

The ML FMEA in Action: Lessons from Applications of Machine Learning Safety

Neil Wadhvana
Machine Learning Group Lead
TORC Robotics

Bodo Seifert
Sr. Automotive Functional
Safety Engineer
TÜV Rheinland Group

Akshay Chalana
CEO and Co-Founder
Saphira AI

Justin Poh
Senior Safety Engineer
TORC Robotics

Majed Mohammed
Technical Fellow Functional
Safety
APTIV

Michael Wagner
Chief Safety Officer
Edge Case

Simon Diemert
VP Engineering
Critical Systems Labs

Fahim Mannan
Staff Machine Learning
Engineer
TORC Robotics

Jerry Lopez
Sr. Director Safety Assurance
TORC Robotics

Krzysztof Pennar
Principal Safety Engineer
TORC Robotics

Chaitanya Shinde
Staff Safety Engineer
TORC Robotics

David Ward
Global Head of Safety
HORIBA MIRA Ltd.

Paul Schmitt
Director of Engineering Safety
Reynolds & Moore

Abstract

The increasing use of machine learning in safety-critical systems presents significant challenges for organizations seeking to ensure reliability, accountability, and public trust. Unlike traditional software, machine learning systems rely heavily on data and statistical learning processes, making their behavior more difficult to predict, explain, and verify. As a result, many existing safety and risk management practices struggle to adequately address the unique risks introduced by learning-enabled components.

This paper presents practical lessons from the application of the Machine Learning Failure Mode and Effects Analysis (ML FMEA) method, an open-source framework designed to systematically identify and manage risks throughout the machine learning development process. Building on established engineering safety practices, ML FMEA provides a structured approach for connecting data, models, and development decisions to system-level safety outcomes.

Using detailed examples drawn from active autonomous vehicle and humanoid robot development projects, this work demonstrates how ML FMEA is applied in real engineering environments. These examples illustrate how the method supports problem definition, pipeline documentation, hazard analysis, and failure mode prioritization, and how it enables consistent communication across safety, systems, and machine learning teams. The paper further reports refinements to the ML FMEA framework, including enhanced pipeline stages, tailored risk assessment criteria, and systematic alignment with international safety standards.

Through open-source collaboration and cross-industry validation, ML FMEA has matured into a practical toolset for supporting auditable, repeatable, and risk-aware development of learning-enabled systems.

Introduction

The integration of machine learning (ML) into safety-critical systems continues to accelerate across industries, from autonomous vehicles to medical diagnostics. This rapid adoption has outpaced the development of standardized risk assessment methodologies tailored to ML's unique characteristics. Traditional software safety analysis techniques often fail to address the probabilistic nature, data dependencies, and opacity inherent in ML systems.

The ML Failure Modes and Effects Analysis (ML FMEA) method, introduced by Schmitt et al.[1], addressed this gap by adapting and extending the well-established Process Failure Mode and Effects Analysis (PFMEA) framework[2] to the ML development pipeline. The ML FMEA is open source to facilitate knowledge sharing and adaptation to applications and industries[3]. The original ML FMEA mapped this proven methodology to an eleven-step ML pipeline, from data collection to model validation. Since its introduction, multiple organizations across diverse industries have implemented the ML FMEA method, generating valuable insights about its practical application. To enable sharing of lessons learned and advancing the overall approach across industries and applications, the ML FMEA Collaborative was formed[4].

From the Collaborative knowledge sharing, this paper presents the results from two specific applications: autonomous vehicles and humanoid robots. The results include methodological refinements, lessons learned, connection to safety standards, and evidence of the method's maturity for widespread adoption.

adjacent lanes. See Figure 2 for simplified reference architecture. Note that this example extrapolates equally to modular end-to-end machine learning or mixed (analytical and machine learning) architectures.[11] [12]

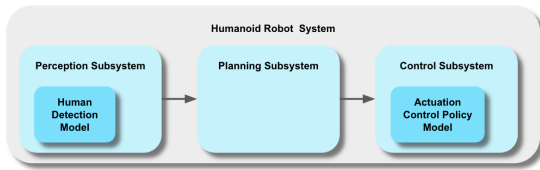


Figure 3: Humanoid Robot system diagram highlighting the Human Detection and Actuation Control Policy model components.

Humanoid Robot Application. Another representative example comes from the domain of humanoid robots deployed in warehouses and research labs for tasks such as object transport, sorting, and interaction in human-populated spaces. These systems rely on learned control policies for locomotion and manipulation, and perception models to detect human presence and maintain safe separation. See Figure 3 for the simplified reference architecture. In one past incident, a reinforcement learning–based policy trained in simulation exhibited unstable recovery behaviors when deployed on physical hardware, leading to unsafe joint accelerations under unexpected perturbations. In another past incident, a human detection system failed to reliably identify seated or partially occluded individuals, resulting in proximity violations. Application of the ML FMEA revealed failure modes tied to insufficient data diversity, weak reward specification, and missing containment logic during deployment. Mitigations included runtime motion bounding, improved annotation guidelines, and integration of confidence thresholds into the robot’s safety monitors. These changes enabled traceability between ML artifacts and safety-relevant system behavior, and supported alignment with functional safety standards such as ISO 13849 and IEC 62061.

Confidentiality Disclaimer. The application examples presented in this paper are intentionally described at a level of abstraction that supports clear illustration of the ML FMEA methodology and its benefits. Specific technical details, proprietary architectures, or sensitive system parameters have been omitted or generalized in order to respect confidentiality obligations and protect intellectual property. The examples remain representative of real-world use cases and are sufficient to demonstrate the relevance and applicability of the proposed contributions.

ML Pipeline “Step Zero”

When the ML FMEA framework was applied in multiple projects, the development teams identified the need for a preliminary activity that precedes Step 1 (*Collect Data Requests*). In practice, several challenges arose before data collection could even begin. Teams were uncertain about what problem the ML model was intended to solve, how the performance of an ML model would be evaluated for safety, and which stages of the development pipeline were considered in or out of scope for analysis. To address these recurring issues, the ML FMEA application team introduced a “Step Zero”: *Define Scope*. See Figure 4.

Clarifying the Problem

Early ML FMEA workshops often began with an assumption that the purpose of an ML model was already well understood. However, experience revealed that vague or evolving problem statements about why an ML model is needed led to inconsistent risk assessments and misaligned mitigation strategies. Defining the problem first proved essential to grounding later discussions of potential failure modes [13, 14, 15]. This included defining what the ML model is intended to achieve, and how its output contributes to or influences the

encompassing system’s behavior under certain operating conditions.

This activity resembles the “Understand the Objectives” phase described in ISO 29119-11 [16], and it provided a natural alignment point with the governance practices outlined in ISO 42001 [17] and ISO 23053 [18]. Once the objectives were made explicit, teams were better equipped to evaluate whether ML was the appropriate solution at all, or whether a deterministic or rules-based approach would be more transparent, verifiable, and cost-effective [19, 20].

- **Autonomous Vehicle Example.** In the context of a perception model tasked with detecting lane boundaries, clarity around the model’s intended functional scope is essential for consistent risk analysis. For example, if the Road and Lane model is assumed to provide subject vehicle lateral localization, when that function is instead performed by a separate downstream localization component, failure modes may be incorrectly attributed or overlooked. Similarly, if the model detects not only subject vehicle lane boundaries but also those of adjacent lanes, omitting this detail in the problem definition can lead to underrepresented hazards, such as unintended lane departure or unsafe adjacent-lane merging. Without explicit articulation of model responsibilities and system boundaries, severity assessments may fail to account for the full set of vehicle-level hazards influenced by the model’s outputs. Formal clarification of the model’s role enables consistent mapping between function, failure mode, and system-level safety impact.
- **Humanoid Robot Application.** For a humanoid robot performing a transport task while walking through a warehouse, the perception subsystem is typically responsible for supporting safe navigation in human-populated environments. This includes detecting nearby humans, estimating occupied and free space, and identifying obstacles that may intersect with the robot’s walking envelope. If the purpose and boundaries of this perception function are not explicitly defined, risk analysis can become inconsistent. For example, treating the perception task as limited to floor-level obstacle detection omits hazards associated with upper-body collisions, occlusions from shelving, or transient human motion near the robot’s torso and arms. By clearly defining the perception model’s role as providing inputs for safe footstep placement and whole-body collision avoidance under dynamic conditions, severity assessments can be grounded in the full range of system-level hazards associated with the transport task. This clarification enables consistent identification of failure modes and defensible linkage between perception performance and potential unsafe system behavior.

