AutoRepairBench: Benchmarking Long-Tailed, Safety-Critical Automotive Repair Reasoning

Chenyu Zuo
Lingnan University
Tuen Mun, Hong Kong
chenyuzuo@ln.edu.hk

Xuecheng Zhou
Sun Yat-sen University
Zhuhai, China
zhouxch33@mail2.sysu.edu.cn

Yongqi Chen
Sun Yat-sen University
Shenzhen, China
chenyq387@mail2.sysu.edu.cn

Lei Xue
Sun Yat-sen University
Shenzhen, China
xuelei3@mail.sysu.edu.cn

Shan Jiang
Sun Yat-sen University
Zhuhai, China
jiangsh73@mail.sysu.edu.cn

Jiaxing Shen✉
Lingnan University
Tuen Mun, Hong Kong
jiaxingshen@ln.edu.hk

Chengzu Dong
Lingnan University
Tuen Mun, Hong Kong
chengzudong@ln.edu.hk

Abstract

Vehicle diagnosis begins with structured signals—an Electronic Control Unit (ECU) identifier, a Diagnostic Trouble Code (DTC), and trigger conditions—but technicians must translate them into an accurate fault description and appropriate repair measures. Automotive repair reasoning is both long-tailed and safety-critical: rare ECU/DTC combinations provide few training examples, while associating a fault with the wrong control unit or recommending unnecessary repairs can mislead maintenance decisions. Benchmarking this capability is challenging because diagnostic resources are fragmented, general-purpose large language models (LLMs) often fail to follow repair constraints, and exact-match metrics may reject valid answers expressed differently. We introduce **AutoRepairBench**, a BMW-centered benchmark and evaluation framework for long-tailed, safety-critical automotive repair reasoning. It supports single-turn diagnosis, multi-pair prompting, and frequency-aware analysis of common and rare cases. Its validated hierarchical evaluator accepts valid paraphrases while checking ECU/DTC associations, component relationships, repair severity, and unnecessary repair recommendations. We also develop **RepairLLM-7B**, a compact, retrieval-free model trained through two-stage distillation: the first stage reduces the dominance of frequent cases, while knowledge-graph-anchored online distillation emphasizes uncertain and rare cases. On a double-blind set of 585 cases labeled by three automotive experts, the evaluator achieved 94.19% accuracy and an F1 score of 94.67%. On the 500-instance main evaluation set, RepairLLM-7B achieved 95.0% single-turn joint accuracy (475/500) and 94.2% per-pair joint accuracy under multi-pair prompting (471/500); it also improved tail accuracy on the separate 100-instance generalization set and reduced safety-critical errors relative to supervised fine-tuning. These results demonstrate the importance of jointly benchmarking overall accuracy, rare-case performance, and repair safety in automotive repair reasoning.

CCS Concepts

• **Information systems** → **Evaluation of retrieval results**; • **Computing methodologies** → **Natural language processing**.

Keywords

automotive repair diagnosis, benchmarking, long-tail learning, large language models, knowledge distillation, safety-critical AI, evaluation

ACM Reference Format:
Chenyu Zuo, Xuecheng Zhou, Yongqi Chen, Lei Xue, Shan Jiang, Jiaxing Shen, and Chengzu Dong. 2026. AutoRepairBench: Benchmarking Long-Tailed, Safety-Critical Automotive Repair Reasoning. In *Proceedings of the 35th ACM International Conference on Information and Knowledge Management (CIKM '26), November 07–11, 2026, Rome, Italy*. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3799682.3840599

1 Introduction

Modern vehicles are increasingly software-defined systems. When a fault occurs, technicians usually do not observe the root cause directly; instead, they receive diagnostic signals such as an Electronic Control Unit (ECU) identifier, a Diagnostic Trouble Code (DTC), and trigger conditions. Turning this signal bundle into a useful maintenance decision requires two forms of reasoning: explaining the likely fault mechanism and selecting a repair plan whose scope matches the evidence. This setting is practically important because diagnostic support can reduce troubleshooting time, but it is also safety-sensitive: a fluent response that binds the fault to the wrong ECU, ignores a topology constraint, or recommends replacement before verification can create unnecessary cost and operational risk [27, 51].

Problem. We study structured automotive repair reasoning, where each input $x = (\text{ECUID}, \text{DTC}, \text{TriggerCondition})$ must be

CIKM '26, Rome, Italy

ACM ISBN 979-8-4007-2539-5/2026/11
https://doi.org/10.1145/3799682.3840599

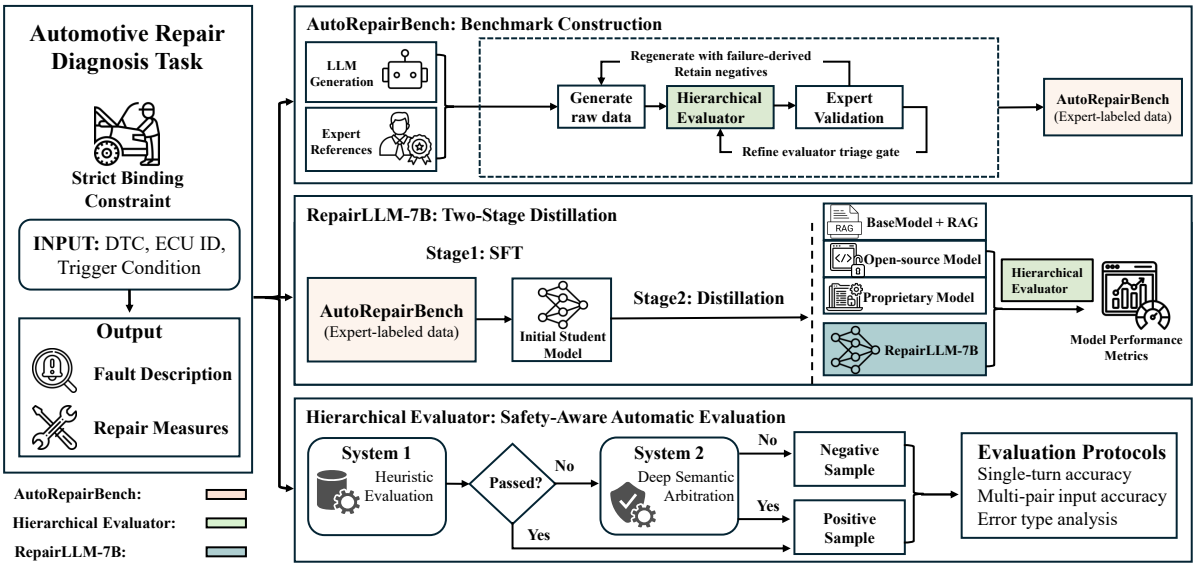

Figure 1: AutoRepairBench pipeline. From the ECU/DTC/trigger task, the stack proceeds as expert-annotated benchmark construction, two-stage distillation for RepairLLM-7B, and hierarchical System 1/System 2 evaluation under single-turn, multi-pair, and error-type protocols.

mapped to a fault description and actionable repair measures. The central difficulty is that the domain is long-tailed. Frequent faults provide many training examples, while rare ECU/DTC combinations may appear only once, even though they often contain the most consequential edge cases. At the same time, outputs are open-ended: different phrasings may be correct, but only if they preserve hard constraints such as ECU/DTC binding, physical directionality, and appropriate repair severity.

Limitations of existing methods. Current resources and methods do not fully address this combination of open-ended generation and hard technical constraints. Public diagnostic knowledge is fragmented across manuals, databases, and informal resources, making benchmark construction difficult. Standard long-tail classification techniques assume closed labels and therefore do not directly solve structured repair generation [5, 8]. Retrieval-augmented systems can provide useful evidence, but superficially similar cases may still differ in ECU binding, vehicle generation, or admissible repair scope. Finally, exact match and generic embedding similarity can reject valid paraphrases or accept unsafe near-matches, while generic LLM-as-a-judge methods may miss domain-specific safety rules [25, 55].

Our idea. We argue that the benchmark, model, and evaluator must be co-designed, following the order in Figure 1. **AutoRepairBench** first constructs expert-annotated instances from generated candidates and diagnostic references, with evaluator filtering in the loop and experts deciding gold labels. **RepairLLM-7B** is then trained as a retrieval-free 7B student with two-stage distillation, so it does not simply memorize frequent service templates. The same **Hierarchical Evaluator** (System 1 heuristic triage, then System 2 deliberative arbitration) scores open-ended outputs with paraphrase tolerance while enforcing ECU/DTC binding and no-over-repair constraints; independent expert validation keeps its scores from becoming a circular proxy for correctness.

