Diversity Meets Relevancy: Multi-Agent Knowledge Probing for Industry 4.0 Applications

Christodoulos Constantinides[*1], **Dhaval Patel**[*2],
Scott Kimbleton[1],
Nishu Garg[1], **Muhammad Paracha**[1]
[1]IBM
[2] IBM Research
{christodoulos.constantinides@, pateldha@us., kimbleton@us., nishu.garg@, muhammadp@}ibm.com

Abstract

Industrial data scientists require deep domain understanding to model asset conditions effectively, yet traditional sources such as Subject Matter Experts (SMEs) and Failure Modes and Effects Analysis (FMEA) documents are often unavailable or incomplete. We present a deployed Multi-Agent System (MAS) that leverages Large Language Models (LLMs) to automatically generate and refine domain-relevant questions, improving modeling decisions across industrial projects. The system addresses two key challenges—ensuring linguistic diversity and maintaining high relevance—by combining established information diversity metrics with a grounded relevancy classifier. We evaluate its effectiveness through diversity benchmarks, compare against direct prompting and AutoAgents baselines, knowledge coverage on downstream FMEA tasks, and controlled user studies. Deployed in real-world projects, the MAS has improved multiple stages of the CRISP-DM methodology, resulting in measurable savings in cost and man-hours.

Extended Version and Code —
https://github.com/chrisconstant/Auto-Q

Introduction

As recommended by established frameworks such as CRISP-DM (Shearer 2000) and Microsoft's Team Data Science Process (TDSP) (Microsoft 2024), Industrial Data Science projects for physical asset health modeling begin by building deep domain understanding through close collaboration with Subject Matter Experts (SMEs). For critical assets such as power generators, this process often involves structured questioning about operational behavior, failure modes, component interactions, and sensor data characteristics. Insights from SMEs guide the formulation of hypotheses, refinement of analysis strategies, and generation of targeted follow-up questions, progressively enhancing both domain knowledge and model performance.

However, this process is time- and resource-intensive due to its sequential nature and because SMEs are typically focused on hands-on asset maintenance, with modeling not being their primary responsibility. Prior work (Mao et al. 2019) examined similar challenges in scientific collaborations between data scientists and domain experts (biomedical scientists), identifying issues in collaboration readiness, technology readiness, and coupling of work, as well as tensions in building common ground that influence both process and outcomes. With recent advancements in Large Language Models (LLMs), a new possibility emerges: can LLMs trained on extensive literature and technical documentation serve as a proxy domain expert, enabling more efficient hypothesis generation and exploration in Industrial Data Science workflows? This aligns with the emerging field of knowledge probing, which investigates how to systematically elicit domain-specific knowledge from LLMs through targeted question generation and answering.

Knowledge probing has been applied in multiple domains (Nori et al. 2023; Bulathwela, Muse, and Yilmaz 2023; Drori, Zhang, and Shuttleworth 2023), exam question generation (Drori, Zhang, and Shuttleworth 2023; Bulathwela, Muse, and Yilmaz 2023). Other work focuses on improving factuality and reliability (Youssef et al. 2023; Wang et al. 2023b; Yin et al. 2023), sometimes leveraging external knowledge graphs (Zheng et al. 2023). However, existing methods often lack mechanisms for systematically diversifying questions or tailoring them to the complex interdependencies found in industrial systems.

In this work, we present `Auto-Q`, a novel LLM-based Multi-Agent System (MAS) designed for diverse knowledge probing in industrial domains. Our approach leverages LLM planning and tool-calling capabilities to iteratively generate domain-specific question–answer pairs through a multi-agent interaction framework. The system integrates a mixture of zero-shot prompting, in-context learning, and progressive refinement techniques—where follow-up questions are systematically derived from prior outputs—to achieve greater breadth and depth of domain coverage. Experimental results demonstrate that `Auto-Q` consistently improves question diversity compared to baseline single-agent LLM methods, offering a scalable and adaptive solution for industrial knowledge elicitation.

[*]These authors contributed equally.

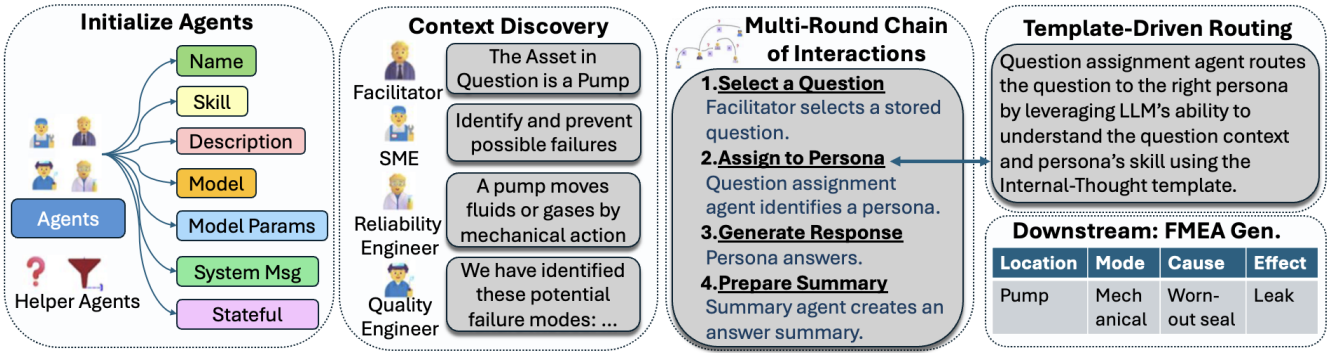

Figure 1: Auto-Q System Diagram. Multiple Personas are spinned up with different skills & roles. The system produces candidate questions, which are then diverted to the right persona to answer using a classifier. Questions are classified as useful/non-useful and filtered out.