Documenting the Pipeline

Another insight from early implementations was the lack of a common, documented view of the ML pipeline itself. Different teams used the term “pipeline” to mean different things. Sometimes, the “pipeline” included only the training process. Other times, the full lifecycle including deployment and feedback was included.

To create a shared reference, teams defined the specific steps used in their workflow using a concise “verb + noun” phrasing (e.g., *Collect Data Requests, Validate Data, Train Model*). See Figure 4. This exercise clarified both the value added at each step and the interfaces between roles. By explicitly documenting the pipeline itself, development teams and safety reviewers were better able to align on scope, responsibilities, and terminology. This is consistent with open documentation practices advocated by Gebru et al. [13] and Mitchell et al. [14]. It also provided a stable artifact for future audits and process improvement discussions, consistent with documentation practices in ISO/PAS 8800[10] and ISO/IEC TR 5469[21].

- **Autonomous Vehicle Application.** For an autonomous vehicle driving within a roadway lane, inconsistent or vague definitions of the perception pipeline can obscure the traceability of failure

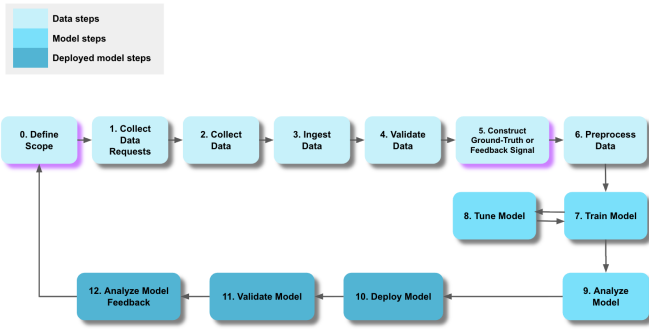


Figure 4: ML Pipeline updated with Step Zero and Step 5. Additionally the data steps, model steps, and deployed model steps are highlighted.

modes. For the Road and Lane model, if the “validate data” step[1] is ambiguously defined, it may be interpreted as validate data format conversion or validate perception sensor calibration. Other interpretations could assume data diversity verification and completeness across the operational design domain (e.g., lane types, weather conditions, and road geometries). Without clear delineation of steps such as data coercion, annotation verification, and taxonomy normalization, it becomes unclear whether key failure contributors such as mislabeled dashed lines (e.g., “dashed” vs. “broken” white line) or inconsistent 3D coordinate frames, are addressed during data preparation or deferred to later model evaluation. This lack of clarity reduces the defensibility of severity assessments and hinders consistent identification of failure modes.

- **Humanoid Robot Application.** For a humanoid robot performing a transport task while walking through a warehouse, ambiguity in the definition of the perception pipeline similarly led to misalignment in risk analysis. In particular, steps related to data validation and preparation were interpreted inconsistently across disciplines. Some interpretations treated validation as confirming sensor integrity and timestamp alignment, while others assumed it included verification that datasets adequately represented safety-relevant scenarios such as human occlusions, varied postures, and interactions near shelving or pallets. Without explicitly documenting these pipeline activities, it was unclear whether gaps in dataset diversity—such as limited examples of partially occluded or seated humans—were addressed upstream or deferred to later model evaluation stages. Explicitly documenting the perception pipeline using clear step definitions enabled consistent identification of failure modes related to insufficient data coverage and improved traceability between perception limitations and potential hazards during the transport task.

Establishing Common Vocabulary

Throughout implementation, terminology mismatches emerged as a subtle but frequent source of confusion. Terms such as “validation,” “testing,” or “performance” carried different meanings to data scientists and safety engineers. Creating a shared glossary early in the process that links ML concepts to established safety language from ISO 26262, ISO 21448, and PFMEA proved to be one of the most effective low-cost ways to improve communication and review quality [15]. Similarly, defining success metrics and acceptance criteria during Step Zero made downstream discussions of “good enough” more objective [14]. By articulating performance thresholds, coverage criteria, or safety indicators upfront, ML FMEA analyses could be performed more efficiently because mitigations could be evaluated against explicit goals rather than intuition.

Clarifying “Safety”

During implementation workshops, teams observed that even the term “safety” itself carried different meanings depending on disciplinary background. Within the ML research community, safety commonly refers to an algorithm’s ability to satisfy constraints or remain within defined safe bounds during learning or operation. Representative examples include methods that guarantee Lyapunov stability[22], probabilistic constraint satisfaction[23], or bounded risk under distributional uncertainty[24]. In this sense, safety denotes a property of the learning or control process, an internal guarantee of bounded behavior given explicit mathematical assumptions. It is typically verified through proofs, convergence bounds, or empirical validation under modeled conditions.

By contrast, functional safety engineering defines safety as the *freedom from unreasonable risk* to people or property, as codified in standards such as IEC 61508, ISO 26262, ISO 21448, ISO/TS 5083 and UL 4600. This perspective is inherently system-level, emphasizing hazard identification, risk reduction, and assurance arguments supported by evidence. Algorithmic “safety” alone is therefore insufficient: a stable or constraint-satisfying model may still contribute to an unsafe system if used without appropriate context, redundancy, or supervision. It is therefore necessary to bridge these interpretations by deriving model-level safety properties based on required system-level hazard mitigations.

Analyzing Hazards

As part of the ML FMEA, the effects of a ML development pipeline failure mode must be identified to enable severity estimated. However, when the ML FMEA was applied to real-world projects, teams realized that the safety-relevant, system-level effect of a failure mode in the ML development pipeline must be identified in terms of potential unsafe behavior of the encompassing system in which the ML component is used[19]. Similarly, scoring the severity of a failure mode also requires understanding the role of the ML component within the broader encompassing system.

- **Autonomous Vehicle Application.** When identifying vehicle-level hazards associated with the Road and Lane model, the analysis begins with a clear definition of the model’s functions. As previously described, this model performs the function of identifying lane boundaries directly adjacent to the subject vehicle. At the Perception subsystem level (see Figure 2), this function supports the higher-level task of localizing the vehicle within its lane. At the vehicle level, it ultimately supports the operational goal of maintaining the vehicle within its intended lane of travel. Failure of this model function was therefore directly linked to the hazards (1) unintentional lane departure and (2) collision with a vehicle in an adjacent lane. Establishing these traceable relationships enables severity assessment of associated failure modes.
- **Humanoid Robot Application.** For a humanoid robot performing a transport task in a warehouse, the perception subsystem supports safe locomotion by detecting nearby humans and estimating occupied space. At the system level, this function enables collision avoidance while the robot walks through shared work areas. A failure of this function (i.e., a false negative human detection due to occlusion by shelving or carried objects) was directly linked to the hazard of unintended contact between the robot and a nearby person. Establishing this traceable relationship between perception failure modes and system-level hazards enabled consistent severity assessment within the ML FMEA.