Implementation challenges and solutions. The first challenge is data reliability: model-generated candidates improve coverage but may hallucinate entities or repairs; we address this with iterative generate–evaluate filtering followed by a full expert review of every retained instance, then refine the evaluator from those labels and retain invalid candidates as stress tests. The second challenge is tail robustness: naive supervised fine-tuning (SFT) is dominated by frequent patterns, whereas aggressive inverse-frequency sampling can overfit noisy rare cases; we use square-root rebalanced SFT followed by knowledge graph (KG)-anchored online distillation. The third challenge is scalable evaluation: repair outputs require paraphrase tolerance while enforcing no-over-repair and entity discipline; we introduce a hierarchical evaluator that combines fast semantic triage with deliberative tuple extraction and safety-aware arbitration.

Results and contributions. The evaluator agrees with expert labels at 94.19% accuracy on a double-blind validation set ($N = 585$). On the 500-instance evaluation set, RepairLLM-7B reaches 95.0% single-turn joint accuracy and improves multi-pair and safety-error performance relative to SFT and general-purpose baselines; tail accuracy also improves on the 100-instance generalization set. Our contributions are:

- **AutoRepairBench.** A structured repair-reasoning benchmark whose instances are expert-annotated in full, with protocols for single-turn, multi-pair, and frequency-aware head/tail evaluation.[1]

[1]Repository: https://github.com/AutoRepairBench/AutoRepairBench

- **RepairLLM-7B.** A retrieval-free 7B student trained with two-stage distillation, combining square-root rebalanced SFT and KG-anchored online distillation with focal-weighted PPO.
- **Hierarchical evaluator.** An evaluator that checks semantic correctness, ECU/DTC binding, topology, action severity, and no-over-repair constraints, validated against expert labels.

2 Related Work

Automotive diagnosis and industrial maintenance. Vehicle diagnosis has a long line of work that uses sensor traces, service records, and technical documents to support troubleshooting. Earlier systems mapped free-text customer or workshop reports to diagnostic categories, while recent work studies large-scale multilingual fault nowcasting from service text [20, 33]. Recent retrieval-based systems further move from classification toward generation: Mahale et al. combine DTC resources, manuals, and multi-step retrieval-augmented generation (RAG) to generate vehicle diagnostic reports, and Lukens et al. evaluate technical-manual RAG for ground-vehicle fault-code recommendation [28, 30]. AutoRepairBench is not intended as another diagnostic report generator. Instead, it benchmarks whether models can produce paired fault descriptions and repair measures under explicit ECU/DTC and trigger-condition grounding, where semantically variable surface forms must still satisfy entity binding, causal consistency, action severity, and no-over-repair constraints [6].

Industrial KG/LLM diagnostic reasoning. Beyond passenger-vehicle repair generation, LLM-based fault diagnosis is also emerging in prognostics and health management and industrial maintenance [9] and communication networks, where symbolic rules have been integrated with a teacher–student GNN to improve fault-scenario identification under limited data [52]. Prior work integrates maintenance logs, technical documents, and domain knowledge into diagnostic models, and several systems combine knowledge graphs with LLMs for aviation assembly, manufacturing, and maintenance decision support [17, 29]. We do not claim novelty for KG-enhanced diagnosis per se. Instead, we focus on its instantiation as an ECU/DTC-grounded automotive benchmark with frequency-aware evaluation, compact repair-model training, and KG-grounded safety rewards.

Long-tailed learning beyond closed labels. Classical long-tailed learning typically assumes a closed label space and mitigates skew through class reweighting or margin adjustment [5, 8]. For LLM-based repair generation, however, tail risk is also knowledge-driven: rare facts are less reliably stored in parametric memory, and factual QA accuracy correlates with the amount of relevant pretraining evidence [19, 43, 53]. Head/torso/tail factual benchmarks further show that popularity affects LLM factuality [40]. Automotive repair has a similar but more constrained long-tail structure: many ECU/DTC/trigger combinations are rare, yet valid outputs must still follow reusable repair templates, component relations, and safety rules [2]. AutoRepairBench therefore reports head/tail slices and trains RepairLLM-7B with tail-aware sampling and rewards, rather than reducing repair generation to a closed-label imbalance problem.

Table 1: AutoRepairBench task contract. The benchmark evaluates both explanatory correctness (FaultDescription) and prescriptive reliability (RepairMeasures) under ECU/DTC grounding.

Field	Definition
Input (x)	ECU ID, DTC, and Trigger Condition (schema keys: ECUID, DTC, TriggerCondition), which provide the diagnostic context.
Output (y)	FaultDescription and RepairMeasures.
FaultDescription	A grounded explanation of the malfunction and its underlying cause, consistent with the input context. Semantically equivalent expressions are accepted, whereas incorrect entities or contradictory claims are penalized.
RepairMeasures	An actionable, appropriately scoped repair plan that may include inspection, verification, or replacement when warranted. Unnecessary replacement and unsafe escalation are penalized.

Knowledge distillation for compact deployment. Distillation for LLMs has moved beyond matching final labels, using rationales, intermediate explanations, and reasoning traces to transfer behavior from larger teachers to compact students [10, 12, 45]. Alignment work such as Direct Preference Optimization further shows that feedback or preference objectives can reshape model behavior after supervised tuning [35]. We position this two-stage distillation in this lineage rather than as a generic distillation framework: its first stage rebalances rare ECU/DTC repair contexts during supervised tuning, whereas its second stage uses KG-grounded feedback and focal safety rewards to penalize fluent but unsupported, unsafe, or over-aggressive repair suggestions [41].

Evaluation for open-ended technical generation. Open-ended technical generation cannot be scored by exact match alone: semantically equivalent repairs may use different wording, whereas a single ECU, DTC, or component error can invalidate an otherwise fluent answer [22, 23]. Domain-specific evaluators use syntax-aware metrics for network-configuration generation [24] and combine LLM judgments with similarity measures validated against human preferences [14]; generic judges nevertheless remain vulnerable to position, verbosity, and instruction-following biases [25, 49, 55]. Unlike generic judges, our evaluator enforces explicit checks on ECU/DTC binding, topology and causal direction, action severity, and no-over-repair behavior, and its scores are calibrated against expert labels.

3 AutoRepairBench: Benchmark Construction and Protocols

Figure 1 overviews benchmark construction, two-stage training, and evaluation. This section defines the task, describes how AutoRepairBench is constructed from heterogeneous resources, and specifies the evaluation protocols used throughout the paper.

3.1 Task definition and output contract

As shown in Table 1, AutoRepairBench frames automotive repair diagnosis as structured generation. Each instance provides an input $x = (\text{ECUID}, \text{DTC}, \text{TriggerCondition})$ and requires the model

to generate y = (FaultDescription, RepairMeasures). We write the human-readable fields ECU ID and Trigger Condition as the schema keys ECUID and TriggerCondition, respectively. **FaultDescription** explains the malfunction and plausible causes consistent with the given identifiers and trigger.
RepairMeasures specifies feasible actions (inspection steps, replacements, or verification procedures) under safety constraints. This schema enables evaluation that is simultaneously semantic (meaning-preserving) and entity-aware (correct ECU/DTC binding) [3].

3.2 Data sources

We build AutoRepairBench from two complementary sources. **Model-driven generation** uses LLMs to propose candidate fault descriptions and repair measures from diagnostic contexts [46]; this improves coverage but can hallucinate entities or unsafe repairs. **Structured diagnostic references** supply normalized fault descriptions, trigger conditions, and service-plan fields from publicly accessible BMW fault-code material [4]. A November 2025 BMWFaultCodes snapshot was used for ECU/DTC-level normalization and annotation. We release only derived tuples and expert annotations, not original source content; raw-source reuse rights must be arranged directly with the source.