Related Work

Knowledge Probing

LLM knowledge probing spans domains including medicine, education, and machine learning assessments. In education, Nori et al. (Drori, Zhang, and Shuttleworth 2023) generate new ML exam questions from course materials, while Bulathwela et al. (Bulathwela, Muse, and Yilmaz 2023) study how scientific pre-training and educational fine-tuning affect question quality. Other work focuses on improving reliability and factuality (Youssef et al. 2023; Wang et al. 2023b; Yin et al. 2023), sometimes using external structures like knowledge graphs (Zheng et al. 2023). These inform our adaptation of multi-agent LLM frameworks for domain-specific knowledge elicitation in industrial systems.

Multi-Agent Systems with LLMs

Multi-agent LLM systems comprise role-specific agents, communication protocols, and coordination mechanisms to achieve complex goals. Applications include medical decision-making (MDAgents (Kim et al. 2024)), scientific hypothesis generation (InternAgent (Team et al. 2025)), and financial analysis (FINCON (Yu et al. 2024)). Despite progress, generating diverse, domain-specific knowledge to guide Industrial Data Scientists remains underexplored. Our work addresses this by automatically instantiating agents tailored to industrial tasks, simulating relevant work environments, and adapting communication protocols across asset domains.

Problem Statement

Our goal is to automatically generate a diverse set of domain-specific questions Q for industrial asset maintenance. We formalize this as an optimization problem:

$$\underset{q}{\text{argmax}} \sum_{q \in Q} f(q), \quad s.t. \sum_{q \in Q} u(q) = |Q|$$

where $f(\cdot)$ is a diversity metric such as BLEU (Papineni et al. 2002) or Type-Token Ratio (TTR) and $u(\cdot)$ is a domain-specific question usefulness classifier that outputs a binary decision $u(q) \in \{0, 1\}$. In other words, we seek to maximize diversity while ensuring that all generated questions are considered useful.

System Design

We detail the process of automatic question generation, termed `Auto-Q`, which follows a recipe for generating tasks using an off-the-shelf pre-trained general language model in an iterative bootstrapping manner. Figure 1 illustrates the content generation workflow within our LLM-based multi-agent framework. The system is composed of multiple agent personas, each representing different user behaviors in response to an incoming instruction (see Table 1).

The process begins with an *initialization phase* in which `Auto-Q` collects user input (Section 1 User Input Request) and instantiates multiple agents, each equipped with a role-specific system prompt (Section 2 Defining Agent and Instruction Data). This is followed by a warm-up interaction round involving three key agents—*DSAgent*, *SMEAgent*, and *SummaryAgent*—to produce an initial catalog of candidate questions (Sections 3 Zero-shot Instruction Generation and 4 In-Context Instruction Generation). To ensure clarity, we structure the discussion around three distinct agents.

Once this initial set is established, the system transitions to the *iterative bootstrapping phase*. In each round, three core sub-tasks are performed: (i) question assignment via a classifier, (ii) answer generation, and (iii) new question generation (Sections 5 Iterative Question Assignment, 6 Zero-shot Answer Generation, and 7 In-Context Question Generation). Between iterations, a quality control step (Section 8 Question Filtering and Quality Control) ensures that only refined and relevant questions progress to the next round.

1 User Input Request

The first step captures user requirements through a structured template. As illustrated in Appendix Prompt 9, the template records details such as the industrial asset or equipment name (and its component, if applicable), the intended AI model type (e.g., anomaly detection, health forecasting,

event prediction), and the source of data (e.g., OT, IoT, PLC, maintenance logs) for model development.

2 Defining Agent and Instruction Data

Our system is built upon two core components: *Agents* and *Instructions*. An **agent** is represented as a triplet $\xi = \langle Name, Role, Skill \rangle$, where the *Role* is expressed through a system prompt and the *Skill* is a concise description of the agent's capabilities. For LLM-based agents, each agent is backed by a Pre-trained Language Model (PLM), denoted as $\mathbb{M}_\xi$ for agent ξ. **Instructions** define the specific tasks or goals assigned to agents, and together with the agent definition, they form the basis for executing domain-specific workflows.

Instruction refers to the content generated by agents and circulated among them. Each instruction is denoted by lowercase letters $x, y, z, s, \ldots$, and is categorized into one of three types: *question*, *answer*, or *document*. We use the notation x.type to indicate the type of an instruction x. The inference output of a model $\mathbb{M}$, which is itself an instruction y, generated from an input instruction x using agent ξ's language model is represented as:

$$y \leftarrow \mathbb{M}_\xi^{\text{prompt}}(x) \tag{1}$$

Here, prompt can correspond to different prompting strategies such as direct prompting, chain-of-thought (CoT) (Kojima et al. 2022; Zhang et al. 2023), or tree-of-thought (ToT) (Yao et al. 2023).

In Step 2, the `Auto-Q` framework dynamically spins up five specialized agents, each with distinct capabilities. Table 1 outlines these agents along with the types of instructions they can consume and produce. Not all agents are able to handle every type of instruction; their roles are intentionally differentiated. A *System Prompt* acts as a model-agnostic background document, carefully crafted to define an agent's role and responsibilities. This System Prompt serves as a persistent global context—always included in the information exchanged between agent ξ and its corresponding model $\mathbb{M}_\xi$.

