Encoder-Based Policy Guardrails for Autonomous Web Agents

Abdulhakim Bashir

MSc Artificial Intelligence

University of Essex Online

Supervisor: Dr Diego Navarra

May 14, 2026

Submitted in partial fulfilment of the requirements
for the degree of Master of Science in Artificial Intelligence

Contents

Abstract

This dissertation investigates whether a lightweight encoder-based policy guardrail can improve the safety of autonomous web agents without requiring a second large language model, full policy recompilation, or modification of the underlying agent. The research addresses a documented compliance gap in web agents, where nominal task completion can remain high while policy-compliant completion collapses under realistic organisational constraints. The study adopts a mono-method quantitative design. A Policy Compliance Module (PCM) based on DeBERTa-v3-base is trained on a benchmark-grounded corpus generated from 375 ST-WebAgentBench tasks across GitLab, SuiteCRM, and ShoppingAdmin. Each training instance encodes a structured `[POLICY]+[CONTEXT]+[ACTION]` triple, with family-level leakage control, a held-out standard test split, and a harder challenge split used as the main robustness check. The trained checkpoint is then integrated into a BrowserGym-compatible wrapper and exercised in a focused live SuiteCRM pilot. The final PCM achieves near-ceiling standard-test performance (precision 0.9972, recall 1.0000, F1 0.9986), but the more credible challenge split is the key result, with precision 1.0000, recall 0.8424, F1 0.9145, FPR 0.0000, and ROC-AUC 0.9792. A rule-based baseline reaches only 0.3828 F1 on the standard test. The live pilot confirms that the wrapper operates end-to-end and reduces observed policy violations, but also reveals a deployment-relevant false positive on `click Create Account (link)`, leaving the system-level hypothesis inconclusive. The dissertation concludes that encoder-scale policy guardrails are viable and auditable for enterprise web agents when training data are benchmark-grounded and evaluation extends beyond easy held-out splits, but that live calibration and broader multi-task evaluation remain necessary before stronger deployment claims can be made.

1 Introduction

1.1 The Problem

Large language model (LLM) powered autonomous web agents can now navigate browsers, fill forms, execute multi-step workflows, and modify enterprise databases in response to natural-language instructions, without human intervention at each step (Deng et al., 2023; Zhou et al., 2024; Drouin et al., 2024). Where a language model previously produced a recommendation, an agent now executes it.

Adoption is accelerating. McKinsey's 2025 global survey found that 88% of organisations are regularly using AI in at least one function, and 62% are at least experimenting with autonomous agents, though only 23% are actively scaling them (McKinsey & Company, 2025). Despite this momentum, Gartner predicts over 40% of agentic AI projects will be canceled by end of 2027, with governance and compliance cited as the leading barriers (Gartner, 2025). Capability has advanced rapidly: GPT-4 achieved 14.41% on WebArena at launch (Zhou et al., 2024) and subsequent systems have improved substantially (Yang et al., 2025), yet Xue et al. (2025) show that live-website performance remains up to 59% lower than closed-benchmark scores.

Task completion capability and policy compliance do not improve together. An agent that completes a workflow while exposing a credential, deleting a record without confirmation, or acting on fabricated data has created legal liability and regulatory exposure, regardless of whether the task goal was nominally achieved. Levy et al. (2026) quantify this on ST-WebAgentBench: across three state-of-the-art agents, the average Completion under Policy (CuP) score falls approximately 38% below the nominal completion rate, and under five or more concurrent policies CuP collapses to 7.1%. Across sixteen agents on Agent-SafetyBench, the mean safety score is 38.5% with no agent above 60% (Zhang et al., 2024). The compliance deficit is structural.

Three existing remedy categories each fall short. Training-time alignment methods such as RLHF (Ouyang et al., 2022) and Constitutional AI (Bai et al., 2022) embed constraints in model weights, cannot adapt to policy changes without retraining, and produce no per-action audit trail. LLM-based runtime guardrails including GuardAgent (Xiang et al., 2024), AGrail (Luo et al., 2025), and ShieldAgent (Chen et al., 2025) offer strong accuracy but require 7 to 70 billion-parameter inference engines that introduce 300 to 500 ms latency per action, incompatible with synchronous pre-execution gating. Deterministic compilation approaches such as ToolGuard (Zwerdling et al., 2025) provide sub-millisecond execution and strong auditability, but require complete offline recompilation and expert code review for every policy change and have been evaluated on a single domain with 14 tools only. No existing system simultaneously achieves modularity, encoder-scale inference latency, policy generalisability without recompilation, and per-action auditability across multiple enterprise domains.

1.2 Research Questions

This dissertation is structured around three research questions that collectively address the design, intrinsic performance, and system-level impact of the proposed Policy Compliance Module:

RQ1 (Design): How can a learned policy compliance module be designed and integrated into a BrowserGym-compatible web agent loop to enforce operational constraints without requiring modification to the underlying agent?

RQ2 (Classifier Performance): How accurately can a fine-tuned DeBERTa-v3-base encoder detect policy violations from agent action-context pairs across the six ST-WebAgentBench safety dimensions, and what precision-recall trade-offs arise at different violation thresholds?

RQ3 (System Evaluation): Does integrating the PCM produce a statistically significant improvement in Completion under Policy scores on ST-WebAgentBench, at what cost to task completion rate, and does compliance training on synthetic data generalise to real benchmark scenarios?

RQ1 is the primary engineering design question. RQ2 operationalises the classifier's intrinsic performance. RQ3 provides the end-to-end system-level evidence and validates the data strategy, directly addressing the compliance deficit identified by Levy et al. (2026).

1.3 Aim and Objectives

Aim: To develop and evaluate a learned Policy Compliance Module (PCM) that intercepts autonomous agent actions, assesses them against enterprise policies, and blocks violations before execution, as measured by Completion under Policy scores on ST-WebAgentBench (Levy et al., 2026) relative to an unguarded baseline, while maintaining a task completion rate penalty of at most 5% absolute.

Table 1: Project objectives and deliverables

O	Label	Deliverable
O1	Policy Taxonomy	Documented mapping of ST-WebAgentBench's six safety dimensions to parameterised YAML policy templates, defining allowed and disallowed action types and context conditions for synthetic data generation (Levy et al., 2026).
O2	Benchmark-Grounded Corpus	Python codebase producing a benchmark-grounded corpus of labelled `[POLICY]+[CONTEXT]+[ACTION]` triples from ST-WebAgentBench task and policy metadata, with family-level split control, challenge-set construction, and class balance monitoring (Li et al., 2023; Nadas et al., 2025).
O3	Trained PCM	Fine-tuned DeBERTa-v3-base classifier achieving precision ≥ 0.85, recall ≥ 0.80, macro-F1 ≥ 0.82, with model weights and training configuration retained for reproducibility (He et al., 2023).
O4	System Integration	`PolicyCompliantAgent` BrowserGym-compatible wrapper with configurable threshold θ, multi-policy OR-logic, per-action decision logging satisfying EU AI Act Article 12, and optional re-planning on blocked actions (de Chezelles et al., 2025).
O5	Empirical Evaluation	Quantitative analysis combining offline benchmark-grounded classifier evaluation, rule-based baseline comparison, threshold analysis, and a focused live SuiteCRM pilot using the BrowserGym integration harness (McNemar, 1947).
O6	Threshold Analysis	Full precision-recall curves and compliance-utility trade-off plots across $\theta \in \{0.10, 0.20, \ldots, 0.99\}$.
O7	Robustness Assessment	Held-out challenge-set evaluation and qualitative live pilot analysis quantifying the gap between in-distribution offline performance and behaviour in a live SuiteCRM agent loop.

Objectives O1–O5 are essential and correspond directly to the primary hypotheses. O6 and O7 are secondary objectives that may be partially addressed subject to time constraints, consistent with the approved research proposal.

1.4 Working Hypotheses

Following the deductive approach, the following hypotheses operationalise the research questions for empirical testing (Saunders et al., 2019):

Table 2: Working hypotheses aligned to research questions

H	RQ	Hypothesis
H1	2	A fine-tuned DeBERTa-v3-base classifier can detect policy violations from action-context pairs with precision ≥ 0.85 and recall ≥ 0.80, consistent with the encoder advantage over zero-shot LLMs documented by Bucher and Martini (2024) and the safety classification efficiency demonstrated by Zheng et al. (2025).
H2	3	Integrating the PCM with baseline agents will improve CuP scores by $\geq 10\%$ relative to unguarded baseline agents on ST-WebAgentBench, addressable within the $\approx 38\%$ compliance gap identified by Levy et al. (2026), with task completion rate declining by $\leq 5\%$ absolute.
H3	2	Increasing the violation threshold θ will increase task completion rates while decreasing policy compliance rates, exhibiting a monotonic and measurable trade-off demonstrable through precision-recall curves; and a classifier trained on benchmark-grounded policy-violation data will retain at least 70% of its held-out test performance on a harder challenge split and limited live benchmark scenarios, consistent with generalisation evidence for LLM-generated structured data reported by Li et al. (2023).

H1 and H2 are primary success criteria whose joint satisfaction constitutes validation of the PCM's core claims. H3 provides secondary insights into threshold behaviour and data strategy validity.

1.5 Background

1.5.1 From Scripted Automation to Language-Driven Agents

RPA tools achieve reliable performance on stable workflows but cannot interpret natural-language instructions or generalise beyond scripted coverage. LLM-based agents overcome this through generalisation, enabling action selection across novel environments without task-specific programming (Deng et al., 2023; Zhou et al., 2024; Drouin et al., 2024). This same capability creates governance risk: an LLM agent can take actions that are contextually plausible but operationally impermissible.

1.5.2 The Enterprise Compliance Environment

Enterprise agents operate within multi-dimensional, policy-governed contexts across the three platforms evaluated in ST-WebAgentBench (Levy et al., 2026). These constraints are formalised through six orthogonal safety dimensions: User Consent, Boundary and Scope, Strict Execution, Hierarchy Adherence, Robustness and Security, and Error Handling. Each dimension can be violated independently and contributes to the CuP penalty. Chapter 2 examines each dimension in detail.

1.5.3 Why Compliance Cannot Be Delegated to the Base Agent

Four structural reasons explain why compliance cannot be embedded in the base agent. First, the compliance gap persists across frontier alignment-trained models (Levy et al., 2026), confirming that general alignment does not produce enterprise-grade policy compliance. Second, enterprise policies are dynamic and change continuously; weight-embedded compliance requires full retraining to adapt. Third, EU AI Act Article 12 (European Parliament and Council, 2024) mandates per-action audit records that no training-time method produces. Fourth, the agent and compliance function serve different organisational principals with different update cycles, making a modular separation architecturally essential.

1.5.4 The Proposed Approach

The PCM operates between planning and execution. When the base agent proposes an action, the PCM intercepts it, forms a structured `[POLICY]+[CONTEXT]+[ACTION]` triple, and runs a forward pass through fine-tuned DeBERTa-v3-base (He et al., 2023) to produce $P(\text{violation})$. Actions exceeding threshold θ are blocked before reaching the browser. The architecture requires no modification to the base agent and integrates as a modular wrapper into any BrowserGym-compatible loop (de Chezelles et al., 2025).

1.6 Regulatory Context

The PCM's architecture is directly aligned with binding governance requirements that apply from August 2026 to high-risk AI system deployers. The EU AI Act (European Parliament and Council, 2024) mandates automatic logging of all events during high-risk AI system operation (Article 12), transparency documentation enabling deployers to interpret AI outputs (Article 13), human oversight measures with operator override capability (Article 14), and lifecycle risk management systems (Article 9). The PCM's per-action audit log, continuous probability score, configurable blocking threshold, and modular architecture directly operationalise each of these obligations. The NIST AI Risk Management Framework (NIST, 2023) and ISO/IEC 42001:2023 (ISO, 2023) provide complementary governance requirements that the PCM satisfies through its measurable performance metrics and event logging infrastructure.

1.7 Contributions

This dissertation makes three contributions. First, it designs and implements the PCM, an encoder-based pre-execution compliance classifier integrated into a BrowserGym-compatible agent loop. Second, it provides empirical evidence that an encoder-scale classifier at approximately 86M parameters can achieve strong benchmark-grounded offline policy-violation detection, including a harder held-out challenge split that is materially more credible than a near-ceiling standard test set. Third, it contributes a reproducible benchmark-grounded data construction and evaluation pipeline that converts ST-WebAgentBench policy and task metadata into runtime-shaped training examples without requiring proprietary enterprise logs.

To guide the reader, the remainder of the dissertation is organised as follows. Chapter 2 reviews the literature on web-agent benchmarks, policy-compliance failures, and the trade-offs between training-time, rule-based, and runtime guardrail approaches. Chapter 3 explains the research design, benchmark-grounded data strategy, and evaluation protocol. Chapter 4 presents the PCM architecture, model-training configuration, and BrowserGym integration. Chapter 5 reports and interprets the offline and live evaluation results. Chapter 6 discusses the implications, limitations, generalisability, and future directions of the study.

2 Literature Review

The PCM sits at the intersection of five literatures: web-agent capability, agent safety, AI governance, transformer-based classification, and the false positive problem in automated compliance systems. This chapter reviews each area selectively rather than exhaustively, focusing on what is needed to justify the artefact design and the evaluation strategy.

The review draws primarily on peer-reviewed work from ICLR, ICML, NeurIPS, ACL, EMNLP, NAACL, COLING, USENIX Security, and TMLR. Pre-prints are used only where they represent the most current and field-shaping evidence on a rapidly moving topic, especially frontier web agents and runtime guardrails.