3.3 Iterative generate–evaluate extraction and negative samples

For model-generated candidates, we apply an iterative generation–evaluation loop: a generator proposes candidates for FaultDescription and RepairMeasures, and an evaluator checks (i) consistency with (ECUID, DTC, TriggerCondition), (ii) feasibility of repair actions, and (iii) basic safety constraints. Failed candidates are regenerated with constraints derived from prior failures, discouraging repetition and guiding the generator toward coherent outputs [38], until the loop terminates with a candidate instance. The resulting candidates are then passed to experts for a full valid/invalid review, with evaluator scores withheld (Section 3.4). **Importantly, we retain invalid instances** as informative negatives for evaluator refinement and stress testing, since most real-world failure cases resemble "almost plausible" technical text.

3.4 Validation, splits, and benchmark release

Benchmark construction uses the evaluator as a first-pass filter, then a full expert review. After extraction and automatic generate–evaluate scoring, every retained instance is labeled **valid** or **invalid** by a domain expert. Five domain experts partitioned the corpus, so each instance is reviewed by a single annotator rather than by independent multi-annotator voting. Annotators saw the diagnostic input and the candidate output, but were blinded to the evaluator score and decision. Expert-accepted items are the gold positives for training and evaluation; expert-rejected items are kept as negatives for evaluator refinement and stress testing. These expert decisions are also used to refine the evaluator, and automatic evaluator acceptance is not treated as a final gold label.

This corpus protocol is distinct from the evaluator-reliability study in Section 5.3. The latter is a disjoint double-blind sample (N = 585) in which three senior experts independently label each case and a majority vote is taken. The full-corpus labels do not use that three-annotator protocol, and we do not report inter-annotator agreement on the partitioned split. The reliability set is kept disjoint from all model-training and model-selection data.

Table 2: Long-tail distribution of unique RepairMeasures (RM) and FaultDescription (FD) strings in AutoRepairBench.

Metric	RepairMeasures	FaultDescription
Unique strings	21,647	85,889
Maximum frequency	9,669	185
Mean frequency	5.4	1.4
Median frequency	2.0	1.0
Strings with frequency 1	10,521 (48.6%)	67,921 (79.1%)
80% Coverage	5,348 (24.7%)	62,396 (72.6%)
90% Coverage	10,514 (48.6%)	74,143 (86.3%)

AutoRepairBench is released with standard splits for supervised learning: training (N_{train} = 117,365), validation (N_{val} = 873), and a held-out *generalization set* (N_{test} = 100). We additionally provide frequency metadata to support head/tail analysis, as summarized in Table 2.

The experiments use two evaluation pools, kept disjoint from the supervised validation split and the evaluator reliability set. The *main evaluation set* contains 500 instances (150 head, 350 tail) and is used for single-turn, multi-pair, and error-type comparisons. The 100-instance generalization set (30 head, 70 tail) is used for overall generalization and head/tail reporting. Instance frequency is the number of corpus records sharing the same `RepairMeasures` string; head instances have frequency $f \geq 101$ and tail instances $f \leq 100$, a cutoff induced by the square-root downsampling rule $\texttt{target} = \min(n, 10\sqrt{n})$. The 500-instance set is held out from the full corpus with fixed seed 1041 on the canonical key `{Input, RepairMeasures}`; the generalization set is sampled with fixed seed 1043.

3.5 Evaluation protocols (beyond joint accuracy)

AutoRepairBench evaluates models under protocols designed to reflect workshop workflows and long-tail reliability:

- **Single-turn accuracy:** independent diagnostic queries. We report joint accuracy, the fraction of instances for which both fields pass, and field-level correctness for FaultDescription and RepairMeasures.
- **Multi-pair completion:** each prompt contains 10 diagnostic pairs. We report per-pair joint accuracy across all multi-pair prompts; a pair passes only when both FaultDescription and RepairMeasures pass.
- **Frequency-aware evaluation:** stratified reporting for high-frequency (head) vs. low-frequency (tail) subsets, revealing long-tail brittleness that joint accuracy can hide.

These protocols make specific failure modes explicit, particularly *tail reliability* (rare faults) and robustness under batched heterogeneous inputs.

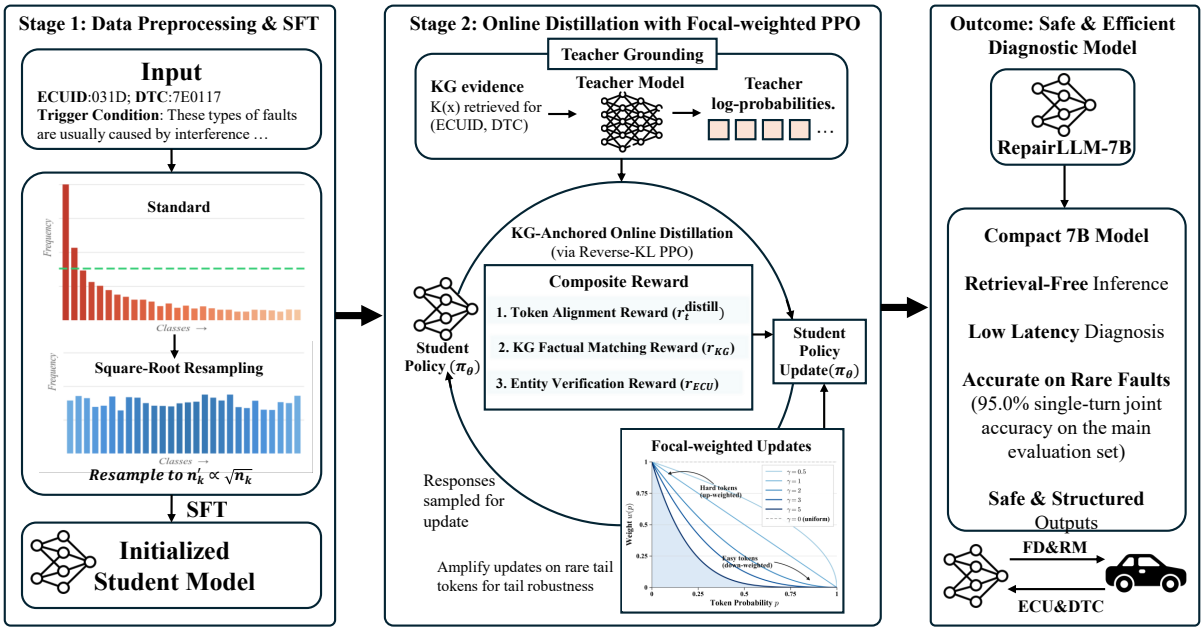

Figure 2: Two-stage distillation. Stage 1 performs square-root rebalanced SFT. Stage 2 performs KG-anchored online distillation with PPO and focal weighting.

4 RepairLLM-7B: Two-Stage Distillation for Long-Tailed Repair Reasoning

Figure 2 summarizes two-stage distillation for RepairLLM-7B, a compact, retrieval-free student for safety-critical automotive repair reasoning, where inputs are highly diverse and long-tailed but valid outputs must remain both semantically correct and operationally safe. Naive SFT tends to be dominated by frequent fault patterns, whereas unconstrained knowledge distillation may transfer fluent but unsafe teacher behaviors. Two-stage distillation addresses this by moving the student from *format-stable supervised generation* to *knowledge-grounded policy alignment*.

Using the same input–output contract as Section 3, training takes $\mathbf{x}$ = (ECUID, DTC, TriggerCondition) as input and produces $\mathbf{y}$ = (FaultDescription, RepairMeasures) as output. It consists of two stages:

(1) **Stage 1:** square-root rebalanced SFT, which initializes a stable student under long-tail data skew;
(2) **Stage 2:** KG-anchored online distillation with focal-weighted proximal policy optimization (PPO), which transfers teacher preferences while enforcing factual and entity-level constraints.

4.1 Motivation: Distillation for Long-Tailed Structured Generation

Automotive repair diagnosis differs from standard long-tail classification [43]: the challenge is not merely to recover rare labels, but to generate *structured, safety-constrained reasoning* for rare ECU/DTC conditions, where multiple surface realizations may be semantically acceptable but only a small subset is consistent with ECU grounding, fault topology, and repair scope. Retrieval-based assistance is also insufficient: semantically adjacent reference cases can still differ in ECU binding, topology, or admissible scope, so nearest-neighbor retrieval does not necessarily recover procedurally valid evidence. We therefore aim to internalize these constraints through knowledge-grounded distillation and tail-aware optimization, following three design principles: the student should not be initialized from head-dominated supervision alone; the teacher policy should be *knowledge-constrained* rather than purely generative; and the distillation signal should emphasize uncertain or rare patterns rather than already well-learned head tokens. Two-stage training operationalizes these principles.