Agents	Instruction Generates	Instruction Consumes	System Prompt
Data Scientist	●, ■	●, 🗋	Prompt A
SME	■	●	Prompt B
Q Assignment	■	●	Prompt C
Q Generator	●	●, ■	Prompt D
Summary	🗋	■	Prompt E

Table 1: Agent and Instructions: Icons are used to represent different instruction types: question (●), answer (■) and document(🗋)

3 Zero-shot Instruction Generation

In this step, we leverage the zero-shot reasoning capabilities of LLMs (He, Xie, and Jha 2023; Kojima et al. 2022) in combination with agent-specific system prompts. Both the *DSAgent* and *SMEAgent* are prompted using only the system prompt and the user input, without any additional contextual information. The user input is transformed into an informal instruction, termed the *system initialization prompt*, as shown in Appendix Prompt 10. The instructions are then processed as follows:

$$\begin{aligned} y_1 &\leftarrow \mathbb{M}_{DSAgent}^{\text{prompt}}(\text{Prompt 10}), \\ y_2 &\leftarrow \mathbb{M}_{SMEAgent}^{\text{prompt}}(\text{Prompt 10}). \end{aligned} \tag{2}$$

The output of the LLM varies depending on the agent type. For the *DSAgent*, y_1 is a list of initial questions, typically around ten. For the *SMEAgent*, y_2 consists of a background document containing information about the asset's components, failure modes, and failure reasons. These outputs provide the foundation for subsequent iterative question generation and refinement.

4 In-Context Instruction Generation

The background document produced by the *SMEAgent* in the previous step (y_2) is typically detailed and structured, averaging around 500 tokens. To facilitate further question generation, the *SummaryAgent* first condenses this document into a concise and readable format, which is then provided to the *DSAgent* as in-context information. This step involves two sequential LLM inferences:

$$y_3 \leftarrow \mathbb{M}_{SummaryAgent}^{\text{prompt}}(y_2), \quad y_4 \leftarrow \mathbb{M}_{DSAgent}^{\text{prompt}}(y_3). \tag{3}$$

The output y_4 constitutes a new set of questions, which we denote as $\mathbb{Q}_1$ in the question catalog. Compared to approaches such as Self-Instruct (Wang et al. 2022) and Self-Align (Sun et al. 2023b,a), the combination of zero-shot instruction generation (Section) and in-context summarization enables `Auto-Q` to automatically generate a robust initial set of seed questions. This generate-and-read procedure enhances both the quality and relevance of the questions and answers. In the next stage, we describe how this initial set $\mathbb{Q}_1$ is used to iteratively generate subsequent question sets $\mathbb{Q}_{i+1}$ based on the previous round $\mathbb{Q}_i$.

5 Iterative Question Assignment

Given a set of questions $\mathbb{Q}_i$, the first step in the iterative process is to generate corresponding answers, denoted $\mathbb{A}_i$. Despite well-defined context and agent roles, LLM agents may still generate questions that are out-of-context or resemble internal dialogue, where the agent could answer them independently. For example, the *DSAgent* might formulate questions that it can answer without external input. Identifying the appropriate agent to respond to each question therefore presents a genuine challenge, which is addressed by the *QAssignment* agent.

We frame question assignment as a relevancy classification problem, solved using LLMs. To implement the *QAssignment* agent in a new domain, we prepare the system prompt based on the problem specification (Appendix Prompt C) and provide examples of questions and the right agent assignment (Appendix Prompt 12).

For illustration, we selected a single question q from $\mathbb{Q}_1$ to prepare Appendix Prompt 12. The LLM generates a concise, persona-centric opinion on the suitability of each agent to answer q, linking these judgments to the relevancy classification. Once Appendix Prompts 11 and 12 are defined, the functioning of *QAssignment* is captured as:

$$y_q \leftarrow \mathbb{M}^{\text{prompt}}_{QAssignment}(\text{Prompt3}, \text{Prompt4}, q) \quad (4)$$

This approach differs from AutoGen-style systems (Wu et al. 2024), where the LLM routes questions to agents without explicit classification. Our method allows multi-class assignment, reflecting the possibility that a question may be answered by multiple agents.

6 Zero-shot Answer Generation

Following the question assignment step, the *SMEAgent* and *DSAgent* receive their respective sets of questions and are tasked with generating the corresponding answers, denoted $\mathbb{A}_i$. Ideally, questions are presented to the LLM in a logical order—such as from easy to hard, short to long, or concept to event—to facilitate coherent answer generation.

Sequential inference, however, can be slow, particularly when producing longer documents or detailed answers. Additionally, feeding a long history into the input may cause truncation due to the context length limitations of LLMs. To mitigate these issues, we leverage zero-shot reasoning in combination with tailored system and helper prompts, enabling efficient generation of answers without requiring extensive in-context examples. This approach balances the quality and completeness of answers while reducing latency and avoiding output truncation.

7 In-Context Question Generation

Given a set of questions $\mathbb{Q}_i$ and their corresponding answers $\mathbb{A}_i$, the *QGenAgent* generates additional questions using two complementary approaches. In the first approach, a random subset of 30 questions from $\mathbb{Q}_i$ is selected as in-context examples to guide the generation of new questions. This process is repeated five times to produce a diverse set of questions:

$$y_{qseq} \leftarrow \mathbb{M}^{\text{prompt}}_{QGenAgent}(\ldots, q_i, q_j, q_k) \quad (5)$$

The second approach leverages the concept of follow-up questions, generating new questions based on the most recently produced answers. To implement this, each newly generated answer is first summarized by the *SummaryAgent* into a_i^*, which is then used by the *QGenAgent* to formulate the next question. This creates a dynamic and interactive question–answer process:

$$y_{qa} \leftarrow \mathbb{M}^{\text{prompt}}_{QGenAgent}(\ldots, q_i, a_i^*) \quad (6)$$

Together, these approaches allow `Auto-Q` to iteratively expand the question catalog, improving both coverage and diversity while maintaining relevance to the industrial asset domain.

8 Question Filtering and Quality Control

At this stage, the system has generated a new set of instructions, $\mathbb{Q}_{i+1}$. To ensure the quality and diversity of the generated questions (Holtzman et al. 2020), we apply a combination of metrics and empirically derived thresholds to filter out redundant or non-useful items. One such metric is the Type-Token-Ratio (TTR), with a threshold set at 2.5. Each instruction is first screened independently, and those failing the chosen quality metrics are removed.

