enterprise deployment.

The architecture itself introduces overhead. Risk classification, validation, routing, and logging add complexity to AI systems. Future work should measure when this overhead is justified and how much can be automated without weakening accountability.

Finally, workforce impact is difficult to quantify. Metrics such as verification burden and progression risk require organizational data that may not be easy to collect. Nevertheless, excluding these concerns would make ethical AI artificially narrow.

13. Conclusion

Enterprise AI needs more than responsible model behavior. It needs controlled reasoning environments. CARE introduces a runtime governance architecture that turns ethical AI principles into enforceable system components: risk classification, data boundaries, schema-aware context, model routing, constrained generation, validation, auditability, and human accountability.

The central contribution of CARE is its shift from principle-level ethics to execution-level governance. In enterprise settings, the question is not only whether an AI model can answer. The question is whether it should see the data, which model should reason over it, whether the output can be trusted, whether an action should be executed, and who remains accountable afterward.

CARE offers a starting architecture for answering those questions.

References

Acemoglu, D., & Restrepo, P. (2018). *Artificial Intelligence, Automation and Work*. NBER Working Paper No. 24196. <https://doi.org/10.3386/w24196>

- Bahner, J. E., Huper, A.-D., & Manzey, D. (2008). Misuse of automated decision aids: Complacency, automation bias and the impact of training experience. *International Journal of Human-Computer Studies*, 66(9), 688-699. <https://doi.org/10.1016/j.ijhcs.2008.06.001>
- Chen, L., Zaharia, M., & Zou, J. (2024). FrugalGPT: How to use large language models while reducing cost and improving performance. *Transactions on Machine Learning Research*. <https://mlanthology.org/tmlr/2024/chen2024tmlr-frugalgpt/>
- Chuang, Y.-N., Sarma, P. K., Gopalan, P., Boccio, J., Bolouki, S., Hu, X., & Zhou, H. (2025). Learning to route LLMs with confidence tokens. *Proceedings of the 42nd International Conference on Machine Learning*, PMLR 267, 10859-10878. <https://proceedings.mlr.press/v267/chuang25b.html>
- European Union. (2024). *Regulation (EU) 2024/1689 of the European Parliament and of the Council laying down harmonised rules on artificial intelligence*. <https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32024R1689>
- Gebru, T., Morgenstern, J., Vecchione, B., Vaughan, J. W., Wallach, H., Daume III, H., & Crawford, K. (2021). Datasheets for datasets. *Communications of the ACM*, 64(12), 86-92. <https://doi.org/10.1145/3458723>
- Lodhiya, H. (2026c). *Schema-Aware Query Translation and Tabular Reasoning for Enterprise Databases*. *International Journal of Research in Engineering, Science and Management*. <https://doi.org/10.65138/ijresm.v9i7.3489>
- Kellogg, K. C., Valentine, M. A., & Christin, A. (2020). Algorithms at work: The new contested terrain of control. *Academy of Management Annals*, 14(1), 366-410. <https://doi.org/10.5465/annals.2018.0174>
- Liang, P., Bommasani, R., Lee, T., Tsipras, D., Soylu, D., Yasunaga, M., Zhang, Y., Narayanan, D., Wu, Y., Kumar, A., Newman, B., Yuan, B., Yan, B., Zhang, C., Cosgrove, C. A., Manning, C. D., Re, C., Acosta-Navas, D., Hudson, D. A., Zelikman, E., Durmus, E., Ladhak, F., Rong, F., Ren, H., Yao, H., Wang, J., Santhanam, K., Orr, L., Zheng, L., Yuksekgonul, M., Suzgun, M., Kim, N., Guha, N., Chatterji, N. S., Khattab, O., Henderson, P., Huang, Q., Chi, R. A., Xie, S. M., Santurkar, S., Ganguli, S., Hashimoto, T., Icard, T., Zhang, T., Chaudhary, V., Wang, W., Li, X., Mai, Y., Zhang, Y., & Koreeda, Y. (2023). Holistic evaluation of language models. *Transactions on Machine Learning Research*. <https://openreview.net/forum?id=iO4LZibEqW>
- Lodhiya, H. (2026a). *Hyper-Automation Anxiety: Assessing Job Insecurity and Progression Collapse in the Age of Prompt-Driven Engineering*. *International Journal of Research in Engineering, Science and Management*. <https://doi.org/10.65138/ijresm.v9i7.3484>
- Li, J., Hui, B., Qu, G., Yang, J., Li, B., Li, B., Wang, B., Qin, B., Geng, R., Huo, N., Zhou, X., Ma, C., Li, G., Chang, K. C.-C., Huang, F., Cheng, R., & Li, Y. (2023). Can LLM already serve as a database interface? A big bench for large-scale database grounded text-to-SQLs. *Advances in Neural Information Processing Systems*, 36. https://proceedings.neurips.cc/paper_files/paper/2023/hash/83fc8fab1710363050bbd1d4b8cc0021-Abstract-Datasets_and_Benchmarks.html