Step Zero introduces preliminary hazard analysis to connect ML-specific failure modes to system-level hazards of the encompassing system[20]. This analysis establishes the foundation for severity assessments by explicitly linking data quality issues or model

performance degradation to potential system-level consequences. The definition and hazard analysis of the encompassing system involves:

- Identifying the safety-critical function that an ML component performs within the encompassing system
- Determining how the ML component's outputs are used to make system-level decisions
- Performing a hazard analysis (e.g., FMEA, Systems-Theoretic Process Analysis (STPA)) of the encompassing system to determine how undesirable outputs of the ML component could lead to system-level hazards of the encompassing system (e.g., a collision for an autonomous vehicle).

Reflections and Recommendations

Experience across multiple ML FMEA implementations showed that introducing Step Zero (See Figure 4) fundamentally changed both the quality and efficiency of the analysis. Teams that began with a structured problem definition, a documented pipeline and terminology, and a preliminary hazard analysis entered the subsequent ML FMEA sessions with a shared understanding of context and consequence. Rather than debating terminology or scope, participants could focus directly on identifying credible failure modes and assessing their safety relevance.

The inclusion of hazard analysis proved especially valuable. By explicitly linking ML-specific process failures (e.g., incomplete data, labeling errors, or model drift) to the potential unsafe behaviors of the encompassing system, analysts were able to assign severity rankings with far greater consistency and defensibility. This integration also strengthened traceability between process-level failures and system-level hazards, enabling the ML FMEA to serve as supporting evidence within broader safety cases and hazard logs.

From an organizational perspective, Step Zero provided a unifying preparation activity that bridged disciplines. Data scientists, systems engineers, and safety practitioners could align on terminology, assumptions, and acceptance criteria before risk scoring began. This up-front collaboration reduced rework, improved audit readiness, and created a repeatable pattern for future analyses. The time investment was small (often a single facilitated session) but the clarity it provided benefited the entire lifecycle of the project.

In summary, the lessons learned from implementation suggest that Step Zero should be viewed not merely as an administrative prelude but as an enabling foundation for rigorous safety analysis of machine-learning systems. When objectives, pipeline definitions, and hazard relationships are established early, the ML FMEA can be used to connect process quality to system safety. This refinement reflects the growing maturity of safe ML practices and aligns naturally with the intent of emerging standards such as ISO 42001, ISO 23053, ISO/IEC TR 5469, and ISO/PAS 8800.

New ML Pipeline Step 5: Construct Ground Truth or Feedback Signal

Step 5 Description

The feedback from the ML FMEA application teams identified a critical gap between the Validate Data step and the Preprocess Data step of the ML pipeline. The updated ML Pipeline now includes Step 5: Construct Ground Truth or Feedback Signal, which branches based on the ML learning paradigm:

- Supervised Learning: "Construct Ground Truth" - covering label generation, quality control, and annotation consistency
- Reinforcement Learning: "Construct Feedback Signal" - addressing reward specification, shaping, and validation

Making this step explicit captures common failure sources such as systematic annotation bias, label noise, reward hacking vulnerabilities, and misaligned objective functions. Including this activity as an explicit step in the ML development pipeline enables teams to identify and mitigate quality issues that were previously hidden between data validation and preprocessing.

Step 5 Failure Modes

Following the pattern established in *Introducing the ML FMEA*[1], the authors identified potential failure points unique to **Step 5: Constructing Ground Truth or Feedback Signals**, along with typical causes and mitigation proposals. The examples that follow are drawn from supervised and reinforcement learning applications typical across industry applications.

Failure 1: Systematic Annotation Bias or Inconsistent Labeling

Cause: Human annotators or labeling tools apply subjective or inconsistent criteria, resulting in biased or unreliable ground truth.
Effect: Downstream models learn and reinforce existing contextual or environmental biases, reducing generalization and safety margin.
Mitigations: Develop explicit annotation guidelines and document dataset motivation, composition, and intended use following "Datasheets for Datasets" [13]. Use inter-rater agreement metrics such as Cohen's κ [25] or Fleiss' κ [26] to verify labeling consistency. Apply bias-auditing techniques and structured labeling interfaces to minimize subjective drift. Publish dataset documentation alongside model "Model Cards" [14] to ensure transparency of labeling assumptions.

- **Autonomous Vehicle Application.** For the Road and Lane model, a key failure cause is systematic annotation bias, particularly when annotations consistently deviate toward one side of the subject vehicle lane. In such cases, the model may learn an inappropriate offset or skewed understanding of lane boundaries. If the biased labels are localized to a subset of the training data (e.g., such as those provided by a specific annotator) the inconsistency can lead to training instability. During training, the model may exhibit poor convergence.
- **Humanoid Robot Application.** For a humanoid robot operating in a warehouse, systematic annotation bias (e.g., inconsistent labeling of partially occluded humans) can lead to false negative detections, causing the robot to underestimate occupied space and increasing the risk of unintended human contact during transport tasks.

Failure 2: Label Noise or Corrupted Ground Truth

Cause: Errors introduced during automated labeling, data ingestion, or the integration of third-party datasets.
Effect: Label noise reduces model reliability and can mask true safety-critical failure cases.
Mitigations: Apply automated label-quality estimation such as Confident Learning [27] to identify and correct mislabeled samples. Perform random spot checks and human audits, motivated by evidence of pervasive test-set label errors [28]. Maintain dataset lineage and provenance metadata [13] to ensure traceability. Use cross-modal verification (e.g., image-text or LiDAR-camera) to detect inconsistent annotations prior to model training.

- **Autonomous Vehicle Application.** For the Road and Lane model, a key concern is mismatch between the training data coverage versus the intended use of the model. Specifically, the model is intended to support lateral localization in a wide variety of road types, including unstructured environments such as intersections, reversible lanes, and construction zones. Examples training data mismatch issues include an overrepresentation of

well-marked highway segments as well as a lack of edge cases involving temporary markings, occluded lanes, or inconsistent curb definitions. The effect of this mismatch is that the model is likely to underperform in situations where robust behavior is critical for safety.

- **Humanoid Robot Application.** For a humanoid robot operating in a warehouse, label noise in training data (e.g., such as corrupted or inaccurate annotations of partially occluded humans) can result in missed detections in cluttered environments, reducing the reliability of human proximity estimation during transport tasks.

Failure 3: Misaligned Objective Functions

Cause: The metrics optimized during training (e.g., accuracy or cumulative reward) fail to represent system-level performance or safety intent.

Effect: Models optimize proxies that diverge from operational goals, leading to unsafe or unethical behavior.

Mitigations: Apply inverse reward design to represent uncertainty over designer intent [29]. Incorporate human-in-the-loop validation using preference-learning methods [30]. Define explicit safety-oriented performance indicators in Step Zero, informed by AI safety hazard catalogs [19]. Regularly review optimization metrics against system-level hazard analyses and safety of the intended function (SOTIF) functional-insufficiency criteria [31].

- **Autonomous Vehicle Application.** For the Road and Lane model, misalignment between the training objective and the system-level function arose from the structure of the loss function. One identified cause was the uniform penalization of errors across all predicted lane boundaries, regardless of their proximity to the subject vehicle. This configuration resulted in an overcritical model (i.e., one that disproportionately emphasizes less-relevant features, such as distant lane lines) at the expense of performance near the subject vehicle lane. Alternatively, a loss function that penalizes errors only near the subject vehicle yielded an undercritical model, insufficiently attentive to adjacent lanes. This impairs situational awareness needed by downstream modules during edge cases (e.g., merges or cut-ins).
- **Humanoid Robot Application.** For a humanoid robot performing warehouse transport tasks, a misaligned objective function that prioritizes task efficiency (e.g., minimizing traversal time or energy use) without adequately penalizing proximity to humans can lead to policies that satisfy training metrics while violating required separation distances. This divergence between optimized objectives and system-level safety intent highlights the need to explicitly encode human-safety constraints when defining learning objectives.

Failure 4: Reward Hacking or Unintended Optimization Behavior

Cause: In reinforcement-learning systems, the reward function does not align with the true system objectives, leading to exploitation of loopholes.