Next, a pre-trained classifier labels each question as either useful or non-useful, with non-useful questions discarded. To further reduce redundancy, the remaining questions are compared to previously generated questions using the BLEU score (Papineni et al. 2002). Any instruction with a BLEU score exceeding the threshold relative to earlier questions is eliminated. Finally, the instructions are sorted in decreasing order of length, and each instruction at position i is removed if its BLEU score surpasses the threshold when compared with any instruction prior to i in the sorted order. This multi-step process ensures that the final set of questions is both diverse and relevant for downstream tasks.

Experiment

In this section, we evaluate the effectiveness of the proposed `Auto-Q` system. Our experiments are designed to address the following research questions:

- **RQ1:** Does the proposed method outperform baseline algorithms in terms of question diversity while maintaining high usefulness?
- **RQ2:** How comprehensive is the knowledge coverage of the generated questions and answers in supporting downstream industrial tasks, such as the generation of a Failure Modes and Effects Analysis (FMEA) document?

Experimental Setup

To validate our approach, we generate question–answer pairs using `Auto-Q` and baseline methods, which are then summarized by an LLM to produce an FMEA document. We assess multiple aspects of the generated questions, including diversity, usefulness, and downstream task coverage.

Our experiments utilize two open-source foundation models: LLaMA (Touvron et al. 2023) and Mistral (Sanseviero et al. 2023), with a focus on two industrial asset classes: Wind Turbines and Air Compressors. Specifically, we employ `Llama-3.1-70b-instruct` and `Mistral-large-instruct-2407`, accessed via our internal platform.

We evaluate the system across several dimensions: token-level diversity (using metrics such as BLEU, TTR, and perplexity), word-level diversity (e.g., bi-gram analysis to capture question types), semantic-level diversity (using embedding-based similarity models such as BERT), question usefulness classification (based on grounded domain information), and performance in the downstream FMEA task. This multi-level evaluation allows us to quantify both the quality and practical relevance of the generated questions and answers.

Token-Level Diversity Evaluation Metrics

To assess the linguistic quality and variety of the generated questions, we employ four commonly used token-level metrics. First, we compute the **Normalized Type-Token Ratio (TTR)**, defined as the ratio of the number of unique tokens to the square root of the total token count. Second, we measure **Distinct-n**, which calculates the proportion of unique n-grams relative to the total number of n-grams; we consider $n = 2$ and $n = 3$ to obtain Distinct-2 and Distinct-3, respectively. Third, **Self-BLEU** quantifies similarity between a single sentence and the rest in a group of generated questions, treating one sentence as the hypothesis and the others as references, with the final score being the average across all sentences. Lastly, we compute **Perplexity**, reflecting the model's estimated conditional probability of tokens given preceding context. Here, we report log-perplexity and average over individual sequences to provide a representative measure for the dataset.

Question-Type Diversity Evaluation Metrics

To evaluate the conceptual depth of generated questions, we analyze bigram frequencies using the first two words of each question. The top 10 bigrams are compared to reference datasets such as SQuAD 2.0 and HotpotQA. Prior work (Craig et al. 2000) indicates that questions starting with "Why" or "How" are indicative of deeper reasoning and higher question complexity.

Semantic Diversity Eval Metrics via Embeddings

For semantic-level assessment, we convert each question into a dense vector representation using pre-trained sentence embedding models (Reimers and Gurevych 2019). We experiment with two models: `all-mpnet-base-v2`, which performs strongly on general semantic benchmarks, and `multi-qa-mpnet-base-dot-v1`, trained specifically on question-answer data. Semantic diversity is quantified by the dispersion of embeddings in the semantic space. We compute dispersion in two ways: the average pairwise cosine distance among all embeddings ($PDist$) and the mean cosine distance from each embedding to the centroid of the set ($CDist$). These metrics provide complementary insights into the breadth of semantic coverage in the generated questions.

Question Usefulness Evaluation

To ensure generated questions are relevant to the target asset class, we employ a ReAct-based relevancy assessment. For each question, the system searches trusted technical and scientific sources—including Wikipedia, arXiv, DuckDuckGo, Wikidata, and Semantic Scholar—and retrieves supporting evidence. Retrieved passages are classified as supporting or contradicting, and a relevancy score from 0.0 (irrelevant) to 1.0 (highly relevant) is assigned based on the evidence. This automated, evidence-driven evaluation ensures that generated questions remain focused, domain-appropriate, and useful for downstream analyses. We evaluated the relevancy of generated questions for a hydroelectric power turbine asset class using 200 selected examples. The distribution of relevancy scores shows that most questions are highly relevant, with 137 examples scoring 0.9 and 50 examples scoring 0.8. Only a small number of questions were less relevant, with 14 examples at 0.7, 10 at 1.0, and a single example at 0.1, indicating that the ReAct-based relevancy-check effectively filters out unrelated questions.

Downstream Application Evaluation

Failure Modes and Effects Analysis (FMEA) is a critical component of reliability-centered maintenance. To evaluate `Auto-Q` in a practical setting, we extend the system with additional domain-specific personas, including reliability engineers, mechanical engineers, and quality control specialists, each with distinct skill sets. Using `Auto-Q`, we generate questions and corresponding answers through closed-book LLM reasoning. The resulting answers are then used to produce FMEA tables for 10 different industrial assets, which are compared against ground truth labels to assess accuracy and coverage.

Baseline: Direct LLM Prompting

A straightforward alternative is for a data scientist to prompt an LLM directly, e.g., "Generate 100 questions for building an anomaly detection model for wind turbine gearboxes." While this approach can quickly produce a large set of questions, it does not guarantee high-quality or domain-relevant outputs. To benchmark this baseline, we conduct eight trials with variations in system prompts, measuring linguistic diversity using `Distinct-2` (Li et al. 2015) over batches of 20 questions.

Baseline: AutoAgents

AutoAgents (Chen et al. 2023) automatically generates agents with defined roles, plans task allocation, and executes these plans to produce outputs. We compare our system against AutoAgents by evaluating the linguistic diversity of generated questions using `Distinct-2`, providing a controlled comparison of structured multi-agent versus direct multi-agent generation approaches.