4.2 Stage 1: Square-Root Rebalanced Supervised Fine-Tuning

We first train an initial student policy $\pi_\theta^{(0)}$ using supervised fine-tuning on the validated benchmark. The goal of this stage is not only to learn task semantics, but also to establish *format stability* and *basic diagnostic faithfulness* before online distillation.

Let n_k denote the empirical frequency of pattern group k in the training set.[2] Direct sampling from the raw distribution overexposes the model to head cases, while aggressive inverse-frequency balancing can overfit noisy tail samples. We therefore adopt a square-root resampling strategy:

$$n'_k \propto \sqrt{n_k}, \tag{1}$$

with upper and lower caps for stability.

[2]In practice, the grouping can be defined over diagnostic output patterns or equivalent long-tail strata derived from the repair corpus.

Compared with raw empirical sampling, Eq. (1) reduces head dominance while preserving enough exposure to frequent patterns to maintain output fluency and template consistency. Compared with strong tail oversampling, it is substantially more robust to annotation noise and rare-pattern variance. Empirically, this stage yields a student that reliably generates the required JSON-like structured output and acquires a reasonable prior over fault descriptions and repair actions, which makes subsequent policy optimization significantly more stable.

Formally, Stage 1 optimizes the standard autoregressive objective:

$$\mathcal{L}_{\text{SFT}} = -\sum_{t=1}^{T} \log \pi_\theta(y_t \mid x, y_{<t}), \tag{2}$$

where minibatches are sampled under the square-root rebalanced distribution.

4.3 Stage 2: KG-Anchored Online Distillation

After obtaining a stable student initializer, we perform online distillation from a stronger teacher model. A key difference from conventional distillation is that our teacher is *grounded* by structured domain knowledge rather than queried in an unconstrained manner.

4.3.1 Knowledge-Grounded Teacher Policy. For each input x, we retrieve curated diagnostic evidence

$$K(x) = \text{Retrieve}(\text{ECUID}, \text{DTC}), \tag{3}$$

from a diagnostic evidence graph built from normalized fault-code and service-plan fields. The retrieved evidence includes fault semantics, ECU-specific constraints, and admissible repair actions. The teacher therefore defines a grounded policy

$$\pi_T(y \mid x, K(x)), \tag{4}$$

instead of a free-form policy $\pi_T(y \mid x)$.

This distinction is critical in technical domains. An unconstrained teacher may produce linguistically plausible but operationally unsafe outputs, especially for rare cases with weak prior support. Conditioning on $K(x)$ narrows the teacher distribution toward evidence-supported modes and reduces, but does not eliminate, hallucinated entities, incorrect ECU associations, and over-scoped repair recommendations [1, 13]. In our implementation, $K(x)$ is instantiated as a structured case-library cache derived from validated records, where evidence retrieval is performed by exact (ECUID, DTC) lookup; the optional Neo4j backend serves only as a fallback query interface rather than a learned multi-hop graph pipeline, so we distill a *knowledge-filtered teacher policy*, not arbitrary teacher fluency.

4.3.2 Reverse-KL Motivated Policy Alignment. Rather than matching teacher logits offline, we optimize the student through online policy learning with PPO, using rewards derived from teacher preference and factual verification. The alignment target is reverse-KL motivated rather than a claim of exact closed-form KL minimization during PPO:

$$D_{\text{KL}}(\pi_\theta \,\|\, \pi_T). \tag{5}$$

Reverse-KL is particularly suitable here: multiple textual completions may be superficially plausible, but only a narrow subset satisfies ECU/DTC grounding and repair-scope constraints. Being *mode-seeking*, it concentrates probability mass on teacher-preferred safe modes instead of averaging across heterogeneous outputs, which is desirable under long-tail ambiguity, where broad imitation can amplify low-quality alternatives.

Table 3: Minimal reproducibility settings for RepairLLM-7B training.

Setting	Value
Base student checkpoint	`Qwen/Qwen2.5-7B-Instruct`
Teacher model/version	`Qwen/Qwen2.5-32B-Instruct-AWQ`
Training setup	SFT: AdamW, 2×10^{-5}, 3 epochs; PPO: AdamW, 5×10^{-7}, clip 0.2
Reward weights	$(\lambda_{\text{distill}}, \lambda_{\text{KG}}, \lambda_{\text{ECU}})$ $= (0.3, 5.0, 5.0)$
Max sequence length	SFT 1024; rollout 768
Hardware / stopping	3× RTX 4090-based GPUs (48GB custom-memory); 1500 PPO iterations

4.3.3 Multi-Signal Reward Fusion. For a sampled trajectory $y = (y_1, \ldots, y_T)$, we define a composite reward:

$$R(y, x) = \lambda_{\text{distill}} \sum_{t=1}^{T} w_t \, r_t^{\text{distill}} \;+\; \lambda_{\text{KG}} \, r^{\text{KG}} \;+\; \lambda_{\text{ECU}} \, r^{\text{ECU}}, \tag{6}$$

with weights $(\lambda_{\text{distill}}, \lambda_{\text{KG}}, \lambda_{\text{ECU}})$ as in Table 3. The reward combines three complementary signals.

Token-level distillation reward. We encourage the student to align with the grounded teacher at each generation step:

$$r_t^{\text{distill}} = \log \pi_T(y_t \mid x, y_{<t}, K(x)) - \log \pi_\theta(y_t \mid x, y_{<t}). \tag{7}$$

This term provides a dense token-level signal, pushing the student toward teacher-supported continuations while still allowing deviation when the student assigns high confidence to a compatible alternative.

KG factual matching reward. Teacher alignment alone does not guarantee factual correctness. We therefore add an external grounding reward r^{KG} that checks whether the generated fault-description and repair-measure fields are semantically consistent with the retrieved evidence $K(x)$. We instantiate this check with normalized entity matching and structured consistency checks over diagnostic content. This term rewards factual agreement with evidence rather than stylistic similarity alone.

ECU entity verification reward. In safety-critical repair reasoning, entity correctness is non-negotiable. We therefore include an ECU verification term r^{ECU} that explicitly checks whether the predicted explanation and action remain bound to the correct ECU context. This term suppresses a common failure mode in compact models: producing locally plausible descriptions that refer to the wrong component, module, or control unit.

These three signals play different roles. The distillation term transfers teacher preference; the KG reward enforces factual grounding; the ECU reward imposes hard entity discipline. Their combination makes the reward substantially more faithful to the true task objective than pure imitation alone.

4.4 Focal-Weighted PPO for Tail Robustness

A central challenge in long-tail repair reasoning is that rare faults are often expressed through rare but decisive lexical patterns, such as uncommon component names, topology descriptors, trigger conditions, or ECU-specific terminology. Under standard policy optimization, these low-probability tokens may contribute too little gradient signal relative to high-frequency head tokens.

To address this, we adapt the focal-loss intuition [26] as a modulation factor on token-level policy updates:

$$w_t = (1 - p_t)^{\gamma}, \qquad \gamma > 0, \tag{8}$$

where

$$p_t = \pi_\theta(y_t \mid x, y_{<t}) \tag{9}$$

is the student probability assigned to the sampled token.

Eq. (8) amplifies learning on uncertain predictions: tokens that already have high probability receive smaller weights, while rare or weakly modeled tokens receive larger updates. In our setting, this serves as a practical mechanism for *tail-sensitive policy correction*. Because many rare faults depend on precisely those uncommon technical tokens that the student is least confident about, focal weighting reallocates optimization effort toward under-learned long-tail patterns without discarding the overall PPO training dynamics. Unlike conventional focal loss in closed-set recognition, the weighting is applied within *online sequence-level policy optimization*, strengthening rare reasoning fragments rather than reweighting classes.

4.5 Overall Optimization

Starting from the Stage-1 initializer $\pi_\theta^{(0)}$, we optimize the student policy with PPO under the reward in Eq. (6) [37, 54]: at each iteration, the student samples structured responses for input x, the teacher provides grounded token probabilities under $K(x)$, and the reward model computes factual and entity-level verification signals; PPO then updates the student toward higher-reward trajectories while preserving training stability through clipped policy updates. Implementation details are in the code release.