- Lodhiya, H. (2026b). *LLM-Advisor: Dynamic Model Selection and Query Routing in Heterogeneous Multi-LLM Architectures*. International Journal of Research in Engineering, Science and Management. <https://doi.org/10.65138/ijresm.v9i7.3488>
- Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., Spitzer, E., Raji, I. D., & Gebru, T. (2019). Model cards for model reporting. *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 220-229. <https://doi.org/10.1145/3287560.3287596>
- NIST. (2023). *Artificial Intelligence Risk Management Framework (AI RMF 1.0)*. NIST AI 100-1. <https://doi.org/10.6028/NIST.AI.100-1>
- Ong, I., Almahairi, A., Wu, V., Chiang, W.-L., Wu, T., Gonzalez, J. E., Kadous, M. W., & Stoica, I. (2025). RouteLLM: Learning to route LLMs from preference data. *International Conference on Learning Representations*. https://proceedings.iclr.cc/paper_files/paper/2025/hash/5503a7c69d48a2f86fc00b3dc09de686-Abstract-Conference.html
- Lodhiya, H. (2026d). *Schema-Aware Query Translation and Tabular Reasoning for Enterprise Databases*. SSRN. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=7168019
- Parasuraman, R., & Riley, V. (1997). Humans and automation: Use, misuse, disuse, abuse. *Human Factors*, 39(2), 230-253. <https://doi.org/10.1518/001872097778543886>
- Pourreza, M., & Rafiei, D. (2023). DIN-SQL: Decomposed in-context learning of text-to-SQL with self-correction. *Advances in Neural Information Processing Systems*, 36. https://papers.neurips.cc/paper_files/paper/2023/hash/72223cc66f63ca1aa59edaec1b3670e6-Abstract-Conference.html
- Scholak, T., Schucher, N., & Bahdanau, D. (2021). PICARD: Parsing incrementally for constrained autoregressive decoding from language models. *Proceedings of EMNLP 2021*. <https://aclanthology.org/2021.emnlp-main.779/>
- Selbst, A. D., Boyd, D., Friedler, S. A., Venkatasubramanian, S., & Vertesi, J. (2019). Fairness and abstraction in sociotechnical systems. *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 59-68. <https://doi.org/10.1145/3287560.3287598>
- Yu, T., Zhang, R., Yang, K., Yasunaga, M., Wang, D., Li, Z., Ma, J., Li, I., Yao, Q., Roman, S., Zhang, Z., & Radev, D. (2018). Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. *arXiv:1809.08887*. <https://arxiv.org/abs/1809.08887>
- Lodhiya, H. (2026e). *Feedback Control for Adaptive Language Model Routing Under Non-Stationary Workloads*. Journal of Computer Science and Technology Studies. <https://doi.org/10.32996/jcsts.2026.8.8.17>

Declarations

Funding: The author received no specific funding for this work.

Competing Interests: The author declares no competing interests.

Ethics Statement: This study uses publicly available benchmark data and does not involve human subjects, animal subjects, private medical records, or personally collected sensitive data.

Data Availability: The benchmark uses the publicly available BIRD Mini-Dev dataset. The raw dataset is not redistributed with this manuscript because of size; the companion scripts download it from the benchmark source and regenerate the reported statistics.

Code Availability: The companion repository is available at <https://github.com/harshillodhiya/CARE> and includes scripts for downloading the dataset, computing CARE benchmark metrics, generating figures, validating outputs, and running unit tests.

Figures

CARE Runtime Governance Architecture

Controls are applied before and after model generation, not only inside the prompt.