Baseline: Human-Generated Q&A Datasets

To contextualize the quality of generated questions, we compare metrics against human-generated datasets, including SQuAD 2.0 (Rajpurkar et al. 2016) and HotpotQA (Yang et al. 2018). Following prior recommendations (Bulathwela, Muse, and Yilmaz 2023), machine-generated questions are considered effective if they achieve comparable or superior linguistic quality metrics relative to these datasets. Our approach produces questions with a broad vocabulary (high TTR), domain specificity (high Self-BLEU), and coherent, fluent text (low perplexity), indicating that `Auto-Q` reliably generates high-quality, domain-focused questions.

Results

Token-Level Diversity

Figure 2 summarizes the results of our trial experiments. We observed that as more questions are generated in the same prompt, linguistic quality metrics deteriorate, with repeated

questions, style/template duplication, and occasional LLM inconsistencies. These observations align with recent findings in LLM prompting literature (He, Xie, and Jha 2023; Zhang et al. 2023; Yao et al. 2023; Wang et al. 2023a), highlighting the need for repeated interrogation of LLMs from multiple viewpoints to achieve better diversity and coverage.

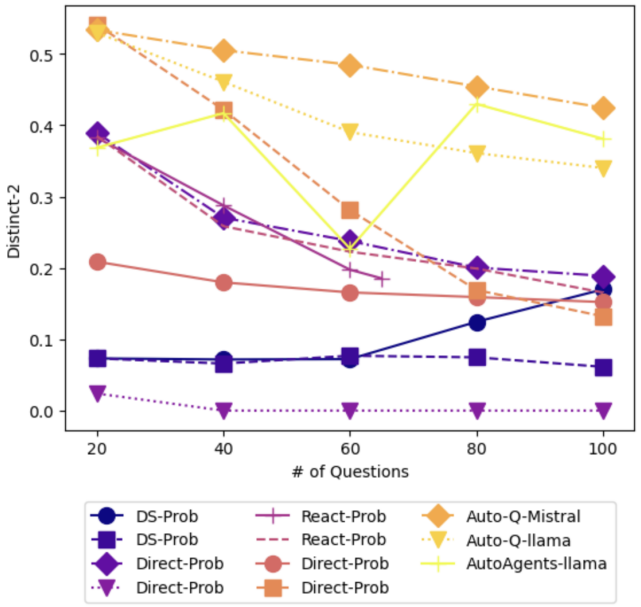

Figure 2: Linguistic diversity of generated questions across different trials and settings: direct prompting vs AutoGen vs ReAct, generic vs data scientist personas, Llama-3.1-70b-instruct vs Mistral-large-instruct-2407.

Metric	D_L^W	D_M^W	SQuAD	Hotpot
TTR (↑)	**4.59**	**4.52**	3.03	3.80
Distinct-2 (↑)				
All	0.10	0.12	0.25	0.29
< Round 2	0.24	**0.28**		
Distinct-3 (↑)				
All	0.24	**0.29**	0.54	0.57
< Round 2	0.42	0.48		
Self-BLEU (↑)	**0.79**	**0.78**	0.11	0.23
Perplexity (↓)	**40.7**	**72.9**	301	273

Table 2: Comparison of linguistic metrics using human-generated questions. D_L^W and D_M^W denote questions generated for Wind Turbines using Llama and Mistral, resp.

Question-Type Diversity

Appendix Table 3 summarizes the leading bigrams in generated questions, highlighting the prevalence of deeper reasoning questions.

Semantic-Level Diversity

We evaluate semantic diversity using sentence embeddings (Reimers and Gurevych 2019), comparing both base-v2 (general-purpose) and base-dot-v1 (Q&A-tuned) models. We quantify diversity with the pairwise cosine distance ($PDist$) and centroid-based distance ($CDist$). Appendix Table 4 shows the results.

The results indicate that $PDist$ is higher but $CDist$ is lower for our generated questions compared to benchmarks. This suggests that embeddings are widely spread across the semantic space but less clustered around a central point, indicating coverage of diverse semantic concepts.

Downstream Application Performance

Appendix table 5 presents the FMEA benchmark results. The consistently high recall for several assets suggests that the LLMs have a reasonable grasp of domain-specific failure modes. However, precision remains low, which may stem from two primary causes: 1) **Terminology mismatch** — the model predicts correct failure modes but uses alternative wording that does not match the ground truth, leading to false negatives in exact-match evaluation. This could be addressed through synonym-aware matching or manual review. 2) **Irrelevant predictions** — the model generates failure modes unrelated to the asset, indicating possible knowledge gaps. In such cases, domain-adapted LLMs may offer improvements. We leave a detailed error analysis and exploration of precision-improvement strategies as future work.

Discussion

- **RQ1: Question diversity.** Our method improves diversity at the token (Table 2), question-type (Table 3), and semantic (Table 4) levels. This holds both against naive prompting (directly requesting 100 questions) and against human-authored benchmark datasets (SQuAD 2.0, HotpotQA). Notably, diversity gains do not appear to compromise question usefulness (see Section X for the usefulness evaluation).
- **RQ2: Downstream FMEA performance.** On the FMEA benchmark, our generated knowledge yields high recall but low precision across assets (Table 5). The high recall indicates substantial latent domain knowledge in general-purpose LLMs, while low precision may result from strict string matching or genuine knowledge gaps. Addressing the first cause may require semantic evaluation methods; addressing the second could involve domain-specific fine-tuning. Both avenues are left for future exploration.

Auto-Q Expert Evaluation

We deployed Auto-Q internally to gather qualitative feedback from domain experts. This section summarizes insights from end users as well as downstream application evaluation.

Expert Evaluation Setup

We conducted two rounds of anonymous human evaluations with 15 participants actively involved in building AI models

for industrial assets. Participants evaluated generated questions across multiple aspects including routing accuracy and overall usefulness.