The resulting student, RepairLLM-7B, is a compact retrieval-free model that internalizes diagnostic knowledge during training and therefore avoids the runtime fragility of retrieval-dependent systems. Relative to standard SFT, it improves not only joint accuracy but also reliability on rare and safety-critical repair cases, combining long-tail rebalanced initialization, knowledge-constrained teacher supervision, and tail-amplified policy optimization [15].

4.6 Training Configuration

Table 3 lists the fixed RepairLLM-7B settings used for all reported results.

5 Safety-Aware Automatic Evaluation

AutoRepairBench requires evaluation that is semantic, entity-aware, and safety-aware. This section presents the hierarchical evaluator and reports its reliability against expert labels.

5.1 Evaluator overview

As shown in Figures 1 and 3, the evaluator is a dual-process design inspired by coarse-to-fine cognition [18]. **System 1** performs fast heuristic triage for clear cases, analogous to a general agent; **System 2** performs deliberative arbitration for ambiguous or high-risk cases, analogous to an expert agent [16, 47]. This design balances throughput with technical safety and is used both for data filtering (Section 3) and model evaluation (Section 6) [50].

Table 4: Calibration of embedding models for diagnostic matching. The ranges are calibration-set similarity ranges; relation-level calibration details are provided in the artifact.

Model	Sim_{cal}	Observed strength	Role
e5-mistral-7b-instruct [42]	**0.84–0.88**	**Best relation consistency in calibration**	**Default**
mxbai-embed-large [21]	0.77–0.82	Strong structured technical matching	Reference
BAAI/bge-m3 [7]	0.86–0.88	High similarity; less stable on relation ambiguity	Candidate
all-mpnet-base-v2 [39]	0.83–0.84	General semantic baseline	Baseline
all-MiniLM-L6-v2 [44]	0.79–0.82	Fast but less discriminative	Triage

5.2 System 1 and System 2

System 1 is a conservative triage layer. It combines semantic similarity, hard-entity checks, directionality checks, and an automatic technical score. Cases that clearly satisfy both semantic and hard-constraint checks are accepted; low-confidence or conflicting cases are routed to System 2 rather than being decided by similarity alone.

5.2.1 Embedding model selection and parallel metrics. The parallel metric suite uses sentence-level semantic similarity [36], code/parts verification, and an automatic technical scoring prompt. During calibration, we compare candidate embedding backbones with a diagnostic matching score

$$C_m = \alpha \cdot \text{Sim}_{\text{cal}} + \beta \cdot \mathcal{R}_{\text{logic}}, \tag{10}$$

where Sim_{cal} is calibration-set semantic similarity, $\mathcal{R}_{\text{logic}}$ measures agreement on diagnostic relation checks, and we set $\alpha = 0.6$ and $\beta = 0.4$ in the current implementation. Table 4 summarizes the calibration ranges. We use `e5-mistral-7b-instruct` as the default semantic backbone because it provided the best trade-off between semantic matching and relation-level consistency in our calibration runs.

The triage gate uses mode-specific thresholds because fault descriptions and repair measures have different risk profiles. The four outcomes of Gate in Eq. 11 are the canonical decision labels used throughout the evaluator and in Algorithm 1 (corresponding to the accept, rescue, and reject paths of Figure 3).

$$\textsc{Gate}(Y_{\text{pred}}) = \begin{cases} \textsc{Accept}_1, & S_{\text{score}} \geq 98 \wedge S_{\text{sim}} \geq \tau \wedge \mathcal{V}_{\text{dir}} \\ \textsc{Accept}_2, & 90 \leq S_{\text{score}} \leq 97 \wedge \mathcal{H}_{\text{match}} \wedge \mathcal{V}_{\text{dir}} \\ \textsc{Rescue}, & 60 \leq S_{\text{score}} < 90 \text{ or } \neg\mathcal{H}_{\text{match}} \\ \textsc{Reject}, & S_{\text{score}} < 60 \text{ or } \mathcal{F}_{\text{conflict}}. \end{cases} \tag{11}$$

Here, S_{score} is an automatic technical score on a 0–100 scale, S_{sim} is the e5-mistral semantic similarity, $\mathcal{H}_{\text{match}}$ checks hard identifiers such as DTCs and module aliases, $\mathcal{V}_{\text{dir}}$ checks directional or topological terms such as bank and cylinder, and $\mathcal{F}_{\text{conflict}}$ flags unsafe scope expansion or entity conflicts. For repair measures, the gate applies stricter safety checks because action severity matters more

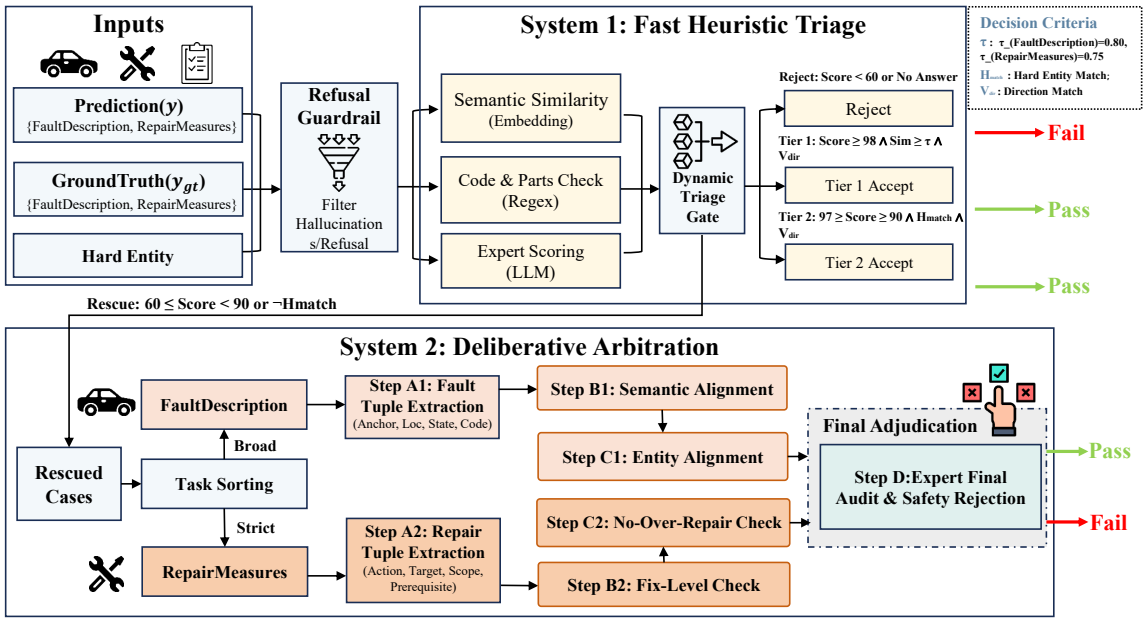

Figure 3: Hierarchical evaluator. System 1 performs conservative triage; uncertain cases are routed to System 2 for arbitration.

than phrasing; for fault descriptions, it allows broader paraphrase variation while still enforcing entity constraints.

A refusal guard catches empty, generic, or refusal-like responses before scoring. For reproducibility, the implementation logs score-parser fallbacks, timeout retries, and model responses used by the automatic scoring prompt rather than silently dropping these cases.

System 2 performs field-specific arbitration. For FaultDescription, it extracts an (anchor, location, state, code) tuple to compare technical meaning under terminology variation:

$$T_{\text{fault}} = \{\text{Anchor, Loc, State, Code}\}. \tag{12}$$

For RepairMeasures, it extracts a compact action tuple:

$$T_{\text{repair}} = \{\text{Action, Target, Scope, Prerequisite}\}. \tag{13}$$

The repair tuple is intentionally action-severity aware. When a prediction contains several verbs, such as "inspect wiring and replace the control unit," the extractor treats the most severe irreversible action as the primary action. This prevents a severe recommendation from being hidden behind a benign preliminary check.

System 2 then runs three checks. First, evidence search resolves controlled terminology and module aliases across vehicle generations or topological inclusions such as cylinder-to-bank. Second, the fix-level check distinguishes verification, repair, coding, and replacement. Third, the no-over-repair check rejects scope expansion when the reference only supports inspection or conditional replacement; narrowly defined equivalence cases (e.g., general wiring inspection covering connector inspection) are allowed but logged for audit. The final decision follows Algorithm 1.