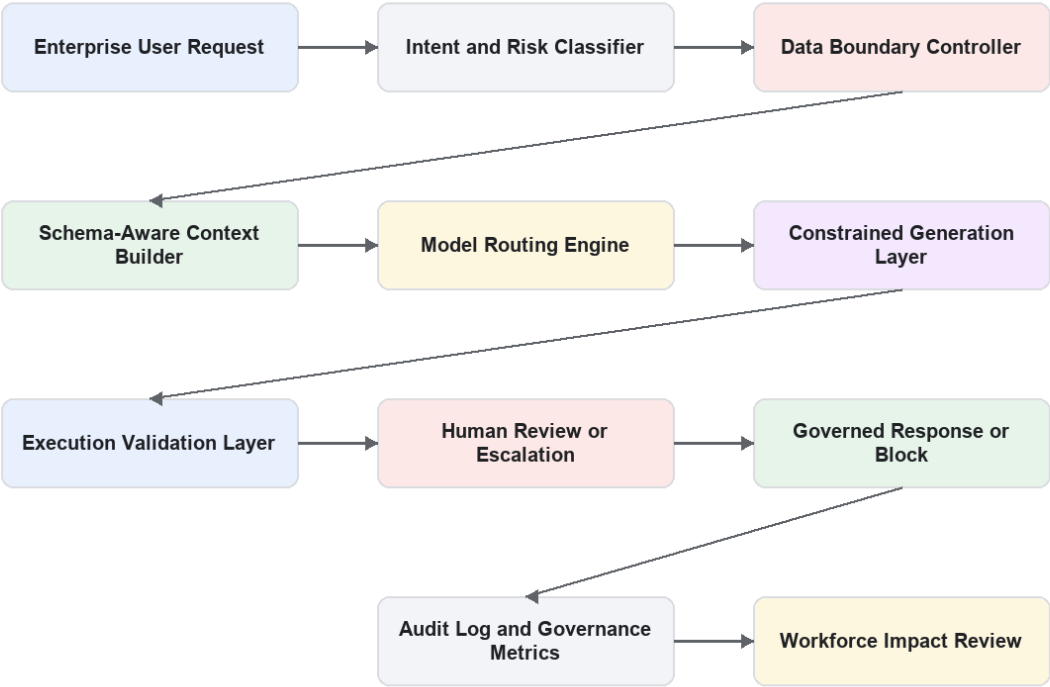


Figure 1

CARE runtime governance architecture.

CARE Runtime Governance Stack



Figure 2

CARE runtime governance stack.

CARE Runtime Control Flow

The model response is treated as a proposal that passes through validation and escalation logic.

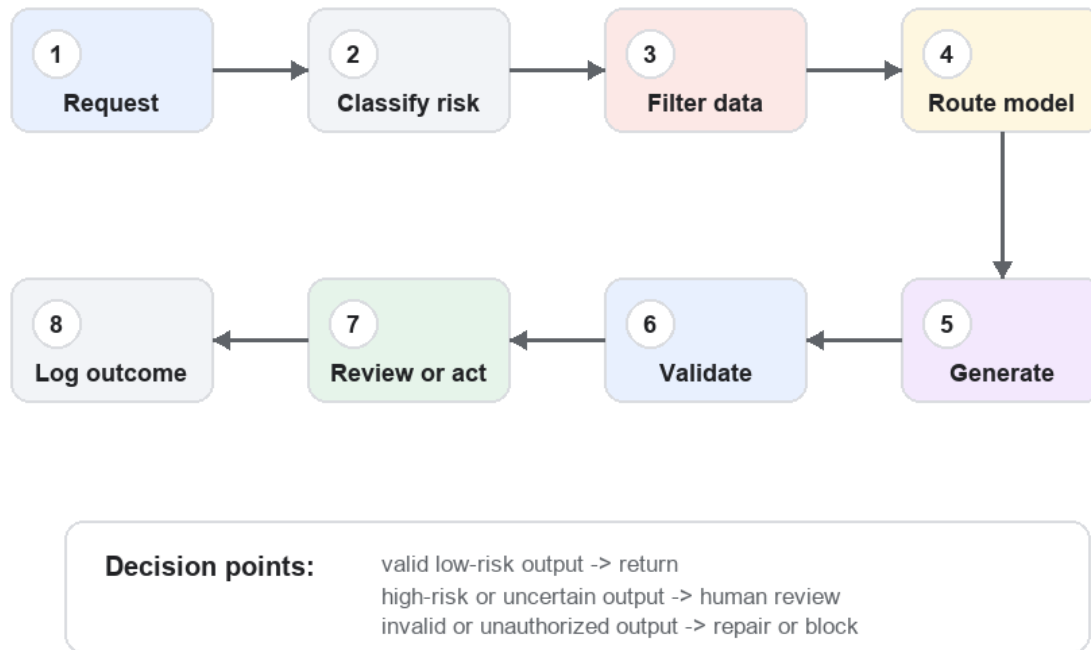


Figure 3

CARE runtime control flow.

CARE Context Reduction on BIRD Mini-Dev

Measured from 500 open benchmark tasks using schemas and gold SQL references.