Question Assignment Accuracy

Routing questions to the appropriate persona is a key capability of Auto-Q. To assess this, each annotator was presented with 20 randomly selected questions and asked to choose the preferred answer from four options:

1. Data Scientist option is significantly better,
2. SME option is significantly better,
3. Both personas should be engaged,
4. Question is irrelevant for the given context.

We computed inter-annotator agreement following the methodology in (Zhou and etc 2023), using tie-discounted accuracy on 50 randomly selected question pairs. The mean agreement score was 72% (SD = 4.08%) across 30 trials, indicating reasonably high consensus.

Appendix Figure 7 shows the distribution of annotator selections for four example questions: the top row shows complete alignment with `QAssignment`, while the bottom row highlights disagreements. Overall, in 80% of cases, questions were correctly routed without additional training, demonstrating the effectiveness of our assignment strategy.

Task Accomplishment

We evaluated the usefulness of the generated questions for supporting data scientists in anomaly detection model development. Annotators reviewed questions grouped by topic and provided ratings on four measures:

1. **Humanness:** Questions should resemble those authored by a real person.
2. **Topic Coverage:** A wide range of topics should be included, ideally covering the top 5–10 relevant areas.
3. **Engagement:** Questions should motivate the subject matter expert during the information discovery process.
4. **Novelty:** The system should help identify previously unconsidered topics or questions.

Each measure was scored on a 1–3 scale (1 = poor, 2 = moderate, 3 = good), providing a structured assessment of question quality and relevance.

Annotators rated the machine-generated questions highly across all qualitative measures. Specifically, the average **Humanness** score was 2.6 (SD = 0.48), and **Topic Coverage** was 2.6 (SD = 0.66). Engagement also scored highly at 2.5 (SD = 0.67), while **Novelty** was slightly lower at 2.2 (SD = 0.74). These results demonstrate the effectiveness of Auto-Q in producing domain-specific, human-like questions that cover relevant topics and stimulate engagement.

Feedback Distribution on Sample Questions

Out of the 20 evaluated questions, two required special attention (Figure 8 in Appendix). In these cases, the persona selected by `QAssignment` differed from the majority vote of annotators. For each question, we collected 15 votes and identified any persona receiving consistently low votes that had been selected by `QAssignment`.

For instance, in the first question, annotators predominantly chose the Data Scientist persona because of the presence of keywords such as "developing and interpreting the algorithm," which strongly suggested a data science context. This highlights opportunities to refine the `QAssignment` logic for better alignment with human judgment.

Deployment and Industrial Impact

We deployed our system in February 2024 to support Data Scientists during industrial maintenance modeling engagements. The deployment architecture is shown in Fig. 3, while Fig. 4 illustrates key milestones in the evolution of Auto-Q. Since deployment, the system has been integrated into day-to-day workflows, streamlining knowledge elicitation and improving decision support for asset modeling projects.

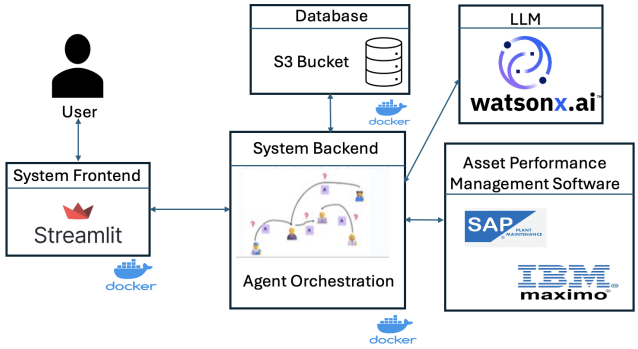

Figure 3: The frontend interacts with the backend to generate Q&A using Watsonx.ai-powered agents. Generated content stored in S3, sampled for quality control, and can be summarized into domain-specific documents for downstream Asset Performance Management systems (e.g., IBM Maximo, SAP). All components are packaged as Docker images.

The use case focused on modeling centrifugal pumps in the Oil & Gas industry. To quantify impact, we assess improvements through the lens of the CRISP-DM methodology. Specifically, we compare each stage of CRISP-DM in projects where Auto-Q was compared against a reference project of the same asset class without Auto-Q. This comparison highlights gains in efficiency, accuracy, and coverage, translating into measurable reductions in modeling effort, cost, and project cycle time.

Improvements on the CRISP-DM Methodology

We evaluated the impact of Auto-Q on a real-world industrial maintenance project involving centrifugal pumps in the Oil & Gas sector. We compared this project with a similar reference project without Auto-Q, quantifying improvements across the CRISP-DM lifecycle.

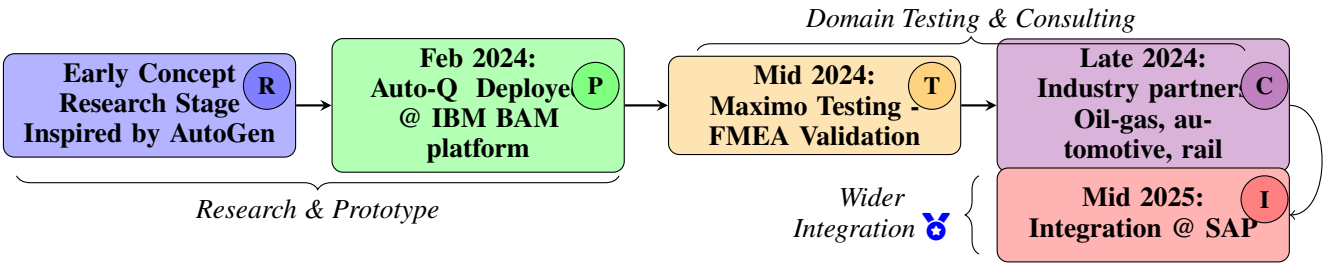

Figure 4: Timeline of Auto-Q development: R = Research, P = Prototype, T = Testing, C = Consulting, I = Integration.