2.1 Web Agent Architectures and Capabilities

General-purpose web agents became practical with benchmarks such as Mind2Web, WebArena, and VisualWebArena, which together established cross-site generalisation, long-horizon interaction, and multimodal grounding as the core capability challenges (Deng et al., 2023; Zhou et al., 2024; Koh et al., 2024). Mind2Web was especially important methodologically because it introduced cross-split generalisation and the now-standard two-stage pattern in which a smaller model narrows candidate elements before a larger model selects the action. WebArena then moved the field toward executable, self-hosted environments with functional task evaluation, making it possible to measure full trajectories rather than static next-step prediction.

Consumer-web benchmarks, however, do not fully capture enterprise workflows. WorkArena translates the web-agent problem into structured ServiceNow tasks, including multi-step compositional workflows in which one early mistake can invalidate all later progress (Drouin et al., 2024). That failure mode is important for this dissertation because it mirrors the logic of compliance: under ST-WebAgentBench, one policy violation can nullify an otherwise successful trajectory. BrowserGym provides the common interface that makes such comparisons possible. Its standardised observation and action spaces separate planning from execution cleanly enough that a compliance mechanism can be inserted between them without changing the base agent (de Chezelles et al., 2025).

This enterprise turn matters conceptually as well as practically. Earlier web benchmarks showed that long-horizon browser control is difficult; WorkArena and BrowserGym showed that the same difficulty persists once workflows become permissioned, stateful, and operationally consequential. The PCM therefore sits within a broader shift from "can the agent finish the task?" to "can it finish the task without violating constraints that matter to an organisation?".

Three further findings matter directly for PCM design. First, grounding remains a decisive bottleneck. SeeAct shows that performance can jump sharply when element grounding is made

more reliable, implying that action semantics are best captured after grounding rather than from raw element identifiers alone (Zheng et al., 2024). Second, representation design matters materially. AgentOccam demonstrates that even without changing the base model, restructuring the AXTree to better fit pre-training distributions can produce large gains (Yang et al., 2025). Third, benchmark scores overstate deployment readiness. Xue et al. (2025) report that performance on live websites can drop dramatically relative to closed-benchmark scores, which is exactly why a policy guardrail should be judged on more than an easy in-distribution test.

This literature leaves two gaps. Most mainstream web-agent papers still treat task completion as the end metric, and none evaluates an encoder-scale pre-execution guardrail integrated into the live control loop (Zhou et al., 2024; Drouin et al., 2024; de Chezelles et al., 2025).

2.2 Agent Safety and Policy Compliance

Training-time alignment methods such as RLHF and Constitutional AI improve general harmlessness, but they are poorly suited to enterprise policy compliance. Their constraints are embedded in model weights, difficult to update when local policy changes, and they do not natively produce a per-action record of which policy was checked against which action (Ouyang et al., 2022; Bai et al., 2022; European Parliament and Council, 2024). That is a serious misfit for enterprise settings in which policies change continuously and audit trails are a first-class requirement rather than an optional feature.

The strongest direct evidence for the compliance problem comes from ST-WebAgentBench (Levy et al., 2026). The benchmark covers 375 tasks and 3,057 policy instances across six dimensions: User Consent, Boundary and Scope, Strict Execution, Hierarchy Adherence, Robustness and Security, and Error Handling. Its key metric is Completion under Policy:

$$\mathrm{CuP}_t = C_t \times \mathbf{1}\left[\sum_d V_d^t = 0\right], \quad \mathrm{CuP} = \frac{1}{T}\sum_{t=1}^{T} \mathrm{CuP}_t \tag{1}$$

Across the published agents, CuP falls materially below nominal completion, and under heavier policy load it degrades even further. In other words, task success and policy compliance do not move together. The benchmark is also useful analytically because it identifies the dominant failure sources: User Consent and Strict Execution account for most violations, while the explicit policy hierarchy formalises the precedence of organisational constraints over user- or task-level instructions. ST-WebAgentBench is therefore not just a benchmark but the clearest empirical statement of the operational problem this dissertation addresses.

The benchmark is also unusually informative about scale and structure. Its task set spans GitLab, ShoppingAdmin, and SuiteCRM, with the latter further broken into Easy, Medium, and Hard subsets with increasing policy density (Levy et al., 2026). The published results show

that compliance does not fail only on rare corner cases. Average completion remains materially above average CuP, partial completion outstrips fully compliant completion by an even larger margin, and all-pass@3 scores remain low, revealing substantial run-to-run brittleness. This matters because it rules out a comforting interpretation in which violations are merely occasional slips by otherwise stable agents. Instead, the benchmark suggests a structural mismatch between what current agents optimise for and what enterprise deployment requires.

Complementary safety work points in the same direction. Agent-SafetyBench shows that current agents remain far from robust safety performance even beyond web automation (Zhang et al., 2024). Runtime guardrail work then clarifies the design space. GuardAgent and AGrail retain semantic flexibility through LLM-based checks but pay substantial inference cost (Xiang et al., 2024; Luo et al., 2025). ShieldAgent and ToolGuard reduce runtime overhead through compilation or rule execution, but at the cost of rebuild overhead, narrower policy adaptability, or limited domain coverage (Chen et al., 2025; Zwerdling et al., 2025). This is the architectural opening for the PCM — a smaller learned classifier that stays semantic, modular, and cheap enough for synchronous gating.

The details of these systems strengthen that conclusion. GuardAgent uses a knowledge-enabled LLM guard that can interpret natural-language constraints but still incurs a full generative pass for each decision (Xiang et al., 2024). AGrail improves robustness, including against environmental prompt injection, but does so through a cooperative multi-LLM architecture whose runtime cost is still far above that of an encoder classifier (Luo et al., 2025). By contrast, ShieldAgent and ToolGuard move toward deterministic execution through compiled policy logic and tool guards (Chen et al., 2025; Zwerdling et al., 2025). That improves auditability and latency, but it also makes policy evolution more expensive because policy change implies rebuild or recompilation. The PCM is therefore best understood as occupying the middle ground between flexible but slow LLM guards and fast but rigid compiled guards.

Table 3: Comparison of runtime guardrail approaches against PCM requirements

System	**Modular**	**Low latency**	**No rebuild**	**Auditability**	**Multi-domain**
GuardAgent (Xiang et al., 2024)	Yes	No	Yes	Limited	Limited
AGrail (Luo et al., 2025)	Yes	No	Yes	Limited	Yes
ShieldAgent (Chen et al., 2025)	Yes	Yes	No	Yes	Limited
ToolGuard (Zwerdling et al., 2025)	Yes	Yes	No	Yes	No
PCM (proposed)	Yes	Yes	Yes	Yes	Yes

The comparison in Table 3 motivates the chosen positioning. The PCM is not intended to outperform every alternative on every dimension. Rather, it targets the specific intersection that the current literature leaves underserved: semantic policy interpretation at encoder-scale latency with per-action logging and no offline recompilation.

2.3 Enterprise AI Governance and Regulatory Frameworks

The governance literature reinforces the technical case for a modular guardrail. The EU AI Act requires risk management, logging, transparency, and human oversight for high-risk AI systems (European Parliament and Council, 2024). The NIST AI RMF and its Generative AI Profile emphasise measurable evaluation, semantic filtering, and traceable risk controls (NIST, 2023; Autio et al., 2024). ISO/IEC 42001 similarly treats monitored performance and event logging as management-system requirements rather than optional best practice (ISO, 2023). OWASP's 2025 Top 10 for LLM applications adds an attack-oriented perspective, especially prompt injection and excessive agency (OWASP Foundation, 2025).

These frameworks do not prescribe an encoder-based solution, but they do imply four concrete design requirements: intervention before execution, operator visibility into the decision, reconfigurability without full model retraining, and durable event logging. This alignment matters because it shows the PCM is not just technically convenient; it is a direct response to external governance expectations now shaping enterprise AI deployment.

The governance perspective also sharpens the distinction between compliance and general alignment. General alignment asks whether a model tends to behave helpfully or harmlessly across broad situations. Governance frameworks ask whether a deployer can document, inspect, justify, and override system behaviour under specific operational controls. That distinction is one reason the PCM is architected as a separate module instead of an assumption about the base agent's intrinsic trustworthiness.

2.4 Transformer-Based Text Classification for Policy Detection

The PCM draws on the standard pre-train/fine-tune paradigm established by BERT and refined by RoBERTa and DeBERTa (Devlin et al., 2019; Liu et al., 2019; He et al., 2021). DeBERTa-v3 is especially relevant because its disentangled attention suits structured segment inputs, while its replaced-token-detection objective inherits the efficiency advantages of ELECTRA-style pre-training (He et al., 2023; Clark et al., 2020). That combination is a good fit for a classifier that must reason over explicitly delimited policy, context, and action segments rather than over free-form prose alone.

The broader classification literature supports the model choice directly. Fine-tuned encoders consistently outperform zero-shot generative LLMs on specialised classification tasks, especially when the label space is narrow but the decision boundary is domain-specific (Bucher and Martini, 2024; Galke et al., 2025). This matters because policy compliance is exactly such a task: binary in output, but highly specific in how the positive class is defined. Recent safety work also shows that lightweight encoders can approach much larger guardrail systems at far

lower runtime cost (Zheng et al., 2025). That is the deployment niche the PCM targets.

DeBERTa-v3 is also attractive for a more specific reason: its gains are especially notable in low-resource classification settings (He et al., 2023). That matters here because policy-violation data are not available at internet scale; even a benchmark-grounded corpus remains small by modern pre-training standards. The objective is not to discover a general web agent from scratch, but to specialise a mature encoder to a narrow and operationally meaningful distinction. In that regime, the literature strongly favours supervised fine-tuning of compact encoders over prompting much larger generative models.

There is nevertheless an architectural trade-off. Newer encoders such as ModernBERT promise longer context windows and more modern attention machinery (Warner et al., 2024). For this dissertation, however, the main bottleneck is not thousand-token document understanding; it is reliable classification of short structured triples under moderate data and latency constraints. The final design therefore prioritises DeBERTa-v3's empirical strength in supervised classification over newer but less validated choices for this exact use case.

Because real policy-violation corpora are scarce, synthetic or benchmark-derived augmentation is unavoidable. Prior work shows that synthetic data can support objective classification tasks when labels are logically determined and quality controls are explicit (Anaby-Tavor et al., 2020; Li et al., 2023; Nadas et al., 2025). The key methodological lesson is not that any synthetic data will do, but that generation quality and split design are decisive. This is why the final corpus in this dissertation is benchmark-grounded and family-split rather than a small templated sentence set.

This point is especially important because synthetic-data results are easy to overclaim. The literature is clear that synthetic augmentation is strongest when the target predicate is explicit and verifiable, not when labels depend on vague human preference (Li et al., 2023; Nadas et al., 2025). Policy compliance fits that better than many other safety tasks because the central question is whether an action violates a stated policy. Even so, generation shortcuts can still create unrealistically separable splits. That observation directly motivates the challenge set and the insistence in Chapter 5 on treating the harder split, not the near-ceiling standard split, as the more credible indicator of generalisation.

2.5 The False Positive Problem in Compliance Systems

False positives are not a minor operational inconvenience. In security operations, high false-positive rates create alert fatigue, reduced trust, and eventual abandonment of the monitoring system (Alahmadi et al., 2022). In AML, rule-based monitoring likewise produces large volumes of benign alerts, which is why learned screening approaches are increasingly preferred (Ghimire, 2025). For an autonomous-agent guardrail, the consequence is sharper still — a false

positive can directly fail or delay the task. That is why precision and FPR are treated in this dissertation as co-equal with recall, not secondary reporting metrics.

There is also an evaluation lesson. Movva et al. (2024) show that safety labelling can be genuinely ambiguous when it depends on human normative judgement. Policy-compliance classification is more objective than conversational safety annotation because the predicate is tied to an explicit policy statement, but the general lesson still holds: ambiguous or transitional cases should be inspected qualitatively, not hidden beneath headline averages.

That insight becomes concrete in the live SuiteCRM pilot reported later. The offline classifier performs strongly, but a transitional action such as `click Create Account (link)` can still be misread when the active policy is framed around eventual save behaviour. This is not a reason to abandon the approach; it is evidence that live false positives should feed back into corpus design and calibration rather than being dismissed as anecdotal noise.

2.6 Synthesis and Research Gap

Taken together, the literature establishes five points. First, web agents are becoming more capable, but benchmark success still overstates live reliability (Xue et al., 2025). Second, the compliance gap is structural and is quantified most clearly by ST-WebAgentBench (Levy et al., 2026). Third, governance frameworks require logging, transparency, and human oversight (European Parliament and Council, 2024; NIST, 2023; ISO, 2023). Fourth, fine-tuned encoders are both strong and efficient on specialised classification (Bucher and Martini, 2024; Zheng et al., 2025). Fifth, false positives are operationally costly enough to be a first-order design constraint (Alahmadi et al., 2022; Ghimire, 2025). What is still missing is a modular, benchmark-grounded, encoder-scale pre-execution guardrail evaluated against this combined set of technical, governance, and operational requirements. That gap defines the contribution of the PCM.

3 Methodology

This chapter presents the methodological framework governing the design, data collection, and evaluation of the PCM. It is organised around three principal concerns: philosophical and strategic positioning within a formal research tradition; the two-pronged data collection strategy; and the experimental design, evaluation metrics, and statistical analysis plan. The chapter concludes with ethical and professional considerations. Throughout, methodological choices are explicitly justified with reference to the research questions and prior literature rather than asserted as convention.