5.3 Reliability validation

We validate the evaluator on a double-blind set of anonymized maintenance cases ($N = 585$), including 328 positive instances (valid logic) and 257 negative instances (hallucinations or unsafe recommendations). Table 5 reports agreement with the majority-vote expert gold. The counts are TP=302, TN=249, FP=8, and FN=26. The higher number of false negatives than false positives reflects a conservative bias appropriate for a safety-oriented benchmark [11].

Table 5: Evaluator agreement with expert labels on a double-blind set ($N = 585$). MCC is the Matthews correlation coefficient.

Metric	Value
Accuracy	94.19%
F1-score	94.67%
MCC	0.885
Cohen's κ	0.883

Annotation protocol. The double-blind validation set ($N = 585$) was sampled from the benchmark pool after excluding all examples used for model training, model selection, or evaluator threshold calibration. This three-annotator protocol applies only to the reliability sample; it is not the labeling procedure used for the full corpus in Section 3.4. Three senior domain experts with at least five years of professional automotive repair experience independently reviewed each case. Annotators were recruited from professional channels and did not participate in model development, data generation, or evaluator calibration, thereby minimizing conflicts of interest. Before annotation, they were instructed with a written guideline that defined the decision criteria for ECU/DTC binding, repair-scope appropriateness, topology consistency, and safety-constraint adherence. Annotators saw the input diagnostic context, the reference output, and the candidate prediction, but not the model identity or the evaluator score and decision. Final labels were determined by majority vote among the three annotators.

Algorithm 1 Evaluator arbitration logic

Require: Y_{gt}, Y_{pred}, E_{hard}, Mode
Ensure: Decision $D \in \{\textbf{PASS}, \textbf{FAIL}\}$
/* **System 1: conservative triage** */
if IsRefusal(Y_{pred}) **then**
  return FAIL
end if
$S_{\text{score}}, S_{\text{sim}}, \mathcal{H}_{\text{match}}, \mathcal{V}_{\text{dir}}, \mathcal{F}_{\text{conflict}} \leftarrow$ GetMetrics($Y_{\text{gt}}, Y_{\text{pred}}, E_{\text{hard}}$)
$G \leftarrow$ Gate($S_{\text{score}}, S_{\text{sim}}, \mathcal{H}_{\text{match}}, \mathcal{V}_{\text{dir}}, \mathcal{F}_{\text{conflict}}$)
if $G =$ Accept$_1$ **then**
  return PASS
else if $G =$ Accept$_2$ **then**
  return PASS
else if $G =$ Reject **then**
  return FAIL
else
  /* **Rescue: route to System 2** */
end if
/* **System 2: structured arbitration** */
Step A: $T_{\text{gt}}, T_{\text{pred}} \leftarrow$ TupleExtract($Y_{\text{gt}}, Y_{\text{pred}}$, Mode)
if IsEmpty(T_{gt}) **then**
  if Mode = `FaultDescription` **and** $S_{\text{score}} \geq 60$ **then**
    return PASS
  else
    return FAIL
  end if
end if
if Mode = `FaultDescription` **then**
  Step B1 & C1: Evidence Search & Semantic Alignment
else if Mode = `RepairMeasures` **then**
  Step B2 & C2: Evidence Search, Fix-Level Check & No-Over-Repair Check
end if
/* **Final adjudication** */
return Step D: Final Adjudication($T_{\text{gt}}, T_{\text{pred}}$, Mode)

Because per-annotator labels were not retained after the vote, annotator-level inter-annotator agreement is not reported; the Cohen's $\kappa = 0.883$ in Table 5 measures evaluator agreement with the majority-vote expert gold, not agreement between annotators.

6 Experiments

We evaluate three aspects of the proposed framework: (i) the reliability of the hierarchical evaluator, (ii) the stage-wise effect of two-stage distillation on the same 7B student backbone, and (iii) the resulting improvements in long-tail robustness and safety-critical error reduction. Together, these experiments answer three questions: **(Q1)** whether the evaluator is sufficiently aligned with expert judgment to support benchmark construction and scalable evaluation; **(Q2)** whether Stage 2 contributes substantial gains beyond SFT alone; and **(Q3)** whether these gains concentrate on long-tail cases, repair-action generation, and safety-critical failure modes.

6.1 Evaluator validation

Model scores use the hierarchical evaluator, calibrated separately from training and selection (Section 5.3).

6.2 Benchmarking setup

We benchmark against representative closed/API and general-purpose baselines: GPT-5, Grok-4-Fast, Llama-4, and Qwen3-Max [31, 32, 34, 48]. These results should be interpreted under the fixed prompt and no-domain-retrieval setting used in this paper, not as a universal ranking of the underlying model families. Because hosted models can change over time, the artifact records the provider-visible model name, access date, prompt, system message, temperature, top-p, maximum tokens, and retry policy for each run. The initial baseline runs were conducted on 2025-12-01 via OpenRouter's chat/completions API. Decoding used a temperature of 0.7, a 60-second timeout, and the API default max-tokens setting, with single-turn user-message prompting and no external internet access or tool use.

We additionally include a 7B RAG baseline to estimate how much of the task can be addressed by external nearest-neighbor evidence. Using `e5-mistral-7b-instruct` as the dense retriever, we retrieve the top-3 most similar reference instances for each query, concatenate them, and feed them to a 7B generator; we report this as a simple retrieval-dependent baseline rather than a tuned RAG upper bound.

As defined in Section 3, experiments use two evaluation pools. The *main evaluation set* (500 instances) supports the single-turn, multi-pair, and safety-critical error comparisons in Table 6. For multi-pair evaluation, we construct fixed batches of 10 diagnostic pairs from this pool, sampled according to the long-tail frequency distribution; the same batches are used for all models. The evaluator scores each of the 10 paired outputs separately, and we aggregate correctness over all 500 pairs. OA is joint accuracy: both FaultDescription (FD) and RepairMeasures (RM) must pass. The 95.0% single-turn OA for RepairLLM-7B corresponds to 475/500 joint successes (95% Wilson CI: 92.72–96.59), and the 94.2% multi-pair OA corresponds to 471/500 per-pair joint successes under batched prompting (95% Wilson CI: 91.79–95.93).

The *generalization set* ($N = 100$) is used for overall generalization and head/tail reporting in Table 6; RepairLLM-7B reaches 88.0% OA (88/100; 95% Wilson CI: 80.19–93.00). Unless otherwise stated, other results in this section are on the 500-instance set. Head/tail labels follow the corpus-level frequency metadata in Section 3.

Beyond joint accuracy, we analyze failures with four error types: *Action Severity Mismatch*, *Topology Error*, *Entity Hallucination*, and *Critical Omission*. Table 7 reports event-level counts over 1,000 output fields (500 fault descriptions and 500 repair measures); a single prediction may contribute multiple error events.

6.3 Stage-wise, tail, and baseline analysis

Table 6 reports a stage-wise ablation on the same 7B backbone and the 500-instance evaluation set. Adding Stage 2 KG-anchored online distillation to the Stage 1 SFT-sqrt student increases single-turn OA from 69.0% to 95.0%, multi-pair OA from 64.0% to 94.2%, and generalization-set OA from 41.0% to 88.0%. The largest gains appear on RepairMeasures, suggesting that two-stage distillation primarily improves prescriptive repair reasoning rather than only surface-level fault descriptions.

These gains extend to rare cases. On the generalization set, tail FD accuracy rises from 52.86% to 88.57% and tail RM accuracy from

Table 6: Main benchmark performance, robustness, and generalization evaluation (%). OA denotes overall accuracy; FD and RM denote FaultDescription and RepairMeasures. H/T indicate head/tail frequency subsets.