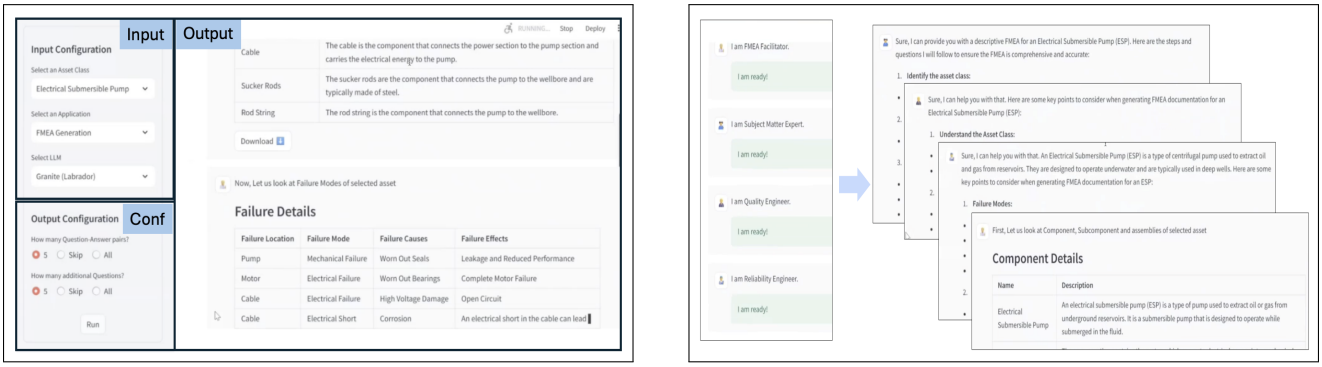

Figure 5: System interface components. Left: UI Layout, Right: Agents and background docs initialization.

Key Deployment Outcomes

- *65 hours saved* in Business Understanding and SME consultation.
- *45 hours saved* in Data Understanding meetings.
- *Improved feature engineering* with targeted sensor selection.
- *Model improvements:* TP $1 \rightarrow 3$, FP $5 \rightarrow 3$, FN $4 \rightarrow 2$.
- *Reduced process iterations:* $2 \rightarrow 1$.
- *Total estimated project savings:* \$85,000.

Business Understanding: Traditionally, this phase relies heavily on SMEs and stakeholders to prioritize assets and failure modes by Risk Priority Number (RPN). The Q&A generated by Auto-Q prepared Data Scientists to ask the right questions, reducing effort from *85 to 20 hours*. The system guided focus on seal failures, which are high-impact, causing rapid fluid release, environmental hazards, and equipment damage, while other failures, such as impeller wear, have less immediate consequences.

Data Understanding: Data Understanding often requires extensive SME collaboration. Auto-Q's downstream FMEA generation identified relevant sensors and variables for seal leakage, explaining the reasoning for selection. This reduced SME meeting hours from *120 to 75*.

Data Preparation: The system suggested targeted feature engineering, such as using the rolling standard deviation of vibration sensor readings, which was validated by SMEs as having significant predictive power. This reduced trial-and-error in feature selection.

Modeling & Evaluation: With properly prepared data, modeling with last year's data held out as test showed measurable improvements. True Positives increased from *1 to 3*, False Positives decreased from *5 to 3*, and False Negatives decreased from *4 to 2*, confirmed by SMEs. This demonstrates enhanced model sensitivity and operational relevance.

Deployment & Iteration: Post-deployment, monitoring showed fewer full process iterations were required. The number of iterations to reach an acceptable model decreased from *2 to 1*, accelerating time-to-value.

Estimated Cost Reduction: Table 8 shows estimated wages by role. Man-hour reductions for the first iteration are approximately \$56,550, with additional savings from avoiding a second iteration estimated at \$28,440.

Conclusion

We introduced Auto-Q, an LLM-powered multi-agent system that generates diverse, domain-specific questions to support industrial asset health modeling. Expert evaluations and real-world deployment demonstrate improvements in FMEA generation, SME engagement, and model development efficiency. These results highlight the potential of multi-agent LLMs to bridge domain knowledge gaps and accelerate AI workflows in Industry 4.0. In the future we will validate the real-world impact in industrial modeling projects of more assets, and integrate a feedback mechanism for addressing irrelevant questions and answers.

References

Bulathwela, S.; Muse, H.; and Yilmaz, E. 2023. Scalable Educational Question Generation with Pre-trained Language Models. arXiv:2305.07871.

Chen, G.; Dong, S.; Shu, Y.; Zhang, G.; Sesay, J.; Karlsson, B. F.; Fu, J.; and Shi, Y. 2023. Autoagents: A framework for automatic agent generation. *arXiv preprint arXiv:2309.17288*.

Craig, S.; Gholson, B.; Ventura, M.; and Graesser, A. 2000. Overhearing Dialogues and Monologues in Virtual Tutoring Sessions: Effects on Questioning and Vicarious Learning. 11.

Drori, I.; Zhang, S. J.; and Shuttleworth, e. 2023. From Human Days to Machine Seconds: Automatically Answering and Generating Machine Learning Final Exams. In *SIGKDD*, KDD '23, 3947–3955. New York, NY, USA. ISBN 9798400701030.

He, Z.; Xie, Z.; and Jha, e. 2023. Large Language Models as Zero-Shot Conversational Recommenders. In *CIKM*, CIKM '23. ACM.

Holtzman, A.; Buys, J.; Du, L.; Forbes, M.; and Choi, Y. 2020. The Curious Case of Neural Text Degeneration. In *International Conference on Learning Representations*.

Kim, Y.; Park, C.; Jeong, H.; Chan, Y. S.; Xu, X.; McDuff, D.; Lee, H.; Ghassemi, M.; Breazeal, C.; and Park, H. W. 2024. Mdagents: An adaptive collaboration of llms for medical decision-making. *Advances in Neural Information Processing Systems*, 37: 79410–79452.