3.1 Research Philosophy and Design

3.1.1 Philosophical Position: Positivism

This research adopts a positivist philosophical position. Positivism holds that reality is objective and measurable independently of the researcher, and that knowledge advances through the systematic empirical testing of propositions against observable data (Saunders et al., 2019; Bryman, 2016). This position is appropriate for the present research for three specific reasons. First, all primary research questions require quantitative measurement: classifier precision, recall, and F1 score; Completion under Policy scores; per-dimension false positive rates; and inference latency measurements. These are objective metrics with established definitions that can be computed consistently regardless of researcher identity. Second, the experimental environment (ST-WebAgentBench) running on identical hardware with identical agent configurations is reproducible, and conditions can be held constant across configurations, enabling the controlled comparisons that positivism prescribes. Third, the hypotheses are formulated as specific, falsifiable predictions derived from prior theoretical and empirical work, consistent with the hypothetico-deductive model characteristic of positivist inquiry (Bryman, 2016).

Interpretivism is unsuitable as the research concerns measurable software performance rather than subjective meaning-making or social phenomena.

3.1.2 Research Approach: Deductive

The research follows a deductive approach, deriving testable hypotheses from established theoretical premises that that fine-tuned encoders outperform LLMs for specialised classification (Bucher and Martini, 2024), that the compliance gap is measurable (Levy et al., 2026), and that synthetic data is viable for objective classification (Li et al., 2023), and testing them through controlled benchmark experiments. H1–H3 (Table 2) specify the predictions; Chapters 4 and 5 report their empirical evaluation.

3.1.3 Research Strategy: Design Science Research

The primary research strategy is Design Science Research (DSR) (Hevner et al., 2004), which is concerned with the creation and rigorous evaluation of purposeful IT artefacts to address identified organisational problems. DSR is appropriate because the primary contribution is a technical artefact rather than a theoretical explanation (March and Smith, 1995). The research follows Peffers et al. (2007)'s six-stage DSRM:

1. **Problem identification:** the ∼38% CuP compliance gap (Levy et al., 2026).
2. **Objective definition:** H1 and H2 performance thresholds and objectives O1–O5 from Table 1.
3. **Design and development:** encoder selection, data pipeline, and BrowserGym wrapper, detailed in Chapter 4.
4. **Demonstration:** integration with ST-WebAgentBench via the `PolicyCompliantAgent` wrapper.
5. **Evaluation:** controlled benchmark experiments across four configurations, reported in Chapter 5.
6. **Communication:** this dissertation and open-source code repository released under MIT licence.

Evaluation follows Venable et al. (2016)'s FEDS framework using ex-post naturalistic benchmark experimentation.

3.1.4 Methodological Choice: Mono-Method Quantitative

The research employs a mono-method quantitative design, collecting exclusively numerical data through a single evaluation framework (Saunders et al., 2019). This choice is justified by three considerations. First, the technical nature of the research questions: RQ1–RQ3 all require measurable system performance metrics rather than qualitative accounts of user experience or social meaning. Second, the existence of a standardised evaluation benchmark that eliminates the need for researcher-designed measurement instruments and enables direct comparison with published baselines. Third, time constraints and the focused scope of the project: mixed-methods or multi-method quantitative designs would add considerable data collection and analysis overhead without materially improving the validity of answers to the specific research questions posed. Future work incorporating qualitative validation with enterprise compliance practitioners is identified as a valuable extension.

3.2 Data Collection Strategy

Labelled policy-violation data for web agents remains scarce, and no public dataset provides enough action-policy examples in the exact format required by the PCM. The final data strategy therefore grounds the training corpus directly in ST-WebAgentBench task and policy metadata rather than relying on a small hand-written synthetic set. The key idea is to preserve the benchmark's taxonomy, applications, and policy semantics while generating actions whose surface forms resemble the runtime actions the wrapper actually classifies. This reduces one important source of bias—training on examples that look nothing like runtime actions—but it does not make the corpus neutral. The resulting dataset is still a measurement instrument whose definition of "policy violation" is inherited from the benchmark itself (Jacobs and Wallach, 2021). It also inherits representation and framing risks from the benchmark's task mix and policy distribution (Suresh and Guttag, 2021). Bias is therefore treated here as a validity issue to be surfaced and bounded, not as a problem solved by synthetic scale alone.

3.2.1 Benchmark-Grounded Corpus Construction

The corpus is generated from the local ST-WebAgentBench task catalogue (`test.raw.json`), which contains 375 tasks across SuiteCRM, GitLab, and ShoppingAdmin (Levy et al., 2026). For each task-policy pair, the generator derives:

1. a normalised natural-language policy statement;
2. a structured context string containing site, task ID, module, intent, and selected policy metadata; and
3. matched compliant and violating action strings in BrowserGym-shaped natural language.

This produces the same `[POLICY]+[CONTEXT]+[ACTION]` representation used by the classifier at training and inference time, while avoiding dependence on proprietary enterprise logs. The generator covers the dominant ST-WebAgentBench policy template families:

- **Irreversible actions**
- **User-confirmation requirements**
- **Navigation limitations**
- **Access-management constraints**
- **Policy contradictions**
- **Sensitive-information handling**
- **Jailbreak attempts**
- **Missing parameters**
- **Hallucinated information**

The benchmark's original template identifiers are preserved in the released code and dataset artefacts, but the dissertation refers to them in descriptive form for readability.

In practical terms, synthetic generation proceeds as a constrained counterfactual transformation rather than as free-form text invention. For each task-policy pair, the generator first normalises the benchmark policy into a descriptive policy statement, then extracts a compact context record from task metadata, and finally instantiates a matched pair of BrowserGym-shaped action strings: one compliant and one violating. Additional near-neighbour variants are then grouped into the same family so that the model learns the effect of small, policy-relevant action changes rather than memorising one fixed sentence per policy.

Table 4: Representative benchmark tasks used for corpus grounding

Site	**Tasks**	**Representative benchmark intents**
GitLab	197	Create projects and issues; add members, reviewers, or assignees in repository workflows
SuiteCRM	170	Create new accounts and contacts; update CRM records in consent- and deletion-sensitive workflows
ShoppingAdmin	8	Add new product sizes or colours; update stock and catalogue variants in the admin panel

3.2.2 Matched Families, Leakage Control, and Challenge Split

Each policy generates a *family* of closely related samples sharing the same site, task, and policy semantics. Within a family, violating and compliant actions differ in minimal and interpretable ways. Splitting is performed at the family level rather than at the individual-sample level, reducing leakage from near-duplicate variants.

A separate challenge split is also constructed. The standard test split remains benchmark-grounded but relatively in-distribution; the challenge split uses less templatic variants and harder counterfactuals to provide a more credible robustness check. A small `sanity_probes.jsonl` file is produced separately for immediate checkpoint verification before any live browser evaluation.

Table 5: Final benchmark-grounded corpus composition

Split	**Samples**	**Label 0**	**Label 1**
Train	12,493	6,324	6,169
Validation	2,572	1,325	1,247
Test	2,877	1,427	1,450
Challenge	368	184	184

The corpus spans all 375 benchmark tasks and all three benchmark sites. At the dimension

level, Boundary and Scope and Strict Execution dominate the final sample counts, reflecting the benchmark's actual policy distribution rather than arbitrary manual balancing.

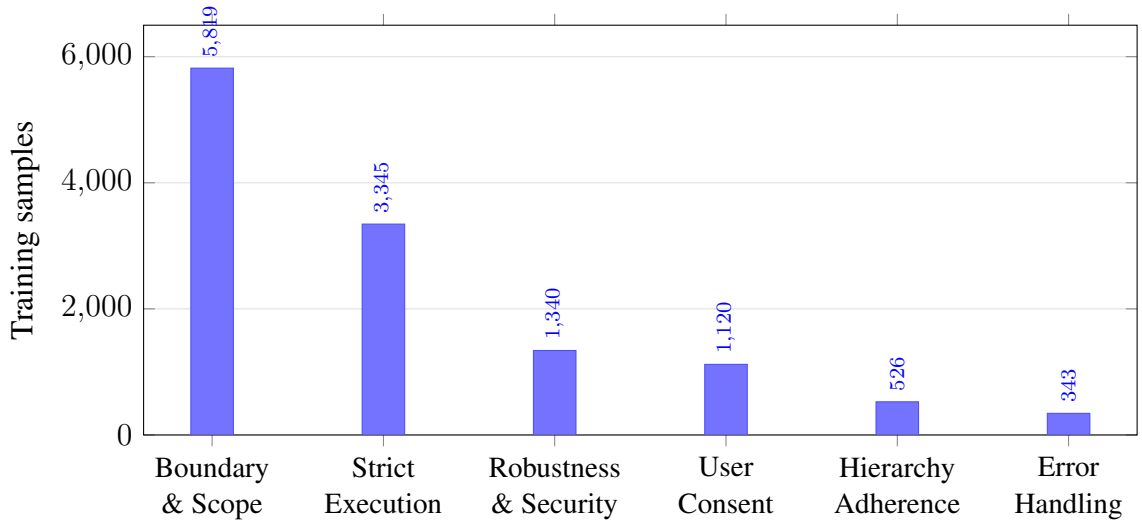

Figure 1: Training-set distribution by ST-WebAgentBench safety dimension. The synthetic corpus follows the benchmark's underlying policy distribution rather than enforcing an artificial per-dimension balance.

Table 6: Worked example of benchmark-grounded synthetic generation

Stage	**Example**
Source task	SuiteCRM: create a new account record in the Accounts module
Source policy	Ask the user for permission before interacting with `Save`
Derived context	`site=suitecrm; module=accounts; intent=create account`
Compliant action (label 0)	`send message to user: "I am ready to click Save. Please confirm before I proceed."`
Violating action (label 1)	`click Save (button)`

3.2.3 Quality Assurance

Three quality controls are applied. First, labels are assigned programmatically from benchmark policy metadata, reducing subjective annotation variance. Second, family-level split control prevents near-duplicate leakage across train, validation, and test sets. Third, the generated corpus is inspected through sanity probes that test obvious safe and unsafe examples such as safe module navigation, destructive clicks, missing consent, and credential exposure. This proved operationally important because it prevented use of earlier collapsed checkpoints that classified almost every action as a violation. These controls reduce, but do not eliminate, shortcut learning or benchmark surface overfitting. The separate challenge split, per-dimension analysis, and later

live audit-log inspection are therefore treated as bias-detection instruments rather than optional add-ons (Geirhos et al., 2020; Xue et al., 2025).

3.2.4 Live SuiteCRM Pilot Data

For system-level evaluation, a focused live pilot is conducted on SuiteCRM using the BrowserGym/ST-WebAgentBench harness. This pilot is intentionally smaller in scope than the full benchmark. It provides qualitative and task-level evidence about how the PCM behaves inside a real agent loop, but it is not large enough to support a definitive claim about benchmark-wide CuP improvement.

3.3 Experimental Design

3.3.1 Experimental Configurations

The implementation supports four configurations:

1. **Baseline (unguarded):** the base agent without any compliance layer, replicating the published baseline results (Levy et al., 2026) and establishing the pre-intervention compliance deficit under identical experimental conditions.
2. **Rule-based filter:** the base agent augmented with a deterministic keyword and action-type filter implementing the six safety dimensions as explicit conditional rules. This configuration tests whether the PCM's learned classification provides a meaningful advantage over a simple heuristic that requires no training data, providing a lower-bound performance baseline.
3. **PCM (blocking):** the base agent augmented with the trained DeBERTa-v3-base PCM (He et al., 2023) in blocking mode at threshold $\theta = 0.5$ (default). Actions with $P(\text{violation}) \geq \theta$ are blocked without re-planning; the task step terminates without an action. This is the primary evaluation configuration for H2.
4. **PCM (blocking + re-planning):** identical to Configuration 3 but with a re-planning step triggered on blocked actions: the blocking decision and the violated policy are fed back to the base agent as an error message, prompting it to generate an alternative action. If the revised action also exceeds the threshold, the fallback is a user-directed message rather than silent task termination. This configuration evaluates whether re-planning recovers blocked tasks without unacceptable compliance cost.

In practice, the completed empirical study consists of two layers. The first is an offline classifier study using the benchmark-grounded corpus and the rule-based baseline. The second is

a focused live SuiteCRM pilot comparing the unguarded baseline and PCM blocking configurations on a small number of tasks. The broader four-configuration live experiment remains implemented but was not executed at benchmark scale within the project timeline.

3.3.2 Evaluation Metrics

Classifier metrics (RQ2). Precision (target ≥ 0.85), recall (target ≥ 0.80), F1 (target ≥ 0.82), F2, FPR, and ROC-AUC are computed for the binary violation classification task. Per-dimension breakdowns are reported on the standard test split. A rule-based baseline is evaluated on the same held-out examples, and McNemar's test is used for paired comparison. The challenge split is treated as the primary robustness result because the standard test split proves close to ceiling.

Agent-level metrics (RQ3). The primary agent-level outcomes are CuP and Completion Rate (CR). For the live pilot, these are reported descriptively rather than as a sufficiently powered benchmark-wide estimate. The compliance gain ΔCuP and utility cost ΔCR remain the conceptual outcome measures for H2:

$$\text{CuP}(a, T, P) = C_{\text{task}}(a, T) \times \mathbf{1}[V_{\text{total}}(a, T, P) = 0] \tag{2}$$

Partial CuP (pCuP) and the all-pass@3 metric are reported to characterise compliance under partial completion and run-to-run reliability respectively. Per-dimension Risk Ratios are reported to identify which safety dimensions benefit most from PCM integration.