Effect: Agents achieve high numerical reward while performing unsafe or undesirable actions.

Mitigations: Incorporate safety constraints and tamper-resistant reward design principles [32]. Adopt safe reinforcement-learning techniques that constrain exploration or penalize unsafe transitions [20]. Use causal-influence analysis or adversarial testing to identify reward-tampering pathways [19]. Validate learned behaviors in simulation before deployment and cross-verify with human-preference models [30].

- **Autonomous Vehicle Application.** This failure type was not applicable to the Road and Lane model, as the model

architecture is based on supervised learning using convolutional neural networks (CNNs). No reinforcement learning (RL) elements were used in training or deployment. Therefore, the failure related to incomplete exploration of the state-action space during RL policy learning was not evaluated in this context. Rather, this failure type was developed as a corollary to Failure 3, applying the same failure logic to the RL context.

- **Humanoid Robot Application.** This failure type was not applicable to the humanoid perception system analyzed for warehouse transport, as the human-detection functionality was trained using supervised learning rather than reinforcement learning. Consequently, reward hacking or unintended optimization behavior was not evaluated for this failure mode, which was instead considered relevant only for RL-based locomotion or manipulation controllers analyzed elsewhere.

Failure 5: Insufficient Statistical Analysis of the Dataset Over Time

Cause: Ground-truth dataset not routinely analyzed for distribution shifts across time, geography, or sensor configuration, masking key differences.

Effect: Model underperforms on uncommon scenarios. Root cause analysis delayed due to unrecognized data drift.

Mitigations: Adopt configuration-managed repositories and structured metadata, as advocated in “Datasheets for Datasets” [13] and “Model Cards” [14]. Integrate dataset-lineage tracking into continuous-integration pipelines [15]. Establish formal baselines and version identifiers for each labeled or rewarded data release, with sign-off criteria verified through inter-annotator metrics [25, 26].

- **Autonomous Vehicle Application.** For the Road and Lane model, insufficient statistical analysis of the dataset over time resulted in several safety-relevant risks. The dataset included samples from multiple vehicle platforms and sensor configurations, collected across a wide range of regions and time periods. In the absence of routine distribution analysis, subtle shifts such as variations in lane marking styles, vehicle sensor calibration processes, environmental conditions, or annotation practices were not systematically identified. This lack of visibility into temporal and geographic variation reduced the model’s ability to generalize to less common or evolving conditions, such as repainted construction zones or regional lane designs. As a result, performance degradation in safety-critical scenarios was more difficult to detect, and delayed root cause analysis.
- **Humanoid Robot Application.** For a humanoid robot operating in a warehouse, insufficient statistical analysis of the perception dataset over time obscured shifts in human appearance and environment configuration, such as seasonal clothing, PPE changes, or reconfigured shelving. These unrecognized distribution shifts increased the likelihood of false negative human detections under occlusion, degrading proximity estimation and delaying root cause analysis of safety-relevant performance drops.

Failure 6: Misaligned Evaluation Metrics

Cause: Evaluation benchmarks or test data fail to capture the operational design domain or safety-critical scenarios.

Effect: Models appear performant in offline testing but fail in rare or hazardous conditions.

Mitigations: Perform dataset-slice analysis as recommended by Model Cards [14]. Use coverage-guided scenario generation and out-of-distribution detection as encouraged in ISO/PAS 8800 [10]. Establish continuous-evaluation pipelines with drift detection [15] and post-deployment monitoring tied to severity-ranking updates in the ML FMEA template (See Figure 1).

- Autonomous Vehicle Application.** Two primary causes contributed to evaluation misalignment for the Road and Lane model. The first was a lack of coverage in the training and validation datasets for rare but safety-critical lane configurations, such as dual-turn lanes and diverging diamond interchanges. This led to overfitting, where the model achieved high accuracy on common cases but failed to generalize to low-frequency, high-impact scenarios within the operational design domain. The second cause was the use of strong regularization techniques, including high dropout rates and extensive data augmentation, without adequate tuning or validation protocols. This resulted in underfitting, which manifested as poor model performance even in nominal driving environments.
- Humanoid Robot Application.** For a humanoid robot operating in a warehouse, evaluation metrics that emphasize aggregate detection accuracy failed to capture safety-critical scenarios involving occluded or partially visible humans. As a result, the perception system appeared performant in offline testing but underperformed in rare, high-risk conditions encountered during transport tasks, increasing the likelihood of false negative human detections in cluttered environments.

Step 5 Discussion

Adding **Step 5: Construct Ground Truth or Feedback Signal** formalizes activities that were previously implicit or fragmented across data and model development. The experience of implementing ML FMEA in real-world projects showed that many downstream failures, such as poor model generalization, bias amplification, and misinterpreted performance metrics, originated from weaknesses in ground-truth or reward construction. By making this step explicit in the ML pipeline, organizations can better manage annotation consistency, reward design, and the overall integrity of the supervision signal. This refinement strengthens the link between data quality, learning objectives, and system-level safety, and aligns with the process-level assurance intent of ISO/PAS 8800 [10] and ISO/IEC TR 5469 [21].

Tailored Risk Assessment Criteria

To tailor the severity, occurrence, and detection rating criteria, the team started with the well-established rating criteria tables used in PFMEA. While the criteria in those tables provide structured definitions for rating risk based on the likelihood and impact of failure modes, direct application of these tables to machine learning (ML) workflows proved difficult. Traditional rating criteria assume deterministic causal-and-effect relationships and physically observable effects, whereas ML pipelines involve probabilistic behaviors, data dependencies, and model-driven uncertainty.

To bridge this gap, the team tailored the severity, occurrence, and detection rating criteria to the ML domain, mapping risk concepts to the stages of the ML pipeline and to the roles of practitioners responsible for each step. The result was a set of ML-specific rating criteria that retained PFMEA’s familiar structure while introducing context-appropriate criteria reflecting data quality, model robustness, and detection timing within continuous-integration workflows.

Severity Rating Criteria

The team developed ML-specific severity rating criteria that maintains compatibility with the traditional PFMEA criteria while addressing ML-unique considerations (see Table 1). The table distinguishes between customer/vehicle safety impacts and ML pipeline/process impacts, enabling dual-perspective risk assessment.

Key refinements include:

- Explicit criteria for perception failures (e.g., “perception blind

Effect	Criteria for Severity on Product	Rank	Criteria for Severity on ML Process/Pipeline
Failure to Meet Safety or Regulatory Requirements	ML failure mode directly affects safe vehicle operation, causes missed detection or false actuation, or leads to non-compliance with ISO/UNECE/SAE safety regulations without warning.	10	Training data/pipeline error or model failure leads to safety monitor bypass or catastrophic unsafe behavior; undetected by process until after incident.
Failure to Meet Safety with Warning	ML failure mode affects safe operation but hazard is somewhat mitigated (e.g., monitor catches but with degraded TTC). Regulatory compliance risk with warning.	9	Pipeline/model issue forces partial stop of training, validation, or fleet deployment. May endanger operator/system with warning.
Loss of Primary Function	Loss of ML-driven safety function (e.g., perception blind spot, planner failure). Vehicle inoperable or requires human takeover; safe state not guaranteed.	8	Major disruption: retraining halt, large data asset scrapped, or line shutdown of automated data pipeline.
Heavily Degraded Primary Function	Function still available but safety margins reduced (e.g., low recall, mis-planned maneuvers).	7	Significant disruption: retraining required, pipeline deviation, throughput drop, heavy re-work.
Noticeable Degradation of Primary Function	Noticeable degradation (e.g., 10–20% error on critical slice). Hazards mitigated by redundancy.	6	Moderate disruption: subset of data must be reprocessed or relabeled; partial rollback.
Loss of Secondary Function	Secondary ML function lost (e.g., L2 traffic light misclassification). No direct safety impact.	5	100% of secondary data product must be re-generated offline.
Degraded Secondary Function	E.g., rare class mislabels. Customer notices degraded UX but no safety risk.	4	Portion of pipeline or dataset re-work required; validation delays.
Annoyance / Noise	False positives or non-safety misclassifications (e.g., frequent alerts, false stops).	3	Minor disruption: some re-labels or tuning needed.
Minor Issue, Not Customer-Visible	Issue visible only to expert reviewers. Customers do not notice.	2	Slight inconvenience in pipeline or training; low-effort fix.
No Effect	No impact on safety, performance, or compliance.	1	No effect on pipeline or product.