Kojima, T.; Gu, S. S.; Reid, M.; Matsuo, Y.; and Iwasawa, Y. 2022. Large Language Models are Zero-Shot Reasoners. In Koyejo, S.; Mohamed, S.; Agarwal, A.; Belgrave, D.; Cho, K.; and Oh, A., eds., *Advances in Neural Information Processing Systems*, volume 35, 22199–22213. Curran Associates, Inc.

Li, J.; Galley, M.; Brockett, C.; Gao, J.; and Dolan, B. 2015. A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055*.

Mao, Y.; Wang, D.; Muller, M.; Varshney, K. R.; Baldini, I.; Dugan, C.; and Mojsilović, A. 2019. How data scientistswork together with domain experts in scientific collaborations: To find the right answer or to ask the right question? *Proceedings of the ACM on Human-Computer Interaction*, 3(GROUP): 1–23.

Microsoft. 2024. Team Data Science Process. https://learn.microsoft.com/en-us/azure/architecture/data-science-process/overview. Accessed: 2024-08-15.

Nori, H.; Lee, Y. T.; Zhang, S.; Carignan, D.; Edgar, R.; Fusi, N.; King, N.; Larson, J.; Li, Y.; Liu, W.; Luo, R.; McKinney, S. M.; Ness, R. O.; Poon, H.; Qin, T.; Usuyama, N.; White, C.; and Horvitz, E. 2023. Can Generalist Foundation Models Outcompete Special-Purpose Tuning? Case Study in Medicine. arXiv:2311.16452.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, 311–318.

Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. *arXiv e-prints*, arXiv:1606.05250.

Reimers, N.; and Gurevych, I. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Sanseviero, O.; Tunstall, L.; Schmid, P.; Mangrulkar, S.; Belkada, Y.; and Cuenca, P. 2023. Mixture of Experts Explained.

Shearer, C. 2000. The CRISP-DM model: the new blueprint for data mining. *Journal of data warehousing*, 5(4): 13–22.

Sun, Z.; Shen, Y.; Zhang, H.; Zhou, Q.; Chen, Z.; Cox, D.; Yang, Y.; and Gan, C. 2023a. SALMON: Self-Alignment with Principle-Following Reward Models. arXiv:2310.05910.

Sun, Z.; Shen, Y.; Zhou, Q.; Zhang, H.; Chen, Z.; Cox, D.; Yang, Y.; and Gan, C. 2023b. Principle-Driven Self-Alignment of Language Models from Scratch with Minimal Human Supervision. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Team, N.; Zhang, B.; Feng, S.; Yan, X.; Yuan, J.; Yu, Z.; He, X.; Huang, S.; Hou, S.; Nie, Z.; et al. 2025. NovelSeek: When Agent Becomes the Scientist–Building Closed-Loop System from Hypothesis to Verification. *arXiv preprint arXiv:2505.16938*.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; Rodriguez, A.; Joulin, A.; Grave, E.; and Lample, G. 2023. LLaMA: Open and Efficient Foundation Language Models. arXiv:2302.13971.

Wang, X.; Wei, J.; Schuurmans, D.; Le, Q. V.; Chi, E. H.; Narang, S.; Chowdhery, A.; and Zhou, D. 2023a. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *The Eleventh International Conference on Learning Representations*.

Wang, Y.; Kordi, Y.; Mishra, S.; Liu, A.; Smith, N. A.; Khashabi, D.; and Hajishirzi, H. 2022. Self-Instruct: Aligning Language Model with Self Generated Instructions.

Wang, Z.; Ye, L.; Wang, H.; Kwan, W.-C.; Ho, D.; and Wong, K.-F. 2023b. Readprompt: A readable prompting method for reliable knowledge probing. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 7468–7479.

Wu, Q.; Bansal, G.; Zhang, J.; Wu, Y.; Li, B.; Zhu, E.; Jiang, L.; Zhang, X.; Zhang, S.; Liu, J.; et al. 2024. Autogen: Enabling next-gen LLM applications via multi-agent conversations. In *First Conference on Language Modeling*.

Yang, Z.; Qi, P.; Zhang, S.; Bengio, Y.; Cohen, W. W.; Salakhutdinov, R.; and Manning, C. D. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. arXiv:1809.09600.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T. L.; Cao, Y.; and Narasimhan, K. 2023. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. arXiv:2305.10601.

Yin, Z.; Sun, Q.; Guo, Q.; Wu, J.; Qiu, X.; and Huang, X. 2023. Do large language models know what they don't know? *arXiv preprint arXiv:2305.18153*.

Youssef, P.; Koraş, O. A.; Li, M.; Schlötterer, J.; and Seifert, C. 2023. Give me the facts! a survey on factual knowledge probing in pre-trained language models. *arXiv preprint arXiv:2310.16570*.

Yu, Y.; Yao, Z.; Li, H.; Deng, Z.; Jiang, Y.; Cao, Y.; Chen, Z.; Suchow, J.; Cui, Z.; Liu, R.; et al. 2024. Fincon: A synthesized llm multi-agent system with conceptual verbal reinforcement for enhanced financial decision making. *Advances in Neural Information Processing Systems*, 37: 137010–137045.

Zhang, Z.; Zhang, A.; Li, M.; and Smola, A. 2023. Automatic Chain of Thought Prompting in Large Language Models. In *The Eleventh International Conference on Learning Representations*.

Zheng, S.; Bai, H.; Zhang, Y.; Su, Y.; Niu, X.; and Jaitly, N. 2023. Kglens: Towards efficient and effective knowledge probing of large language models with knowledge graphs. *arXiv preprint arXiv:2312.11539*.

Zhou, C.; and etc, P. L. 2023. LIMA: Less Is More for Alignment. arXiv:2305.11206.