	Model	Main Benchmark						Robustness & Generalization						
		Single-turn			Multi-pair			Generalization (N=100)			Frequency-aware Evaluation			
		OA	FD	RM	OA	FD	RM	OA	FD	RM	FD-H	FD-T	RM-H	RM-T
Closed API	GPT-5	17.00	30.00	17.40	16.00	28.60	16.40	16.00	58.00	17.00	53.33	60.00	20.00	15.71
	Grok-4-Fast	7.00	34.60	8.20	6.80	33.00	8.40	7.00	52.00	8.00	53.33	51.43	3.33	10.00
General LLMs	Llama-4	5.60	32.40	7.20	5.00	31.00	6.00	2.00	44.00	4.00	40.00	45.71	3.33	4.29
	Qwen3-Max	9.20	36.80	9.80	8.40	35.40	8.60	5.60	56.00	6.00	56.67	55.71	6.67	5.71
Ours	SFT-standard	47.20	58.40	48.00	41.60	55.00	42.80	25.00	36.00	26.00	63.33	35.71	56.67	24.29
	SFT-sqrt	69.00	76.00	70.00	64.00	78.00	66.00	41.00	58.00	42.00	70.00	52.86	43.33	41.43
	w/o focal	84.20	87.60	84.80	80.20	85.20	81.00	65.00	74.00	66.00	**91.67**	77.14	**90.00**	73.57
	RepairLLM-7B	**95.00**	**95.20**	**95.00**	**94.20**	**95.00**	**94.60**	**88.00**	**89.00**	**88.00**	90.00	**88.57**	**90.00**	**87.14**
RAG	RAG (top-3 retrieval)	47.60	68.60	48.20	21.80	45.20	23.20	–	–	–	–	–	–	–

Table 7: Safety-critical error type analysis. Counts and percentages are event-level measurements over 1,000 evaluated output fields.

	Model	Error Type			
		Action Severity Mismatch	Topology Error	Entity Hallucination	Critical Omission
Closed	GPT-5	145 (14.5%)	254 (25.4%)	202 (20.2%)	172 (17.2%)
	Grok-4-Fast	131 (13.1%)	254 (25.4%)	265 (26.5%)	144 (14.4%)
General	Llama-4	143 (14.3%)	259 (25.9%)	176 (17.6%)	199 (19.9%)
	Qwen3-Max	131 (13.1%)	254 (25.4%)	226 (22.6%)	129 (12.9%)
Ours	SFT-sqrt	32 (3.2%)	55 (5.5%)	66 (6.6%)	27 (2.7%)
	RepairLLM-7B	**6 (0.6%)**	**10 (1.0%)**	**10 (1.0%)**	**4 (0.4%)**

41.43% to 87.14%. The head–tail gap for FD shrinks from 17.14% under SFT-sqrt to 1.43% after two-stage distillation, indicating reduced frequency sensitivity under this split.

The ablations support two conclusions. First, SFT-sqrt outperforms standard SFT, so long-tail-aware initialization matters. Second, removing focal weighting in Stage 2 lowers tail and held-out performance, suggesting that focal-weighted PPO helps emphasize rare but informative tokens. Table 6 does not isolate KG grounding or ECU verification, so we do not claim independent ablations for those terms.

Under our fixed no-retrieval prompt, general-purpose baselines struggle particularly with repair-action generation. The simple top-3 RAG baseline also underperforms RepairLLM-7B, suggesting that retrieved evidence alone is insufficient in this configuration.

6.4 Safety-critical error analysis

Table 7 decomposes failures into action severity mismatch, topology error, entity hallucination, and critical omission. Relative to SFT-sqrt, RepairLLM-7B reduces these counts from 32 to 6, 55 to 10, 66 to 10, and 27 to 4, respectively. These reductions matter because wrong entities, topology, or repair scope can be unsafe even when the generated text is fluent.

Under the fixed prompts and evaluator, RepairLLM-7B also yields fewer error events than all four general-purpose baselines across all categories, supporting improved safety-relevant constraint adherence within AutoRepairBench.

7 Limitations

AutoRepairBench is a research benchmark, not a replacement for certified technician judgment or original equipment manufacturer (OEM) service procedures. The current release is primarily BMW-centered: while the evaluation framework is reusable across OEMs, ECU/DTC ontologies, diagnostic mappings, and safety rules require OEM-specific reconstruction, and systematic cross-OEM validation remains future work. The baseline comparison uses fixed no-retrieval prompts and should therefore not be interpreted as a universal ranking against general-purpose LLMs. We also do not separately evaluate the Qwen2.5-32B-Instruct teacher or the zero-shot Qwen2.5-7B-Instruct base model, limiting direct analysis of teacher–student gaps. Finally, our results are primarily point estimates from the current runs without systematic multi-seed or repeated-call analysis; characterizing run-to-run variability remains future work.

8 Conclusion

We introduced AutoRepairBench, RepairLLM-7B, and a hierarchical evaluator for long-tailed, safety-aware automotive repair reasoning. Two-stage distillation improves single-turn, multi-pair, tail, and safety-error performance, while the evaluator provides scalable scoring aligned with expert labels. These findings support jointly developing benchmarks, long-tail-aware training, and domain-specific evaluation.

9 Acknowledgments

This work was supported in part by Lingnan University under Grant SDS24A17; the National Natural Science Foundation of China (NSFC) under Grant W2412110; the Hong Kong Research Grants Council (RGC) Theme-based Research Scheme under Grant T43-513/23-N; and the Guangdong Provincial Pearl River Talents Program under Grant 2024QN11X183.

GenAI Usage Disclosure

The authors used generative AI tools for language polishing, LaTeX formatting, consistency checks, and reviewer-style issue triage. In addition, as described in Section 3, LLMs were used within the benchmark-construction pipeline to generate candidate fault descriptions and repair measures before automatic filtering and expert validation. Final benchmark labels, data-release decisions, experimental results, tables, and figures were verified and approved by the human authors.

References

[1] Mohamed Ali, Zaki Taha, and Mohamed Mabrouk Morsey. 2026. Ontology-grounded knowledge graphs for mitigating hallucinations in large language models for clinical question answering. *Journal of Biomedical Informatics* 175 (2026), 104993. Publisher: Elsevier.

[2] Sanket Badhe, Deep Shah, and Nehal Kathrotia. 2026. Long-Tail Knowledge in Large Language Models: Taxonomy, Mechanisms, Interventions and Implications. *arXiv preprint arXiv:2602.16201* (2026).

[3] Mehmet Selman Baysan, Serkan Uysal, İrem İşlek, Çağla Çığ Karaman, and Tunga Güngör. 2025. LLM-as-a-Judge: automated evaluation of search query parsing using large language models. *Frontiers in big Data* 8 (2025), 1611389. Publisher: Frontiers Media SA.

[4] BMWFaultCodes. 2026. BMW Fault Code Lookup. https://bmwfault.codes/

[5] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. 2019. Learning imbalanced datasets with label-distribution-aware margin loss. *Advances in neural information processing systems* 32 (2019).

[6] Jiao Chen, Ruyi Huang, Zuohong Lv, Jianhua Tang, and Weihua Li. 2025. Faultgpt: Industrial fault diagnosis question answering system by vision language models. *arXiv preprint arXiv:2502.15481* (2025).

[7] Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In *Findings of the association for computational linguistics: ACL 2024*. 2318–2335.

[8] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. 2019. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 9268–9277.

[9] Vicente González-Prida, Carlos Parra Márquez, Pablo Viveros Gunckel, Fredy Kristjanpoller Rodríguez, and Adolfo Crespo Márquez. 2025. Digital Transformation in Aftersales and Warranty Management: A Review of Advanced Technologies in I4. 0. *Algorithms* 18, 4 (2025), 231. Publisher: MDPI.

[10] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. 2021. Knowledge distillation: A survey. *International journal of computer vision* 129, 6 (2021), 1789–1819. Publisher: Springer.

[11] Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, and others. 2024. A survey on llm-as-a-judge. *The Innovation* (2024). Publisher: Elsevier.

[12] Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2024. MiniLLM: On-Policy Distillation of Large Language Models. In *International Conference on Learning Representations*.

[13] Xinyan Guan, Yanjiang Liu, Hongyu Lin, Yaojie Lu, Ben He, Xianpei Han, and Le Sun. 2024. Mitigating large language model hallucinations via autonomous knowledge graph-based retrofitting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 18126–18134. Issue: 16.

[14] Beichen Guo, Zhiyuan Wen, Yu Yang, Peng Gao, Ruosong Yang, and Jiaxing Shen. 2025. SGSimEval: A Comprehensive Multifaceted and Similarity-Enhanced Benchmark for Automatic Survey Generation Systems. In *Proceedings of the 21st International Conference on Advanced Data Mining and Applications*. 393–407. doi:10.1007/978-981-95-3456-2_27

[15] Muyu He, Muhammad Ali Shafique, Anand Kumar, Tsach Mackey, and Nazneen Rajani. 2025. The Valley of Code Reasoning: Scaling Knowledge Distillation of Large Language Models. *arXiv preprint arXiv:2510.06101* (2025).