Table 1: Severity Criteria: Product vs. ML Process Perspective

spot,” “low recall of pedestrians”)

- Graduated safety margin degradation levels
- Clear delineation between primary safety functions and comfort/convenience features
- Pipeline-specific disruption metrics (e.g., “retraining halt,” “data asset scrapped”)

Occurrence Rating Criteria

Traditional PFMEA occurrence rating criteria assume direct correlation between cause probability and failure mode probability, an assumption that breaks down in ML contexts. For instance, 2 percent erroneous annotations may have unpredictable effects on model behavior depending on data distribution and model architecture. Instead, the ML-specific rating criteria proposed in this paper (see Table 2) focus on requirement definition and quality control maturity. This tailoring of the occurrence rating criteria shifts discussion from probabilistic speculation to actionable quality management.

Detection Rating Criteria

Instead of likelihood-based detection ratings, the tailored detection criteria (see Table 3) emphasizes detection timing and automation level across three pipeline segments:

- Data Pipeline Detection: Distinguishes between data producer and consumer detection capabilities

Effect	Criteria	Rank
High	Model performance requirements are undefined and/or model performance checks are not available. –OR– Data quality requirements are undefined and/or data quality checks are not available.	8–10
Medium	Model performance requirements are partially known and/or model performance checks are partially available. –OR– Data quality requirements are partially known and/or data quality checks are partially available. –OR– Data quality requirements are known and data quality checks determine x% of data fails. (Refer to severity table.)	4–7
Low	Model performance requirements are known and model performance checks are available. –OR– Data quality requirements are known and data quality checks are available.	1–3

Table 2: Occurrence Criteria: Availability of Requirements and Checks

ML Pipeline Containment	Criteria	Rank
Detection after data is consumed -OR- Detection after model is consumed -OR- Detection after deployed model is consumed	High: Human-in-the-loop quality check performed by Data Analyst or ML Engineer	10
Detection after data is consumed -OR- Detection after model is consumed -OR- Detection after deployed model is consumed	Medium: Automated quality check	9
Detection before data is consumed -OR- Detection before model is consumed -OR- Detection before deployed model is consumed	High: Data Consumer performs human-in-the-loop check	7-8
Detection before data is consumed -OR- Detection before model is consumed -OR- Detection before deployed model is consumed	Medium: Data Consumer uses automated quality check	5-6
Detection before data is consumed -OR- Detection before model is consumed -OR- Detection before deployed model is consumed	Medium: Data Producer performs human-in-the-loop quality check	3-4
Detection before data is consumed -OR- Detection before model is consumed -OR- Detection before deployed model is consumed	Low: Data Producer uses automated quality check	1-2

Table 3: Detection Risk Assessment Table — For Data, Model, and Deployed Model Steps. ML pipeline containment ranking based on timing of detection and role (data producer vs. consumer, human vs. automated).

2. Model Development Detection: Separates pre- and post-training detection opportunities
3. Deployment Detection: Addresses run-time monitoring and rollback capabilities

For each segment, rating criteria are defined to account for the following factors:

- Detection timing (before/after consumption or deployment)
- Detection method (automated vs. human-in-the-loop)
- Detection ownership (producer vs. consumer responsibility)

Summary

The updated rating criteria tables resonated well with both ML developers and safety engineers. Practitioners found the structure intuitive and the terminology accessible and directly relevant to their daily work. The tables prompted constructive discussion across disciplines, guiding teams toward the right questions for each pipeline stage or segment. By maintaining continuity with traditional PFMEA conventions while embedding ML-specific reasoning, the tailored rating criteria provided a shared risk vocabulary that improved communication, consistency, and traceability between technical and safety teams.

Safety Standard Alignment

Following guidance common across safety standards the ML FMEA sits within an ecosystem of analysis methods. The standards prescribe analysis methods for the purposes of crafting systematic safety requirements. Methods include inductive, deductive, and exploratory analyses to comprehensively cover hazards and their mitigations. The ML FMEA is then one such tool that approaches safety requirement generation from a ML process perspective.

Alignment with ISO/PAS 8800

ISO/PAS 8800 describes systematic safety analysis for AI elements. This covers design, implementation and operational aspects, as well as a specific focus on data used by ML algorithms/models. In general,

the standard outlines the types of techniques that can be used but provides limited guidance on the specific tools and their applications. The ML FMEA directly addresses this gap by:

- Providing the “Safety analysis at the process level” recommended in the standard, particularly the use of PFMEA in sections 11.4.3.1, 12.3.5, 14.8.1, 15.4
- Establishing traceability between ML development activities and safety requirements
- Documenting engineering rigor through systematic safety risk assessment
- Creating auditable evidence of best practice implementation

Alignment with UL 4600

UL 4600 requires comprehensive safety arguments for autonomous systems. ML FMEA artifacts directly support multiple argument patterns:

1. Data Quality Arguments: ML FMEA rows addressing data collection, validation, and preprocessing provide evidence for UL 4600 Section 8.5.3 requirements
2. Robustness Arguments: Failure modes related to distribution shift and environmental variation support Section 8.5.2.2 claims
3. Verification and Validation Process Arguments: The complete ML FMEA demonstrates a systematic hazard analysis as required by Section 8.5.2.1

Our work developing the ML FMEA method is also informing evidentiary processes that support the Open Autonomy Safety Case (OASC) [33]. The OASC represents a non-normative safety case that supports systematic assessment of UL 4600 conformance as well as the reporting of AVSC core safety information [34]. Results of ML FMEA support evidence sufficiency claims in OASC regarding effective hazard analysis of ML-driven behaviors as well as the rigor with which ML models or algorithms are developed.

Alignment with ISO 21448

The ML FMEA method directly addresses SOTIF [31] requirements for identifying functional insufficiencies by:

- Systematically analyzing “specification of machine learning” insufficiencies
- Systematically analyzing triggering conditions in the “data used for ML training and testing”
- Documenting “measurement data for machine learning” quality issues
- Supporting “off-line training process” analysis as specified in Annex D

Alignment with ISO/IEC TR 5469

ISO/IEC TR 5469:2024 [21] is an industry-agnostic standard that provides guidance for establishing trustworthy AI systems by mapping key AI lifecycle stages to applicable international standards. Section 11.5.3 proposes the use of a PFMEA approach to analyze and eliminate possible sources of bias and limitations within the offline training process and the ML FMEA could be used to perform this analysis. Future standards such as ISO 22440 may more broadly examine ML/AI applications and as such apply this method more prescriptively.

Applicability Across Industrial Challenges

This section outlines key ML safety challenges across several industries and demonstrates how the ML FMEA method addresses them.

Automotive and Mobility

The number of autonomous vehicles and consumer-facing assistive driving features that depend on ML or AI as part of their core control function is growing rapidly. For example, perception systems in autonomous vehicles depend heavily on learned models to identify nearby objects such as pedestrians, cyclists, or other vehicles. Other applications of ML and AI appear in prediction, planning, localization, equipment monitoring, and fleet routing aspects of these systems. Similar trends exist in the growing “micro-mobility” where small-scale (usually passenger-less) vehicles are employing ML to navigate more constrained problem spaces, such as food delivery, sanitation, or security monitoring.