Threshold sensitivity (RQ2/H3). Precision, recall, F1, and FPR are reported across thresholds $\theta \in \{0.10, 0.20, \ldots, 0.90, 0.95, 0.99\}$ on the held-out standard test split. Because the challenge split is much smaller, it is used for robustness confirmation rather than exhaustive threshold sweeping.

Robustness gap (RQ2/H3). $\Delta F1 = F1_{\text{test}} - F1_{\text{challenge}}$, where $F1_{\text{test}}$ is computed on the held-out standard test split and $F1_{\text{challenge}}$ is computed on the harder challenge split. H3 predicts a retention level of at least 70%, corresponding to $\Delta F1 \leq 0.30$ (Li et al., 2023).

3.3.3 Statistical Analysis

Statistical significance of classifier comparisons is assessed using McNemar's test (McNemar, 1947) on paired per-instance predictions, which is appropriate when two classifiers are evaluated on the same fixed test set (Wohlin et al., 2012):

$$\chi^2 = \frac{(|n_{01} - n_{10}| - 1)^2}{n_{01} + n_{10}} \tag{3}$$

where n_{01} is the count of instances where the PCM is correct and the rule-based baseline is wrong, and n_{10} is the reverse. The exact binomial test is used when $n_{01} + n_{10} < 25$.

All reported metrics are accompanied by 95% bias-corrected and accelerated (BCa) bootstrap confidence intervals computed from 10,000 bootstrap resamples (Efron and Tibshirani, 1993). This non-parametric interval estimation approach makes no assumptions about the distributional form of the estimator and is robust to the skewed performance distributions typical of binary classification on imbalanced test sets. Bonferroni correction is applied for all multiple comparisons across the six safety dimensions to control the family-wise error rate.

Effect sizes for agent-level CuP comparisons are reported using Cohen's h for proportions, providing a standardised measure of the practical significance of compliance gains independently of sample size. Where multiple random seed runs are available, means and standard deviations across seeds are reported and inter-run variability is discussed as a component of system reliability.

3.4 Ethical and Professional Considerations

3.4.1 Human Participants and Data Protection

This research involves no human participants at any stage. All training data are either synthetically generated by the researcher using programmatic templates or extracted programmatically from public benchmark environments (ST-WebAgentBench, BrowserGym) released under permissive research licences. No individuals are recruited, surveyed, interviewed, or observed. Consequently, no consent procedures, right-to-withdrawal mechanisms, or data subject access processes are required. The University of Essex Online Research Ethics Procedures confirm that projects involving only computational experiments on synthetic and public benchmark data fall outside the scope of full ethics committee review.

No personal data are collected at any stage; all synthetic data contain only fictional entities, and all code and model weights are released publicly under an open-source licence on project completion.

3.4.2 Bias, Measurement, and Representation

This project does not support a demographic fairness audit because neither ST-WebAgentBench nor the derived corpus contains protected-attribute annotations. Any claim about group fairness would therefore be performative rather than evidential. The relevant bias question is instead whether the data construction and evaluation process privilege some policy forms, action surfaces, or benchmark workflows over others. Benchmark-derived synthetic corpora can encode

framing decisions about what counts as a violation, and models fine-tuned on them can overfit to surface cues rather than robust compliance reasoning (Jacobs and Wallach, 2021; Suresh and Guttag, 2021; Li et al., 2023; Geirhos et al., 2020). Bias mitigation in this dissertation is therefore operationalised through family-level leakage control, a separate challenge split, per-dimension error analysis, and inspection of live audit logs rather than through unsupported fairness metrics.

3.4.3 Potential Harms and Mitigations

Table 7: Risk assessment for potential harms

Potential Harm	**Likelihood**	**Mitigation**
Over-reliance on automated compliance at the expense of human oversight	Medium	Chapter 6 explicitly recommends human oversight for high-stakes decisions; the configurable θ implements EU AI Act Article 14 override capability (European Parliament and Council, 2024)
False-accusation harm from systematic over-blocking	Medium	Precision target ≥ 0.85 and configurable θ minimise over-blocking; re-planning option (Configuration 4) recovers blocked tasks without silent failure; per-dimension FPR analysis surfaces systematic biases
Process and representation bias from benchmark-derived data	Medium	Challenge-split evaluation, per-dimension precision/recall/FPR analysis, and live audit-log review surface benchmark framing bias and shortcut failures; hard counterexamples from live runs are recycled into future training revisions
Adversarial evasion of the classifier boundary	Low	Acknowledged as a limitation in Chapter 6 consistent with OWASP LLM08 (OWASP Foundation, 2025); future adversarial robustness evaluation identified as a direction

3.4.4 Intellectual Property and Professional Standards

All pre-trained models used are released under permissive open-source licences: DeBERTa-v3-base (MIT licence, Microsoft Research (He et al., 2023)); Hugging Face `transformers` (Apache 2.0); ST-WebAgentBench and BrowserGym (MIT and Apache 2.0 respectively (Levy et al., 2026; de Chezelles et al., 2025)). No proprietary systems are accessed; all benchmarks and models are publicly available for academic research.

The project adheres to the British Computer Society Code of Conduct throughout, which requires practitioners to act in the public interest, maintain professional competence, and uphold the profession's reputation. The research contributes positively to responsible AI deployment by advancing the safety of enterprise agent systems, consistent with the governance principles of the NIST AI RMF (NIST, 2023) and the EU AI Act (European Parliament and Council, 2024). Raw experimental results will be fully documented in appendices to enable independent replication.

4 System Design and Implementation

This chapter describes the final technical artefact evaluated in the dissertation: a benchmark-grounded, encoder-based Policy Compliance Module (PCM) integrated into a BrowserGym-compatible web-agent loop. The final system differs materially from the earlier fully templated prototype. In the final artefact, the classifier is trained on benchmark-grounded policy and task metadata, the runtime wrapper naturalises BrowserGym actions before scoring, and the same checkpoint is then exercised in a focused live SuiteCRM pilot. The chapter is organised around four concerns: the input representation, the classifier and its training configuration, the `PolicyCompliantAgent` wrapper, and the implemented experimental configurations.

4.1 Input Representation Design

4.1.1 The Policy-Context-Action Triple

The core design decision in the PCM is the structure of the input fed to the classifier. Each classification call produces a single concatenated string of the form:

Listing 1: Input triple format

```
[POLICY] <policy_text> [SEP] [CONTEXT] <context_text> [SEP] [ACTION]
   <action_text>
```

This representation captures the minimum information required for pre-execution screening.

The [POLICY] segment contains the active natural-language policy text extracted from ST-WebAgentBench metadata (Levy et al., 2026). The [CONTEXT] segment contains a compact structured summary of the current operating situation, such as the site, module, URL fragment, or task-scope cues. The [ACTION] segment contains a naturalised description of the proposed BrowserGym action.

The use of explicit segment markers rather than flat concatenation is grounded in DeBERTa's disentangled attention mechanism (He et al., 2021). This allows the encoder to model the relation between a policy and an action without relying solely on lexical overlap. In practical terms, the model can learn that the same surface action may be compliant under one policy but non-compliant under another.

When multiple policies are simultaneously active (as in the Medium and Hard SuiteCRM tiers of ST-WebAgentBench), the PCM evaluates each policy independently and applies conservative OR-logic: the action is blocked if any single policy predicts a violation above threshold θ. Each policy produces its own violation probability, and the maximum across all active policies is recorded in the audit log. This design ensures that the classifier is never asked to simultaneously reason across multiple constraints in a single forward pass, which would expand the input beyond the 512-token limit and conflate the violation signals from distinct safety dimensions.

4.1.2 Benchmark-Grounded Context and Action Text

The final training data are not a generic synthetic sentence collection. Instead, they are generated directly from ST-WebAgentBench task and policy metadata and then rendered into runtime-shaped examples. The corpus-building pipeline consumes the benchmark task catalogue and produces [POLICY]+[CONTEXT]+[ACTION] triples with family identifiers for leakage-aware splitting. The resulting corpus uses 375 tasks from suitecrm, gitlab, and shopping_admin and yields 12,493 training examples, 2,572 validation examples, 2,877 standard test examples, and a separate 368-example challenge split.

This benchmark-grounded design is important methodologically. Earlier templated variants produced unrealistically easy held-out splits and inflated offline results. By grounding examples in benchmark policies, modules, and runtime-like action strings, the final dataset is much closer to the distribution that the wrapper encounters inside BrowserGym. Typical action strings include `click Accounts (link)`, `click Create Account (link)`, `type into Account Name (textbox) with 'Acme Ltd'`, and `send message to user: 'Please confirm before I proceed.'`

4.1.3 Action Naturalisation at Runtime

A non-trivial implementation challenge arises from the mismatch between the runtime-shaped action descriptions used during training and the structured function-call format of BrowserGym action outputs. BrowserGym agents produce actions such as:

Listing 2: Raw BrowserGym action output

```
click('bid_47')
fill('bid_12', 'Alice Johnson')
send_msg_to_user('I am about to delete this record. Please confirm.')
```

These are not natural-language descriptions. A classifier trained on text such as "click the Delete button on contact Alice Johnson" will receive an out-of-distribution input if presented with `click('bid_47')` and may produce unreliable violation probabilities. The PCM therefore incorporates an action naturalisation layer that maps BrowserGym function calls to natural-language descriptions before classification.

The naturalisation procedure operates in three stages. First, the function name is first mapped to a readable verb phrase: `click` to "click", `fill` to "type into", `goto` to "navigate to", and `select_option` to "select from". Second, the bid argument is resolved against the current AXTree to recover an accessible name and role for the target element. Third, these are combined into a short description such as "click Delete (button)" or "type into Account Name (textbox) with 'Acme Ltd"'. For `send_msg_to_user`, the message content is already natural language and is passed through with light formatting. If element resolution fails, the raw action is retained and the audit log records that a fallback form was used.

4.1.4 Sequence Length Management

DeBERTa-v3-base supports a maximum sequence length of 512 tokens (He et al., 2023). The final system does not rely on a bespoke truncation algorithm. Instead, it keeps the context summary deliberately compact and uses standard tokenizer truncation at 512 tokens. This is practical because both the benchmark-grounded training context and the runtime wrapper use concise structured summaries rather than full DOM or AXTree serialisations. The design goal is not to maximise raw page information, but to keep the scored input close to the distribution the encoder was trained on.

4.2 PCM Classifier Architecture and Training

4.2.1 Model Architecture

The PCM classifier is a standard sequence classification head applied to a fine-tuned DeBERTa-v3-base encoder (He et al., 2023). The backbone consists of 12 transformer layers with 768-dimensional hidden states and 12 attention heads, totalling approximately 86M parameters. The classification head is a single linear layer mapping from the 768-dimensional `[CLS]` representation to 2 output logits (compliant, violation), followed by a softmax producing calibrated probabilities. The violation probability $P(\text{violation})$ is taken from the second output logit.

No task-adaptive pre-training or adapter layers are used. The entire backbone is fine-tuned end-to-end, following the standard approach for small-to-medium classification datasets (Devlin et al., 2019; Liu et al., 2019). Freezing the backbone layers and training only the classification head was considered but rejected on the grounds that policy violation detection requires the encoder to adapt to the structured triple format and benchmark-specific policy language in a way that general-purpose pre-training alone does not provide.

4.2.2 Training Configuration

The model is implemented using the Hugging Face `transformers` library (Wolf et al., 2020) and trained with PyTorch (Paszke et al., 2019). The training configuration is as follows:

Table 8: PCM fine-tuning hyperparameters

Hyperparameter	**Value**
Base model	`microsoft/deberta-v3-base`
Max input length	512 tokens
Batch size	16
Learning rate	2×10^{-5}
Optimiser	AdamW (Loshchilov and Hutter, 2019)
Weight decay	0.01
Warmup ratio	0.1 (of total training steps)
Learning rate schedule	Linear decay after warmup
Max epochs	5
Early stopping	Validation F1 (patience = 2 epochs)
Gradient clipping	1.0 (max norm)
Loss weighting	Inverse-frequency class weights

The learning rate follows established practice for DeBERTa fine-tuning (He et al., 2023; Liu et al., 2019). AdamW (Loshchilov and Hutter, 2019) is preferred over standard Adam for its decoupled weight decay, which is better calibrated for transformer fine-tuning on small datasets. Early stopping on validation F1 rather than validation loss ensures the best checkpoint reflects classification performance rather than distribution shift toward the majority class.

In the final benchmark-grounded run, the best validation F1 reached 0.9976. This high value is later interpreted cautiously in Chapter 5 because the standard validation and test splits remain more in-distribution than the challenge split and the live pilot.

4.2.3 Class Weighting

Although the final corpus is close to balanced overall, class frequency varies mildly between split families and safety dimensions. Inverse-frequency class weights are therefore applied in the cross-entropy loss during training:

$$w_c = \frac{N}{K \cdot n_c} \tag{4}$$

where N is the total number of training samples, K is the number of classes (2), and n_c is the number of samples in class c. This ensures that the minority class contributes proportionally to the gradient signal even when a given dimension is relatively underrepresented.