[16] Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2023. Large language models cannot self-correct reasoning yet. *arXiv preprint arXiv:2310.01798* (2023).

[17] Sagar Jose, Khanh TP Nguyen, Kamal Medjaher, Ryad Zemouri, Mélanie Lévesque, and Antoine Tahan. 2024. Advancing multimodal diagnostics: Integrating industrial textual data and domain knowledge with large language models. *Expert Systems with Applications* 255 (2024), 124603. Publisher: Elsevier.

[18] Daniel Kahneman. 2011. *Thinking, fast and slow*. macmillan.

[19] Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2023. Large language models struggle to learn long-tail knowledge. In *International conference on machine learning*. PMLR, 15696–15707.

[20] Ali Khodadadi, Soroush Ghandiparsi, and Chen-Nee Chuah. 2022. A natural language processing and deep learning based model for automated vehicle diagnostics using free-text customer service reports. *Machine Learning with Applications* 10 (2022), 100424. Publisher: Elsevier.

[21] Sean Lee, Aamir Shakir, Darius Koenig, and Julius Lipp. 2024. Open source strikes bread-new fluffy embeddings model. *URL https://www.mixedbread.ai/blog/mxbai-embed-large-v1* (2024).

[22] Scott H Lee. 2018. Natural language generation for electronic health records. *NPJ digital medicine* 1, 1 (2018), 63. Publisher: Nature Publishing Group UK London.

[23] Dawei Li, Bohan Jiang, Liangjie Huang, Alimohammad Beigi, Chengshuai Zhao, Zhen Tan, Amrita Bhattacharjee, Yuxuan Jiang, Canyu Chen, Tianhao Wu, and others. 2025. From generation to judgment: Opportunities and challenges of llm-as-a-judge. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. 2757–2791.

[24] Fuliang Li, Bocheng Liang, Haozhi Lang, Jiajie Zhang, Jiaxing Shen, Chengxi Gao, and Xingwei Wang. 2026. PreConfig: A Unified Language Model Framework for Network Configuration Automation. *IEEE Transactions on Cognitive Communications and Networking* 12 (2026), 6320–6330. doi:10.1109/TCCN.2026.3662678

[25] Haitao Li, Qian Dong, Junjie Chen, Huixue Su, Yujia Zhou, Qingyao Ai, Ziyi Ye, and Yiqun Liu. 2024. Llms-as-judges: a comprehensive survey on llm-based evaluation methods. *arXiv preprint arXiv:2412.05579* (2024).

[26] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*. 2980–2988.

[27] Weikun Lin and Kehua Miao. 2025. An Automotive Fault Diagnosis Framework Based on Knowledge Graphs and Large Language Models. *Electronics* 14, 21 (2025), 4180. Publisher: MDPI.

[28] Sarah Lukens, Matthew Bishof, Nadir Siddiqui, and Destiny West. 2025. An Evaluation Framework for Fault Diagnosis Using Technical Manuals in Retrieval-Augmented Large Language Models. In *Annual Conference of the PHM Society*, Vol. 17. Issue: 1.

[29] Yunfei Ma, Shuai Zheng, Zheng Yang, Hongcheng Pan, and Jun Hong. 2025. A knowledge-graph enhanced large language model-based fault diagnostic reasoning and maintenance decision support pipeline towards industry 5.0. *International Journal of Production Research* (2025), 1–22. Publisher: Taylor & Francis.

[30] Yashashree Mahale, Shrikrishna Kolhar, and Anjali S More. 2025. Automated vehicle fault diagnosis and report generation using hybrid machine learning with multi-step RAG approach. *Discover Computing* 28, 1 (2025), 283. Publisher: Springer.

[31] Meta AI. 2025. The Llama 4 herd: The beginning of a new era of natively multimodal AI innovation. https://ai.meta.com/blog/llama-4-multimodal-intelligence/

[32] OpenAI. 2025. Introducing GPT-5. https://openai.com/index/introducing-gpt-5/

[33] John Pavlopoulos, Alv Romell, Jacob Curman, Olof Steinert, Tony Lindgren, Markus Borg, and Korbinian Randl. 2024. Automotive fault nowcasting with machine learning and natural language processing. *Machine learning* 113, 2 (2024), 843–861. Publisher: Springer.

[34] Qwen Team. 2025. Qwen3-Max: Just Scale it. https://qwen.ai/blog?id=qwen3-max

[35] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems* 36 (2023), 53728–53741.

[36] Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*. 3982–3992.

[37] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[38] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems* 36 (2023), 8634–8652.

[39] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. MPNet: Masked and Permuted Pre-training for Language Understanding. *Advances in Neural Information Processing Systems* 33 (2020), 16857–16867.

[40] Kai Sun, Yifan Xu, Hanwen Zha, Yue Liu, and Xin Luna Dong. 2024. Head-to-tail: How knowledgeable are large language models (LLMs)? AKA will LLMs replace knowledge graphs?. In *Proceedings of the 2024 conference of the North American chapter of the association for computational linguistics: human language technologies (volume 1: long papers)*. 311–325.

[41] Khushboo Thaker and Yony Bresler. 2025. Knowledge Distillation with Structured Chain-of-Thought for Text-to-SQL. *arXiv preprint arXiv:2512.17053* (2025).

[42] Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Improving text embeddings with large language models. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: long papers)*. 11897–11916.

[43] Pengkun Wang, Zhe Zhao, HaiBin Wen, Fanfu Wang, Binwu Wang, Qingfu Zhang, and Yang Wang. 2024. Llm-autoda: Large language model-driven automatic data augmentation for long-tailed problems. *Advances in Neural Information Processing Systems* 37 (2024), 64915–64941.

[44] Wenhui Wang, Bin Bi, Ming Yan, Chen Wu, Zuyi Bao, Liwei Xia, Jingjing Peng, Luo Si, and Zhifang Wang. 2020. MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. In *Advances in Neural Information Processing Systems*, Vol. 33. 5776–5788.

[45] Xinyu Wang, Yong Jiang, Zhaohui Yan, Zixia Jia, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021. Structural knowledge distillation: Tractably distilling information for structured predictor. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 550–564.

[46] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*. 13484–13508.

[47] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems* 35 (2022), 24824–24837.

[48] xAI. 2025. Grok 4 Fast. https://x.ai/news/grok-4-fast

[49] Zhiyuan Zeng, Jiatong Yu, Tianyu Gao, Yu Meng, Tanya Goyal, and Danqi Chen. 2024. Evaluating large language models at evaluating instruction following. In *International Conference on Learning Representations*, Vol. 2024. 40193–40219.

[50] Qiyuan Zhang, Yufei Wang, Yuxin Jiang, Liangyou Li, Chuhan Wu, Yasheng Wang, Xin Jiang, Lifeng Shang, Ruiming Tang, Fuyuan Lyu, and others. 2025. Crowd comparative reasoning: Unlocking comprehensive evaluations for LLM-as-a-judge. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 5059–5074.

[51] Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, Longyue Wang, Anh Tuan Luu, Wei Bi, Freda Shi, and Shuming Shi. 2025. Siren's Song in the AI Ocean: A Survey on Hallucination in Large Language Models. *Computational Linguistics* 51, 4 (2025), 1373–1418.

[52] Haihong Zhao, Bo Yang, Jiaxu Cui, Qianli Xing, Jiaxing Shen, Fujin Zhu, and Jiannong Cao. 2024. Effective Fault Scenario Identification for Communication Networks via Knowledge-Enhanced Graph Neural Networks. *IEEE Transactions on Mobile Computing* 23, 4 (2024), 3243–3258. doi:10.1109/TMC.2023.3271715

[53] Qihao Zhao, Yalun Dai, Hao Li, Wei Hu, Fan Zhang, and Jun Liu. 2024. Ltgc: Long-tail recognition via leveraging llms-driven generated content. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 19510–19520.

[54] Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. 2026. Self-Distilled Reasoner: On-Policy Self-Distillation for Large Language Models. *arXiv preprint arXiv:2601.18734* (2026).

[55] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and others. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems* 36 (2023), 46595–46623.