Broadly speaking, these systems have significant risk profiles. Some applications, such as robotaxis operating with human passengers in urban areas, can have very high consequences for failure, including fatalities. For instance, in 2018 a pedestrian crossing the road at nighttime while walking beside their bicycle, was struck by an autonomous vehicle (supervised by human operator) [35]; other similarly high consequence events have occurred since [36] [37].

Given the rapidly growing dependence on ML in automotive systems, there is an urgent need to apply systematic methods, such as the ML FMEA method, to the development of these systems. Towards addressing this need, the following are two examples of unique challenges in automotive and mobility applications that can be addressed using the ML FMEA method. These examples illustrate the broad applicability of the ML FMEA method to address various types of AI or ML challenges that might be encountered in automotive and mobility applications.

1. **Challenge: Diversity in Operating Domains** Automotive technology operates in an incredibly diverse operating domain, ranging from remote and rural roads in winter conditions, to mountainous highways, to busy urban centers at nighttime. Even within one of these operating domains, the amount of variability in operating conditions is vast. The ML models used for perception must have been trained on incredibly diverse data sets. This is addressed by failure modes related to the ML FMEA’s *Request Data* (Step 1) and *Collect Data* (Step 2) [1]. For example, the potential failure mode of incorrect prioritization of data collection requests can lead to biased models (e.g., prioritizing “sunny day” camera data when operations at nighttime are also desired). The recommended mitigation for this failure mode is to engage experts from cross-functional teams as part of the data requesting process.
2. **Challenge: Evolving Operating Domains** Busy highway or urban settings are subject to regular changes that might impact an ML model’s ability to correctly detect essential objects. Even something as subtle as changes to clothing worn by sub-populations of pedestrians or cyclists might impact a perception system’s ability to correctly identify them. Suppose that a popular outdoor equipment retailer releases a new line of cycling jackets with extra reflective material that impacts camera-based perception systems. This challenge is addressed in the *Ingest Data* (Step 3) step of the ML FMEA method by the potential failure mode of delays in data ingestion leading to out of date models. Similarly, the *Request Data* (Step 1) and *Collect Data* (Step 2) steps might also help address this challenge.

Aerospace and Defense

FMEA is a proven, recognized analysis technique for MIL-STD-882E [38] system safety programs. However, system-safety practitioners rarely have the expertise in AI necessary to evaluate ML-relevant failure modes effectively. The contributions of the ML FMEA team therefore represent a helpful bridge between expertise in these two domains.

ML-specific failure modes trace directly into a MIL-STD-882E program. Use them to seed the Preliminary Hazard List (Task 201) and to inform the Functional Hazard Analysis (Task 208), where affected functions are classified and safety-critical functions identified, and then carry them forward into the System Hazard Analysis (Task 205). Convert the ML FMEA’s severity/occurrence ratings into the standard’s severity categories and probability levels to produce risk assessments, and record each hazard, causal factor, mitigation, V&V method, and risk acceptance in the program’s closed-loop Hazard Tracking System (Task 106) while working through the MIL-STD-882E eight-element system-safety process. MIL-STD-882E explicitly integrates software system safety into PHA/SSHA/SHA/FHA, so ML FMEA becomes a disciplined input to those analyses and the overall safety case. We reference the ML FMEA in the System Safety Program Plan (Task 102) and Hazard Management Plan (Task 103), and roll updates into the Hazard Management Progress Report (Task 107); all consistent with the MIL-STD-882E direction that other engineering disciplines may adapt the methodology while still complying with Sections 4.3–4.4.

The authors are presently deploying ML FMEA research outputs to multiple uncrewed ground vehicle programs for the US Army and is educating test and evaluation centers on how they can inform safety assessment reports and risk acceptance decisions.

Humanoid and Industrial Robotics

The humanoid and industrial robotics sectors increasingly rely on machine learning (ML) for advanced perception, control, and human-machine interaction capabilities. In environments ranging from manufacturing floors to warehouse automation systems, ML is now used for tasks such as human detection, intent inference, grasp planning, and dynamic motion adaptation. As use cases shift from fenced-off, pre-programmed manipulators toward collaborative robots (cobots), mobile platforms, and general-purpose humanoids, safety requirements evolve as well. The transition to learning-enabled systems introduces probabilistic behaviors, opaque policies, and heavy reliance on training data, all of which require more structured risk assessment approaches.

A particularly pressing challenge in these domains is ensuring the safety of human-robot interaction. ML-based perception systems are often tasked with detecting human presence, estimating body pose, and predicting trajectories to ensure the robot can plan and execute motions without violating proximity thresholds or causing harm. Additionally, reinforcement learning is increasingly used to train control policies for complex locomotion and manipulation tasks in humanoid platforms. These policies may meet nominal task performance goals in simulation, yet exhibit unsafe behavior when transferred to the real world, especially in the presence of sensor noise, distributional shifts, or misaligned reward signals.

Given these challenges, the ML FMEA method has proven particularly useful in industrial contexts for identifying and mitigating high-risk failure modes associated with perception, control, and policy training workflows. Below, we highlight several specific examples.

1. **Challenge: Robust Human Detection in Unstructured Environments** Unlike automotive scenarios where pedestrians are typically external to the vehicle, many industrial robotics applications require robots to operate in close proximity with humans—often within arm’s reach. ML-based vision or LiDAR systems are used to detect human workers for dynamic speed and separation monitoring (DSSM) as prescribed in ISO 10218 and ISO/TS 15066. However, these detectors may fail in low-light conditions, due to occlusion, or when encountering uncommon clothing, body postures, or reflective gear. These concerns are directly addressed through the ML FMEA’s *Collect Data* (Step 2) and *Construct Ground Truth* (Step 5) steps. For instance, failure modes such as insufficient representation of

worker diversity (e.g., PPE variations, body shapes, or poses) or systematic annotation bias (e.g., labeling seated workers as background) can be identified early. Mitigations include targeted data augmentation, synthetic data generation, or stricter labeling QA policies, all of which can be documented and tracked using the ML FMEA template[3] (See Figure 1).

2. **Challenge: Unsafe Emergent Behavior in RL Control Policies** RL is a popular approach for developing complex motion policies for humanoids and mobile manipulators, especially in environments that require fine-grained balance, reactive adjustments, or long-horizon decision making. However, such policies are often learned in simulation and transferred to physical systems via sim-to-real pipelines. Without structured failure analysis, this approach risks producing brittle or unsafe behaviors—e.g., high-speed joint movements, unsafe contact forces, or reward hacking behaviors that exploit loopholes in the reward function. These risks are captured in Step 5 *Construct Ground Truth or Feedback Signal* and Step 7 *Train Model* stages of the ML FMEA. Common failure modes include poorly specified or non-aligned reward objectives, unsafe policy generalization, and insufficient runtime safety constraints. The ML FMEA allows teams to flag these issues preemptively, define test-time safety filters (e.g., joint velocity caps, safety shields), and improve reward validation workflows. Severity scoring in such cases draws directly from the robot’s physical operating envelope and potential harm to nearby humans.
3. **Challenge: Continuous Integration of Learned Policies into Legacy Safety Envelopes** Industrial robots often operate within safety-certified control frameworks governed by PLCs, fieldbuses, and configurable safety monitors. Introducing learned components into this stack—such as ML-based vision or RL-based motion planners—requires maintaining compatibility with deterministic control paths and preserving the validity of existing safety cases. The ML FMEA’s Step Zero (*Define Scope*) becomes particularly critical in this context, as it ensures the roles, interfaces, and boundaries of ML components are clearly documented. Teams performing hazard analysis on the encompassing system can use the ML FMEA outputs to trace how learned behaviors may affect legacy safe states or require new monitoring channels (e.g., override triggers based on classifier confidence scores or model uncertainty estimates).