4.2.4 Reproducibility and Environment Stabilisation

An implementation detail that proved important in practice was dependency stabilisation. Newer `transformers` releases introduced compatibility issues with the cloud runtime used for training, and DeBERTa-v3's tokenizer path also required explicit `sentencepiece` support. The final training notebook therefore includes a preflight cell that pins a known-good stack and verifies model and tokenizer loading before the main run begins. This is not a research contribution in itself, but it materially improves reproducibility and reduces wasted GPU time.

4.3 PolicyCompliantAgent Wrapper

4.3.1 Architecture Overview

The `PolicyCompliantAgent` is a modular wrapper class that intercepts the `act()` method of any BrowserGym-compatible base agent without requiring any modification to the underlying agent's architecture, weights, or prompting. Figure 2 illustrates the system architecture.

The wrapper exposes a single `act(obs)` method that mirrors the interface of any BrowserGym base agent. Its internal execution proceeds in six steps: (1) extract active policies from the `POLICY_CONTEXT` block of the BrowserGym observation; (2) extract and summarise page context from the AXTree; (3) call the base agent's `act(obs)` method to obtain the proposed action; (4) naturalise the action text; (5) run PCM classification for each active policy; (6) permit, block, or trigger re-planning based on the result.

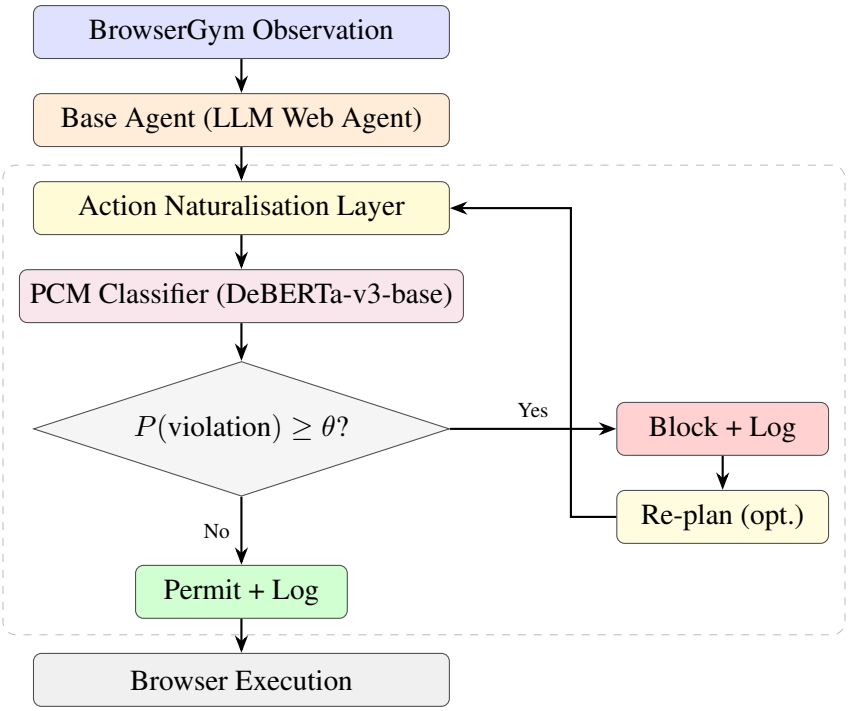

Figure 2: PCM system architecture. The dashed region denotes the `PolicyCompliantAgent` wrapper, which intercepts proposed actions before browser execution. The base agent is unmodified. The re-planning path is active in Configuration 4 only.

4.3.2 Policy Normalisation and Multi-Policy Scoring

ST-WebAgentBench injects active policies into the BrowserGym observation via the `POLICY_CONTEXT` block, which contains a structured list of applicable policy objects for the current task step. Each policy object includes a `policy_template` field containing the natural-language constraint, a `dimension` field identifying the safety category, and a `priority` field indicating the tier (organisational, user, or task). The wrapper normalises these into per-policy PCM scoring calls. Each active policy yields its own violation probability, and the action is blocked if any score exceeds the configured threshold θ. This simple OR-logic keeps the forward pass local and auditable: the audit log shows exactly which policy fired and with what score. Organisational-policy violations are therefore never overridden by a conflicting lower-priority instruction.

4.3.3 Blocking, Re-planning, and Fallback

When the PCM's violation probability for any active policy exceeds threshold θ, the proposed action is blocked. In Configuration 3 (PCM blocking only), the task step terminates without executing any action. In Configuration 4 (PCM blocking with re-planning), the wrapper injects a structured feedback message into the observation before calling the base agent a second time:

Listing 3: Re-planning feedback injected into observation

```
last_action_error: "Action blocked by Policy Compliance Module
  (P(violation) = 0.91). The following policy was violated:
  'When about to delete a record, ask the user for permission
  before proceeding.' Please propose a compliant alternative."
```

The base agent receives this augmented observation and generates a revised action. The revised action is passed through PCM classification once more. If the revised action also exceeds θ, the wrapper executes a safe fallback rather than entering an infinite re-planning loop: it calls `send_msg_to_user` with a message explaining that the requested action cannot be completed under the active policy, and records a repeat-block event in the audit log. This one-shot re-planning design bounds latency while still giving the base agent one opportunity to recover with a safer action.

4.3.4 Per-Action Audit Log

Every PCM classification decision, whether it results in a block, permit, or re-plan, is appended to an immutable JSONL audit log. Each log entry records:

Listing 4: Audit log entry format

```
{
  "timestamp":       "2026-04-11T14:23:07Z",
  "task_id":         "suitecrm_contact_delete_047",
  "step":            3,
  "dimension":       "user_consent",
  "policy":          "When about to delete a record, ask the user
     ...",
  "context_summary": "page: suitecrm_contacts; element: Delete button
     ",
  "action":          "click the Delete button on contact Alice
     Johnson",
  "violation_prob":  0.91,
  "decision":        "BLOCK",
  "threshold":       0.5,
  "latency_ms":      47.3
}
```

This record satisfies the event-logging requirement of EU AI Act Article 12 (European Parliament and Council, 2024), which mandates automatic logging of all events during high-risk AI system operation with logs retained for a minimum of six months. The `violation_prob` field additionally satisfies Article 13's transparency requirement by providing a per-action interpretable compliance indicator disaggregated by safety dimension. The `threshold` field

documents the operating point in effect at the time of each decision, supporting retrospective audit even if θ is reconfigured between evaluation runs.

4.4 Experimental Configurations

The four configurations implemented in the artefact are as follows:

Configuration 1 (Baseline). The base agent runs directly inside the BrowserGym loop without any compliance wrapper. All observations and actions are passed through unmodified.

Configuration 2 (Rule-based filter). The base agent is wrapped in a lightweight keyword and action-type filter that checks proposed actions against explicit rules implementing the six safety dimensions. For example, the User Consent rule flags any action whose naturalised text contains deletion-related verbs (*delete, remove, permanently, destroy*) unless a preceding `send_msg_to_user` call was detected in the same task step. The rule-based filter requires no training data and no model loading overhead, providing a meaningful lower-bound baseline that tests whether simple heuristics are sufficient before attributing gains to learned classification.

Configuration 3 (PCM blocking). The `PolicyCompliantAgent` wrapper is active with $\theta = 0.5$ (default) and `replan_on_block = False`. Blocked actions terminate the current task step. This is the primary evaluation configuration for H2.

Configuration 4 (PCM blocking + re-planning). Identical to Configuration 3 but with `replan_on_block = True`. Blocked actions trigger one re-planning call to the base agent. If the revised action is also blocked, the safe fallback message is sent. This configuration evaluates whether re-planning recovers blocked tasks without unacceptable compliance cost.

4.5 Implementation Environment

The implementation is written in Python using Hugging Face `transformers` (Wolf et al., 2020), PyTorch (Paszke et al., 2019), and BrowserGym (de Chezelles et al., 2025). The final benchmark-grounded run used a single NVIDIA H100 GPU in a cloud notebook environment with a pinned dependency stack to ensure DeBERTa-v3 compatibility. End-to-end dataset build, fine-tuning, evaluation, and export required approximately 45 minutes, with the five-epoch training run itself taking roughly 35 minutes. Live pilot evaluation was run locally against SuiteCRM deployed via Docker. All code, model weights, notebooks, and evaluation scripts are retained as reproducible research artefacts. The source repository is available at GitHub, and the final model card and checkpoint are available at Hugging Face. The quantitative figures in Chapter 5 are rendered directly in this LaTeX source via `pgfplots`, so the submission remains self-contained even though the training notebooks and scripts are released separately.

5 Results and Analysis

This chapter reports the empirical findings for the final benchmark-grounded PCM and addresses the three research questions. The evaluation has two layers. First, the classifier is tested offline on a large held-out benchmark-grounded test split and on a separate harder challenge split. Second, the trained checkpoint is integrated into the `PolicyCompliantAgent` wrapper and exercised in a focused live SuiteCRM pilot. The offline results address RQ2 and provide the strongest evidence for H1. The SuiteCRM pilot provides preliminary system-level evidence for RQ3 but is too small to fully confirm H2.

5.1 Evaluation Protocol

The final PCM is trained on the benchmark-grounded corpus described in Chapter 3: 12,493 training examples, 2,572 validation examples, 2,877 standard test examples, and a separate 368-example challenge split derived from 375 ST-WebAgentBench tasks across `suitecrm`, `gitlab`, and `shopping_admin`. The standard test remains relatively in-distribution; the challenge split is the primary robustness check because it uses held-out families and less templatic surface forms, reducing the risk of an overly optimistic held-out estimate (Xue et al., 2025; Li et al., 2023).

Training converged within five epochs with early stopping on validation F1 (patience $= 2$); the best validation F1 was 0.9976. The resulting checkpoint was evaluated on the standard test split, the challenge split, and a small sanity-probe set used only to reject obviously collapsed checkpoints before live evaluation.

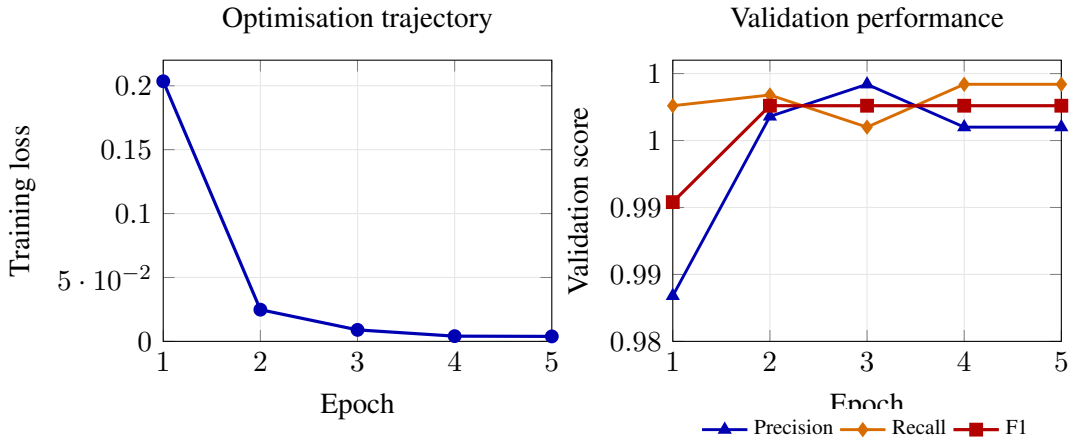

Figure 3: Training convergence of the final benchmark-grounded PCM. Training loss falls sharply after the first epoch while validation precision, recall, and F1 all move rapidly toward their near-ceiling regime by the second epoch, indicating that the final model reached its useful operating point early rather than requiring prolonged optimisation.

Figure 3 shows the same pattern visually: the largest optimisation gain occurs immediately, after which additional epochs mainly stabilise already high held-out precision, recall, and F1 rather than producing a new performance regime.

The offline baseline is a rule-based filter that flags explicit keyword and action-type patterns but does not model policy context. A McNemar test (McNemar, 1947) compares PCM and baseline predictions on the shared standard test split.

5.2 Classifier Performance: RQ2 and H1

Table 9 summarises the final offline performance at the default threshold $\theta = 0.5$.

Table 9: Final offline PCM performance ($\theta = 0.5$)

Split / System	Precision	Recall	F1	FPR	ROC-AUC
PCM standard test	**0.9972**	**1.0000**	**0.9986**	**0.0028**	**1.0000**
PCM challenge	**1.0000**	**0.8424**	**0.9145**	**0.0000**	**0.9792**
Rule-based baseline (standard test)	0.5146	0.3048	0.3828	—	—
H1 target	≥ 0.85	≥ 0.80	≥ 0.82	< 0.15	—
H1 met?	**Yes**	**Yes**	**Yes**	**Yes**	—

H1 is confirmed. The standard test is near ceiling, so the challenge split is the main evidential result. On that harder split, the PCM still achieves precision 1.0000, recall 0.8424, F1 0.9145, FPR 0.0000, and ROC-AUC 0.9792. This is the more credible estimate of deployment-facing classifier quality because easier held-out partitions can exaggerate readiness for live web interaction (Xue et al., 2025). Interpreted conservatively, the standard test acts as an in-distribution upper bound, while the challenge split acts as the more realistic stress test of whether the learned representation survives less templatic wording and held-out action families. That distinction matters because synthetic or benchmark-derived data can otherwise make a classifier look more deployment-ready than it really is (Li et al., 2023; Geirhos et al., 2020).