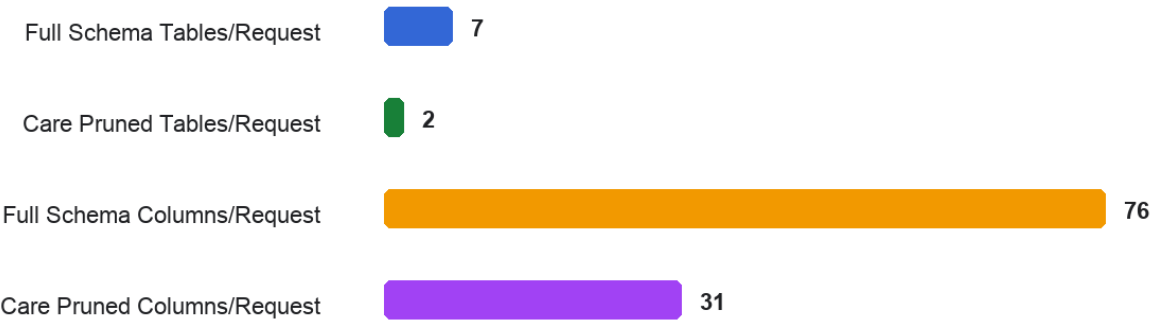


Figure 4

CARE context reduction on BIRD Mini-Dev.

CARE Risk Distribution on BIRD Mini-Dev

Deterministic CARE policy simulation over 500 benchmark tasks.

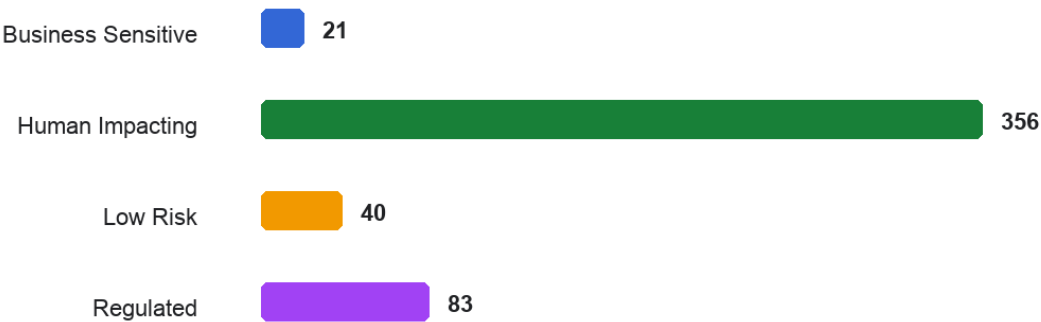


Figure 5

CARE risk distribution on BIRD Mini-Dev.

CARE Routing Distribution on BIRD Mini-Dev

Routes are simulated from explicit risk-tier rules.

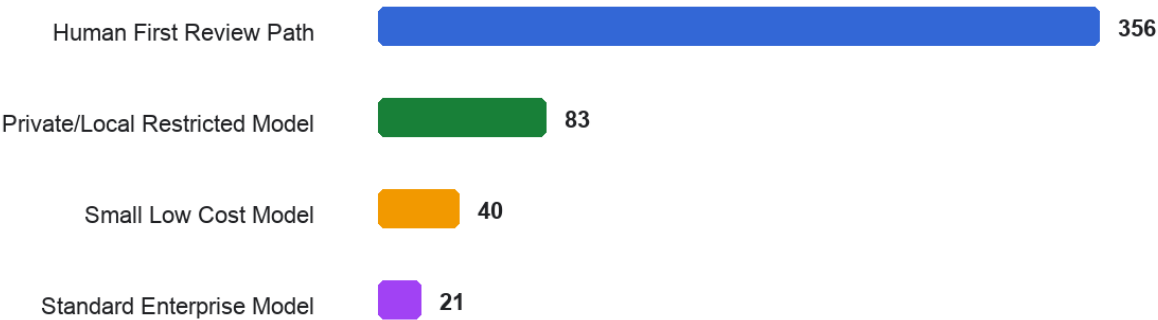


Figure 6

CARE routing distribution on BIRD Mini-Dev.

CARE Governance Outcomes on BIRD Mini-Dev

Outcomes separate direct responses, validation paths, and human review.

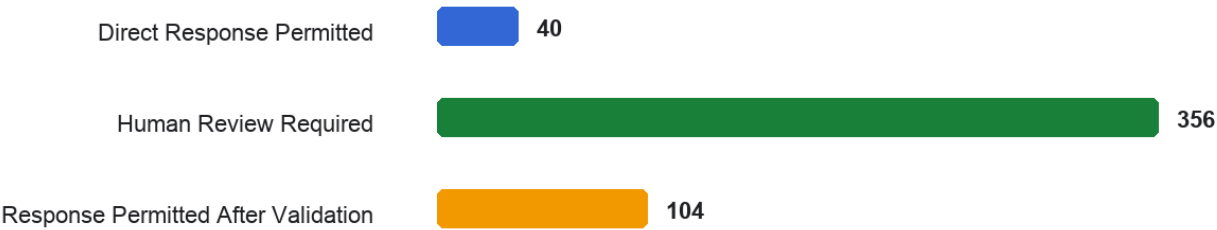


Figure 7

CARE governance outcomes on BIRD Mini-Dev.