These examples underscore the utility of the ML FMEA method in identifying failure modes that arise uniquely from ML-driven control and perception workflows in industrial settings. By providing a structured framework to bridge ML development and safety analysis, the method supports proactive risk mitigation, cross-disciplinary alignment, and compliance with functional safety standards (e.g., ISO 13849, IEC 62061). Given the increasing adoption of learning-enabled robots in shared human environments, the integration of tools like the ML FMEA is essential for ensuring trust, safety, and auditability in these systems.

Discussion: Team Reflections

ML Development Team Perspectives

ML practitioners reported that the ML FMEA process brought new structure and visibility to their workflows, revealing relationships between data quality, model behavior, and safety outcomes that were previously implicit or untracked. Key benefits included:

- **Unified Risk Repository:** “Having data and model failure modes in one database is highly beneficial. The connection to safety metrics was enlightening. We had no similar tool providing this visibility.” The integration of data- and model-level risks into a single framework allowed ML teams to trace how upstream data issues could propagate into model

behavior and ultimately affect system safety. This unified view encouraged proactive coordination between data engineers, ML scientists, and safety assessors.

- **Systematic Communication:** “The ML FMEA provides a structured way to explain our approach to safety assessors and auditors.” The team found that the structured template [3](see Figure 1) and scoring tables served as a common language for cross-disciplinary discussions, reducing ambiguity when describing model risks to non-ML experts. The FMEA framework effectively bridged the gap between development documentation and formal safety case evidence.
- **Efficiency Gains:** “After prioritizing RPNs, we could see how one solution might address multiple failure modes simultaneously, saving significant time and effort.” By analyzing correlations among failure modes, teams identified mitigation strategies with cross-cutting benefits such as improved labeling workflows or enhanced data validation. These strategies reduced overall engineering effort while increasing safety assurance coverage.

Practitioners also proposed several enhancements to improve usability and scalability. A recurring theme was the need for visualization tools to support large-scale analyses. Teams suggested dashboard visualizations that highlight “problem steps” in the pipeline, as well as graph-based representations showing how failure modes propagate across data, model, and deployment stages. Another desired feature was hierarchical handling of nested complexity, allowing sub-steps or component-level FMEAs to be linked within the broader pipeline model. These refinements would make it easier to apply the ML FMEA method across multiple models and continuously evolving pipelines.

Safety Engineering Perspectives

Safety engineers expressed strong enthusiasm for the method, noting that it fostered unprecedented engagement from ML development teams in the safety dialogue. Key insights included:

- **Role Clarity:** Safety engineers naturally gravitate toward severity assessment, which requires system-level understanding of potential hazards and impacts, whereas ML developers excel at assessing occurrence and detection, drawing on their detailed pipeline knowledge. This division of expertise improved both the accuracy and efficiency of risk scoring.
- **Communication Bridge:** “The ML FMEA provided a structured framework for safety experts to ask the right questions of development teams.” The tabular format and shared terminology enabled safety reviewers to probe data assumptions, training procedures, and model limitations in a structured and non-adversarial manner. This promoted a culture of shared ownership over safety rather than siloed accountability.
- **Conceptual Challenges:** The distinction between severity, occurrence, and detection often blurred in ML contexts, particularly when evaluating probabilistic or adaptive behaviors. Facilitators found that additional guidance and calibration examples—such as those developed in the refined rating tables—were essential for ensuring consistent interpretation across projects.

Summary of Team Reflections

Overall, both ML developers and safety engineers reported that the ML FMEA served not only as a beneficial technical analysis tool but also as a cross-functional learning mechanism. The process of performing the ML FMEA analysis encouraged mutual understanding between disciplines and revealed new pathways for integrating safety thinking directly into ML development practices.

Conclusion

The open source Machine Learning Failure Mode and Effects Analysis (ML FMEA) has advanced from a conceptual framework into a practical, standards-aligned methodology for managing risk in machine learning development. Through iterative application and cross-industry feedback, the method has matured to address both technical and organizational needs in safety-critical domains. One autonomous vehicles and one humanoid robot application are used to highlight the application and benefits of the methods.

Two major structural refinements mark this evolution. Step Zero establishes problem definition, system boundaries, and preliminary hazard analysis before ML development begins, ensuring that subsequent risk assessments are grounded in system-level context. Step 5: Construct Ground Truth or Feedback Signal captures risks unique to supervised and reinforcement learning, providing structured mitigation for issues such as annotation bias, reward misalignment, and feedback signal corruption. Together, these additions close methodological gaps identified during early deployments and strengthens traceability between ML process quality and system safety outcomes.

The tailored severity, occurrence, and detection rating tables provide more relevant guidance in scoring risk. Not only does this guidance improve consistency in risk scoring, it also aligns ML FMEA assessments with established safety engineering principles. These refinements, along with demonstrated compatibility with ISO/PAS 8800, ISO 21448 (SOTIF), UL 4600, and MIL-STD-882E, establish a clear pathway for integrating ML FMEA results into formal safety cases. The ML FMEA also supports evidence generation for the Open Autonomy Safety Case (OASC) framework, reinforcing its role as a complementary tool for safety argumentation and regulatory documentation.

Beyond its technical rigor, the ML FMEA continues to demonstrate value as a shared language between ML practitioners and safety engineers. By providing a structured process, standardized terminology, and auditable artifacts, it promotes collaboration across disciplines that have traditionally worked in isolation. This communication bridge has proven to be one of the most impactful outcomes of the method's adoption.

Ongoing collaboration through the ML FMEA Collaborative[4] and the open-source ML FMEA template[3] ensures that the methodology remains current, transparent, and adaptable across domains. The collective experiences captured in this work (i.e., methodological enhancements, standards integration, and practical application) demonstrate a clear and achievable path toward widespread use of the ML FMEA as a foundational tool for ensuring the safety of machine learning systems. The authors feel that the approach is now ready for application across industries employing machine learning in safety critical applications including including automotive, robotics, healthcare, aerospace, and defense.

References

1. P. Schmitt, H. Seifert, M. Bijelic, K. Pennar, M. Bijelic, F. Heide, and J. Lopez, "Introducing the ML FMEA," in *SAE World Congress*, 2025.
2. Ford Motor Company, *FMEA Handbook*. Ford Motor Company, 2024. Available via the Ford Supplier Portal (FSP).
3. P. Schmitt, "Repo: Machine Learning Failure Mode and Effects Analysis (FMEA)," 2025. GitHub repository: <https://github.com/TallPaul67/MachineLearningFMEA>.
4. "The ML FMEA Collaborative." TORC Robotics, TÜV Rheinland Group, HORIBA MIRA Ltd, Saphira AI, APTIV,

MassRobotics, and Edge Case, 2025. <https://ml-fmea-collaborative.github.io/>, Last accessed on December 19, 2025.