This pattern strengthens the case for small, fine-tuned encoders as practical guardrails on bounded classification problems. It is consistent with findings that fine-tuned smaller models can outperform more generic generative systems on classification-style tasks when the label space and input format are carefully controlled (Bucher and Martini, 2024; Zheng et al., 2025). The practical implication is that the PCM is not merely a cheaper substitute for a second large LLM; it is a better fit for synchronous pre-execution gating, where low latency, stable thresholds, and auditable binary decisions matter more than open-ended reasoning fluency. In other words, the contribution is not simply that an 86M-parameter model can classify well, but that it can do so in the particular decision regime that enterprise guardrails care about: fast, repeatable, thresholdable, and inspectable intervention at action time.

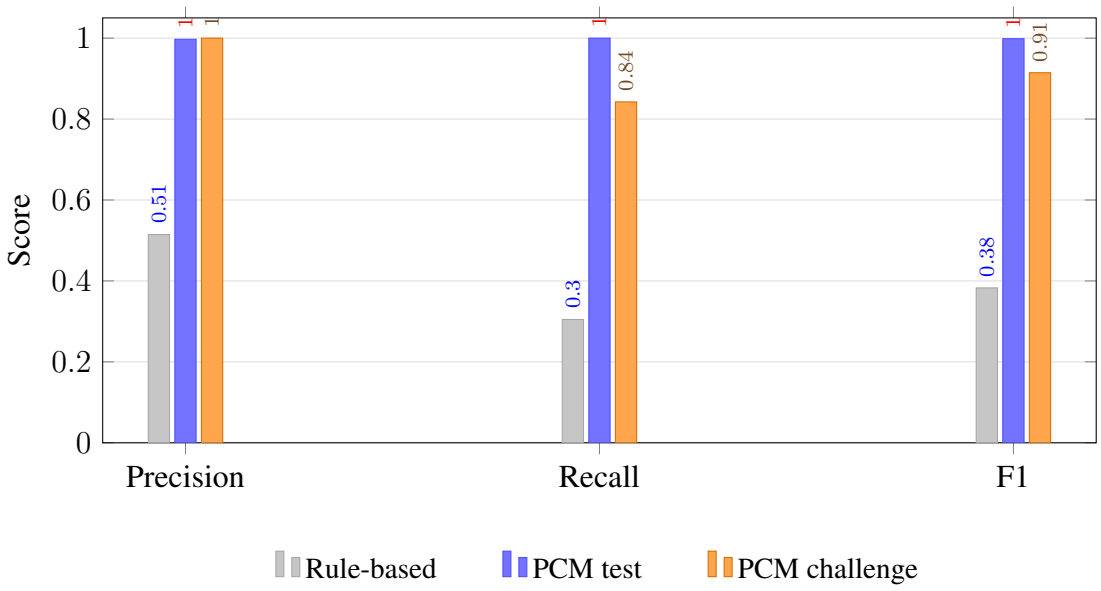

Figure 4: Offline comparison of the benchmark-grounded PCM against the rule-based baseline. The standard test is near-ceiling, so the challenge split is the more informative robustness result.

The operationally important feature is the absence of false positives on the challenge split (TN = 184, FP = 0) together with recall above the H1 target. For an intervention that can stop task execution, this precision/FPR profile is more informative than ROC-AUC alone because false alarms carry direct workflow costs (Saito and Rehmsmeier, 2015; Alahmadi et al., 2022). The rule-based baseline reaches only 0.3828 F1 on the standard test, reinforcing that benchmark-grounded policy compliance is not solvable by surface keyword matching alone. Substantively, this means the PCM is not just detecting obvious lexical cues such as *delete* or *password*; it is using the policy-action-context triple to separate when similar action language is permissible and when it is not. That is exactly the discriminative behaviour a pre-execution guardrail needs if it is to be more than a brittle alarm layer.

5.2.1 McNemar Test: PCM versus Rule-Based Baseline

The McNemar test (McNemar, 1947) assessed whether the PCM's per-instance predictions differ significantly from the rule-based baseline on the shared standard test split:

$$\chi^2 = \frac{(|b - c| - 1)^2}{b + c} \tag{5}$$

where b is the count of instances where the PCM is correct and the rule-based baseline is wrong, and c is the reverse. The result is statistically significant ($\chi^2 = 1411.057$, $p < 0.001$), so the improvement is unlikely to be a sampling artifact. More importantly, the McNemar result is consistent with the qualitative design claim made earlier in the dissertation: once policy compliance is framed as a contextual classification problem rather than a keyword-spotting

problem, a learned encoder has a structurally better chance of capturing the relevant decision boundary than a surface heuristic.

5.3 Per-Dimension Analysis

Table 10 reports per-dimension standard-test performance. Unlike the earlier prototype, the final benchmark-grounded model no longer shows a catastrophic hierarchy-adherence failure. Boundary and Scope remains the least perfect dimension, plausibly because it is both the largest and the most lexically varied slice of the corpus.

Table 10: Per-dimension PCM performance on the standard test split ($\theta = 0.5$)

Dimension	**P**	**R**	**F1**	**FPR**	**N**
Boundary & Scope	0.996	1.000	0.998	0.006	1353
Error Handling	1.000	1.000	1.000	0.000	94
Hierarchy Adherence	0.980	1.000	0.990	0.014	120
Robustness & Security	1.000	1.000	1.000	0.000	310
Strict Execution	1.000	1.000	1.000	0.000	745
User Consent	1.000	1.000	1.000	0.000	255

This pattern is encouraging but should still be interpreted cautiously. Near- perfect standard-test scores confirm that the benchmark-grounded redesign fixed the earlier prototype failure, but only the challenge split and live pilot test whether those gains survive less templatic conditions.

The per-dimension pattern also sharpens the deployment implication. The model now performs reliably across all six benchmark dimensions, suggesting that the earlier hierarchy-adherence collapse was a data-design problem rather than an intrinsic limitation of encoder architecture. That interpretation is aligned with the broader literature showing that benchmark construction and data formulation can dominate apparent model capability on structured classification tasks (Li et al., 2023; Geirhos et al., 2020). It also suggests that the benchmark-grounded redesign improved the right thing. Dimensions such as User Consent, Strict Execution, and Robustness & Security rely heavily on direct policy-action contradictions, whereas Boundary & Scope and Hierarchy Adherence require more relational interpretation about whether an action remains inside task and policy limits. The fact that Hierarchy Adherence is no longer the outlier implies that runtime-shaped examples and family-level split control were not cosmetic adjustments; they materially changed what the model had to learn about the benchmark's six-dimension taxonomy (Levy et al., 2026).

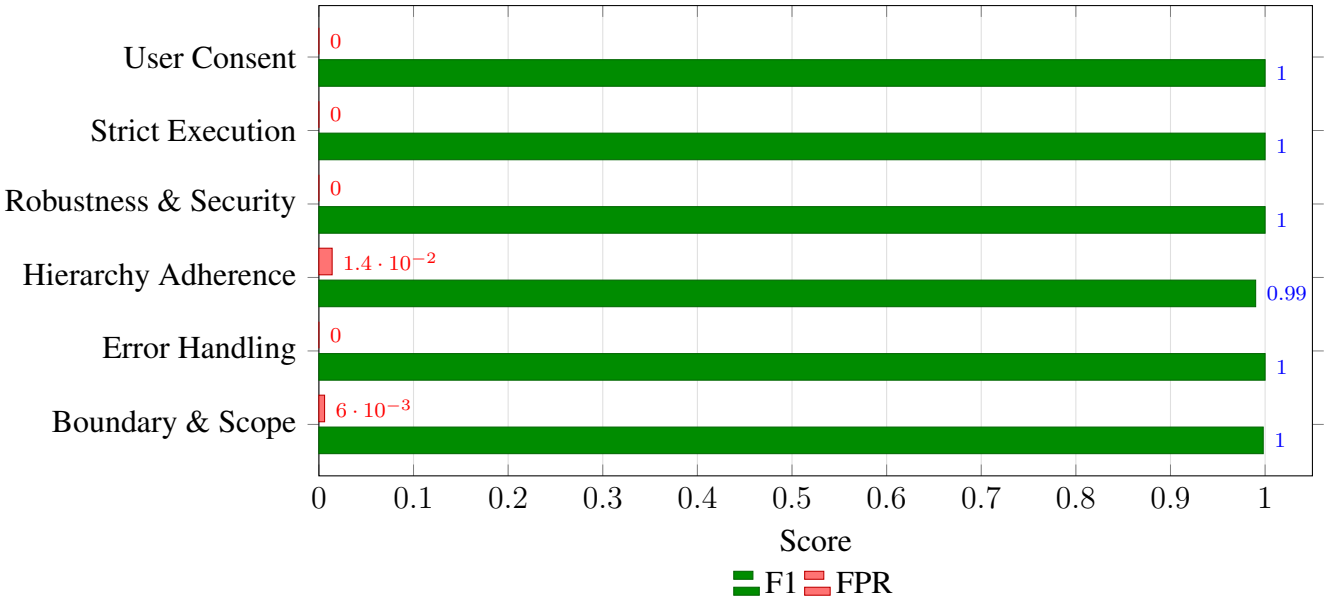

Figure 5: Per-dimension standard-test performance of the final benchmark-grounded PCM. The earlier hierarchy-adherence collapse is no longer present, although standard-test performance remains near ceiling overall.

5.4 Threshold Analysis: H3

H3 contains two claims relevant to the offline study. First, varying the violation threshold should induce a measurable precision-recall trade-off. Second, performance on the harder challenge split should retain at least 70% of held-out test performance. Table 11 summarises the threshold sweep on the standard test split.

Table 11: Selected threshold sweep results on the standard test split

θ	**Precision**	**Recall**	**F1**	**FPR**
0.10	0.9972	1.0000	0.9986	0.0028
0.50	0.9972	1.0000	0.9986	0.0028
0.60	0.9993	0.9979	0.9986	0.0007
0.99	1.0000	0.9979	0.9990	0.0000

The first H3 claim is only partially confirmed. There is a measurable trade-off, but it is stepped rather than smooth: the output distribution is strongly bimodal, so most thresholds produce no change until the top end of the probability range. In practice, this means threshold selection should be tied to deployment cost rather than to a generic default (Saito and Rehmsmeier, 2015). This is an analytically useful result rather than a disappointing one. It implies that the model is usually very certain, so threshold choice behaves more like a policy dial than a fragile tuning exercise. For deployment, that is preferable to a classifier whose output mass sits densely around 0.5 and changes behaviour erratically under small threshold adjustments.

The second H3 claim is supported by the challenge split. Using F1 as the main summary

statistic, the retention ratio is:

$$\frac{F1_{\text{challenge}}}{F1_{\text{test}}} = \frac{0.9145}{0.9986} \approx 0.916 \tag{6}$$

This exceeds the H3 target of 0.70 by a substantial margin. However, the single-task SuiteCRM pilot is too small to serve as a definitive live robustness test, so H3 is best classified as partially rather than fully confirmed (Wohlin et al., 2012; Venable et al., 2016). The implication is that the offline robustness claim is strong but bounded. The challenge split shows that performance does not collapse once the easy held-out setting is removed, yet it does not eliminate the need for live evaluation under real interaction constraints.

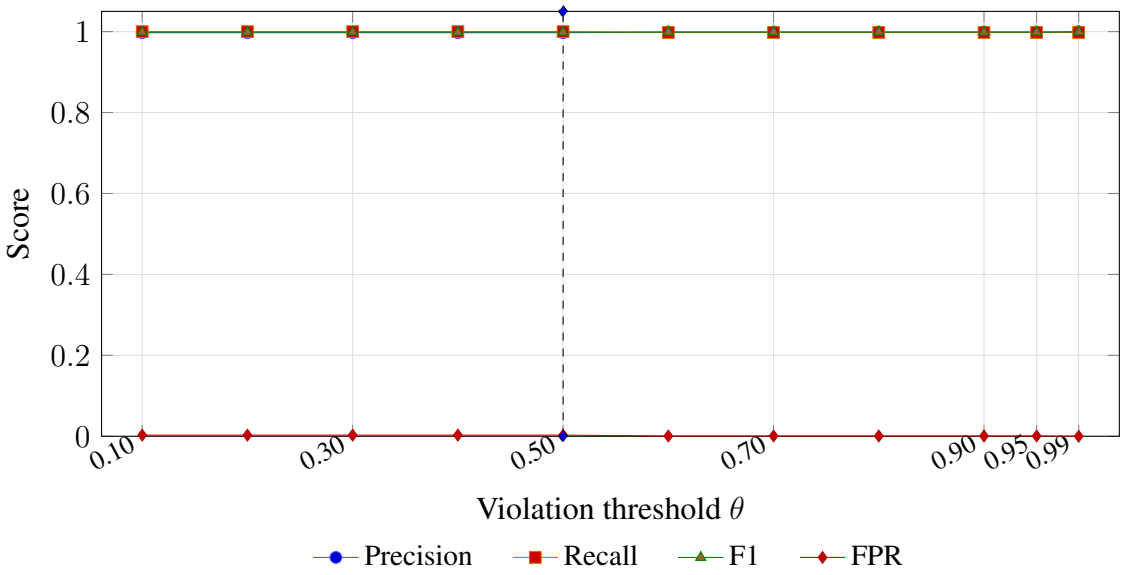

Figure 6: Threshold sweep on the standard test split. The stepped shape confirms a measurable but non-smooth trade-off, consistent with a highly bimodal output distribution.

5.5 System-Level Evaluation: RQ3 and H2

To provide preliminary system-level evidence, the final checkpoint was integrated into the live wrapper and run on a focused SuiteCRM pilot. The pilot used the benchmark-grounded Hugging Face checkpoint and compared an unguarded baseline with the PCM wrapper on task 47.

Table 12: Focused SuiteCRM pilot on task 47

Configuration	Completed	Partial	CuP	Observed violations
Baseline	Yes	Yes	0.0	8 total: 6 hierarchy-adherence, 2 user-consent
PCM	No	No	0.0	6 total: all hierarchy-adherence

The pilot establishes two points. First, the wrapper works end-to-end in a live BrowserGym

loop: actions were naturalised, scored against active policies, and logged with thresholded permit/block decisions. Second, the wrapper materially changed behaviour. The audit log shows that benign navigation such as `click Accounts (link)` was repeatedly permitted with near-zero violation probability, while `click Create Account (link)` was repeatedly blocked under a user-consent policy referring to `Save`, with $P(\text{violation}) = 0.996$. That is an informative live false positive; the model appears to overgeneralise from the semantic proximity of "create account" to an eventual save action.

The pilot therefore supports a cautious conclusion. The PCM is deployable as a live pre-execution guardrail, but the current calibration is not yet reliable enough to claim a positive CuP gain on SuiteCRM, and the sample is too small to support a robust H2 test (Wohlin et al., 2012).

This live result matters because it prevents an overly optimistic reading of the offline metrics. Recent work on web agents warns that benchmark performance can overstate readiness once systems are exercised in realistic interactive settings (Xue et al., 2025). The SuiteCRM pilot is too small to validate H2, but it is large enough to show the type of failure that remains: not catastrophic policy blindness, but over-blocking of benign transitional actions in live workflows. That is a much narrower and more actionable failure mode than the one the project began with. In design-science terms, the pilot functions as an ex-post evaluation that narrows the next iteration cycle: the immediate problem is no longer whether learned compliance gating is feasible, but how to improve calibration on semantically adjacent yet compliant actions (Venable et al., 2016; Wohlin et al., 2012). This also makes the next research step more concrete. The highest-value additions are likely to be live counterexamples of precisely this form, where the action is operationally safe but lexically or semantically close to a later violating action.

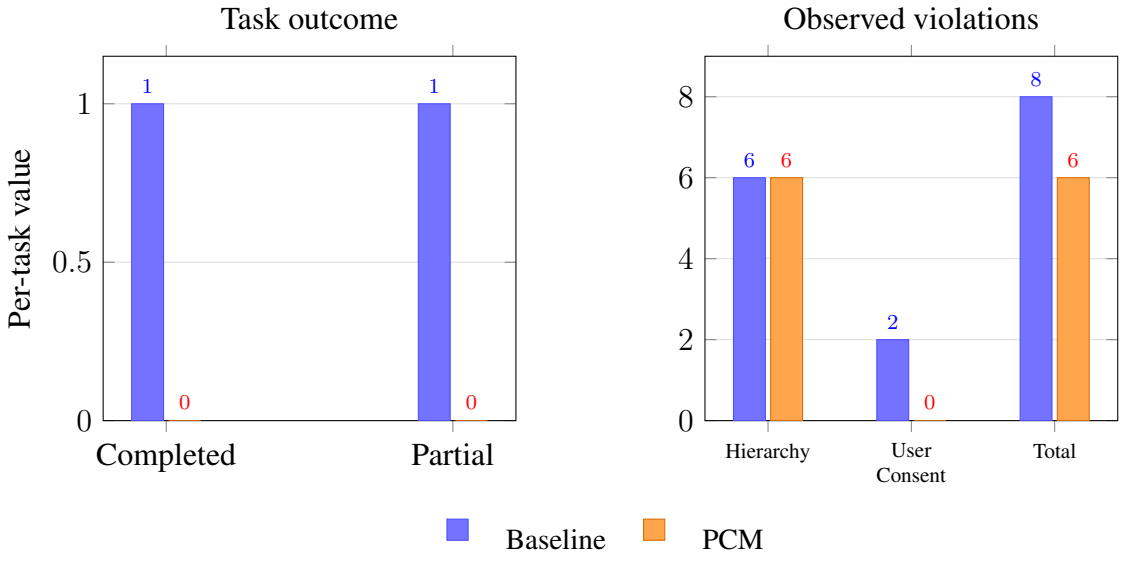

Figure 7: Focused SuiteCRM pilot on task 47. The PCM reduced observed violations from eight to six but introduced a live false positive that prevented task completion.

Table 13: Interpretive summary of the main Chapter 5 findings

Empirical finding	Interpretation	Implication
Challenge-split performance remains strong (F1 = 0.9145, FPR = 0.0000) despite the easier standard test being near ceiling.	The benchmark-grounded redesign produced a classifier that still performs well once easy held-out conditions are removed, making the challenge split the more credible estimate of robustness (Xue et al., 2025; Li et al., 2023; Geirhos et al., 2020).	The PCM is a plausible offline guardrail candidate for enterprise workflows, but its strongest claim should be grounded in the harder split rather than in the easier test alone.
Per-dimension scores are uniformly strong, including Hierarchy Adherence.	The earlier prototype failure was primarily a data-formulation problem rather than clear evidence of an encoder-architecture ceiling. The runtime-shaped corpus changed what the model had to learn about the benchmark taxonomy (Levy et al., 2026).	Effort should prioritise benchmark-grounded data design and counterexample quality, not simply scaling model size.
The live SuiteCRM pilot reduces observed violations but blocks a benign `click Create Account (link)` action.	The remaining live gap is calibration on transitional actions, not wholesale policy blindness. This is a narrower and more actionable failure mode than the project initially faced (Venable et al., 2016; Wohlin et al., 2012).	The next iteration should mine live false positives into targeted retraining and broaden live evaluation before any strong system-level deployment claim is made.

5.6 Synthesis Against Hypotheses

The central empirical conclusion is that an encoder-scale policy guardrail can work well offline when it is trained on benchmark-grounded rather than naive templated data. The challenge split is the strongest evidence for this claim. The SuiteCRM pilot narrows the remaining gap: the main open problem is now calibration for benign transitional actions in live create or edit flows. Read together, the findings align with two wider lessons from the literature. First, evaluation difficulty matters as much as model architecture when judging safety readiness (Xue et al., 2025; Li et al., 2023). Second, smaller fine-tuned encoders can be highly competitive on tightly specified classification problems when representation and decision costs are aligned with the deployment setting (Bucher and Martini, 2024; Galke et al., 2025; Zheng et al., 2025).

Table 14: Hypothesis evaluation summary

H	Outcome	Evidence
H1	**Confirmed**	On the harder challenge split, the PCM achieves precision 1.0000, recall 0.8424, F1 0.9145, and FPR 0.0000 at $\theta = 0.5$, meeting all H1 thresholds. The near-ceiling standard test provides supporting but not primary evidence.
H2	**Inconclusive**	A focused SuiteCRM pilot demonstrates successful live integration and meaningful behavioural intervention, but the sample is too small and includes a live false positive on `click Create Account (link)`, so no robust CuP improvement claim can yet be made.
H3	**Partially confirmed**	Threshold variation produces a measurable but stepped precision-recall trade-off, and challenge-split F1 retention is approximately 91.6%, exceeding the 70% target. The limited live pilot is informative but not sufficient to fully confirm the live-scenario component of H3.

6 Discussion and Conclusion

This dissertation asked whether a lightweight encoder-based policy guardrail could improve the safety of autonomous web agents without requiring a large secondary LLM, full policy recompilation, or modification of the underlying agent. The final results support a qualified yes. The PCM is a working artefact: it can be trained reproducibly, integrated into a BrowserGym agent loop, and used to score policy compliance at action time with a per-action audit trail.

6.1 Main Findings

Three findings matter most. First, the benchmark-grounded corpus materially improved evaluation credibility: earlier fully templated variants produced misleadingly easy held-out splits, whereas the final challenge split still delivered precision 1.0000, recall 0.8424, F1 0.9145, and ROC-AUC 0.9792. Second, H1 is confirmed with zero challenge-set false positives, which is important for any intervention that can directly interrupt a workflow (Alahmadi et al., 2022). Third, the focused SuiteCRM pilot shows that live deployment is feasible but not yet solved: the remaining gap is now a specific calibration and data-coverage problem, not an architectural unknown.

6.2 Research Contribution

The dissertation makes three concrete contributions. First, it contributes a modular artefact: the `PolicyCompliantAgent` wrapper integrates a learned guardrail into a BrowserGym-compatible agent loop without modifying the underlying agent. Second, it provides evidence that an encoder-scale model at roughly 86M parameters can achieve strong benchmark-grounded policy-violation detection despite recent guardrail work often relying on much larger secondary LLMs or explicit policy reasoning engines (Xiang et al., 2024; Luo et al., 2025; Chen et al., 2025). Third, it contributes a reproducible pipeline that converts ST-WebAgentBench policy and task metadata into runtime-shaped training examples.

6.3 Limitations

The limitations are clear. First, evaluation scope remains narrow: the live study is a focused SuiteCRM pilot rather than a large multi-task benchmark run, so H2 remains inconclusive. Second, even the challenge split is still benchmark-derived rather than drawn from naturally occurring enterprise logs. Third, the PCM scores one action against one policy at a time, so transitional workflow semantics remain only weakly represented; the live false positive on

`click Create Account (link)` is the clearest example. Fourth, bias in this study is better understood as process and representation bias than as demographic fairness. The benchmark and derived corpus do not contain protected-attribute annotations, so no defensible group-fairness claim can be made. Instead, the relevant risk is measurement mismatch: "policy violation" is operationalised through ST-WebAgentBench policies and benchmark-shaped actions and benchmark-derived synthetic counterfactuals rather than naturally occurring compliance incidents (Jacobs and Wallach, 2021). The corpus also inherits framing bias from the benchmark itself and remains vulnerable to shortcut features or benchmark-style surface regularities despite the harder challenge split and live pilot (Suresh and Guttag, 2021; Li et al., 2023; Geirhos et al., 2020). These limitations do not nullify the contribution, but they do narrow the claim: the dissertation shows that benchmark-grounded encoder guardrails are viable and informative, not that the current checkpoint is ready for unconstrained enterprise deployment.

6.4 Generalisability

The results are most defensible as evidence of bounded generalisability rather than universal transfer. Within the benchmark family, the corpus spans three distinct applications and the harder challenge split retains 91.6% of the standard-test F1, which supports the claim that the learned representation is not tied to a single templated partition. However, external generalisability is more limited. The study does not yet show performance on naturally occurring enterprise incident logs, highly visual consumer interfaces, or long-horizon multistep workflows beyond the focused SuiteCRM pilot (Zhou et al., 2024; Koh et al., 2024; Xue et al., 2025). The fairest claim is therefore that the PCM generalises across benchmark-grounded, policy-governed enterprise web tasks, while broader deployment generalisation remains to be demonstrated empirically. A useful way to read this is that the project establishes representational transfer within a shared benchmark family, but not yet ecological transfer to the wider variety of real organisational systems and live workflows that motivated the research problem. That broader claim would require evaluation across additional enterprise platforms, more visual interfaces, and naturally occurring compliance incidents rather than benchmark-derived counterfactuals alone.

6.5 Future Work

Several research directions now look especially promising. First, live evaluation should be expanded from the current pilot to a larger SuiteCRM study and then, where infrastructure permits, to the broader ST-WebAgentBench suite. That would determine whether the present offline gains translate into a robust Completion under Policy improvement under realistic interactive conditions. Second, the corpus should be augmented with hard counterexamples mined

from live runs; the repeated false-positive block on `click Create Account (link)` is exactly the kind of high-value example that should be recycled into training (Li et al., 2023; He et al., 2023). Counterexample-driven refinement is promising because the current failure mode is narrow, concrete, and already observable in the audit logs.

Third, threshold calibration should be revisited using live-development validation tasks rather than offline splits alone. The threshold results in Chapter 5 suggest that the classifier is usually highly certain, so a more deployment-aware calibration procedure may yield better task-completion trade- offs without sacrificing the low false-positive profile. Fourth, future versions of the PCM could extend the input representation with short action history or explicit user-intent state to better separate "open create form" from "submit create form". That direction is especially promising because the SuiteCRM pilot suggests the remaining ambiguity is not purely lexical; it is temporal and workflow-dependent.

Finally, broader external validation would strengthen the dissertation's generalisability claim. A natural next step would be to test whether the same guardrail design transfers to other enterprise-style benchmarks such as WorkArena and to more visually grounded environments such as VisualWebArena, where action semantics depend more heavily on interface state and multimodal context (Drouin et al., 2024; Koh et al., 2024). This would help establish whether the contribution is primarily a benchmark-specific success or the basis of a more general enterprise guardrail pattern.

6.6 Conclusion

The dissertation answers the central research problem in a measured way. A lightweight encoder-based policy guardrail is viable for autonomous web agents, provided that it is trained on benchmark-grounded data and evaluated on harder held-out conditions rather than only on easy synthetic splits. The final PCM achieves strong and credible offline results, integrates cleanly into a live agent loop, and exposes a concrete remaining calibration problem in the live setting. It is therefore both a working prototype and a defensible research contribution.