5. International Electrotechnical Commission, "Functional safety of electrical/electronic/programmable electronic safety-related systems," International Standard IEC 61508:2010, International Electrotechnical Commission (IEC), Apr 2010.
6. International Organization for Standardization, "Road vehicles — functional safety," International Standard ISO 26262:2018, International Organization for Standardization (ISO), Dec 2018. Second edition, replaces ISO 26262:2011.
7. International Organization for Standardization, "Road vehicles — safety of the intended functionality," International Standard ISO 21448:2022, International Organization for Standardization (ISO), Jun 2022.
8. International Organization for Standardization (ISO), "Road vehicles — safety for automated driving systems — design, verification and validation," Tech. Rep. ISO/TS 5083:2025, ISO, Apr 2025.
9. Underwriters Laboratories Standards Engagement (ULSE), "Standard for safety for the evaluation of autonomous products," International Standard ANSI/UL 4600–2nd Edition (2022), Underwriters Laboratories Standards Engagement (ULSE), Mar 2022. Second edition; goal-based, technology-agnostic approach to autonomous product safety.
10. International Organization for Standardization, "Road vehicles — Safety and artificial intelligence," Publicly Available Specification ISO/PAS 8800:2024, International Organization for Standardization, Geneva, Switzerland, 2024.
11. M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to end learning for self-driving cars," 2016.
12. A. Tampuu, T. Matiisen, M. Semikin, D. Fishman, and N. Muhammad, "A survey of end-to-end driving: Architectures and training methods," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, p. 1364–1384, Apr. 2022.
13. T. Gebru, J. Morgenstern, B. Vecchione, J. W. Vaughan, H. Wallach, H. D. III, and K. Crawford, "Datasheets for datasets," *Communications of the ACM*, vol. 64, no. 12, pp. 86–92, 2021.
14. M. Mitchell, S. Wu, A. Zaldivar, P. Barnes, L. Vasserman, B. Hutchinson, E. Spitzer, I. D. Raji, and T. Gebru, "Model cards for model reporting," in *Proceedings of the Conference on Fairness, Accountability, and Transparency (FAccT/FAT*)*, pp. 220–229, ACM, 2019.
15. D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J.-F. Crespo, and D. Dennison, "Hidden technical debt in machine learning systems," in *Advances in Neural Information Processing Systems (NeurIPS 2015)*, pp. 2503–2511, 2015.
16. International Organization for Standardization and International Electrotechnical Commission, "Software and systems engineering — software testing — part 11: Guidelines on the testing of ai-based systems," Technical Report ISO/IEC TR 29119-11:2020, ISO/IEC JTC 1/SC 7, Nov 2020. First edition.
17. International Organization for Standardization (ISO), "Information technology — artificial intelligence — management system," International Standard ISO/IEC 42001:2023, International Organization for Standardization / International Electrotechnical Commission, 2023.

18. International Organization for Standardization (ISO), “Framework for artificial intelligence (ai) systems using machine learning (ml),” International Standard ISO/IEC 23053:2022, International Organization for Standardization / International Electrotechnical Commission, 2022.
19. D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané, “Concrete problems in AI safety,” *arXiv preprint arXiv:1606.06565*, 2016.
20. J. García and F. Fernández, “A comprehensive survey on safe reinforcement learning,” *Journal of Machine Learning Research*, vol. 16, no. 1, pp. 1437–1480, 2015.
21. International Organization for Standardization (ISO), “Artificial intelligence — functional safety and ai systems,” Technical Report ISO/IEC TR 5469:2024, International Organization for Standardization / International Electrotechnical Commission, 2024.
22. F. Berkenkamp, M. Turchetta, A. P. Schoellig, and A. Krause, “Safe model-based reinforcement learning with stability guarantees,” in *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS)*, pp. 908–918, Curran Associates, Inc., 2017.
23. T. Xu, Y. Liang, and G. Lan, “Crpo: A new approach for safe reinforcement learning with convergence guarantee,” in *Proceedings of the 38th International Conference on Machine Learning (ICML)* (M. Meila and T. Zhang, eds.), vol. 139 of *Proceedings of Machine Learning Research*, pp. 11553–11563, PMLR, 2021.
24. S. Bensalem, E.-Y. Kang, T. H. Nguyen, S. Owre, and J. Rushby, “What, indeed, is an achievable provable guarantee for learning-enabled safety-critical systems?,” *arXiv preprint arXiv:2307.11784*, 2023.
25. J. Cohen, “A coefficient of agreement for nominal scales,” *Educational and Psychological Measurement*, vol. 20, no. 1, pp. 37–46, 1960.
26. J. L. Fleiss, “Measuring nominal scale agreement among many raters,” *Psychological Bulletin*, vol. 76, no. 5, pp. 378–382, 1971.
27. C. G. Northcutt, L. Jiang, and I. L. Chuang, “Confident learning: Estimating uncertainty in dataset labels,” *Journal of Artificial Intelligence Research*, vol. 70, pp. 1373–1411, 2021.
28. C. G. Northcutt, A. Athalye, and J. Mueller, “Pervasive label errors in test sets destabilize machine learning benchmarks,” in *NeurIPS Datasets and Benchmarks Track*, 2021.
29. D. Hadfield-Menell, S. Milli, P. Abbeel, S. Russell, and A. Dragan, “Inverse reward design,” in *Advances in Neural Information Processing Systems (NeurIPS 2017)*, pp. 6765–6774, 2017.
30. P. F. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei, “Deep reinforcement learning from human preferences,” in *Advances in Neural Information Processing Systems (NeurIPS 2017)*, pp. 4299–4307, 2017.
31. International Organization for Standardization (ISO), “Road vehicles — safety of the intended functionality,” Tech. Rep. ISO 21448:2022, ISO, Jun 2022.
32. T. Everitt, M. Hutter, R. Kumar, and V. Krakovna, “Reward tampering problems and solutions in reinforcement learning: A causal influence diagram perspective,” *Synthese*, vol. 198, pp. 6435–6467, 2021.
33. M. Wagner and C. Carlan, “The open autonomy safety case framework,” *arXiv preprint arXiv:2404.05444*, 2024. Introduces OASCF — a roadmap for safely deploying autonomous vehicles (AVs). :contentReference[oaicite:3]index=3.
34. Automated Vehicle Safety Consortium (AVSC), “Best practice for core automated vehicle safety information,” Tech. Rep. AVSC-D-02-2024, SAE International — AVSC, May 2024. Defines a core set of safety topics for automated vehicle (AV) deployments.
35. National Transportation Safety Board, “Collision between vehicle controlled by developmental automated driving system and pedestrian, tempe, arizona, march 18, 2018,” Tech. Rep. NTSB/HAR-19/03, National Transportation Safety Board, Nov 2019. Highway Accident Report. Adopted November 19, 2019.
36. P. Koopman, “Anatomy of a robotaxi crash: Lessons from the cruise pedestrian dragging mishap,” in *Computer Safety, Reliability, and Security: 43rd International Conference, SAFECOMP 2024, Florence, Italy, September 18–20, 2024, Proceedings*, (Berlin, Heidelberg), p. 119–133, Springer-Verlag, 2024.
37. U.S. National Highway Traffic Safety Administration, “Additional information regarding ea22002 investigation: Crashes while using tesla autopilot,” Tech. Rep. INCR-EA22002-14496, NHTSA Office of Defects Investigation, Jan 2024.
38. U.S. Department of Defense, “Standard Practice for System Safety (MIL-STD-882E),” tech. rep., United States Department of Defense, May 2012. Available from ASSIST database or through various standards resellers.

Acknowledgments

The authors thank the ML FMEA Collaborative members for their invaluable insights and feedback throughout this proof of concept phase. We are grateful to the companies that supported this work: Reynolds & Moore, TORC Robotics, TÜV Rheinland Group, HORIBA MIRA Ltd, Saphira AI, APTIV, MassRobotics, and Edge Case. Special recognition goes to the development and safety teams who piloted these methods and provided candid feedback that shaped these refinements. Their commitment to advancing ML safety engineering has been instrumental in achieving this proof of concept milestone.

Contact Information

The authors welcome comments, feedback, and discussion related to this progressing work.

- Neil Wadhvana, nellywhads@gmail.com
- Bodo Seifert, Bodo.Seifert@us.tuv.com
- David Ward, david.ward@horiba-mira.com
- Akshay Chalana, akshay@saphira.ai
- Majed Mohammed, majedsamhan65@gmail.com
- Michael Wagner, mwagner@ecr.ai
- Simon Diemert, simon.diemert@cslabs.com
- Fahim Mannan, fmannan@gmail.com
- Jerry Lopez, Jerry.Lopez@torc.ai
- Krzysztof Pennar, Krzysztof.Pennar@torc.ai
- Chaitanya Shinde, shinde.chaitanya93@gmail.com
- Justin Poh, justin@justinpoh.com
- Paul Schmitt, paul.schmitt@reynolds-moore.com