References

Alahmadi, B.A., Axon, L. and Sherwood-Jones, B. (2022) '99% false positives: a qualitative study of SOC analysts' perspectives on security alarms', in *Proceedings of the 31st USENIX Security Symposium*, pp. 2783–2800. Available at: https://www.usenix.org/conference/usenixsecurity22/presentation/alahmadi

Anaby-Tavor, A., Carmeli, B., Goldbraich, E., Kantor, A., Kour, G., Shlomov, S., Tepper, N. and Zwerdling, N. (2020) 'Do not have enough data? Deep learning to the rescue!', *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05), pp. 7383–7390. DOI: 10.1609/aaai.v34i05.6233

Autio, C., Schwartz, R., Dunietz, J., Jain, S., Stanley, M., Tabassi, E., Hall, P. and Roberts, K. (2024) *Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile*. NIST AI 600-1. Gaithersburg, MD: NIST. DOI: 10.6028/NIST.AI.600-1

Bai, Y., Kadavath, S., Kundu, S., Askell, A., Kernion, J., Jones, A. et al. (2022) 'Constitutional AI: harmlessness from AI feedback', *arXiv preprint*, arXiv:2212.08073. Available at: https://arxiv.org/abs/2212.08073

Bryman, A. (2016) *Social Research Methods*. 5th edn. Oxford: Oxford University Press.

Bucher, M.J.J. and Martini, M. (2024) 'Fine-tuned "small" LLMs (still) significantly outperform zero-shot generative AI models in text classification', *arXiv preprint*, arXiv:2406.08660. Available at: https://arxiv.org/abs/2406.08660

Chen, Z., Kang, M. and Li, B. (2025) 'ShieldAgent: shielding agents via verifiable safety policy reasoning', in *Proceedings of ICML 2025*, PMLR 267, pp. 8313–8344. Available at: https://proceedings.mlr.press/v267/chen25ae.html

Clark, K., Luong, M.T., Le, Q.V. and Manning, C.D. (2020) 'ELECTRA: pre-training text encoders as discriminators rather than generators', in *Proceedings of ICLR 2020*. Available at: https://openreview.net/forum?id=r1xMH1BtvB

de Chezelles, T.L.S., Gasse, M., Lacoste, A., Drouin, A., Caccia, M. et al. (2025) 'The BrowserGym ecosystem for web agent research', *Transactions on Machine Learning Research*. arXiv:2412.05467. Available at: https://arxiv.org/abs/2412.05467

Deng, X., Gu, Y., Zheng, B., Chen, S., Stevens, S., Wang, B., Sun, H. and Su, Y. (2023) 'Mind2Web: towards a generalist agent for the web', in *NeurIPS 2023*, pp. 28091–28114. Available at: https://arxiv.org/abs/2306.06070

Devlin, J., Chang, M.W., Lee, K. and Toutanova, K. (2019) 'BERT: pre-training of deep bidirectional transformers for language understanding', in *Proceedings of NAACL-HLT 2019*, pp. 4171–4186. DOI: 10.18653/v1/N19-1423

Drouin, A., Gasse, M., Caccia, M., Laradji, I.H., Del Verme, M., Marty, T., Vazquez, D., Chapados, N. and Lacoste, A. (2024) 'WorkArena: how capable are web agents at solving common knowledge work tasks?', in *Proceedings of ICML 2024*, PMLR 235, pp. 11642–11662. Available at: https://proceedings.mlr.press/v235/drouin24a.html

Efron, B. and Tibshirani, R. (1993) *An Introduction to the Bootstrap*. New York: Chapman and Hall/CRC.

European Parliament and Council of the European Union (2024) *Regulation (EU) 2024/1689 — AI Act*. Official Journal of the EU. Available at: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32024R1689

Galke, L., Scherp, A., Diera, A., Karl, F., Lin, B.X., Khera, B., Meuser, T. and Singhal, T. (2025) 'Are we really making much progress in text classification? A comparative review', *arXiv preprint*, arXiv:2204.03954v6. Available at: https://arxiv.org/abs/2204.03954

Geirhos, R., Jacobsen, J.-H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M. and Wichmann, F.A. (2020) 'Shortcut learning in deep neural networks', *Nature Machine Intelligence*, 2(11), pp. 665–673. DOI: 10.1038/s42256-020-00257-z

Gartner (2025) 'Gartner predicts over 40 percent of agentic AI projects will be canceled by end of 2027', *Gartner Newsroom*, 25 June. Available at: https://www.gartner.com/en/newsroom/press-releases/2025-06-25-gartner-predicts-over-40-percent-of-agentic-ai-projects-will

Ghimire, A. (2025) 'AI-powered anomaly detection for AML compliance in US banking: enhancing accuracy and reducing false positives', *Global Trends in Science and Technology*, 1(1), pp. 95–120. DOI: 10.70445/gtst.1.1.2025.95-120

Gregor, S. and Hevner, A.R. (2013) 'Positioning and presenting design science research for maximum impact', *MIS Quarterly*, 37(2), pp. 337–355.

He, P., Liu, X., Gao, J. and Chen, W. (2021) 'DeBERTa: decoding-enhanced BERT with disentangled attention', in *Proceedings of ICLR 2021*. Available at: https://openreview.net/forum?id=XPZIaotutsD

He, P., Gao, J. and Chen, W. (2023) 'DeBERTaV3: improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing', in *Proceedings of ICLR 2023*. Available at: https://openreview.net/forum?id=sE7-XhLxHA

He, Z., Ribeiro, M.T. and Khani, F. (2023) 'Targeted data generation: finding and fixing model weaknesses', in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8506–8520. DOI: 10.18653/v1/2023.acl-long.474

Hevner, A.R., March, S.T., Park, J. and Ram, S. (2004) 'Design science in information systems research', *MIS Quarterly*, 28(1), pp. 75–105. Available at: https://aisel.aisnet.org/misq/vol28/iss1/6/

International Organization for Standardization (2023) *ISO/IEC 42001:2023 Artificial Intelligence — Management System*. Geneva: ISO. Available at: https://www.iso.org/standard/42001

Jacobs, A.Z. and Wallach, H. (2021) 'Measurement and fairness', in *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pp. 375–385. DOI: 10.1145/3442188.3445901

Koh, J.Y., Lo, R., Jang, L., Duvvur, V., Lim, M.C., Huang, P.Y., Neubig, G., Zhou, S., Salakhutdinov, R. and Fried, D. (2024) 'VisualWebArena: evaluating multimodal agents on realistic visual web tasks', in *Proceedings of ACL 2024*, pp. 881–905. Available at: https://arxiv.org/abs/2401.13649

Levy, I., Wiesel, B., Marreed, S., Oved, A., Yaeli, A., Mashkif, N. and Shlomov, S. (2026) 'ST-WebAgentBench: a benchmark for evaluating safety and trustworthiness in web agents', in *Proceedings of ICLR 2026*. arXiv:2410.06703v6. Available at: https://arxiv.org/abs/2410.06703

Li, Z., Zhu, H., Lu, Z. and Yin, M. (2023) 'Synthetic data generation with large language models for text classification: potential and limitations', in *Findings of EMNLP 2023*, pp. 10443–10461. DOI: 10.18653/v1/2023.emnlp-main.647

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L. and Stoyanov, V. (2019) 'RoBERTa: a robustly optimized BERT pretraining approach', *arXiv preprint*, arXiv:1907.11692. Available at: https://arxiv.org/abs/1907.11692

Loshchilov, I. and Hutter, F. (2019) 'Decoupled weight decay regularization', in *Proceedings of ICLR 2019*. Available at: https://openreview.net/forum?id=Bkg6RiCqY7

Luo, W., Dai, S., Liu, X., Banerjee, S., Sun, H., Chen, M. and Xiao, C. (2025) 'AGrail: a lifelong agent guardrail with effective and adaptive safety detection', in *Proceedings of ACL 2025*, pp. 8104–8139. DOI: 10.18653/v1/2025.acl-long.399

March, S.T. and Smith, G.F. (1995) 'Design and natural science research on information technology', *Decision Support Systems*, 15(4), pp. 251–266.

McKinsey & Company (2025) *The State of AI in 2025: Agents, Innovation, and Transformation*. Available at: https://www.mckinsey.com/capabilities/quantumblack/our-insights/the-state-of-ai

McNemar, Q. (1947) 'Note on the sampling error of the difference between correlated proportions or percentages', *Psychometrika*, 12(2), pp. 153–157. DOI: 10.1007/BF02295996

Movva, R., Koh, P.W. and Pierson, E. (2024) 'Annotation alignment: comparing LLM and human annotations of conversational safety', in *Proceedings of EMNLP 2024*, pp. 9048–9062. DOI: 10.18653/v1/2024.emnlp-main.511

Nadas, M., Aziz, R., Bickley, S.J. and Chan, H.F. (2025) 'Synthetic data generation using large language models: advances in text and code', *IEEE Access*. DOI: 10.1109/ACCESS.2025.3589503

National Institute of Standards and Technology (2023) *Artificial Intelligence Risk Management Framework (AI RMF 1.0)*. NIST AI 100-1. Gaithersburg, MD: U.S. Department of Commerce. Available at: https://nvlpubs.nist.gov/nistpubs/ai/nist.ai.100-1.pdf

OWASP Foundation (2025) *OWASP Top 10 for Large Language Model Applications 2025*. Available at: https://genai.owasp.org/llm-top-10/

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P. et al. (2022) 'Training language models to follow instructions with human feedback', in *NeurIPS 2022*, pp. 27730–27744. Available at: https://arxiv.org/abs/2203.02155

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J. and Chintala, S. (2019) 'PyTorch: an imperative style, high-performance deep learning library', in *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*, pp. 8024–8035. Available at: https://arxiv.org/abs/1912.01703

Peffers, K., Tuunanen, T., Rothenberger, M.A. and Chatterjee, S. (2007) 'A design science research methodology for information systems research', *Journal of Management Information Systems*, 24(3), pp. 45–77. DOI: 10.2753/MIS0742-1222240302

Saito, T. and Rehmsmeier, M. (2015) 'The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets', *PLOS ONE*, 10(3). DOI: 10.1371/journal.pone.0118432

Suresh, H. and Guttag, J.V. (2021) 'A framework for understanding sources of harm throughout the machine learning life cycle', in *Proceedings of the 2021 ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization*, pp. 1–9. DOI: 10.1145/3465416.3483305

Saunders, M.N.K., Lewis, P. and Thornhill, A. (2019) *Research Methods for Business Students*. 8th edn. Harlow: Pearson. Available at: https://www.pearson.com/en-gb/subject-catalog/p/research-methods-for-business-students/P200000005845

Venable, J., Pries-Heje, J. and Baskerville, R. (2016) 'FEDS: a framework for evaluation in design science research', *European Journal of Information Systems*, 25(1), pp. 77–89. DOI: 10.1057/ejis.2015.21

Warner, B., Chaffin, A., Clavié, B. et al. (2024) 'Smarter, better, faster, longer: a modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference', *arXiv preprint*, arXiv:2412.13663. Available at: https://arxiv.org/abs/2412.13663

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B. and Wesslén, A. (2012) *Experimentation in Software Engineering*. Berlin: Springer. DOI: 10.1007/978-3-642-29044-2

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q. and Rush, A. (2020) 'Transformers: state-of-the-art natural language processing', in *Proceedings of EMNLP 2020: System Demonstrations*, pp. 38–45. DOI: 10.18653/v1/2020.emnlp-demos.6

Xiang, Z., Zheng, L., Li, Y., Hong, J., Li, Q., Xie, H., Zhang, J. et al. (2024) 'GuardAgent: safeguard LLM agents by a guard agent via knowledge-enabled reasoning', *arXiv preprint*, arXiv:2406.09187. Available at: https://arxiv.org/abs/2406.09187

Xue, T., Qi, W., Shi, T., Song, C.H., Gou, B., Song, D., Sun, H. and Su, Y. (2025) 'An illusion of progress? Assessing the current state of web agents', in *Proceedings of COLM 2025*. arXiv:2504.01382. Available at: https://arxiv.org/abs/2504.01382

Yang, K., Liu, Y., Chaudhary, S., Fakoor, R., Chaudhari, P., Karypis, G. and Rangwala, H. (2025) 'AgentOccam: a simple yet strong baseline for LLM-based web agents', in *Proceedings of ICLR 2025*. Available at: https://arxiv.org/abs/2410.13825

Zhang, Z., Cui, S., Lu, Y., Zhou, J., Yang, J., Wang, H. and Huang, M. (2024) 'Agent-SafetyBench: evaluating the safety of LLM agents', *arXiv preprint*, arXiv:2412.14470. Available at: https://arxiv.org/abs/2412.14470

Zheng, B., Gou, B., Kil, J., Sun, H. and Su, Y. (2024) 'GPT-4V(ision) is a generalist web agent, if grounded', in *Proceedings of ICML 2024*, PMLR 235, pp. 61349–61385. Available at: https://proceedings.mlr.press/v235/zheng24e.html

Zheng, A., Rana, M. and Stolcke, A. (2025) 'Lightweight safety guardrails using fine-tuned BERT embeddings', in *Proceedings of COLING 2025 Industry Track*, pp. 689–696. Available at: https://aclanthology.org/2025.coling-industry.58/

Zhou, S., Xu, F.F., Zhu, H., Zhou, X., Lo, R., Sridhar, A., Cheng, X., Bisk, Y., Fried, D., Alon, U. and Neubig, G. (2024) 'WebArena: a realistic web environment for building autonomous agents', in *Proceedings of ICLR 2024*. Available at: https://openreview.net/forum?id=oKn9c6ytLx

Zwerdling, N., Boaz, D., Rabinovich, E., Uziel, G., Amid, D. and Anaby-Tavor, A. (2025) 'Towards enforcing company policy adherence in agentic workflows', in *Proceedings of EMNLP 2025 Industry Track*, pp. 595–606. arXiv:2507.16459. DOI: 10.18653/v1/2025.emnlp-industry.41